Building Desktop Applications with Multiple Forms in PyQt6

CS/CE 355L/373L: Database Systems

Lab 02

Lab done by:
Syed Asghar Abbas Zaidi (sz07201)

Instructor:
Muhammad Umer Tariq
Saima Shaheen

September 3, 2024

Contents

1 Overview

In this exercise, I developed a Library Management System that will allow the users to search for books by providing criteria, view the details of a particular book, and delete the book from the system.

2 Main Window

I implemented the Book Search Form with the specified functionality. When the form is loaded, it displays all the books stored in a Python list within the table widget. The table widget includes columns for ISBN, Title, Category, Type, and Issued status. Additionally, I configured the Category combo box to show the options Database, OOP, and AI, allowing users to filter and search books based on these categories.

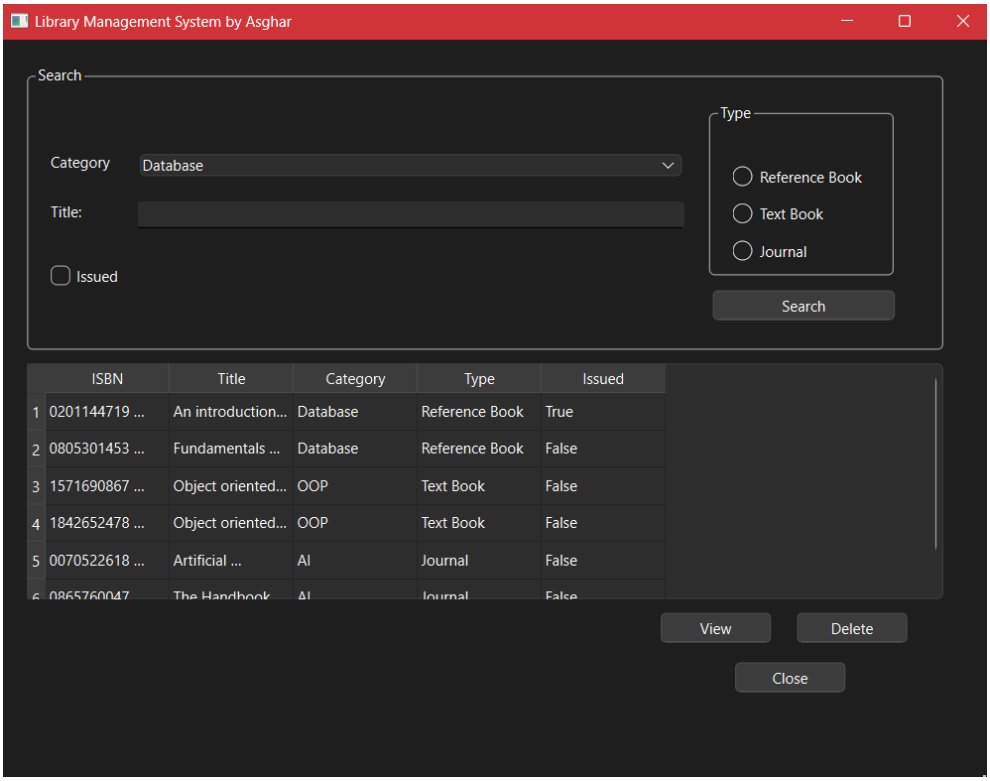

Figure 1: Library Management System using PyQt6

3 Search Functionality

I implemented a search feature that allows users to filter books based on various criteria including Category, Type, Title, and Issued status. Users can input their desired search parameters and click the 'Search' button to display only those books that meet the specified criteria. For instance, if a user selects 'Database' as the Category, leaves the Title field empty, chooses 'Reference Book' for the Type, and checks the Issued Checkbox, the search results will show only the books that match all these conditions.

