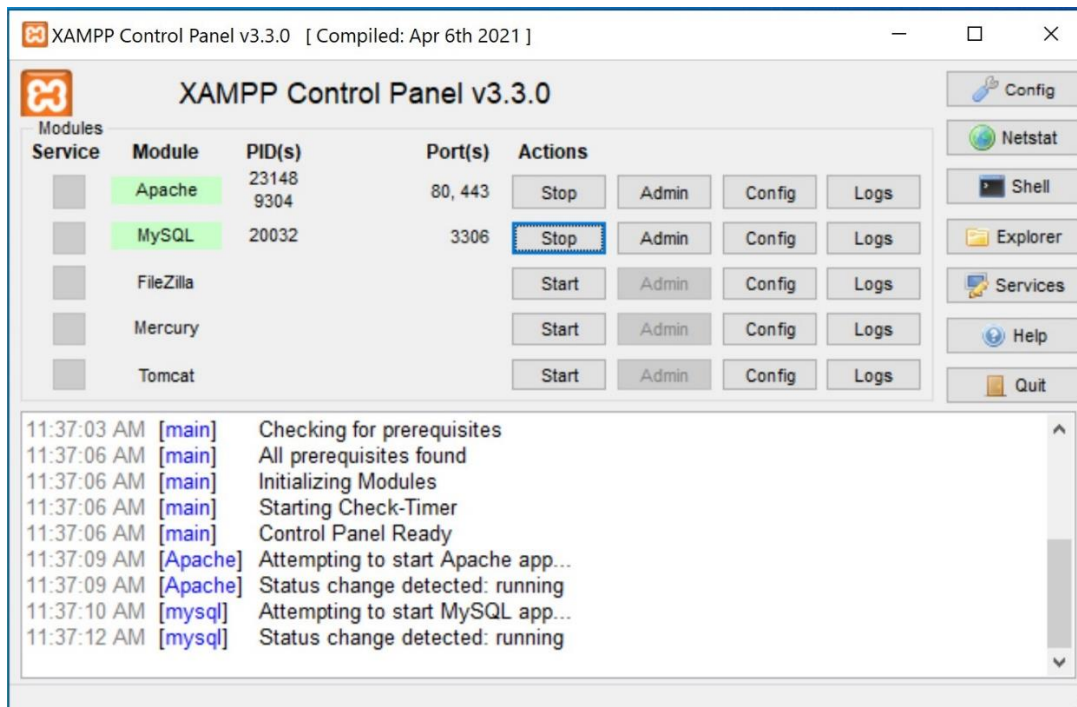


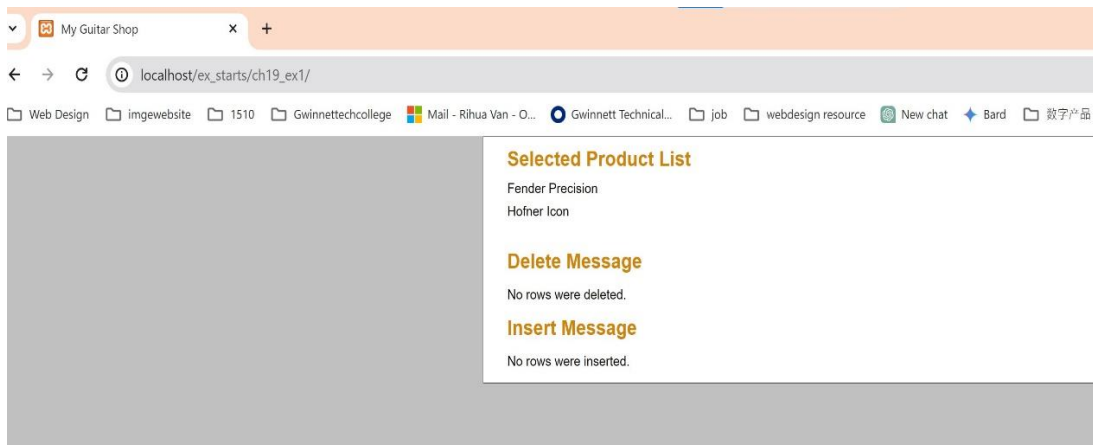
## Review and test the code.

