

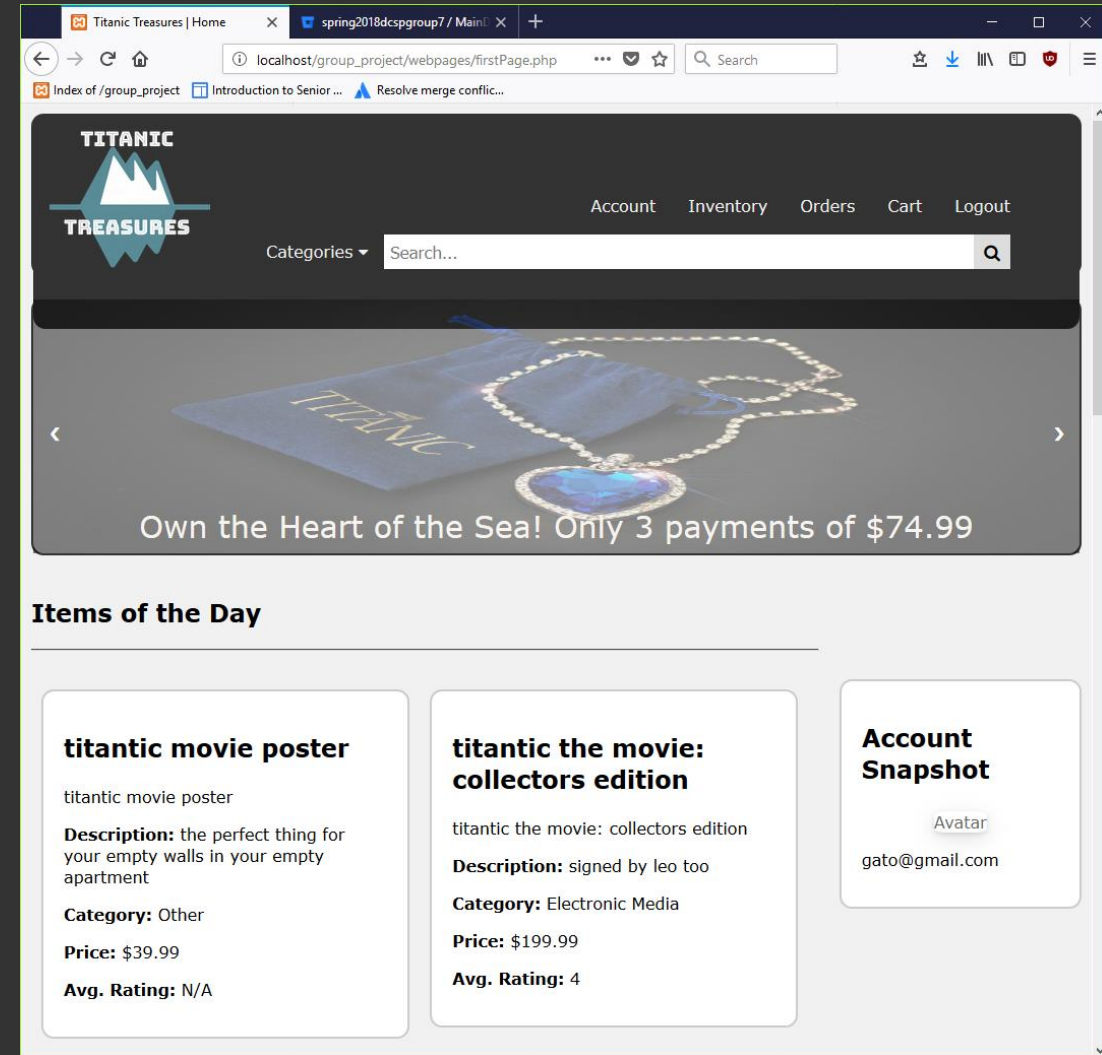


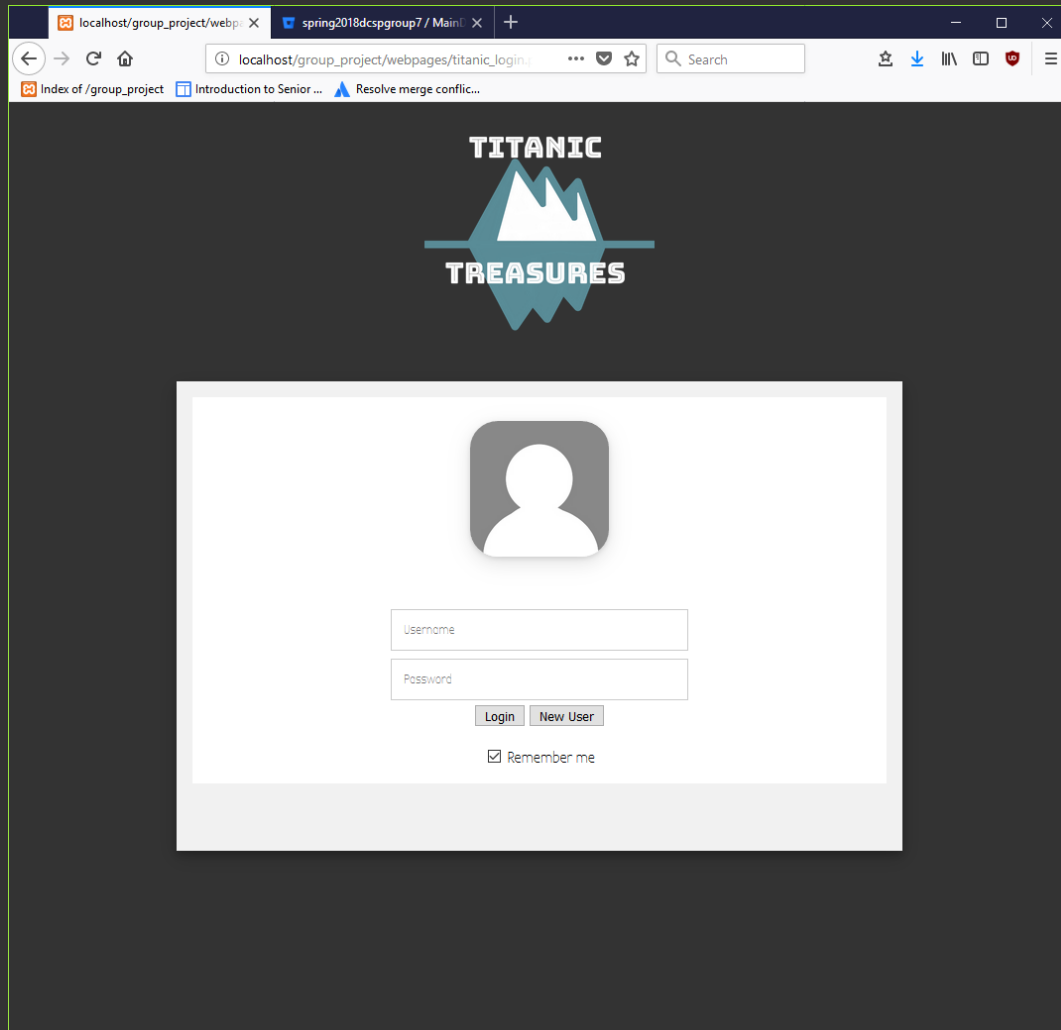
A World Where it is Titanic (the movie) All Day, Everyday



