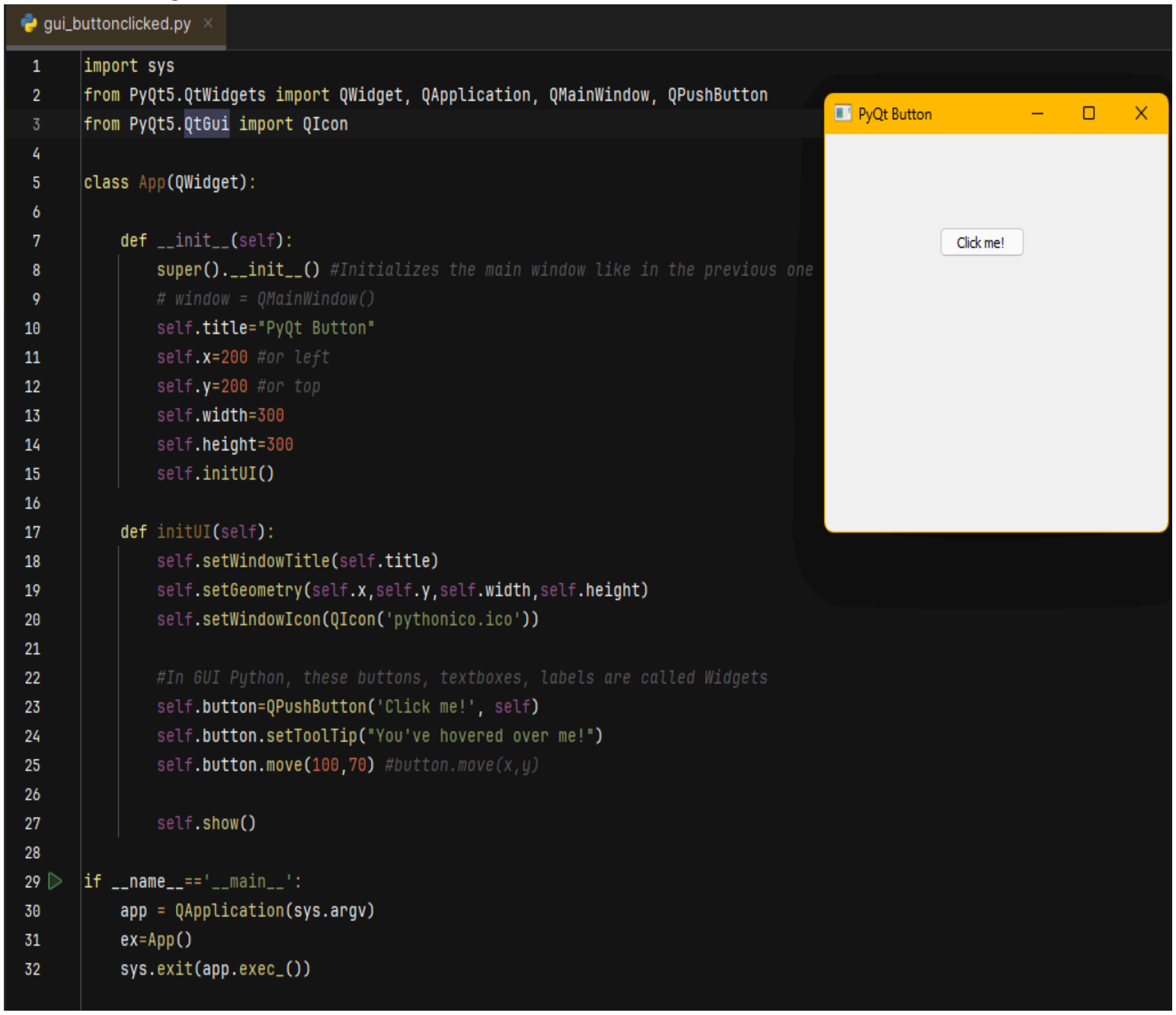
Activity #5
Introduction to Event Handling in GUI Development