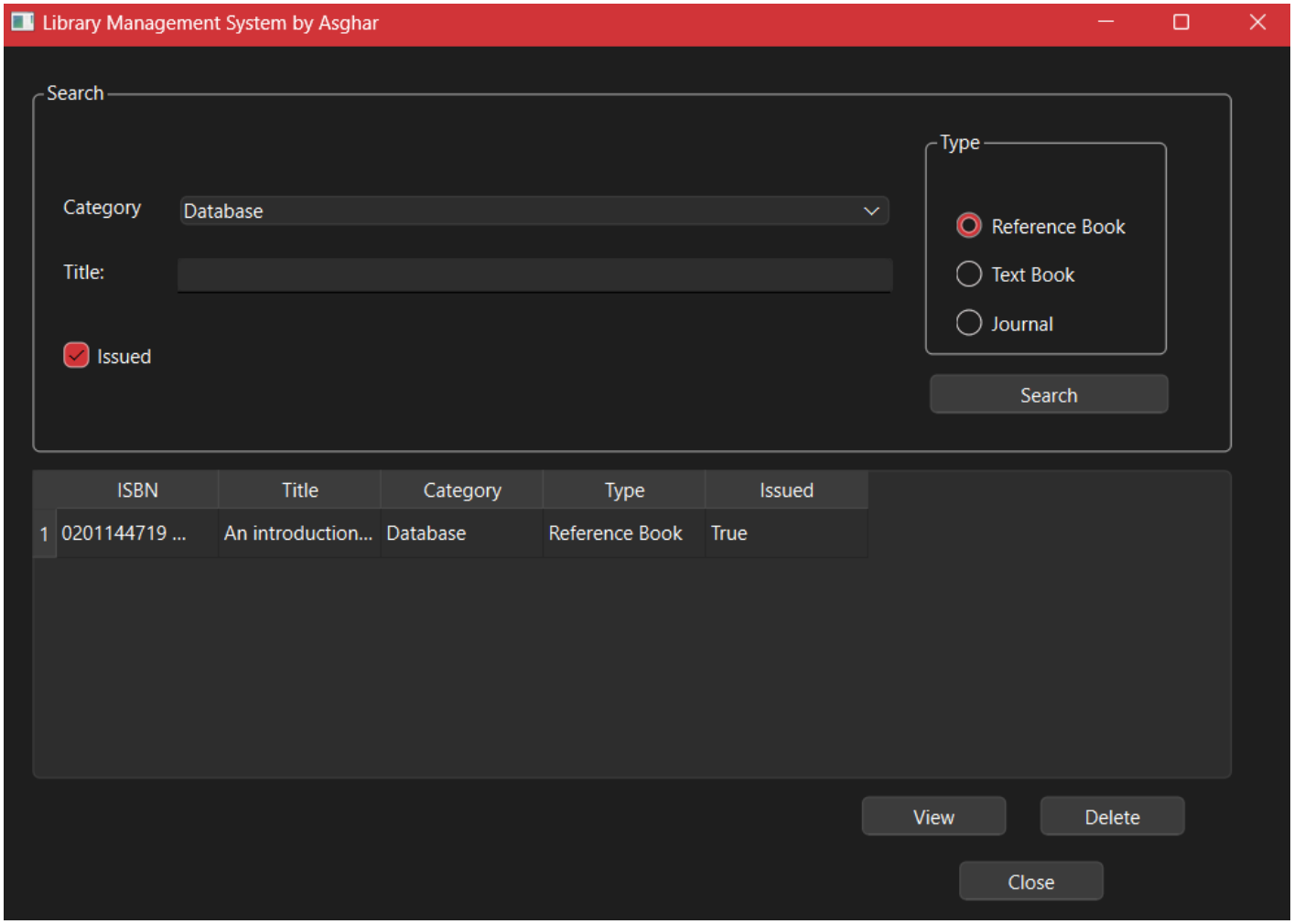

Figure 2: Search Output

4 Deletion of a book entry

The **Delete** button prompts the user to confirm their intention to delete a book through a confirmation window. As shown in Figure 3, if the user selects 'Yes', the book will be removed from the system. Conversely, if the user selects 'No,' the book will remain in the system, and no changes will be made.

