CSCE 608 - DATABASE SYSTEMS
Project 1

Geethik Narayana Kamineni
UIN - 825008990

10th October 2017

Part (a) - Description

The project is about **"Online Book Store"**. The main application which the project serves is to provide an interface for selling books. Web application is being developed to act as the user interface for the project. The application was designed to serve the functionality of e-commerce domains like Amazon Inc., but restricted only for sale of books.

The application in the background stores the collection of books which are available for sale. The properties and the descriptions of the books stored are:

- Name of the book, Author, Price of the book,
- Page count of the book, Genre of the book,
- Quantity of stock available for that book.

These books are represented in the form of thumbnails of images on the website so that it gives the user or customer an appealing interface while the purchase is being made. The users need to register for the website if they are interested in making a purchase. The database stores the customer data when the user registers. It stores the details like:

- Email address (which needs to be unique for each customer),
- Name, Address and the mobile number

The functionalities provided in the website are as follows:

- Provides search functionality across the database of books available using
  - Book name
  - Author name
- Provides the cart functionality where the user can store books he likes and re-visit the website to purchase them any time later
- Provides the update and delete function to alter the quantity of the books in the cart.
- Stores the details of the orders after they are being successfully placed.
- Insert functionality to add new users and to store their personal information like address.
- Stores the shipping details of the user, and retrieves them before placing the order where the user can update them.
- Login functionality to protect the user data and preferences.
- Updates the cart functionality to be empty after the purchase has been done.

Payments functionality has been neglected in the design to reduce the complexity of the development process. But can be extended using the existing design in future if required for commercial purposes. So, after the user updates the shipping address and submitted, it is assumed to imply that the order has been placed.

Part (b) - Data Collection

The database of the project is named “Bookstore” which involved 6 tables - Genre, Books, Customers, Cart, Orders, Shipping. The schema for the table is being developed as per the code in *createtables.sql* The Genre table of the database stores the genre ID and the genre of the books available for sale on the website. Currently the website only supports books across four genres which are -

- “Comics”
- “Horror”
- “Science-fiction” and
- “Fantasy”.

So, the genre table is being populated with four entries manually.
The data for the database table Books is being taken from https://play.google.com/store/books according to the four genres specified above. The database is currently populated with forty entries, where each genre currently has ten books. From above link, only book titles author name, genre and their images are being collected, rest all the attributes like page count of the book, quantity of stock being available and pricing are being generated manually at random.

The data collection for the Customers database table which contained thousand entries was generated from http://www.mockaroo.com/ using the random attribute generator to populate it. The email address which are the primary key of the database table being generated are checked for duplicates using MS Excel, and duplicate entries removed and new entries generated to ensure that it resembles the primary key of the data. Zip code data of the customers are assumed to be independent of the address and was generated using auto increment function.

The database tables are generated manually as per the code in *insertValues.sql* but since they are actually obtained from the users shopping on the website, so the queries to populate the data tables Cart, Orders and Shipping are disregarded and filled only when the user who logs into the website make a purchase or stores his selection of future purchases in the cart.

Part (c) - ER Diagram

The Entity relationship diagram of the database “Bookstore” is as follows:

