Project Design
T-Shirt Sale Website
University of British Columbia Okanagan

COSC 304 - Fall 2017

Version 2.0
Date: 10/28/2017

Table of Contents

Project Team/Contacts:

Removed

Introduction:

PleaseBuyOurTShirts.xyz is an online T-shirt store that allows customers to purchase T-shirts from our storefront and ship them to a desired address.

This document details the design guidelines for PleaseBuyOurTshirts.xyz that are to be delivered. This site is tailored towards a small company with minimal stock that wishes to establish a straightforward and efficient internet presence.

Mission Statement:

To design, implement, and build an efficient and accessible T-shirt storefront that lends itself towards a safe and fulfilling experience for buyers, administrators, and visitors. We wish to allow users to search and browse our shirts, register to buy a shirt and track the status of their items. The administrators of PleaseBuyOurTshirts.xyz can manage users, view/generate reports on analytics, maintain the site, and manage products.

Executive Summary:

PleaseBuyOurTShirts.xyz is an online marketplace where both guests, and registered users can buy unique shirts overflowing with character.

Guests may browse, add items to their cart and checkout/pay with their credit card or paypal. A user's cart will last as long as they remain in a session. The system will not attempt to save information on guest users.

Users who choose to create an account will input their preferred username (provided it is available), password, and other provided information. The system can (whether or not it's at account creation or checkout) save their name, email, shipping and billing addresses, credit card info, and other preferences. When logged in, carts will be saved for registered users until the next time they log back into the website. Upon purchasing a T-shirt, users may choose between different colors and sizes of shirt.

Users can be administrators, however, only company personnel are authorized to take this position. Administrators have the authority to perform maintenance and make changes to the prices, images and listings of products for sale.

If time/deadlines permit, deeper user tracking/product prediction will be implemented into the site. With these new statistics, reports will be drawn up based on this information and targeted advertisements will be shown.

The online T-shirt website will be modeled with an Entity-Relationship diagram that utilizes the Unified Modeling Language. The site will be implemented using Node.JS, the npm MySQL package and Express JS. PleaseBuyOurTShirts.xyz will be hosted on the company's Ubuntu Apache server. The MySQL database will be utilized as the main database for the system.

Domain Assumptions:

- **User:** User has a unique identification number (id) of type integer, unique username and email address with capacity of 50 characters. User also creates password for the specified username with maximum of 250 characters. The DB also stores firstname, lastname (of 50 characters) and the answer (of 250 characters) to the security question upon registration. At the end it also defines if the user is administrator or not.

- **Product:** Each product has a unique id of type integer, name with 250 characters, maximum price of ($ 9999999.99), size and colour of 50 characters. Products also keeps track of the amount of items left in the store, whether it is on sale or not (boolean) and what would be the sales price if it is on sale. Each product is also tagged in a category and has at least one image. The user can save product(s) in the cart section and place them in order later.

- **Shipment:** Each shipment is associated with a unique id, and method of shipping the product (e.i. Shipping to another region with airplane would be considered as air method) to the specified location. Shipment also shows the tracking status(i.e. 0 -> Not yet shipped, 1 -> shipped, 2 -> in transit, 3 -> delivered).