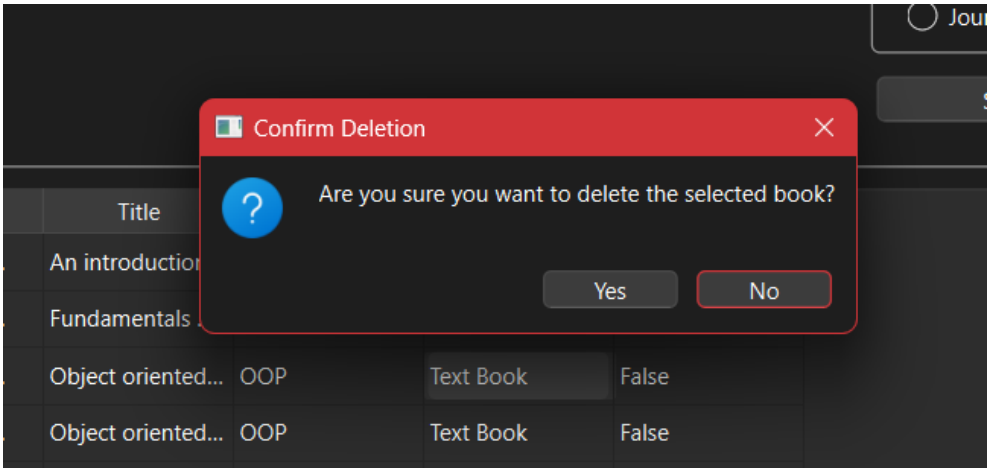

Figure 3: Confirmation Window for Deleting a Book

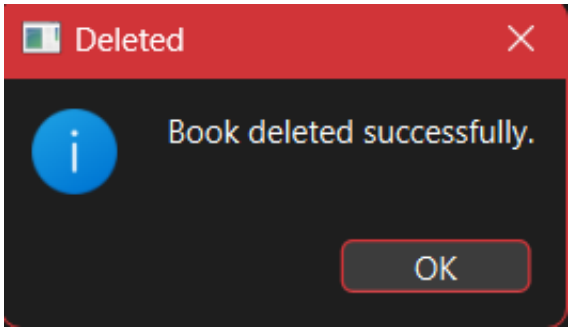

Figure 4: Message Box upon successful deletion

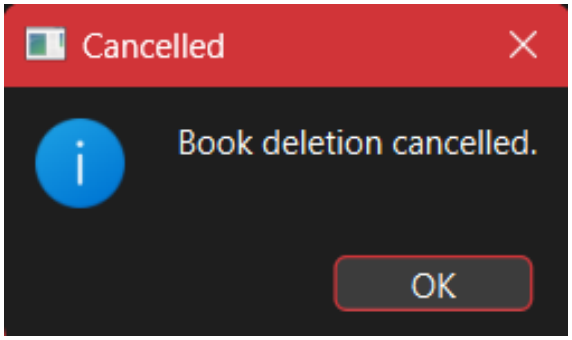

Figure 5: Message Box upon cancelled deletion

5 Search and Deletion Side-case catered

Many students encountered an issue when deleting book entries. If a book was searched and then selected for deletion, the row number shown in the table widget might not match its actual position in the original Python list. For example, a book displayed in row 2 of the table might actually be in row 4 of the list. To resolve this, I didn't reconstruct the table widget with each search. Instead, I "hide" entries during searches. This ensures that the table widget always displays the correct index of entries, regardless of any search operations.

