QSCU Merchandise Store

Document Status: Milestone 2

October 29th 2018

COSC 304 Team Members:

Team Members Removed

Table Of Contents

1. Introduction **2**
- 1.1 Mission Statement 2
- 1.2 Purpose of Document 2
- 1.3 Executive Summary 2

2. Domain Assumptions **3**
- 2.1 Products 3
- 2.2 Customers 3
- 2.3 Orders 3
- 2.4 Shipping 3
- 2.5 Warehousing 3

3. UML Diagram **4**
- 3.1 UML Description 4

4. Relational Schema **5**
- 4.1 SQL DDL 5
- 4.2 Entity Description 8
- 4.3 Relational Schema 11

5. Website Interface **11**
- 5.1 Site Map 12
- 5.2 Explanation of outline 13

6. Going Forward **13**
- 6.1 Known issues 14

1. Introduction

1.1 Mission Statement

Our goal is to create a convenient and simpler way to purchase *Quantitative Science Course Union (QSCU)*'s merchandise - the most premium clothing and party accessories in Kelowna! Additionally, we aim to provide first-rate customer service and ensure QSCU's branding expands within the region.

1.2 Purpose of Document

This document highlights the design and concepts for QSCU's merchandise store. A website interface and and UML diagram will also be outlined in this document.

1.3 Executive Summary

Since 2011, the *Quantitative Science Course Union (QSCU)* has been representing UBC Okanagan students in Mathematics, Computer Science, Statistics, Physics and Data Science. The QSCU organizes both fun and educational events while also assisting first year students in their calculus and programming classes. These events are usually held on campus, however the course union strives to bring students and professors together in casual settings in hopes of creating social networking and research opportunities.

Shop.qscu.org will serve as a merchandise store for the Quantitative Science Course Union (**qscu.org**). Through this website, the course union will have the ability to advertise and sell their branded merchandise while also creating a platform to showcase previous years' t-shirts.

Some of the available merchandise are as the following:

- T-Shirts (S - XL, Various colours)
- Hoodies (S - XL, Various colours)
- Lanyards (1 Colour and logo)
- Ping Pongs (1 Logo)
- Solo Cups (S, M, L)
- Hats (1 Size, Multiple colours)
- Rain Jackets (S - XL, Various colours)
- And many more!

The users are required to create an account in order to purchase any of the merchandise and check out their cart. However, everyone will have the ability to browse the various items sold on **shop.qscu.org**.

Once a user has registered and has an account, they will have the ability to edit their account information such as their email, password, previous orders, default shipping address, billing address etc. When placing an order the user is provided with payment and shipping options. The user must enter valid credit card information (Name on Card, Credit Card Number, Expiry date and CCV) and select their desired shipping address. After the payment has been verified, the user is directed to a confirmation email that displays a summary of the order that has just been placed.

2. Domain Assumptions

Assumptions about our domain (shipping methods, number of stores/warehouses):

2.1 Products

- A product might be 'unpopular' (unlikely, since its QSCU merch) and thus not be in any orders
- Products can come in different size variations

2.2 Customers

- A customer can have no items or multiple items in their cart
- A customer can place multiple orders

2.3 Orders

- Each order (each order number) can contain multiple products
- Each order is placed by exactly one customer.

