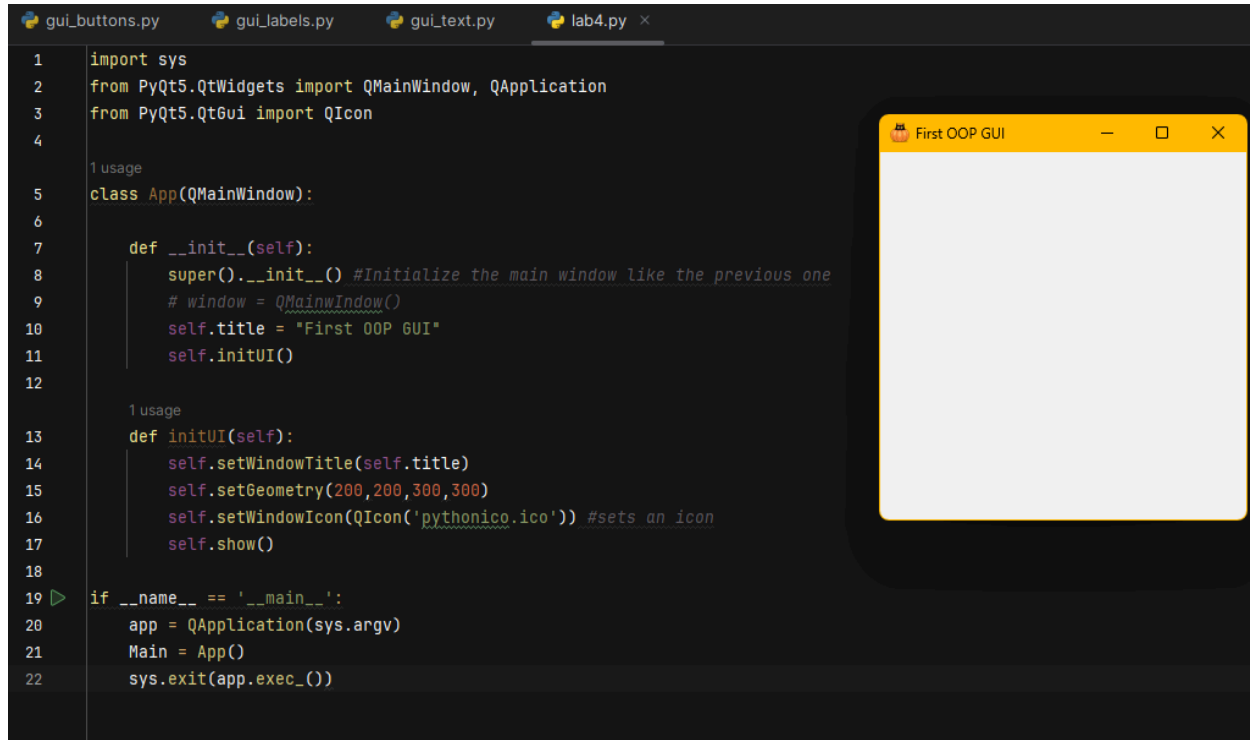


Activity #4 - Introduction to GUI Development using Pycharm	
Magistrado, Aira Pauleen M.	10/14/24
CPE009/CPE21S4	Maria Rizette Sayo

Procedure:

- 1) Adding Icon:



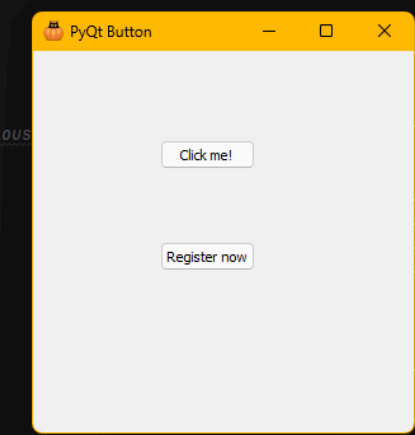
```

1  import sys
2  from PyQt5.QtWidgets import QMainWindow, QApplication
3  from PyQt5.QtGui import QIcon
4
5  1 usage
6  class App(QMainWindow):
7
8      def __init__(self):
9          super().__init__() #Initialize the main window like the previous one
10         # window = QMainWindow()
11         self.title = "First OOP GUI"
12         self.initUI()
13
14     1 usage
15     def initUI(self):
16         self.setWindowTitle(self.title)
17         self.setGeometry(200,200,300,300)
18         self.setWindowIcon(QIcon('pythonico.ico')) #sets an icon
19         self.show()
20
21 if __name__ == '__main__':
22     app = QApplication(sys.argv)
23     Main = App()
24     sys.exit(app.exec_())
  