Home Page

The home page is the heart of the operation, so to speak. It is your chance to make a first impression, but also a lasting one. We ended up spending a lot of time on the design of this page so as to really make a great impression. We drew inspiration from Amazon for most of the website.





Login

Arguably the second most important page. We wanted a simple and clean design for this form similar to Amazon.



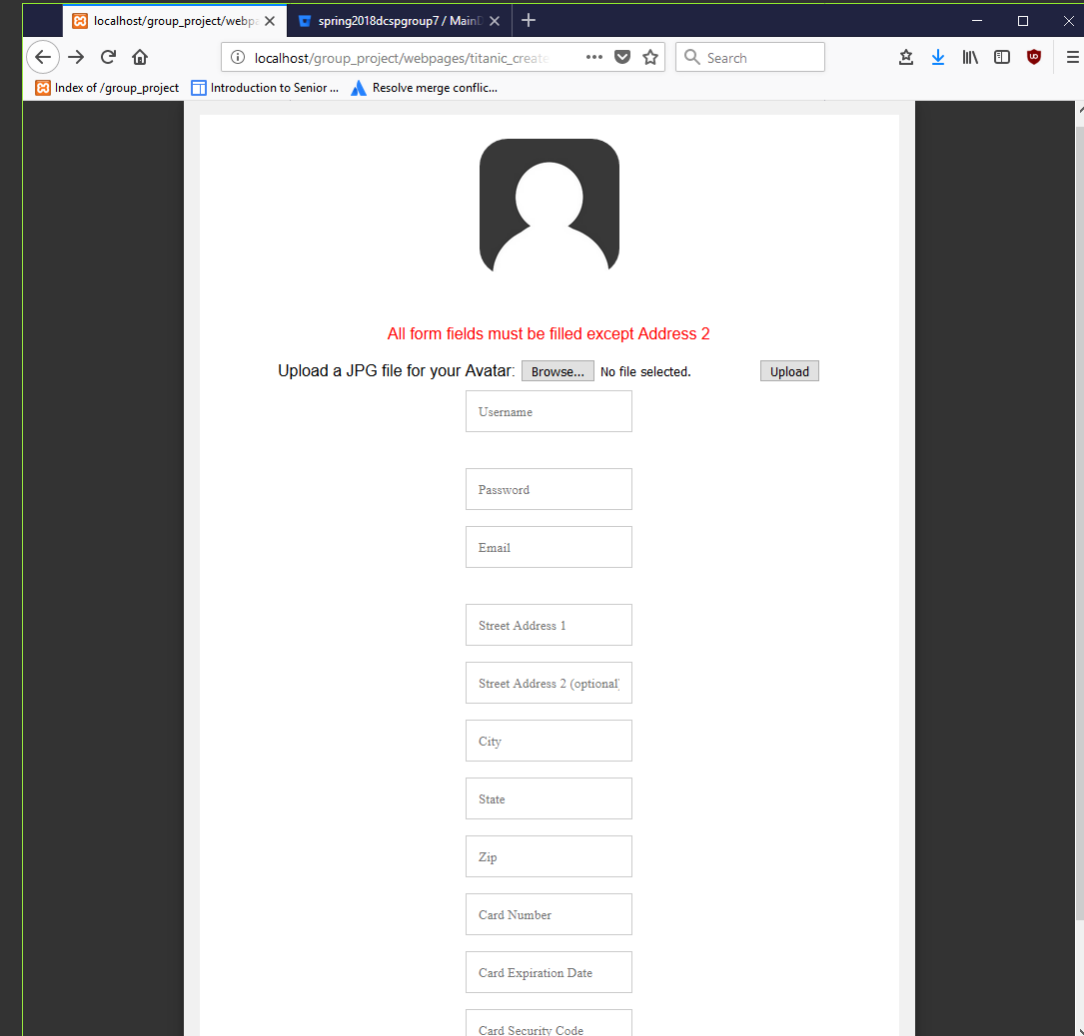
TITANIC

TREASURES

Create Account

After login, obviously comes create account. For how can you login without an account? Well, you obviously can because you have the info.