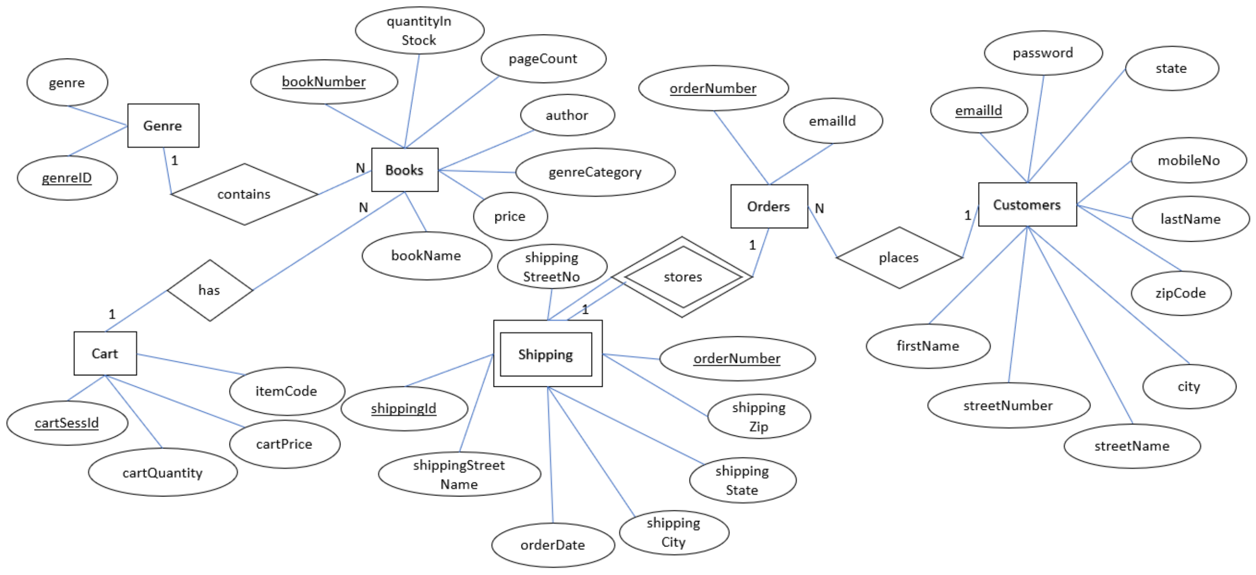


Figure 1:

Entity-Relationship diagram

The entities are represented by the rectangles, which inturn represent the database tables. The ER diagram is arrived at by initially fixing on the functionalities of the website as follows:

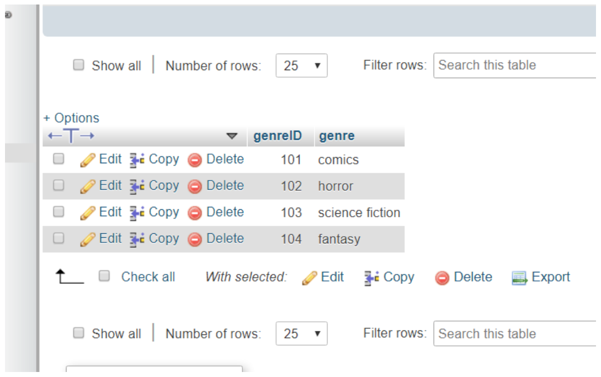
- The entities were fixed by the purpose of selling of books. Since books have genres it forms a new entity, as each genre may contain many books in order to reduce the redundancy in data a new entity is formed.
- Since, it needs to serve the sales functionality we need to have a cart as an entity and customers to represent the users.
- In order to store the purchases being done and record them, we need to maintain a orders and shipping entities.
- The attributes which highlight the key features of the entities are then developed.
- The relations between the entities are developed in such a manner to reduce the dependency in the data and to obtain an efficient database schema which is normalised. The relations existing in the current database schema between genre (contains)$\longrightarrow$ books which is one-many relation.
- In a similar manner, the relation between the cart (has)$\longrightarrow$books is also a one-many relation since, many books can be placed in the cart of a single user.
- The relation between customers(places) $\longrightarrow$orders is a one-many relation as each customer can place multiple orders. Whereas the relation between shipping(stores) $\longrightarrow$ orders, is a weak entity relation as the existence of shipping details will only be valid if there is a valid order being placed in the orders table of the database.
- So, as the ER diagram represents there are 4 relations and 6 entities in the table of which one is weak entity relation.

Part (d) - Normalisation

The rules of normalisation are having atomic values (1NF), all non-key attributes are functionally dependent on the primary key(2NF) and remove the fields that are independent of the primary key or avoid transitive dependency(3NF). All these 3 normal forms are satisfied if the relation is in BCNF according to which, every non-trivial functional dependency $X \longrightarrow Y$, is in BCNF if X is a super key. If the relation satisfies BCNF then all the other three lower forms of normalisation are satisified.