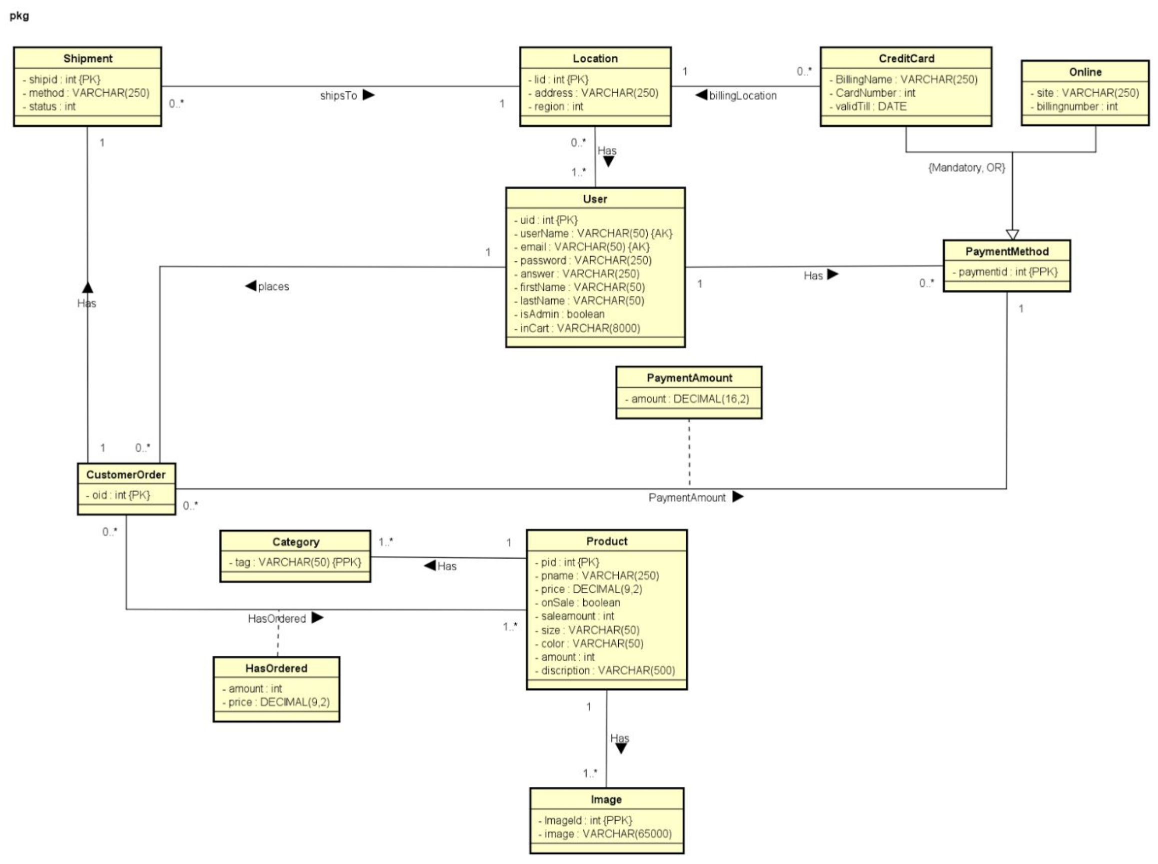
- **Location:** Location contains a unique id of type integer, the address of shipment destination, and the region of type integer predefined between zero to eleven inclusive. For the region zero to nine inclusive is assigned to the provinces of Canada, integer ten is assigned to northwest Territories, and number eleven is reserved for addresses outside Canada.

- **Payment Method:** The user can pay for the products using a credit card or an online website(i.e. Paypal,). For subclass credit card, we require billing name, card number, and expiry date of the card to process the payment. As for online websites, the website url and billing number is required to complete a payment.

- **Order:** user can keep track of each order using the unique id number assigned to each order, and it keeps track of amount of the product has been ordered.

- **Image:** Each image has an id of type integer and the image data as a blob.

Entity Description:



Relational Assumptions:

- **User:** a user can place zero to many orders.The user may also complete zero to many payments at a time. The user can place none of the visited products in order or can place all of the visited products in the order. User can have many shipment address.

- **PaymentMethod:** Payment Method has two subclasses that is (mandatory, or) which means the payment can be completed either way but it should be completed using credit card or a website.

- **Location:** A location belongs to only one user. A location can receive as many shipments as possible.

- **Shipment:** Every Shipment has one destination address, but there is only one shipment per order.

- **Order:** A single order can contain one or many amounts of products and each order has one payment.

- **CreditCard Billing:** Each bill can be send to only one destination location and each location can receive many bills.

- **Product:** Each product may have one or many images, however each image can belong to only one product. Each product can also belong to one or many category and category can contain none or many products.

