Final Report:

Daily Deals Shopping

A three tier database application

Team 28 Members:

Mandeep Pabla,

Brandon Archbold,

Yilin Zhao

Description

We propose a web-based database application that will be a shopping website allowing users to browse and purchase the daily deals on the site. Our application will only allow customers to a limited selection of curated items every day. Features include: user login, sign-up, user authentication, sessions, client-side/server-side validation. The stakeholders for this application are any consumers who shop online, this project is important to them because our project will provide a clean interface to shop for unique curated products with no frills or distractions. The appeal of this website is that is a quick place for shoppers to see the daily deals and quickly purchase items they want or check in the next day to see what is fresh in stock. This will enable us to develop a sleek and fast product for the customers.

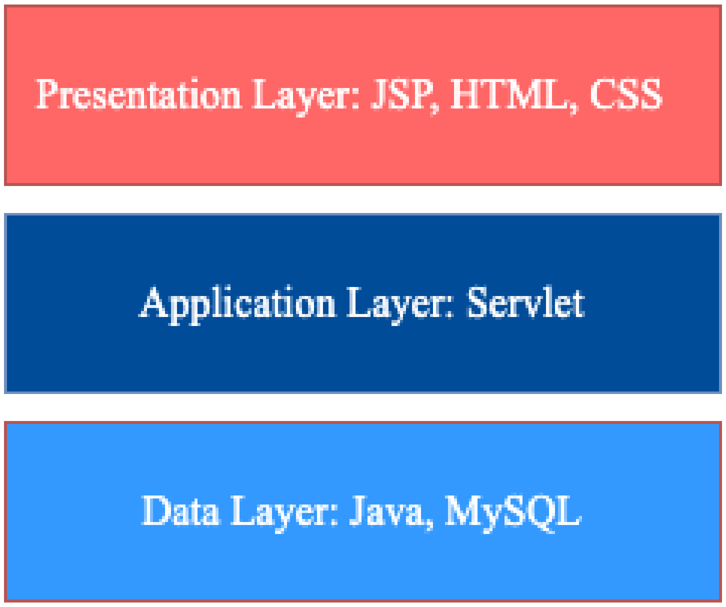
System Environment

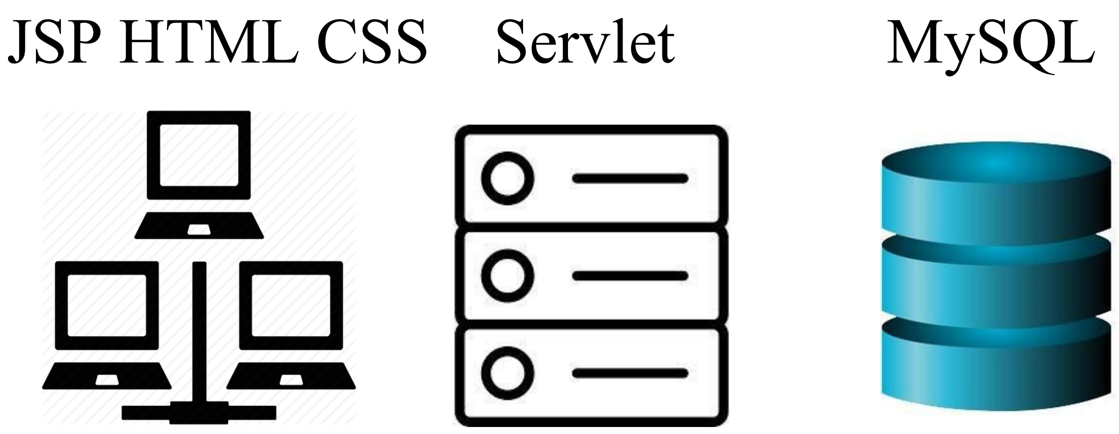
The presentation layer uses JSP, HTML, and CSS to display render data received from servlets and forwards user requests to corresponding servlets.

The application layer uses servlets to handle HTTP request made by the presentation layer and communicates with data access objects to update or query the MySQL database. Servlets in application layer can also communicate with each other.

The data layer includes MySQL database and data access objects(DAO) for each table in the database. DAOs communicates with the database using JDBC.

Apache Tomcat is a servlet container that is used to deploy the application. It is used for servlet based java applications.


Presentation Layer: JSP, HTML, CSS
Application Layer: Servlet
Data Layer: Java, MySQL



System Environment

- Application Languages: Java 8, JavaEE 7, HTML5, CSS3, JSP, XML, MySQL
- Software Used: Apache Tomcat, MySQL workbench
- RDBMS: MySQL Community Server 8.0.17

Functional Requirements

Users will be able to access our application through their web browser. The application will allow users to sign-up and will store the username & password hash in a table inside the database. If the user wants to log in, then we can query the corresponding login information and authenticate the user. Users will be able to select products they would like to purchase and continue through the checkout process. The following features will be supported on our website:

I. Account Registration
   A. This feature will allow any person to create an account with a valid email. Registration will require first name, last name, email, and password. These items will all be stored in the database and used for user authentication. The password will be saved as a hash in the database.
   B. Emails will be validated to ensure they are formatted correctly.
   C. Passwords will have a minimum requirement of 8 characters, and at least one number and one symbol

II. Log in system
   A. Customers will be able to securely log into the website by entering their email and password, the login information will be securely sent using SSL when the final project is launched.
   B. The hashed password will be queried from database by email. Then the hash will be compared with the hashed password from user's input. The plain text is never used after account registration.

III. Browse products
   A. Customers will be able to browse items from our main page. They will be able to view detailed descriptions when they click on the items. They will be taken to

another page for the product details.

B. Everyday there will be 10-20 items to browse. This enables shoppers to quickly visit our site and see what deals there are today and then quickly purchase the desired items if wanted.

IV. Product detail page outlining the product description

A. Reviews

1. Each product will feature a section allowing registered customers to leave a review with a thumbs up or thumbs down to go along with their review. The site will then calculate which percentage of users recommend the product.
2. The percentage of people who liked the product will be displayed under the product name

B. other information regarding the product

1. Product dimensions
2. Product specification
3. additional images

C. Price

1. Customers will be able to view product prices.

V. Shopping cart

A. In the shopping cart, there will be items' name and quantity as well as a total price.

B. The shopping cart will also show any available sales and promotions a customer has

VI. Checkout and confirmation page

A. At the checkout page, customers can make adjustments to quantities of the items, fill out payment & delivery information, and submit order. Customers will receive an order#.

B. Customers will be able to choose from saved delivery and payment information.

C. Customers will have the option to save their delivery information for future use.

D. Customers will have the option to save their payment information for future use.

E. If Customers choose to save their payment/shipment information, the information will be saved under customer's profile, otherwise the information will only be used once for the order.

VII. Order history page

A. From the order history page, customers can view their previous orders, including items' name, quantity, and total price.

VIII. Account management page

A. A simple account management page will provide quick access to common features such as:

B. changing password
C. changing address
D. changing email address
E. and other account-related settings.

IX. Wishlist feature
A. Customers will be able to wishlist items they would like to buy in the future. This will enable them to purchase the item at a later time. The list of items on a customer's wish list will also be stored in the database. Wishlisted items can only be purchased when the item is in stock.

X. Orders to be filled (Business view)
A. The software will support a view of all the orders that have been placed and need to be filled.
B. Each order page will display what the customer purchased, the quantity of items, as well as the customer's shipping address

Description of a Typical User

A typical user of our website is someone who enjoys online shopping but does not have the time to browse through the endless amount of products available on other websites to have to hunt for a deal. Our website will provide customers seeking daily deals a quick place to stop by and buy some sweet items. Our typical user base is any and all online shoppers interested in a simpler way to shop.

Standard User Flow

A standard flow a user would go through when visiting our site would be the following:

1. Navigate to our website using their browser
2. View and browse through today's available products
3. click on a product to visit the details page
4. view user reviews, item specifications
5. scroll through images of the product
6. click on the add to cart button
7. click on the shopping cart to proceed to checkout
8. use email and password to login or sign up as a new user
9. Once on the checkout page, the user can select the quantity of items
10. Enter payment information
11. Enter shipping and billing info
12. View a confirmation page
13. Confirm order

Non-functional issues

GUI

We will use HTML, CSS, and Javascript and React to build the front end of our website to be displayed in the browser. JSP will be used for webpage scripting and Apache Tomcat will deploy Java Servlets and JSP.

Products will be displayed in a 4 x 5 grid where 10-20 products will be displayed at any given time. The top of the website will feature our logo, and navigation links on the top right hand side. There will be an account page to manage your account and a logout button. If a user does not have an account, they will be directed to do so when attempting to purchase a product or when clicking on the account link.

Each product in the grid will display an image of the product, the name of the product, the price, and show the percentage of users that recommend the product.

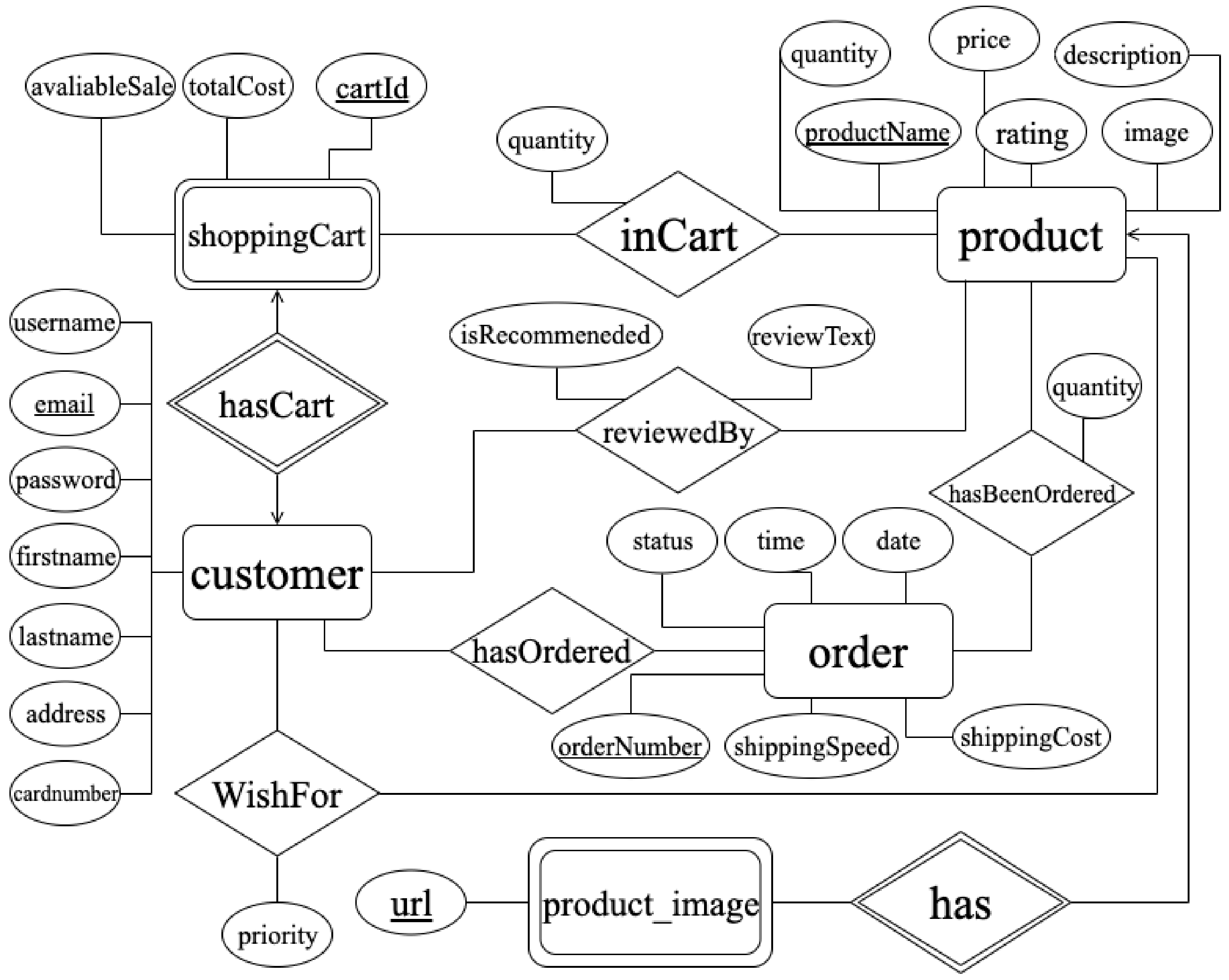
On the product details page you will see the name of the product at the top, below it the percentage of people that liked the product, with a thumbs up for ratings of over 70% and thumbs down for ratings below 40%. Ratings between 40% and 70% will be represented as an orange circle. On the left there will be a product preview pane where the user may browse through several images of the product (if available). On the right of the image will be the price, and below it the product specifications and details. On the right side of this section there will be a button that lets users add the product to the shopping cart.

