Hochschule Fulda
University of Applied Sciences

Global Distributed Software Development

GDSD Team Project WS2020

Global Software Development

Project

FuldaMarkt

Milestone 4

Beta Launch, QA and Usability Testing and Final Commitment for Product Features (P1 list)

1st February 2021

Team 2 (local)

Syeda Tasneem Rumy, 1189706 (Team Lead, GitHub Master & Backend Lead),

syeda-tasneem.rumy@informatik.hs-fulda.de

Syed Sumair, 1310771 (Frontend Lead)

Ahmed Abdullah, 1250378 (Backend Developer)

Revision History

Revision	Status	Date
Version 0.1	Submitted	1st February 2021

CEO & CTO: Prof. Dr. Rainer Todtenhöfer, Fulda University, Germany

Contents

1. Product Summary 3
2. Usability Test Plan 4
3. QA Test Plan 6
4. Code Review 8
5. Self-check on Security Best Practices 12
6. Adherence to original Non-functional specs 13

1. Product Summary

- **Name of Product:** “Fulda Markt”
- **List of committed** functions
    - Register User
    - Login User
    - Market Search View
        - based on multiple categories filter
        - based on multiple order of sorting
    - Post a new item for sale
        - Input different fields for an item listing
        - Upload multiple images for an item
    - Item details
        - Item details with multiple images
        - Item details with related items(a/c to selected item category)
    - Messaging
        - Contact Seller via item
        - Contact Buyer via User’s dashboard messages section
    - User Wishlist
        - Add an item to wishlist
        - Remove an item from wishlist
    - Admin rights
        - Approve/decline any new or existing item from items list for market listing
        - Enable/disable new or existing user from users list
        - Change role of any user to make a normal user or give admin rights
    -
- **URL to access product:** https://ec2-34-226-123-197.compute-1.amazonaws.com/

2. Usability Test Plan

- **Test Objectives**:

The goal of this test is to measure effectiveness and efficiency of the search function, which will be the function used most frequently on the website. Additionally, in order to gauge user satisfaction and ensure that the search function meets the user's expectation of the product presented, a simple Lickert scale questionnaire will be provided and filled out by the tester.

- **Test Background and Setup:**

The test assumes that the server instance is running, and user has access to a web browser with internet access. The server uses the following technology stack:

- Server Host: Amazon AWS EC2 1vCPU 1 GB RAM

- Operating System: Ubuntu Server 20.04 LTS

- Server Database: MySQL v.14.14 Distribution 5.7.27

- Web Server: Apache v.2.4.29 (Ubuntu)

- Server-Side Language: PHP 7.2.24

The starting point for the task to be carried out by the tester is to open the website.

The URL for the website is : https://ec2-34-226-123-197.compute-1.amazonaws.com/

The intended users behind the test are users looking for a specific product (e.g. looking to buy new headphones). The intention behind the test is to measure user satisfaction after executing the task of searching for a desired item on the website.

- **Usability Task Description:**

The tester should follow the instructions below, then provide feedback to assess efficiency, effectiveness, and user satisfaction.

Instructions:

1. Open the website by copying the URL in the section "Test Background and Setup" into a browser.
2. Click on the "Market" item found in the user dashboard in the upper side of the website.
3. Type into the search bar what item to look for (e.g. iphone) and click on the magnifier icon.
4. Click on a desired item to view its details.

Feedback form to be filled out by the tester:

- Describe how would you measure the effectiveness of the search function in three lines or less:

The function was working as expected and should satisfy the expectations of our customer base.

- Describe how would you measure the efficiency of the search function in three lines or less:

The function presented was sufficiently efficient in providing fast results to the user's search criteria in a responsive way.

- Express your level of satisfaction with the presented user interface:

☐Highly Satisfied

☒Satisfied

☐Neutral

☐Dissatisfied

☐Highly dissatisfied

- Express your level of satisfaction with the responsiveness (i.e. load times, rendering):

☒Highly Satisfied

☐Satisfied

☐Neutral

☐Dissatisfied

☐Highly dissatisfied

- How likely are you to use the website again in the future?

