



---

# Building Desktop Applications using PyQt6

---

**CS/CE 355L/373L: Database Systems**  
**Lab 08**

**Lab done by:**  
Syed Asghar Abbas Zaidi (sz07201)

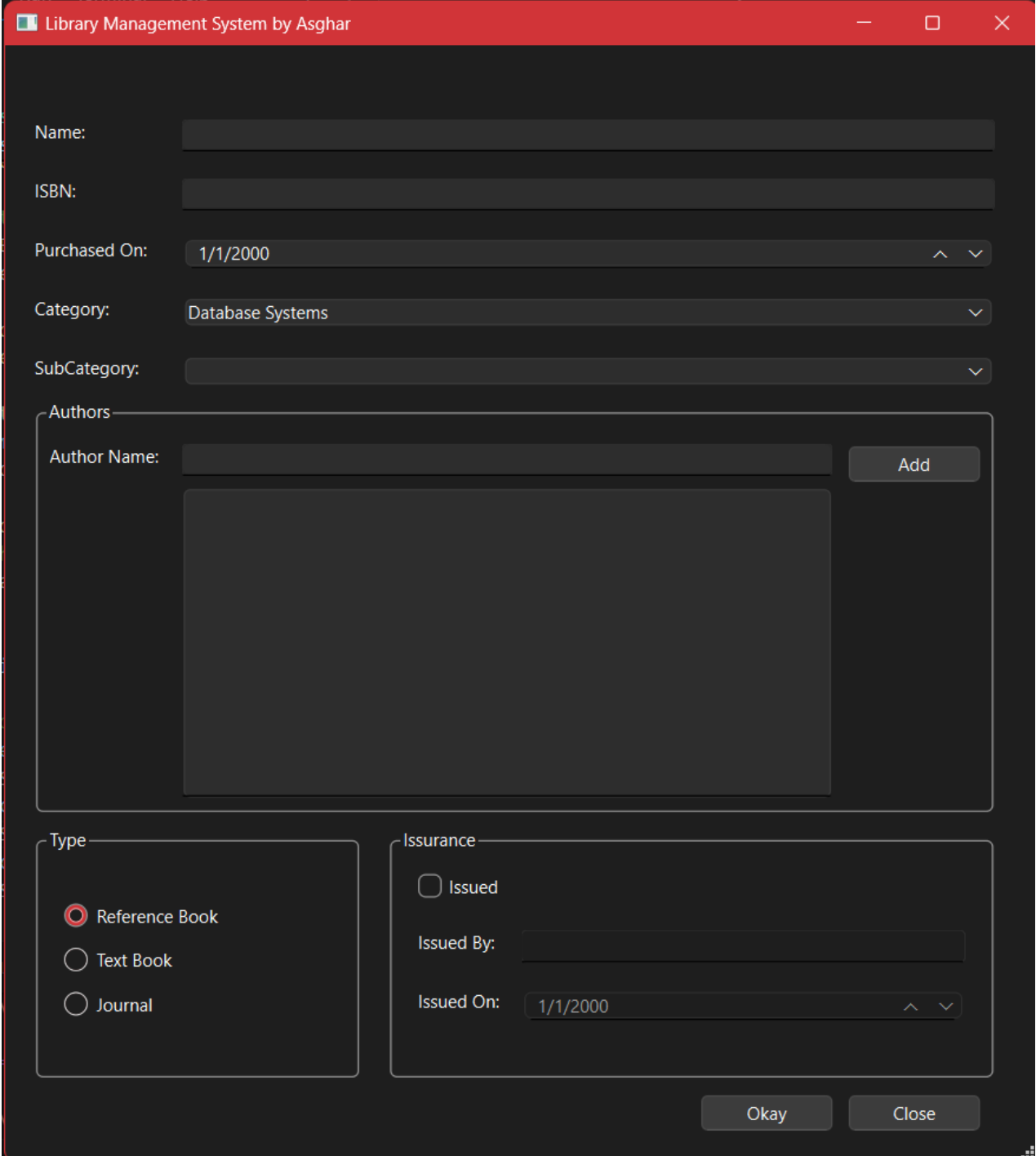
**Instructor:**  
Muhammad Umer Tariq  
Saima Shaheen

August 26, 2024

# 1 Overview

My system's theme is "Dark Mode" by default. As such you will see the windows reflecting that change. Due to my Windows OS not being activated, I can't change that, hopefully it's not a issue.