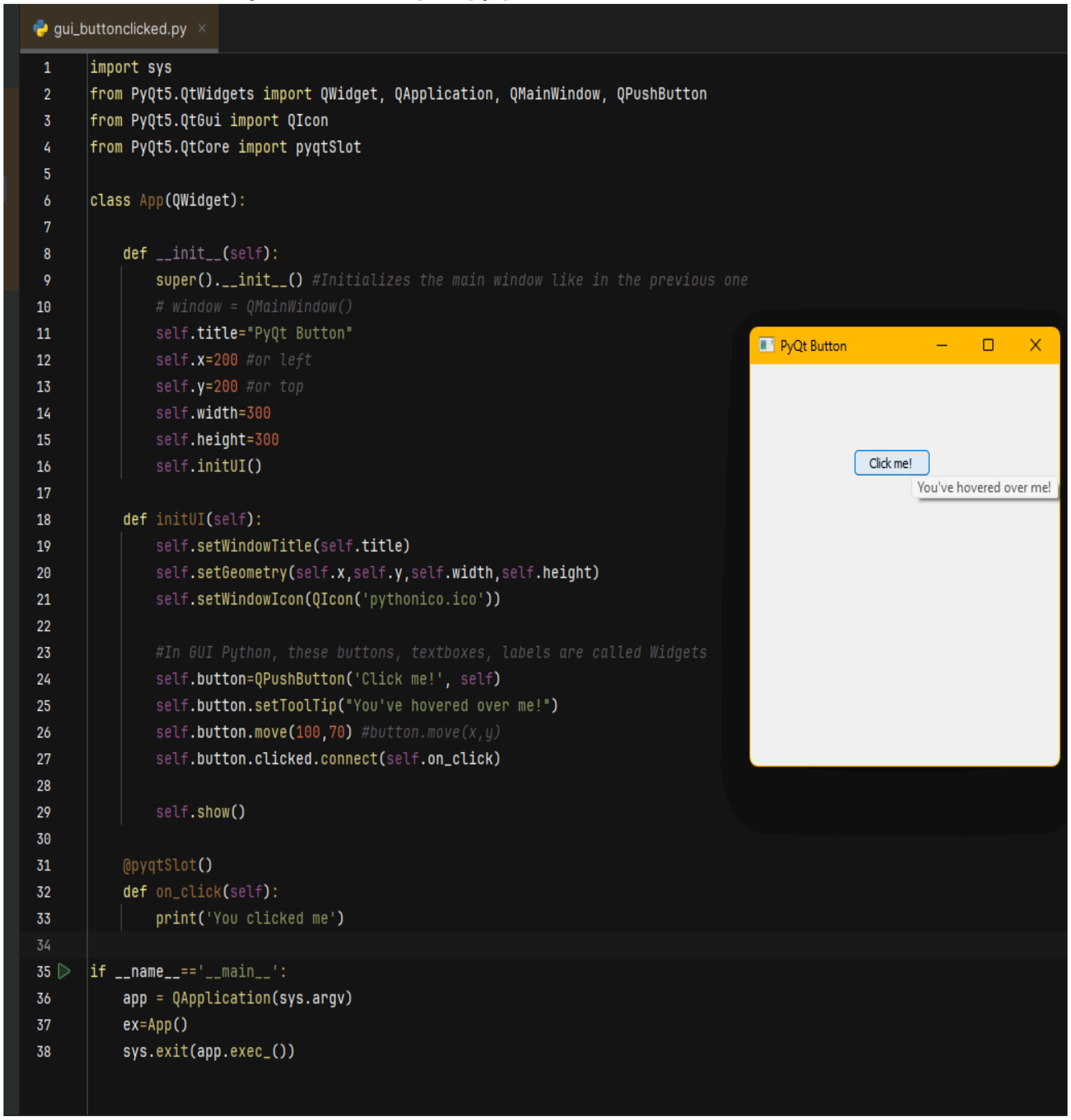
Magistrado, Aira Pauleen M.	20/21/24
CPE 009B-CPE21S4	Maria Rizette Sayo

