|  |  |
| --- | --- |
| **Laboratory Activity No. 9** | |
| **Introduction to GUI Development using Pycharm** | |
| **Course Code:** CPE103 | **Program:** BSCPE |
| **Course Title:** Object-Oriented Programming | **Date Performed:** 03/22/25 |
| **Section:** BSCpE 1A | **Date Submitted:** 03/22/25 |
| **Name:** Nerio, Hannah Grace A. | **Instructor:** Maria Rizzete H. Sayo |
| **1. Objective(s):** | |
| This activity aims to familiarize students with the Pycharm framework for GUI Development | |
| **2. Intended Learning Outcomes (ILOs):** | |
| The students should be able to:   * 1. Identify the main components in a GUI Application   2. Create a simple GUI Application using Pycharm 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.  Pycharm is an integrated development environment used for programming in Python. It provides code analysis, a graphical debugger, an integrated unit tester, integration with version control systems, and supports web development with Django. | |
| **4. Materials and Equipment:** | |
| Desktop Computer with Anaconda Python or Pycharm Windows Operating System | |
| **5. Procedure:** | |

|  |
| --- |
| 1. Run the program and observe the output.   **Adding an icon**   1. Download any .ico picture from <https://icon-icons.com/> or any similar sites. 2. Place the icon in your folder (ex. Oopfa1<lastname>\_lab8) 3. Run the program again, the program should now have an icon similar to the program below.     **Creating Buttons**   1. Create a new .py file named **gui\_buttons.py** then copy the program as shown below: |

|  |
| --- |
| 1. Run the program and observe the output. 2. Add a new button named button2 named Register to the GUI that will display “this button does nothing.. yet..” when it is hovered.   **Creating Text Fields**   1. Create a new file named **gui\_text.py** and copy the code shown below: |

|  |
| --- |
| 1. Run the program and observe the error. 2. Add an import QLineEdit to the Pycharm.Widgets import 3. Run the program and observe the output. 4. Add the following code below self.textbox.resize()   self.textbox.setText("Set this text value")  4. Run the program again then resize the textbox so that it fits in the Window and that its height is just above the written text’s height.  **Creating Labels**  1. Create a new file called **gui\_labels.py** and copy the following code below: |

|  |
| --- |
| 1. Run the program and observe the output. 2. Add the necessary Widget at the import line to make the program run. 3. Center the label by adjusting the parameters of .move(). This is called Absolute Positioning. 4. Create a new label called “This program is written in Pycharm” and place it at the center and below the Hello World! |
| **6. Supplementary Activity:** |
| **Task**  Create an Object-Oriented GUI Application for a simple Account Registration System with the following required information: first name, last name, username, password, email address, contact number.  Requirements:   * The GUI must be centered on your screen. * The GUI Components should be organized according to the order of information required using Absolute Positioning. * The position of the components should be automatically computed based on the top component. * All the text fields should be accompanied with their corresponding label on the left side while the text field is on the right side. * There should a program title other than the Window Title. * There should be a submit button and clear button at the bottom (submit button on the left, clear button on the right). * The program should be launched on **main.py** while the GUI Codes should be on a separate file called   **registration.py** |

|  |
| --- |
| **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)   Common GUI applications used by home users, students, and office employees include tools like the Microsoft Office Suite, web browsers, and media players. The Microsoft Office Suite, with programs such as Word, Excel, and PowerPoint, provides an intuitive interface for creating documents, analyzing data, and making presentations, making it essential for both professional and educational tasks. Web browsers like Google Chrome and Mozilla Firefox offer user-friendly navigation tools to access websites, search the internet, and manage bookmarks, simplifying online activities. Media players, such as VLC Media Player, are designed for playing audio and video files, with graphical controls for volume, playback, and file selection, enhancing entertainment and learning experiences.   1. Based from your answer in question 1, why do you think home users, students, and office employees use those GUI programs?   Home users, students, and office employees rely on these GUI programs because they simplify tasks and improve productivity. The Microsoft Office Suite is widely used due to its intuitive design and powerful features, making it easy to create documents, analyze data, and deliver presentations efficiently. Web browsers are essential for accessing information, completing online tasks, and staying connected, offering user-friendly tools for searching, bookmarking, and browsing the internet. Media players like VLC Media Player enhance entertainment and learning experiences by allowing seamless playback of audio and video files.   1. 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 simplifies the creation of GUI applications by offering essential features like code completion, error checking, and debugging, making development more efficient and user-friendly. It supports popular GUI frameworks such as Tkinter and PyQt, allowing developers to focus on designing interfaces without worrying about low-level implementation details. Without GUI frameworks or tools like PyCharm, developers would need to manually write extensive code for tasks like rendering windows, managing events, and handling user interactions. This process is not only time-consuming but also more prone to errors, as it requires a deeper understanding of graphical programming.   1. 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)   GUI programs can be created and deployed on platforms such as Windows, macOS, and Linux. Windows is a popular choice due to its widespread use in homes, schools, and offices, offering compatibility with various tools and frameworks like Tkinter and PyQt. Developers often target Windows for its large user base and seamless integration with desktop applications. macOS, on the other hand, is favored for its polished interface and focus on user experience, making it ideal for professional or creative applications. Tools like PyObjC support the development of Python-based apps on macOS. Meanwhile, Linux is highly regarded by developers and IT professionals for its flexibility and open-source environment. With frameworks like Gtk and Qt, Linux is an excellent platform for creating customizable and stable applications.   1. What is the purpose of **app = QApplication(sys.argv)**, **ex = App()**, and **sys.exit(app.exec\_())**?   The lines of code app = QApplication(sys.argv), ex = App(), and sys.exit(app.exec\_()) are essential for running a PyQt-based GUI application. The first line, app = QApplication(sys.argv), initializes the application by creating an instance of QApplication, which sets up the GUI environment and handles system-level arguments passed via sys.argv. The second line, ex = App(), creates an instance of the custom GUI class App, where the interface and logic are defined, effectively setting up the main application window. Finally, sys.exit(app.exec\_()) starts the application's event loop using app.exec\_(), keeping the GUI responsive to user interactions. It also ensures a clean exit when the application is closed by terminating the program properly with sys.exit(). |
| **7. Conclusion:** |
| This laboratory activity introduced the fundamentals of GUI development using PyCharm, helping us understand how to create interactive applications with graphical elements. We explored essential components like buttons, text fields, and labels, learning how to position and organize them effectively within a user interface. By working with GUI frameworks such as Tkinter and PyQt, we saw how these tools simplify the development process, making applications more intuitive and accessible for users. Additionally, we gained hands-on experience in structuring GUI programs, applying object-oriented programming principles to enhance maintainability and scalability. Understanding the role of GUI frameworks is crucial, as developing applications without them would require extensive manual coding for event handling and interface rendering. |
| **8. Assessment Rubric:** |