**COMP3005 Project Report**

Tom Lam – 101114541

1. **Conceptual design**

**Cardinality:**

* Participation type is indicated by the number of participants next to each entity in the ER diagram. Specifically, “1..\*” or “1” means total participation while “0..\*” or “0..1” means partial participation
* A book can be published by exactly 1 publisher, A publisher can publish 1 to many books
* A book can be collected by 0 to 1 owner, an owner can collect 0 to many book. There’s no book that belongs to 2 different owner
* A book can be ordered by 0 to many orders, an order can order 1 to many book
* An order can be placed by exactly 1 customer, a customer can place 0 to many order

**Others:**

* Different authors have different name (to avoid the use of author\_id)
* Different publishes have different name (to avoid the use of publisher\_id)
* Different customer has different name (to avoid the use of customer\_id)

**ER Diagram**

Diagram

Description automatically generated

1. **Relational Schema**

Book(ISBN, publisher\_name, book\_name, genre, description, num\_of\_pages, price)

Author(ISBN, name)

Publisher(name, email, bank\_account, balance, address, phone\_number)

Owner(name, bank\_account, balance, address, email, phone\_number)

Collect(ISBN, owner\_name, unit\_in\_stock, unit\_sold, revenue, expense, publisher\_split)

Order(order\_id, billing\_address, shipping\_address, status, ordered\_date, estimated\_arrivate, location)

Customer(name, bank\_account, balance, billing\_address, shipping\_address)

CustomerOrder(order\_id, customer\_name)

OrderBook(ISBN, order\_id, unit\_ordered)

1. **Normalization**

Functional dependencies

Book(ISBN, publisher\_name, book\_name, genre, description, num\_of\_pages, price)

ISBN 🡪 name, publisher\_name, book\_name, genre, description, num\_of\_pages, price

Author(ISBN, name)

ISBN, name 🡪 ISBN, name

Publisher(name, email, bank\_account, balance, address, phone\_number)

name 🡪 email, bank\_account, balance, address, email, phone\_number

Owner(name, bank\_account, balance, address, email, phone\_number)

name 🡪 bank\_account, balance, address, email, phone\_number

Collect(ISBN, owner\_name, unit\_in\_stock, unit\_sold, revenue, expense, publisher\_split)

ISBN 🡪 owner\_name, unit\_in\_stock, unit\_sold, revenue, expense, publisher\_split

Order(order\_id, billing\_address, shipping\_address, status, ordered\_date, estimated\_arrivate, location)

order\_id 🡪 billing\_address, shipping\_address, status, ordered\_date, estimated\_arrivate, location

Customer(name, bank\_account, balance, billing\_address, shipping\_address)

name 🡪 bank\_account, balance, billing\_address, shipping\_address

CustomerOrder(order\_id, customer\_name)

order\_id 🡪 customer\_name

OrderBook(ISBN, order\_id, unit\_ordered)

ISBN, order\_id 🡪 unit\_ordered

All relations are already in BCNF because their functional dependency is also the superkey

1. **Database Schema Diagram**

Graphical user interface, application, Word

Description automatically generated

1. **Implementation:**
   1. **Architectures:**

The table creation “table.sql” and insertion “insertion.sql” files can be found in the “database\_initialize” directory.

The program entry point is in Main.java. It initializes the Application and run it.

The functionalities are implemented by 3 main classes:

* View.java handles all interaction between the store and its users (Customer or Owner)
* JDBCController.java handles all interaction with the store and the database. This is where all database querries and updates are implemented
* Application.java implements all application requirements stated in the **“Problem Statement”** of Look-Inna-Book by using the View and JDBCController classes to interact with the store users with the store databases

The remainings are helper classes of which main functionalities are storage and display (toString)

* Customer.java stores informations about a customer
* Owner.java stores information about an owner
* Book.java stores temporary book information got from the database
* BookOrder.java stores a Customer’s chosen Book with the number of that book that’s going to be ordered when checkout.
* Basket.java stores Customer’s BookOrder(s) before checkout. Basket will be emptied after checkout or restart
* Order.java stores an order information when tracking an order
  1. **Program initialization:**

1. Create the database and initialize it by running everything inside “table.sql” and “insertion.sql”.
2. In Main.java. Please change the following paramenters  
    PORT: database port  
    DATABASE\_NAME: name of created database  
    USERNAME: user name to log into database  
    PASSWORD: password of the username  
   to the appropriate values so that JDBCController can connects to the database
   1. **Interface:**

**Main views:**

When first started, you will be greeted with the Main view and required to choose either Customer or Owner view:

A picture containing text

Description automatically generated

Customer view (not logged in):

Text, table

Description automatically generated with medium confidence

Owner view (not logged in):

Table

Description automatically generated

You are freely allow to switch between view by exiting back to the Main view.