In the case of bookstore,

- Table - Genre
  - Genre ID
  - Genre


Show all
Number of rows: 25
Filter rows: Search this table
+ Options
Check all
With selected: Edit Copy Delete Export
Show all
Number of rows: 25
Filter rows: Search this table

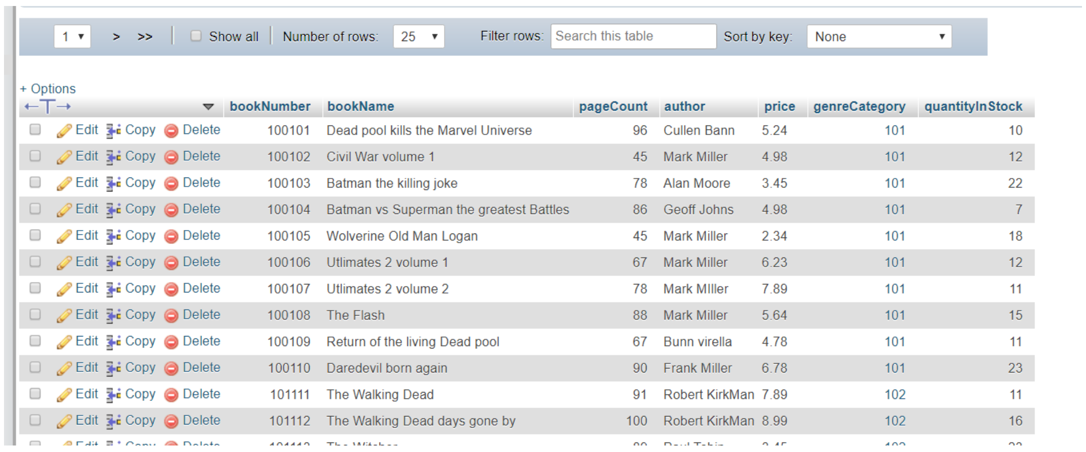
	genreID	genre
Edit Copy Delete	101	comics
Edit Copy Delete	102	horror
Edit Copy Delete	103	science fiction
Edit Copy Delete	104	fantasy

Figure 2:

Here the non-trivial functional dependency is GenreID $\longrightarrow$Genre and GenreID is a superkey, so it is in BCNF.

- Table - Books
  - Book Number
  - Book Name
  - Page Count

- Price
- Quantity In Stock
- Author
- Genre category



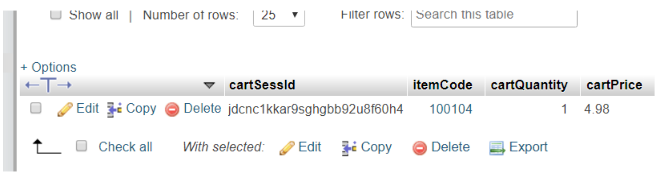
1 > >> Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

bookNumber	bookName	pageCount	author	price	genreCategory	quantityInStock
100101	Dead pool kills the Marvel Universe	96	Cullen Bann	5.24	101	10
100102	Civil War volume 1	45	Mark Miller	4.98	101	12
100103	Batman the killing joke	78	Alan Moore	3.45	101	22
100104	Batman vs Superman the greatest Battles	86	Geoff Johns	4.98	101	7
100105	Wolverine Old Man Logan	45	Mark Miller	2.34	101	18
100106	Utlimates 2 volume 1	67	Mark Miller	6.23	101	12
100107	Utlimates 2 volume 2	78	Mark MIller	7.89	101	11
100108	The Flash	88	Mark Miller	5.64	101	15
100109	Return of the living Dead pool	67	Bunn virella	4.78	101	11
100110	Daredevil born again	90	Frank Miller	6.78	101	23
101111	The Walking Dead	91	Robert KirkMan	7.89	102	11
101112	The Walking Dead days gone by	100	Robert KirkMan	8.99	102	16

Figure 3:

Here the non-trivial functional depenedency is Book Number $\longrightarrow$ BookName, PageCount, Price, Author, QuantityInStock, GenreCategory. This is no multi-valued dependency in the table as each book is assumed to have only one author so even it is atomic. This is in BCNF.

