Online Bookstore Project

Final Project Report Submission
COMP3005: Database Management Systems
Fall 2021

By: Natalie Jeong (100794762)

TABLE OF CONTENTS

1.0 INTRODUCTION

As outlined by the problem statement provided, the goal of this project was to design and implement an online bookstore 'LookInnaBook' (LIB) that can maintain a database of product information. The application supports functionalities for regular and administrative users, where a collection of books available in the store can be browsed, purchased and managed.

Outlined requirements included the use of a relational database management system to showcase the concepts we've covered as a part of the course. The LIB program takes a web application approach which combines the use of front end templating languages like Pug with backend databases using Postgresql to perform queries following the fundamentals of CRUD.

Hypertext Transfer Protocol (HTTP) and the use of Synchronous/Asynchronous calls between the client and server is used to make the application functional. A combination of AJAX/XML requests, Javascript execution environments (Browsers, Node.js), and JSON combined with RESTful practices allow the application to run as a whole.

2.0 CONCEPTUAL DESIGN

Entity Relation (ER) Diagram

The following is an Entity Relationship Diagram illustrating the structure of the online bookstore database:

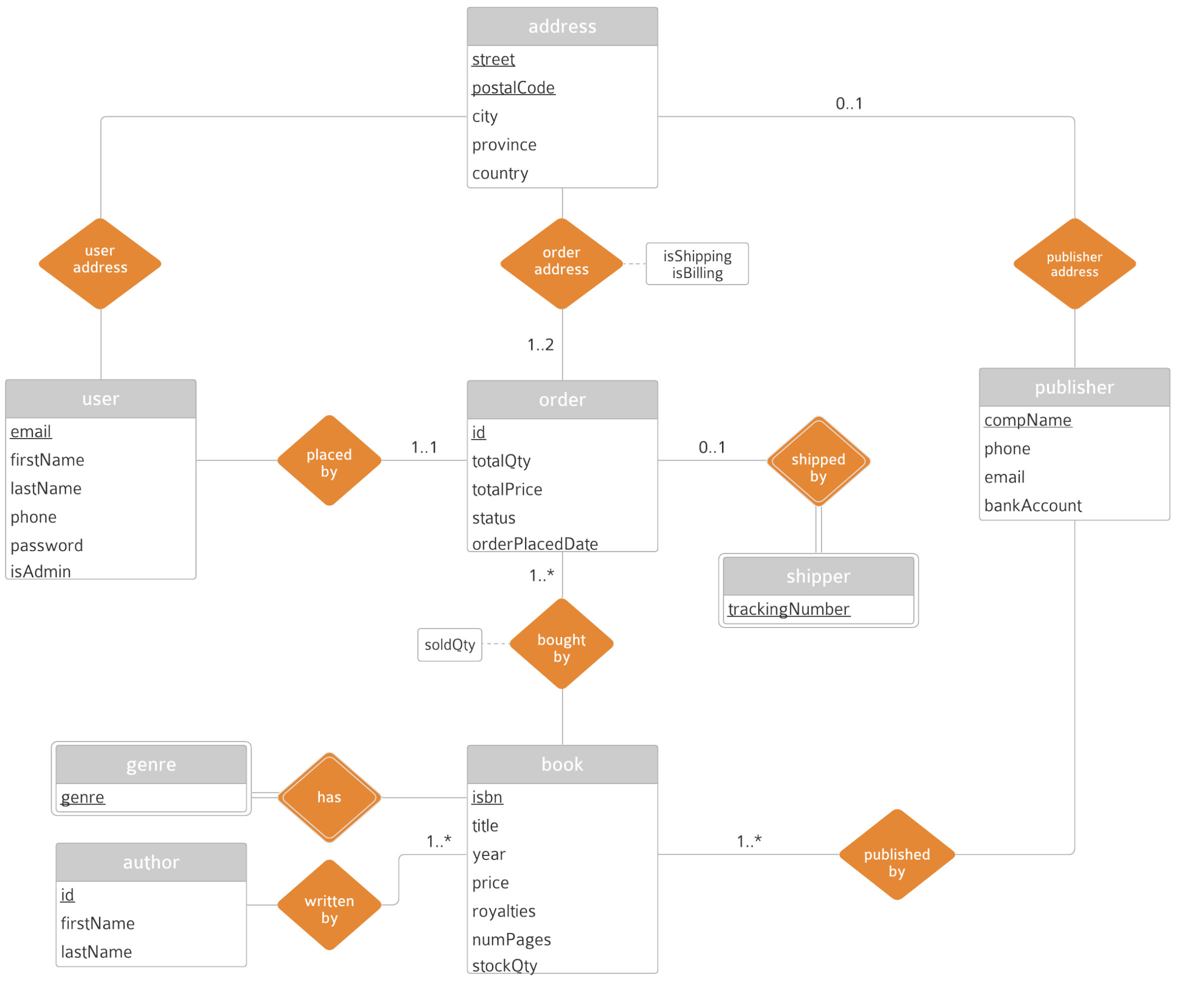


***Figure 1**. Entity Relationship (ER) Diagram used for ‘LookInnaBook’.*

Assumptions

Entities as shown by Figure 1 above are summarized as follows:

Address – The “address” entity represents a physical location, composed of attributes including street, postal code, city, province, and country. Given that no two locations will have the same street and postal code together, they as a tuple are used as the primary keys. It is a strong entity which holds partial participation to three relations: “user address”, “order address” and “publisher address”.

UserAddress – The “userAddress” relation establishes the relationship between a user and their address. In theory, a user can have multiple addresses on their profile, while the same address can be used by multiple users (ie. various family members within the same household). As such, the relation is shown as a many-to-many relationship.

OrderAddress – The “orderAddress” relation establishes the relationship between an order and its address. An order must have a minimum of one address (when shipping and billing addresses are the same), to a maximum of two (when shipping and billing addresses vary). As such, the line between “order” and “order address” is shown with a cardinality constraint of *1..2*. The relation also holds boolean attributes *isShipping* and *isBilling* which tracks whether an associated address is either shipping, billing or both. Furthermore, since multiple orders can be placed under the same address, the relation is shown as a many-to-many relationship.

PublisherAddress – The “publisherAddress” relation establishes the relationship between a publisher and its associated address. In theory, a publisher can have multiple associated addresses (multiple branches/ locations), while an address can only belong to one publisher. As such, the relation is shown as a one-to-many relationship from the “publisher” to the “address”. Note that the cardinality constraint of *0..1* was used between the “address” and “publisher address” because since address is shared amongst “user”, “order” and “publisher” there will be addresses in the database that do not belong to a publisher.

User – The “user” entity represents any user of the bookstore including customers and administrative users. Its attributes include email, first name, last name, phone, password, and a boolean isAdmin value which tracks if the user has an administration status. Given that no two users can have the same email, this attribute is used as the primary key. The “user” is a strong entity which holds partial participation to two relations: “user address” and “placed by”.

placedBy – The “placed by” relation establishes the relationship between a user and their order. Since an order must belong exclusively to one user, the line between “order” and “placed by” is shown with a cardinality constraint of *1..1*. Meaning the participation of “order” in “placed by” is total. Meanwhile, given that multiple orders can be placed by the same user, the relation is shown as a one-to-many relationship from “user” to “order”.

Order – The “order” entity represents a basket of products that have been checked-out and placed. The entity contains a unique identifying order id number, the status, and the order placed date. The primary key is the order id since there are no two orders that can share the same unique id number. It is a strong entity which holds partial participation to “shipped by”, and total participation to three other relations: “order address”, “placed by”, and “bought by”.

shippedBy – The “shipped by” relation establishes the relationship between an order and the shipper. Since an order is not automatically assigned to a tracking number at the time of placing a new order, the line between “order” and “shipped by” is shown with a cardinality constraint of *0..1*. This means that an order can exist within the database without a tracking number. Meanwhile, a tracking number provided by the “shipper” must exclusively belong to one order. Thus the participation of “shipper” in “shipped by” is total as indicated by the double lines. As such, the relation is shown as a one-to-many relationship from “order” to the “shipper”.

Shipper – The “shipper” entity represents a shipping facility that would provide orders with their tracking number. Its only attribute is the tracking number, which thus is also the primary key. Since in the context of the bookstore, a tracking number cannot exist without an order, it is represented as a weak entity which holds total participation to the relation “shipped by”.

boughtBy – The “bought by” relation establishes the relationship between an order and the books bought. An order must have a minimum of one book. As such, the line between “order” and “bought by” is shown with a cardinality constraint of *1..**. The relation also holds the attribute *soldQty* which tracks the number of each books sold in an order. Furthermore, since an order can have multiple books, and books can be placed by multiple orders, the relation is shown as a many-to-many relationship.

Book – The “book” entity represents a book product for sale. It contains the attributes ISBN, title, year, price, royalties, total number of pages, and stock quantity that the store has available. Given that no two books can have the same ISBN, this attribute is used as the primary key. The “book” is a strong entity which holds partial participation to two relations: “bought by” and “has”; and total participation to two other relations: “written by” and “published by”.

publishedBy – The “published by” relation establishes the relationship between a book and the publisher. A book must have a minimum of one publisher. As such, the line between “book” and “published by” is shown with a cardinality constraint of *1..**. Since a publisher can also have multiple books published, the relation is shown as a many-to-many relationship.

Publisher – The “publisher” entity represents a book publishing company. Its attributes are company name, phone, email, and bank account number. Given that no two companies can have the same name, the company name is used as the primary key. The “publisher” is a strong entity which holds partial participation to two relations: “publisher address” and “published by”.

has – The “has” relation establishes the relationship between a book and its genre. A book can have anywhere from zero to many genres. Meanwhile, a genre must have at least one book in order to exist in the system. Thus the participation of “genre” in “has” is total as indicated by the double lines. Furthermore, the relation is shown as a many-to-many relationship.

Genre – The “genre” entity represents possible genres for books in the store. Its only attribute is the genre, which thus is also the primary key. Since in the context of the bookstore, a genre cannot exist without a book, it is represented as a weak entity which holds total participation to the relation “has”.

writtenBy – The “written by” relation establishes the relationship between a book and its author. A book must have a minimum of one author. As such, the line between “book” and “author” is shown with a cardinality constraint of *1..**. Since an author can also have many books written, the relation is shown as a many-to-many relationship.

Author – The “author” entity represents an author of a book. Its attributes include the author’s id, first name and last name. Given that it is possible (though rare) to have an author with the same first and last name, the id is added/ used as the primary key. The “author” is a strong entity which holds partial participation to “written by”.