I have attached the code that you can use to verify various conditions we were asked to fulfill. I can also be called out for a viva if verification of some sort is needed



The screenshot shows a desktop application window titled "Library Management System by Asghar". The interface is dark-themed and contains the following elements:

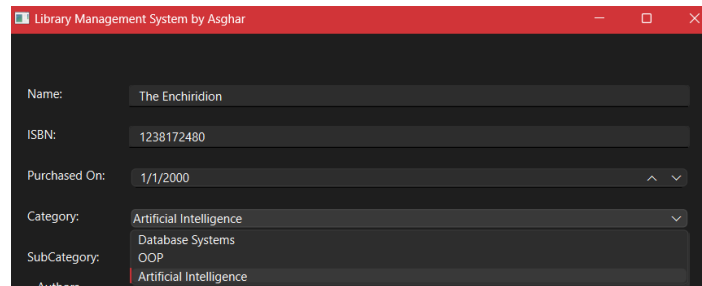
- Name:** A text input field.
- ISBN:** A text input field.
- Purchased On:** A date picker showing "1/1/2000".
- Category:** A dropdown menu with "Database Systems" selected.
- SubCategory:** A dropdown menu.
- Authors:** A section containing:
  - Author Name:** A text input field.
  - Add:** A button.
  - A large empty rectangular area below the input field.
- Type:** A section with three radio buttons:
  - ☒ Reference Book
  - ☐ Text Book
  - ☐ Journal
- Issurance:** A section containing:
  - ☐ Issued
  - Issued By:** A text input field.
  - Issued On:** A date picker showing "1/1/2000".
- Buttons:** "Okay" and "Close" buttons at the bottom right.

Figure 1: Library Management System App

## 2 Proofs

I will now be showcasing few screenshots that proofs that my code is in-fact properly working.

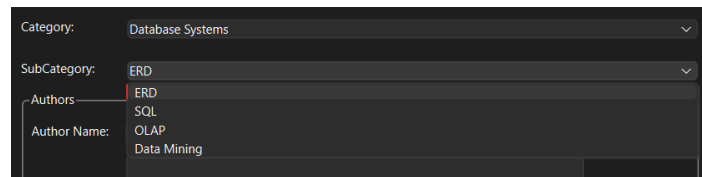
### 2.0.1 Category Combo



The screenshot shows a window titled "Library Management System by Asghar". It contains a form with the following fields:

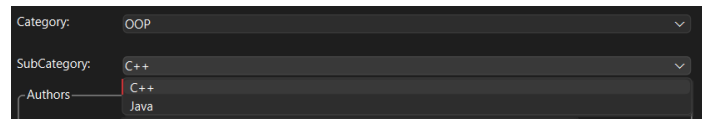
- Name: The Enchiridion
- ISBN: 1238172480
- Purchased On: 1/1/2000
- Category: Artificial Intelligence (selected)
- SubCategory: OOP
- Authors: Artificial Intelligence

### 2.0.2 Subcategory



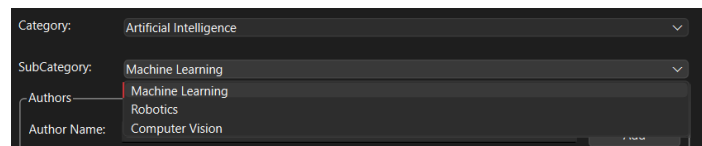
The screenshot shows the same form as above, but with the SubCategory dropdown menu open. The options are:

- ERD
- SQL
- OLAP
- Data Mining



The screenshot shows the same form as above, but with the SubCategory dropdown menu open. The options are:

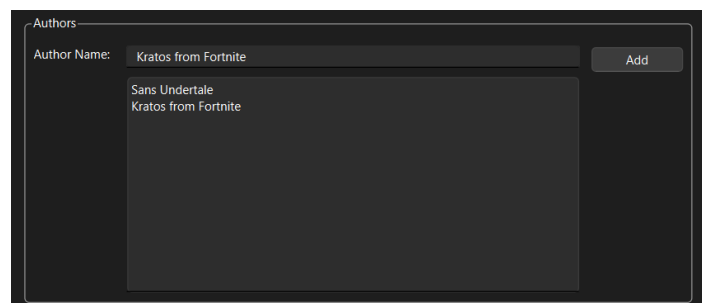
- C++
- Java



The screenshot shows the same form as above, but with the SubCategory dropdown menu open. The options are:

- Machine Learning
- Robotics
- Computer Vision

### 2.0.3 Adding Author's name



The screenshot shows a dialog box titled "Authors". It contains a form with the following fields:

- Author Name: Kratos from Fortnite
- Add button

Below the form, there is a list of authors:

- Sans Undertale
- Kratos from Fortnite

### 3 Error Messages

Attaching few error messages, I have attached the code in text form in Section 5. You can test-run my code using that as well

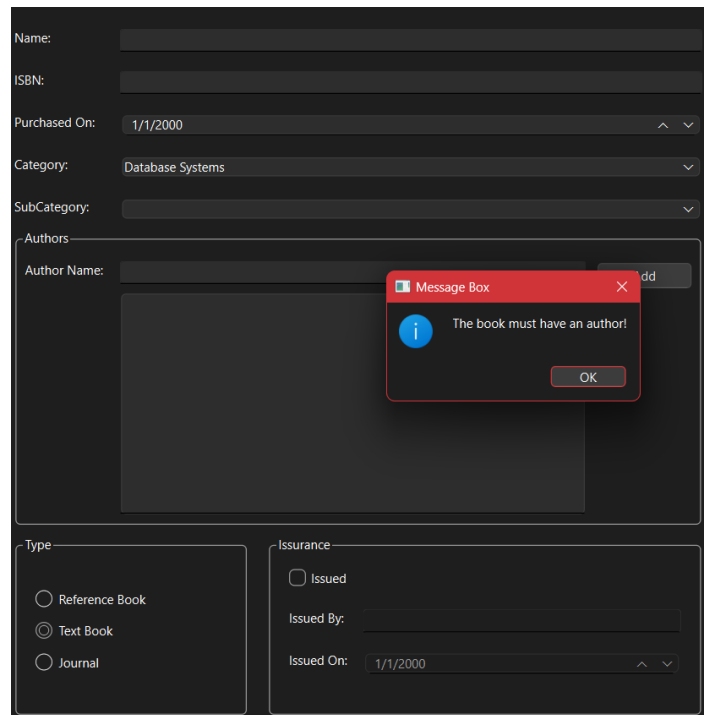


Figure 2: Error shown if author not picked if a Book is selected

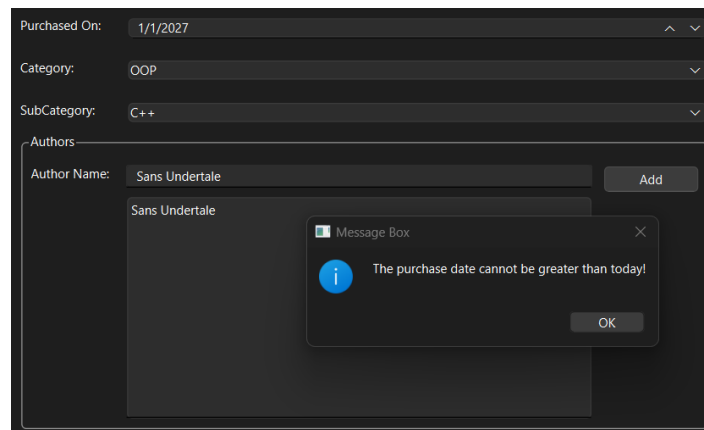


Figure 3: If you select purchase date far further in the future

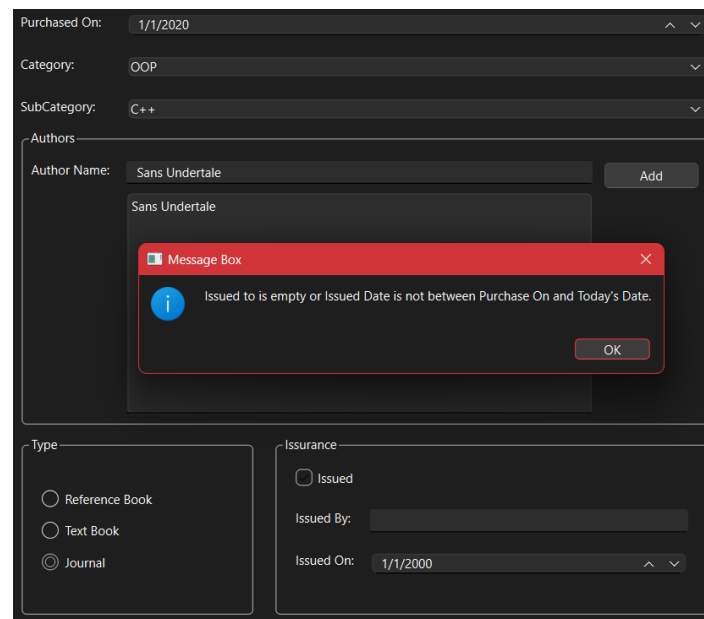


Figure 4: Issue Box not properly filled

## 4 Code Snippets

As we were asked to attach our code snippets as well. I will be doing that.

```

Example > app.py > UI > __init__
1  # Import all the required libraries
2  from PyQt6 import QtWidgets, uic, QtGui, QtCore
3  from PyQt6.QtCore import QDate
4  from PyQt6.QtWidgets import QMessageBox # Import QMessageBox correctly
5  import sys
6
7  class UI(QtWidgets.QMainWindow):
8      def __init__(self):
9          # Call the inherited classes __init__ method
10         super(UI, self).__init__()
11
12         # Load the .ui file
13         uic.loadUi('LibraryManagementSystem.ui', self)
14
15         # Show the GUI
16         self.show()
17
18         # Adding items to comboBox programmatically
19         self.Category_ComboBox.addItem("Database Systems")
20         self.Category_ComboBox.addItem("OOP")
21         self.Category_ComboBox.addItem("Artificial Intelligence")
22
23         # Event Handling
24         self.Category_ComboBox.currentIndexChanged.connect(self.on_combobox_changed)
25         self.AddAuthorButton.clicked.connect(self.handle_AuthorButtonClick)
26         self.Issued_checkBox.toggled.connect(self.Issued_checkBox_Toggled)
27
28         ##Initialize the state of the widgets based on the checkbox state