- Table - Cart
  - Cart Sess Id
  - Cart Price
  - Cart Quantity
  - Item code



Show all | Number of rows: 25 Filter rows: Search this table

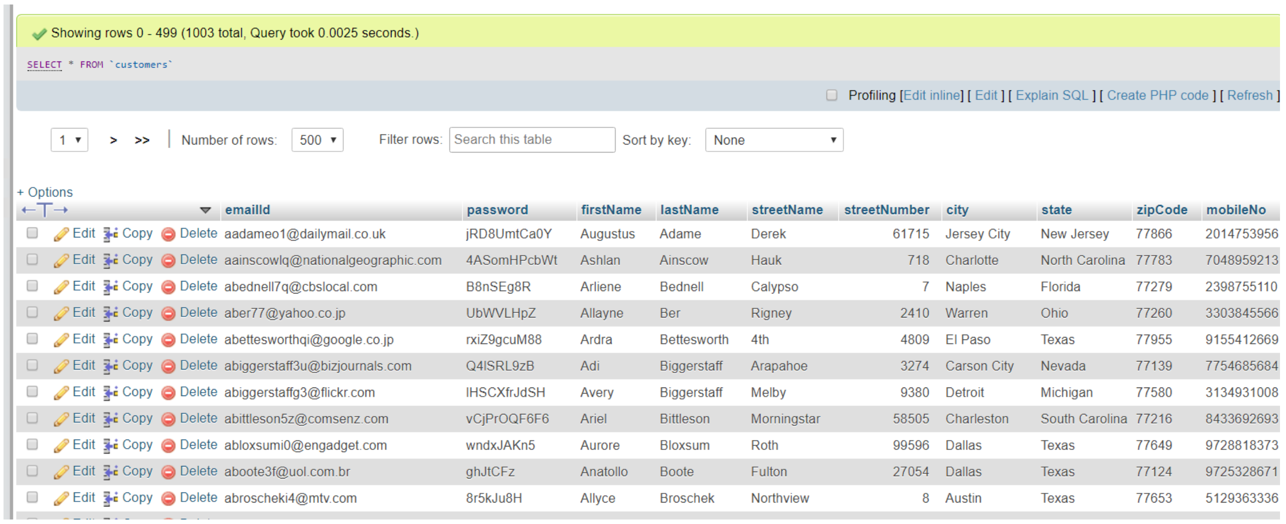
cartSessId	itemCode	cartQuantity	cartPrice
jdcnc1kkar9sghgbb92u8f60h4	100104	1	4.98

Figure 4:

Here the non-trivial functional dependency is CartSessId $\longrightarrow$CartPrice, CartQuantity, ItemCode. Here since there is no dependency, this is in BCNF. Since, the cartSessId is the only key, so only one item can be purchased or put in the cart at a time. If we want to add more items to the same cart then the primary key needs to be altered. In order to keep the application simple, only the CartSessId is considered as the key.

- Table - Customers
  - email Id
  - password
  - First Name
  - Last Name
  - Street Number
  - Street Name
  - City

- State
- ZipCode
- Mobile No



Showing rows 0 - 499 (1003 total, Query took 0.0025 seconds.)

```
SELECT * FROM `customers`
```