```

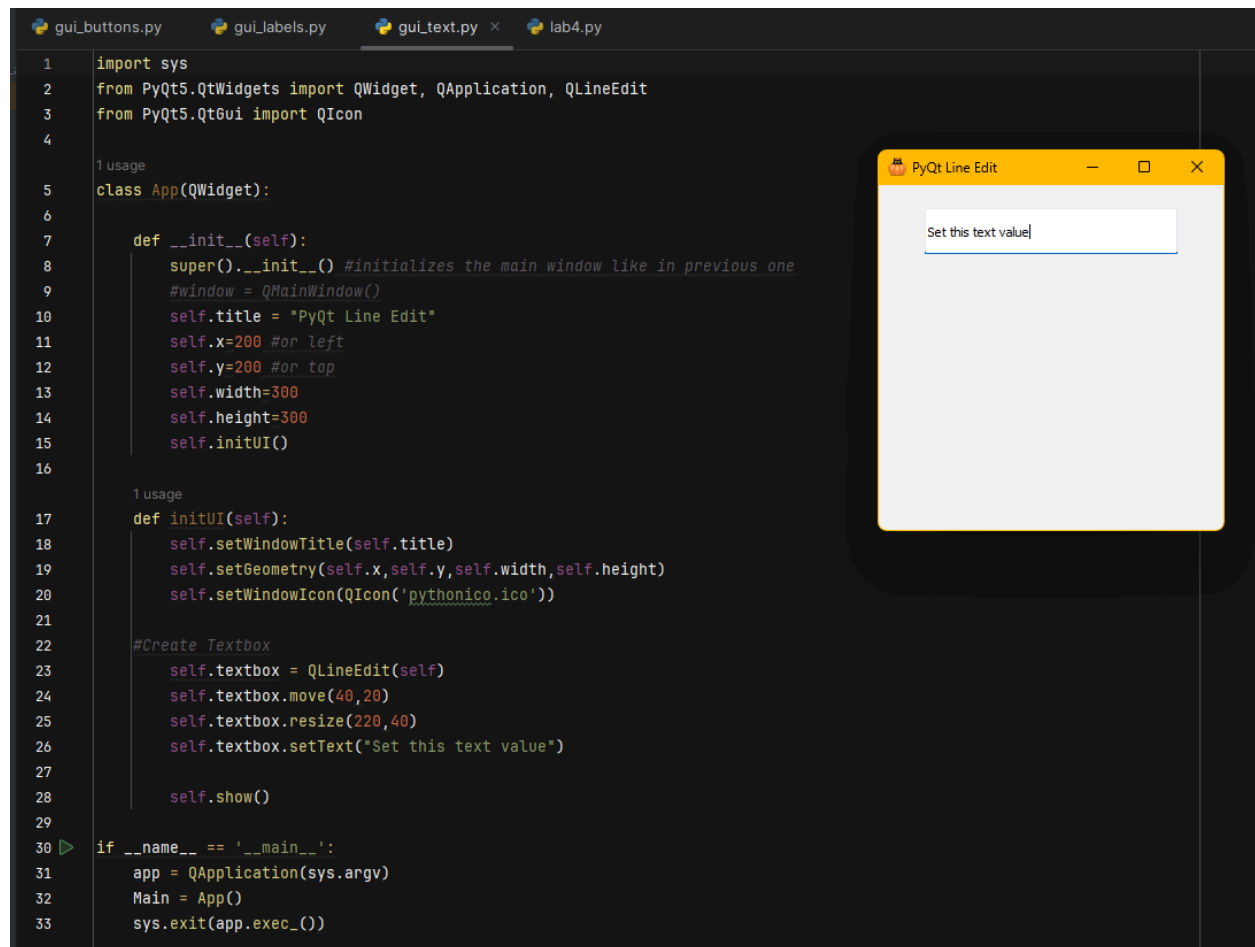
The screenshot shows the PyCharm IDE with a dark theme. The left pane displays the code for `gui_buttons.py`. The code imports `sys`, `QMainWindow`, `QApplication`, and `QIcon` from `PyQt5`. It defines a `class App(QMainWindow)` with an `__init__` method that calls `super().__init__()`, sets the window title to "First OOP GUI", and calls `self.initUI()`. The `initUI` method sets the window title, geometry (200, 200, 300, 300), and icon (using `pythonico.ico`), and calls `self.show()`. The main block of code creates a `QApplication` instance, instantiates the `App` class, and calls `sys.exit(app.exec_())`. The right pane shows a preview of the running application, which is a window titled "First OOP GUI" with a yellow title bar and a white content area.

