|  |
| --- |
| Project Report |
| Online Auto Parts Store |
| CS 6314.002 Web Programming Languages Team 10 |

|  |
| --- |
| Kartheek Kopparapu [kxk060100]  Sriraam Ramakrishnan [sxr163730] |

Table of Contents

[1. Introduction 1](#_Toc499680163)

[2. Data Requirements 1](#_Toc499680164)

[3. Entity and Relations 1](#_Toc499680165)

[3.1 Entity 1](#_Toc499680166)

[3.2 Relations 3](#_Toc499680167)

[4. ER Diagram 4](#_Toc499680168)

[5. Relational Schema 5](#_Toc499680169)

[6. Normalization to 3NF 6](#_Toc499680170)

[6.1. Normalization in 1NF/2NF 6](#_Toc499680171)

[6.2. Normalization in 3NF 7](#_Toc499680172)

[7. SQL Code 7](#_Toc499680173)

[7.1 Tables 7](#_Toc499680174)

[8. Languages Used 11](#_Toc499680175)

[9. Screenshots 12](#_Toc499680176)

[9.1 Login Page 12](#_Toc499680177)

[9.2 Registration Page 12](#_Toc499680178)

[9.3 Buy Parts Page 13](#_Toc499680179)

[9.4 All Parts Page 13](#_Toc499680180)

[9.5 Add Part Page 14](#_Toc499680181)

[9.6 Update Part Page 14](#_Toc499680182)

[9.7 User Cart Page 15](#_Toc499680183)

[9.8 Order History Page 15](#_Toc499680184)

[10. Work Division 16](#_Toc499680185)

[10.1 Kartheek Kopparapu [kxk060100] 16](#_Toc499680186)

[10.2 Sriraam Ramakrishnan [sxr163730] 16](#_Toc499680187)

# 1. Introduction

Auto Parts Store supports customers to purchase car parts based on their car models. The system will provide information regarding the parts like company, Id, price. The system also stores customer information and can provide the history of the customer’s purchases. The customer can order any amount of any part on the website as long as the required quantity and the part is available in the store. Administrators will have special access which will allow them to add new parts, update quantities of existing parts, delete parts and bring back deleted parts (make them active again). This project implements a small version of the industry level auto parts store database.

# 2. Data Requirements

* The store should have parts to sell to customers.
* Each part should have minimum one quantity to be sold.
* Each part needs to have at least one supplier.
* Each part should have a unique identification number, should belong to a type, and have a price.
* Each part should have only one type of warranty.
* Each part should belong to the store and have their current state (active/deleted).
* When a part is sold (checked out), the status needs to be updated.
* Based on the part that was checked out by the customer, their quantity will change to display the current store inventory.
* Customers should have a unique identification number and a name.
* When a customer makes an order from the store, an address should be provided.
* When a customer makes an order from the store, the parts quantity is checked with the store inventory to not place an order that cannot be met.
* Each order generates a unique id and other order details such as part names, quantity and total price.
* The online store needs to have a warehouse and server.
* Order total cost is generated based on the parts in the order and the quantity of each part.

# 3. Entity and Relations

### 3.1 Entity

**Customer**

Customer will be the one buying the parts from the store. Customer will have attributes like Username, Fname, Lname, Street, City, State, Zipcode, Phone, and Email. Username is the key attribute that will be used to identify each customer uniquely.

**Store**

Store sells parts. There is only one online store. It has an URL and a storeID. It will be using entity WHouse that is mapped to this store using the foreign key WHouseID. Server is mapped to the online store using the foreign key ServerID.

**Orders**

An order will be generated if a customer buys parts from the online store. OrderId is key attribute which contains details like Parts, Quantity, Cost of each part and the Total cost for the order. Username and StoreID are the foreign keys.

**Part**

Parts are sold in the online store. Different car models may use different parts in one subcategory. Each of them have a unique partNo, company, price, warranty, deleted, and subcategory. Parts are mapped to WarrantyID as a foreign key. Each part has only one type of warranty.

**Warranty**

Each part has a warranty. WarrantyID is the unique attribute which mentions their type.

**Category**

Category contains a list of categories that can be applied to subcategories and each category has multiple subcategories.

**SubCategory**

Each subcategory is mapped to category using CatID as foreign key. Each subcategory can have only one category.

**Supplier**

Supplier supplies parts. One kind of part can be supplied by many suppliers. Each contains the details like their unique SupplierID, name and info.

**WHouse**

WHouse stores parts for OnlineStore. Each warehouse has unique WHouseID which is used by one single online store.

**Server**

There is only one server for online store. It has ServerID and its geographic location.

### 3.2 Relations

**places**

Maps customer to order and information is stored in the Order table. Many customers can place many orders. Only a customer can place an order.

**OInventory**

Maps order to parts and store. It stores OrderID, PartNo and OrQuantity. Each order can have many OInventory that has many parts.

**appliedTo**

Maps warranty to parts. One warranty is applied to many parts but each part has only one warranty. A part may have some warranty or have no warranty.

**belongsTo**

Maps subcategory to parts. Each part can only have one subcategory. One subcategory can have many parts.

**partOf**

Maps Category to Subcategory. Each Category can have many subcategories but one Subcategory can have only a single Category. Each subcategory must have a category.

**supplies**

Maps supplier to Parts. Supplies has attributes SupplyDate, SupplierID, PartNo, and SpQuantity. All attributes other than SpQuantity are part of key. Many suppliers can supply many parts.

**SInventory**

Maps parts to store. Has attributes StoreID, PartNo, and StQuantity. StoreID & PartNo are key.

**uses**

Maps OnlineStore to WHouse. They are one to one and required.