So, yeah. This is following the simple ideology of the other basic form pages. No need to clutter them up with the nav bar or anything like that.



localhost/group_project/webp... spring2018dcspgroup7 / Main... +

localhost/group_project/webpages/titanic_create... Search

Index of /group_project Introduction to Senior ... Resolve merge conflic...



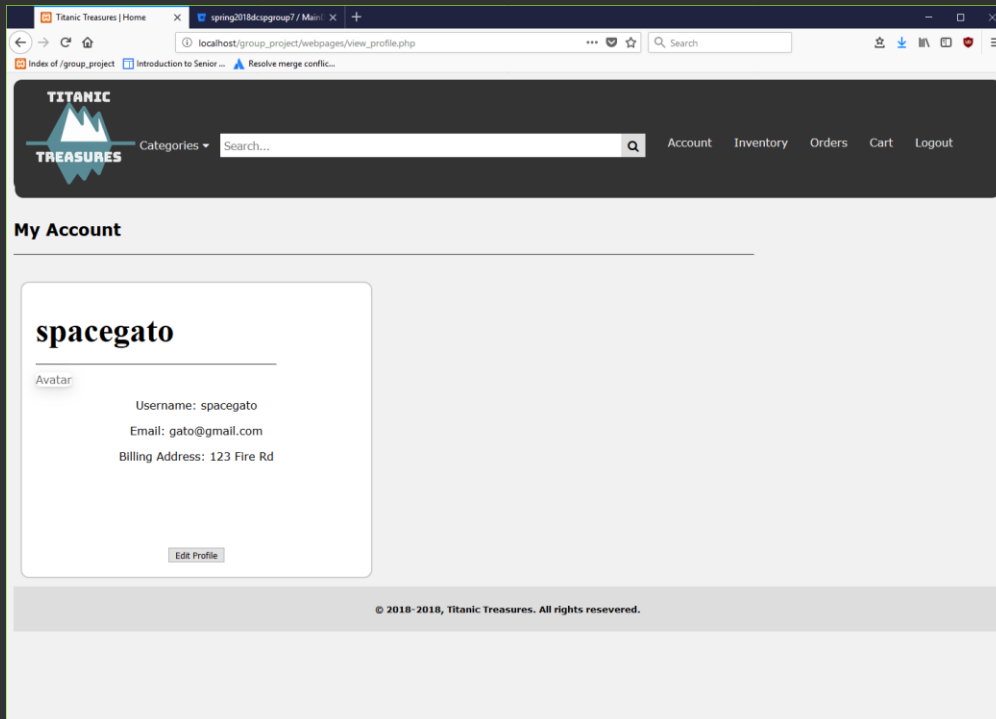
All form fields must be filled except Address 2

Upload a JPG file for your Avatar: No file selected.

TITANIC

TREASURES

View Profile



Once you login or create your account, you will be redirected to the home page. From there you can click on Account to see your profile.

