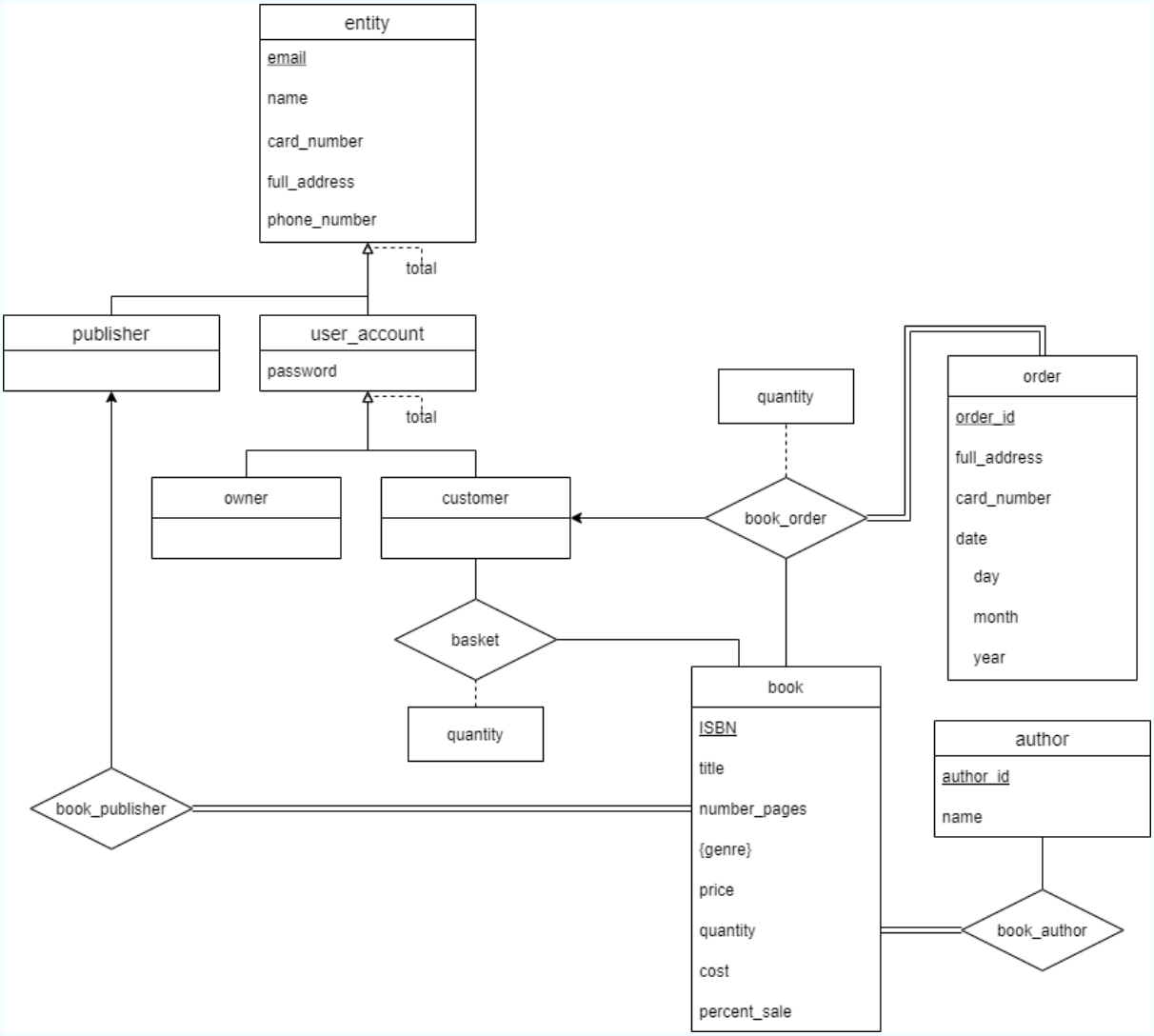
COMP 3005 : Final Project

Group Members : Suhail Mohamed, Abdallah El-Dali

2.1 Conceptual Design:



Assumption made regarding cardinalities and relationship types:

- **Book - publisher:** a book can only have one publisher, hence the arrow going from book to publisher (this assumption was allowed by the professor). A publisher can have many books they have published, example: the harry potter series can all be published under the *scholastic* publishing company, this is the reason we have the non-arrow from publisher to book. Additionally, we made the assumption every book must have a publisher associated with it, hence the total participation from the book-side of the relation.

COMP 3005 : Final Project

Group Members : Suhail Mohamed, Abdallah El-Dali

- **Book - customer:** the book-customer relationship is described using the basket relationship. Like a real basket, a customer can put many books in their basket, oppositely a book may be in many customers basket's, because of this the relationship between customer and book is many-to-many. A basket for a customer only exists once that customer has put something in their basket, if their basket is empty then there is no basket associated with that customer. Not every book has been added to some basket, some books have no one interested in them. Due to this the relationship between the entities are partial on both sides.