2) Creating Buttons:

```
gui_buttons.py x gui_labels.py gui_text.py lab4.py
1 import sys
2 from PyQt5.QtWidgets import QWidget, QApplication, QMainWindow, QPushButton
3 from PyQt5.QtGui import QIcon
4
5 1 usage
6 class App(QWidget):
7     def __init__(self):
8         super().__init__() #Initialize the main window like the previous
9         # window = QMainWindow()
10        self.title = "PyQt Button"
11        self.x=200 #or left
12        self.y=200 #or top
13        self.width=300
14        self.height=300
15        self.initUI()
16
17 1 usage
18 def initUI(self):
19     self.setWindowTitle(self.title)
20     self.setGeometry(self.x,self.y,self.width,self.height)
21     self.setWindowIcon(QIcon('pythonico.ico'))
22
23     #In GUI Python, these buttons, textboxes, labels are called Widgets
24     #first button
25     self.button = QPushButton('Click me!', self)
26     self.button.setToolTip("You've hovered over me!")
27     self.button.move(100,70) #button.move(x,y)
28
29     #second button
30     self.button2 = QPushButton('Register now', self)
31     self.button2.setToolTip("Click the button to register!")
32     self.button2.move(100, 150)
33     self.show()
34
35 if __name__ == '__main__':
36     app = QApplication(sys.argv)
37     Main = App()
38     sys.exit(app.exec_())
```