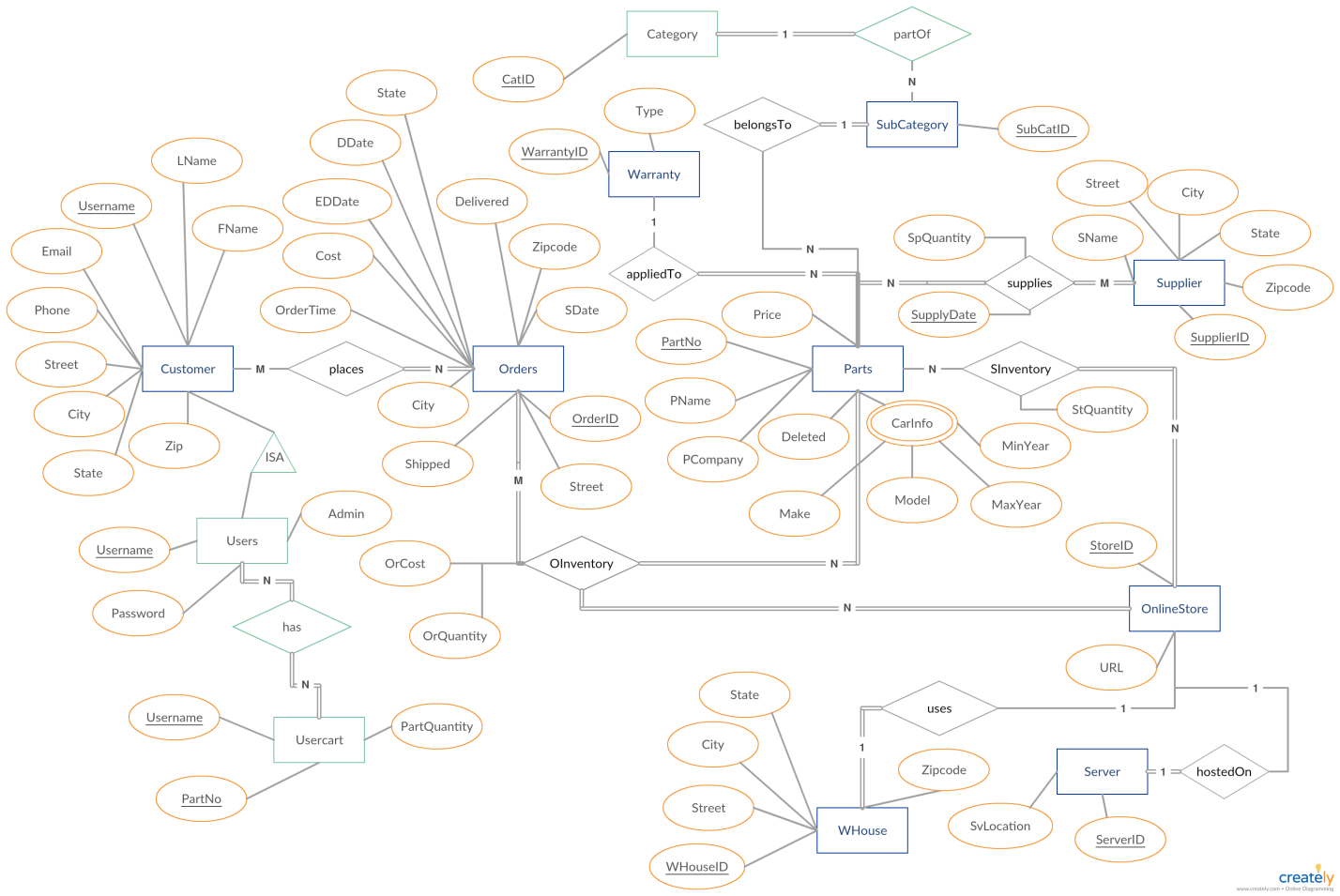
**hostedOn**

Maps OnlineStore to Server. They are one to one and required.