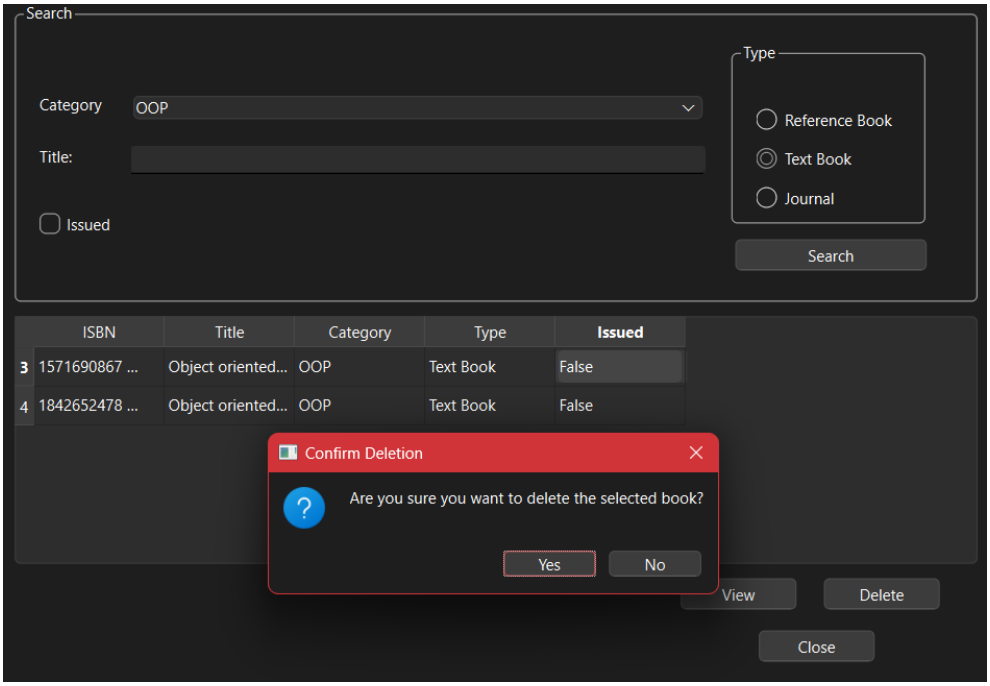

Figure 6: Deleting third entry in the actual list (Table Widget also labels it correctly)

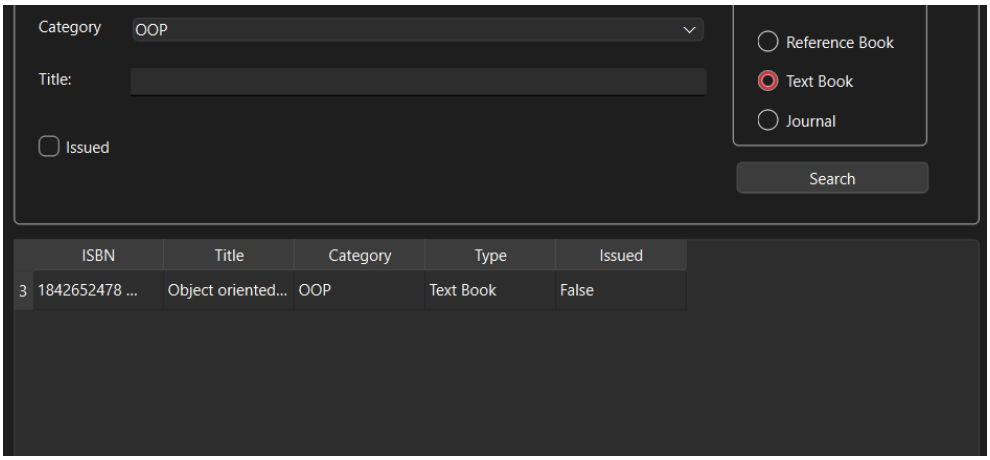

Figure 7: Successful Deletion of Third Entry, The Fourth now has become the Third entry

6 View Book

The **View** button directs the user to the View Book form, where the details of the selected book are displayed. When clicked, the View button will open the View Book form as an overlay on top of the Search Book form. The book details will be shown in a view-only mode, meaning that the user will be able to see the information but will not be able to make any edits.

