Course: CEN 4010 (Principles of Software Engineering)

Semester: Spring 2018

Milestone 4: Beta Launch and Reviews

Team name: Five Bros Chilling in a Hot Tub (Group 3)

Project name: Florida Atlantic University Original Web eLectronics Store (FAUOWLS)

Demo video link: <https://youtu.be/ThvnzmyEmhc>

Team:

* Joshua Cordes, Product Owner, Developer, jcordes@fau.edu
* Martin Daniel Sanchez, Scrum Master, Developer, martinsanche2016@fau.edu
* Brett Marks, Front-end Team Lead, Developer, marksb2016@fau.edu
* Alvaro Feola, GitHub Master, Developer, afeola2013@fau.edu
* William Young, Backend Team Lead, Developer, wyoung2012@fau.edu

Date: 16 April 2018

|  |  |
| --- | --- |
| History Table | |
| 19 Feb. 2018 | Initial submission of Milestone 1 |
| 22 March 2018 | Revisions of Milestone 1 added to Milestone 3 |
| 26 March 2018 | Initial submission of Milestone 3 |
| 16 April 2018 | Initial submission of Milestone 4 |

2.2 Product Summary

To facilitate inventory-tracking, we are producing for FAU’s Perry Weinthal the Florida Atlantic University Original Web eLectronics Store (FAUOWLS), where students and university employees will be able to search for and purchase electronics items they seek, and Perry and his team will be able to keep track of all equipment and service requests as they come in and go out. FAUOWLS will allow students and faculty at FAU can purchase or rent, at little to no cost, many electronic parts that might otherwise be pricey for those on a budget.

Major committed functions:

* Prospective users create an account using their FAU Z-number as an account name.
* Items are separated into product Categories, navigable from the home page.
* Admin imports to Inventory a spreadsheet file with info on Items to update or add.
* Admin exports from Inventory a spreadsheet file with info on Items currently in Inventory.
* Admin creates Kits containing multiple existing Items in Inventory, with the correct Quantity of each Item subtracted from Inventory when each quantity of the Kit is added.
* Admin sees in-browser table of current Inventory.

Unique features:

* Site includes items separated by Categories, avoiding potentially inefficient searches.
* When uploading a spreadsheet to Inventory, if an Item’s Category is not yet in the database, a new one is added to which other Items can then be added in the future.
* When Admin/Staff cancels an order, Inventory Quantities are updated automatically.

Product links:

* Home page, geared toward Clients: <http://lamp.cse.fau.edu/~CEN4010_S2018g03/FAUOWLS/>
* Inventory page for Admin and Engineering Lab Staff: http://lamp.cse.fau.edu/~CEN4010\_S2018g03/FAUOWLS/inventory/

2.3 Usability Test Plan

Test objectives:

Among the most important functionalities of web store Florida Atlantic University Original Web eLectronics Store (FAUOWLS) is the ability of an Admin to upload a spreadsheet of Items to be sold or rented in FAUOWLS. The primary objective, then, of this usability test is to show that the user can upload to FAUOWLS an attachment in the form of a comma-separated value (CSV) file, with the correct field data types for Items in Inventory, and the info will populate the database.

If the user attempts to upload a document of incorrect file type, the attempt shall be rejected and no changes made to the database. For a given Item in the spreadsheet, if it is listed in the CSV file as having the same Part Number as an item already in Inventory, this info shall update the appropriate fields in the database. If a category that is not on the database is listed in the CSV file, then it shall be created on the Categories table of the database with an appropriate Category code. If an upload is successful, the user shall see in Inventory that Items meant to be added or updated were in fact done so and that no additional unintended changes were made.

Test plan:

* System setup: User shall have Admin privileges and a computer with Internet access and any standard web browser, such as Mozilla Firefox or Google Chrome.
* Starting point: User shall have one browser tab or window open with the list of current Inventory. This represents what is in the Inventory database prior to user’s test.
* Task to be accomplished:
  + User uploads CSV file with appropriate fields for Inventory items: perry\_part\_num, short\_description, long\_description, location\_in\_lab, quantity, purchase\_or\_rent, retail\_price, retail\_price\_promo, retail\_markup, jobber\_price, jobber\_markup, bulk\_price, bulk\_markup, cost\_to\_replace, cost\_avg, category\_code, category\_name, barcode1, barcode2, barcode3, barcode4, barcode5
* Intended user: FAUOWLS user with Admin privileges
* Completion criteria: Inventory automatically updates information on Items already in database and adds information on new items from user’s CSV.
* URL of the system being tested:
  + Main URL: <http://lamp.cse.fau.edu/~CEN4010_S2018g03/FAUOWLS/>
  + Tested function direct URL: <http://lamp.cse.fau.edu/~CEN4010_S2018g03/FAUOWLS/Testing/upload.html>
* Questionnaire form:
  + Rate your response to each question on the following scale:
    1. Completely unsatisfied
    2. Somewhat unsatisfied
    3. Neutral
    4. Somewhat satisfied
    5. Completely satisfied
  + How satisfied were you with the ease of use in updating Inventory through spreadsheet upload? \_\_\_\_\_\_
  + How satisfied were you with the time span between start of upload attempt and seeing information updated in Inventory? \_\_\_\_\_\_
  + How satisfied were you with the convenience of use of this function for updating information on many Items all at once? \_\_\_\_\_\_

2.4 QA Test Plan

**Test objective:**

The objective of the QA Test Plan is to test our software and seek out these major objectives. These objectives include finding defects, gaining confidence in the quality of our software, to prevent any future defects, to make sure the end result meets the business and user requirements, and to gain confidence of the customer by providing a demo of the software through a beta demo.

We will be achieving this through three test objectives. The first test objective being to see if the upload function is capable of accepting a CSV file and creating a new product with a new category and respective barcode. The second test objective test if a CSV file can be uploaded with an existing product and update the products inventory in the database. The last objective of our test will be successfully uploading a CSV file and updating a product, creating a product, creating new categories, and creating respective barcodes all at the same time to test the stability of the program with large CSV files.

**Hardware and Software setup:**

* Windows 7 OS
* Intel Core i7 - 4710HW @2.50 GHz
* 64 bit OS
* Mozilla and Chrome browsers
* <http://lamp.cse.fau.edu/~CEN4010_S2018g03/FAUOWLS/Testing/upload.html>
* PHP/MySQL
* HTML

**Feature to be tested:**

The feature being tested is the upload function that allows a user with admin privileges to input a CSV file into the software and update the inventory within the database. The user should be able to attach a CSV file to the upload page and update items in the database such as inventory #, product category, product barcode, product description etc. Through this test will be testing four features within the upload function which cover what will be inputted by the user in most cases.

The first feature is updating an inventory product that is already within the database. The user will upload a CSV file with new values for whatever he/she chooses, and the item will be updated within the database with the given values. The second feature to be tested is uploading a CSV file with new inventory items. When the user uploads a CSV file with a new product, the function should add new inventory to the database and fill in all the parameters given through the file. The third feature is adding specific barcodes that do not overlap to each item if specified in the CSV file. The last feature is a culmination of the first three in which a user can upload a CSV file and make changes, add products and categories, or add barcodes all within the same file.

**Actual test:**

* 1. Test case 1: adding new product with new category
     1. New column should be made within the database
     2. New category will be added if CSV file has the correct parameters
     3. No unneeded changes will occur because of these changes
  2. Test case 2: update inventory that's already in database with new barcode
     1. No new columns will be created
     2. Barcode for specific items will be added to the correct table
     3. No unneeded changes will occur because of these changes.
  3. Test case 3: update inventory, create barcodes, and create new products all at once.
     1. New barcode should be made within the correct table for either new or existing products
     2. New category and Product will be added if CSV file has correct parameters
     3. New category will be added to existing product if CSV file has correct parameters.
     4. CSV file will update existing inventory, barcodes, and add a category simultaneously.