3.0 REDUCTION TO RELATION SCHEMAS

As a result of the ER-Diagram and the stated assumptions above, the following schema can be extracted:

address(street, postalCode, city, province, country)

orderAddress(o_id, street, postalCode, isShipping, isBilling)

userAddress(email, street, postalCode)

user(email, u_firstName, u_lastName, u_phone, password, isAdmin)

order(o_id, email, totalQty, totalPrice, status, orderPlacedDate)

shipper(o_id, trackingNumber)

book_boughtBy_order(o_id, isbn, soldQty)

book_publishedBy_publisher(isbn, compName)

publisherAddress(street, postalCode, compName)

publisher(compName, p_phone, p_email, p_bankAccount)

book(isbn, title, year, price, royalties, numPages, stockQty)

author(a_id, a_firstName, a_lastName)

book_writtenBy_author(a_id, isbn)

genre(isbn, genre)

Note that the “shipper” and “genre” are weak entities with no descriptive attributes aside from its primary key. And the information presented in the two tables is redundant in its respective relational schema tables. As such, the primary key of the dependent strong entity can be pulled into the weak entity’s schema and the relational schemas (“has” and “shipped_by”) do not need to be present.

Additionally, recall that the relation “placedBy” holds a one-to-many relationship from “user” to “order” where the participation of “order” is total. This means that every entity in the entity set “order” must participate in “placedBy”. As such, the “order” schema can be combined with “placedBy” to form a single schema consisting of the union of attributes of both schemas. Therefore the resulting “order” schema consists of the attributes {o_id, email, totalQty, totalPrice, status, orderPlacedDate}. And the relational schema “placedBy” does not need to be present.

4.0 NORMALIZATION OF RELATION SCHEMAS

Book Schema Relation

The book schema is given by:

$$book(isbn, title, year\ price, royalties, numPages, stockQty)$$

Functional dependencies for books would therefore translate to:

$$book\ = \{\, isbn\ \rightarrow title, year, price, royalties, numPages, stockQty \,\}$$

Note that the title and year of publication are not sufficient to form a nontrivial functional dependency as even the same book published in the same year will have a different and unique ISBN based on their published format (e.g. paperback, hardcover, ebook, audiobook will have a different and unique ISBN).

$$isbn+ = \{\, isbn, title, year, price, royalties, numPages, stockQty \,\}$$

Since the closure on all functional dependencies for the book schema satisfy one of the criteria to be considered in BCNF, namely that for functional dependencies in the form of α →β, that α is a superkey for the schema, therefore the book schema is in good normal form.

Users Schema Relation

The users schema is given by:

$$\text{users(email, u\_firstName, u\_lastName, u\_phone, password, isAdmin)}$$

Functional dependencies for users would therefore translate to:

$$users\ = \{\, email\ \rightarrow title, u\_firstName, u\_lastName, u\_phone, password, isAdmin \,\}$$

Note that a user's first and last names are not sufficient to form a nontrivial functional dependency as it is possible, however rare, for two separate people to have matching first and last names. By using email as the primary key we ensure that there is only one online bookstore account for any valid email address (e.g. a Google gmail address and thus Google account is unique)

$$email + = \{\, email, title, u\_firstName, u\_lastName, u\_phone, password, isAdmin \,\}$$

Since the closure on all functional dependencies for the users schema satisfy one of the criteria to be considered in BCNF, namely that for functional dependencies in the form of α →β, that α is a superkey for the schema, therefore the users schema is in good normal form.

Orders Schema Relation

The orders schema is given by:

$$orders(o\_id, email, status, orderPlacedDate)$$

Functional dependencies for orders would therefore translate to:

$$orders\ = \{\, o\_id\ \rightarrow\ email, status, orderPlacedDate \,\}$$

Note that the email here is not sufficient to form a nontrivial functional dependency since a particular account may place multiple different orders over time.

$$o\_id+ = \{ email, status, orderPlacedDate \}$$

Since the closure on all functional dependencies for the orders schema satisfy one of the criteria to be considered in BCNF, namely that for functional dependencies in the form of α →β, that α is a superkey for the schema, therefore the orders schema is in good normal form.

Author Schema Relation

The author schema is given by:

$$author(a\_id, a\_firstName, a\_lastName)$$

Functional dependencies for author would therefore translate to:

$$author = \{ a\_id \rightarrow a\_firstName, a\_lastName \}$$

Note that an author's first and last names are not sufficient to form a nontrivial functional dependency as two authors, however rare, may share the same first and last names.

$$a\_id+ = \{ a\_id, a\_firstName, a\_lastName \}$$

Since the closure on all functional dependencies for the author schema satisfy one of the criteria to be considered in BCNF, namely that for functional dependencies in the form of α →β, that α is a superkey for the schema, therefore the author schema is in good normal form.

Address Schema Relation

The address schema is given by:

$$address(street, postalCode, city, province, country)$$

Functional dependencies for address would therefore translate to:

$$address = \{ street, postalCode \rightarrow city, province, country \}$$

Note that postal codes are not unique globally. It is also not unique along one side of a city block (street numbers needed together with postal code for a unique identifier).

$$(street, postalCode)+ = \{ street, postalCode, city, province, country \}$$

Since the closure on all functional dependencies for the address schema satisfy one of the criteria to be considered in BCNF, namely that for functional dependencies in the form of α →β, that α is a superkey for the schema, therefore the address schema is in good normal form.

Publisher Schema Relation

The publisher schema is given by:

$$publisher(compName, p\_phone, p\_email, p\_bankAccount)$$

Functional dependencies for publisher would therefore translate to:

$$publisher = \{ compName \rightarrow p\_phone, p\_email, p\_bankAccount,$$

$$p\_bankaccount \rightarrow \ compName,$$

$$p\_email \ \rightarrow \ compName \}$$

Note: Assumed that no two publishers have the same corporate name (incorporated names with legal ending are usually protected by law).

$$(compName)+ = \{ compName, p\_phone, p\_email, p\_bankAccount \}$$

$$(p\_bankAccount)+ = \{ compName, p\_phone, p\_email, p\_bankAccount \}$$

$$(p\_email)+ = \{ compName, p\_phone, p\_email, p\_bankAccount \}$$

Since the closure on all functional dependencies for the publisher schema satisfy one of the criteria to be considered in BCNF, namely that for functional dependencies in the form of α →β, that α is a superkey for the schema, therefore the publisher schema is in good normal form.

PublisherAddress Schema Relation

The publisherAddress schema is given by:

$$publisherAddress(street, postalCode, compName \ )$$

Functional dependencies for publisherAdress would therefore translate to:

$$publisherAddress \ = \{ street, postalCode \ \rightarrow \ compName \}$$

Note that we are making the assumption that only the address of the publisher headquarters will be needed.

$$(street, postalCode)+ = \{ street, postalCode, compName \}$$

Since the closure on all functional dependencies for the publisherAddress schema satisfy one of the criteria to be considered in BCNF, namely that for functional dependencies in the form of α →β, that α is a superkey for the schema, therefore the publisherAddress schema is in good normal form.

Book_writtenBy_author Schema Relation

The book_writtenBy_author schema is given by:

$$book\_writtenBy\_author(isbn, a\_id \ )$$

Functional dependencies for book_writtenBy_author would therefore translate to:

$$book\_writtenBy\_author \ = \{ isbn, a\_id \ \rightarrow \ isbn, a\_id \}$$

Note that both attributes are needed as a book may be written by more than one author and an author may write more than one book.

$$(isbn, a\_id)+ = \{ isbn, a\_id \}$$

Since the closure on all functional dependencies for the book_writtenBy_author schema satisfy one of the criteria to be considered in BCNF, namely that α →β is a trivial dependency, therefore the book_writtenBy_author schema is in good normal form.

Genre Schema Relation

The genre schema is given by:

$$genre(isbn, genre\ )$$

Functional dependencies for genre would therefore translate to:

$$genre\ \ = \{\, isbn, genre\ \ \rightarrow\ \ isbn, genre\ \}$$

Note that both attributes are needed as a book may have more than one genre and a particular genre may apply to more than one book.

$$(isbn, genre)+ = \{\, isbn, genre\ \}$$

Since the closure on all functional dependencies for the genre schema satisfy one of the criteria to be considered in BCNF, namely that α →β is a trivial dependency, therefore the genre schema is in good normal form.

Shipper Schema Relation

The shipper schema is given by:

$$shipper(o\_id, trackingNumber\ )$$

Functional dependencies for shipper would therefore translate to:

$$shipper\ \ = \{\, o\_id\ \ \rightarrow\ \ trackingNumber\ \}$$

Note that every order is assumed to have a unique trackingNumber given the assumption stated in the project problem statement, specifically that, there will be no multiple order numbers for multiple books shipped from multiple warehouses.

$$(o\_id)+ = \{\, o\_id, trackingNumber\ \}$$

Since the closure on all functional dependencies for the shipper schema satisfy one of the criteria to be considered in BCNF, namely that for functional dependencies in the form of α →β, that α is a superkey for the schema, therefore the shipper schema is in good normal form.

Book_boughtBy_order Schema Relation

The book_boughtBy_order schema is given by:

$$book\_boughtBy\_order(o\_id, isbn, soldQty)$$

Functional dependencies for book_boughtBy_order would therefore translate to:

$$book\_boughtBy\_order\ \ = \{\, o\_id, isbn\ \ \rightarrow\ \ soldQty\ \}$$

Note that both attributes are needed as for a given order, multiple different books may be ordered.

$$(o\_id, isbn)+ = \{\, o\_id, isbn, soldQty\ \}$$

Since the closure on all functional dependencies for the book_boughtBy_order schema satisfy one of the criteria to be considered in BCNF, namely that for functional dependencies in the form of α →β, that α is a superkey for the schema, therefore the book_boughtBy_order schema is in good normal form.

Book_publishedBy_publisher Schema Relation

The book_publishedBy_publisher schema is given by:

$$\mathrm{book\_publishedBy\_publisher}(\mathrm{isbn}, \mathrm{compName})$$

Functional dependencies for book_publishedBy_publisher would therefore translate to:

$$\mathrm{book\_publishedBy\_publisher} = \{ \mathrm{isbn} \rightarrow \mathrm{compName} \}$$

Note that isbn are unique to their format and publisher.

$$(isbn)+ = \{ isbn, compName \}$$

Since the closure on all functional dependencies for the book_publishedBy_publisher schema satisfy one of the criteria to be considered in BCNF, namely that for functional dependencies in the form of α →β, that α is a superkey for the schema, therefore the book_publishedBy_publisher schema is in good normal form.