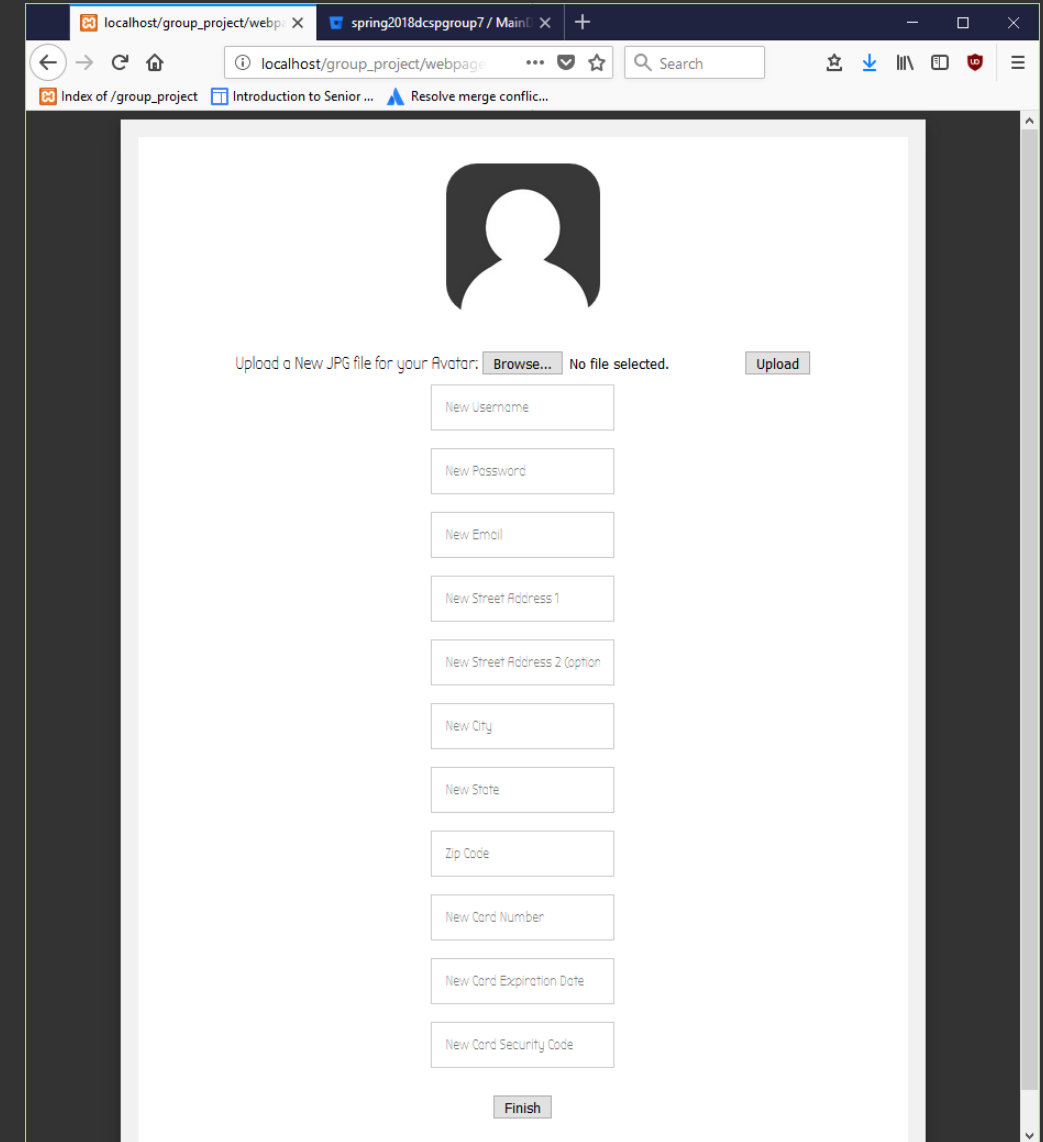
TITANIC

TREASURES

Edit Profile

Once you click on Account to view your profile there is a button to edit your profile information and/or avatar.

You can also delete your account from here, should you ever want to leave us (you won't though)...

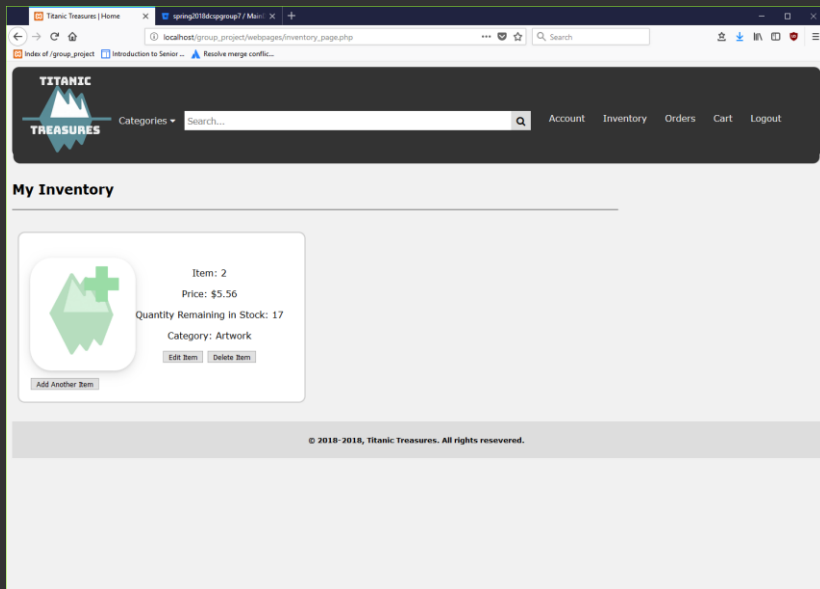


The screenshot shows a web browser window with the URL `localhost/group_project/webpage`. The page is titled "Edit Profile" and features a dark background. At the top, there is a navigation bar with links to "Index of /group_project", "Introduction to Senior ...", and "Resolve merge conflic...". Below the navigation bar, the main content area is white and contains a profile section. The profile section includes a placeholder for a profile picture (a black circle with a white silhouette) and a text input field for the "New Username". Below the username field, there are several other text input fields for "New Password", "New Email", "New Street Address 1", "New Street Address 2 (optional)", "New City", "New State", "Zip Code", "New Card Number", "New Card Expiration Date", and "New Card Security Code". At the bottom of the form, there is a "Finish" button. To the left of the form, there is a section for uploading a new avatar, which includes a "Browse..." button, the text "No file selected.", and an "Upload" button.

TITANIC

TREASURES

Inventory



All user accounts are also seller accounts. You can buy and sell to your hearts content here at Titanic Treasures.

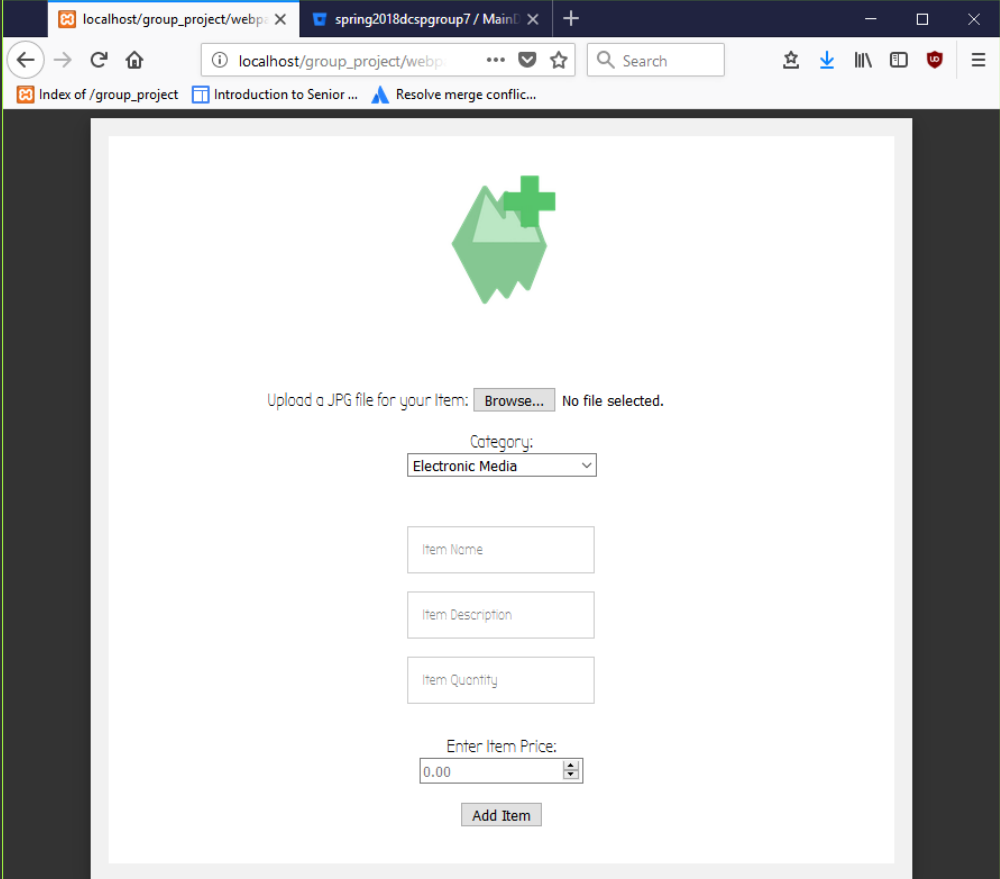
This is the page where you can view your inventory (the things people can buy from you).