- **Book - author:** every book has an author, hence total participation on the book side. A book can have many authors, hence the non-arrow to the author entity from the book. Not every author has to have written a book, hence the partial participation from author and many authors can collaborate to write a book hence the non-arrow from author to book

- **Book - order - customer:** for a *book-order* pairing it can only be the case that it is associated with one customer, as an order can only be associated with one customer. example: order_id #1 can only be associated with 1 customer Snoop Dog, it cannot be the case order_id #1 is associated with customers Snoop Dog and Ahmed El-Roby, as an order is associated with the customer who paid for it and only 1 customer can pay at checkout. Because of this for a book-order pairing it must be the case it can only be associated with one customer, hence the arrow from book-order to customer.
  For a *customer-book* pairing it can be the case that it is associated with many orders, for example: Snoop Dog may order *Harry Potter and the philosopher's stoned* in order_id#1 and later on may order the book again in order_id#420, because of this we can have a book-customer pairing associated with many orders hence the non-arrow from book-customer to order. Additionally all orders will be associated with a book-customer pairing, as every order must have a customer it goes to and products it ships, hence the total participation from the order entity.
  Lastly for a given *customer-order* pairing it can be associated with many books, for example: Snoop Dog may order *Harry Potter and the philosopher's stoned* in order_id#1 he may also order *Catcher in the High* in the same order_id#1. Because of this a customer-order pairing can be associated with many books, hence the non-arrow from customer-order to books.

- **Entity - publisher - user_account:** The Entity entity set contains the most basic traits a publisher and a user_account have such as name, email, address, billing information, etc. However, an entity can only be either a user_account or a publisher, not both, and a publisher cannot be a user_account and vice versa, which is the reason why we used a total disjoint between the two. The only difference between the two is that user_account must have a password attribute since it needs to be used to login into the bookstore, while publisher isn't a user, thus doesn't require it.

COMP 3005 : Final Project

Group Members : Suhail Mohamed, Abdallah El-Dali

- **user_account - owner - customer:** Simply put, an owner and a customer are user accounts in our bookstore. However, like the above explanation, a user_account must either be an owner, or a customer, not both, and a customer cannot be an owner and vice versa, which is why we used a total disjoint between the two.

COMP 3005 : Final Project

Group Members : Suhail Mohamed, Abdallah El-Dali

2.2 Reduction to Relational Schemas:

publisher(*publisher_email, name, card_number, full_address, phone_number*)
owner(*owner_email, password, name, card_number, full_address, phone_number*)
customer(*customer_email, password, name, card_number, full_address, phone_number*)

book(*ISBN, title, number_pages, publisher_email, cost, price, percent_sale, quantity*)
book_genre(*ISBN, genre_type*)
author(*author_id, name*)
book_author(*ISBN, author_id*)
basket(*customer_email, ISBN, quantity*)

order(*order_id, customer_email, full_address, card_number, day, month, year*)
book_order(*ISBN, order_id, quantity*)

Notes:

- book_publisher is removed as the relationship between book-publisher is many-to-one, with total participation on the many side hence each book is given its associated publisher's email.

- **book_genre** table is needed to represent the multi-valued attribute genre in book

- **order** is given the customer_email attribute as it is equivalent to hold customer_email in the **order** table as opposed to holding it in the **book_order** table.
  Example: order_id#12 is only associated with Snoop Dog

book_order

ISBN	Order_id	Quantity	Customer_email
2	12	5	Snoop@gmail.com
3	12	2	Snoop@gmail.com
0	12	1	Snoop@gmail.com
2	13	3	Ahmed@gmail.com

As explained above we already know an order can only be associated with the one customer who paid for it at checkout and every order has one customer it is associated with. As there is a many-to-one relationship from order to customer with total participation on the order side it is equivalent to hold the customer_email information in the **order** table as opposed to holding the information in the **book_order** table. We can determine what customer is associated with a specific order by simply joining **book_order** and **order** together.

Example:

`book_order`

ISBN	Order_id	Quantity
2	12	5
3	12	2

`order`

Order_id	Customer_email	Full_address	...
12	Snoop@gmail.com	Compton, Ca	...

Joining the two above tables would give us the equivalent information as holding customer_email in the **`order`** table.

- **`book_order`** has its primary key as ISBN, order_id as a book can be associated to one order only once, if a user does order one book many times in one order the quantity attribute is increased to reflect this.
  Example:

  `book_order`

ISBN	Order_id	Quantity
2	13	5

  No other tuple is needed to express the book with ISBN 2 ordered in order_id#13 has been ordered.