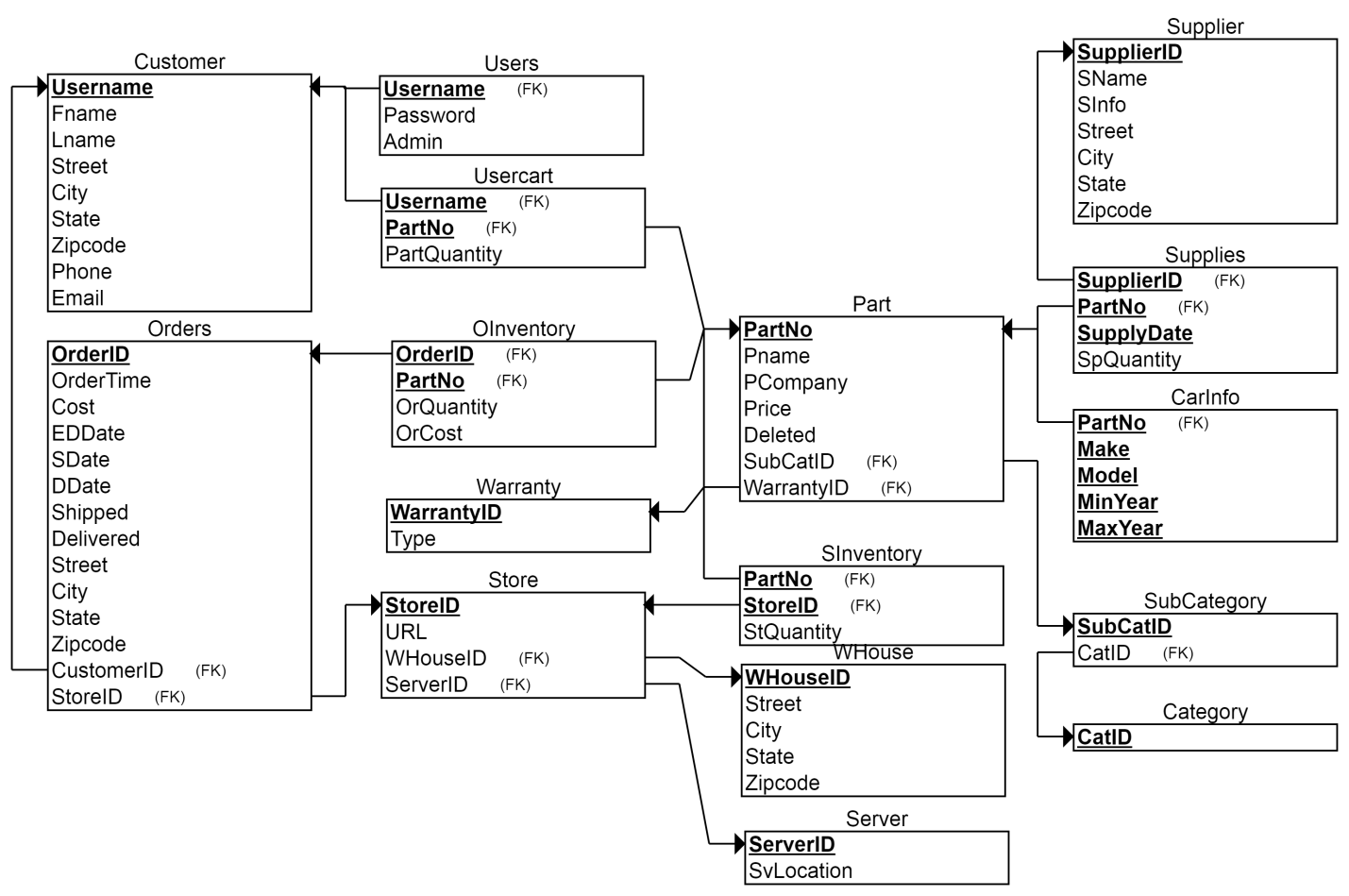
**has**

Maps Users to Useracart. They are one to one and required.

# 4. ER Diagram



# 5. Relational Schema



# 6. Normalization to 3NF

### 6.1. Normalization in 1NF/2NF

The schema is already in 2NF as there are no non-prime attributes that depend on partial prime attributes or non-prime attributes.



### 6.2. Normalization in 3NF

The only tables that needed modification to become 3NF are Customer, Order, Supplier, WHouse. The reason for this is due the transitive property where the prime attribute of each of the table give zip code that can give city & state.



# 7. SQL Code

### 7.1 Tables

DROP TABLE IF EXISTS Customer;

CREATE TABLE Customer

(

Username VARCHAR(30) NOT NULL,

Fname VARCHAR(20) NOT NULL,

Lname VARCHAR(20) NOT NULL,

Street VARCHAR(50),

City VARCHAR(50),

State VARCHAR(20),

Zipcode MEDIUMINT(5) ZEROFILL,

Phone VARCHAR(15),

Email VARCHAR(50),

PRIMARY KEY (Username)

) ENGINE=InnoDB CHARACTER SET=utf8;

DROP TABLE IF EXISTS Users;

CREATE TABLE Users

(

Username VARCHAR(30) NOT NULL,

Password VARCHAR(256) NOT NULL,

Admin TINYINT(1) NOT NULL DEFAULT '0',

PRIMARY KEY (Username),

FOREIGN KEY (Username) REFERENCES Customer(Username) ON UPDATE CASCADE,

) ENGINE=InnoDB CHARACTER SET=utf8;

DROP TABLE IF EXISTS Usercart;

CREATE TABLE Usercart

(

Username VARCHAR(30) NOT NULL,

PartNo VARCHAR(10) NOT NULL,

PartQuantity INT(11) NOT NULL DEFAULT '1',

PRIMARY KEY (Username, PartNo),

FOREIGN KEY (Username) REFERENCES Customer(Username) ON UPDATE CASCADE,

FOREIGN KEY (PartNo) REFERENCES Part(PartNo) ON UPDATE CASCADE,

) ENGINE=InnoDB CHARACTER SET=utf8;

DROP TABLE IF EXISTS Warranty;

CREATE TABLE Warranty

(

WarrantyID INT NOT NULL AUTO\_INCREMENT,

Type VARCHAR(20) NOT NULL,

PRIMARY KEY (WarrantyID)

) ENGINE=InnoDB CHARACTER SET=utf8;

DROP TABLE IF EXISTS Supplier;

CREATE TABLE Supplier

(

SupplierID INT NOT NULL AUTO\_INCREMENT,

SName VARCHAR(20) NOT NULL,

SInfo VARCHAR(20),

Street VARCHAR(50),

City VARCHAR(50) ,

State VARCHAR(20),

Zipcode MEDIUMINT(5) ZEROFILL,

PRIMARY KEY (SupplierID)

) ENGINE=InnoDB CHARACTER SET=utf8;

DROP TABLE IF EXISTS WHouse;

CREATE TABLE WHouse

(

WHouseID INT NOT NULL AUTO\_INCREMENT,

Street VARCHAR(50) NOT NULL,

City VARCHAR(50) NOT NULL,

State VARCHAR(20) NOT NULL,

Zipcode MEDIUMINT(5) ZEROFILL NOT NULL,

PRIMARY KEY (WHouseID)

) ENGINE=InnoDB CHARACTER SET=utf8;

DROP TABLE IF EXISTS Server;

CREATE TABLE Server

(

ServerID INT NOT NULL AUTO\_INCREMENT,

SvLocation VARCHAR(20) NOT NULL,

PRIMARY KEY (ServerID)

) ENGINE=InnoDB CHARACTER SET=utf8;

DROP TABLE IF EXISTS Category;

CREATE TABLE Category

(

CatID VARCHAR(50) NOT NULL,

PRIMARY KEY (CatID)

) ENGINE=InnoDB CHARACTER SET=utf8;

DROP TABLE IF EXISTS Orders;

CREATE TABLE Orders

(

OrderID VARCHAR(100) NOT NULL,

OrderTime DATETIME NOT NULL,

Cost DECIMAL(10,2) NOT NULL,

EDDate DATETIME DEFAULT '0000-00-00 00:00:00',

SDate DATETIME DEFAULT '0000-00-00 00:00:00',

DDate DATETIME DEFAULT '0000-00-00 00:00:00',

Shipped BOOLEAN NOT NULL DEFAULT 0,

Delivered BOOLEAN NOT NULL DEFAULT 0,

Street VARCHAR(50),

City VARCHAR(50),

State VARCHAR(20),

Zipcode MEDIUMINT(5) ZEROFILL,

Username INT NOT NULL,

StoreID INT NOT NULL,

PRIMARY KEY (OrderID),

FOREIGN KEY (Username) REFERENCES Customer(Username) ON UPDATE CASCADE,

FOREIGN KEY (StoreID) REFERENCES Store(StoreID) ON UPDATE CASCADE

) ENGINE=InnoDB CHARACTER SET=utf8;