Event Handling



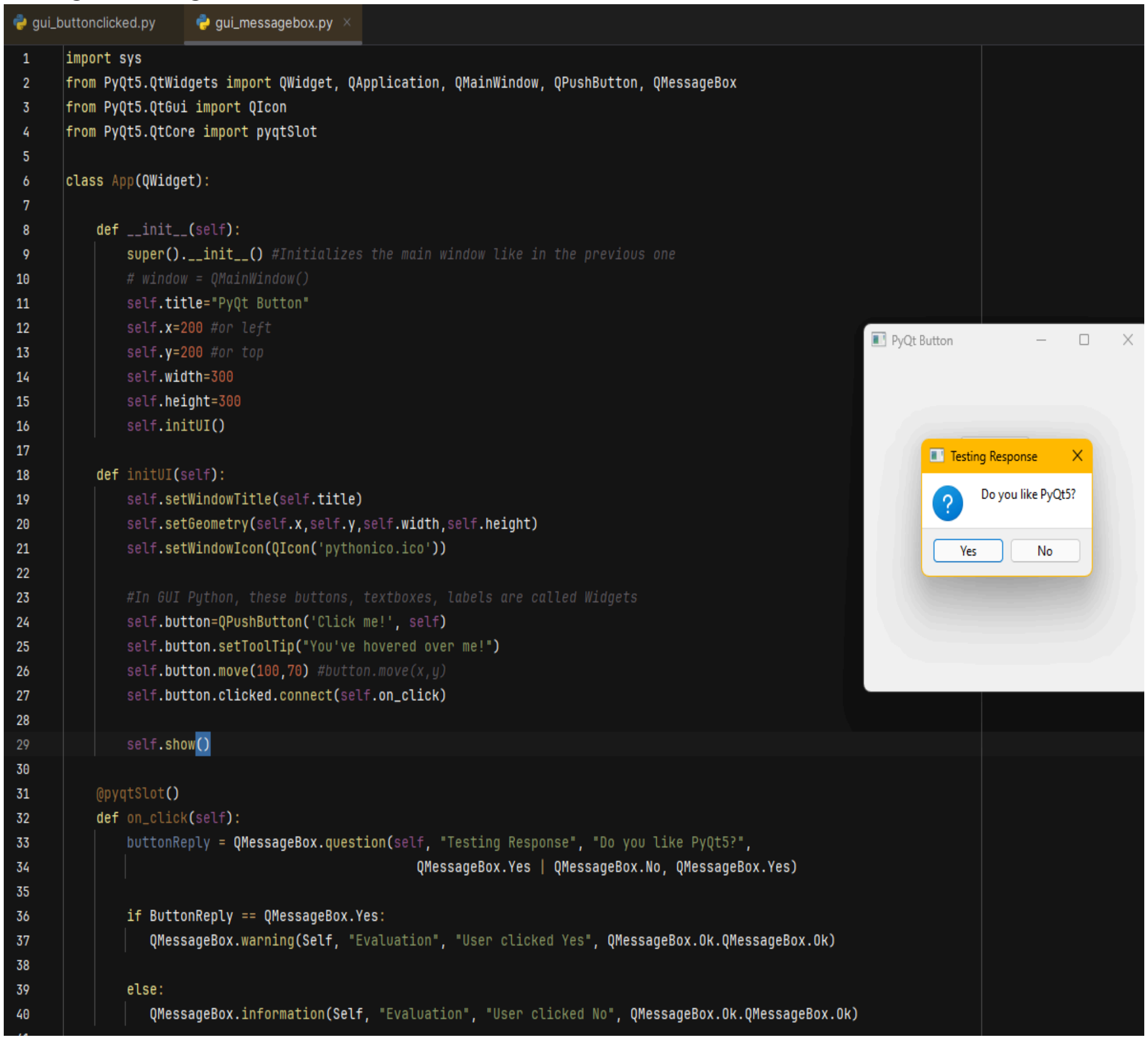
```python
import sys
from PyQt5.QtWidgets import QWidget, QApplication, QMainWindow, QPushButton
from PyQt5.QtGui import QIcon

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_())
```