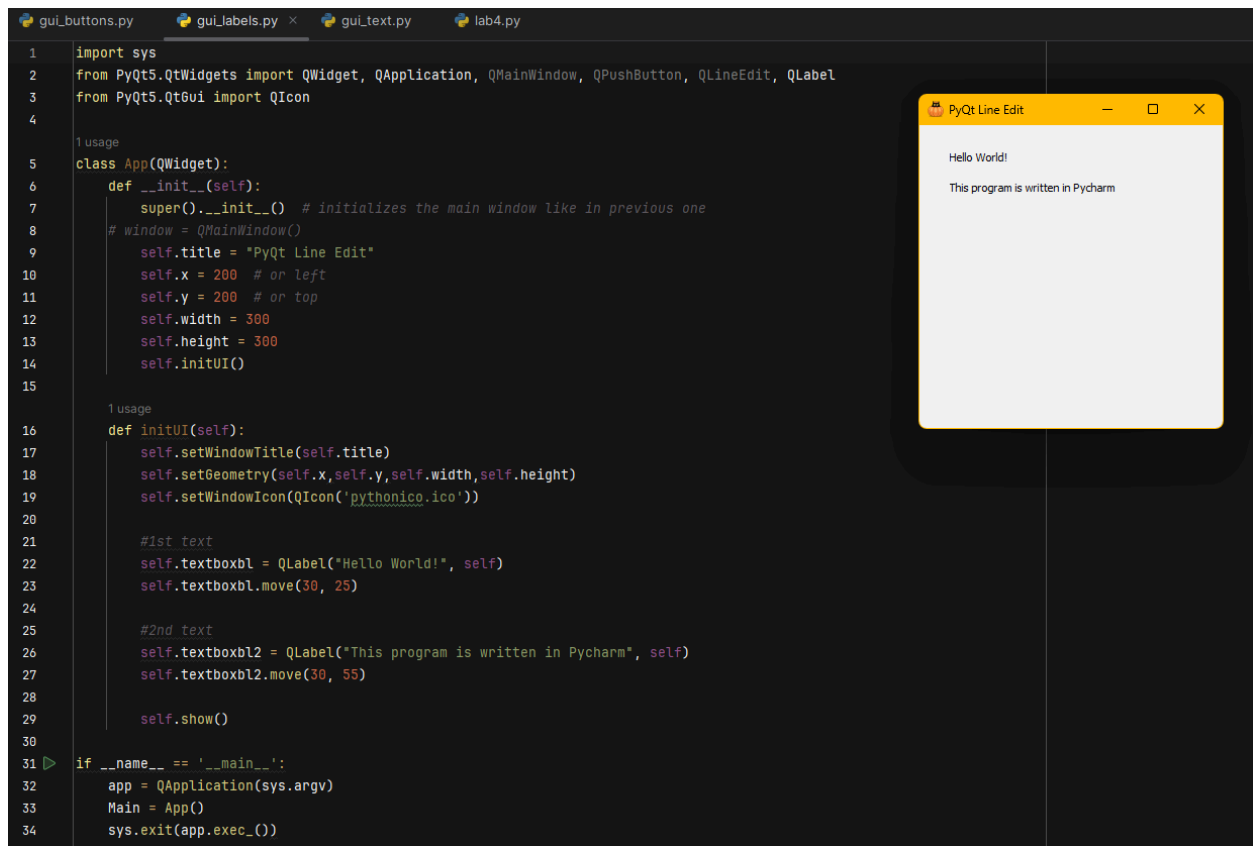
3) Creating Text Fields



```
1 import sys
2 from PyQt5.QtWidgets import QWidget, QApplication, QLineEdit
3 from PyQt5.QtGui import QIcon
4
5 1 usage
6 class App(QWidget):
7     def __init__(self):
8         super().__init__() #initializes the main window like in previous one
9         #window = QMainWindow()
10        self.title = "PyQt Line Edit"
11        self.x=200 #on left
12        self.y=200 #on top
13        self.width=300
14        self.height=300
15        self.initUI()
16
17 1 usage
18 def initUI(self):
19     self.setWindowTitle(self.title)
20     self.setGeometry(self.x,self.y,self.width,self.height)
21     self.setWindowIcon(QIcon('pythonico.ico'))
22
23     #Create Textbox
24     self.textbox = QLineEdit(self)
25     self.textbox.move(40,20)
26     self.textbox.resize(220,40)
27     self.textbox.setText("Set this text value")
28
29     self.show()
30
31 if __name__ == '__main__':
32     app = QApplication(sys.argv)
33     Main = App()
34     sys.exit(app.exec_())
```

The image shows a code editor with a dark theme. The top bar has four tabs: 'gui_buttons.py', 'gui_labels.py', 'gui_text.py' (which is active), and 'lab4.py'. The code in the editor is a Python script for a PyQt5 application. It defines a class 'App' that inherits from 'QWidget'. The '.__init__' method sets the window title to 'PyQt Line Edit', sets the geometry to (200, 200, 300, 300), and calls 'initUI'. The 'initUI' method sets the window title, geometry, and icon, creates a 'QLineEdit' widget, moves it to (40, 20), resizes it to (220, 40), sets its text to 'Set this text value', and calls 'show'. The main block of the script creates a 'QApplication' object, instantiates the 'App' class, and calls 'exec_()' to run the application. To the right of the code editor, there is a preview of the application window. The window has a yellow title bar with the text 'PyQt Line Edit' and standard window control buttons. The main area of the window is white and contains a single-line text input field with the text 'Set this text value|'.