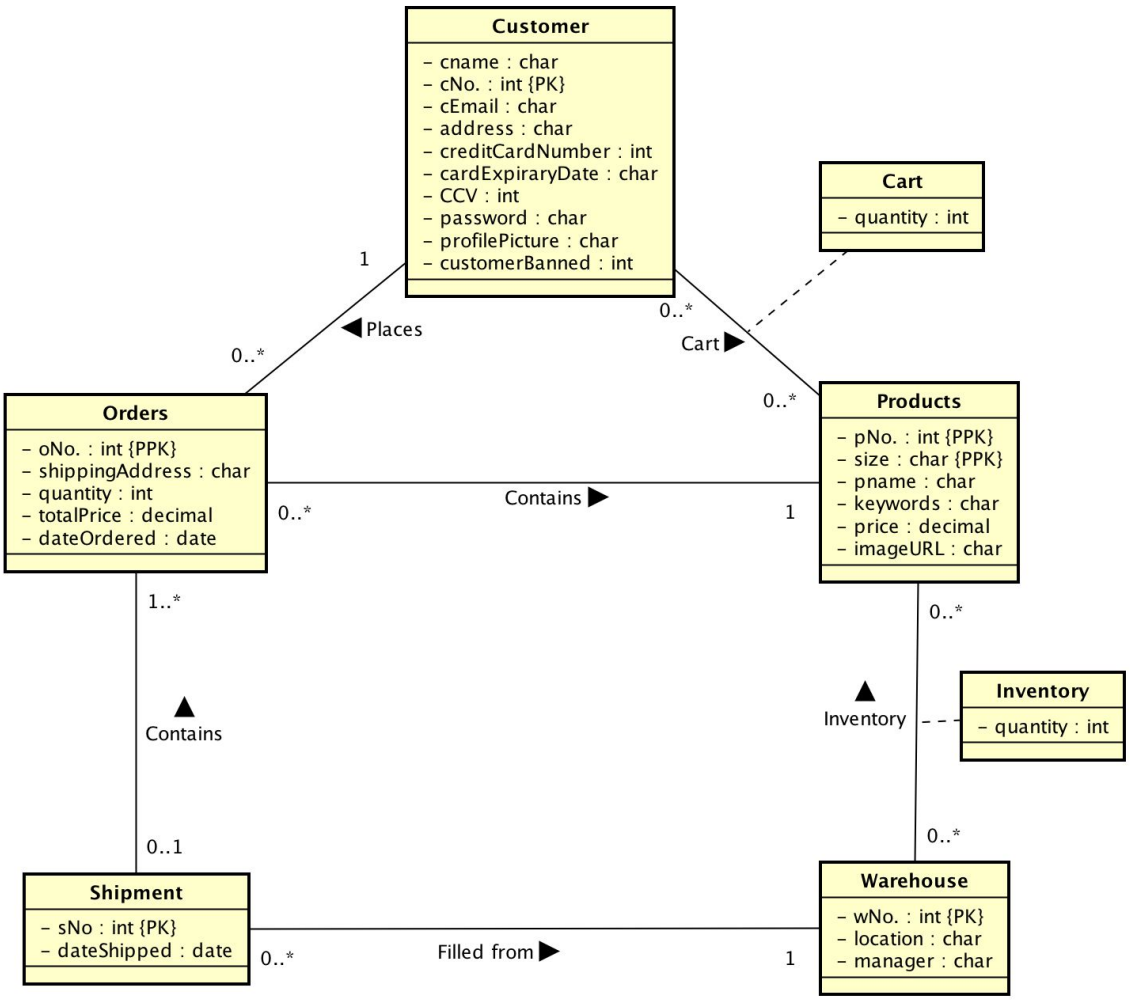
2.4 Shipping

- A product is not necessarily shipped on the same day as it is ordered
- A shipment comes from one warehouse and can contain multiple orders that go to the same one shipping address

2.5 Warehousing

- A product can be stored in multiple warehouses, or could be out of stock (in no warehouses)
- A warehouses contains multiple types of products

3. UML Diagram



3.1 UML Description

Displayed above is the UML diagram that represents our database structures

1. Customer: It is assumed that each email address entered into the database is unique, ie: no two customers can share the same email address. The customerBanned attribute will be a 0 or 1. 1 will indicate that a customer has been banned by the administrator and user will not be able to login to site, while a 0 will indicate the customer is not banned and has all the permissions of a non-administrative user.

2. Products: a product is uniquely identified by its product number and its size. For instance, we can have a purple shirt that is available in multiple sizes. Thus, to differentiate between quantity of specific sizes, it was decided that size (S, M, L) would be part of the primary key. Products that are generally considered “one size fits all” will have their size element equal to “-1”.
    2.1. Product shares a relation with customer called cart. Each customer can have a cart that contains products, and number of products.