OrderAddress Schema Relation

The orderAddress schema is given by:

$$orderAddress(o\_id, street, postalCode, isShipping, isBilling)$$

Functional dependencies for orderAddress would therefore translate to:

$$orderAddress = \{ o\_id, street, postalCode \rightarrow isShipping, isBilling \}$$

Note that an order may have multiple associated addresses for (e.g. shipping and billing), street numbers, street names and postal codes are not unique globally.

$$(o\_id, street, postalCode)+ = \{ o\_id, street, postalCode, isShipping, isBilling \}$$

Since the closure on all functional dependencies for the orderAddress schema satisfy one of the criteria to be considered in BCNF, namely that for functional dependencies in the form of α →β, that α is a superkey for the schema, therefore the orderAddress schema is in good normal form.

userAddress Schema Relation

The userAddress schema is given by:

$$userAddress(email, street, postalCode)$$

Functional dependencies for userAddress would therefore translate to:

$$userAddress = \{ email, street, postalCode \rightarrow email, street, postalCode \}$$

Note that a user may have multiple associated addresses on their accounts. Additionally, multiple users within the same household will have the same address. The address on their profile may also differ from the order address.

$$(email, street, postalCode)+ = \{ email, street, postalCode \}$$

Since the closure on all functional dependencies for the userAddress schema satisfy one of the criteria to be considered in BCNF, namely that α →β is a trivial dependency, therefore the userAddress schema is in good normal form.

5.0 DATABASE SCHEMA DIAGRAM

The final schema diagram for the database is as follows:

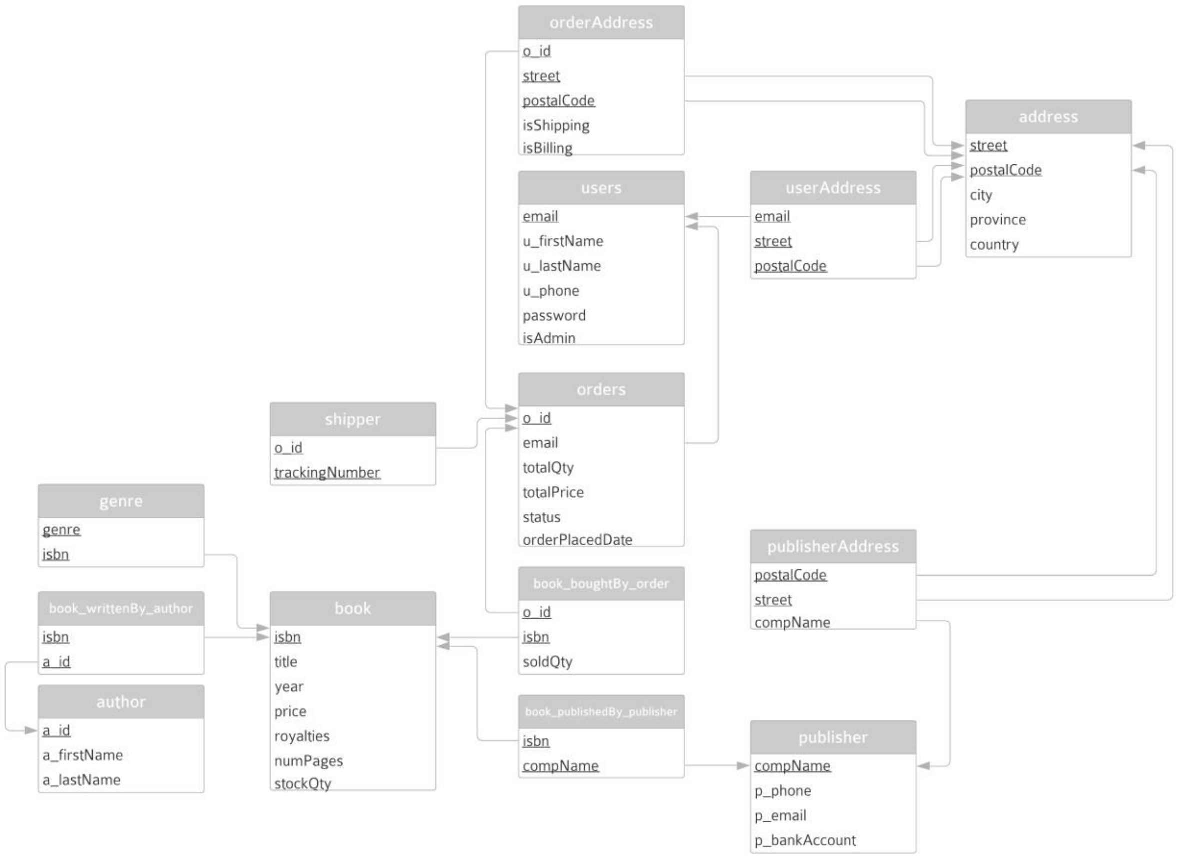


Figure 2. *Schema Diagram for 'LookInnaBook'.*

6.0 IMPLEMENTATION

FUNCTIONALITY REQUIREMENTS

User

The required basic functionalities of the user include:

- Browse a collection of books that are available in the bookstore
- Search the bookstore by book name, author name, isbn, genre
- View a selected book's information including author(s), genre, publisher, number of pages, price
- Add many books to the shopping cart
- Checkout the cart as a registered user
- Insert billing and shipping address at the time of checkout to complete an order
- Track an order using an order number

Administrator

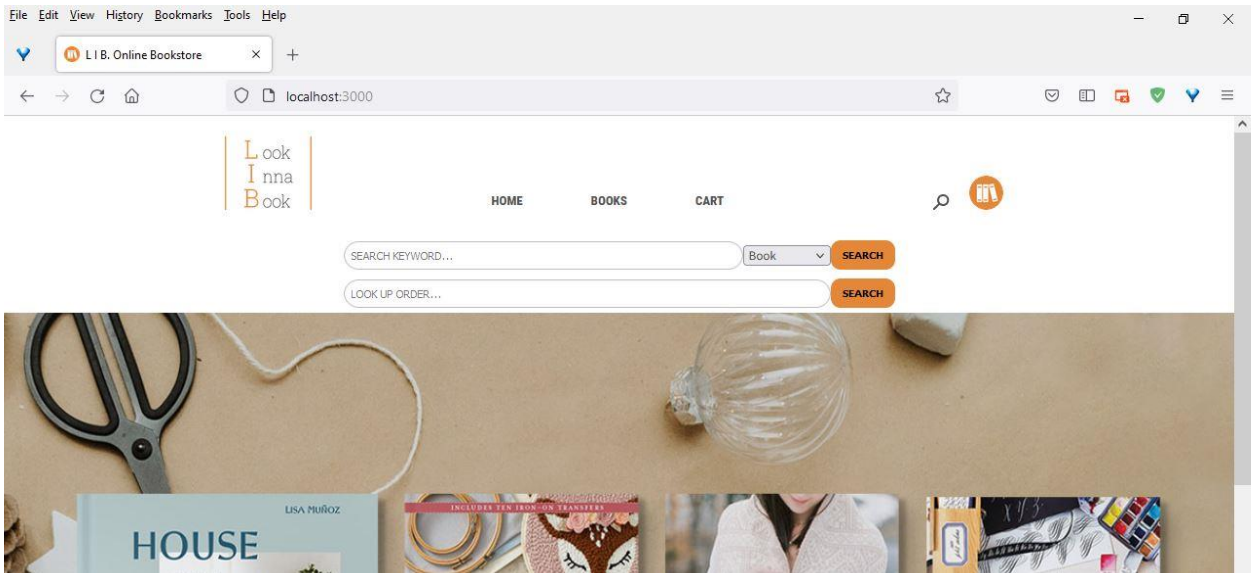
The required basic functionalities of the administrator include:

- Add new books to the bookstore collection
- Remove books from the bookstore
- Generate reports that show sales vs. expenditures, sales per genre, sales per author
- Place orders for new books given a threshold

IMPLEMENTATION

Home Page

When first starting up the website, the page will land on the *home* screen.

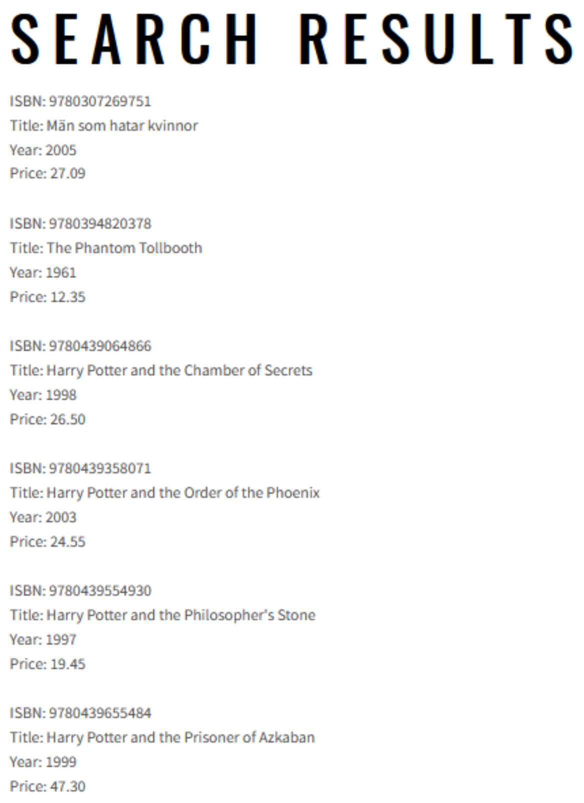


The header bar has several buttons that allow for basic navigation including:

- Home links to “/”, the homepage
- Book links to “/books”, the book page
- Cart links to “/cart”, the cart page
- The search icon links to “/”, the home page which has the search feature integrated
- The profile icon opens up links to:
    - The login or sign up pages (if not logged in)
    - The logged-in user's profile page or the logout pages (if logged in)

Search Books

A guest, user or admin can access the search functionalities of the database by the two search bars on the home page. Note that the keyword query will search for all books with the collection using the attributes of the selected table in the dropdown menu. This includes: *book*, *author*, *genre* and *publisher*. The keyword search is also approximate and case-insensitive hence searching “ha” will return the following results (books that contain “ha” in string fields).



SEARCH RESULTS

ISBN: 9780307269751
Title: Män som hatar kvinnor
Year: 2005
Price: 27.09

ISBN: 9780394820378
Title: The Phantom Tollbooth
Year: 1961
Price: 12.35

ISBN: 9780439064866
Title: Harry Potter and the Chamber of Secrets
Year: 1998
Price: 26.50

ISBN: 9780439358071
Title: Harry Potter and the Order of the Phoenix
Year: 2003
Price: 24.55

ISBN: 9780439554930
Title: Harry Potter and the Philosopher's Stone
Year: 1997
Price: 19.45

ISBN: 9780439655484
Title: Harry Potter and the Prisoner of Azkaban
Year: 1999
Price: 47.30

The results are clickable so that when a book is found, a user can select a specific item to then be navigated to the book’s view page. From here, the product can be added to their cart.

Similarly, the order lookup field takes in an integer and verifies if the order exists. If successful, a user is taken to the corresponding order page which displays the item’s tracking and delivery information.

By having a front end implementation, the application is able to perform basic checks (such as input types, regex requirements) before sending/ requesting a call from the back end.