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

1 > >> | Number of rows: 500 Filter rows: Search this table Sort by key: None

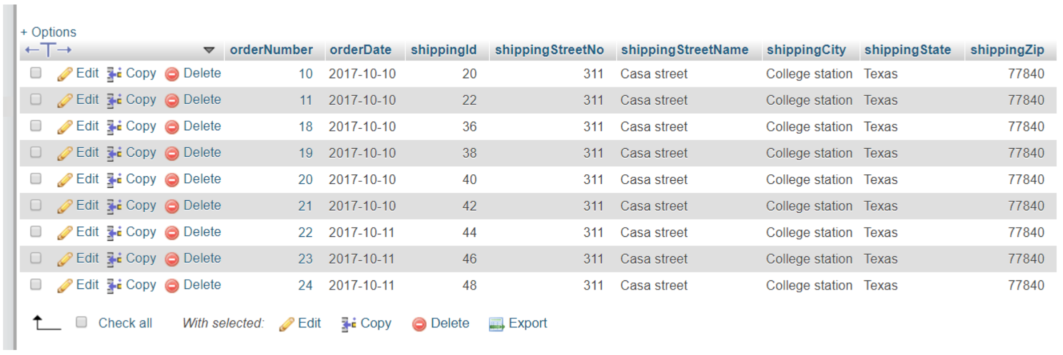
+ Options

emailId	password	firstName	lastName	streetName	streetNumber	city	state	zipCode	mobileNo
aadameo1@dailymail.co.uk	jRD8UmtCa0Y	Augustus	Adame	Derek	61715	Jersey City	New Jersey	77866	2014753956
aainscowlq@nationalgeographic.com	4ASomHPcbWt	Ashlan	Ainscow	Hauk	718	Charlotte	North Carolina	77783	7048959213
abednell7q@cbslocal.com	B8nSEg8R	Arliene	Bednell	Calypso	7	Naples	Florida	77279	2398755110
aber77@yahoo.co.jp	UbWVLHpZ	Allayne	Ber	Rigney	2410	Warren	Ohio	77260	3303845566
abettesworthqi@google.co.jp	rxiZ9gcuM88	Ardra	Bettesworth	4th	4809	El Paso	Texas	77955	9155412669
abiggerstaff3u@bizjournals.com	Q4ISRL9zB	Adi	Biggerstaff	Arapahoe	3274	Carson City	Nevada	77139	7754685684
abiggerstaffg3@flickr.com	IHSCXfrJdSH	Avery	Biggerstaff	Melby	9380	Detroit	Michigan	77580	3134931008
abittleson5z@comsenz.com	vCjPrOQF6F6	Ariel	Bittleson	Morningstar	58505	Charleston	South Carolina	77216	8433692693
abloxsumi0@engadget.com	wndxJAKn5	Aurore	Bloxsum	Roth	99596	Dallas	Texas	77649	9728818373
aboote3f@uol.com.br	ghJtCFz	Anatollo	Boote	Fulton	27054	Dallas	Texas	77124	9725328671
abroscheki4@mtv.com	8r5kJu8H	Allyce	Broschek	Northview	8	Austin	Texas	77653	5129363336

Figure 5:

Here the non-trivial functional dependency is emailId ⟶password, FirstName, lastName, streetNumber, streetName, city, State, ZipCode , MobileNo. Here, actually this won't be in BCNF if the ZipCode actually determines the state and the city independent of the emailId, so then decomposition of the table is required. But since, as stated earlier in the Part(b) - Data collection, ZipCode is assumed to be independent of the city, state and same is assumed for state and city (Although same name of city can exist in different locations as Hyderabad in India and Pakistan). So, this is in BCNF.

- Table - Orders
  - Order Number
  - Email Id



+ Options

orderNumber	orderDate	shippingId	shippingStreetNo	shippingStreetName	shippingCity	shippingState	shippingZip
10	2017-10-10	20	311	Casa street	College station	Texas	77840
11	2017-10-10	22	311	Casa street	College station	Texas	77840
18	2017-10-10	36	311	Casa street	College station	Texas	77840
19	2017-10-10	38	311	Casa street	College station	Texas	77840
20	2017-10-10	40	311	Casa street	College station	Texas	77840
21	2017-10-10	42	311	Casa street	College station	Texas	77840
22	2017-10-11	44	311	Casa street	College station	Texas	77840
23	2017-10-11	46	311	Casa street	College station	Texas	77840
24	2017-10-11	48	311	Casa street	College station	Texas	77840

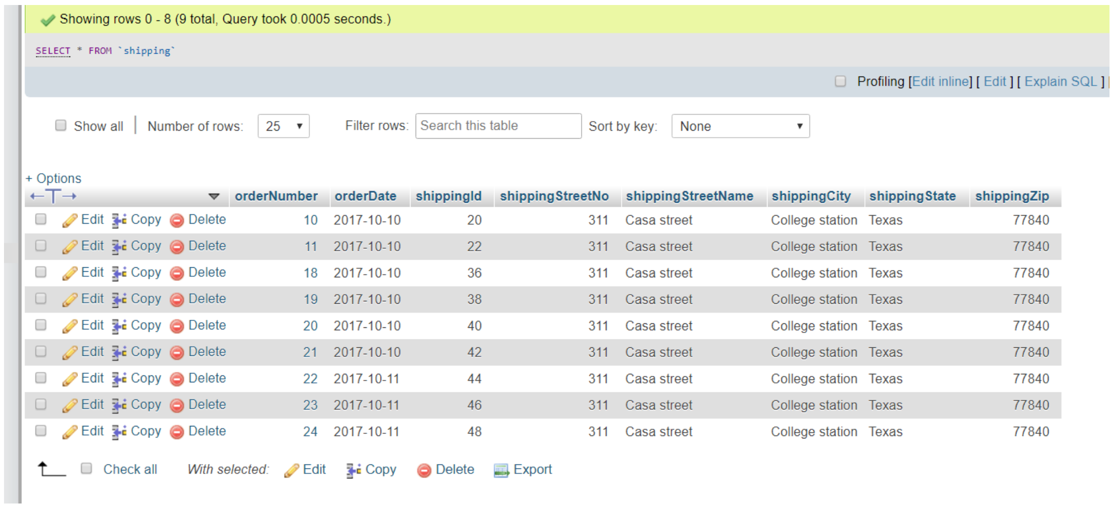
Check all With selected: Edit Copy Delete Export

Figure 6:

Here the non-trivial functional dependency orderNumber ⟶emailId. Here it is in BCNF.