3. Warehouse: this is where our products are stored. We assume to have more than one warehouse, hence the need for this relation.
    3.1. A warehouse shares a relation with product, called inventory. Each warehouse contains a certain amount of each product.

4. Orders: orders are placed by customers on the website. Each order has a order number, product number, and size that uniquely identifies the tuple as the primary key. This was done as an order may contain multiple different products, so while it increases the space complexity of our database, it allows us to seperate the specific products in each order.

5. Shipment: a tuple in the shipment field is created once an order is shipped from the order relation. A shipment is filled from one warehouse. It is assumed that the shipment number is the same as the corresponding order number from the Order relation.

4. Relational Schema

4.1 SQL DDL

A relational schema as constructed from your UML diagram written using SQL DDL.

```
CREATE TABLE Products (
     pNo        INTEGER,
     size       VARCHAR(3),
     pname      VARCHAR(50),
     keywords   VARCHAR(150),
     price           DECIMAL(10,2),

     PRIMARY KEY     (pNo,size)
);
```

```
CREATE TABLE Customer (
     cname          VARCHAR(40),
     cNo            INTEGER,
     cEmail         VARCHAR(30),
     address        VARCHAR(50),
     creditCardNum  INTEGER,
     creditCardCcv  INTEGER,
     creditCardDate INTEGER,
     password       VARCHAR(200),
     profilePicture VARCHAR(100),
     customerBanned INTEGER,

     PRIMARY KEY (cNo));

CREATE TABLE Warehouse (
     wNo       INTEGER,
     location  VARCHAR(50),
     manager   VARCHAR(40),

     PRIMARY KEY (wNo));

CREATE TABLE Inventory (
     quantity  INTEGER,
     pNo       INTEGER,
     size      VARCHAR(3),
     wNo       INTEGER,

     PRIMARY KEY (pNo, size, wNo),
     FOREIGN KEY (pNo, size) REFERENCES Products(pNo,size)
          ON DELETE SET NULL ON UPDATE CASCADE,
     FOREIGN KEY (wNo) REFERENCES Warehouse(wNo)
          ON DELETE SET NULL ON UPDATE CASCADE);
CREATE TABLE Shipment (
     sNo            INTEGER,
     wNo            INTEGER,
     dateShipped    DATE,
     PRIMARY KEY (sNo),
     FOREIGN KEY (wNo) REFERENCES Warehouse(wNo)
          ON DELETE NO ACTION ON UPDATE CASCADE);
```

```
CREATE TABLE Cart (
     cNo       INTEGER,
     pNo       INTEGER,
     size      VARCHAR(3),
     quantity  INTEGER,
     PRIMARY KEY (cNo, pNo, size),
     FOREIGN KEY (cNo) REFERENCES Customer(cNo)
          ON DELETE NO ACTION ON UPDATE CASCADE,
     FOREIGN KEY (pNo, size) REFERENCES Products(pNo, size)
          ON DELETE SET NULL ON UPDATE CASCADE);

CREATE TABLE Orders (
     oNo       INTEGER,
     cNo       INTEGER,
     shippingAddress VARCHAR(50),
     pNo       INTEGER,
     size      VARCHAR(3),
     quantity  INTEGER,
     totalPrice     DECIMAL(10,2),
     sNo       INTEGER,

     PRIMARY KEY (oNo, pNo, size),
     FOREIGN KEY (pNo, size) REFERENCES Products(pNo,size)
          ON DELETE SET NULL ON UPDATE CASCADE,
     FOREIGN KEY (sNo) REFERENCES Shipment(sNo)
          ON DELETE NO ACTION ON UPDATE CASCADE);
```

4.2 Entity Description

Cart

Attribute	**Description**
cNo.	Customer's ID
pNo.	The ID of a product that is in the cart
size	The size of the product that is in the cart
quantity	The quantity of the product that is in the cart

Customer

Attribute	**Description**
cNo.	Auto incremented integer ID to represent a given customer
cname	Name of customer
cEmail	User's email, used for contact and for user to login
address	User's address associated with their credit card
creditCardNumber	User's credit card number
ccv	User's credit card CCV
creditCardExpiryDate	User's credit card expiration date in form mmyy
password	User's password used to login into site
profilePicture	Path to image
customerBanned	Binary value to indicated if a customer has been banned by an administrator