4) Creating Labels



```
1 import sys
2 from PyQt5.QtWidgets import QWidget, QApplication, QMainWindow, QPushButton, QLineEdit, QLabel
3 from PyQt5.QtGui import QIcon
4
5 1 usage
6 class App(QWidget):
7     def __init__(self):
8         super().__init__() # initializes the main window like in previous one
9         # window = QMainWindow()
10        self.title = "PyQt Line Edit"
11        self.x = 200 # or left
12        self.y = 200 # or top
13        self.width = 300
14        self.height = 300
15        self.initUI()
16
17 1 usage
18 def initUI(self):
19     self.setWindowTitle(self.title)
20     self.setGeometry(self.x,self.y,self.width,self.height)
21     self.setWindowIcon(QIcon('pythonico.ico'))
22
23     #1st text
24     self.textboxbl = QLabel("Hello World!", self)
25     self.textboxbl.move(30, 25)
26
27     #2nd text
28     self.textboxbl2 = QLabel("This program is written in Pycharm", self)
29     self.textboxbl2.move(30, 55)
30
31     self.show()
32
33 31 ▶ if __name__ == '__main__':
34     app = QApplication(sys.argv)
35     Main = App()
36     sys.exit(app.exec_())
```

Supplementary Activity:

registration.py

```
from PyQt5.QtWidgets import QMainWindow, QApplication, QLabel, QLineEdit,
QPushButton

from PyQt5.QtGui import QIcon

from PyQt5.QtCore import Qt

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()

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

    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)

    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)

    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)

    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)


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)


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)


self.submit_button = QPushButton('Submit', self)

self.submit_button.setToolTip("Click to submit")

self.submit_button.move(100, 300)

self.submit_button.clicked.connect(self.submit)


self.clear_button = QPushButton('Clear', self)

self.clear_button.setToolTip("Click to clear")

self.clear_button.move(200, 300)

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())

def submit(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()

    print(f"First Name: {first_name}")

    print(f"Last Name: {last_name}")

    print(f"Username: {username}")

    print(f>Password: {password}")

    print(f>Email: {email}")

    print(f>Contact: {contact}")

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()

```

main.py

```

import sys

from PyQt5.QtWidgets import QApplication

```

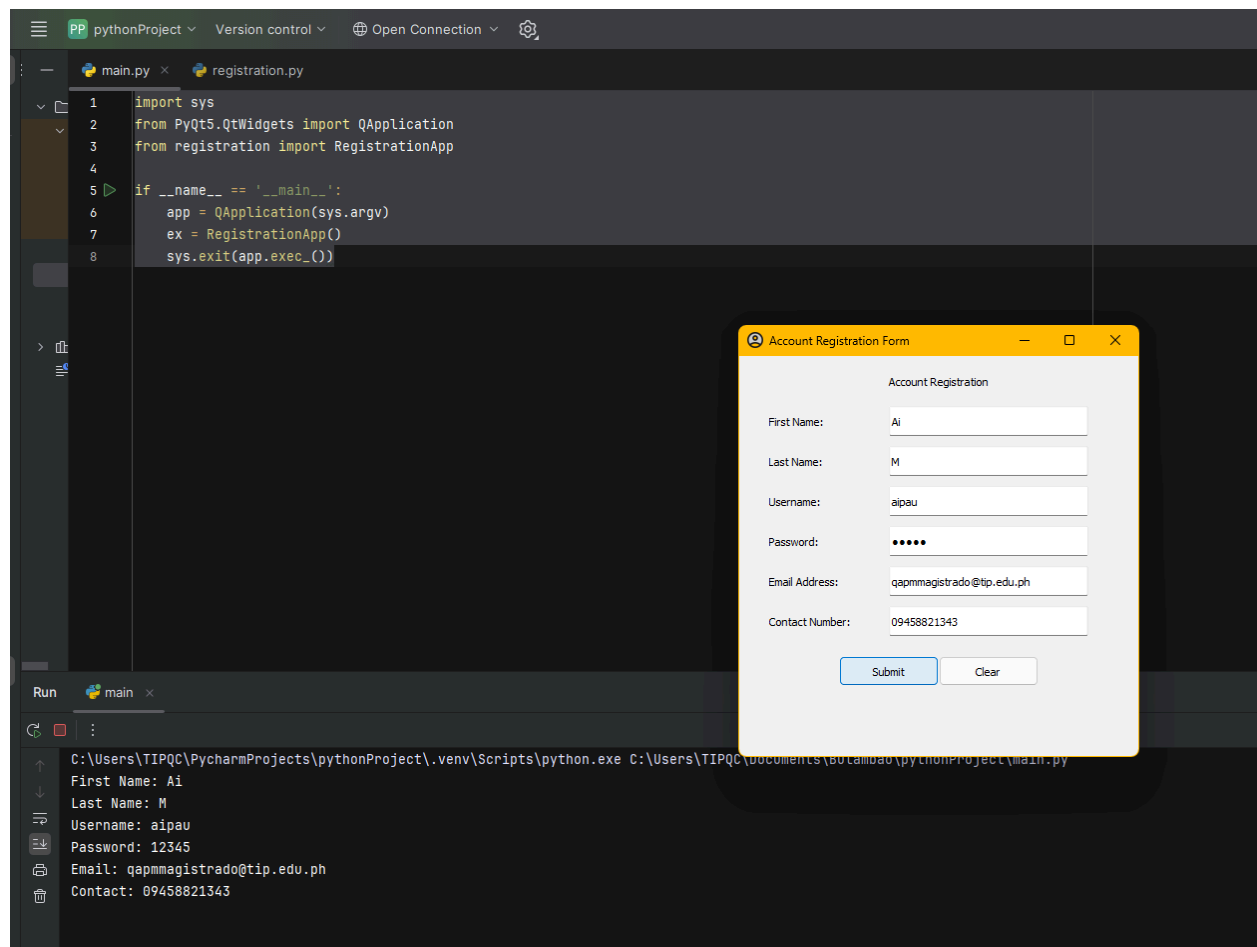
```
from registration import RegistrationApp
```

```
if __name__ == '__main__':
```

```
    app = QApplication(sys.argv)
```

```
    ex = RegistrationApp()
```

```
    sys.exit(app.exec_())
```