29         self.Issued_checkBox_Toggled(self.Issued_checkBox.isChecked())
30
31         ##OkayButton event
32         self.OkayButton.clicked.connect(self.handle_OkayButtonClick)
33         self.CloseButton.clicked.connect(self.close_window)
34
35     def close_window(self):
36         self.close()

```

```

37 def handle_AuthorButtonClick(self):
38     Author_Name= self.AuthorName_LineEdit.text()
39     self.AuthorName_textEdit.append(Author_Name)
40
41 def on_combobox_changed(self):
42     # Clear the contents of the second combo box
43     self.SubCategory_ComboBox.clear()
44
45     # Get the selected item from the first combo box
46     selected_item = self.Category_ComboBox.currentText()
47
48     # Conditional logic to change contents of the second combo box
49     if selected_item == "Database Systems":
50         self.SubCategory_ComboBox.addItem(["ERD", "SQL", "OLAP", "Data Mining"])
51     elif selected_item == "OOP":
52         self.SubCategory_ComboBox.addItem(["C++", "Java"])
53     elif selected_item == "Artificial Intelligence":
54         self.SubCategory_ComboBox.addItem(["Machine Learning", "Robotics", "Computer Vision"])
55
56 def Issued_checkBox_Toggled(self, checked):
57     self.IssuedBy_LineEdit.setEnabled(checked)
58     self.IssuedOn_dateEdit.setEnabled(checked)
59
60 def handle_okayButtonClick(self):
61     ISBN_LineEdit_text = self.ISBN_LineEdit.text() # Retrieve the text from the QLineEdit
62     num_characters = len(ISBN_LineEdit_text) # Calculate the number of characters
63
64     purchase_date = self.Purchase_DateEdit.date() # Retrieve the date from the QDateEdit
65     today_date = QDate.currentDate() #Get today's date
66
67     if purchase_date > today_date:
68         msg_box = QMessageBox()
69         msg_box.setWindowTitle("Message Box")
70         msg_box.setText("The purchase date cannot be greater than today!")
71         msg_box.setIcon(QMessageBox.Icon.Information) # Use an information icon

