**CEN 4010 Principle of Software Engineering, Spring 2018**

*Team Name: Grupo Fivo*

*Team 5*

**Neil Maniquis**

nmaniquis2017@fau.edu

**Noah Leach**

nleach2013@fau.edu

**Diego Segura**

dsegura2015@fau.edu

**George Bechtel**

gbechtel2013@fau.edu

**Franklin Carrillo**

fcarrillo2012@fau.edu

**Date: 4/19/18**

**Product summary**

Our website is called Perry’s Parts Pavillion Access Center.

Committed functions:

* List all active products: <http://lamp.eng.fau.edu/~CEN4010_S2018g05/links.php>
* Search by product name or keyword: <http://lamp.eng.fau.edu/~CEN4010_S2018g05/>
* Customer account creation
* User login
* Staff can manually add item information

PPPAC supports partial word search. ex searching for “ohm” will bring up all products with “ohm” in the product name or keyword.

When creating an account, there is a realtime update to show if the passwords match. This is to help insure that the password was entered without error.

Customers can:

* create an account
* view items
* order items to be shipped
* upload files to be 3D printed

Vendors can:

* create an account
* list items for sale

**Usability test plan**

We will be testing the product/keyword search for the web site. The objectives we will test for are:

* Is the search function easy to use?
* Are the search results accurate?
* Are the search results fast?

The system which the test will be performed on must include a web browser and an internet connection. The intended user is a student at FAU. They should go to <http://lamp.eng.fau.edu/~CEN4010_S2018g05/>. There, they will find a search form titled “Search product names and keywords” where they can enter a query.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Strongly Disagree | Disagree | Neutral | Agree | Strongly Agree |
| Are the search results accurate? |  |  |  |  |  |
| Does a search query produce results quickly? |  |  |  |  |  |
| Is the search function easy to use? |  |  |  |  |  |

**QA test plan**

The objectives we will test are:

* Does the search query produce distinct results?
* Do duplicate links appear when there are multiple identical products?
* Can the search query find results with incomplete search terms?

In addition to the basic requirements of having a web browser and an internet connection, the web browsers used should be either Chrome or Firefox.

The product/keyword search will be tested.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Browser | Test # | Test title | Test description | Test input | Expected output | Test results |
| Chrome | 1 | Distinct Results | Does the search query produce distinct results? | 5% MCF 0.25W | 4 links to 0.25W resistors | PASS |
| Chrome | 2 | Duplicate Links | Do duplicate links appear when there are multiple identical products? | 1 Ohm | Link to a 1 ohm resistor, but not to a 1k ohm resistor | PASS |
| Chrome | 3 | Search Fragments | Can the search query find results with incomplete search terms? | Diod | Link to a diode | PASS |
| Firefox | 1 | Distinct Results | Does the search query produce distinct results? | 5% MCF 0.25W | 4 links to 0.25W resistors | PASS |
| Firefox | 2 | Duplicate Links | Do duplicate links appear when there are multiple identical products? | 1 Ohm | Link to a 1 ohm resistor, but not to a 1k ohm resistor | PASS |
| Firefox | 3 | Search Fragments | Can the search query find results with incomplete search terms? | Diod | Link to a diode | PASS |

**Code Review**

The coding style used includes the use of comments to help clarify code and a tab size of 4.

**Neil:** Here is my code to search the database for a matching product/keyword. I attempted to put many notes to help you understand the code. This section is taken from searchresult.php

<?php

try {

//form var

//htmlspecialchars converts certain char to help prevent hacking

$findtxt = $\_POST[htmlspecialchars("keywordinput")];

//puts % incase string is in the middle of a keyword

$wildcardfindtxt = "%" . $findtxt ."%";

//creates empty array to hold history of results

$listresult = array();

//entering mysql

$servername = "localhost";

$username = "CEN4010\_S2018g05";

$password = "SQLgroup5";

//connecting to mysql by PDO

$conn = new PDO("mysql:host=$servername;dbname=CEN4010\_S2018g05",trim($username),trim($password));

$conn->setAttribute(PDO::ATTR\_ERRMODE, PDO::ERRMODE\_EXCEPTION);

//searching products with matching keyword string

//returns product and link pair

//prepared statments help prevent hacking

//required search as not all products have keywords

$stmt = $conn->prepare("SELECT \* FROM Link WHERE

Product LIKE :Product");