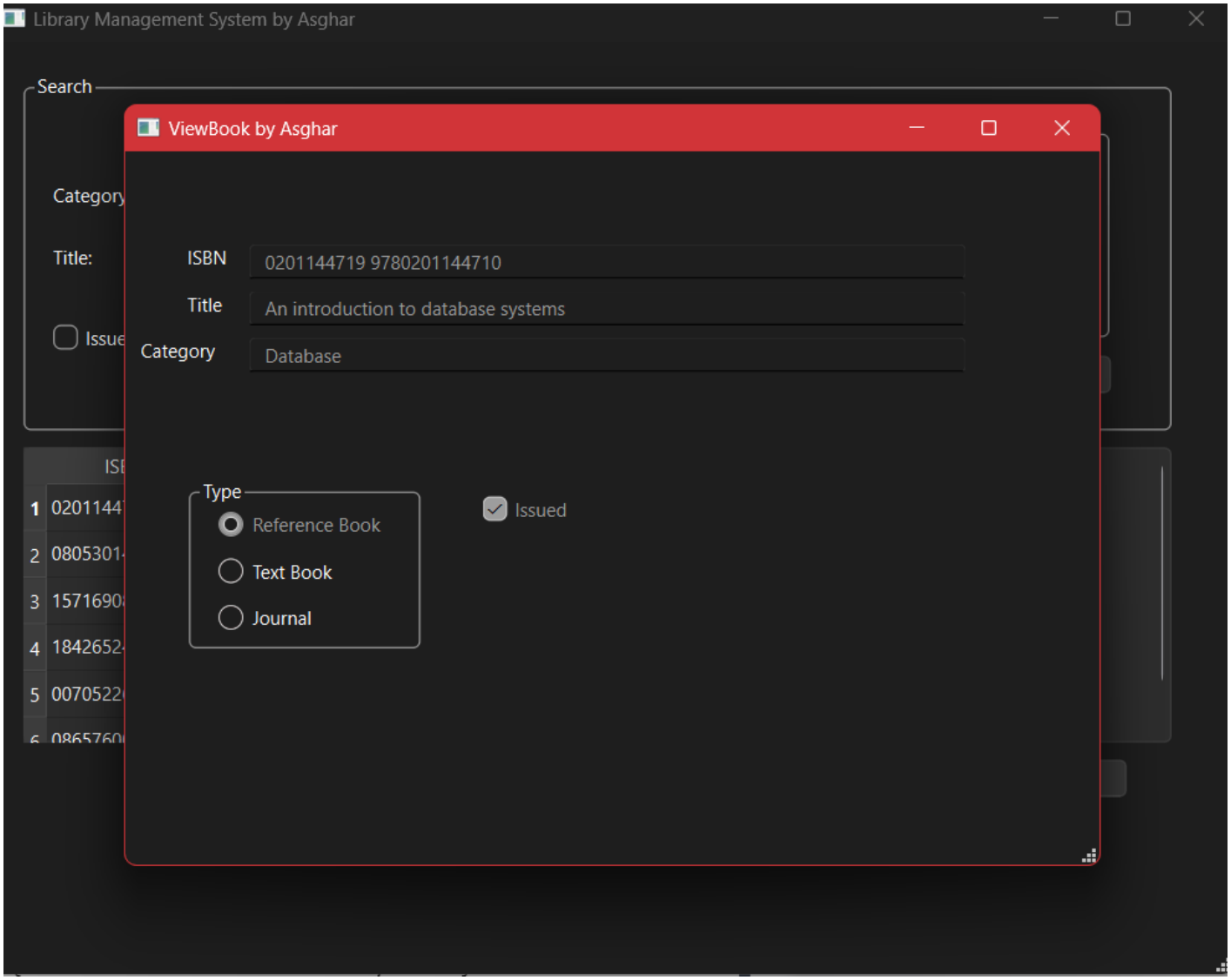

Figure 8: View Book Form in PyQt6

7 Close Button

Clicking on **Close** Button will close and shut down the running application of "Library Management System by Asghar"