- Table - Shipping
  - Order Number
  - Order Date
  - Shipping Id
  - Shipping StreetNo

- ShippingStreetName
- ShippingCity
- ShippingState
- ShippingZip



✓ Showing rows 0 - 8 (9 total, Query took 0.0005 seconds.)

```
SELECT * FROM `shipping`
```

Profiling [Edit inline] [Edit] [Explain SQL]
Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None
+ Options

	orderNumber	orderDate	shippingId	shippingStreetNo	shippingStreetName	shippingCity	shippingState	shippingZip
Edit Copy Delete	10	2017-10-10	20	311	Casa street	College station	Texas	77840
Edit Copy Delete	11	2017-10-10	22	311	Casa street	College station	Texas	77840
Edit Copy Delete	18	2017-10-10	36	311	Casa street	College station	Texas	77840
Edit Copy Delete	19	2017-10-10	38	311	Casa street	College station	Texas	77840
Edit Copy Delete	20	2017-10-10	40	311	Casa street	College station	Texas	77840
Edit Copy Delete	21	2017-10-10	42	311	Casa street	College station	Texas	77840
Edit Copy Delete	22	2017-10-11	44	311	Casa street	College station	Texas	77840
Edit Copy Delete	23	2017-10-11	46	311	Casa street	College station	Texas	77840
Edit Copy Delete	24	2017-10-11	48	311	Casa street	College station	Texas	77840

Figure 7:

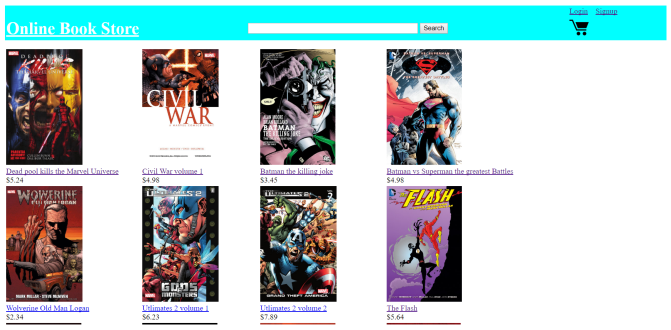
Here the non-trivial dependency is (orderNumber, shippingId) $\longrightarrow$orderDate, ShippingStreetNo, ShippingStreetName, ShippingCity, ShippingState, ShippingZip

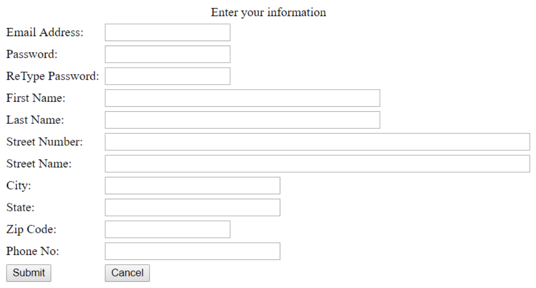
This table might not be normalised as per the application as the user fills the data in this table, so zip might also be independent on the primary key of the above relation. So, this can further be decomposed to put zip of the shipping in a new table else if we assumed still that Zip doesn't determine the other quantities, then this is in BCNF.

Part (e) - User interface

The interface is hosted at https://cryptic-reaches-45863.herokuapp.com.