☐Highly likely

☒Likely

☐Somewhat likely

☐Unlikely

☐Highly unlikely

3. QA Test Plan

Test Objectives

We'll run through the main functionality "Search" of the application "Fulda Markt", that needs to be tested as this function is used almost every time when the user uses this application. This test plan includes the trivial use cases performed by the users by setting different search parameters. This results the run rate and the pass rate of the search function.

HW and SW setup

A laptop and mobile with a stable internet connection and an internet browser are required to perform this test plan.

The starting point for the task to be carried out by the tester is to open the website.

The URL for the website is : https://ec2-34-226-123-197.compute-1.amazonaws.com/

Feature to be tested

Search functionality with different parameters

QA Test Plan

Test #	Title	Environment/Device	Description	Input	Expected/Correct Output	Test Result
1	Search by Category	Laptop/PC browser	Select specific category to see all listing of only respective category	Select "Electronics" by category dropdown	Page should show all the listing of Electronics category type	Pass
2	Search by Keyword	Laptop/PC browser	Enter an item keyword to search if it shows the list with relevant keyword	Enter "Chair" in the search text area	Page should show all the items which contains "Chair" in their title	Pass
3	Search by Sorting	Laptop/PC browser	Select any sort function to show listing in specific order	Select "Price Ascending" by sort dropdown	Page should show all the listing in order of price low to high	Pass
4	Search by Category	Mobile	Select specific category to see all listing of only respective category	Select "Services" by category dropdown	Page should show all the services	Pass
5	Search by Keyword	Mobile	Enter an item keyword to search if it shows the list with relevant keyword	Enter "iphone" in the search text area	Page should show all the items which contains "iphone" in their title	Pass
6	Search by Sorting	Mobile	Select any sort function to show listing in specific order	Select "Date Descending" by sort dropdown	Page should show all the listing in order of date new to old	Pass

Run rate is ratio between number test cases executed/total test cases of test specification. In our case, the test specification has total 6 test cases, and the tester has executed all 6 test cases, So the run rate is 6/6= 1.0(100%)

Pass rate is ratio between numbers test cases passed / test cases executed. In our case, there are 6 test cases executed, and the number of test cases that are passed are also 6, so the pass rate is 6/6= 1(100%)

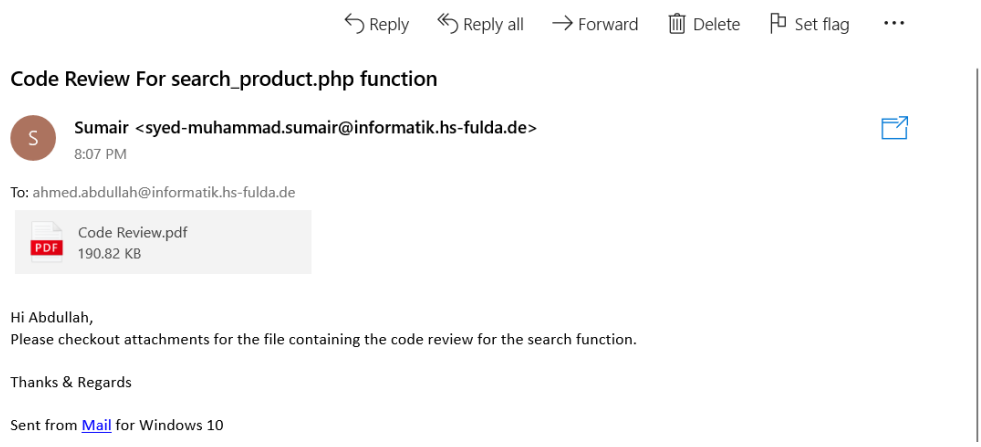
4. Code Review

Name of File reviewed: search_product.php

Author of File: Ahmed Abdullah

Function of File: Search functionality, Retrieve data from database based on query provided by frontend.

