Requirements Analysis

1. Develop a POS software package that will **(a)function as a cash register** and **(b)keep an inventory file**:

    a. **"Function as a cash register"**:
        - The program functions as a cash register by allowing the cashier to add and remove books to and from the current transaction. The program will display the subtotal, tax, and total of the transaction, and print a receipt for the transaction.
    b. **"Keep an inventory file"**:
        - The program keeps an inventory file by reading and writing to a plaintext file called "books.txt" located near the executable. The books.txt file is formatted in a way inspired by XML, using "<tag>"s to delimit information about books.

The program will be organized into the following three modules:

2. Cashier Module - The **(a)user enters information for the books being purchased** and the program **(b)calculates the sales tax and total price**. In addition, the **(c)books being purchased are automatically subtracted from the Inventory Database**.
    a. **"User enters information for the books being purchased"**
        - The program will accept the ISBN of the book being purchased as input. The ISBN was chosen as the user input because 1. ISBNs are always unique and also function as the "bar code", or UPC, of a book, and 2. Since ISBNs are printed as barcodes on books, a cashier using a barcode scanner will be able to scan the book into the program without having to type the number manually.
    b. **"Calculates the sales tax and total price"**
        - The program will calculate the sales tax and total price of the current transaction, and will print out a receipt containing that information during checkout.

   c. **“Books being purchased are automatically subtracted from the Inventory Database”**
      - The program will automatically subtract books from the inventory immediately after checkout, without the user having to edit the book quantity manually.
3. Inventory Database Module - There shall be a **(a)file containing a list of all the books in Serendipity’s inventory**. The following **(b)information for each book will be stored in the file**: ISBN, Title, Author, Publisher, Date Added, Quantity-On-Hand, Wholesale Cost, Retail Price. The Inventory Database module will allow the user to **(c)look up information on any book** in the file, **(d)add new books** to the file, **(e)delete books**, and **(f)change any information** in the database.
   a. **File containing a list of all the books in Serendipity’s inventory**
      - The program keeps a .txt file containing every book in Serendipity’s inventory.
   b. **Information for each book will be stored in the file**
      - Within the text file, information for each book is delimited using XML style “<tag>”s, which are then read and parsed by the program into usable form.
   c. **Look up information on any book**
      - The program allows the user to look up information on individual and all books in the inventory using a combination of the Inventory and Report modes.
   d. **Add new books**
      - Using the Inventory mode, users can create new books from within the program by providing the required information (ISBN, title, author, etc.) which is then added to the file automatically. In addition, the program will prevent books with duplicate ISBNs from being added.
   e. **Delete books**
      - Using the Inventory mode, users can delete books from within the program by providing the books ISBN, which is then deleted from the file automatically.
   f. **Change any information**
      - Using the Inventory mode, users can change any attribute of any book in the file by first providing the books ISBN, and any change is saved to the file automatically.

4. Report Module - The Report module with analyze the information in the Inventory Database to produce any of the following reports: **(a)Inventory List, (b)Inventory Wholesale Value, (c)Inventory Retail Value, (d)List By Quantity, (e)List By Cost, (f)List By Age.**
    a. **Inventory List**
        - The program will display a list of all books in the database, including every tracked attribute, in a formatted, easy to read table.
    b. **Inventory Wholesale Value**
        - The program will display a list of all books in the database, including wholesale value, quantity, individual total wholesale value (quantity * wholesale value), and inventory total wholesale value in a formatted, easy to read table.
    c. **Inventory Retail Value**
        - The program will display a list of all books in the database, including retail value, quantity, individual total retail value (quantity * retail value), and inventory total retail value in a formatted, easy to read table.
    d. **List By Quantity**
        - The program will display a list of all books in the database, ordered by quantity-on-hand, starting with the greatest quantity first. This is accomplished internally with a selection sort.
    e. **List By Cost**
        - The program will display a list of all books in the database, ordered by wholesale cost, starting with the greatest cost first. This is accomplished internally with a selection sort.
    f. **List By Age**
        - The program will display a list of all books in the database, ordered by purchase date, starting with the oldest first. This is accomplished internally with a selection sort.
5. Coding Requirements: The demonstration of the following concepts in code: **(a)Main OOP concepts, (b)Friends, (c)Templates, (d)Operator Overloading, (e)Exception Handling**
    a. **Main OOP concepts**
        - The program uses classes to fulfill individual modules roles (Cashier, Inventory, Report), and has inheritance through the Book and InventoryBook classes.
    b. **Friends**

- The program uses the friend declaration in the Book and InventoryBook classes, granting ostream access to the << operator overload.

c. **Templates**
   - The program uses templates in the swap() function in the Report class.

d. **Operator Overloading**
   - The program uses operator overloads in the Book and InventoryBook classes, such as the ++ operator in InventoryBook which increases the quantity of the book by one.

e. **Exception Handling**
   - The program uses exception handling in the getUserInput() functions in main.cpp, where it will catch any exceptions thrown by std::stoi() and std::stod() and give the user an appropriate error message.