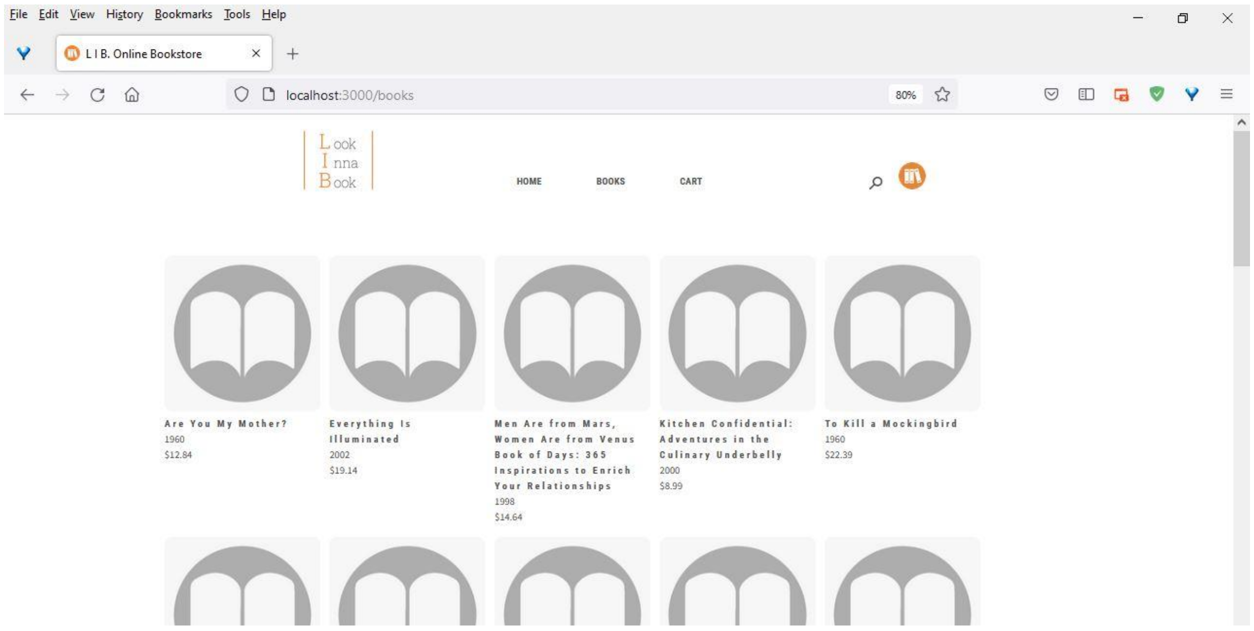
ORDER NUMBER 20

Order Placed On: 2019-12-28
Status: Order Delivered
Tracking Info: VTK22CRM8HS

Current Status:
Processed at transit facility.
77 King St W M5K 1A2 Toronto Ontario Canada

Books Page

If a user is not looking for a specific book, they can browse the *books* page which outlines all the collection of books available within the store.



Are You My Mother?
1960
$12.84

Everything Is Illuminated
2002
$19.14

Men Are from Mars, Women Are from Venus Book of Days: 365 Inspirations to Enrich Your Relationships
1998
$14.64

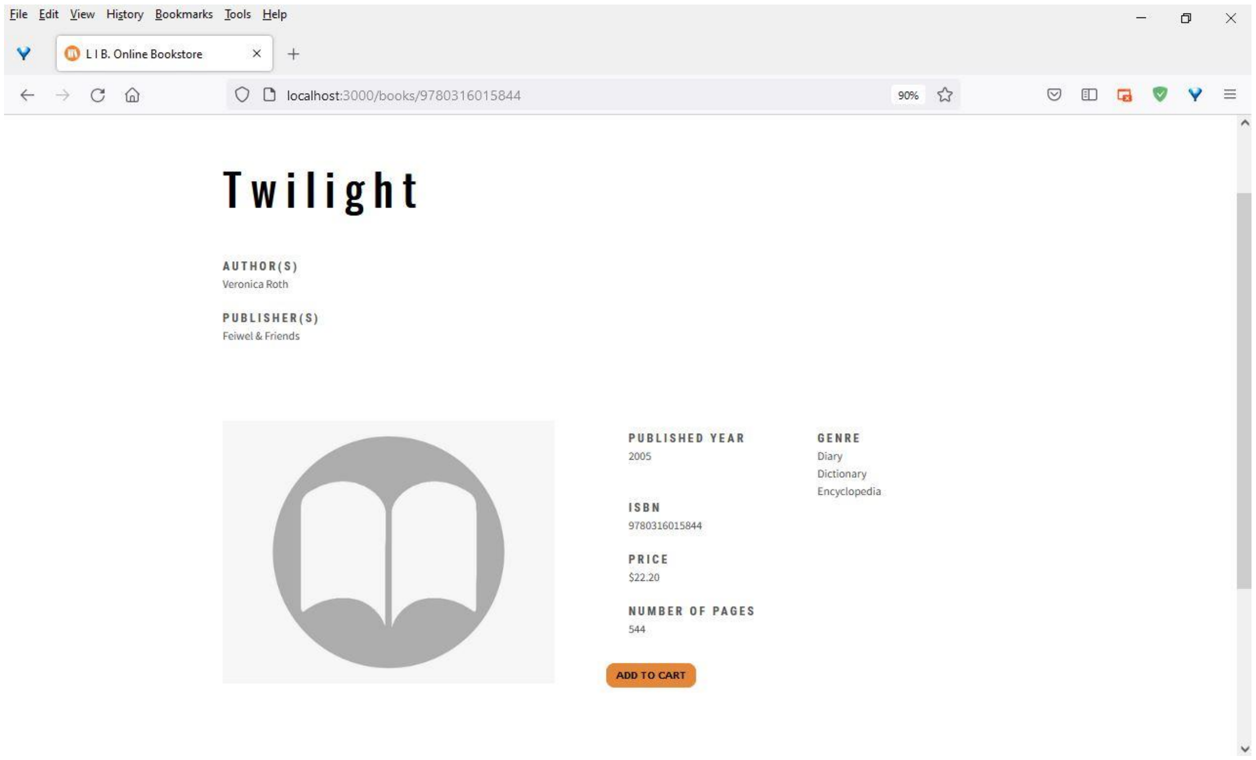
Kitchen Confidential: Adventures in the Culinary Underbelly
2000
$8.99

To Kill a Mockingbird
1960
$22.39

Each icon represents a row inside the book table within the database, and displays the product's title, year and price. A user can also select a specific book, at which point a redirect will occur to a view a page for the clicked product.

It is here where the full details of a book is shown including the

- Author(s)
- Publisher(s)
- Year
- Genre
- ISBN
- Price
- Number of Pages


File Edit View History Bookmarks Tools Help
L I B. Online Bookstore
localhost:3000/books/9780316015844
90%
ADD TO CART

Twilight

AUTHOR(S)
Veronica Roth

PUBLISHER(S)
Feiwel & Friends

PUBLISHED YEAR
2005

GENRE
Diary
Dictionary
Encyclopedia

ISBN
9780316015844

PRICE
$22.20

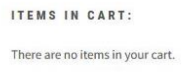
NUMBER OF PAGES
544

A guest/ user will also be presented with the option to add items to cart using the "Add to Cart" button for each book on the view page.

Cart Page

The *cart* page allows a user to view what items have been added to their shopping basket. Given the restriction that a user must be registered with the store in order to checkout, the front end integrates the use of cookies/ session to control the flow.

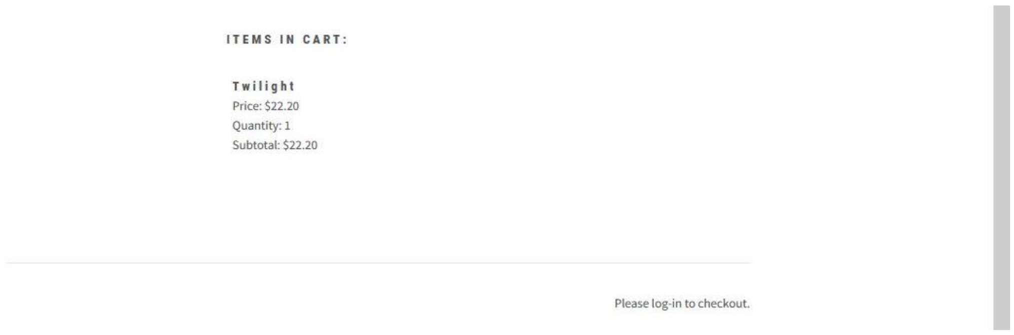
Hence by default, if there are no items in the cart the page will show



ITEMS IN CART:

There are no items in your cart.

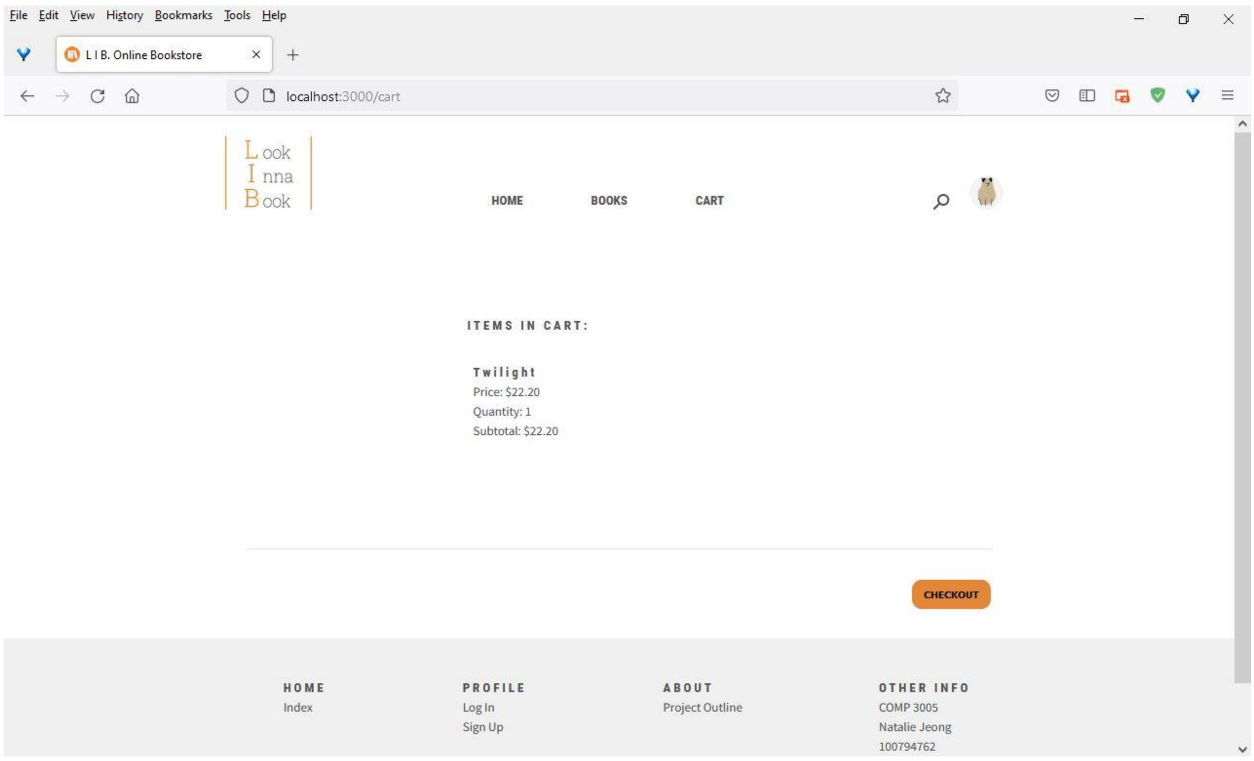
If there are items in the cart but you are not signed-in, the view will restrict the possibility to proceed and ask for the guest to log-in


ITEMS IN CART:

Twilight
Price: $22.20
Quantity: 1
Subtotal: $22.20

Please log-in to checkout.