Below this section will be a product details page with further details about the product (if available). Below this section will be a reviews section with reviews of the product.

Security

For user authentication we will use SSL (when the website goes live) and only store the hash of the user's password so that if the database gets hacked, the user's passwords are safe from attackers. This will be the only access control as we would like anyone to be able to buy products from our website as long as they create an account. Users will be authenticated with their username and password combination. Users will only be able to access the account information of their own account and will not be able to view other members orders or personal information. All product information will be public to all users.

Project Data Model and DB Design



Customer: This entity set keeps track of all the customers and the information that is relevant to each. The email is the customer's primary key as it is required to create an account and it must be unique so it is an obvious choice. Customer has many relationships as they customer can review products, place items in their cart, order products, and place items in their wishlist. The relation schema for Customer is the following: Customer(username, email, password, firstName, lastName, address, cardNumber).

Order: This entity set keeps track of all orders with a primary key of a unique orderNumber. This set is used by customers to see what they have ordered, and by the business to see what has been ordered and what needs to be shipped. The relation schema for Order is the following: Order(orderNumber, Status, shipping_speed, Time, Date, shipping_cost).

Product: This entity set contains all the available products with the productName as the primaryKey as no two products can share the same name.Products can be placed in shopping carts, ordered, reviewed, or placed in a customer's wish list. The relation schema for Product is the following: Product(productName, price, rating, description, image).

hasCart: This relationship occurs when a customer places a product in their shopping cart for checkout. The cart will total the cost of products in the cart.The schema for this relation is hasCart(email, cartID)

in cart: This relationship links the products and quantity of products in a given shopping cart. The schema for this relation is inCart(cartID, productName, quantity)

reviewed by: This relationship is used for customer reviews of products. If an entry exists that means that a particular customer has left a review for that particular product. The schema for this relation is reviewedBy(email, productName, isRecommended, reviewText)

hasOrdered: This relationship indicated which customer has made a particular order. This links the customer to the order. The schema for this relation is hasOrdered(email, orderNumber)

hasBeenOrdered: This relationship indicates which products are in a particular order and the quantity of those products. The schema for this relation is (orderNumber, productName, quantity)

wishes for: This relationship indicates which products a customer has on their wishlist. The customer can view a list of their items organized by the priority in which they would like. The schema for this relation is wishesFor(email, productName)

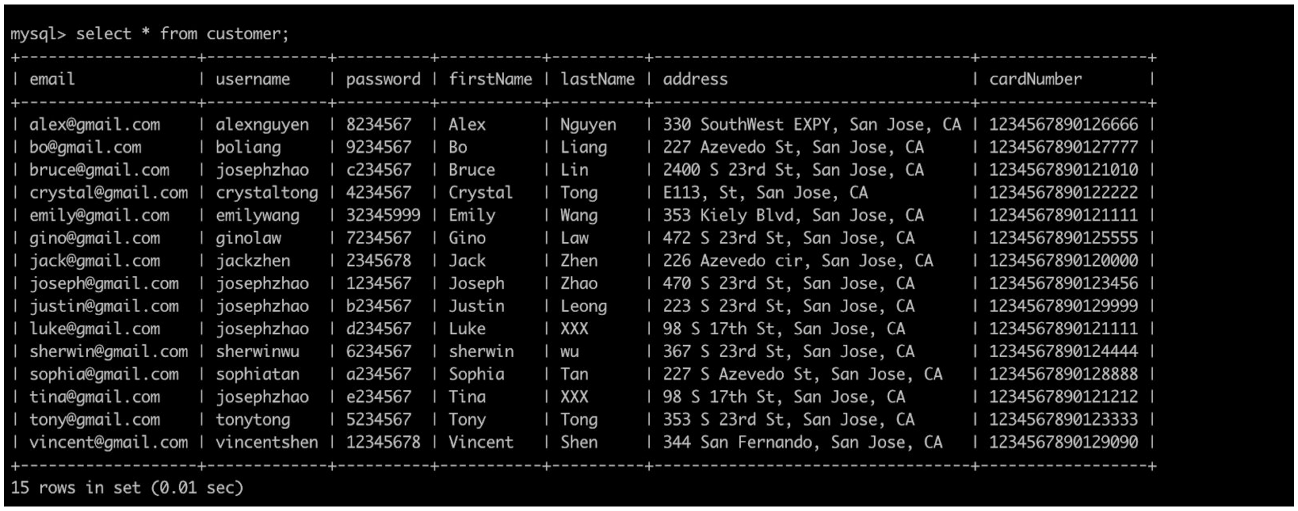
ShoppingCart: The entityset contains all available sales and the total cost of products. The relation schema for ShoppingCart is the following: ShoppingCart(cartID, availableSales, total_cost)

product_image: This entity set contains the images that belong to a product, since a product can have multiple images. The schema for this is product_image(productName, url)

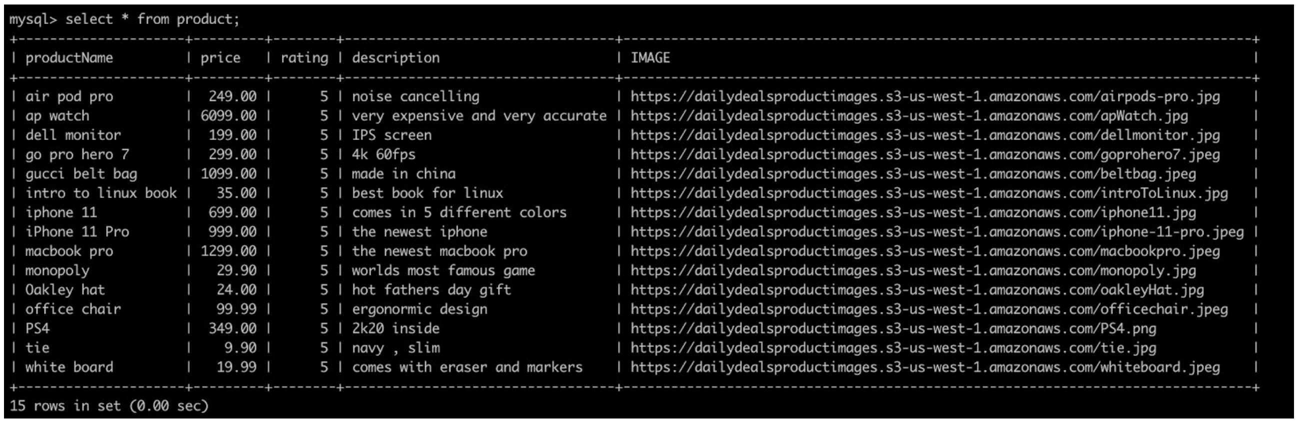
Tables

```
CREATE TABLE `dailydeals`.`customer` (
  `email` VARCHAR(150) NOT NULL,
  `username` VARCHAR(45) NOT NULL,
  `password` VARCHAR(45) NOT NULL,
  `firstName` VARCHAR(45) NOT NULL,
  `lastName` VARCHAR(90) NOT NULL,
  `address` VARCHAR(90) NOT NULL,
  `cardNumber` VARCHAR(120) NOT NULL,
  PRIMARY KEY (`email`));


CREATE TABLE `dailydeals`.`product` (
  `productName` VARCHAR(150) NOT NULL,
  `price` DECIMAL(6,2) NOT NULL,
  `rating` TINYINT(1) NOT NULL,
  `description` VARCHAR(250) NOT NULL,
  `image` LONGBLOB NOT NULL,
  PRIMARY KEY (`productName`));

CREATE TABLE `dailydeals`.`order` (
  `orderNumber` INT NOT NULL,
  `status` VARCHAR(45) NOT NULL,
  `shipping_speed` VARCHAR(45) NOT NULL,
  `date` DATE NOT NULL,
  `time` TIME NOT NULL,
  `shipping_cost` DECIMAL(6,2) NOT NULL,
  PRIMARY KEY (`orderNumber`));


CREATE TABLE `dailydeals`.`wishFor` (
  `email` VARCHAR(150) NOT NULL,
  `productName` VARCHAR(150) NOT NULL,
  `priority` VARCHAR(30) NOT NULL,
  FOREIGN KEY (`email`) REFERENCES `customer`(`email`),
  FOREIGN KEY (`productName`) REFERENCES `product`(`productName`)
);

CREATE TABLE `dailydeals`.`review`(
  `email` VARCHAR(150) NOT NULL,
  `productName` VARCHAR(150) NOT NULL,
  `isRecommended` BOOLEAN NOT NULL,
  `reviewText` VARCHAR(1000),
  FOREIGN KEY (`email`) REFERENCES `customer`(`email`),
  FOREIGN KEY (`productName`) REFERENCES `product`(`productName`)
);

CREATE TABLE `dailydeals`.`hasBeenOrdered`(
  `orderNumber` INT NOT NULL,
  `productName` VARCHAR(150) NOT NULL,
  `quantity` SMALLINT NOT NULL,
  FOREIGN KEY (`orderNumber`) REFERENCES `order`(`orderNumber`),
  FOREIGN KEY (`productName`) REFERENCES `product`(`productName`)
);

create table `inCart`(`email` varchar(150) NOT NULL,
`productName` VARCHAR(150) NOT NULL,
`quantity` INT(2) NOT NULL,
FOREIGN KEY(`email`) REFERENCES `customer`(`email`),
FOREIGN KEY(`productName`) REFERENCES `product`(`productName`));

create table `reviewedBy`(`email` varchar(150) NOT NULL,
  `productName` VARCHAR(150) NOT NULL,
  `isRecommended` TINYINT(1) NOT NULL,
  `reviewText` VARCHAR(1000),
  FOREIGN KEY(`email`) REFERENCES `customer`(`email`),
  FOREIGN KEY(`productName`) REFERENCES `product`(`productName`));

create table `shoppingCart`(`email` varchar(150) NOT NULL,
`avalibleSales` INT,
`totalCost` DECIMAL(6, 2) NOT NULL,
FOREIGN KEY(`email`) REFERENCES `customer`(`email`));

CREATE TABLE `dailydeals`.`websiteReview`(
  `email` VARCHAR(150) NOT NULL,
  `reviewText` VARCHAR(1000),
  FOREIGN KEY (`email`) REFERENCES `customer`(`email`)
);
```