TITANIC

TREASURES

Add Item


After viewing your inventory, if you discover you want to sell something else you can quickly add an item to the aforementioned inventory.



localhost/group_project/webp... spring2018dcspgroup7 / Main... +

localhost/group_project/webp... Search

Index of /group_project Introduction to Senior ... Resolve merge conflic...



Upload a JPG file for your Item: No file selected.

Category:

Enter Item Price:

Edit Item

Avatar

Upload a New JPG file for your Item: No file selected.

New Category:

Enter New Item Price:

Once an item is added to your inventory, you can edit any and all attributes of said item.

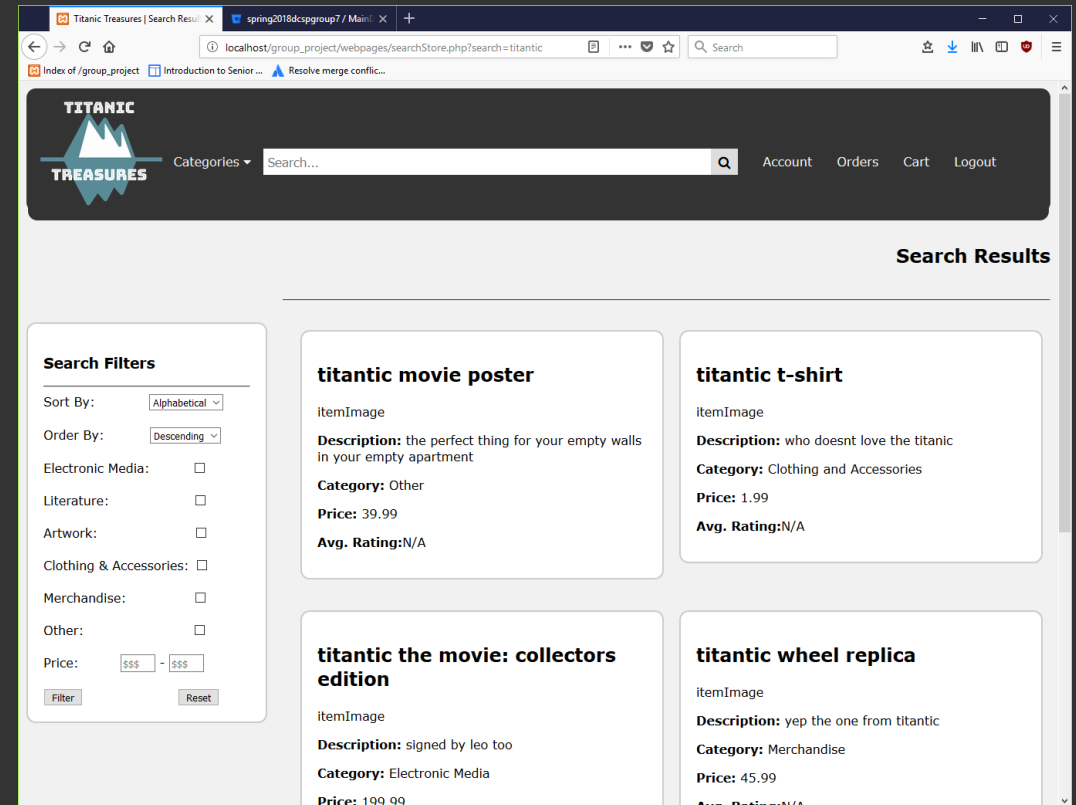
This is also where you can delete items from your inventory if you so choose to do so.



Search the Store

Finally! Let's talk about the most important algorithm of our website, the search algorithm.