DROP TABLE IF EXISTS SubCategory;

CREATE TABLE SubCategory

(

SubCatID VARCHAR(50) NOT NULL,

CatID VARCHAR(50) NOT NULL,

PRIMARY KEY (SubCatID),

FOREIGN KEY (CatID) REFERENCES Category(CatID) ON UPDATE CASCADE

) ENGINE=InnoDB CHARACTER SET=utf8;

DROP TABLE IF EXISTS Store;

CREATE TABLE Store

(

StoreID INT NOT NULL,

URL VARCHAR(50) NOT NULL,

WHouseID INT NOT NULL,

ServerID INT NOT NULL,

PRIMARY KEY (StoreID),

FOREIGN KEY (WHouseID) REFERENCES WHouse(WHouseID) ON UPDATE CASCADE,

FOREIGN KEY (ServerID) REFERENCES Server(ServerID) ON UPDATE CASCADE

) ENGINE=InnoDB CHARACTER SET=utf8;

DROP TABLE IF EXISTS Part;

CREATE TABLE Part

(

PartNo VARCHAR(10) NOT NULL,

Pname VARCHAR(50) NOT NULL,

PCompany VARCHAR(50) NOT NULL,

Price DECIMAL(10,2) NOT NULL,

SubCatID VARCHAR(50) NOT NULL,

WarrantyID INT NOT NULL,

Deleted TINYINT(1) NOT NULL DEFAULT '0',

PRIMARY KEY (PartNo),

FOREIGN KEY (SubCatID) REFERENCES SubCategory(SubCatID) ON UPDATE CASCADE,

FOREIGN KEY (WarrantyID) REFERENCES Warranty(WarrantyID) ON UPDATE CASCADE

) ENGINE=InnoDB CHARACTER SET=utf8;

DROP TABLE IF EXISTS Supplies;

CREATE TABLE Supplies

(

SupplyDate DATETIME NOT NULL,

SpQuantity INT NOT NULL,

SupplierID INT NOT NULL,

PartNo VARCHAR(10) NOT NULL,

PRIMARY KEY (SupplyDate, SupplierID, PartNo),

FOREIGN KEY (SupplierID) REFERENCES Supplier(SupplierID) ON UPDATE CASCADE,

FOREIGN KEY (PartNo) REFERENCES Part(PartNo) ON UPDATE CASCADE

) ENGINE=InnoDB CHARACTER SET=utf8;

DROP TABLE IF EXISTS SInventory;

CREATE TABLE SInventory

(

StQuantity INT NOT NULL,

StoreID INT NOT NULL,

PartNo VARCHAR(10) NOT NULL,

PRIMARY KEY (StoreID, PartNo),

FOREIGN KEY (StoreID) REFERENCES Store(StoreID) ON UPDATE CASCADE,

FOREIGN KEY (PartNo) REFERENCES Part(PartNo) ON UPDATE CASCADE

) ENGINE=InnoDB CHARACTER SET=utf8;

DROP TABLE IF EXISTS CarInfo;

CREATE TABLE CarInfo

(

Make VARCHAR(30) NOT NULL,

Model VARCHAR(30) NOT NULL,

MinYear YEAR(4) NOT NULL,

MaxYear YEAR(4) NOT NULL,

PartNo VARCHAR(10) NOT NULL,

PRIMARY KEY (Make, Model, MinYear, MaxYear, PartNo),

FOREIGN KEY (PartNo) REFERENCES Part(PartNo) ON UPDATE CASCADE

) ENGINE=InnoDB CHARACTER SET=utf8;

DROP TABLE IF EXISTS OInventory;

CREATE TABLE OInventory

(

OrQuantity INT NOT NULL,

OrderID INT NOT NULL,

PartNo VARCHAR(10) NOT NULL,

PRIMARY KEY (OrderID, PartNo),

FOREIGN KEY (OrderID) REFERENCES Orders(OrderID) ON UPDATE CASCADE,

FOREIGN KEY (PartNo) REFERENCES Part(PartNo) ON UPDATE CASCADE

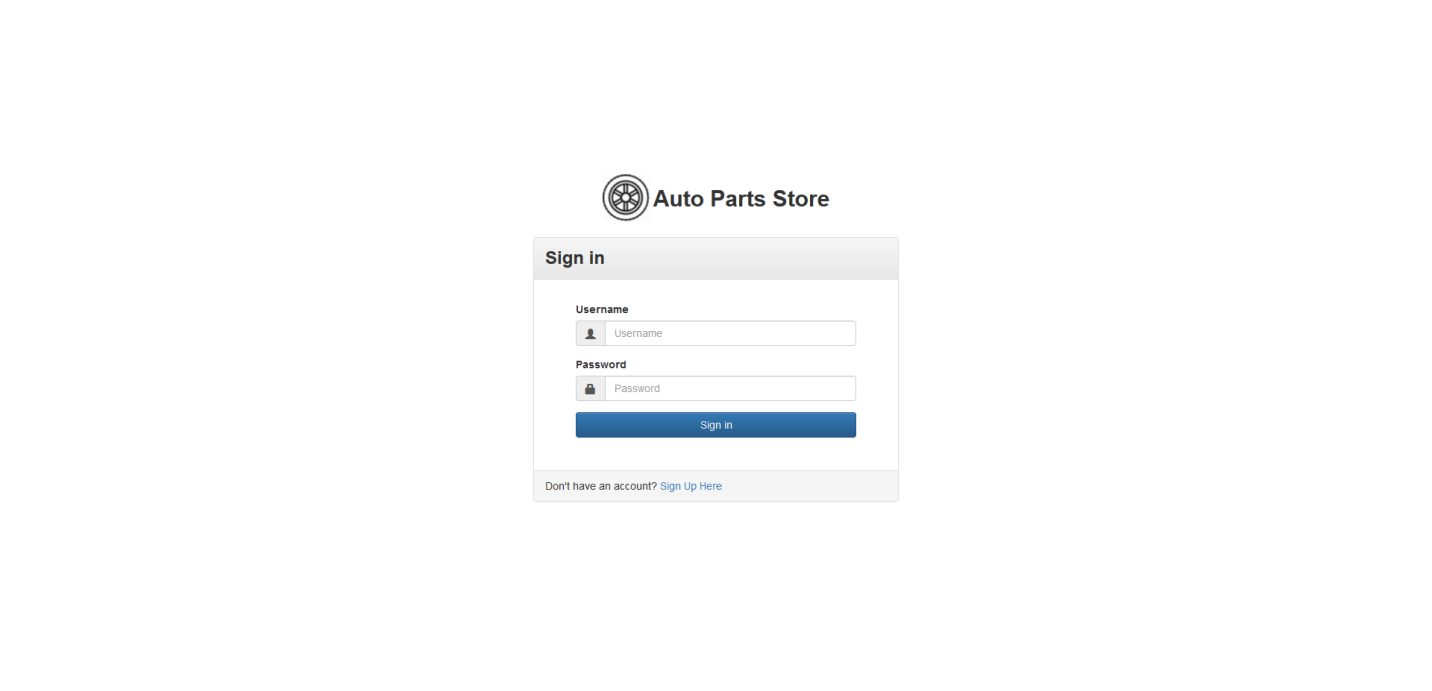
) ENGINE=InnoDB CHARACTER SET=utf8;