COMP 3005 : Final Project

Group Members : Suhail Mohamed, Abdallah El-Dali

2.3 Normalization of Relation Schemas:

Normalizing the **book** relation:

book(*ISBN, title, number_pages, publisher_email, cost, price, percent_sale, quantity*)

ISBN	Title	Num Pages	Publisher email	cost	price	Percent sale	Quantity
1	Hunger Games	256	publisher@gmail.com	15$	18$	0.78	13
2	Ender's Game	183	publisher@gmail.com	13$	16$	0.66	34
3	Harry Potter	221	NewPublisher@gmail.com	11$	14$	0.55	21
4	1984	221	NewPublisher@gmail.com	13$	16$	0.66	34
5	Ender's Game	186	NewPublisher@gmail.com	11$	15$	0.44	12

F = {

$ISBN \rightarrow ISBN, title, number\_pages, publisher\_email, cost, price, percent\_sale, quantity$

}

Testing if $ISBN$ ***is a superkey: calculating*** $\Rightarrow (ISBN)^+$

Result = $ISBN$

$ISBN \rightarrow ISBN, title, number\_pages, publisher\_email, cost, price, percent\_sale, \ldots$

$ISBN$ is a subset of Result, so add $ISBN, title, number\_pages, publisher\_email, cost, \ldots$ to Result.

Result = $ISBN, title, number\_pages, publisher\_email, cost, price, percent\_sale, \ldots$

$(ISBN)^+ = ISBN, title, number\_pages, publisher\_email, cost, price, percent\_sale, quantity$

$(ISBN)^+ = R$

$ISBN$ is a superkey

ISBN is the superkey of the relation as it uniquely identifies all attributes of the table. Since the only functional dependencies (worth showing, of course there are trivial dependencies we haven't shown) in F are dependencies that include ISBN, our superkey, in α, we can say our relation is in BCNF form, as apart from any trivial dependencies, the only dependencies α → that exist within our relation have ISBN included as a value in α.

COMP 3005 : Final Project

Group Members : Suhail Mohamed, Abdallah El-Dali

Normalizing the **order** relation:

order(*order_id, customer_email, full_address, card_number, day, month, year*)

Order_id	Customer Email	Full Address	Card Number	Day	Month	Year
1	Snoop @Gmail.com	Compton, Ca	12345678	08	03	2014
2	Snoop @Gmail.com	Hollywood, Ca	78910112	07	01	2020
1	Snoop @Gmail.com	Compton, Ca	12345678	08	03	2014
3	Kanye @Gmail.com	Hollywood, Ca	78910112	07	01	2020

F = {

$order\_id \rightarrow order\_id, customer\_email, full\_address, card\_number, day, month, year$

}

Testing if $order\_id$ ***is a superkey: calculating*** $\Rightarrow (order\_id)^+$***:***

Result = $order\_id$

$order\_id \rightarrow order\_id, customer\_email, full\_address, card\_number, day, month,$ ...

$order\_id$ is a subset of Result, so add $order\_id, customer\_email, full\_address,$ ... to Result.

Result = $order\_id, customer\_email, full\_address, card\_number, day, month, year$

$(order\_id)^+ = order\_id, customer\_email, full\_address, card\_number, day, month, year$

$(order\_id)^+ = R$

$order\_id$ is a superkey

Like in the book relation, order has only 1 functional dependency that exists within it that isn't trivial. The one relation that exists within has α as a superkey. Because of this the relation order is in BCNF form as all dependencies within it are either trivial or have α being a superkey.

COMP 3005 : Final Project

Group Members : Suhail Mohamed, Abdallah El-Dali

Normalizing **owner/customer** relations:

Owner (*owner_email* , *password, name, card_number, full_address, phone_number*)
customer(*customer_email*, *password, name, card_number, full_address, phone_number*)

Owner/ Customer Email	Password	Name	Card Number	Full address	Phone Number
SpongeBob @gmail.com	FryCook	SpongeBob SquarePants	12345678	43 st, Bikini Bottom	4561234561
Patrick@gmail.com	Rock	SpongeBob SquarePants	12345678	41 st, Bikini Bottom	4561234561
Gary @gmail.com	Rock	Gary the Snail	34512312	43 st, Bikini Bottom	8971234561

F = {

$owner/customer\_email \rightarrow owner/customer\_email, password, name, card\_number, full\_address, phone\_number$

}

Testing if $owner/customer\_email$ ***is a superkey: calculating ⇒*** $(owner/customer\_email)^+$

Result = $owner/customer\_email$

$owner/customer\_email \rightarrow owner/customer\_email, password, name, card\_number, \ldots$

$owner/customer\_email$ is a subset of Result, so add $owner/customer\_email, password, name, \ldots$ to Result.

Result = $owner/customer\_email, password, name, card\_number, full\_addres, phone\_number$

$(owner/customer\_email)^+ = owner/customer\_email, password, name, card\_number, \ldots$

$(owner/customer\_email)^+ = R$

$owner/customer\_email$ is a superkey

In either customer or owner the only non-trivial relation in it is in BCNF form, with the reasoning being the same as why order and book are in BCNF, the only dependency that exist within it are either trivial or have α being a superkey.