```

```

72         msg_box.setStandardButtons(QMessageBox.StandardButton.Ok) # Add OK button
73         msg_box.exec() # Display the message box
74
75     #Show error if number of characters are greater than 12 in ISBN
76     elif num_characters > 12:
77         msg_box = QMessageBox()
78         msg_box.setWindowTitle("Message Box")
79         msg_box.setText("The Length of ISBN can't be greater than 12!")
80         msg_box.setIcon(QMessageBox.Icon.Information) # Use an information icon
81         msg_box.setStandardButtons(QMessageBox.StandardButton.Ok) # Add OK button
82         msg_box.exec() # Display the message box
83
84
85     elif not self.Journal_RadioButton.isChecked():
86         author_text = self.AuthorName_textEdit.toPlainText().strip()
87         if not author_text:
88             msg_box = QMessageBox()
89             msg_box.setWindowTitle("Message Box")
90             msg_box.setText("The book must have an author!")
91             msg_box.setIcon(QMessageBox.Icon.Information) # Use an information icon
92             msg_box.setStandardButtons(QMessageBox.StandardButton.Ok) # Add OK button
93             msg_box.exec() # Display the message box
94
95     elif self.Issued_checkBox.isChecked():
96         issued_by_text = self.IssuedBy_LineEdit.text().strip()
97         issued_on_date = self.IssuedOn_dateEdit.date()
98
99         # Get today's date
100        today_date = QDate.currentDate()
101
102        # Validate the "Issued By" field
103        if not issued_by_text or issued_on_date > today_date:
104            msg_box = QMessageBox()
105            msg_box.setWindowTitle("Message Box")
106            msg_box.setText("Issued to is empty or Issued Date is not between Purchase On and Today's Date.")

```

```
107         msg_box.setIcon(QMessageBox.Icon.Information) # Use an information icon
108         msg_box.setStandardButtons(QMessageBox.StandardButton.Ok) # Add OK button
109         msg_box.exec() # Display the message box
110
111     elif self.Journal_RadioButton.isChecked():
112         msg_box = QMessageBox()
113         msg_box.setWindowTitle("Message Box")
114         msg_box.setText("Book Added Successfully")
115         msg_box.setIcon(QMessageBox.Icon.Information) # Use an information icon
116         msg_box.setStandardButtons(QMessageBox.StandardButton.Ok) # Add OK button
117         msg_box.exec() # Display the message box
118
119     else:
120         msg_box = QMessageBox()
121         msg_box.setWindowTitle("Message Box")
122         msg_box.setText("Book Added Successfully")
123         msg_box.setIcon(QMessageBox.Icon.Information) # Use an information icon
124         msg_box.setStandardButtons(QMessageBox.StandardButton.Ok) # Add OK button
125         msg_box.exec() # Display the message box
126
127
128 if __name__ == "__main__":
129     app = QtWidgets.QApplication(sys.argv) # Create an instance of QtWidgets.QApplication
130     window = UI() # Create an instance of our class
131     app.exec() # Start the application
```