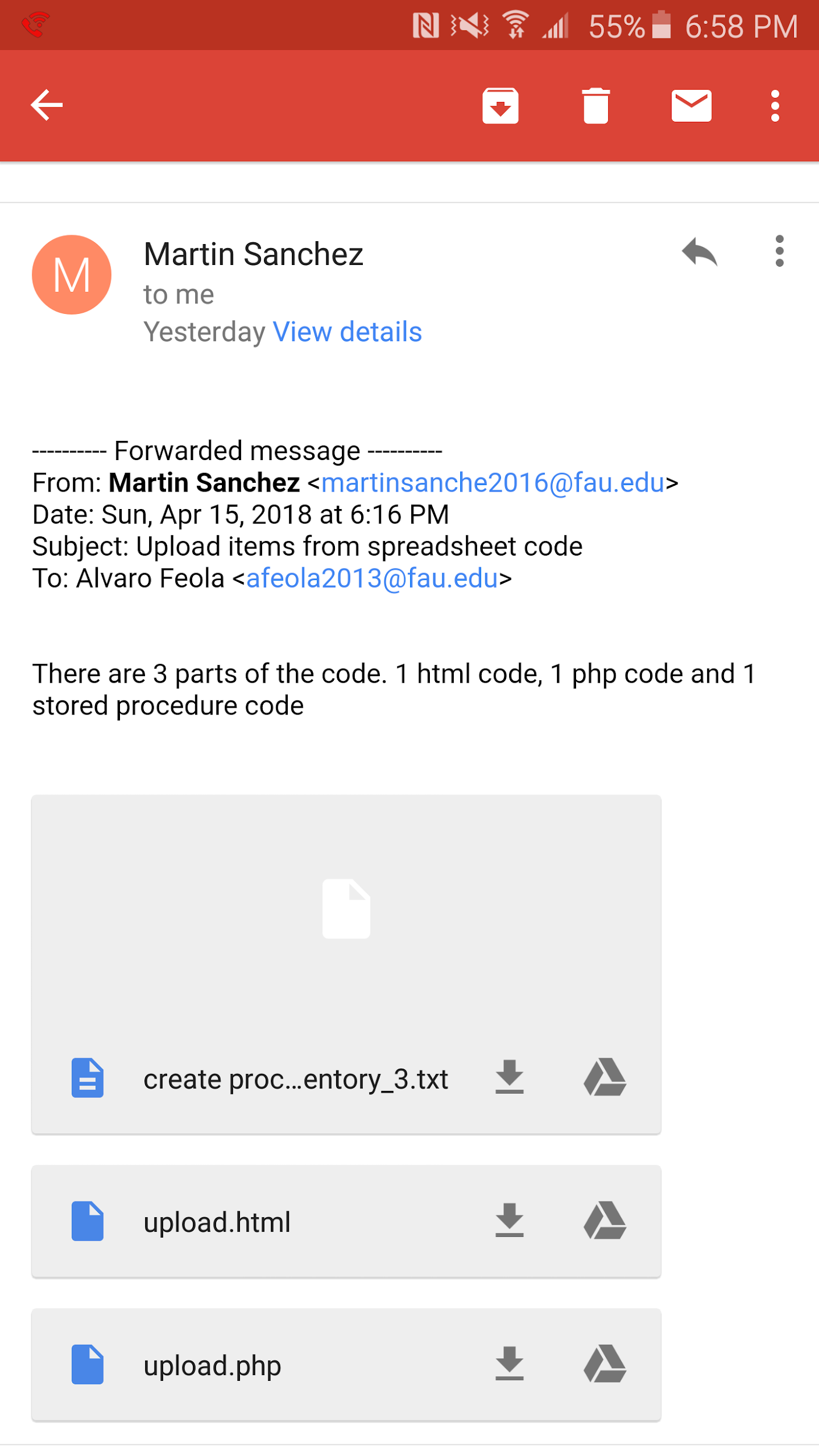
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test #** | **Test Title** | **Test description** | **Test input (csv file)** | **Expected output** | **Results** | **Browser** |
| 1 | Moz1 | Add new item in inventory with new category | Testitem08 | new item inventory | Pass | Mozilla |
| 2 | Chr1 | Add new item in inventory with new category | Testitem07 | new item inventory | Pass | Chrome |
| 3 | Moz2 | update inventory and add new barcode | Testing2 | updated inventory # | Pass | Mozilla |
| 4 | Chr2 | update inventory and add new barcode | Testing3 | updated inventory # | Pass | Chrome |
| 5 | Moz3 | update 8 items, create 2 barcodes, create 2 new products | Testitem0-10 | Update 8 items, create 2 new barcodes and create 2 products all at once | Pass | Mozilla |
| 6 | Chr3 | update 8 items, create 2 barcodes, create 2 new products | Testitem0-10 | Update 8 items, create 2 new barcodes and create 2 products all at once | Pass | Chrome |