COMP 3005 : Final Project

Group Members : Suhail Mohamed, Abdallah El-Dali

Normalizing **publisher** relation:

publisher(*publisher_email*, *name*, *card_number*, *full_address*, *phone_number*)

F = {

$publisher\_email \rightarrow publisher\_email, name, card\_number, full\_address, phone\_number$

}

Testing if $publisher\_email$ ***is a superkey: calculating ⇒*** $(publisher\_email)^+$

Result = $publisher\_email$

$publisher\_email \rightarrow publisher\_email, name, card\_number, full\_address, phone\_number$

$publisher\_email$ is a subset of Result, so add $publisher\_email$ to Result

Result = $publisher\_email, name, card\_number, full\_address, phone\_number$

$(publisher\_email)^+ = publisher\_email, name, card\_number, full\_address, phone\_number$

$(publisher\_email)^+ = R$

$publisher\_email$ is a superkey

This table has only one functional dependency in it that isn't trivial, this functional dependency has α as a superkey. Because of this, the relation is in BCNF form as all dependencies within it are either trivial or have α as a superkey

Normalizing **author** relation:

author(*author_id*, *name*)

F = {

$author\_id \rightarrow author\_id, name$

}

Testing if $author\_id$ ***is a superkey : calculating ⇒*** $(author\_id)^+$

Result = $author\_id$

$author\_id \rightarrow name$

$author\_id$ is a subset of Result, so add $author\_id, name$ to Result

Result = $author\_id, name$

$(author\_id)^+ = (author\_id, name)$

$(author\_id)^+ = R$

$(author\_id)^+$ is a superkey

This table has only one functional dependency in it that isn't trivial, this functional dependency has α as a superkey. Because of this the relation is in BCNF form as all dependencies within it are either trivial or have α as a superkey.

COMP 3005 : Final Project

Group Members : Suhail Mohamed, Abdallah El-Dali

Normalizing **basket** relation:

basket(*customer_email*, *ISBN*, *quantity*)

Customer_email	ISBN	Quantity
Snoop@Gmail.com	1	5
Snoop@Gmail.com	2	10
Kanye@Gmail.com	1	6

F = {

$customer\_email,\ ISBN \rightarrow customer\_email,\ ISBN,\ quantity$

}

Testing if $customer\_email,\ ISBN$ ***is superkey: calculating*** $\Rightarrow (customer\_email,\ ISBN)^+$

Result = $(customer\_email,\ ISBN)$

$customer\_email,\ ISBN \rightarrow customer\_email,\ ISBN,\ quantity.$

$customer\_email,\ ISBN$ is a subset of Result, so add $customer\_email,\ ISBN,\ quantity$ to Result.

Result = $(customer\_email,\ ISBN,\ quantity)$

$(customer\_email,\ ISBN)^+ = (customer\_email,\ ISBN,\ quantity)$

$(customer\_email,\ ISBN)^+ = R$

$customer\_email,\ ISBN$ is a superkey

This table has only one functional dependency in it that isn't trivial, this functional dependency has α as a superkey. Because of this, the relation is in BCNF form as all dependencies within it are either trivial or have α as a superkey.

COMP 3005 : Final Project

Group Members : Suhail Mohamed, Abdallah El-Dali

Normalizing **book_order** relation:

book_order(*ISBN*, *order_id*, *quantity*)

ISBN	Order_id	Quantity
1	2	5
1	3	5
2	2	3

F = {

$ISBN,\ order\_id \rightarrow ISBN,\ order\_id,\ quantity$

}

Testing if $ISBN,\ order$ ***is a superkey : calculating*** ⇒ $(ISBN,\ order)^+$

Result = $ISBN,\ order$

$ISBN,\ order\_id \rightarrow ISBN,\ order\_id,\ quantity$

$ISBN,\ order\_id$ is a subset of Result. So add $ISBN,\ order\_id,\ quantity$ to Result.

Result = $ISBN,\ order\_id,\ quantity$

$(ISBN,\ order)^+ = ISBN,\ order\_id,\ quantity$

$(ISBN,\ order)^+ = R$

$ISBN,\ order$ is a superkey

This table has only one functional dependency in it that isn't trivial, this functional dependency has α as a superkey. Because of this, the relation is in BCNF form as all dependencies within it are either trivial or have α as a superkey.

COMP 3005 : Final Project

Group Members : Suhail Mohamed, Abdallah El-Dali

Normalizing **book_genre** relation:

book_genre(*ISBN*, *genre_type*)

ISBN	genre_type
1	Horror
2	Romance
1	Fantasy

F = {

$ISBN, genre\_type \rightarrow ISBN, genre\_type$

}

This table only has trivial functional dependencies within it, because of this it is in BCNF form.