The user interface can be used to have a glance at the books and if interested to register into the website by clicking the signup link and entering the details as in the figures below:





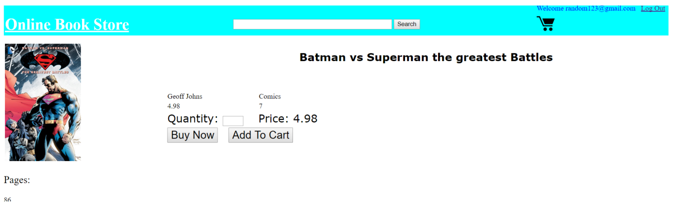
Enter your information

Email Address:	
Password:	
ReType Password:	
First Name:	
Last Name:	
Street Number:	
Street Name:	
City:	
State:	
Zip Code:	
Phone No:	

Figure 8: Home page

Figure 9: Signup form

The user can then click on book to know more about the book and choose the book to be placed in the cart if he likes and wants to purchase it as in figures below:



Batman vs Superman the greatest Battles

Geoff Johns
4.98

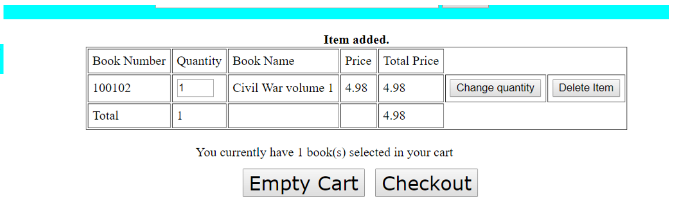
Comics
7

Quantity:

Price: 4.98

Pages:

86



Item added.

Book Number	Quantity	Book Name	Price	Total Price
100102	1	Civil War volume 1	4.98	4.98
Total	1			4.98

You currently have 1 book(s) selected in your cart

Figure 10: Details page

Figure 11: Cart

The user then is asked for the shipping information and then when he submits the shipping information the payment process is skipped and order is being placed. So, in this manner, the interface can be used to place orders of the books, and the usage of the database to serve the functionality is as follows:

- Customers table to store the details of the customers who register on the website and use them to allow for login
- Books to store and retrieve the properties of the books available
- Cart to aid in the purchase process
- Orders and shipping to store the order details for the purchases being done.

Part (f) - Source code

The source code is as follows:
SQL files: *createtable.sql* and *insertValues.sql* and rest all
PHP files :

- *index.php*, *signup.php*, *validatesignup.php*
- *signin.php*, *validateuser.php*, *topmenu.php*
- *addcustomer.php*, *cart.php*, *allitemslist.php*
- *checkconnection.php*, *countcart.php*, *showcart.php*
- *itemdetails.php*, *itemlist.php*, *logout.php*
- *shippinginfo.php*,*searchitems.php*, *placeorder.php*

JS files - *checkform.js*
CSS files - *style.css*
to make the interface which are being provided along with the report
The instructions to test the interface using WAMP server are as follows:
Download necessary software:

- Apache Web Server - to host from our local computer, "localhost" is the database host
- MySQL - was choosen as the RDMS
- PHP - was the chosen language for the web application UI, embeded in html and css
- Alternatively from downloading each software seperately there is bundle of these programs provided online WINDOWS download - http://www.wampserver.com/en/ MAC downlaod - https://www.mamp.info/en/ LINUX download - follow instructions provided at http://lamphowto.com/

Configure software:

- In the WAMP server the MySQL login credentials used are Username: root password: (none)

- If other username and password are used then the php files will have to be edited. Anywhere the following variables are displayed please change to match your MySQL login.
- In all the PHP files of connect queries DB user = 'root' and for DB password="(alter password here) and the DB name='bookstore'
- Download the source code and place them in C:\wamp64\www\ folder of the wamp64 installation directory.
- In the MySQL console of the WAMP server run the SQL scripts to generate and setup (populate) the database.
- Run the following commands in MySQL console - "source createtable.sql" and "source insertValues.sql".
- After this the website can launched by localhost/php/index.php and then various php files traversed.
- Since only single item purchase is allowed, so if the cart contains an item, addition of another item to cart just leaves a blank screen, to redirect either click on cart to check its capacity or redirect to index page by clicking on "Online Book Store" at the top.

Part (g) - Discussion

The difficulties in the development were in designing the efficient schema of the database which is normalised. Normalising a database too extremely is even not productive, so a balance to achieve the BCNF was challenging. Moreover, to develop the various PHP pages and the collection of data was tedious task. Through this project understood the big picture of development of website and database design which goes at the back of it. Got a better understanding of the normalisation forms. The website can further be improved in the following aspects:

- To include payments
- Allow purchase of multiple items at the same time.
- Also can create an admin account, to keep updating the books and other items.