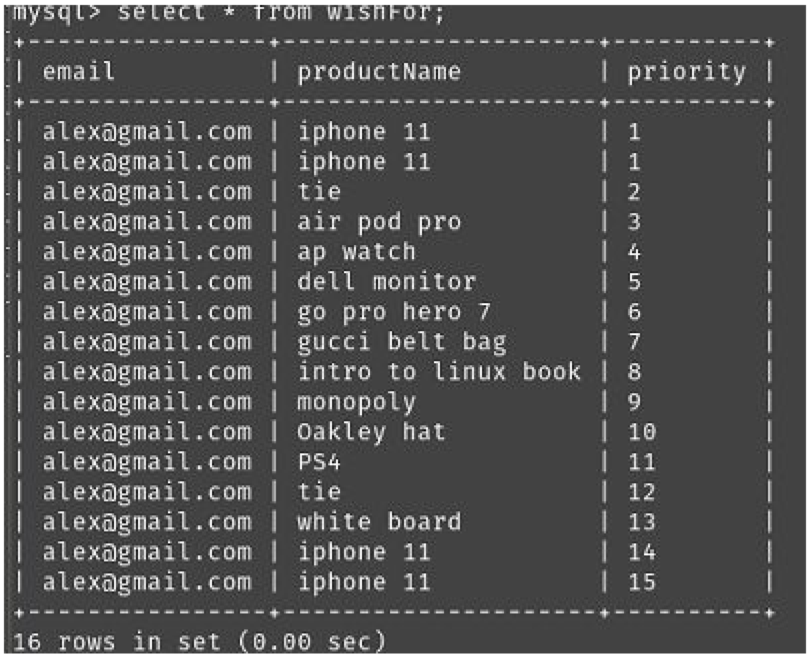
Tables with data



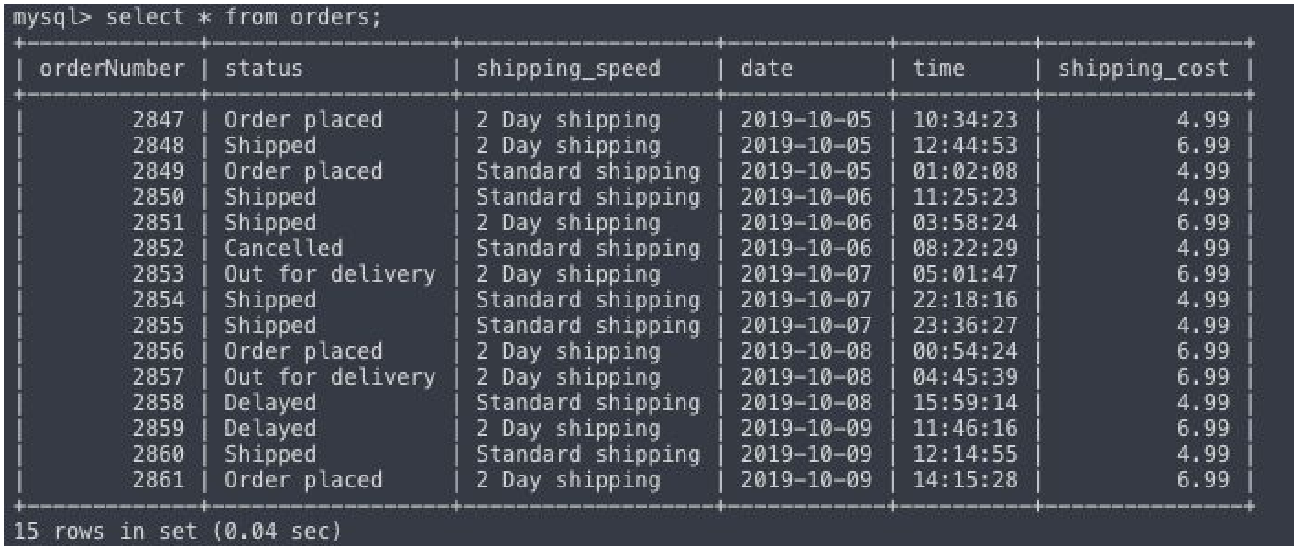
```
mysql> select * from customer;
+-------------------+-------------+----------+-----------+----------+----------------------------------+------------------+
| email             | username    | password | firstName | lastName | address                          | cardNumber       |
+-------------------+-------------+----------+-----------+----------+----------------------------------+------------------+
| alex@gmail.com    | alexnguyen  | 8234567  | Alex      | Nguyen   | 330 SouthWest EXPY, San Jose, CA | 1234567890126666 |
| bo@gmail.com      | boliang     | 9234567  | Bo        | Liang    | 227 Azevedo St, San Jose, CA     | 1234567890127777 |
| bruce@gmail.com   | josephzhao  | c234567  | Bruce     | Lin      | 2400 S 23rd St, San Jose, CA     | 1234567890121010 |
| crystal@gmail.com | crystaltong | 4234567  | Crystal   | Tong     | E113, St, San Jose, CA           | 1234567890122222 |
| emily@gmail.com   | emilywang   | 32345999 | Emily     | Wang     | 353 Kiely Blvd, San Jose, CA     | 1234567890121111 |
| gino@gmail.com    | ginolaw     | 7234567  | Gino      | Law      | 472 S 23rd St, San Jose, CA      | 1234567890125555 |
| jack@gmail.com    | jackzhen    | 2345678  | Jack      | Zhen     | 226 Azevedo cir, San Jose, CA    | 1234567890120000 |
| joseph@gmail.com  | josephzhao  | 1234567  | Joseph    | Zhao     | 470 S 23rd St, San Jose, CA      | 1234567890123456 |
| justin@gmail.com  | josephzhao  | b234567  | Justin    | Leong    | 223 S 23rd St, San Jose, CA      | 1234567890129999 |
| luke@gmail.com    | josephzhao  | d234567  | Luke      | XXX      | 98 S 17th St, San Jose, CA       | 1234567890121111 |
| sherwin@gmail.com | sherwinwu   | 6234567  | sherwin   | wu       | 367 S 23rd St, San Jose, CA      | 1234567890124444 |
| sophia@gmail.com  | sophiatan   | a234567  | Sophia    | Tan      | 227 S Azevedo St, San Jose, CA   | 1234567890128888 |
| tina@gmail.com    | josephzhao  | e234567  | Tina      | XXX      | 98 S 17th St, San Jose, CA       | 1234567890121212 |
| tony@gmail.com    | tonytong    | 5234567  | Tony      | Tong     | 353 S 23rd St, San Jose, CA      | 1234567890123333 |
| vincent@gmail.com | vincentshen | 12345678 | Vincent   | Shen     | 344 San Fernando, San Jose, CA   | 1234567890129090 |
+-------------------+-------------+----------+-----------+----------+----------------------------------+------------------+
15 rows in set (0.01 sec)
```



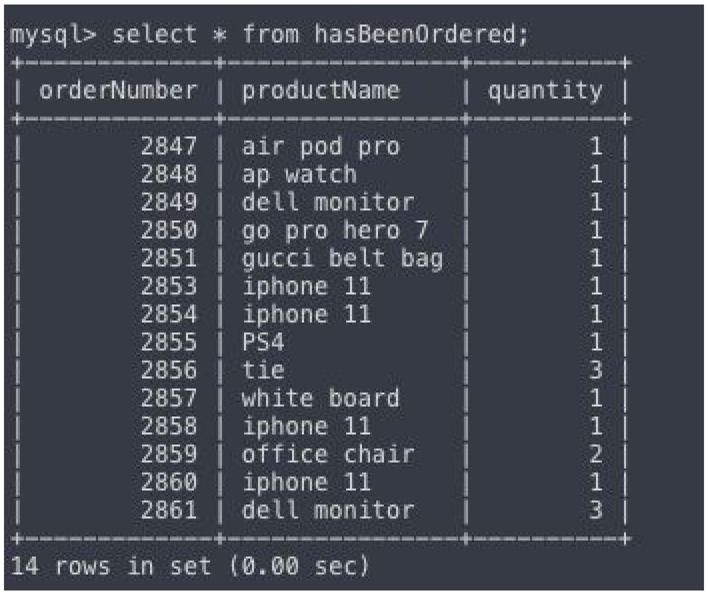
```
mysql> select * from product;
+---------------------+---------+--------+----------------------------------+-------------------------------------------------------------------------------+
| productName         | price   | rating | description                      | IMAGE                                                                         |
+---------------------+---------+--------+----------------------------------+-------------------------------------------------------------------------------+
| air pod pro         |  249.00 |      5 | noise cancelling                 | https://dailydealsproductimages.s3-us-west-1.amazonaws.com/airpods-pro.jpg    |
| ap watch            | 6099.00 |      5 | very expensive and very accurate | https://dailydealsproductimages.s3-us-west-1.amazonaws.com/apWatch.jpg        |
| dell monitor        |  199.00 |      5 | IPS screen                       | https://dailydealsproductimages.s3-us-west-1.amazonaws.com/dellmonitor.jpg    |
| go pro hero 7       |  299.00 |      5 | 4k 60fps                         | https://dailydealsproductimages.s3-us-west-1.amazonaws.com/goprohero7.jpeg    |
| gucci belt bag      | 1099.00 |      5 | made in china                    | https://dailydealsproductimages.s3-us-west-1.amazonaws.com/beltbag.jpeg       |
| intro to linux book |   35.00 |      5 | best book for linux              | https://dailydealsproductimages.s3-us-west-1.amazonaws.com/introToLinux.jpg   |
| iphone 11           |  699.00 |      5 | comes in 5 different colors      | https://dailydealsproductimages.s3-us-west-1.amazonaws.com/iphone11.jpg       |
| iPhone 11 Pro       |  999.00 |      5 | the newest iphone                | https://dailydealsproductimages.s3-us-west-1.amazonaws.com/iphone-11-pro.jpeg |
| macbook pro         | 1299.00 |      5 | the newest macbook pro           | https://dailydealsproductimages.s3-us-west-1.amazonaws.com/macbookpro.jpeg    |
| monopoly            |   29.90 |      5 | worlds most famous game          | https://dailydealsproductimages.s3-us-west-1.amazonaws.com/monopoly.jpg       |
| Oakley hat          |   24.00 |      5 | hot fathers day gift             | https://dailydealsproductimages.s3-us-west-1.amazonaws.com/oakleyHat.jpg      |
| office chair        |   99.99 |      5 | ergonormic design                | https://dailydealsproductimages.s3-us-west-1.amazonaws.com/officechair.jpeg   |
| PS4                 |  349.00 |      5 | 2k20 inside                      | https://dailydealsproductimages.s3-us-west-1.amazonaws.com/PS4.png            |
| tie                 |    9.90 |      5 | navy , slim                      | https://dailydealsproductimages.s3-us-west-1.amazonaws.com/tie.jpg            |
| white board         |   19.99 |      5 | comes with eraser and markers    | https://dailydealsproductimages.s3-us-west-1.amazonaws.com/whiteboard.jpeg    |
+---------------------+---------+--------+----------------------------------+-------------------------------------------------------------------------------+
15 rows in set (0.00 sec)
```