Normalizing **book_author** relation:

book_author(*ISBN*, *author_id*)

ISBN	Author_id
1	4
1	8
2	4

F = {

$ISBN, author\_id \rightarrow ISBN, author\_id$

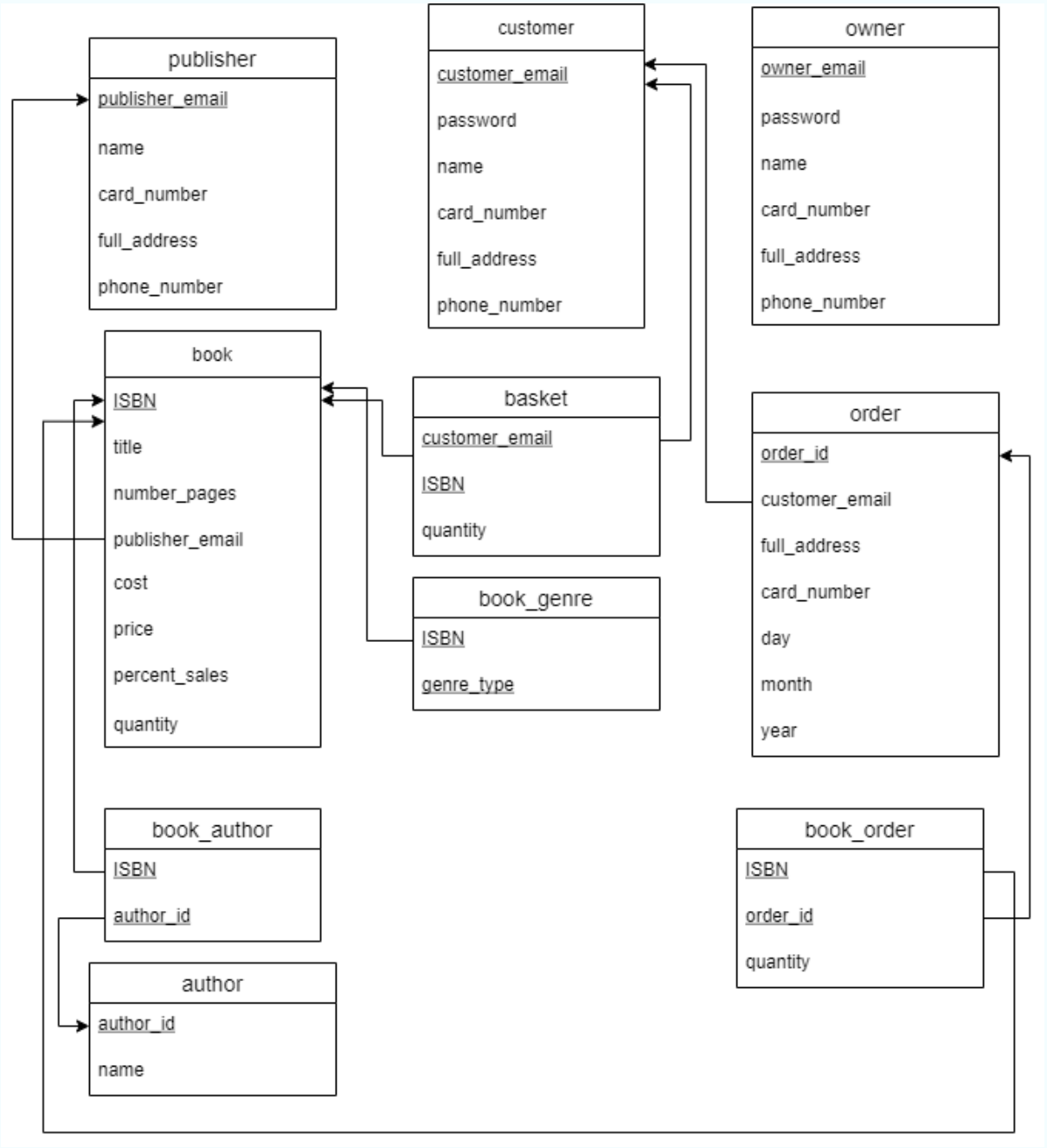
}

This table only has trivial functional dependencies within it, because of this it is in BCNF form.

COMP 3005 : Final Project

Group Members : Suhail Mohamed, Abdallah El-Dali

2.4 Database Schema Diagram:



COMP 3005 : Final Project

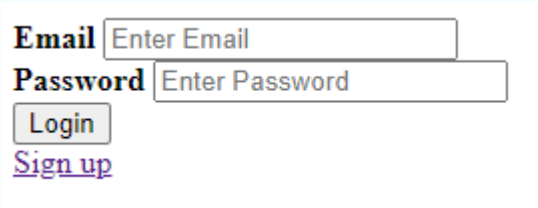
Group Members : Suhail Mohamed, Abdallah El-Dali