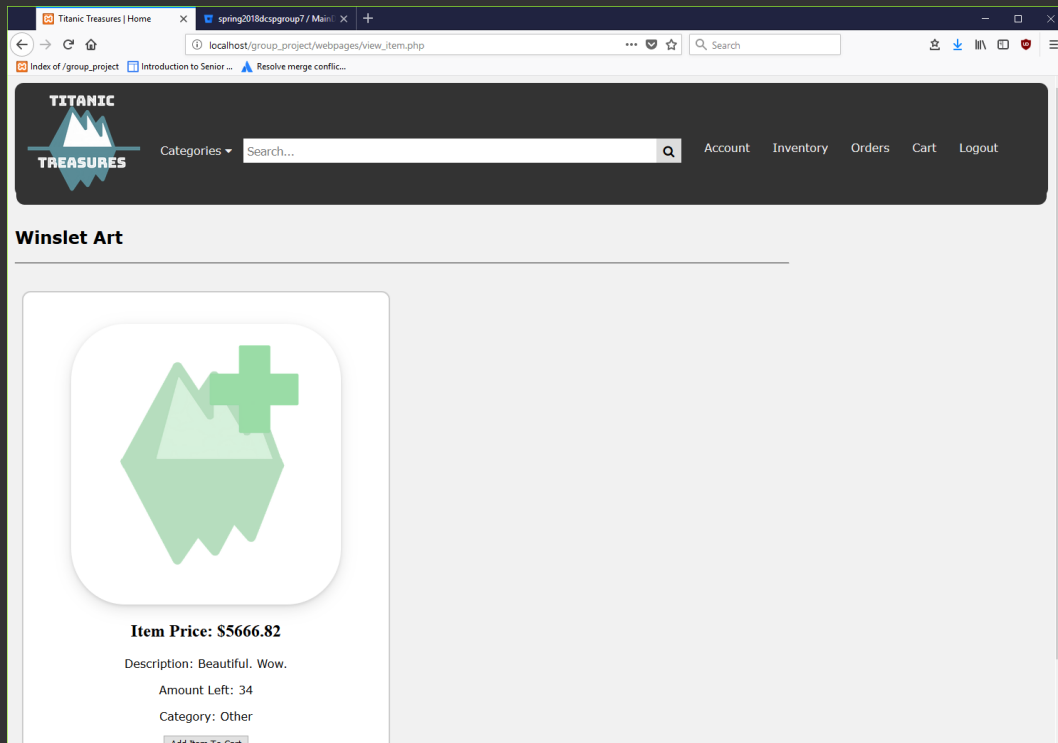
This little gem is the easiest way to scour our millions of titanic (the movie) collectibles. Search—and filter—to your heart(of the ocean)s content.



TITANIC

TREASURES

View Item



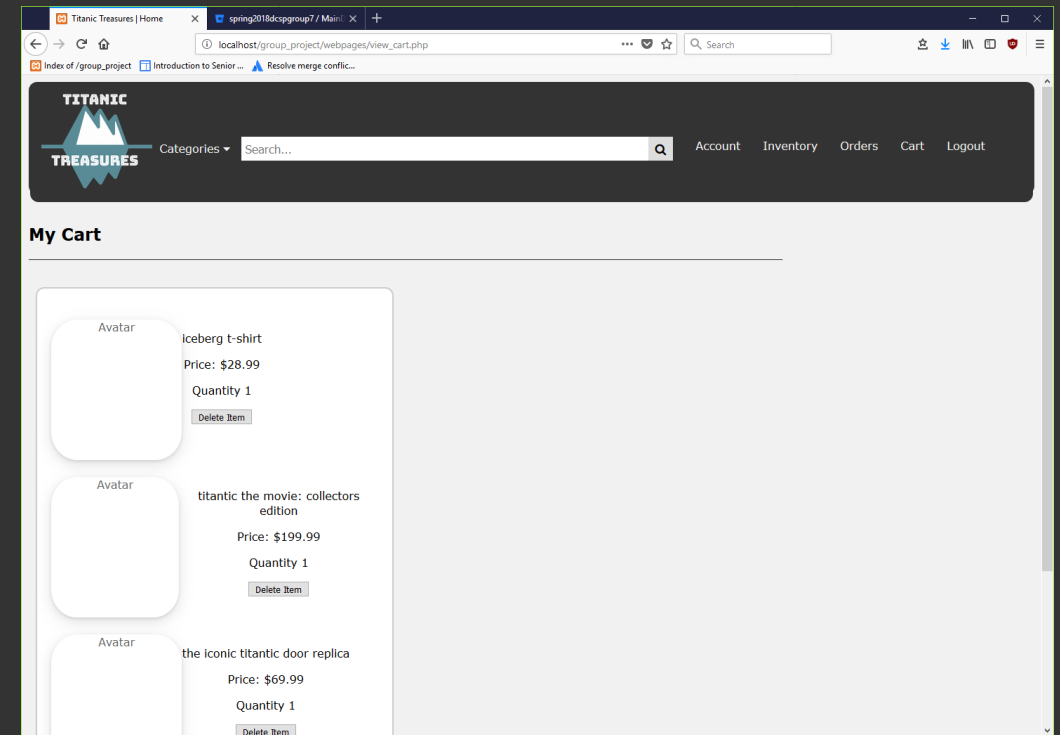
Now that you have found exactly the priceless work of art you want, you can view its glorious details and decide whether or not you want to add it to your cart.

TITANIC

TREASURES

View Cart

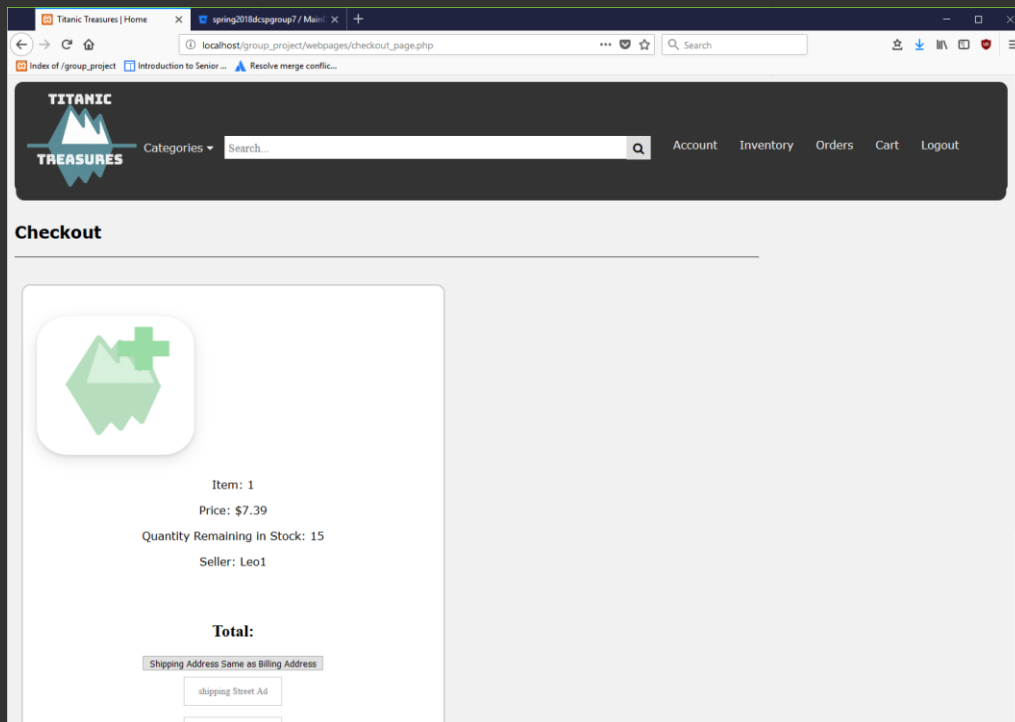
Now let us look at the treasure trove you have decided to purchase today. Let the items glisten in your virtual basket and let all the world be envious at your spoils.