Add Additional: from PyQt5.QtCore import pyqtSlot



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

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.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_())
```

Adding a message box:

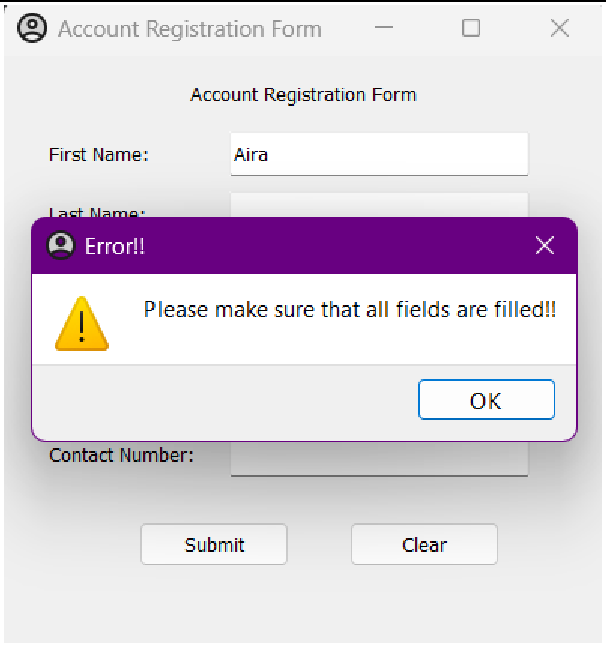


```python
import sys
from PyQt5.QtWidgets import QWidget, QApplication, QMainWindow, QPushButton, QMessageBox
from PyQt5.QtGui import QIcon
from PyQt5.QtCore import pyqtSlot

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.button.clicked.connect(self.on_click)

        self.show()

    @pyqtSlot()
    def on_click(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)
```

Supplementary Activity:

Python

```python
import sys
import csv
from PyQt5.QtWidgets import QMainWindow, QApplication, QLabel, QLineEdit,
QPushButton, QMessageBox
from PyQt5.QtGui import QIcon
from PyQt5.QtCore import Qt
from PyQt5.QtCore import pyqtSlot

class RegistrationApp(QMainWindow):

    def __init__(self):
        super().__init__()
        self.title = "Account Registration Form"
        self.width = 400
        self.height = 400
        self.initUI()

    def initUI(self):
        self.setWindowTitle(self.title)
        self.setGeometry(100, 100, self.width, self.height)
        self.setWindowIcon(QIcon('pythonico.ico'))
        self.center()

        #Program Title
        self.program_title = QLabel("Account Registration Form", self)
        self.program_title.setAlignment(Qt.AlignCenter)
        self.program_title.setGeometry(0, 10, self.width, 30)

        #Enter First Name
        self.label_first_name = QLabel("First Name:", self)
        self.label_first_name.move(30, 50)
        self.textbox_first_name = QLineEdit(self)
        self.textbox_first_name.move(150, 50)
        self.textbox_first_name.resize(200, 30)

        #Enter Last Name
        self.label_last_name = QLabel("Last Name:", self)
        self.label_last_name.move(30, 90)
        self.textbox_last_name = QLineEdit(self)
        self.textbox_last_name.move(150, 90)
        self.textbox_last_name.resize(200, 30)

        #Enter Username
        self.label_username = QLabel("Username:", self)
        self.label_username.move(30, 130)
        self.textbox_username = QLineEdit(self)
        self.textbox_username.move(150, 130)
        self.textbox_username.resize(200, 30)
        #Enter Password
        self.label_password = QLabel("Password:", self)
        self.label_password.move(30, 170)
        self.textbox_password = QLineEdit(self)
        self.textbox_password.move(150, 170)
        self.textbox_password.resize(200, 30)
        self.textbox_password.setEchoMode(QLineEdit.Password)

        #Enter Email
        self.label_email = QLabel("Email Address:", self)
        self.label_email.move(30, 210)
        self.textbox_email = QLineEdit(self)
        self.textbox_email.move(150, 210)
        self.textbox_email.resize(200, 30)

        #Enter Contact Number
        self.label_contact = QLabel("Contact Number:", self)
        self.label_contact.move(30, 250)
        self.textbox_contact = QLineEdit(self)
        self.textbox_contact.move(150, 250)
        self.textbox_contact.resize(200, 30)