2.5 Implementation:

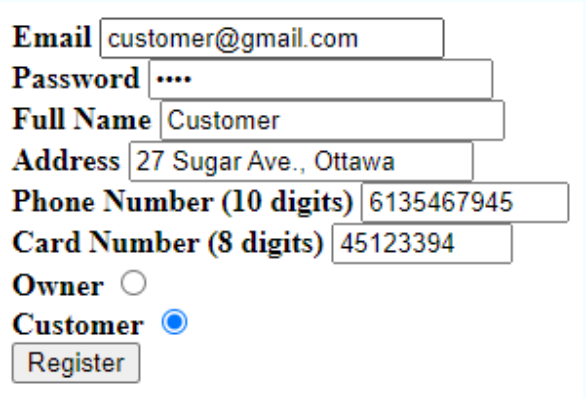
For this project we decided to create a web-based application using Typescript as the main language and NodeJS as our runtime engine. We used 'node-postgres' which is a NodeJS module that provides an API to interact with Postgres Database. We used this module extensively in our own module, database.ts, which is a script that contains most of the queries to the database needed for the application. For example, we created a function called 'getAllBooks()' which queries the database through the node-postgres API to get a table with all the books stored in the database.

The rest of the application, like the back-end, then uses this database.ts script to query all the necessary information, like checking if a customer exists, make an order, create a book, etc.

Login interface:



Registration (by clicking 'Sign up' above):



COMP 3005 : Final Project

Group Members : Suhail Mohamed, Abdallah El-Dali

Bookstore

Book Store

Orders
Basket
ISBN (digits only):
Title:
Genre:
Search

ISBN	Title	Genre(s)	Number of pages	Author(s)	Publisher	Price	Quantity	
2	Games Spice in Space	Fantasy,Horror,Romance,Biography,Fiction	314	Edgar Allen Poe	MeraglioRuissellJJr	$67.43	15	1 Add to Basket
3	Games Pear Attack	Biography,Apocalypse	682	HP Lovecraft	CityJudge	$26.07	15	1 Add to Basket
4	Hot Spice Apocalypse	Comedy,Romance,Non-fiction,Fiction,Fantasy	989	HP Lovecraft	TarkettInc	$60.24	15	1 Add to Basket
5	Good Student Disaster	Adult	904	Dr Seuss	StritzelAwningSvc	$44.68	15	1 Add to Basket
6	Hot Monkey Love	Biography	895	Stephen King	L&PConstruction	$29.89	15	1 Add to Basket
7	Big Taste Attack	Non-fiction,Adult	265	Harper Lee	StaffordWilliamPIi	$49.46	15	1 Add to Basket
8	Hot Student Escape	Apocalypse,Non-fiction,Romance,Adult	823	Edgar Allen Poe	DodsonMacPa	$41.67	15	1 Add to Basket
9	Hot Taste Apocalypse	Non-fiction,Biography,Comedy,Apocalypse	87	Stephen King,HP Lovecraft,Frank Kafka	Bacompt	$34.28	15	1 Add to Basket
10	It Hunger Love	Non-fiction,Horror,Romance	965	Frank Kafka	Century21TwinOaksRltyInc	$20.89	15	1 Add to Basket
11	Games Delish Escape	Biography	792	Stephen King	WelschMetalProductsInc	$72.98	15	1 Add to Basket
12	Good Hunger Apocalypse	Fantasy	266	Dr Seuss	CellularOne	$67.14	15	1 Add to Basket
13	Big Spice Friendship	Non-fiction,Adult	971	Ronald Dahl	CohenMortimerSEsq	$62.01	15	1 Add to Basket

Basket (empty since new user):

Basket

Orders
Bookstore

ISBN	Quantity

Address:
Card Number (8 digits):
Checkout

COMP 3005 : Final Project

Group Members : Suhail Mohamed, Abdallah El-Dali

Order (empty since new user):

Orders

Bookstore
Basket

Order Number	Address	Card Number	Date	ISBN(s)	Quantity

Searching by genre 'Horror':

Book Store

Orders
Basket
ISBN (digits only):
Title:
Genre: Horror
Search

ISBN	Title	Genre(s)	Number of pages	Author(s)	Publisher	Price	Quantity	
2	Games Spice in Space	Fantasy,Horror,Romance,Biography,Fiction	314	Edgar Allen Poe	MeraglioRuissellJJr	$67.43	15	1 Add to Basket
10	It Hunger Love	Non-fiction,Horror,Romance	965	Frank Kafka	Century21TwinOaksRltyInc	$20.89	15	1 Add to Basket
18	It Taste in Space	Comedy,Apocalypse,Horror,Romance,Fantasy	210	Frank Kafka	USTsubakiInc	$52.09	15	1 Add to Basket
21	Bro	Horror	241	Gabriel,Obama	ddvd	$25.00	254	1 Add to Basket

Basket after adding 15 books of Gamers Spice in Space (ISBN 2):

Basket

Orders
Bookstore

ISBN	Quantity
2	15

Address:
Card Number (8 digits):
Checkout

Checking out the basket with a different address than the ones I registered with:

Basket

Orders
Bookstore

ISBN	Quantity
2	15

Address: New Address, Ottawa
Card Number (8 digits): 12345678
Checkout

You order will be delivered in 10 days...