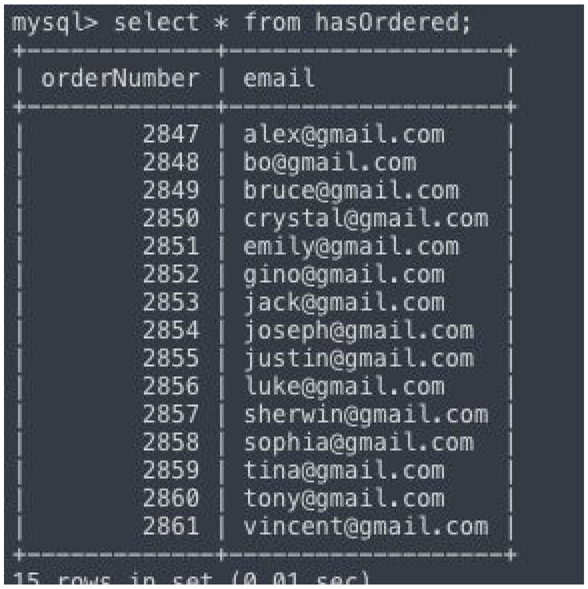
```
mysql> select * from wishFor;
+----------------+---------------------+----------+
| email          | productName         | priority |
+----------------+---------------------+----------+
| alex@gmail.com | iphone 11           | 1        |
| alex@gmail.com | iphone 11           | 1        |
| alex@gmail.com | tie                 | 2        |
| alex@gmail.com | air pod pro         | 3        |
| alex@gmail.com | ap watch            | 4        |
| alex@gmail.com | dell monitor        | 5        |
| alex@gmail.com | go pro hero 7       | 6        |
| alex@gmail.com | gucci belt bag      | 7        |
| alex@gmail.com | intro to linux book | 8        |
| alex@gmail.com | monopoly            | 9        |
| alex@gmail.com | Oakley hat          | 10       |
| alex@gmail.com | PS4                 | 11       |
| alex@gmail.com | tie                 | 12       |
| alex@gmail.com | white board         | 13       |
| alex@gmail.com | iphone 11           | 14       |
| alex@gmail.com | iphone 11           | 15       |
+----------------+---------------------+----------+
16 rows in set (0.00 sec)
```



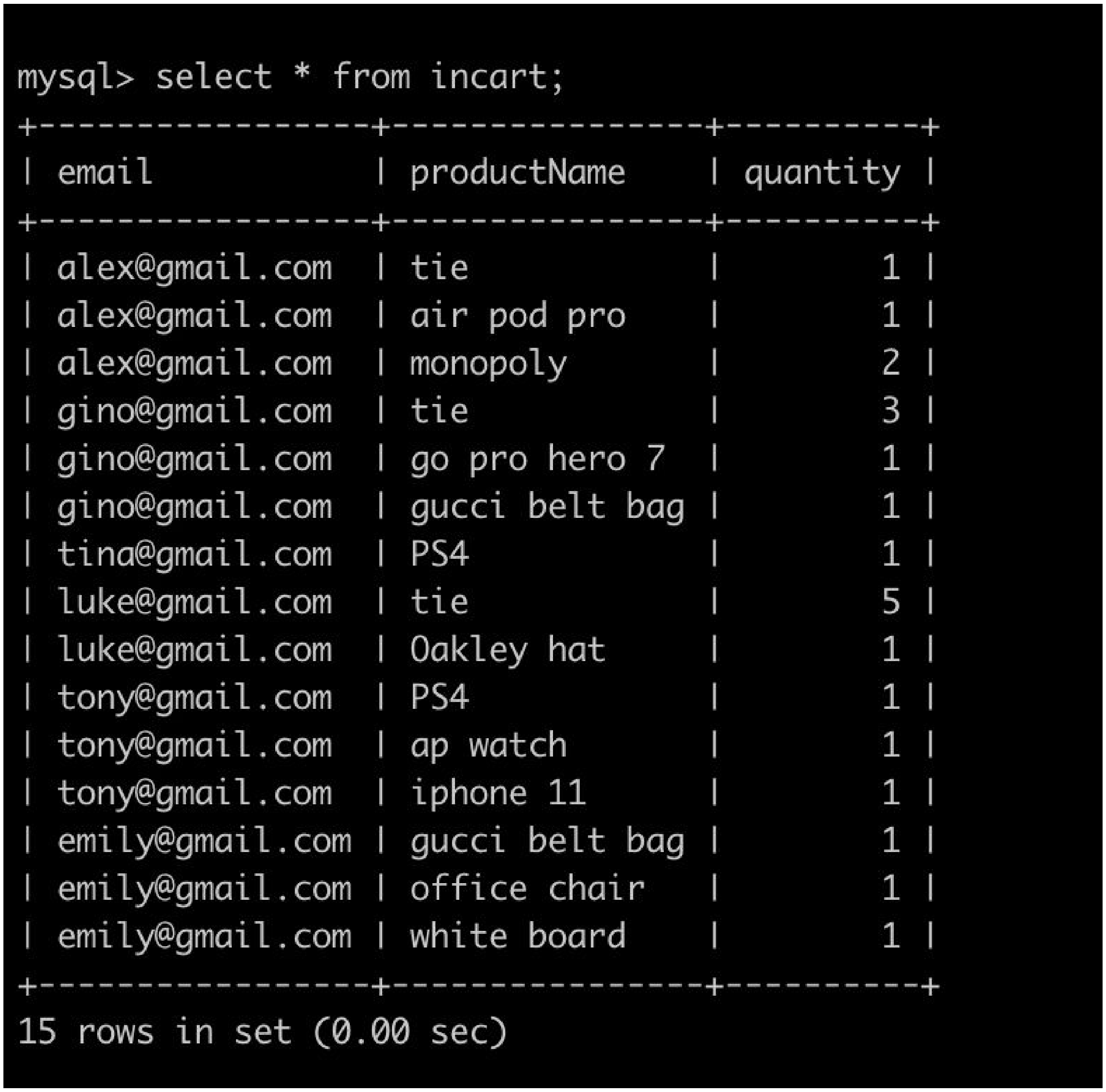
```
mysql> select * from orders;
+-------------+------------------+-------------------+------------+----------+---------------+
| orderNumber | status           | shipping_speed    | date       | time     | shipping_cost |
+-------------+------------------+-------------------+------------+----------+---------------+
|        2847 | Order placed     | 2 Day shipping    | 2019-10-05 | 10:34:23 |          4.99 |
|        2848 | Shipped          | 2 Day shipping    | 2019-10-05 | 12:44:53 |          6.99 |
|        2849 | Order placed     | Standard shipping | 2019-10-05 | 01:02:08 |          4.99 |
|        2850 | Shipped          | Standard shipping | 2019-10-06 | 11:25:23 |          4.99 |
|        2851 | Shipped          | 2 Day shipping    | 2019-10-06 | 03:58:24 |          6.99 |
|        2852 | Cancelled        | Standard shipping | 2019-10-06 | 08:22:29 |          4.99 |
|        2853 | Out for delivery | 2 Day shipping    | 2019-10-07 | 05:01:47 |          6.99 |
|        2854 | Shipped          | Standard shipping | 2019-10-07 | 22:18:16 |          4.99 |
|        2855 | Shipped          | Standard shipping | 2019-10-07 | 23:36:27 |          4.99 |
|        2856 | Order placed     | 2 Day shipping    | 2019-10-08 | 00:54:24 |          6.99 |
|        2857 | Out for delivery | 2 Day shipping    | 2019-10-08 | 04:45:39 |          6.99 |
|        2858 | Delayed          | Standard shipping | 2019-10-08 | 15:59:14 |          4.99 |
|        2859 | Delayed          | 2 Day shipping    | 2019-10-09 | 11:46:16 |          6.99 |
|        2860 | Shipped          | Standard shipping | 2019-10-09 | 12:14:55 |          4.99 |
|        2861 | Order placed     | 2 Day shipping    | 2019-10-09 | 14:15:28 |          6.99 |
+-------------+------------------+-------------------+------------+----------+---------------+
15 rows in set (0.04 sec)
```



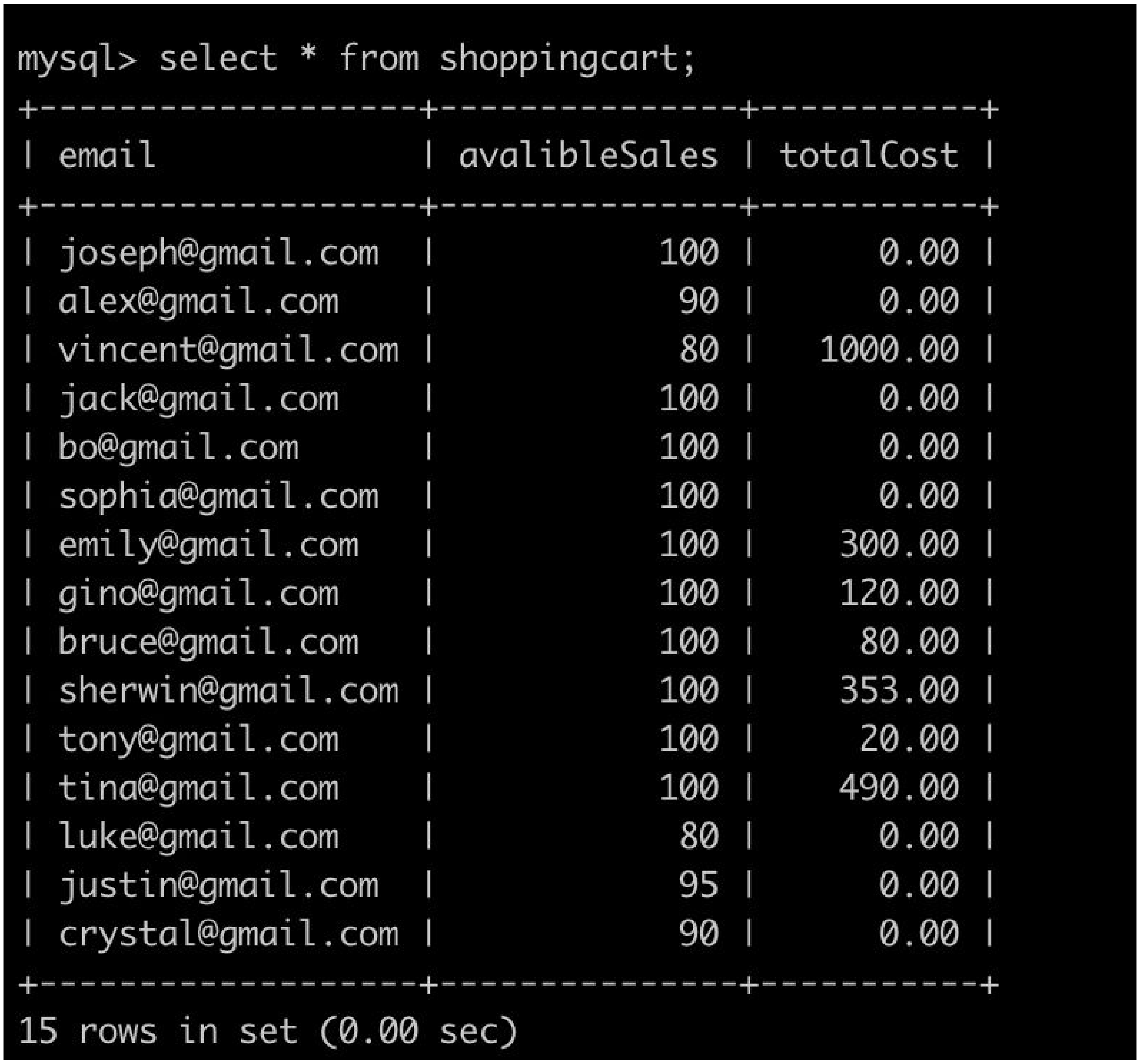
```
mysql> select * from hasBeenOrdered;
+-------------+----------------+----------+
| orderNumber | productName    | quantity |
+-------------+----------------+----------+
|        2847 | air pod pro    |        1 |
|        2848 | ap watch       |        1 |
|        2849 | dell monitor   |        1 |
|        2850 | go pro hero 7  |        1 |
|        2851 | gucci belt bag |        1 |
|        2853 | iphone 11      |        1 |
|        2854 | iphone 11      |        1 |
|        2855 | PS4            |        1 |
|        2856 | tie            |        3 |
|        2857 | white board    |        1 |
|        2858 | iphone 11      |        1 |
|        2859 | office chair   |        2 |
|        2860 | iphone 11      |        1 |
|        2861 | dell monitor   |        3 |
+-------------+----------------+----------+
14 rows in set (0.00 sec)
```