If there are items in the cart and you are logged-in, the view will present the option to proceed by a checkout button on the screen



ITEMS IN CART:

Twilight
Price: $22.20
Quantity: 1
Subtotal: $22.20

HOME
Index

PROFILE
Log In
Sign Up

ABOUT
Project Outline

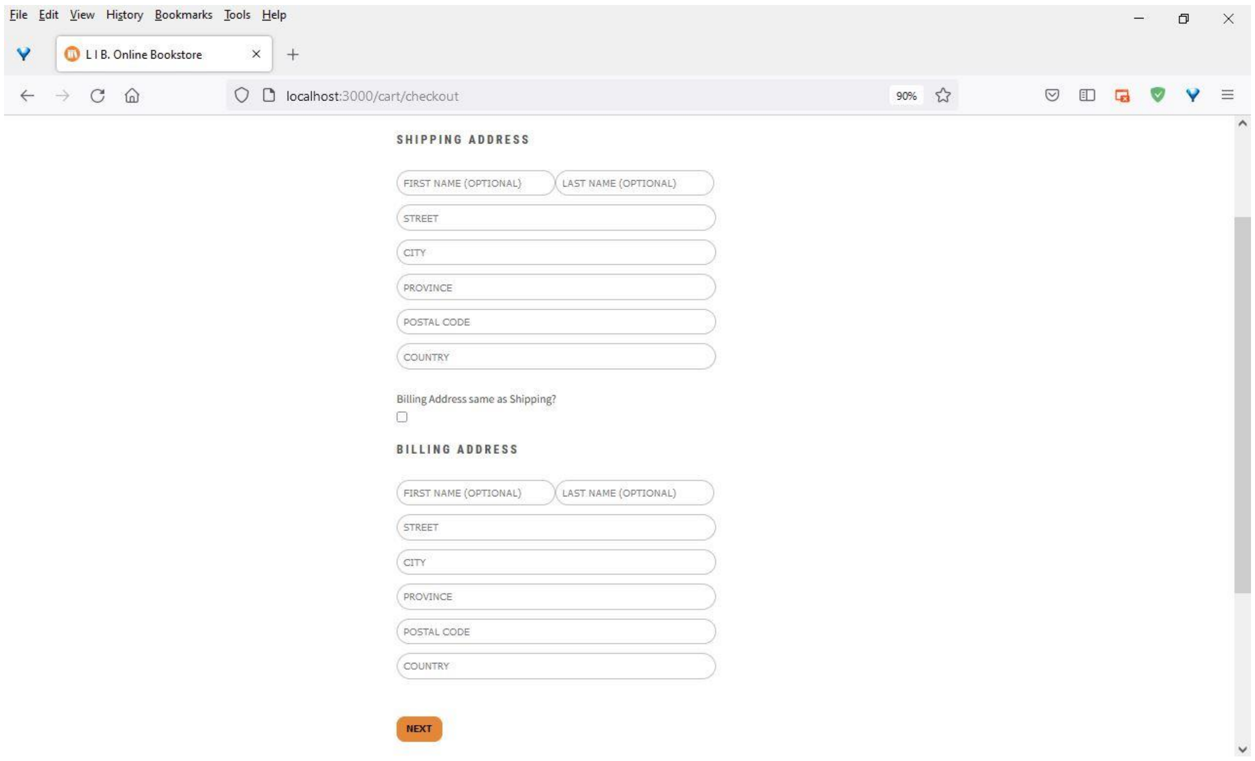
OTHER INFO
COMP 3005
Natalie Jeong
100794762

Therefore the verification that the user is registered in the database needs to be completed only once at the time of log-in.

Checkout

Upon clicking the checkout button, the registered user is faced with a forum to input their shipping and billing address information.

If the two addresses are the same, a user can select the checkbox option labelled "Billing Address same as Shipping?". However, if they differ, then both sections must be completed.


File Edit View History Bookmarks Tools Help
L I B. Online Bookstore
localhost:3000/cart/checkout
90%

SHIPPING ADDRESS

FIRST NAME (OPTIONAL) LAST NAME (OPTIONAL)
STREET
CITY
PROVINCE
POSTAL CODE
COUNTRY

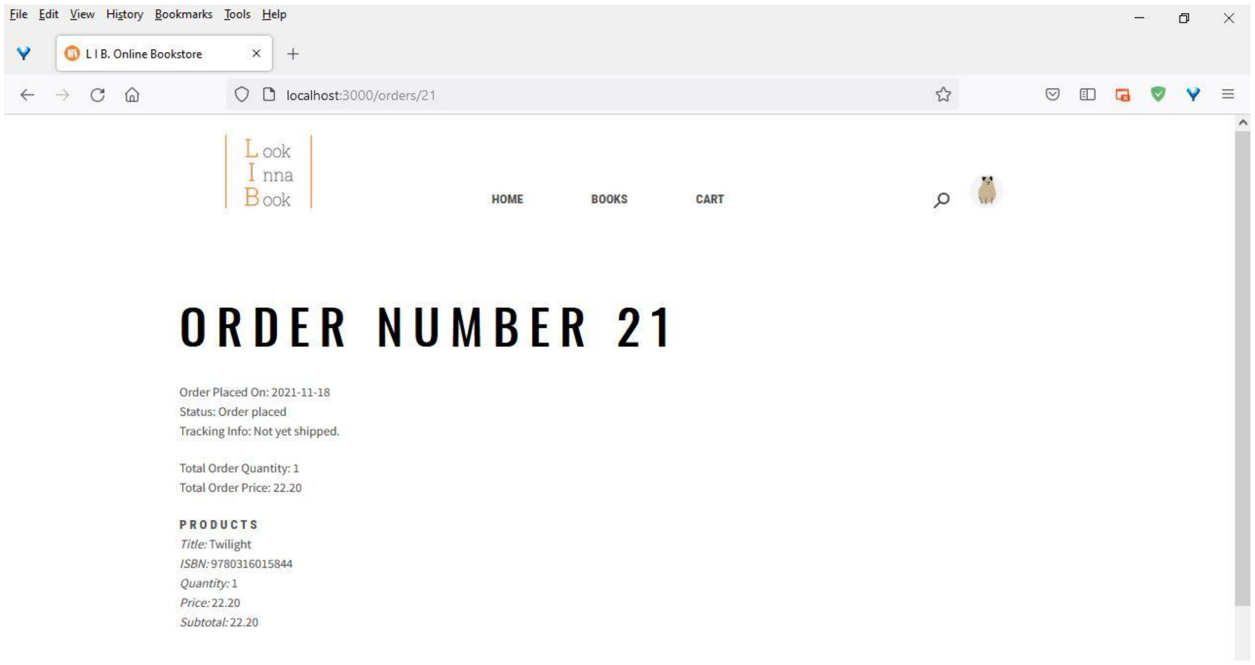
Billing Address same as Shipping?
☐

BILLING ADDRESS

FIRST NAME (OPTIONAL) LAST NAME (OPTIONAL)
STREET
CITY
PROVINCE
POSTAL CODE
COUNTRY

NEXT

After all fields are complete, a user must select "Next" in order to complete the order. If successful, the user will land on the orders view page for the new completed order



ORDER NUMBER 21

Order Placed On: 2021-11-18
Status: Order placed
Tracking Info: Not yet shipped.

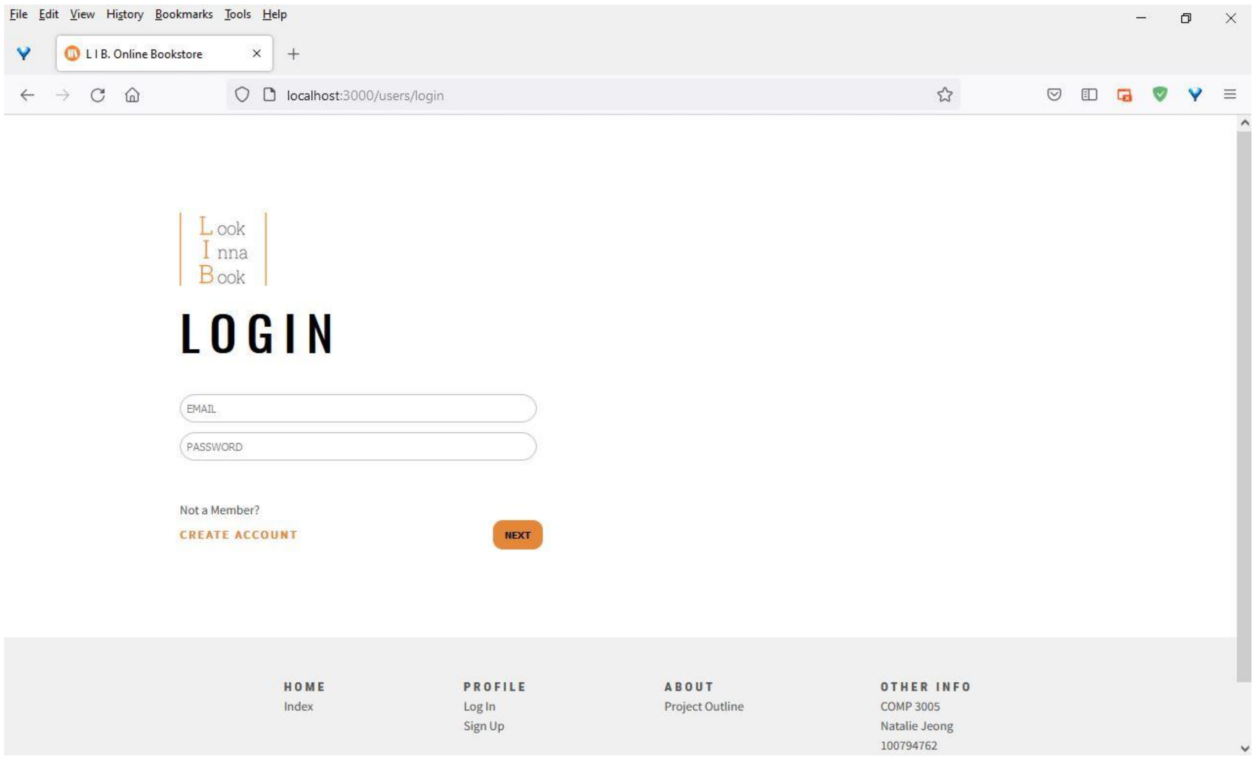
Total Order Quantity: 1
Total Order Price: 22.20

PRODUCTS
Title: Twilight
ISBN: 9780316015844
Quantity: 1
Price: 22.20
Subtotal: 22.20

Also note that when completing an order, the tracking status and number is set to "Order placed" and "Not yet shipped", respectively. This is to emulate the scenario of what would occur in on a live functional website where an order exists but the product has not been shipped yet.