        #Submit Entry Button
        self.submit_button = QPushButton('Submit', self)
        self.submit_button.setToolTip("Click to submit")
        self.submit_button.move(90, 310)
        self.submit_button.clicked.connect(self.submit)

        #Clear Entry Button
        self.clear_button = QPushButton('Clear', self)
        self.clear_button.setToolTip("Click to clear")
        self.clear_button.move(230, 310)
        self.clear_button.clicked.connect(self.clear)

        self.show()

    def center(self):
        qr = self.frameGeometry()
        cp = QApplication.desktop().availableGeometry().center()
        qr.moveCenter(cp)
        self.move(qr.topLeft())


    @pyqtSlot()
    def submit(self):
        if not all([self.textbox_first_name.text(),
self.textbox_last_name.text(),
                    self.textbox_username.text(),
self.textbox_password.text(),
                    self.textbox_email.text(), self.textbox_contact.text()]):
            QMessageBox.warning(self, "Error!!", "Please make sure that all
fields are filled!!", QMessageBox.Ok)
            return

        buttonReply = QMessageBox.question(self, "Submission confirmation",
"Are you sure you want to submit?",
                                           QMessageBox.Yes | QMessageBox.No,
QMessageBox.No)

        if buttonReply == QMessageBox.Yes:
            self.to_csv()

            QMessageBox.information(self, "Registration Successful", "Your
registration was successful!", QMessageBox.Ok)
        else:
            QMessageBox.information(self, "Registration Cancelled",
"Registration was cancelled.", QMessageBox.Ok)

    def clear(self):
            self.textbox_first_name.clear()
            self.textbox_last_name.clear()
            self.textbox_username.clear()
            self.textbox_password.clear()
            self.textbox_email.clear()
            self.textbox_contact.clear()

    def to_csv(self):
        first_name = self.textbox_first_name.text()
        last_name = self.textbox_last_name.text()
        username = self.textbox_username.text()
        password = self.textbox_password.text()
        email = self.textbox_email.text()
        contact = self.textbox_contact.text()

        with open('registrations.csv', 'a', newline='') as file:

file.write(f"{first_name},{last_name},{username},{password},{email},{contact}\
n")

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


Account Registration Form

Account Registration Form

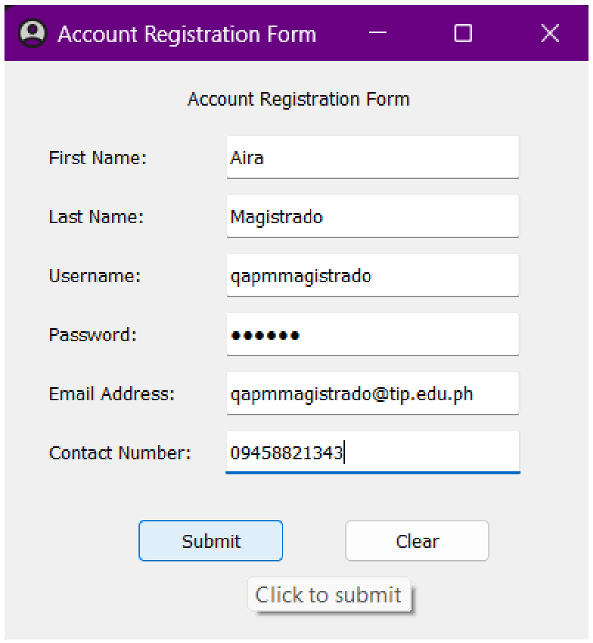
First Name:	Aira
Last Name:	
Contact Number:	

Error!!

Please make sure that all fields are filled!!