COMP 3005 : Final Project

Group Members : Suhail Mohamed, Abdallah El-Dali

Orders after checkout:

Orders

Bookstore
Basket

Order Number	Address	Card Number	Date	ISBN(s)	Quantity
21	New Address, Ottawa	12345678	Sun Dec 12 2021 00:00:00 GMT-0500 (Eastern Standard Time)	2	15

Basket after checkout (should be empty):

Basket

Orders
Bookstore

ISBN	Quantity

Address:
Card Number (8 digits):
Checkout

Checking if the book replenished itself to the number of books sold last month (including the current one):

2	Games Spice in Space	Fantasy,Horror,Romance,Biography,Fiction	314	Edgar Allen Poe	MeraglioRuissellJJr	$67.43	15	1 Add to Basket

Owner's interface (this shows a report of salers per day):

Office

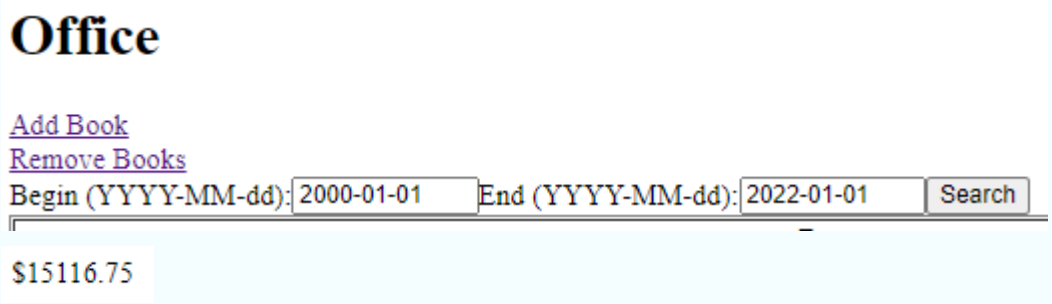
Add Book
Remove Books
Begin (YYYY-MM-dd): End (YYYY-MM-dd): Search

Date	Sales
Tue Sep 02 2008 00:00:00 GMT-0400 (Eastern Daylight Time)	$1691.72
Sat Sep 27 2008 00:00:00 GMT-0400 (Eastern Daylight Time)	$1185.33
Wed Feb 18 2009 00:00:00 GMT-0500 (Eastern Standard Time)	$498.97
Sun Apr 10 2011 00:00:00 GMT-0400 (Eastern Daylight Time)	$804.24
Sun Jun 26 2011 00:00:00 GMT-0400 (Eastern Daylight Time)	$131.9
Wed Feb 08 2012 00:00:00 GMT-0500 (Eastern Standard Time)	$
Fri Mar 22 2013 00:00:00 GMT-0400 (Eastern Daylight Time)	$
Sun Dec 15 2013 00:00:00 GMT-0500 (Eastern Standard Time)	$1385.83
Thu Feb 06 2014 00:00:00 GMT-0500 (Eastern Standard Time)	$1931.3
Wed Jun 25 2014 00:00:00 GMT-0400 (Eastern Daylight Time)	$171.2
Thu Jul 31 2014 00:00:00 GMT-0400 (Eastern Daylight Time)	$728.34
Sun Oct 05 2014 00:00:00 GMT-0400 (Eastern Daylight Time)	$167.52
Thu Nov 13 2014 00:00:00 GMT-0500 (Eastern Standard Time)	$2453.75
Mon Jan 11 2016 00:00:00 GMT-0500 (Eastern Standard Time)	$608.38
Sat Jan 30 2016 00:00:00 GMT-0500 (Eastern Standard Time)	$
Thu Apr 06 2017 00:00:00 GMT-0400 (Eastern Daylight Time)	$854.46
Mon Mar 26 2018 00:00:00 GMT-0400 (Eastern Daylight Time)	$
Mon Aug 26 2019 00:00:00 GMT-0400 (Eastern Daylight Time)	$1382.36
Tue Nov 19 2019 00:00:00 GMT-0500 (Eastern Standard Time)	$
Wed Oct 20 2021 00:00:00 GMT-0400 (Eastern Daylight Time)	$
Sun Dec 12 2021 00:00:00 GMT-0500 (Eastern Standard Time)	$1121.45