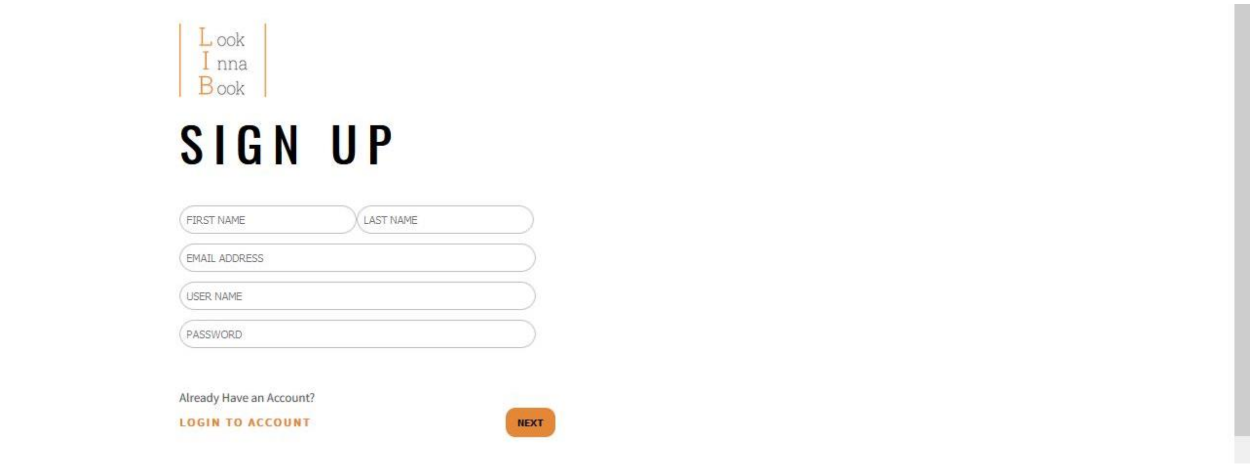
Sign-In/ Sign-Out

In order to log-in/ log-out of the website, a user can click on the profile icon on the top dropdown menu and select “Login”.



If validation against the database is correct, the user will be authenticated and the page will be redirected to the user’s ‘view’ page with their own profile information. Otherwise, an error message will show below the input field to communicate what went wrong.

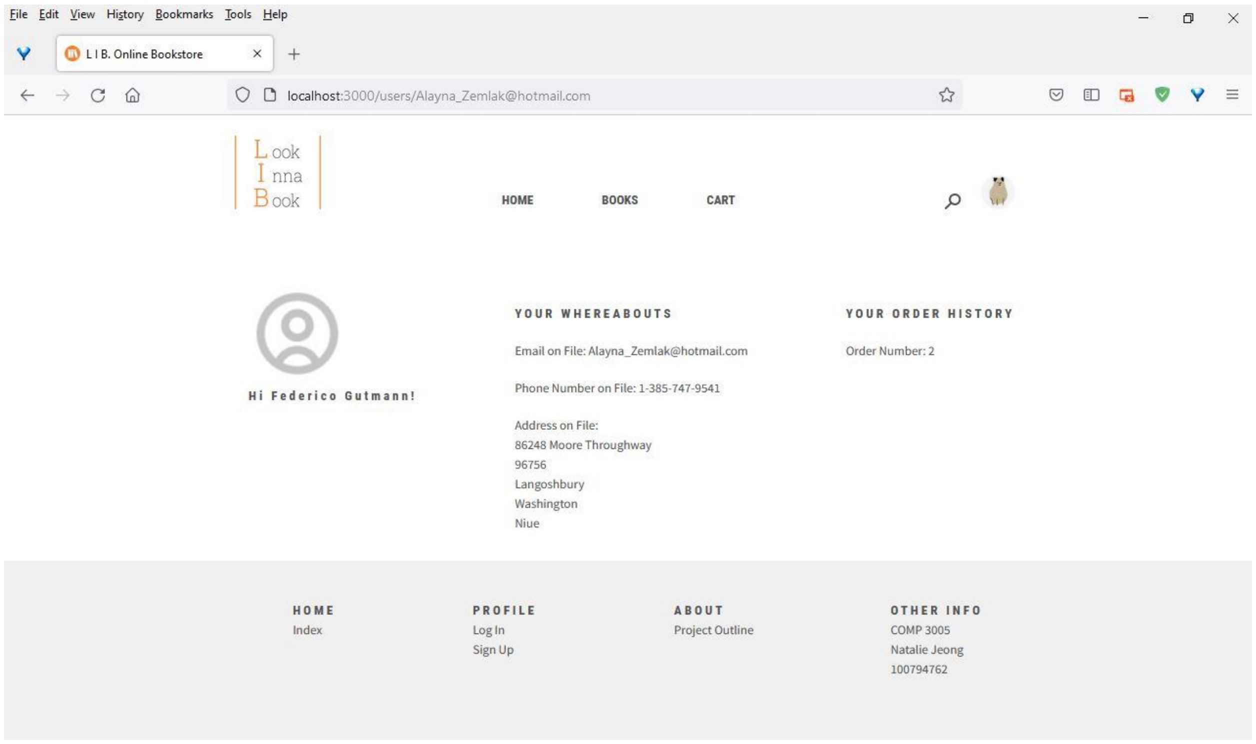
In the case a user does not have an account, the ‘Create Account’ link on the page can be clicked to switch between signup/login mode.



Note that when a user is signed-in, the dropdown menu will change to show different menu options: “My Profile” and “Logout” (in lieu of “Login” and “Signup”).

User Page

Upon log-in, a user is directed to their user profile where details of their information can be viewed including their email address, phone number and address on file. It also has a section for their order history that shows orders previously and currently placed.


File Edit View History Bookmarks Tools Help
L I B. Online Bookstore
localhost:3000/users/Alayna_Zemlak@hotmail.com
Look Inna Book
HOME
BOOKS
CART

Hi Federico Gutmann!

YOUR WHEREABOUTS

Email on File: Alayna_Zemlak@hotmail.com

Phone Number on File: 1-385-747-9541

Address on File:
86248 Moore Throughway
96756
Langoshbury
Washington
Niue

YOUR ORDER HISTORY

Order Number: 2

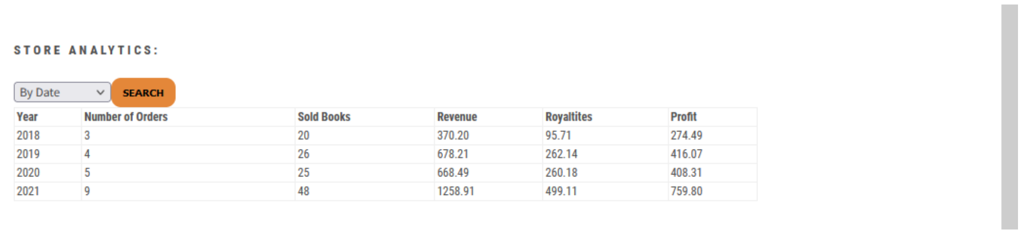
HOME	PROFILE	ABOUT	OTHER INFO
Index	Log In	Project Outline	COMP 3005
	Sign Up		Natalie Jeong
			100794762

Again, the orders are made clickable so that if a user wants to view further details about a specific order (such as tracking information) it can be easily accessed.

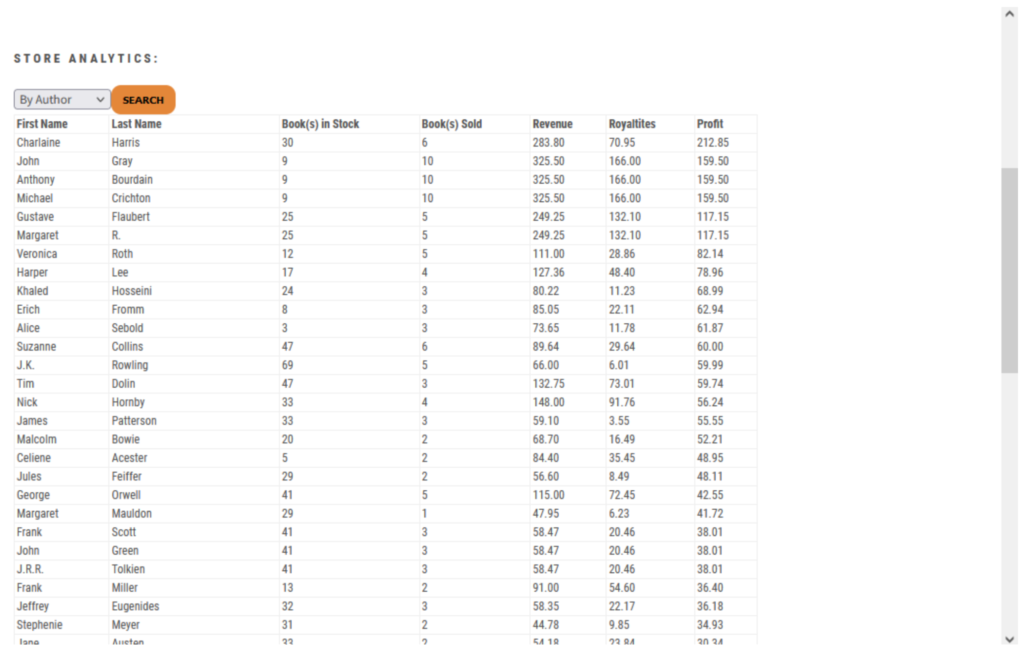
Sales Reports (Admin Only)

With the combination of cookies/ sessions and the *isAdmin* boolean value on the user, the application is able to render an additional component on the user profile page, if the user happens to be an administrator.

The store analytics section allows admins to generate and view information about the sales over a period of time. The report provides information about the sales, revenue, royalties and profit on the store to date. Using the dropdown menu, an admin can also select to view analysis by *Date*, *Author*, *Genre* or *Publisher*.