OK

Submit Clear

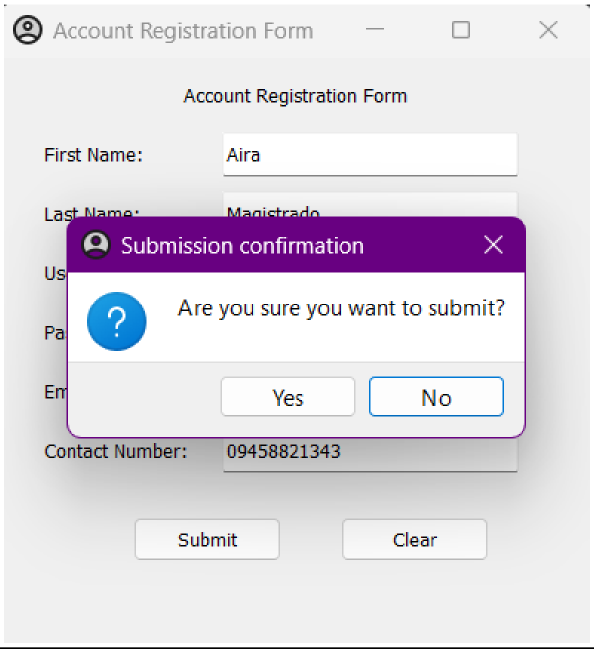

Account Registration Form

Account Registration Form

First Name:	Aira
Last Name:	Magistrado
Username:	qapmmagistrado
Password:	••••••
Email Address:	qapmmagistrado@tip.edu.ph
Contact Number:	09458821343

Submit Clear

Click to submit


Account Registration Form

Account Registration Form

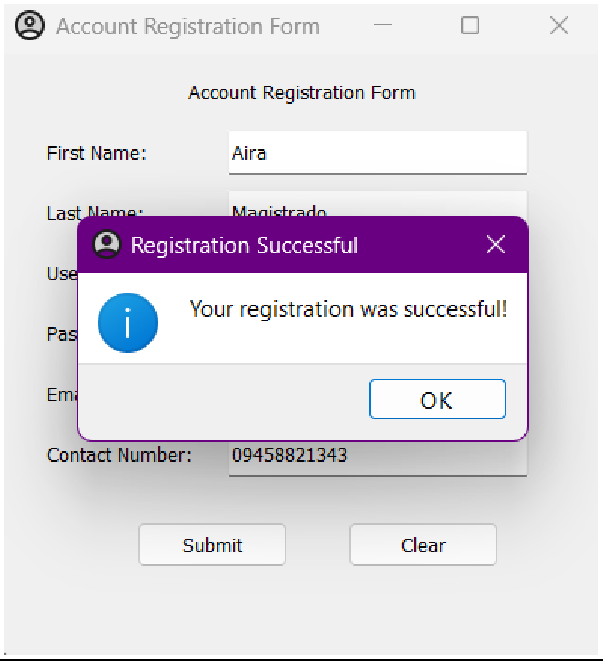
First Name:	Aira
Last Name:	Magistrado
Contact Number:	09458821343

Submission confirmation

Are you sure you want to submit?

Yes No

Submit Clear


Account Registration Form

Account Registration Form

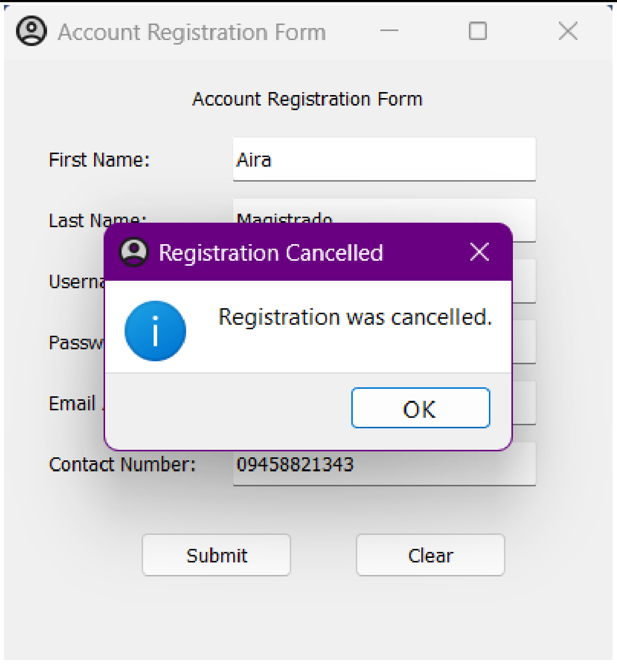
First Name:	Aira
Last Name:	Magistrado
Contact Number:	09458821343

Registration Successful

Your registration was successful!