$stmt->bindParam(':Product', $wildcardfindtxt);

$stmt->execute();

$flag = $stmt->setFetchMode(PDO::FETCH\_ASSOC);

$result= $stmt->fetchAll();

if (count($result) > 0) {

//at least 1 product was found; display on screen

for ($x = 0; $x < count($result); $x++){

echo '<br>

<a href="'.$result[$x]["Link"].'/index.html">'.$result[$x]["Product"].'</a>

';

array\_push($listresult, $result[$x]["Product"]);

}

}

//searches keyword table and returns a product array

$sql = $conn->prepare("SELECT Product FROM Keyword WHERE Keyword LIKE :Keyword

");

$sql->bindParam(':Keyword', $wildcardfindtxt);

$sql->execute();

$flag = $stmt->setFetchMode(PDO::FETCH\_ASSOC);

$result= $stmt->fetchAll();

$resultremaining = array();

if (count($result) > 0) {

//at least 1 product was found

for ($x = 0; $x < count($result); $x++){

//compairs results with previous list

if(!in\_array($result[$x]["Product"], $listresult)){

array\_push($reusltremaining, $result[$x]["Product"]);

}

}

}

if (count($resultremaining) >0){

//get product link pair

$sql = $conn->prepare("SELECT \* FROM Link WHERE Product LIKE :Product

");

foreach($resultremaining as $item){

$sql->bindParam(':Product', $item);

$sql->execute();

$flag = $stmt->setFetchMode(PDO::FETCH\_ASSOC);

$result= $stmt->fetchAll();

if (count($result)> 0){

for ($x = 0; $x < count($result); $x++){

echo '<br>

<a href="'.$result[$x]["Link"].'/index.html">'.$result[$x]["Product"].'</a>

';

}

}

}

unset($item);

}

elseif(count($listresult) == 0){

echo "<p>0 results found for " . $findtxt .'</p>';

}

}

catch(PDOException $e){

echo "Connection failed: " . $e->getMessage();

}

catch(Exception $e){

echo $e->getMessage();

}

?>

**George:** Great. My plan is to work on Milestone 4 tomorrow and then after Tuesday and most of the rest of the week I'll be working on the web store.

**Neil:** This code now supports keywords. The duplicate links are caused by multiple instances of the same product in the database.

echo $wildcardfindtxt. " wildcardfindtxt";

$sql->execute();

$result= $sql->fetchAll();

$resultremaining = array();

if (count($result) > 0) {

//at least 1 product was found

for ($x = 0; $x < count($result); $x++){

//compairs results with previous list

if(!in\_array($result[$x]["Product"], $listresult)){

array\_push($resultremaining, $result[$x]["Product"]);

}

}

}

if (count($resultremaining) >0){

//get product link pair

$sql = $conn->prepare("SELECT \* FROM Link WHERE Product LIKE :Product

");

foreach($resultremaining as $item){

$sql->bindParam(':Product', $item);

$sql->execute();

$flag = $stmt->setFetchMode(PDO::FETCH\_ASSOC);

$result= $sql->fetchAll();



if (count($result)> 0){

for ($x = 0; $x < count($result); $x++){

echo '<br>

<a href="'.$result[$x]["Link"].'/index.html">'.$result[$x]["Product"].'</a>

';

}

}

}

unset($item);

}

elseif(count($listresult) == 0){

echo "<p>0 results found for " . $findtxt .'</p>';

}

}

catch(PDOException $e){

echo "Connection failed: " . $e->getMessage();

}

catch(Exception $e){

echo $e->getMessage();