Relational Schema:

```
CREATE TABLE User (
        uid             INTEGER,
        username        VARCHAR(50)       NOT NULL,
        email           VARCHAR(50)       NOT NULL,
        password        VARCHAR(250)      NOT NULL,
        answer          VARCHAR(250)      NOT NULL,
        firstName       VARCHAR(50)       NOT NULL ,
        lastName        VARCHAR(50)       NOT NULL,
        isAdmin         BOOLEAN,
        PRIMARY KEY (uid),
        UNIQUE (userName),
        UNIQUE (email),
        FOREIGN KEY ( uid ) REFERENCES hasLocation (uid )
          ON DELETE CASCADE ON UPDATE CASCADE
);

CREATE TABLE hasLocation (
        lid                    INTEGER,
        uid                    INTEGER,
        PRIMARY KEY ( uid ,lid )
);

CREATE TABLE Location (
        lid                    INTEGER,
        address                VARCHAR(250)          NOT NULL,
        region                 INTEGER               NOT NULL,
        PRIMARY KEY    (lid),
        FOREIGN KEY ( lid ) REFERENCES hasLocaiton ( lid )
          ON DELETE CASCADE ON UPDATE CASCADE
);

CREATE TABLE Product (
        pid                  INTEGER,
        pname                VARCHAR(250),
        price                DECIMAL (9,2),
        onSale               BOOLEAN,
        salesAmount          INTEGER,
        size                 VARCHAR(50),
        color                VARCHAR(50),
        amount               INTEGER,
        PRIMARY KEY (pid)
);

CREATE TABLE hasCategory (
        Tag             VARCHAR (50),
        Pid             INTEGER ,
        PRIMARY KEY ( pid , tag ),
        FOREIGN KEY ( pid ) REFERENCES Product ( pid )
          ON DELETE CASCADE ON UPDATE CASCADE
        FOREIGn KEY ( tag ) REFERENCES category ( tag)
          ON DELETE CASCADE ON UPDATE CASCADE
        );
CREATE TABLE category (
        tag                 VARCHAR( 50),
        PRIMARY KEY ( tag )
);

CREATE TABLE shipment (
        shipid              INTEGER,
        lid                 INTEGER,
        method              VARCHAR(250),
        Status              INTEGER ,
        PRIMARY KEY ( shipid),
        FOREIGN KEY ( lid ) REFERENCES Location ( lid )
          ON DELETE CASCADE ON UPDATE CASCADE

);

CREATE TABLE CreditCard (
        paymentid           INTEGER,
        lid                 INTEGER,
        BillingName         VARCHAR(250),
        cardNumber          INTEGER,
        validTill           DATE,
         PRIMARY KEY ( paymentid),
        FOREIGN KEY ( paymentid ) REFERENCES PaymentMethod ( paymentid )
          ON DELETE CASCADE ON UPDATE CASCADE,
        FOREIGN KEY ( lid ) REFERENCES Location ( lid )
           ON DELETE CASCADE ON UPDATE CASCADE
);

CREATE TABLE Online (
        paymentid           INTEGER,
        site                VARCHAR(250),
        billingNumber       INTEGER ,
        PRIMARY KEY ( paymentid )
        FOREIGN KEY ( paymentid ) REFERENCES PaymentMethod ( payment )
          ON DELETE CASCADE ON UPDATE CASCADE
);
```

```
CREATE TABLE PaymentMethod (
        uid                  INTEGER,
        paymentid          INTEGER,
        PRIMARY KEY (uid, paymentid)
        FOREIGN KEY (uid ) REFERENCES User( uid )
          ON DELETE CASCADE ON UPDATE CASCADE
);

CREATE TABLE inCart (
        uid                  INTEGER,
        pid                  INTEGER,
        amount             INTEGER,
        PRIMARY KEY (uid, pid),
        FOREIGN KEY ( uid ) REFERENCES User ( uid )
          ON DELETE CASCADE ON UPDATE CASCADE,
        FOREIGN KEY ( pid ) REFERENCES Product ( pid )
          ON DELETE CASCADE ON UPDATE CASCADE
);

CREATE TABLE Order (
        oid                   INTEGER,
        uid                   INTEGER,
        shipid               INTEGER,
        Paymentid          INTEGER,
        PRIMARY KEY (oid),
        FOREIGN KEY  (uid ) REFERENCES User (uid) ,
          ON DELETE CASCADE ON UPDATE CASCADE ,
        FOREIGN KEY ( shipid ) REFERENCES Shipment ( shipid )
          ON DELETE CASCADE ON UPDATE CASCADE,
        FOREIGN KEY ( paymentid ) REFERENCES paymentMethod ( paymentid )
           ON DELETE CASCADE ON UPDATE CASCADE
);

CREATE TABLE HasOrdered (
        oid                INTEGER,
        pid                INTEGER,
        amount            INTEGER,
        PRIMARY KEY(oid,pid),
        FOREIGN KEY(oid) REFERENCES Order(oid),
        ON DELETE CASCADE ON UPDATE CASCADE,
    FOREIGN KEY(pid) REFERENCES Product(pid)
        ON DELETE CASCADE ON UPDATE CASCADE

);

CREATE TABLE Image (
    pid           INTEGER
    ImageId       INTEGER,
    Image         blob,
    PRIMARY KEY (pid, ImageId ),
    FOREIGN KEY ( pid ) REFERENCES Product ( pid)
        ON DELETE CASCADE ON UPDATE CASCADE
);
```

Web Interface:

PleaseBuyOurTshirts.xyz's landing page consists of a navigation bar, a section based clothing description, and a footer with site info. The navigation bar will use mouse-over menus which display links relevant to the navigation text that was moused over. The navigation bar will be present on every page, allowing for consistent navigation across the site. The bar also includes mouse-over search and login elements that that expand/display the relevant fields upon activation.

The landing page will feature sections with featured shirts presented there. Each section highlights the key features of each shirt and provides a link to the shirt's respective page.

Each user has access to their account page that lists their orders with shipping status, account details, and account options. Users will be able to edit account details here and change personal information.

Administrators will have access to advanced features on the website that allow for them to manage aspects of the site. Administrators also have access to all user-level features.

Planned Features:

Users:

- Guest-checkout
- List of shipped items with delivery status
- Search by category, color, or size
- Saved carts for registered customers
- Suggested products
- Saved shipping and payment info
- T-Shirt options
- Shipping updates

Staff:

- Administrative access to the site
- System maintenance
- Report generation based on user analytics
- Inventory management
- Product Management