Inventory

Attribute	Description
pNo.	Product ID
size	Size of product
quantity	Quantity of product in inventory
wNo.	Warehouse ID where each product of a specific size is stored

Orders

Attribute	Description
oNo.	Auto incremented ID for an order
cNo.	Customer ID
shipping Address	Destination address of order
pNo.	Product ID
size	Size of product
quantity	Quantity of product
totalPrice	Total price of order
dateOrdered	Date order was placed

Products

Attribute	Description
pNo.	Auto incremented product ID
size	Size of product
pname	Name of product
keywords	Keywords associated with product
price	Price of product
imageURL	URL of image of product

Shipment

Attribute	Description
sNo.	Shipment ID
wNo.	Id of warehouse from which shipment is being shipped
dateShipped	Date of shipment

Warehouse

Attribute	Description
wNo.	Auto incremented warehouse ID
location	Location of warehouse in the form City, country
manager	Name of manager of warehouse

4.3 Relational Schema

Customer(cname, cNo., cEmail, address, creditCardNumber, cardExpiryDate, CCV, password, profilePicture, customerBanned)
Products(pNo., size, pname, keywords, price, imageURL)
Warehouse(wNo., location, manager)
Shipment(sNo., dateShipped, *oNo.*)
Orders(oNo., shippingAddress, quantity, totalPrice, dateOrdered, *cNo., pNo., size*)
Cart(*cNo., pNo., size*, quantity)
Inventory(*wNo., pNo., size*, quantity)

Note: Foreign Keys are *italicized* and Primary Keys are underlined

5. Website Interface

Any other supporting documentation including description of web interface (if possible). As part of this, it is good if you show the list of features that you plan to support in your application and a site map

Supported Features

- ➔ User Accounts
  - ◆ Profile Picture-uploadable image MANDATORY
  - ◆ Account Info
  - ◆ Password Recovery MANDATORY
- ➔ Item Search Bar MANDATORY
  - ◆ Item Sort/Filter
- ➔ Shipping Calculator
- ➔ Best Selling Items
- ➔ Item Review Section MANDATORY
- ➔ Item Questions Section? SUPER EXTRA
- ➔ Item Rating (Out of 5 Liams) NECESSARY
- ➔ Shopping Cart MANDATORY
- ➔ Admin Users
  - ◆ Enable/disable users accounts

5.1 Site Map

Link to full site map (images provided below in case this link doesn't work properly): https://www.gloomaps.com/hwAbsYmEQe

Basic design of Web Pages

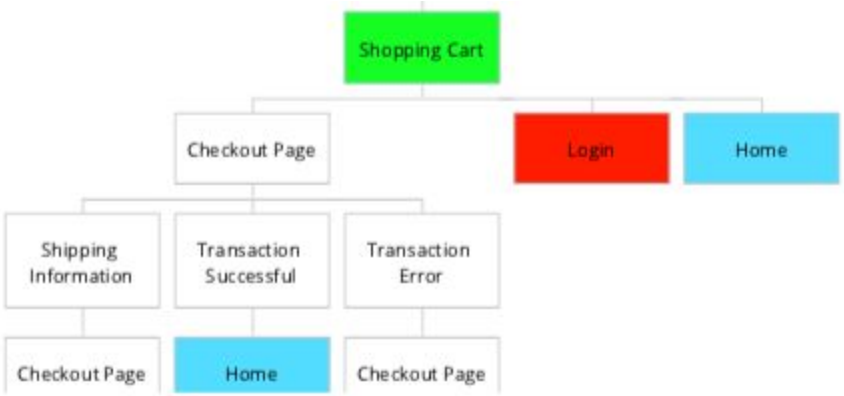


Image 1: Shopping cart branch of site map. It is a direct branch from Home.