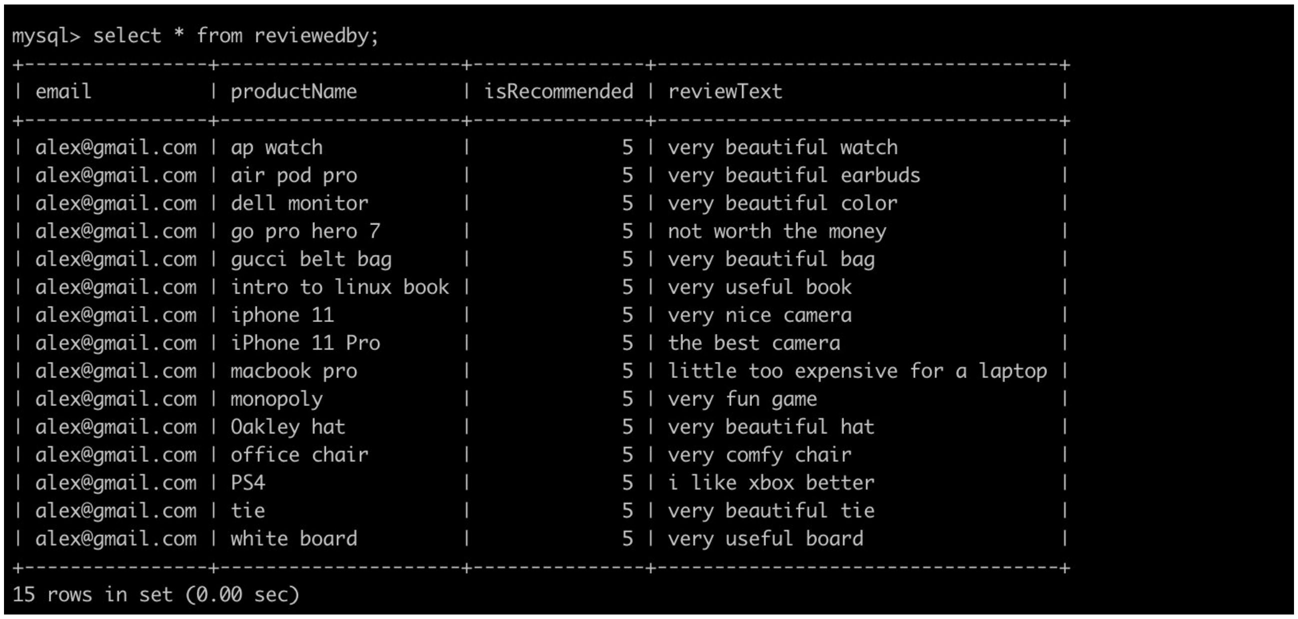
```
mysql> select * from hasOrdered;
+-------------+--------------------+
| orderNumber | email              |
+-------------+--------------------+
|        2847 | alex@gmail.com     |
|        2848 | bo@gmail.com       |
|        2849 | bruce@gmail.com    |
|        2850 | crystal@gmail.com  |
|        2851 | emily@gmail.com    |
|        2852 | gino@gmail.com     |
|        2853 | jack@gmail.com     |
|        2854 | joseph@gmail.com   |
|        2855 | justin@gmail.com   |
|        2856 | luke@gmail.com     |
|        2857 | sherwin@gmail.com  |
|        2858 | sophia@gmail.com   |
|        2859 | tina@gmail.com     |
|        2860 | tony@gmail.com     |
|        2861 | vincent@gmail.com  |
+-------------+--------------------+
15 rows in set (0.01 sec)
```



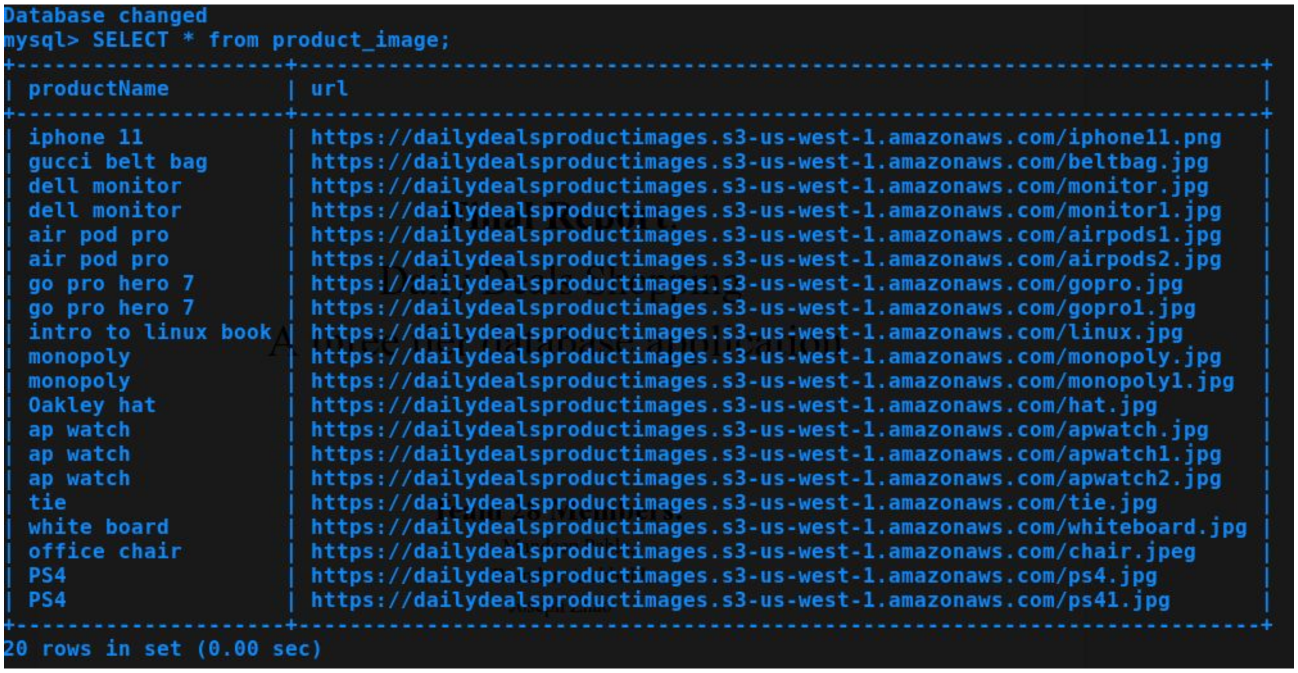
```
mysql> select * from incart;
+-----------------+----------------+----------+
| email           | productName    | quantity |
+-----------------+----------------+----------+
| alex@gmail.com  | tie            |        1 |
| alex@gmail.com  | air pod pro    |        1 |
| alex@gmail.com  | monopoly       |        2 |
| gino@gmail.com  | tie            |        3 |
| gino@gmail.com  | go pro hero 7  |        1 |
| gino@gmail.com  | gucci belt bag |        1 |
| tina@gmail.com  | PS4            |        1 |
| luke@gmail.com  | tie            |        5 |
| luke@gmail.com  | Oakley hat     |        1 |
| tony@gmail.com  | PS4            |        1 |
| tony@gmail.com  | ap watch       |        1 |
| tony@gmail.com  | iphone 11      |        1 |
| emily@gmail.com | gucci belt bag |        1 |
| emily@gmail.com | office chair   |        1 |
| emily@gmail.com | white board    |        1 |
+-----------------+----------------+----------+
15 rows in set (0.00 sec)
```



```
mysql> select * from shoppingcart;
+-------------------+-------------+-----------+
| email             | avalibleSales | totalCost |
+-------------------+-------------+-----------+
| joseph@gmail.com  |         100 |      0.00 |
| alex@gmail.com    |          90 |      0.00 |
| vincent@gmail.com |          80 |   1000.00 |
| jack@gmail.com    |         100 |      0.00 |
| bo@gmail.com      |         100 |      0.00 |
| sophia@gmail.com  |         100 |      0.00 |
| emily@gmail.com   |         100 |    300.00 |
| gino@gmail.com    |         100 |    120.00 |
| bruce@gmail.com   |         100 |     80.00 |
| sherwin@gmail.com |         100 |    353.00 |
| tony@gmail.com    |         100 |     20.00 |
| tina@gmail.com    |         100 |    490.00 |
| luke@gmail.com    |          80 |      0.00 |
| justin@gmail.com  |          95 |      0.00 |
| crystal@gmail.com |          90 |      0.00 |
+-------------------+-------------+-----------+
15 rows in set (0.00 sec)
```



```
mysql> select * from reviewedby;
+----------------+---------------------+---------------+----------------------------------+
| email          | productName         | isRecommended | reviewText                       |
+----------------+---------------------+---------------+----------------------------------+
| alex@gmail.com | ap watch            |             5 | very beautiful watch             |
| alex@gmail.com | air pod pro         |             5 | very beautiful earbuds           |
| alex@gmail.com | dell monitor        |             5 | very beautiful color             |
| alex@gmail.com | go pro hero 7       |             5 | not worth the money              |
| alex@gmail.com | gucci belt bag      |             5 | very beautiful bag               |
| alex@gmail.com | intro to linux book |             5 | very useful book                 |
| alex@gmail.com | iphone 11           |             5 | very nice camera                 |
| alex@gmail.com | iPhone 11 Pro       |             5 | the best camera                  |
| alex@gmail.com | macbook pro         |             5 | little too expensive for a laptop |
| alex@gmail.com | monopoly            |             5 | very fun game                    |
| alex@gmail.com | Oakley hat          |             5 | very beautiful hat               |
| alex@gmail.com | office chair        |             5 | very comfy chair                 |
| alex@gmail.com | PS4                 |             5 | i like xbox better               |
| alex@gmail.com | tie                 |             5 | very beautiful tie               |
| alex@gmail.com | white board         |             5 | very useful board                |
+----------------+---------------------+---------------+----------------------------------+
15 rows in set (0.00 sec)
```