### 1. Run the Application.

#### 1)Run Apache.



#### 2)Open ex\_starts/ch19\_ex1.



### 2. Review the Code.

Open the database.php, product\_db.php, category\_db.php, and index.php files in my code editor.

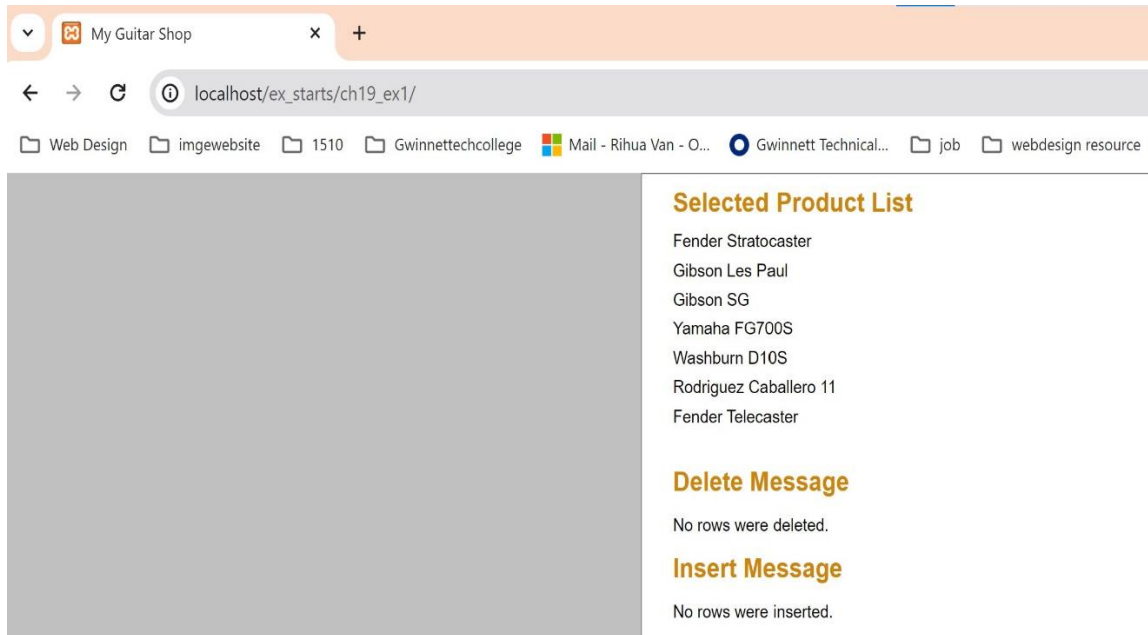
```
1 <?php
2 2 references | 0 implementations
3 class CategoryDB {
4     0 references | 0 overrides
5     public static function getCategoryies() {
6         $db = Database::getDB();
7         $query = 'SELECT categoryID, categoryName
8             FROM categories
9             ORDER BY categoryID';
10        try {
11            $statement = $db->prepare($query);
12            $statement->execute();
13            $rows = $statement->fetchAll();
14            $statement->closeCursor();
15
16            $categories = [];
17            foreach ($rows as $row) {
18                $categories[] = new Category($row['categoryID'],
19                    $row['categoryName']);
20            }
21            return $categories;
22        } catch (PDOException $e) {
23            Database::displayError($e->getMessage());
24        }
25    }
26 }
```