}

?>

**Self-check on best practices for security**

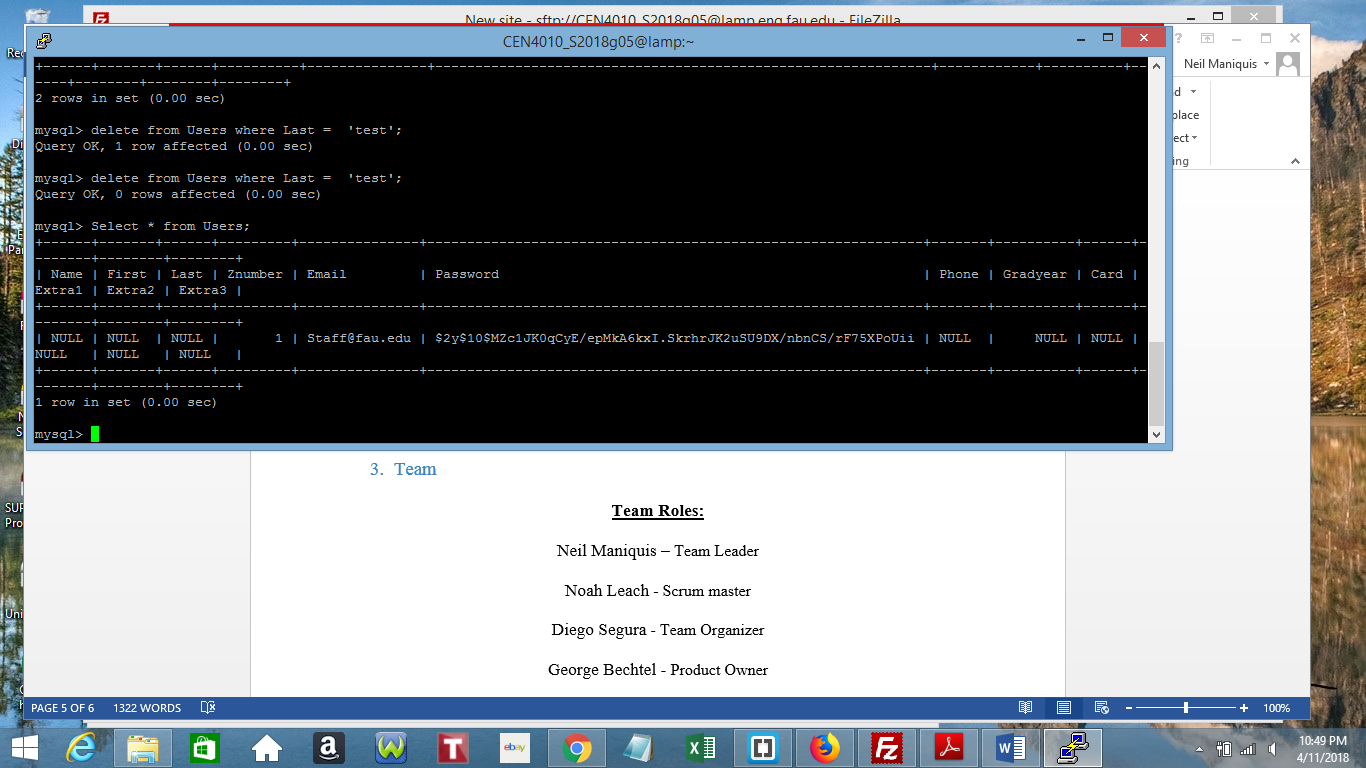
Major assets we are protecting:

Emails, passwords, product information, order history, class history

Database password encryption:

Passwords are encrypted using

password\_hash($password, PASSWORD\_DEFAULT);



Input data validation:

Data obtained through textboxes are passed through htmlspecialchars()

$email = $\_POST[htmlspecialchars("email")];

Accessing the DB is done through prepared statements

$sql = $conn->prepare("SELECT Product FROM Keyword WHERE Keyword = :Keyword ");

$sql->bindParam(':Keyword', $wildcardfindtxt);

$sql->execute();

**Self-check: Adherence to original Non-functional specs**

1. Ability to go to any item page from the homepage within 1 minute – DONE

2. Ability to create a new user account under 5 minutes – DONE

3. Ability to add a new item for sale under 8 minutes – DONE

4. Ability to update the inventory of 10 items in under 5 minutes – ISSUE – Requirement is related to a non-committed function

5. Passwords are not stored in plaintext and identical passwords will be stored differently for different users – DONE

6. Usability of website for customer on a mobile device should be within 1 minute of desktop experience – ON TRACK

**Video Demo**

<https://youtu.be/E31DpzTzXH0>

**Team**

**Team Roles:**

Neil Maniquis – Team Leader

Noah Leach - Scrum master

Diego Segura - Team Organizer

George Bechtel - Product Owner

Franklin Carrillo - Web developer