TITANIC

TREASURES

Checkout



At last, the time has come! You can now purchase all of the items in your cart and have them delivered right to your doorstep¹.

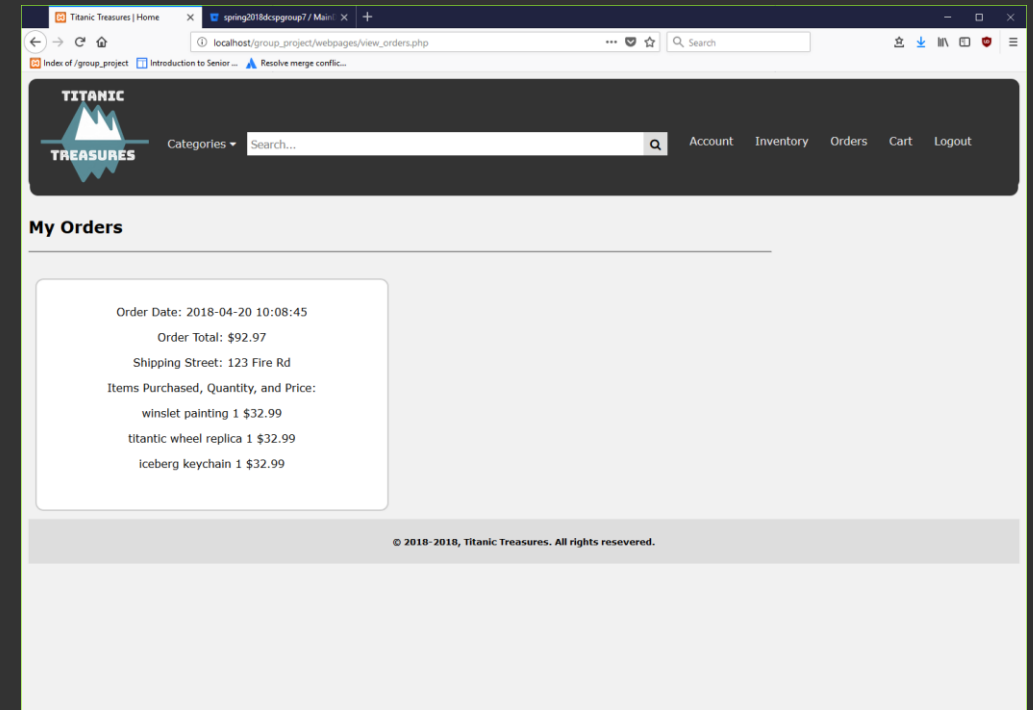
1. Nothing will be delivered. If you send us money and somehow it makes it to us you have done so as a donation for us creating this beautiful website. You're welcome.

TITANIC

TREASURES

View Orders

Finally, you can look back upon what you have purchased previously and wonder why you ever went a day without some of these items.



Backend – Login Helper

```
<?php
$GLOBALS['hn'] = 'localhost';
$GLOBALS['db'] = 'group7_project_database'; // your NetID
$GLOBALS['un'] = 'root'; // your NetID
$GLOBALS['pw'] = ''; // your MySQL password on pluto
?>
```

Generates global variables for help with database querying.

Backend – User Class

The user class employs the database by both storing local class variables with the users data (for sessions) and in the database (for security and organization).

```
1 <?php
2 class User{
3     private $userID;
4     private $isAdmin;
5     public $username;
6     public $password;
7     public $email;
8     public $billingStreetOne;
9     public $billingStreetTwo;
10    public $billingCity;
11    public $billingState;
12    public $billingZip;
13    public $avatarImg;
14    public $cardNumber;
15    public $cardExpDate;
16    public $cardSecureCode;
17
18    public function __construct(){
19        $this->username = "";
20        $this->password = "";
21        $this->email = "";
22        $this->billingStreetOne = "";
23        $this->billingStreetTwo = "";
24        $this->billingCity = "";
25        $this->billingState = "";
26        $this->billingZip = "";
27        $this->avatarImg = "";
28        $this->cardNumber = "";
29        $this->cardExpDate = "";
30        $this->cardSecureCode = "";
31    }
32
33
34    public static function createUser($username,$password,$email,$
35        $instance = new self();
36        $instance->username = $username;
37
38        $instance->email = $email;
39        $instance->billingStreetOne = $billingStreetOne;
```