Questions

1. What are the common GUI Applications that general end-users such as home users, students, and office employees use? (give at least 3 and describe each)

The common GUI applications are Google Chrome, a user-friendly web browser for research and watching instructional or entertaining videos, watch educational or entertainment videos, Zoom, which gained popularity during the quarantine period when classes were conducted virtually and online meetings were used, and Canva, which allows users to create designs without the need for graphic design expertise because it offers a wide selection of templates for both personal and professional use.

2. Based from your answer in question 1, why do you think home users, students, and office employees use those GUI programs?

These types of graphical user interface (GUI) programs are used by a variety of users because they are simple to use, convenient for all users, and can be accessed from a variety of devices. They are designed with accessibility and usability in mind.

3. How does Pycharm help developers in making GUI applications, what would be the difference if developers made GUI programs without GUI Frameworks such as Pycharm or Tkinter?

PyCharm makes it easier for developers to create programs that have graphical user interfaces (GUIs). It offers features like code suggestions, debugging tools, and error highlighting. These features reduce errors and speed up the debugging process. Without the availability of tools such as PyCharm, developers would have to perform time-consuming and tedious manual tasks like creating and managing windows. These tasks are made easier by programs like PyCharm, which offer pre-built features and components.

4. What are the different platforms a GUI program may be created and deployed on? (Three is required then state why might a program be created on that specific platform).

The different platform that GUI programs may be created or employed are Windows because it can reach a wide range of users and companies use it a lot. Building and deploying apps is made simpler by the platform's exceptional development tools. MacOS renowned for its superior user experience and high performance. Especially for users in creative and professional fields, developers choose macOS to ensure their applications meet high standards and offer a polished, seamless experience. and Android which allows your application to reach a global audience because of its large user base and support for a wide variety of devices.

5. What is the purpose of `app = QApplication(sys.argv)`, `ex = App()`, and `sys.exit(app.exec_())`?

The function of `app = QApplication(sys.argv)` is to start your app in the code. It is the one in charge of setting things up so your program can run as GUI. `ex = App()` is in charge of the main window for your app. It's the foundation where everything else in your app will go. `sys.exit(app.exec_())` is in charge of the main loop of the app. It runs your app and then safely shuts it down when you're done.

Conclusion:

In conclusion, In this lab activity, I was introduced to and learned the PyCharm framework for GUI development. Like how to add icons, buttons, text fields, and labels. In the supplementary, I was tasked to make a simple Account Registration System from scratch. My understanding of design and usability was improved by adding buttons, centering the GUI, and aligning text fields and labels to the basic account registration system. My knowledge of PyCharm and Python's capabilities for building interactive programs has grown, and the different procedures have given me the skills and capability to create GUI applications.