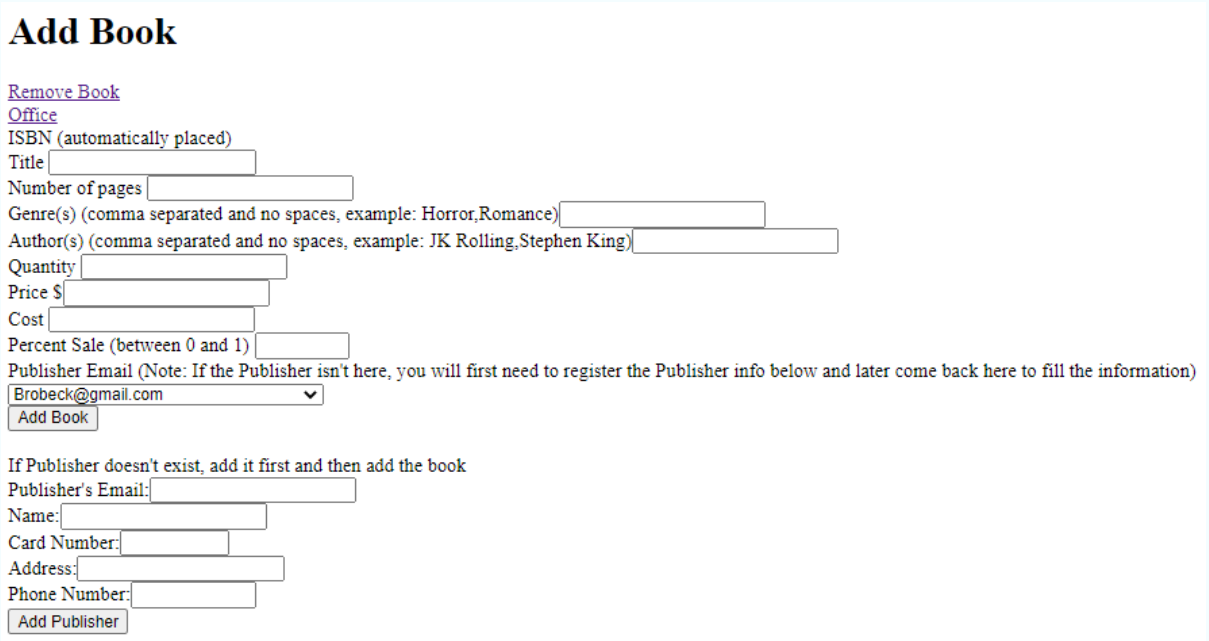
COMP 3005 : Final Project

Group Members : Suhail Mohamed, Abdallah El-Dali

Report showing the sales between 01-01-2000 to 01-01-2022 (i.e.: the sum of all the sales):



Adding a new book:



COMP 3005 : Final Project

Group Members : Suhail Mohamed, Abdallah El-Dali

After creating a new book:

Add Book

Remove Book
Office
ISBN (automatically placed)
Title Database
Number of pages 100
Genre(s) (comma separated and no spaces, example: Horror,Romance) Educational
Author(s) (comma separated and no spaces, example: JK Rolling,Stephen King) Ahmed El-Roby
Quantity 100
Price $ 100
Cost 50
Percent Sale (between 0 and 1) 0.2
Publisher Email (Note: If the Publisher isn't here, you will first need to register the Publisher info below and later come back here to fill the information)
MulliganThomasOEsq@gmail.com
Add Book

If Publisher doesn't exist, add it first and then add the book
Publisher's Email:
Name:
Card Number:
Address:
Phone Number:
Add Publisher

note: the picture below is after adding the book and going to 'Remove Book' which queries from the *book* table

22	Database	Educational	100	Ahmed El-Roby	MulliganThomasOEsq	100.00	100	Remove Book

Removing a book (the list goes on…):

Remove Books

Add Book
Office

ISBN	Title	Genre(s)	Number of pages	Author(s)	Publisher	Price	Quantity	
3	Games Pear Attack	Biography,Apocalypse	682	HP Lovecraft	CityJudge	26.07	15	Remove Book
4	Hot Spice Apocalypse	Comedy,Romance,Non-fiction,Fiction,Fantasy	989	HP Lovecraft	TarkettInc	60.24	15	Remove Book
5	Good Student Disaster	Adult	904	Dr Seuss	StritzelAwningSvc	44.68	15	Remove Book
6	Hot Monkey Love	Biography	895	Stephen King	L&PConstruction	29.89	15	Remove Book

COMP 3005 : Final Project

Group Members : Suhail Mohamed, Abdallah El-Dali

After removing the book, Games Pear Attack (ISBN 3):

Remove Books

Add Book
Office

ISBN	Title	Genre(s)	Number of pages	Author(s)	Publisher	Price	Quantity	
4	Hot Spice Apocalypse	Comedy,Romance,Non-fiction,Fiction,Fantasy	989	HP Lovecraft	TarkettInc	60.24	15	Remove Book
5	Good Student Disaster	Adult	904	Dr Seuss	StritzelAwningSvc	44.68	15	Remove Book
6	Hot Monkey Love	Biography	895	Stephen King	L&PConstruction	29.89	15	Remove Book
7	Big Taste Attack	Non-fiction,Adult	265	Harper Lee	StaffordWilliamPIi	49.46	15	Remove Book

2.7 GitHub Repository:

https://github.com/abdallah-eldali/bookstore-comp3005-project

2.8 Appendix I:

Time Slots:

2:00 pm
2:30 pm
3:00 pm
3:30 pm
4:00 pm