8 Code

```
from PyQt6 import QtWidgets, uic
from PyQt6.QtCore import Qt
from PyQt6.QtWidgets import QMessageBox  # Import QMessageBox correctly
import sys

books = [
    ["0201144719 9780201144710", "An introduction to database systems", "Database", "Reference Book", "True"],
    ["0805301453 9780805301458", "Fundamentals of database systems", "Database", "Reference Book", "False"],
    ["1571690867 9781571690869", "Object oriented programming in Java", "OOP", "Text Book", "False"],
    ["1842652478 9781842652473", "Object oriented programming using C++", "OOP", "Text Book", "False"],
    ["0070522618 9780070522619", "Artificial intelligence", "AI", "Journal", "False"],
    ["0865760047 9780865760042", "The Handbook of artificial intelligence", "AI", "Journal", "False"]
]

class UI(QtWidgets.QMainWindow):
    def __init__(self):
        # Call the inherited classes __init__ method
        super(UI, self).__init__()
        # Load the .ui file
        uic.loadUi('Lab02.ui', self)

        # Set up the table with books data
        self.booksTableWidget.setRowCount(len(books))
        for i in range(len(books)):
            for j in range(5):
                item = QtWidgets.QTableWidgetItem(books[i][j])
                # Make the items non-editable
                item.setFlags(Qt.ItemFlag.ItemIsEnabled | Qt.ItemFlag.ItemIsSelectable)
                self.booksTableWidget.setItem(i, j, item)

        #Adding the items to the Category ComboBox
        self.Category_ComboBox.addItem("Database")
        self.Category_ComboBox.addItem("OOP")
        self.Category_ComboBox.addItem("AI")

        # Connect the search function with the search button (to be implemented)
        self.searchButton.clicked.connect(self.search)

        # Connect the view function with the view button (to be implemented)
        self.viewButton.clicked.connect(self.view)
```

```
41
42          # Connect the delete function with the delete button (to be
                implemented)
43          self.deleteButton.clicked.connect(self.delete)
44
45          # Connect the close function with the close button (to be
                implemented)
46          self.closeButton.clicked.connect(self.close)
47
48          #event handling
49          # self.Category_ComboBox.currentIndexChanged.connect(self.
                on_combobox_changed)
50
51      def search(self):
52          # Get search criteria
53          category = self.Category_ComboBox.currentText() #.currentText() is
                typically used with QLineEdit
54          title = self.Title_Lineedit.text() #.text() is typically used with
                QLineEdit
55          issued = self.Issued_checkBox.isChecked()
56
57          # Determine the selected book type based on the radio buttons
58          if self.referenceBookRadioButton.isChecked():
59              book_type = "Reference Book"
60          elif self.textBookRadioButton.isChecked():
61              book_type = "Text Book"
62          elif self.journalRadioButton.isChecked():
63              book_type = "Journal"
64          else:
65              book_type = ""
66
67
68
69          # # Now I will filter the books based on the criteria
70          # filtered_books = []
71          i = -1
72          for book in books:
73              book_isbn = book[0]
74              book_title = book[1]
75              book_category = book[2]
76              book_type_value = book[3]
77              book_issued = book[4]
78              book = [book_isbn, book_title, book_category, book_type_value,
                    book_issued]
79              # print(book)
80              # print(category, book_category, title, book_title, book_type,
                    book_type_value, issued, book_issued)
81
82              i = i + 1
83              if (category == "" or category == book_category) and \
```

```
                (title == "" or title in book_title) and \
                (book_type == "" or book_type == book_type_value) and \
                (not issued or book_issued == "true"):
                self.booksTableWidget.setRowHidden(i, False)
            else:
                self.booksTableWidget.setRowHidden(i, True)


        # print(filtered_books)
        # # Clear the table widget
        # self.booksTableWidget.clearContents()
        # self.booksTableWidget.setRowCount(0)  # Reset row count