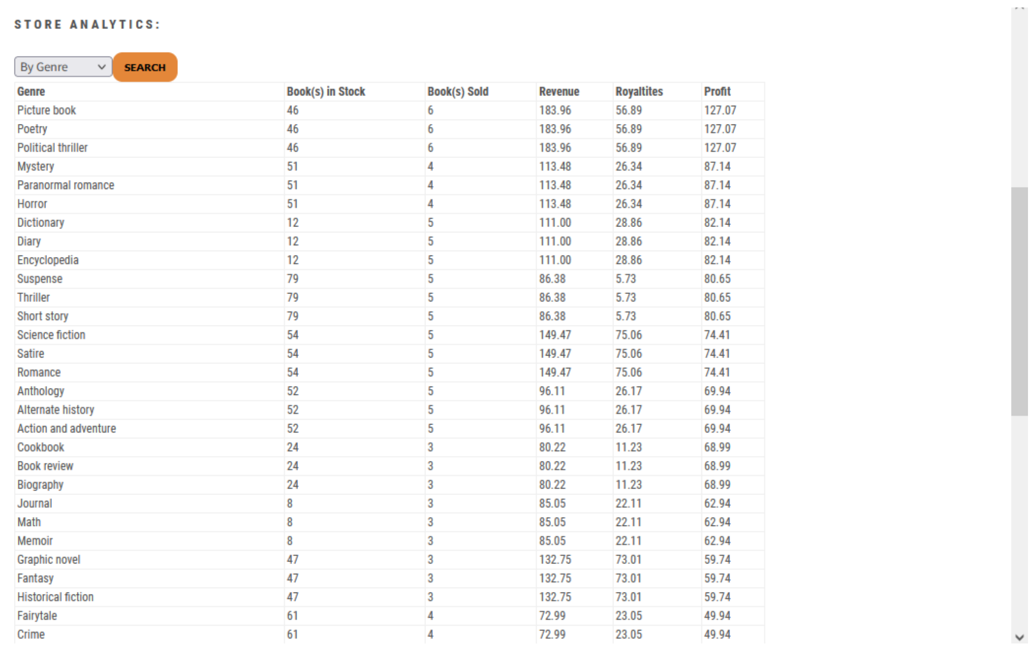
Year	Number of Orders	Sold Books	Revenue	Royaltites	Profit
2018	3	20	370.20	95.71	274.49
2019	4	26	678.21	262.14	416.07
2020	5	25	668.49	260.18	408.31
2021	9	48	1258.91	499.11	759.80



STORE ANALYTICS:

By Author — SEARCH

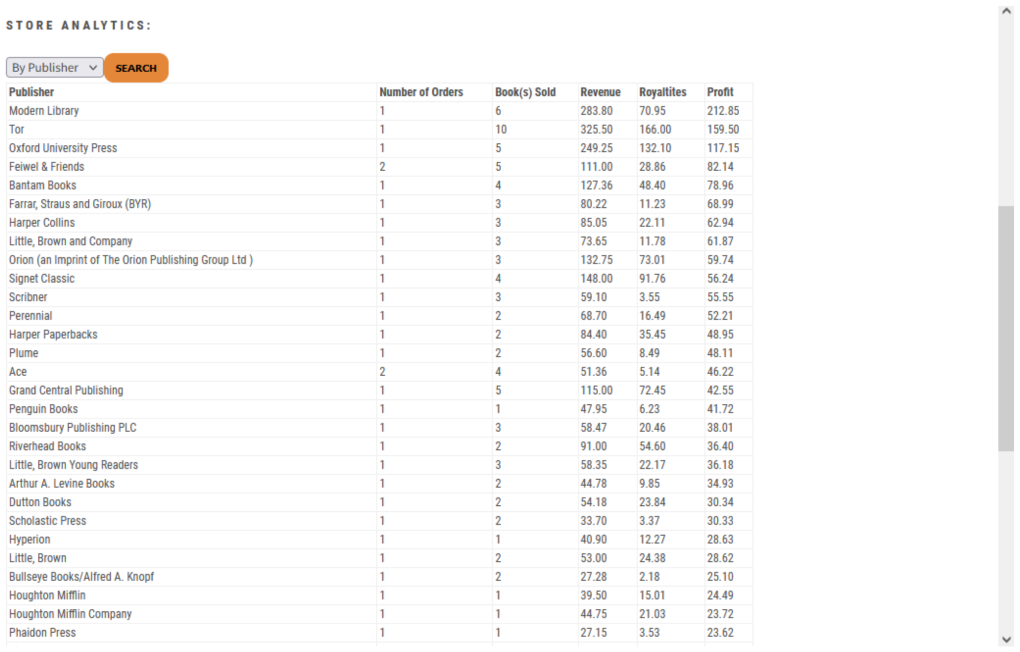
First Name	Last Name	Book(s) in Stock	Book(s) Sold	Revenue	Royaltites	Profit
Charlaine	Harris	30	6	283.80	70.95	212.85
John	Gray	9	10	325.50	166.00	159.50
Anthony	Bourdain	9	10	325.50	166.00	159.50
Michael	Crichton	9	10	325.50	166.00	159.50
Gustave	Flaubert	25	5	249.25	132.10	117.15
Margaret	R.	25	5	249.25	132.10	117.15
Veronica	Roth	12	5	111.00	28.86	82.14
Harper	Lee	17	4	127.36	48.40	78.96
Khaled	Hosseini	24	3	80.22	11.23	68.99
Erich	Fromm	8	3	85.05	22.11	62.94
Alice	Sebold	3	3	73.65	11.78	61.87
Suzanne	Collins	47	6	89.64	29.64	60.00
J.K.	Rowling	69	5	66.00	6.01	59.99
Tim	Dolin	47	3	132.75	73.01	59.74
Nick	Hornby	33	4	148.00	91.76	56.24
James	Patterson	33	3	59.10	3.55	55.55
Malcolm	Bowie	20	2	68.70	16.49	52.21
Celiene	Acester	5	2	84.40	35.45	48.95
Jules	Feiffer	29	2	56.60	8.49	48.11
George	Orwell	41	5	115.00	72.45	42.55
Margaret	Mauldon	29	1	47.95	6.23	41.72
Frank	Scott	41	3	58.47	20.46	38.01
John	Green	41	3	58.47	20.46	38.01
J.R.R.	Tolkien	41	3	58.47	20.46	38.01
Frank	Miller	13	2	91.00	54.60	36.40
Jeffrey	Eugenides	32	3	58.35	22.17	36.18
Stephenie	Meyer	31	2	44.78	9.85	34.93
Jane	Austen	33	2	54.18	23.84	30.34



STORE ANALYTICS:

By Genre — SEARCH

Genre	Book(s) in Stock	Book(s) Sold	Revenue	Royaltites	Profit
Picture book	46	6	183.96	56.89	127.07
Poetry	46	6	183.96	56.89	127.07
Political thriller	46	6	183.96	56.89	127.07
Mystery	51	4	113.48	26.34	87.14
Paranormal romance	51	4	113.48	26.34	87.14
Horror	51	4	113.48	26.34	87.14
Dictionary	12	5	111.00	28.86	82.14
Diary	12	5	111.00	28.86	82.14
Encyclopedia	12	5	111.00	28.86	82.14
Suspense	79	5	86.38	5.73	80.65
Thriller	79	5	86.38	5.73	80.65
Short story	79	5	86.38	5.73	80.65
Science fiction	54	5	149.47	75.06	74.41
Satire	54	5	149.47	75.06	74.41
Romance	54	5	149.47	75.06	74.41
Anthology	52	5	96.11	26.17	69.94
Alternate history	52	5	96.11	26.17	69.94
Action and adventure	52	5	96.11	26.17	69.94
Cookbook	24	3	80.22	11.23	68.99
Book review	24	3	80.22	11.23	68.99
Biography	24	3	80.22	11.23	68.99
Journal	8	3	85.05	22.11	62.94
Math	8	3	85.05	22.11	62.94
Memoir	8	3	85.05	22.11	62.94
Graphic novel	47	3	132.75	73.01	59.74
Fantasy	47	3	132.75	73.01	59.74
Historical fiction	47	3	132.75	73.01	59.74
Fairytale	61	4	72.99	23.05	49.94
Crime	61	4	72.99	23.05	49.94



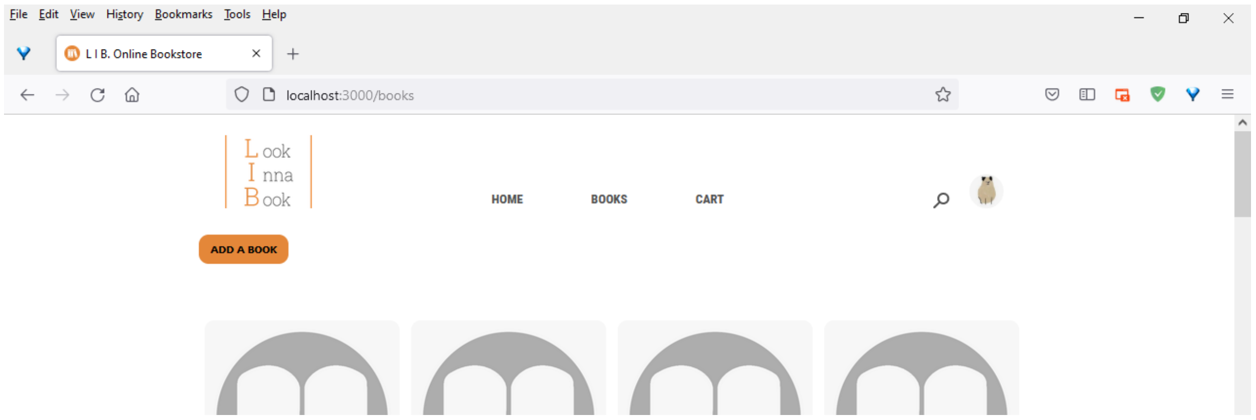
STORE ANALYTICS:

By Publisher — SEARCH

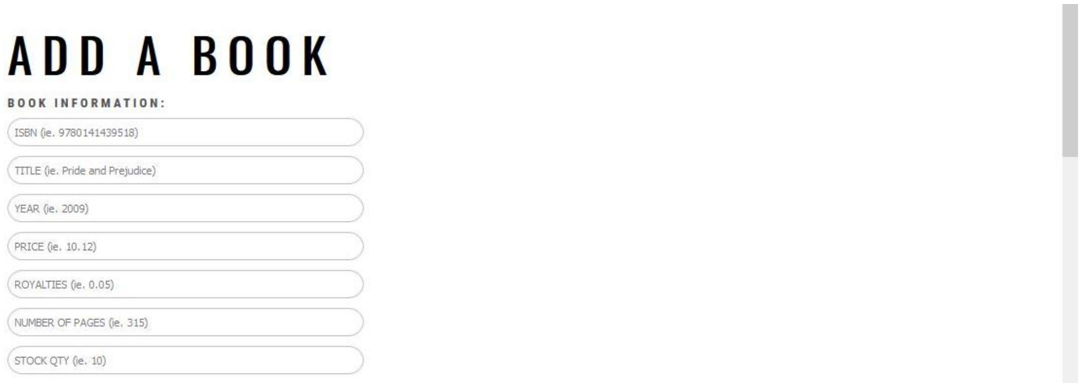
Publisher	Number of Orders	Book(s) Sold	Revenue	Royaltites	Profit
Modern Library	1	6	283.80	70.95	212.85
Tor	1	10	325.50	166.00	159.50
Oxford University Press	1	5	249.25	132.10	117.15
Feiwel & Friends	2	5	111.00	28.86	82.14
Bantam Books	1	4	127.36	48.40	78.96
Farrar, Straus and Giroux (BYR)	1	3	80.22	11.23	68.99
Harper Collins	1	3	85.05	22.11	62.94
Little, Brown and Company	1	3	73.65	11.78	61.87
Orion (an Imprint of The Orion Publishing Group Ltd)	1	3	132.75	73.01	59.74
Signet Classic	1	4	148.00	91.76	56.24
Scribner	1	3	59.10	3.55	55.55
Perennial	1	2	68.70	16.49	52.21
Harper Paperbacks	1	2	84.40	35.45	48.95
Plume	1	2	56.60	8.49	48.11
Ace	2	4	51.36	5.14	46.22
Grand Central Publishing	1	5	115.00	72.45	42.55
Penguin Books	1	1	47.95	6.23	41.72
Bloomsbury Publishing PLC	1	3	58.47	20.46	38.01
Riverhead Books	1	2	91.00	54.60	36.40
Little, Brown Young Readers	1	3	58.35	22.17	36.18
Arthur A. Levine Books	1	2	44.78	9.85	34.93
Dutton Books	1	2	54.18	23.84	30.34
Scholastic Press	1	2	33.70	3.37	30.33
Hyperion	1	1	40.90	12.27	28.63
Little, Brown	1	2	53.00	24.38	28.62
Bullseye Books/Alfred A. Knopf	1	2	27.28	2.18	25.10
Houghton Mifflin	1	1	39.50	15.01	24.49
Houghton Mifflin Company	1	1	44.75	21.03	23.72
Phaidon Press	1	1	27.15	3.53	23.62

Create New Book (Admin Only)

It should be noted that an admin user has access to all the functionalities of a regular user (ie. cart, checkout, orders etc). However, an additional button will appear on the '/books' page to add a new book



By clicking this button (which is hidden to regular users, as well as guests), an admin can reach the page to input a new book



ADD A BOOK

BOOK INFORMATION:

ISBN (ie. 9780141439518)

TITLE (ie. Pride and Prejudice)

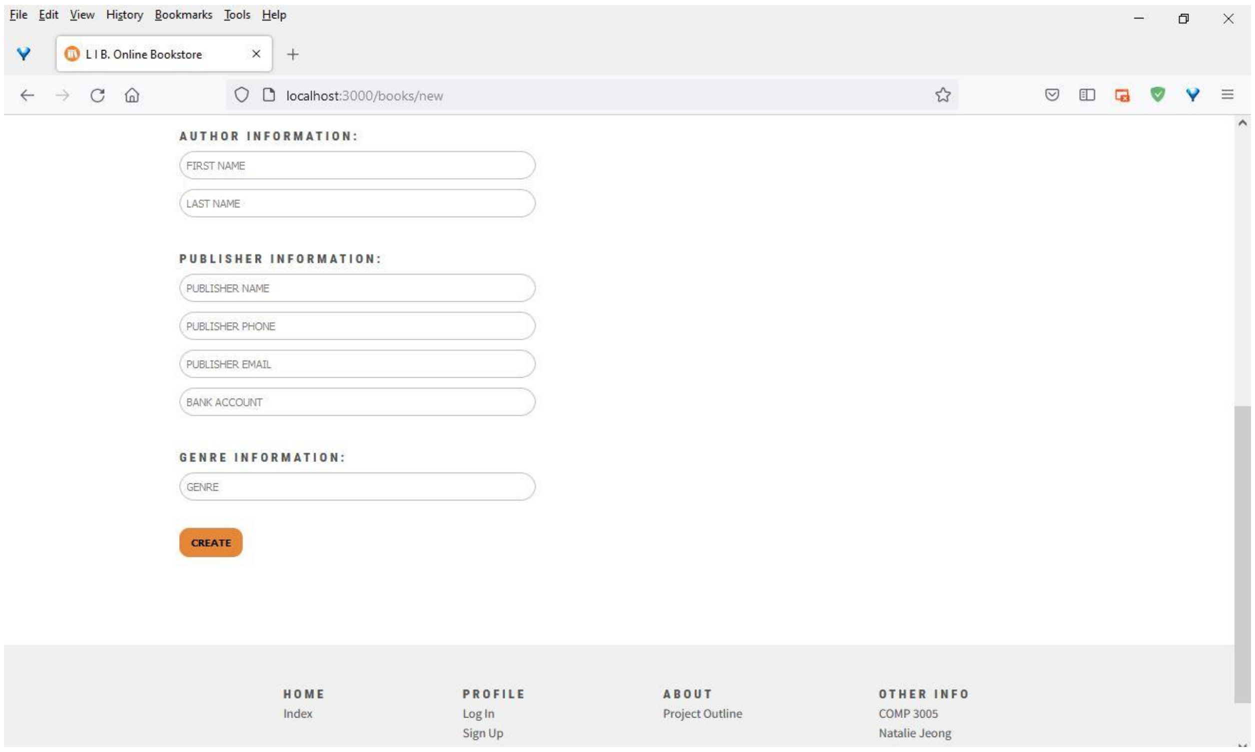
YEAR (ie. 2009)

PRICE (ie. 10.12)

ROYALTIES (ie. 0.05)

NUMBER OF PAGES (ie. 315)

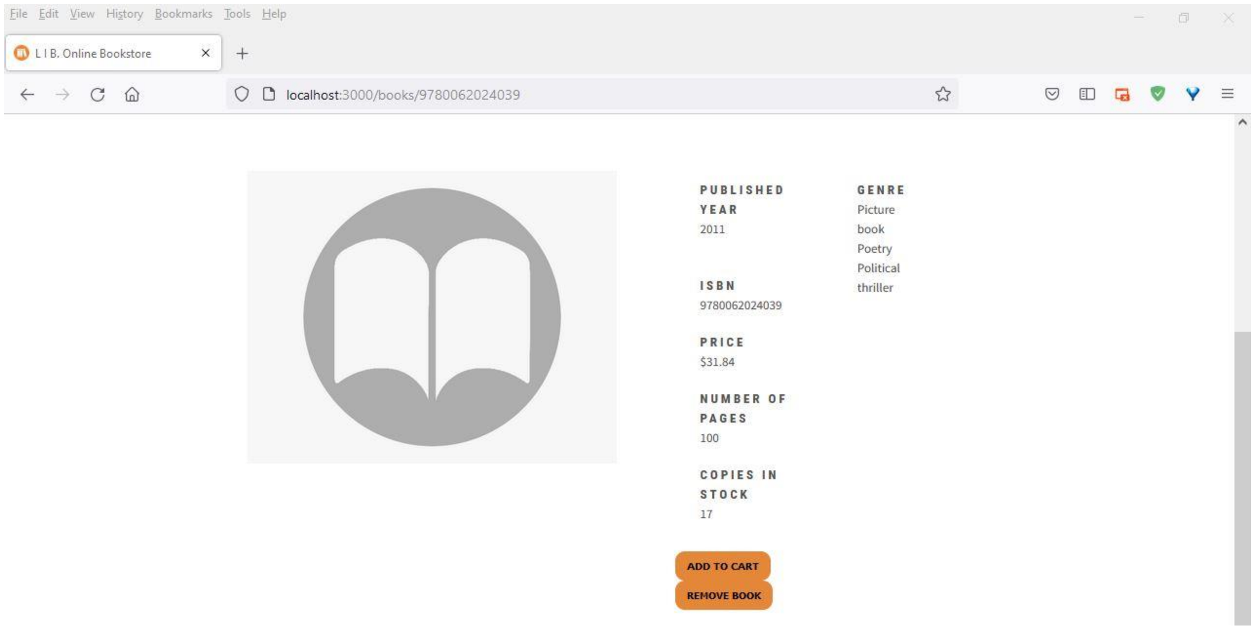
STOCK QTY (ie. 10)



Upon clicking "Create", you will either be redirected to the books collection page (where the new book should show in the list), or remain on the page if an error occurs.

Remove Book (Admin Only)

Similar to adding a book, when a user has an admin status, a “Remove Book” button will be shown for every book



Clicking this remove button simply sets the stock quantity to “-1” and hides it from the user. This ensures that no new orders can be placed, while retaining all current orders being fulfilled/ sales statistic on the product. None of the related data (such as publisher or author) are removed from the system, as keeping this relationship within the database was preferred. This is to cover for scenarios where a book may be deleted for a temporary time, then re-added at a later time.

Orders Given a Threshold (Admin Only)

For this feature of the web application, the functions and trigger outlined in file ‘Function & Trigger (Re-stock).sql’ are used. At the time of seeding, it automatically creates a function, a trigger function, and a trigger within Postgresql. These together automatically orders additional copies of any book that drop below a stock quantity of 10. The number of copies automatically ordered is equal to the number of copies sold in the previous month. This ensures that popular books with high sale volumes are automatically restocked.

7.0 BONUS FEATURES

Graphical User Interface/ Front End Design

There were no requirements to create an application with a front end design. The front end components allow for an easy to use, friendly design and the ability to temporarily have local storage in the browser (for features like the cart/ checkout, and login/ logout). This is especially important for larger applications as making calls to the database can be more expensive than having a first ‘line-of-defense’ through the front end. This example can be seen in features like the search where quick user input verification can be complete before a query. This makes a program more durable and robust since it reduces the possibilities of getting a database error.

Approximate Searching

As mentioned in the search section of the report, users have the ability to enter incomplete, case-insensitive key terms into the search bar. The system will attempt to match the results to any products which contain the given pattern. Such features were implemented through the % wildcard, LIKE and LOWER integrated functionalities within Postgresql.

Sign Up

In addition to the log-in/ log-out feature implemented, a new user can also sign up through the website. The input fields also have a regex check so that the data that get inputted into the system is consistent and the correct type. Given that a non-registered user cannot shop on the website, such features would be critical to a business’s growth in real applications.

8.0 GITHUB REPOSITORY

The GitHub repository for this project can be found using the following link:

https://github.com/natjeo/3005-project