# 8. Languages Used

* AngularJS
* jQuery
* Bootstrap
* PHP
* MySQL Database

# 9. Screenshots

### 9.1 Login Page

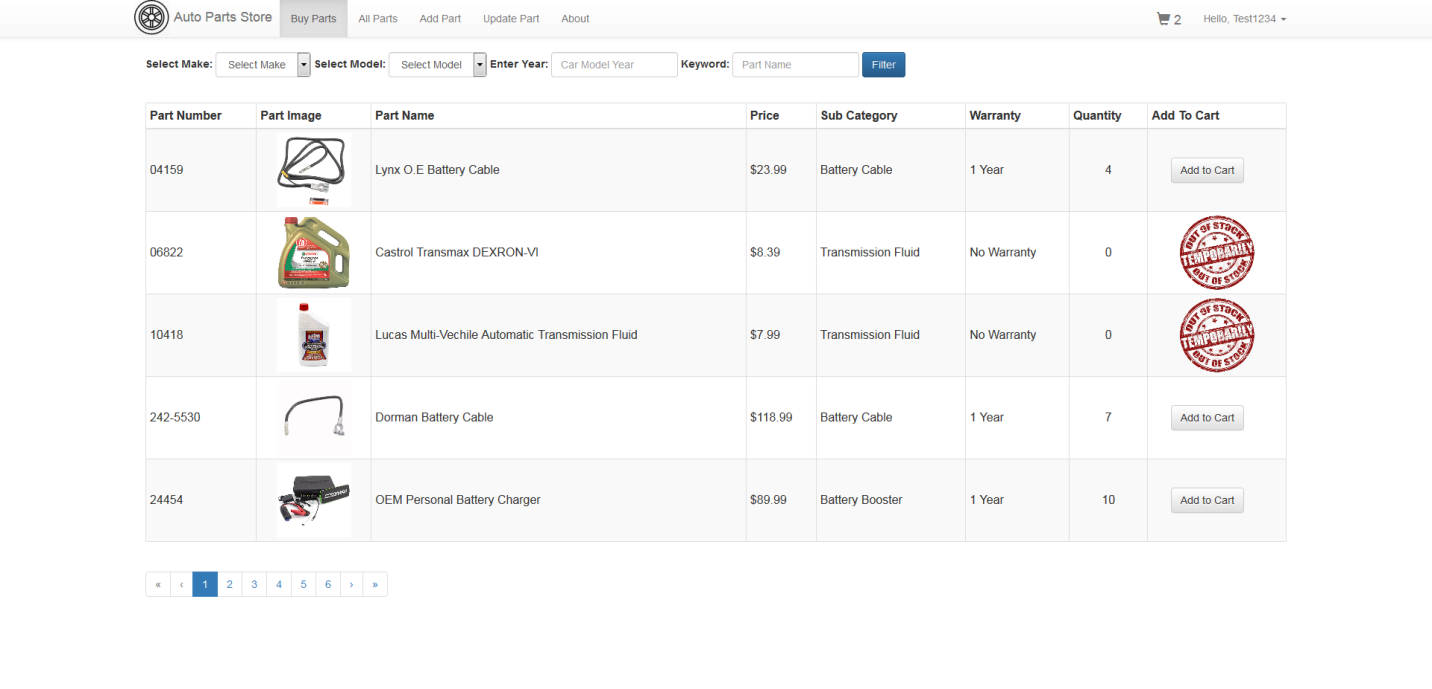


### 9.2 Registration Page



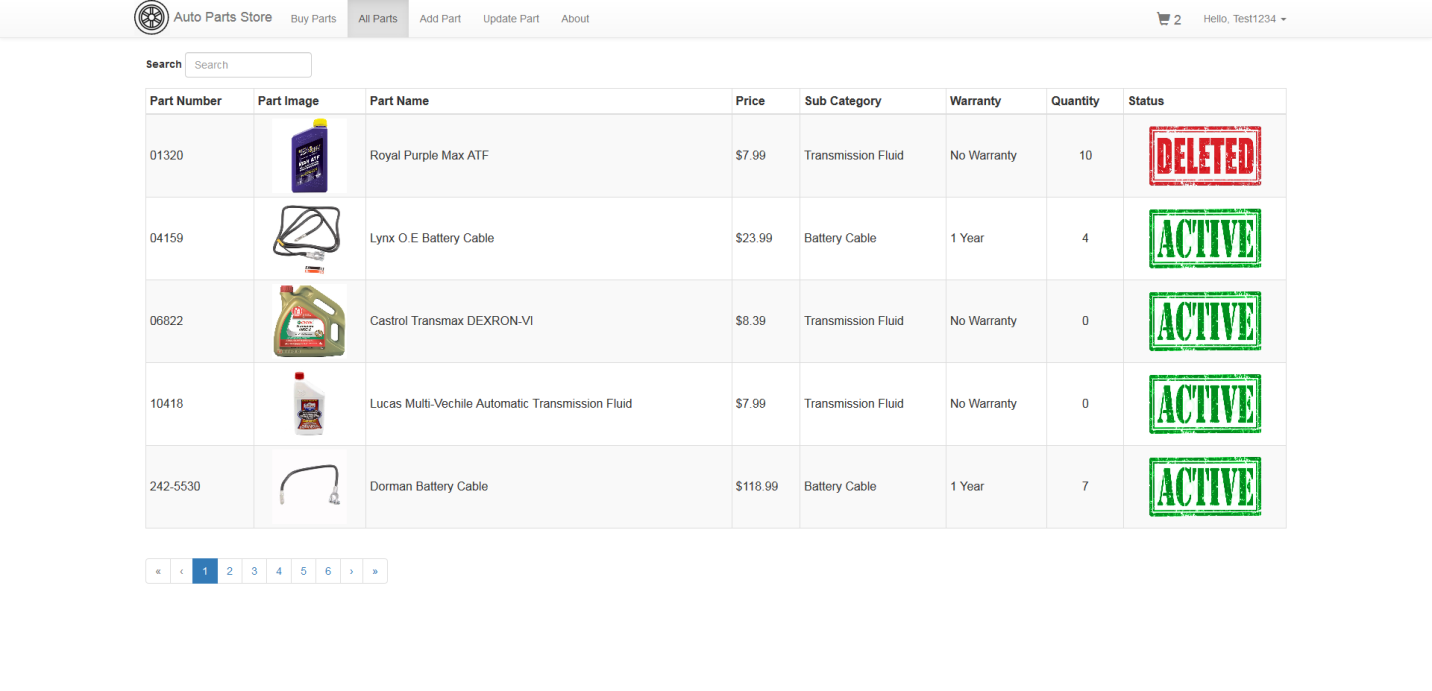