        # # Update the table widget with the filtered books
        # self.booksTableWidget.setRowCount(len(filtered_books))
        # for i, book in enumerate(filtered_books):
        #     for j, value in enumerate(book):
        #         item = QtWidgets.QTableWidgetItem(value)
        #         item.setFlags(Qt.ItemFlag.ItemIsEnabled | Qt.ItemFlag.ItemIsSelectable)
        #         self.booksTableWidget.setItem(i, j, item)

    def view(self):
        # TO BE IMPLEMENTED

        # Ensure a row is selected

        selected_row = self.booksTableWidget.currentRow()

        if selected_row == -1:
            QtWidgets.QMessageBox.warning(self, "No Selection", "Please select a book to view.")
            return

        ISBN = books[selected_row][0]
        Title = books[selected_row][1]
        Category = books[selected_row][2]
        Type = books[selected_row][3]
        Issued = books[selected_row][4]

        print(ISBN, Title, Category, Type, Issued)
        self.view_book = ViewBook(ISBN, Title, Category, Type, Issued)
        self.view_book.show()


    def delete(self):
        # selected_rows = self.booksTableWidget.selectedRanges() #Sees which range has been selected
        # if not selected_rows:
```

```
130         #     QtWidgets.QMessageBox.warning(self, "Warning", "No book
                selected.")
131         #     return
132         # selected_row = selected_rows[0].topRow()  # Get the top row of the
                selection, basically tells me the index of the row
133         # print(selected_rows)
134         # print(selected_row)
135
136         selected_row = self.booksTableWidget.currentRow()
137         print(selected_row)
138         # TO BE IMPLEMENTED
139         reply = QtWidgets.QMessageBox.question(self,"Confirm Deletion","Are
                you sure you want to delete the selected book?",
140             QtWidgets.QMessageBox.StandardButton.Yes | QtWidgets.QMessageBox
                    .StandardButton.No,
141             QtWidgets.QMessageBox.StandardButton.No
142         )
143
144         if reply == QtWidgets.QMessageBox.StandardButton.Yes:
145             # Remove the row
146             self.booksTableWidget.removeRow(selected_row)
147             books.pop(selected_row) #deletes from the python list itself so
                    that it won't be detected in the search as well
148             QtWidgets.QMessageBox.information(self, "Deleted", "Book deleted
                     successfully.")
149         else:
150             QtWidgets.QMessageBox.information(self, "Cancelled", "Book
                    deletion cancelled.")
151
152     def close(self):
153         # TO BE IMPLEMENTED
154         self.close()
155
156 class ViewBook(QtWidgets.QMainWindow):
157     def __init__(self, ISBN, Title, Category, Type, Issued):
158         super(ViewBook, self).__init__()
159         uic.loadUi('View Book.ui', self)
160
161         print('HELLLLOOOW WORLD')
162
163         self.ISBN = ISBN
164         self.Title = Title
165         self.Category = Category
166         self.Type = Type
167         self.Issued = Issued
168
169         self.ISBN_LineEdit.setText(self.ISBN)
170         self.ISBN_LineEdit.setDisabled(True)
171
172         self.Title_LineEdit.setText(self.Title)
```

```
        self.Title_LineEdit.setDisabled(True)

        self.Category_LineEdit.setText(self.Category)
        self.Category_LineEdit.setDisabled(True)

        # Compare the string with the three possible values
        if self.Type == "Reference Book":
            self.referenceBookRadioButton.setChecked(True)
            self.referenceBookRadioButton.setDisabled(True)
        elif self.Type == "Text Book":
            self.textBookRadioButton.setChecked(True)
            self.textBookRadioButton.setDisabled(True)
        elif self.Type == "Journal":
            self.journalRadioButton.setChecked(True)
            self.journalRadioButton.setDisabled(True)
        else:
            print("Unknown book type")  # Handle unexpected cases

        if self.Issued == "True":
            self.Issued_checkBox.setChecked(True)
            self.Issued_checkBox.setDisabled(True)


app = QtWidgets.QApplication(sys.argv)  # Create an instance of QtWidgets.
    QApplication
window = UI()  # Create an instance of our UI class
window.show()  # Show the UI
app.exec()  # Start the application
```