Backend – Search Class

An expertly implemented search algorithm that uses both keywords and filters to find the perfect item to fit any occasion.

```
1 <?php
2 Class Search {
3     private $keywords; //list of terms searched in search bar; can be blank?
4     private $foundItemIDs; //list of items found by search algorithm and filter
5     private $priceRangeNumOne; //first num for price range filter; minimum value
6     private $priceRangeNumTwo; //second num for price range filter; maximum value
7     private $categories; //a list of categories set by the user;
8     private $descending; //value that is set to 1 if the user wants to view the ite
9     private $ascending; //value that is set to 1 if the user wants to view the item
0     private $alphabetical; //value set to 1 if the user wants to items to be sorted
1     private $numerical; //value set to 1 if the user wants the items to be sorted i
2
3     public function __construct($keywords) {
4         //inititalize class members
5         $this->keywords = $keywords;
6         $this->foundItemIDs = [];
7         $this->priceRangeNumOne = -1;
8         $this->priceRangeNumTwo = -1;
9         $this->categories = [];
0         $this->descending = 0;
1         $this->ascending = 0;
2         $this->numerical = 0;
3         $this->alphabetical = 0;
4
5         //apply a basic search filter with just the keywords
6         $this->applyFilter();
7     }
8
9     public function getKeywords() {
0         return ($this->keywords);
1     }
2
3     public function setKeywords($keywords) {
4         $this->keywords = $keywords;
5     }
6
7     public function getFoundItemIDs() {
8         return ($this->foundItemIDs);
9     }
0 }
```

Backend – Item Class

The item class uses the database to both store the item in the database and set instantiated variables to make accessing item specific qualities easy.

```
1 <?php
2 class Item {
3     private $itemName;
4     private $itemID;
5     private $itemDescription;
6     private $itemCategory;
7     //TODO: make note of deviation of design document
8     private $itemImage;
9     private $itemQuantity;
10    private $itemPrice;
11    private $ratings;
12    private $comments;
13
14    public function __construct() {
15        $this->itemName = '';
16        $this->itemID = 0;
17        $this->itemDescription = '';
18        $this->itemCategory = '';
19        $this->itemImage = '';
20        $this->itemQuantity = 0;
21        $this->itemPrice = 0.0;
22        $this->comments = [];
23        $this->ratings = [];
24    }
25
26    public static function existingItem($itemID) {
27        //create an instance of the class
28        $instance = new self();
29
30        //initialize class members
31        $instance->itemID = $itemID;
32        $instance->ratings = [];
33        $instance->comments = [];
34
35        //connect to database
36        require_once 'login.php';
37        $conn = new mysqli($GLOBALS['host'], $GLOBALS['username'], $GLOBALS['password'], $GLOBALS['db']);
38        if($conn->connect_error) die($conn->connect_error);
39
40        //if item is in items table, get class values
41        $query = "SELECT * FROM items WHERE itemID = $itemID";
```

Backend – Cart Class

```

1 <?php
2 class Cart{
3     private $itemsInCart;
4     private $cartID;
5     private $cartTotal;
6
7     public function __construct($userID)
8     {
9         //initialize class members
10        $this->itemsInCart = [];
11        $this->cartTotal = 0;
12
13        //connect to database
14        //require_once 'login.php';
15        $conn = new mysqli('localhost', 'root', 'YES', 'group7_project_database');
16        if($conn->connect_error) die($conn->connect_error);
17
18        //query database to see if the cart already exists
19        $query = "SELECT cartID FROM cart WHERE userID = '$userID'";
20        $cartIDExists = $conn->query($query);
21        if(!$cartIDExists)
22        {
23            //if it doesn't, create a new cart w/ the userID
24            $query = "INSERT INTO cart VALUES ($userID)"; //TODO: verify this is t
25            $result = $conn->query($query);
26            if(!$result) die($conn->error);
27
28            //grab new cartID from database using userID
29            $query = "SELECT cartID FROM cart WHERE userID = '$userID'";
30            $result = $conn->query($query);
31            $result->data_seek(0);
32            $this->cartID = $result->fetch_array(MYSQLI_ASSOC)['cartID'];
33        }
34        else
35        {
36            //grab cartID from database using userID
37            $result = $cartIDExists;
38            $result->data_seek(0);
39            $this->cartID = $result->fetch_array(MYSQLI_ASSOC)['cartID'];
40            //create elements for itemsInCart associative array
41            //using itemIDs as indexes and quantities as values

```

The cart class stores all of the users items in cart to the database and sets instantiated variables to make accessing and performing methods easier.

Backend – Order Class

The order class stores all of a users order in the database and allows for methods to be used to access and display said information.

```
1 <?php
2 class Order{
3     //TODO: document new class members
4     private $orderId;
5     private $orderTotal;
6     private $shippingName;
7     private $shippingStreetOne;
8     private $shippingStreetTwo;
9     private $shippingCity;
10    private $shippingState;
11    private $shippingZip;
12    private $orderDate;
13    private $itemsInOrder;
14
15    public function __construct(){
16        $this->orderId = 0.0;
17        $this->orderTotal = 0;
18        //TODO: either add a shippingName field in the orders table
19        //or delete this shippingName class member
20        $this->shippingName = '';
21        $this->shippingStreetOne = '';
22        $this->shippingStreetTwo = '';
23        $this->shippingCity = '';
24        $this->shippingState = '';
25        $this->shippingZip = 0;
26        $this->orderDate = '';
27        $this->itemsInOrder = [];
28    }
29
30
31    public static function newOrder($userID, $cartID, $shippingName, $shippingStreetOne,
32    {
33        //create new instance of the class
34        $instance = new self();
35
36        //initialize class members
37        $instance->orderTotal = 0.0;
38        $instance->shippingName = $shippingName;
39        $instance->shippingStreetOne = $shippingStreetOne;
40        if (!$shippingStreetTwo) $instance->shippingStreetTwo = NULL;
41        else $instance->shippingStreetTwo = $shippingStreetTwo;
```



References:

W3 Schools – super helpful

Textbook – helped with some class issues

The Power of Friendship – helped us keep calm and code on