### 9.3 Buy Parts Page

Only Admins can access.



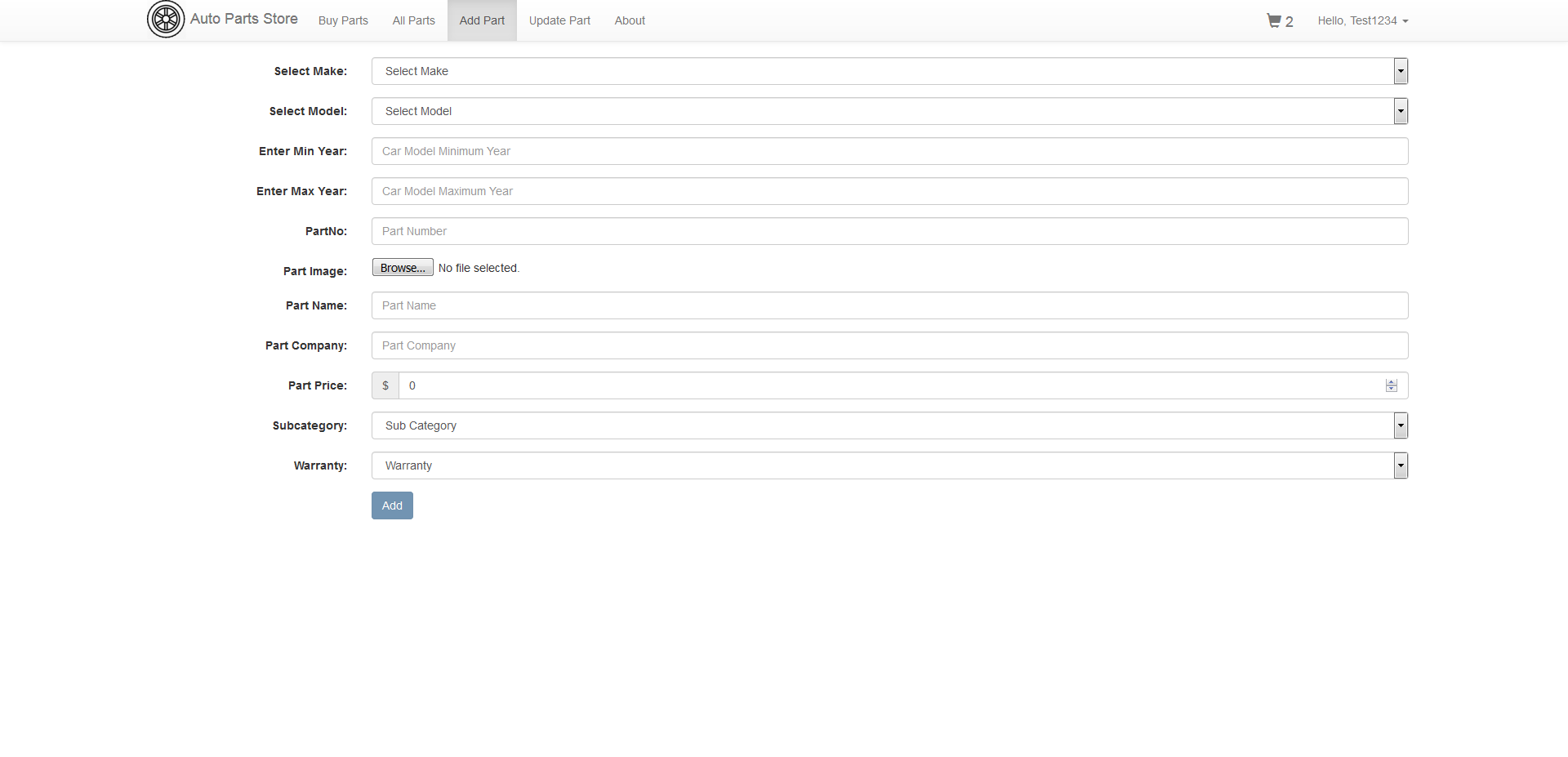
### 9.4 All Parts Page

Shows both deleted and active parts. Only Admins can access.