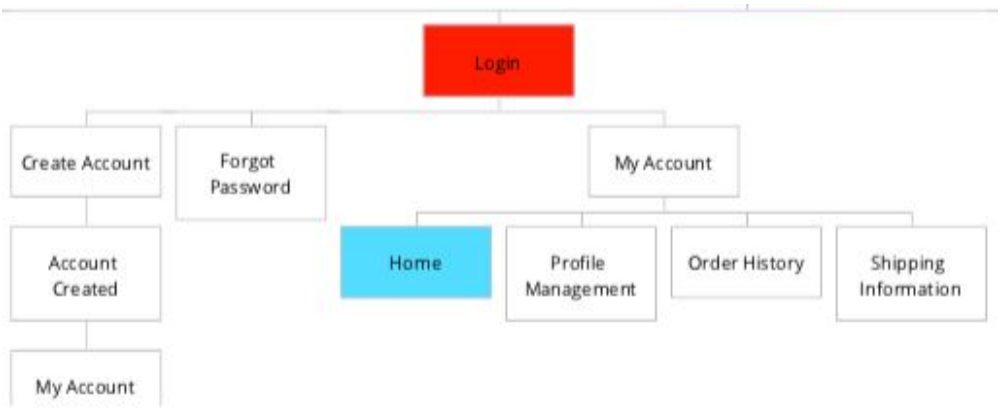


Image 2: Login branch of the site map. It is a direct branch from Home.

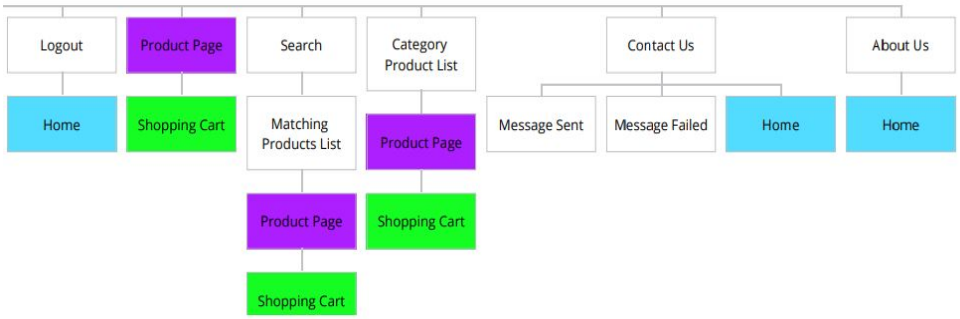


Image 3: Other branches of the site map. Top level are directly accessible from Home.

5.2 Explanation of outline

Above are three images that constitute the project sitemap.

Image 1 is the shopping cart branch. A user's shopping cart is accessible directly from the home page of the website via a shopping cart icon. Additionally, the shopping cart can be accessed from the product page as shown with the green labels in the above images. Technically, the shopping cart may also be accessed from any page on the site with a link in the header, but the sitemap would simply get messy if every link were to be shown explicitly. From the shopping cart page, a user will be prompted to login if they haven't already. If they choose to login, it will take them to the login branch. If they are logged in already they may proceed to checkout.

Image 2 is the Login branch. This branch is directly accessible from the home page, and all other pages via the header. Additionally, a user will be sent to the login branch if they try to view their cart without being logged in. At the login page, a user will either login and go to their account, create an account, or be linked to the forgotten password page.

Image 3 is all the other main branches that are accessible from the home page. Specifically, the logout page, which will then return the user to the home page, the basic product page, which can lead a user to the shopping cart branch and the contact us page which will give a user the option to send an email to the QSCU, and when they press send, will lead them to a page detailing whether the message was sent properly or if it failed.

The colours in the sitemap are simply in place to help with showing where certain pages will lead to. Each of the highest level pages in the three images are direct children of the home page.

6. Going Forward

This website will eventually be operating as part of QSCU's main website - qscu.org. Once the project has been graded, the fake data will be removed and replaced with real QSCU merchandise such as the annual t-shirts and hoodies.

This site will play a major role in the course union's sales and will be accessed by all students, faculty and staff of UBC Okanagan.

6.1 Known issues

- ➔ We must purchase a domain and an online database to store our data
- ➔ There is not a function currently implemented to email invoices to users however this can be adapted in the future