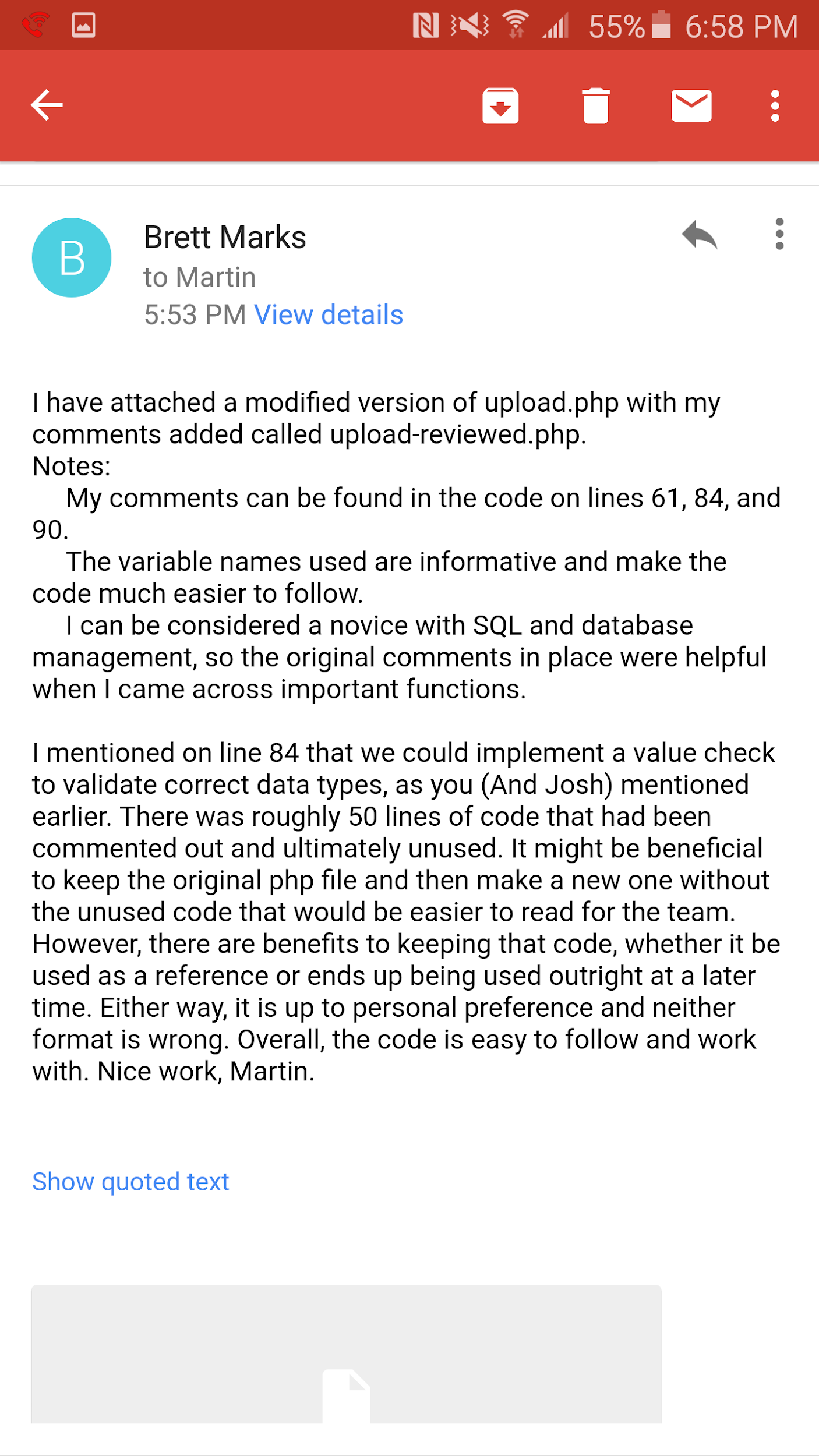
2.5 Code Review

Although a particular coding style was not discussed, all members of our team abided by the typical rule set of descriptive comments and meaningful variable names within the code. This made working together easier, since our differing levels of experience in different programming languages could have led to sometimes difficult-to-comprehend functions had these conventions not been in place. FAUOWLS integrates the typical web development languages that are found in the industry (HTML, CSS, Javascript, PHP and SQL). Most team members worked heavily in all of the listed languages, while the team leads worked mostly with their respective languages. Proper coding style in HTML helped the back-end team read and understand code that they were not working on directly, but still needed to understand the functionality. The same is said for well-commented code on the back end that helped the front-end team read and comprehend back-end functions and procedures. As stated before, most of the team worked heavily on both sides of the project, and writing readable code was a major component in working efficiently as a group.  
 The following is a sample from the code submitted for peer review and the emailed feedback:

* //Connect to Database and import file data into Import\_Inventory table
* $db = NEW MySQLi("localhost", "CEN4010\_S2018g03", "cen4010\_s2018", "CEN4010\_S2018g03") **or** **die**("Couldn't connect to database:<br>" . mysqli\_error($db). "<br>" . mysqli\_errno($db)); //Good use of the die() function. From my experience, the exit() function would also suffice. - Brett Marks

* **if**(!mysqli\_select\_db($db,"CEN4010\_S2018g03"))
* **die**("Couldn't select database:<br>" . mysqli\_error($db). "<br>" . mysqli\_errno($db));
* **if** (!$db->query("TRUNCATE TABLE Import\_Inventory")) {
* echo "TRUNCATE failed: (" . $db->errno . ") " . $db->error;
* }
* $filename=$target\_file;
* $handle = fopen($filename, 'r');
* $first = 0;
* **while** (($data = fgetcsv($handle, 1000, ',')) !== FALSE){
* **if** ($first==0){
* $first =1;
* }
* **else**{
* //echo "Dato $data[17]";
* **for**($i = sizeof($data); $i <22;$i++) //I believe this loop would be a good place to check if data being imported matches the proper format. -Brett Marks
* {
* $data[$i] = "";
* }
* $import="INSERT into Import\_Inventory(perry\_part\_num, short\_description, long\_description, location\_in\_lab, quantity, purchase\_or\_rent, retail\_price, retail\_price\_promo,retail\_markup,jobber\_price,jobber\_markup,bulk\_price,bulk\_markup,cost\_to\_replace,cost\_avg,category\_code,category\_name,barcode1,barcode2,barcode3,barcode4,barcode5)
* values('$data[0]','$data[1]','$data[2]','$data[3]','$data[4]','$data[5]','$data[6]','$data[7]','$data[8]','$data[9]','$data[10]','$data[11]','$data[12]','$data[13]','$data[14]','$data[15]','$data[16]','$data[17]','$data[18]','$data[19]','$data[20]','$data[21]')"; //Good use of indexing $data here, seems much more efficient than assigning each field to a variable and then inserting the variable names. -Brett Marks
* $db->query($import) **or** **die**(mysqli\_error($db));
* }
* }
* fclose($handle);
* //call procedure that moves data from Import\_Inventory table to Inventory table
* $res=0;
* $db->query("CALL pr\_import\_inventory(@res)") **or** **die**(mysqli\_error($db));
* if($res!=0){
* print "Error with pr\_import\_inventory procedure";
* }
* else{
* print "Import done";
* }
* //print "Import done";
* }
* }
* ?>





2.6 Self-check on Best Practices for Security

1. Major assets being protected:
   1. Inventory
   2. Customer information
   3. Customer comments
   4. Search functions
2. All user passwords are confirmed to be encrypted in the database.
3. Spreadsheet Importation- This page will allow the admin/staff to upload a spreadsheet of parts, and input these parting into the inventory. To better protect from attacking SQL injections, we are creating and implementing MySQL procedures when inserting/updating information into the database. The stored procedure will insert the row only if a part number exists; and will also validate if the category exists, and it not, will create a new category.

SQL stored procedure code:  
***delete*** *from Import\_Inventory where perry\_part\_num is null;****insert*** *into Categories (category\_code, category\_name) select category\_code,max(category\_name) from Import\_Inventory where category\_code <> 0 and category\_code is not null and category\_name <> '' and category\_name is not null and category\_code not in (select category\_code from Categories) group by category\_code;****insert*** *into Categories ( category\_name) select distinct category\_name*

2.7 Self-check: Adherence to Original Non-functional Specs

1. To sign in to FAUOWLS, user shall connect to FAU Single Sign-on with corresponding credentials. [ON TRACK]
2. If a student orders an item for a specific course, the system  
   shall not need to check if student is currently registered for the course.  
   [DONE]
3. Expected load is no more than 300 simultaneous users. [ON TRACK]