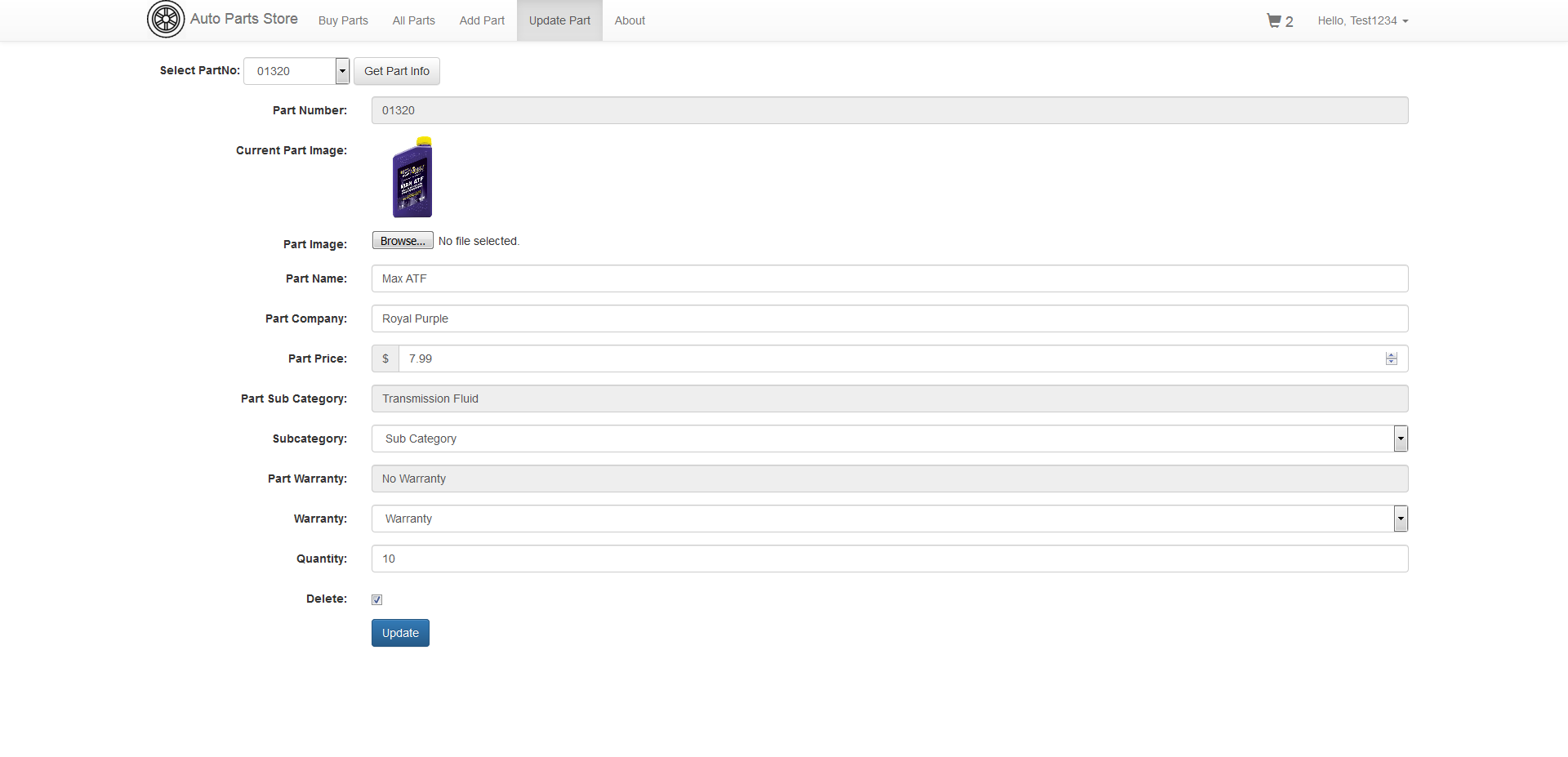
### 9.5 Add Part Page

Only Admins can access.