## 5 Appendix

Although I have attached the code screenshot snippets, I am also attaching the text

```
1 # Import all the required libraries
2 from PyQt6 import QtWidgets, uic, QtGui, QtCore
3 from PyQt6.QtCore import QDate
4 from PyQt6.QtWidgets import QMessageBox # Import QMessageBox correctly
5 import sys
6
7 class UI(QtWidgets.QMainWindow):
8     def __init__(self):
9         # Call the inherited classes __init__ method
10         super(UI, self).__init__()
11
12         # Load the .ui file
13         uic.loadUi('LibraryManagementSystem.ui', self)
14
15         # Show the GUI
16         self.show()
17
18         # Adding items to comboBox programmatically
19         self.Category_ComboBox.addItem("Database Systems")
20         self.Category_ComboBox.addItem("OOP")
21         self.Category_ComboBox.addItem("Artificial Intelligence")
22
23         # Event Handling
24         self.Category_ComboBox.currentIndexChanged.connect(self.
25             on_combobox_changed)
26         self.AddAuthorButton.clicked.connect(self.handle_AuthorButtonClick)
27         self.Issued_checkBox.toggled.connect(self.Issued_checkBox_Toggled)
28
29         ##Initialize the state of the widgets based on the checkbox state
30         self.Issued_checkBox_Toggled(self.Issued_checkBox.isChecked())
31
32         ##OkayButton event
33         self.OkayButton.clicked.connect(self.handle_OkayButtonClick)
34         self.CloseButton.clicked.connect(self.close_window)
35
36     def close_window(self):
37         self.close()
38
39     def handle_AuthorButtonClick(self):
40         Author_Name= self.AuthorName_LineEdit.text()
41         self.AuthorName_textEdit.append(Author_Name)
42
43     def on_combobox_changed(self):
44         # Clear the contents of the second combo box
45         self.SubCategory_ComboBox.clear()
```



```
46     # Get the selected item from the first combo box
47     selected_item = self.Category_ComboBox.currentText()
48
49     # Conditional logic to change contents of the second combo box
50     if selected_item == "Database Systems":
51         self.SubCategory_ComboBox.addItem(["ERD", "SQL", "OLAP", "Data
52             Mining"])
53     elif selected_item == "OOP":
54         self.SubCategory_ComboBox.addItem(["C++", "Java"])
55     elif selected_item == "Artificial Intelligence":
56         self.SubCategory_ComboBox.addItem(["Machine Learning", "
57             Robotics", "Computer Vision"])
58
59     def Issued_checkBox_Toggled(self, checked):
60         self.IssuedBy_LineEdit.setEnabled(checked)
61         self.IssuedOn_dateEdit.setEnabled(checked)
62
63     def handle_OkayButtonClick(self):
64         ISBN_LineEdit_text = self.ISBN_LineEdit.text() # Retrieve the text
65             from the QLineEdit
66         num_characters = len(ISBN_LineEdit_text) # Calculate the number of
67             characters
68
69         purchase_date = self.Purchase_DateEdit.date() # Retrieve the date
70             from the QDateEdit
71         today_date = QDate.currentDate() #Get today's date
72
73         if purchase_date > today_date:
74             msg_box = QMessageBox()
75             msg_box.setWindowTitle("Message Box")
76             msg_box.setText("The purchase date cannot be greater than today!")
77             msg_box.setIcon(QMessageBox.Icon.Information) # Use an
78                 information icon
79             msg_box.setStandardButtons(QMessageBox.StandardButton.Ok) # Add
80                 OK button
81             msg_box.exec() # Display the message box
82
83         #Show error if number of characters are greater than 12 in ISBM
84         elif num_characters > 12:
85             msg_box = QMessageBox()
86             msg_box.setWindowTitle("Message Box")
87             msg_box.setText("The Length of ISBN can't be greater than 12!")
88             msg_box.setIcon(QMessageBox.Icon.Information) # Use an
89                 information icon
90             msg_box.setStandardButtons(QMessageBox.StandardButton.Ok) # Add
91                 OK button
92             msg_box.exec() # Display the message box
```

```
86     elif not self.Journal_RadioButton.isChecked():
87         author_text = self.AuthorName_textEdit.toPlainText().strip()
88         if not author_text:
89             msg_box = QMessageBox()
90             msg_box.setWindowTitle("Message Box")
91             msg_box.setText("The book must have an author!")
92             msg_box.setIcon(QMessageBox.Icon.Information) # Use an
93                 information icon
94             msg_box.setStandardButtons(QMessageBox.StandardButton.Ok) #
95                 Add OK button
96             msg_box.exec() # Display the message box
97
98     elif self.Issued_checkBox.isChecked():
99         issued_by_text = self.IssuedBy_LineEdit.text().strip()
100         issued_on_date = self.IssuedOn_dateEdit.date()
101
102         # Get today's date
103         today_date = QDate.currentDate()
104
105         # Validate the "Issued By" field
106         if not issued_by_text or issued_on_date > today_date:
107             msg_box = QMessageBox()
108             msg_box.setWindowTitle("Message Box")
109             msg_box.setText("Issued to is empty or Issued Date is not
110                 between Purchase On and Today's Date.")
111             msg_box.setIcon(QMessageBox.Icon.Information) # Use an
112                 information icon
113             msg_box.setStandardButtons(QMessageBox.StandardButton.Ok) #
114                 Add OK button
115             msg_box.exec() # Display the message box
116
117     elif self.Journal_RadioButton.isChecked():
118         msg_box = QMessageBox()
119         msg_box.setWindowTitle("Message Box")
120         msg_box.setText("Book Added Successfully")
121         msg_box.setIcon(QMessageBox.Icon.Information) # Use an
122             information icon
123         msg_box.setStandardButtons(QMessageBox.StandardButton.Ok) # Add
124             OK button
125         msg_box.exec() # Display the message box
126
127     else:
128         msg_box = QMessageBox()
129         msg_box.setWindowTitle("Message Box")
130         msg_box.setText("Book Added Successfully")
131         msg_box.setIcon(QMessageBox.Icon.Information) # Use an
132             information icon
133         msg_box.setStandardButtons(QMessageBox.StandardButton.Ok) # Add
134             OK button
135         msg_box.exec() # Display the message box
```

```
127
128
129 if __name__ == "__main__":
130     app = QtWidgets.QApplication(sys.argv) # Create an instance of
131         QtWidgets.QApplication
132     window = UI() # Create an instance of our class
133     app.exec() # Start the application
```