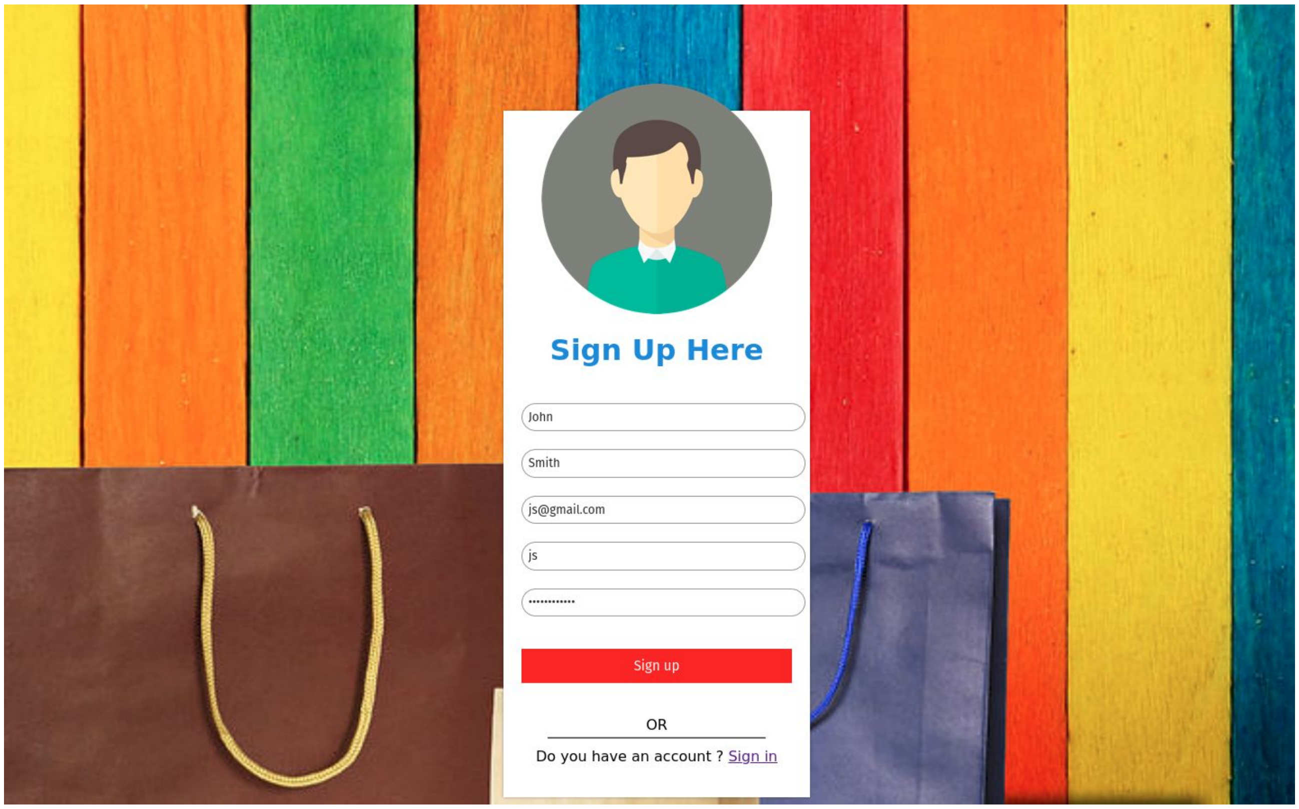


```
Database changed
mysql> SELECT * from product_image;
+---------------------+---------------------------------------------------------------------------+
| productName         | url                                                                       |
+---------------------+---------------------------------------------------------------------------+
| iphone 11           | https://dailydealsproductimages.s3-us-west-1.amazonaws.com/iphone11.png   |
| gucci belt bag      | https://dailydealsproductimages.s3-us-west-1.amazonaws.com/beltbag.jpg    |
| dell monitor        | https://dailydealsproductimages.s3-us-west-1.amazonaws.com/monitor.jpg    |
| dell monitor        | https://dailydealsproductimages.s3-us-west-1.amazonaws.com/monitor1.jpg   |
| air pod pro         | https://dailydealsproductimages.s3-us-west-1.amazonaws.com/airpods1.jpg   |
| air pod pro         | https://dailydealsproductimages.s3-us-west-1.amazonaws.com/airpods2.jpg   |
| go pro hero 7       | https://dailydealsproductimages.s3-us-west-1.amazonaws.com/gopro.jpg      |
| go pro hero 7       | https://dailydealsproductimages.s3-us-west-1.amazonaws.com/gopro1.jpg     |
| intro to linux book | https://dailydealsproductimages.s3-us-west-1.amazonaws.com/linux.jpg      |
| monopoly            | https://dailydealsproductimages.s3-us-west-1.amazonaws.com/monopoly.jpg   |
| monopoly            | https://dailydealsproductimages.s3-us-west-1.amazonaws.com/monopoly1.jpg  |
| Oakley hat          | https://dailydealsproductimages.s3-us-west-1.amazonaws.com/hat.jpg        |
| ap watch            | https://dailydealsproductimages.s3-us-west-1.amazonaws.com/apwatch.jpg    |
| ap watch            | https://dailydealsproductimages.s3-us-west-1.amazonaws.com/apwatch1.jpg   |
| ap watch            | https://dailydealsproductimages.s3-us-west-1.amazonaws.com/apwatch2.jpg   |
| tie                 | https://dailydealsproductimages.s3-us-west-1.amazonaws.com/tie.jpg        |
| white board         | https://dailydealsproductimages.s3-us-west-1.amazonaws.com/whiteboard.jpg |
| office chair        | https://dailydealsproductimages.s3-us-west-1.amazonaws.com/chair.jpeg     |
| PS4                 | https://dailydealsproductimages.s3-us-west-1.amazonaws.com/ps4.jpg        |
| PS4                 | https://dailydealsproductimages.s3-us-west-1.amazonaws.com/ps41.jpg       |
+---------------------+---------------------------------------------------------------------------+
20 rows in set (0.00 sec)
```

Implementation

Our database application was implemented using Java Server Pages for the front end Java Servlets and Java Objects to function as our middleware and finally MySQL connected using JDBC for our database. We used Apache Tomcat server to deploy Java Servlets and JSP. Furthermore, we followed the Data Access Object (DAO) pattern in order to perform specific data operations without exposing the details of our database.

Sign Up Page



Sign Up Here

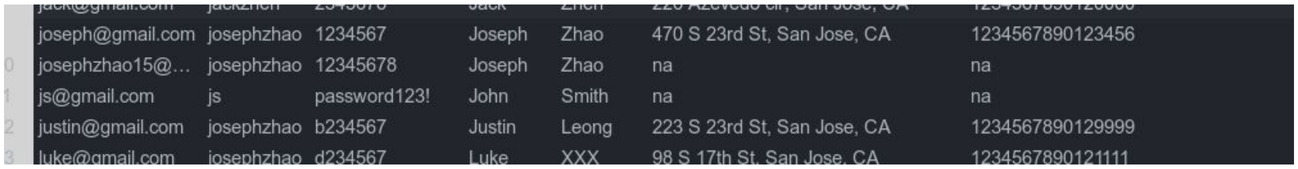
John

Smith

js@gmail.com

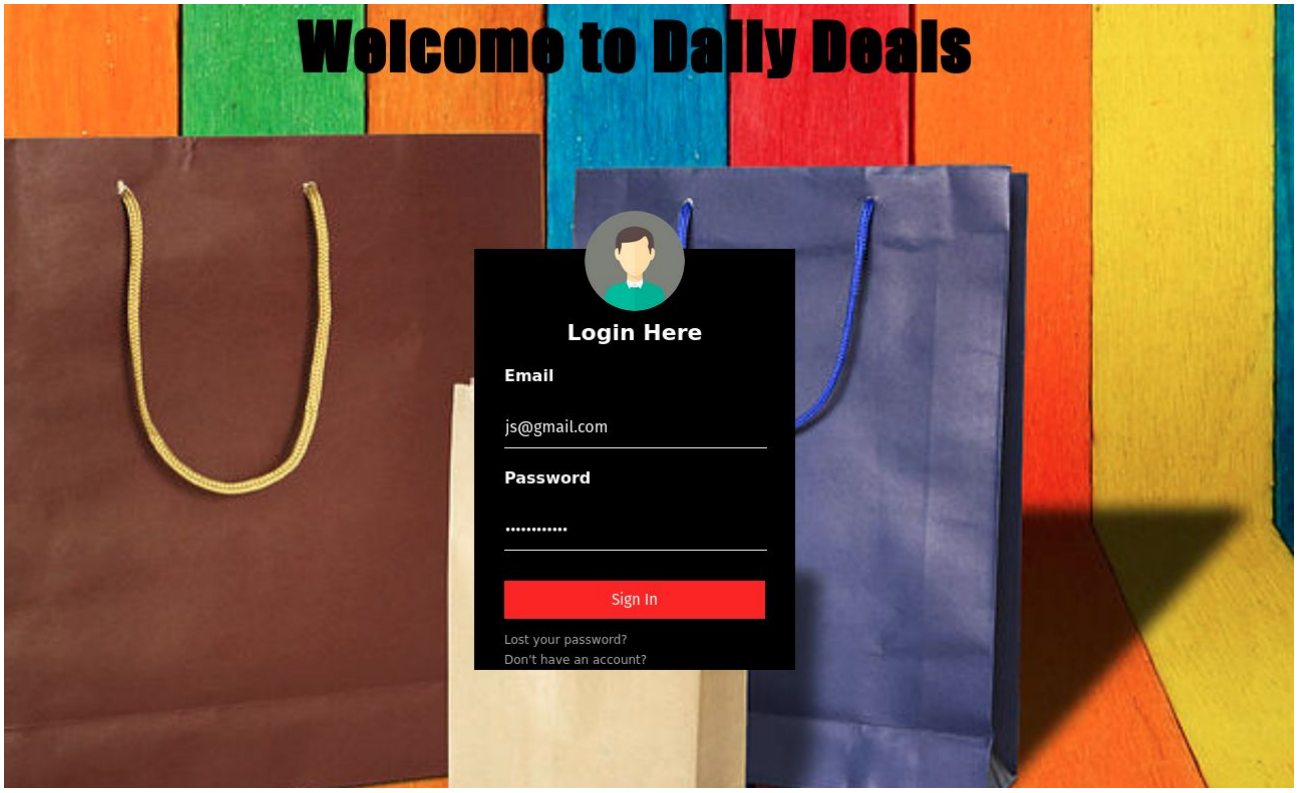
js

••••••••••••



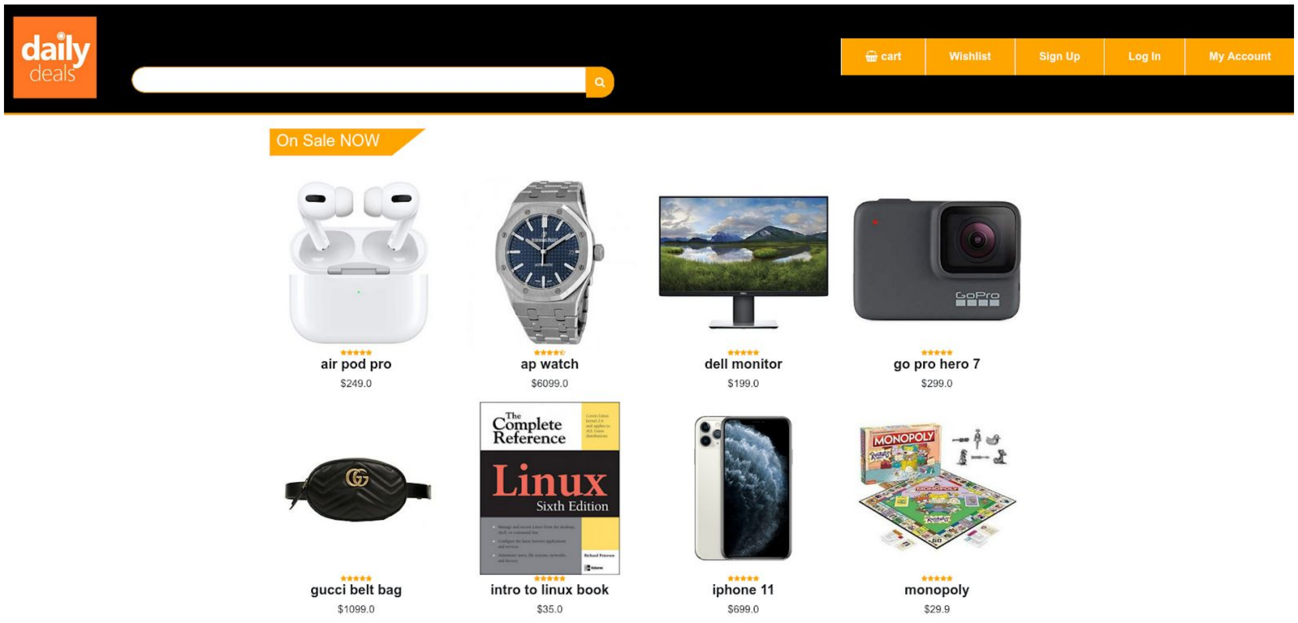
	jack@gmail.com	jackzhen	2345678	Jack	Zhen	220 Azevedo cir, San Jose, CA	[illegible]
	joseph@gmail.com	josephzhao	1234567	Joseph	Zhao	470 S 23rd St, San Jose, CA	1234567890123456
0	josephzhao15@…	josephzhao	12345678	Joseph	Zhao	na	na
1	js@gmail.com	js	password123!	John	Smith	na	na
2	justin@gmail.com	josephzhao	b234567	Justin	Leong	223 S 23rd St, San Jose, CA	1234567890129999
3	luke@gmail.com	josephzhao	d234567	Luke	XXX	98 S 17th St, San Jose, CA	1234567890121111

Login Page

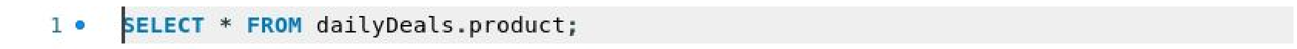


Email
js@gmail.com
Password
••••••••••••
Sign In
Lost your password?
Don't have an account?