OK

Submit Clear

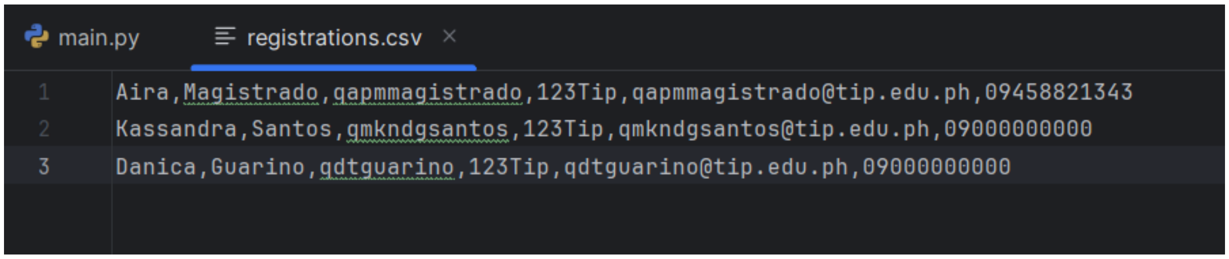

Account Registration Form

Account Registration Form

First Name:	Aira
Last Name:	Magistrado
Userna	
Passw	
Email	
Contact Number:	09458821343

Registration Cancelled

Registration was cancelled.



```
Aira,Magistrado,qapmmagistrado,123Tip,qapmmagistrado@tip.edu.ph,09458821343
Kassandra,Santos,qmkndgsantos,123Tip,qmkndgsantos@tip.edu.ph,09000000000
Danica,Guarino,qdtguarino,123Tip,qdtguarino@tip.edu.ph,09000000000
```

Questions:
1. What are the other signals available in PyQt5? (give at least 3 and describe each)
Some signals available in the PyQt5 include QPushButton, QLineEdit, and QCheckBox.The QPushbutton is the one in charge of the command buttons that are used in performing certain actions when clicked for example in the program above when clear was clicked it clears the user input in the registration form. The QLineEdit allows the user to enter and edit a line on the code. Like in the supplementary, it is used to enter the user's personal information. The QCheckBox allows the user to select or deselect the options available by clicking on them.

2. Why do you think that event handling in Python is divided into signals and slots?
I think that event handling in Python uses signals and slots to make the code easier to read and understand. Signals show what actions are taking place, such as a button click, whereas slots show what happens in response, such as performing an action when the button is clicked.

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?
Message boxes are used to make a GUI application more user-friendly by providing alerts, instructions, and confirmations to users regarding their actions. Like in the supplementary when something on the form is not completed, it will display a warning message and inform the user of the error, and when the registration is successful, it will also notify the user. Using textboxes guides the user through the application and prevents them from making mistakes

4. What is Error-handling and how was it applied in the task performed?
Error handling is the process of detecting and responding to program errors. It was used in the task to check if all required fields had been completed before allowing the user to submit the registration form. If any fields are empty, a warning message box is displayed.

5. What maybe the reasons behind the need to implement error handling?
Error handling is needed to implement because it helps provides users with clear feedback on issues, ensures correct data entry, and increases the application's reliability and user experience.

Conclusion:
In conclusion, I learned the basics of event handling in GUI applications. By using event handling with signals and slots, I discovered how to control what happens when buttons are clicked or text is typed. Signals indicate which actions are taking place, such as clicking a button, whereas slots specify what happens in response, such as performing an action when the button is clicked. I also learned about message boxes, which improve the user experience by providing instructions and guidance. I also understood the importance of implementing error handling as it gives users clear feedback about their inputs. For example, in the supplementary material, if users attempt to submit an incomplete form, a warning message appears, prompting them to correct their entries. Then, a confirmation message after successful registration reassures them that their submission was successful.