**Customer View:**

When not logged in, the customer can browse books, add book to basket, remove book from basket, show basket, and log in.

Text, table

Description automatically generated with medium confidence

When logged in, the customer gains the ability to checkout the basket, track order, and log out. And of course, loses the ability to log in.

Text

Description automatically generated

The default insertion file “insertion.sql” gives 5 Customers to choose from: Customer1, Customer2, Customer3, Customer4, Customer5

Browsing books, adding and removing to/from basket works the same regardless of whether the user is logged in or not. More over, when logging in or logging out, the basket is not reseted.

**Customer Browse Book:**

When choose to browse book, you will be brought to the Browse Book view. Which gives to option to search using the filters, show, clear, and modifies the filters.

Text

Description automatically generated

Initially, all filters are empty:

Text

Description automatically generated

Add filters to search for books that are writen by Author3 and published by either Publisher2 or Publisher1:

Text

Description automatically generated

Text

Description automatically generated

Text

Description automatically generated

Resulting filters:

Text

Description automatically generated with medium confidence

When choose to search for book, the application will show the books that are collected by an owner after applying all of the filters added above

Text

Description automatically generated

Similarly, to browse for book by genre, publisher, book name, ISBN, adjust the corresponding filters

**Add/remove book order to/from customer basket**

Choose the ISBN from the list printed and the quantity. For example:

Add 10 books with ISBN 3:

A picture containing table

Description automatically generated

Add 10 books with ISBN 5:

Table

Description automatically generated with medium confidence

The basket information after adding:

Text

Description automatically generated

Similarly, to remove, choose the ISBN and quantity to remove, if the quantity exceeds the current quantity in the basket. The application will completely remove the entry

Text, letter

Description automatically generatedText, letter

Description automatically generated

Text

Description automatically generated

**Customer check out:**

The application will show your basket information and prompt to enter your biling and shipping address. If you leave it bank, the store will use the default value stored in your profile. In this case, the default billing address is “BillingAddress1” for Customer1

***Text, letter

Description automatically generated***

The basket will be emptied and your customer balance will be updated after placing the order.

Before, the balance was $98800

After, it’s $98800 – $117 = $98683

Text

Description automatically generated

After placing order, JDBCController will automatically order more books from Publisher if the number of books falls under the fixed MIN\_BOOK\_THRESHOLD (50).

**Customer track order:**

You will be showed the IDs of your placed order. When prompted, enter one of the ID and the application will show all information about that order

***Text, letter

Description automatically generated***

**Owner View:**

When not logged in, owner can only browse book and log in

Table

Description automatically generated with medium confidence

The default insertion file “insertion.sql” gives 2 owner “Owner1” and “Owner2” to log in from. When logged in, owner gains additional ability

Text

Description automatically generated

Similar to Customer, browsing books works the same for regardless of whether the owner is logged in or not.

**Owner browse book:**

Browsing books for owner works very similar to Customer browse book. Indeed it has the same menu as Customer’s and filter works the same as Customer browse book filters.

Text

Description automatically generated

The key difference is that Owner can only browse books that are not collected by any owner (including themselves).

Text

Description automatically generated

**Owner Add/Remove Book**

To add book, enter the ISBN listed, choose the price to sell the book for as well as the ratio of revenue to share with the publisher (the publisher is very generous so you can literally not share any penny with them)

Graphical user interface, text, application, email

Description automatically generated

After adding, the application will automatically order the fixed AMOUNT\_ TO\_ORDER (300) numbers of books from the publisher, as well as send money from the Owner to the Publisher to pay for that.

Table

Description automatically generated

Previous balance is $101317

The new balance is $101317 – (0.3 split ratio \* 300 books \* $60 per book) = $95917

Similarly, removing books from collection requires entering the ISBN from the list. Although there won’t be any refund when the owner removes the books from his collection.

Graphical user interface, text, application

Description automatically generated

**Owner show all book in collection**

As the name suggest, this shows all books that are owned by this owner in collection

Text

Description automatically generated

**Owner show sale record**

The owner sale record menu is similar to owner browse book menu and customer browse book menu. We can modify the filter to change how sale records is being aggregated

Text

Description automatically generated

Without any filter, “(1) Show records” shows all sale information of all books

Text

Description automatically generated

Text

Description automatically generated

To show sale information for a certain genre(s), adjust the genre filter

For example: Only show sales for Genre1

Graphical user interface, text, application

Description automatically generated

A picture containing background pattern

Description automatically generated

Text

Description automatically generated

Similarly, to show sale per author and sale per publisher, just modify the corresponding filter

1. **Github Repository:**

<https://github.com/Tom1072/Look-Inna-Book>

1. **Demo availability:**

I am available any time from 9 am to 5 pm on December 20th. But here are my preferences:

11:00 AM to 11:20 AM

11:20 AM to 11:40 AM

11:40 AM to 12:00 PM