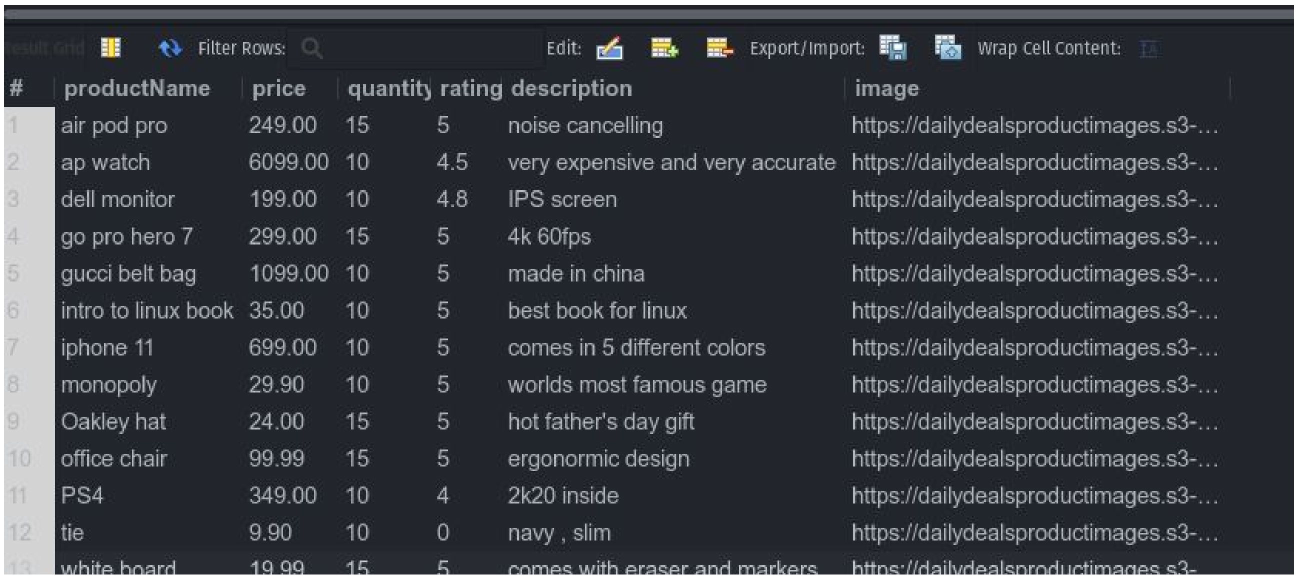
Main page



air pod pro $249.0
ap watch $6099.0
dell monitor $199.0
go pro hero 7 $299.0
gucci belt bag $1099.0
intro to linux book $35.0
iphone 11 $699.0
monopoly $29.9

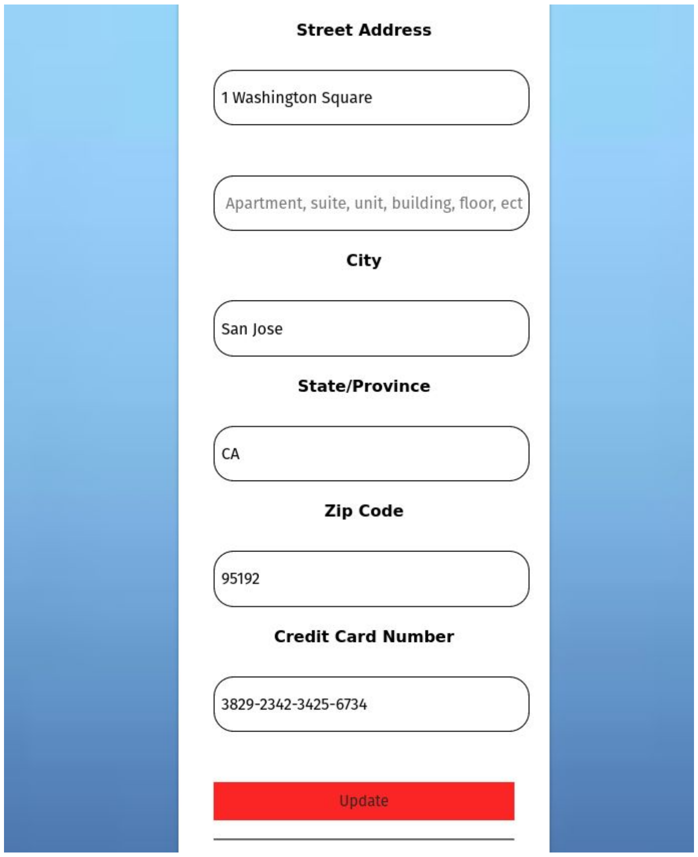


```sql
SELECT * FROM dailyDeals.product;
```

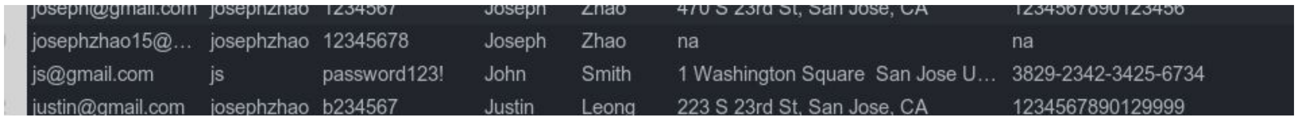


#	productName	price	quantity	rating	description	image
1	air pod pro	249.00	15	5	noise cancelling	https://dailydealsproductimages.s3-…
2	ap watch	6099.00	10	4.5	very expensive and very accurate	https://dailydealsproductimages.s3-…
3	dell monitor	199.00	10	4.8	IPS screen	https://dailydealsproductimages.s3-…
4	go pro hero 7	299.00	15	5	4k 60fps	https://dailydealsproductimages.s3-…
5	gucci belt bag	1099.00	10	5	made in china	https://dailydealsproductimages.s3-…
6	intro to linux book	35.00	10	5	best book for linux	https://dailydealsproductimages.s3-…
7	iphone 11	699.00	10	5	comes in 5 different colors	https://dailydealsproductimages.s3-…
8	monopoly	29.90	10	5	worlds most famous game	https://dailydealsproductimages.s3-…
9	Oakley hat	24.00	15	5	hot father's day gift	https://dailydealsproductimages.s3-…
10	office chair	99.99	15	5	ergonormic design	https://dailydealsproductimages.s3-…
11	PS4	349.00	10	4	2k20 inside	https://dailydealsproductimages.s3-…
12	tie	9.90	10	0	navy , slim	https://dailydealsproductimages.s3-…
13	white board	19.99	15	5	comes with eraser and markers	https://dailydealsproductimages.s3-

Account Management Page

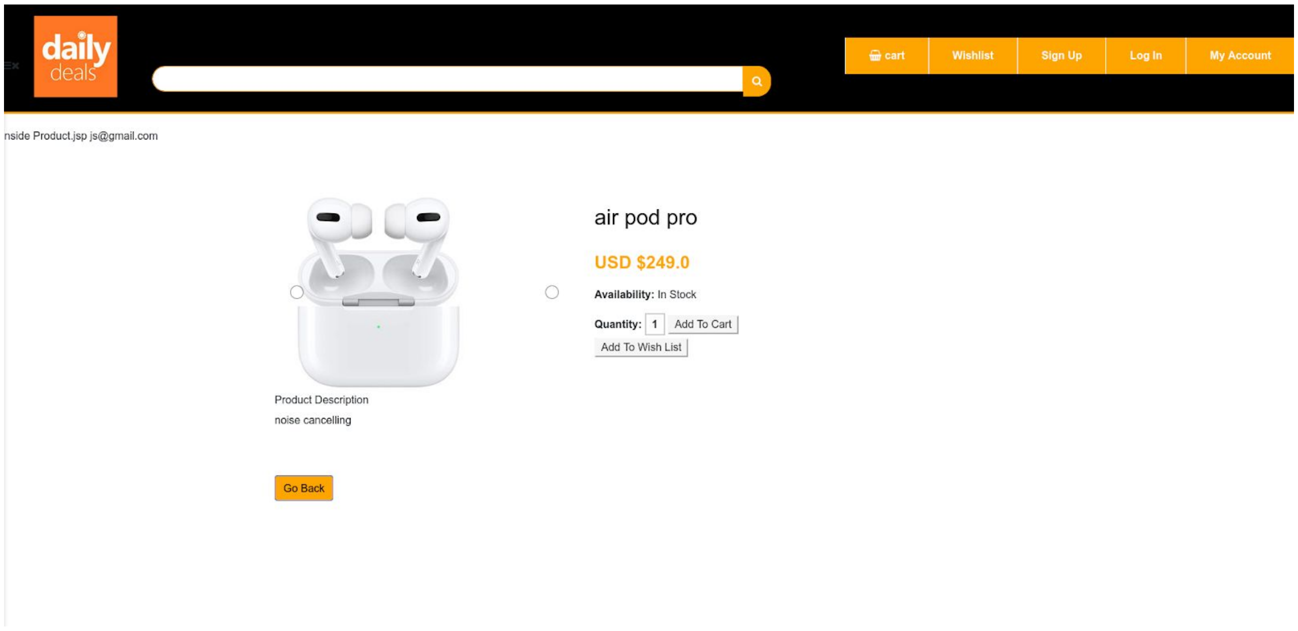


Street Address
1 Washington Square
Apartment, suite, unit, building, floor, ect
City
San Jose
State/Province
CA
Zip Code
95192
Credit Card Number
3829-2342-3425-6734



joseph@gmail.com	josephzhao	1234567	Joseph	Zhao	470 S 23rd St, San Jose, CA	1234567890123456
josephzhao15@…	josephzhao	12345678	Joseph	Zhao	na	na
js@gmail.com	js	password123!	John	Smith	1 Washington Square San Jose U…	3829-2342-3425-6734
justin@gmail.com	josephzhao	b234567	Justin	Leong	223 S 23rd St, San Jose, CA	1234567890129999