### 3. Switch to MySQLi: Review [book\\_apps/ch20\\_guitar\\_shop\\_mysqli/model](#).

```
1 <?php
2 10 references | 0 implementations
3 class CategoryDB {
4     6 references | 0 overrides
5     public static function getCategoryies() {
6         $db = Database::getDB();
7         $query = 'SELECT categoryID, categoryName
8             FROM categories
9             ORDER BY categoryID';
10        try {
11            $result = $db->query($query);
12            $categories = [];
13            for ($i = 0; $i < $result->num_rows; $i++) {
14                $row = $result->fetch_assoc();
15                $categories[] = new Category($row['categoryID'],
16                    $row['categoryName']);
17            }
18            $result->free();
19            return $categories;
20        } catch (mysqli_sql_exception $e) {
21            Database::displayError($e->getMessage());
22        }
23    }
24
25    4 references | 0 overrides
26    public static function getCategory($category_id) {
27        $db = Database::getDB();
28        $query = 'SELECT categoryID, categoryName
29            FROM categories
30            WHERE categoryID = ?';
31    }
```

## 4. Modify the code and display Guitars Category.

```
1 <?php
2 require_once('util/main.php');
3
4 require_once('model/database.php');
5 require_once('model/product_db.php');
6 require_once('model/category_db.php');
7 require_once('model/product.php');
8 require_once('model/category.php');
9
10 /*****
11  * Select some products
12  *****/
13
14 // Sample data
15 // $cat_id = 2;
16
17 $cat_id = 1;
18
19 // Get the products
20 $products = ProductDB::getProductsByCategory($cat_id);
21
22 /*****
23  * Delete a product
24  *****/
25
26 // Sample data
27 $product_name = 'Fender Telecaster';
28
```



## 5. Add a Product Method.

```
53 $price = '949.99';
54 $discount_percent = '10';
55
56 // Insert the data(Add a product)
57 $category = CategoryDB::getCategory($category_id);
58 $product = new Product($category, $code, $name, $description, $price, $discount_percent);
59 $product_id = ProductDB::addProduct($product);
60
61 // Display an appropriate message
62 if ($product_id > 0) {
63     $insert_message = "1 row was inserted with this ID: $product_id";
64 } else {
65     $insert_message = "No rows were inserted.";
66 }
67
68 include 'home.php';
69 ?>
```

## 6. Create getProductByName() Method

Add a new method named `getProductByName()` in the model (likely within `product_db.php`) that accepts a product name as a parameter.

```
60 }
61
62 1 reference | 0 overrides
63 public static function getProductByName($name) {
64     $db = Database::getDB();
65     $query = 'SELECT categoryID, productID, productCode, productName,
66             description, listPrice, discountPercent
67             FROM products
68             WHERE productName = :name';
69     try {
70         $statement = $db->prepare($query);
71         $statement->bindValue(':name', $name);
72         $statement->execute();
73
74         $row = $statement->fetch();
75         $statement->closeCursor();
76
77         if ($row) {
78             $category = CategoryDB::getCategory($row['categoryID']);
79             return self::loadProduct($row, $category);
80         } else {
81             return NULL;
82         }
83     } catch (PDOException $e) {
84         Database::displayError($e->getMessage());
85     }
86 }
```

## 7. Delete Product Using New Method

Modify the `index.php` file to use the `getProductByName()` method to fetch the ID of the product named “Fender Telecaster”.

Use the product ID to delete the product from the database, and then display a message indicating whether the product was successfully deleted.

```
22
23 /*****
24  * Delete a product
25  *****/
26
27 // Sample data
28 $product_name = 'Fender Telecaster';
29
30 // Delete the product and display an appropriate message
31 $product = ProductDB::getProductByName($product_name);
32 if ($product) {
33     $product_id = $product->getID();
34     $row_count = ProductDB::deleteProduct($product_id);
35     if ($row_count > 0) {
36         $delete_message = "$row_count row was deleted.";
37     } else {
38         $delete_message = "No rows were deleted.";
39     }
40 } else {
41     $delete_message = "There is no product with that name.";
42 }
43
```

## 8. Test the Code