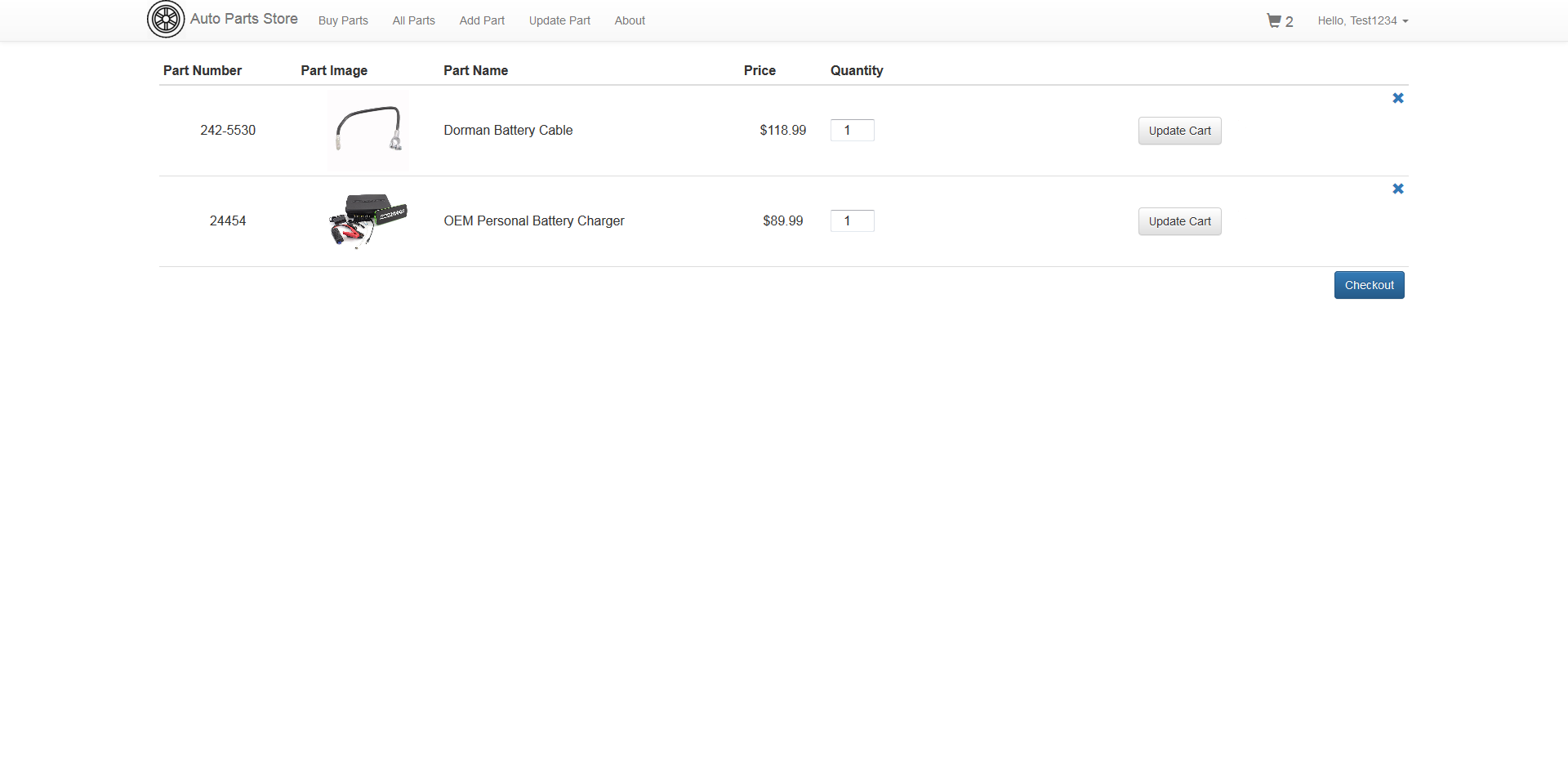


### 9.6 Update Part Page

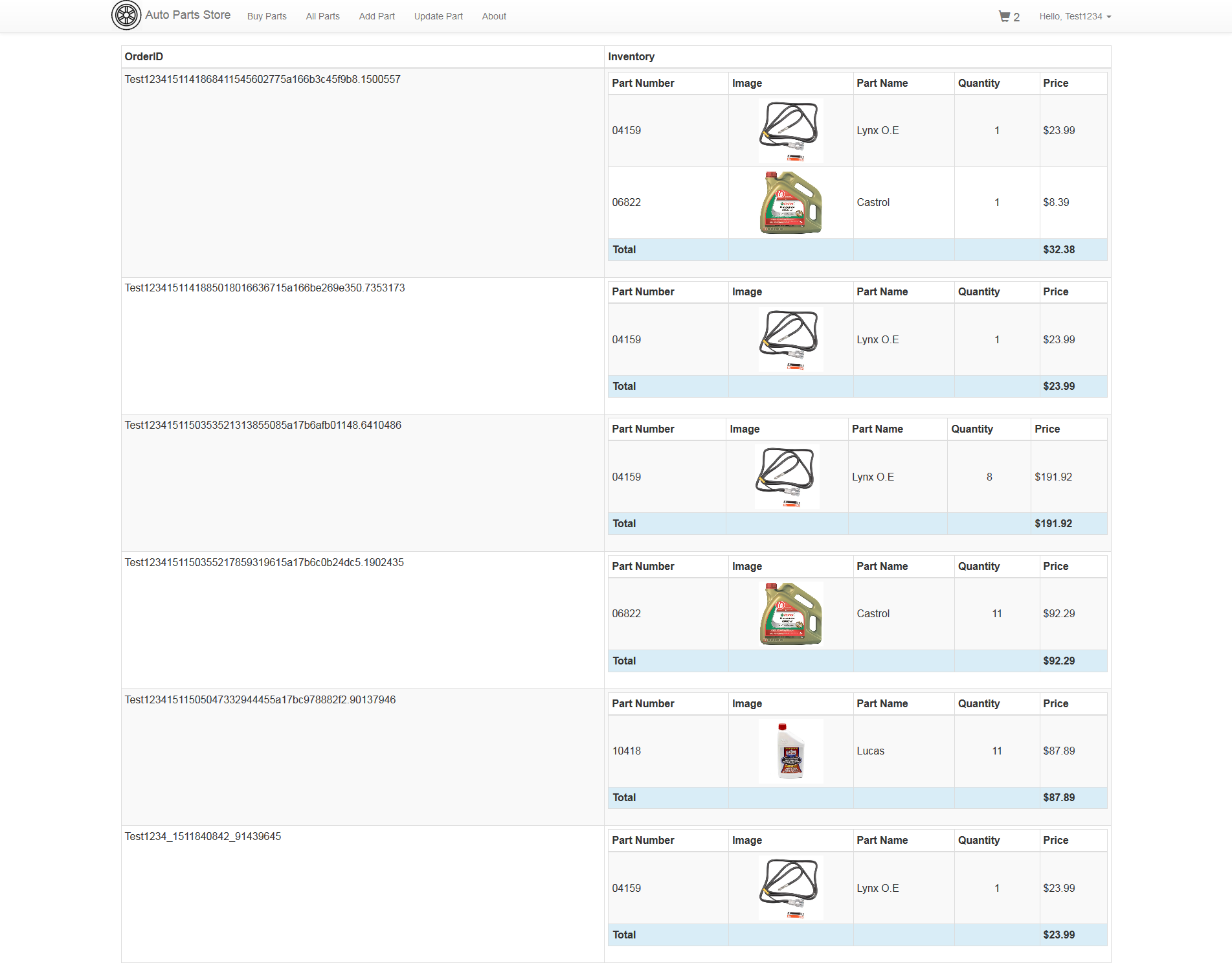
Only Admins can access.



### 9.7 User Cart Page



### 9.8 Order History Page



# 10. Work Division

### 10.1 Kartheek Kopparapu [kxk060100]

* Registration Page
* All Parts Page
* Update Part Page
* Order History Page

### 10.2 Sriraam Ramakrishnan [sxr163730]

* Login Page
* Buy Parts Page
* Add Part Page
* User Cart Page