Product Page



nside Product.jsp js@gmail.com

air pod pro

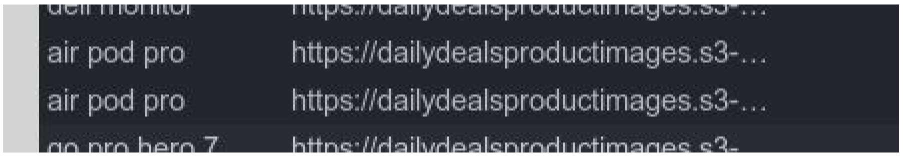
USD $249.0

Availability: In Stock

Quantity: 1

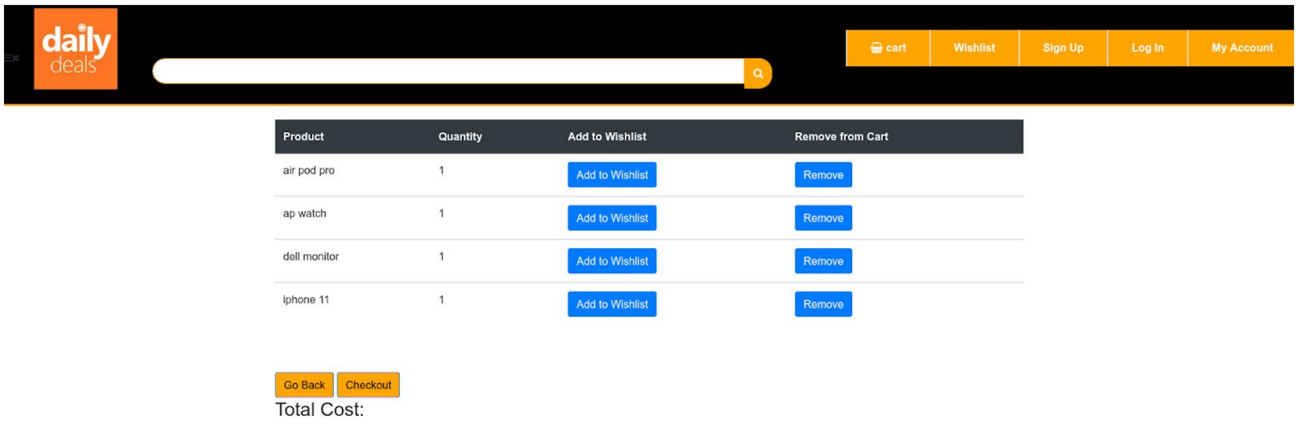
Product Description

noise cancelling



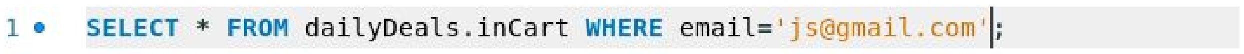
```
dell monitor     https://dailydealsproductimages.s3-…
air pod pro      https://dailydealsproductimages.s3-…
air pod pro      https://dailydealsproductimages.s3-…
go pro hero 7    https://dailydealsproductimages.s3-
```

Cart

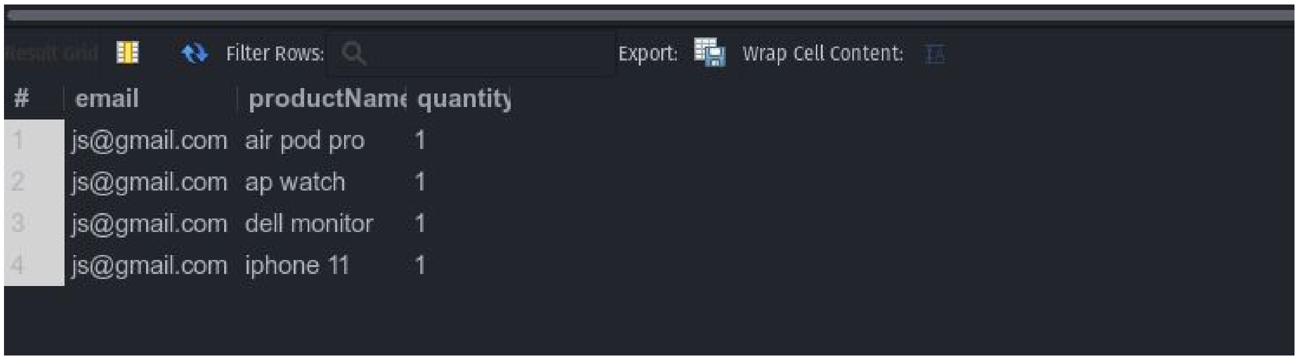


Product	Quantity	Add to Wishlist	Remove from Cart
air pod pro	1	Add to Wishlist	Remove
ap watch	1	Add to Wishlist	Remove
dell monitor	1	Add to Wishlist	Remove
iphone 11	1	Add to Wishlist	Remove

Total Cost:

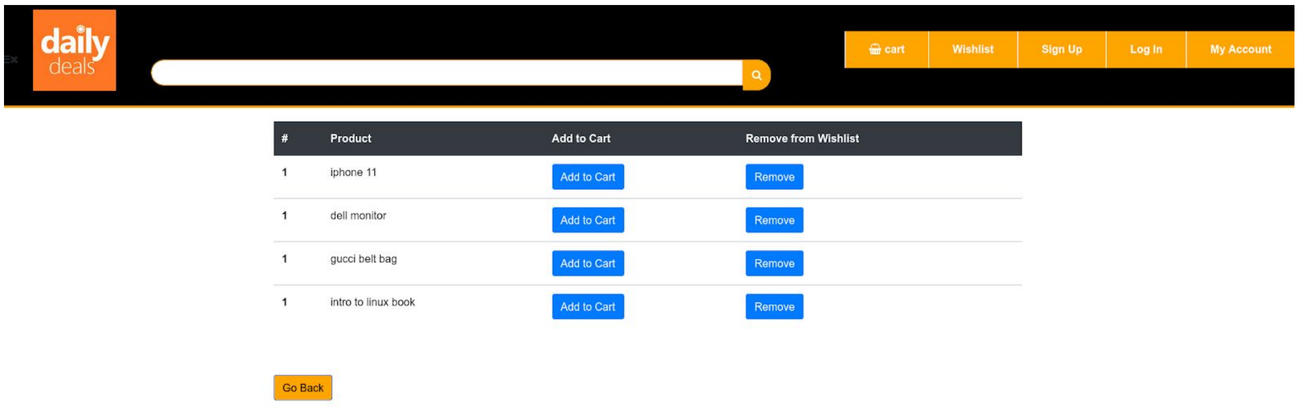


```
SELECT * FROM dailyDeals.inCart WHERE email='js@gmail.com';
```

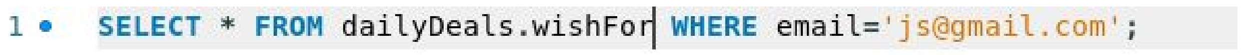


#	email	productName	quantity
1	js@gmail.com	air pod pro	1
2	js@gmail.com	ap watch	1
3	js@gmail.com	dell monitor	1
4	js@gmail.com	iphone 11	1

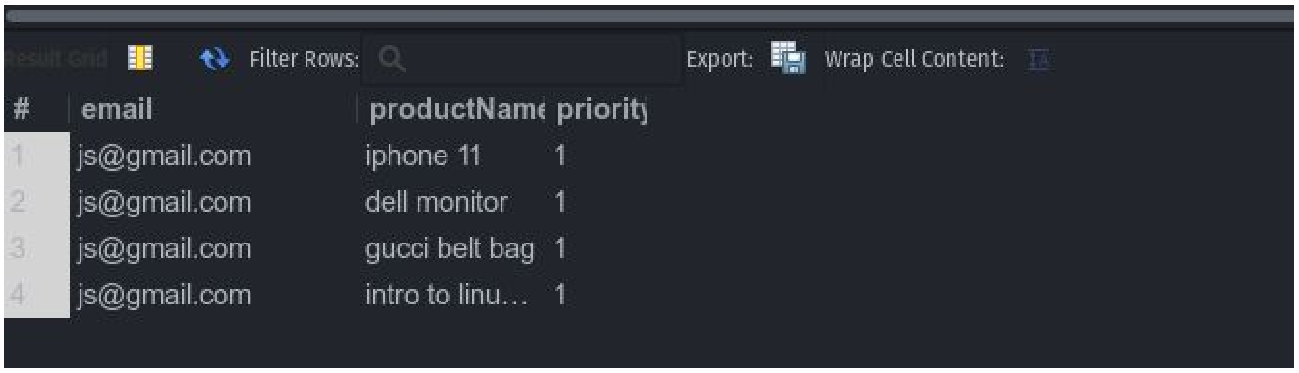
Wishlist



#	Product	Add to Cart	Remove from Wishlist
1	iphone 11	Add to Cart	Remove
1	dell monitor	Add to Cart	Remove
1	gucci belt bag	Add to Cart	Remove
1	intro to linux book	Add to Cart	Remove



```
SELECT * FROM dailyDeals.wishFor WHERE email='js@gmail.com';
```



#	email	productName	priority
1	js@gmail.com	iphone 11	1
2	js@gmail.com	dell monitor	1
3	js@gmail.com	gucci belt bag	1
4	js@gmail.com	intro to linu…	1

Running & Installation

To run our code there are a few considerations that are needed.

1. Before anything else you should set up a mysql database, I'll assume you know how to do that
    a. In MySQL create a new schema called dailyDeals
    b. Import our database file with default information into the dailyDeals schema:

https://drive.google.com/open?id=1y-sD3MT82hMu9CZrhh9USQuo6LQSTmJP

2. First you must clone our project onto your machine, then import the dailydeals folder from our repo into Eclipse EE.
3. Once imported you must install the JSTL and Connector J libraries (JDBC).
    a. Download version 1.2.7 of JSTL here https://github.com/eclipse-ee4j/jstl-api/releases
    b. Download the jar file of Connector J from here https://dev.mysql.com/downloads/connector/j/
    c. In Eclipse open the dailydeals project and then right click on it in Project Explorer
    d. Then click on properties.
    e. Click Java Build Path
    f. Select the Libraries Tab
    g. Click Add External JARs
    h. Select and add both the JSTL and Connector J jars
    i. Click Apply
    j. Click Deployment Assembly on the left hand side
    k. Click Add…
    l. Select Java Build Path Entries
    m. Select the Jar files you just added, you may have to do this one at a time
    n. click finish
    o. Finally click Apply and Close
4. You must also have Tomcat 8.5 or above downloaded in a directory
    a. You can download Tomcat 9 here https://tomcat.apache.org/download-90.cgi
5. You must also enter your MySQL username and password in all of the files contained in the following folder:
    a. dailydeals/src/main/java/DAOs
    b. Open each file and simply press ctrl + f and search for password to jump to the correct location in each file, insert your mysql password in between the quotes for password, and your username in the DAO objects as well
6. Now navigate through the project in eclipse to dailydeals/src/main/webapp
7. Right click on the loginPage or SignUp page
8. Choose run as run on server
    a. The first time it will ask you to configure the server
    b. Select the tomcat version you selected and then select the location you downloaded tomcat to
    c. Then click finish
9. The server should start and our application should run
10. if not right click on the jsp file again and then click run as run on server and that's it!

Project Conclusion

Brandon Archbold: I learned a lot about web development including how the front end talks to the backend particularly with JSP files, Java Servlets and accessing the database using database access objects in Java using JDBC. I also learned more about how to use a database, how database design can impact the project. I learned how to run and test a project using JSP, tomcat, and eclipse. I also learned how to pass information between all of the different files and tools we used. I also learned that we were a bit too ambitious with our functional requirements since we did not know any of the technologies we needed to use to build this project besides java.

Yilin Zhao: From this project, I learn in depth about the three-tier architecture with Java. I learned the configuration of a servlet web project and how to configure servlets with web.xml. I also learned about sessions and url parameter. I used them to pass variables in different scenarios. In this project, I used JSP for the first time. I learned JSP tags as well as JSTL.

Mandeep Pabla: I learned how to use JSP for web page scripting and Apache Tomcat for deploying Java Servlets and JSP. I learned how to use JDBC API to connect to our MySQL database. I learned how to use

CSS and HTML in order to create most of the front end. I created Servlets that used HTTP methods such as POST request to access information on the web page. I wrote code in the DAO java files to make queries and update tuples in our database.

Future Improvements

Future improvements to our project would include allowing users to view their order history in an easier to access location. We would also include a backend portal to sellers in the future as well as a way to add and remove products in the GUI. We could also host our application on AWS or a similar cloud platform.