Email Screenshot:



Code Review:

```
<?php

        require 'project.php';

//This function should return an associative array of results.
        function get_product($dbi, $errors, $search_by_data)
        {
        $executed = false;//[Code Review by Syed Sumair] This variable should be named as
is_executed to indicate its usage properly
        $text_input = $search_by_data['text_input'];
        $category = $search_by_data['category'];
        $sort_by = $search_by_data['sort_by'];

        //Input validation to check if input is alphanumeric and less than 40 characters
        if(!empty($text_input)){
        if(!ctype_alnum(str_replace(' ','',$text_input)) || (strlen($text_input)) >= 40){
////[Code Review by Syed Sumair] such static values should be defined in a separate
constants file.
        return 1;
        }
        }

        //check database connection
        if(!$dbi ){
        die('Could not connect: ' . mysqli_error());
        $errors[] = "Search query failed: " . $ex->getMessage();
```

```
        return $executed;
        }

        mysqli_select_db($dbi, 'fuldamarkt_proddb');


        // check sorting to see whether to sort by ascending or descending order for
price and date sorting
        if(strpos($sort_by,"ascending") !== false){
        if(strpos($sort_by,"Price") !== false){
        $sort_by = "Price"; //formating
        }
        if(strpos($sort_by,"date") !== false){
        $sort_by = "date_inserted";
        }
        $sql_query = "SELECT * FROM MARKET_TABLE WHERE STATUS='available' AND TITLE LIKE
'%$text_input%' AND market_category LIKE '%$category%' ORDER BY $sort_by ASC;";
        }
        else if(strpos($sort_by,"descending") !== false){
        if(strpos($sort_by,"Price") !== false){
        $sort_by = "Price";
        }
        if(strpos($sort_by,"date") !== false){
        $sort_by = "date_inserted";
        }
        $sql_query = "SELECT * FROM MARKET_TABLE WHERE STATUS='available' AND TITLE LIKE
'%$text_input%' AND market_category LIKE '%$category%' ORDER BY $sort_by DESC;";
        }
        else{ //[Code Review by Syed Sumair] This statement could have been handled by
another [else if] condition so it doesn't execute all the time whenever above conditions
fails.
        $sql_query = "SELECT * FROM MARKET_TABLE WHERE STATUS='available' AND TITLE LIKE
'%$text_input%' AND market_category LIKE '%$category%' ORDER BY $sort_by ASC;";
        }



        try {
        $query_result = mysqli_query($dbi, $sql_query);
        // $executed = $query->execute(true); //[Code Review by Syed Sumair] There should
be no commented out code left. if necessary then mention its purpose or in which case
this needs to be uncommented
        } catch (\Exception $ex) {
        $errors[] = "Search query failed: " . $ex->getMessage();

        return $executed;
        }

        $query_result2 = $query_result->fetch_all(MYSQLI_ASSOC);
        return $query_result2;
        }
```

```
        $temp_name = 'market.twig';
        $title = 'Market Page';
        $body = 'Search for items';
        $errors = '';
        $message = '';
        $search_data = array();
        $search_data_pictures = array(); /*array containing multiple entries of
(directory:images), the directory is a string containg the location of a posts' images.
the ['images'] is a subarray containing names of pictures inside the directory */
//[Code Review by Syed Sumair] variable names should be simple and possibly short


        if ($request->getMethod() == "POST") {
        $search_by_data = array(
        'text_input' => trim($request->get('text_input')),
        'category' => trim($request->get('categories')),
        'sort_by' => trim($request->get('sort_by'))
        );



        $search_data = get_product($dbi, $errors, $search_by_data); //this is a 2d array
with index starting at 0 for first row


        if($search_data != 1){
        if(!empty($search_data)){ //check if there are results to the search query
        foreach($search_data as $val){//[Code Review by Syed Sumair] $val variable name
should be named by its type of object so it can be distinguished
        if(isset($val['picture'])){ //check if a post has a picture directory
        if(file_exists($_SERVER['DOCUMENT_ROOT'] . $val['picture'])){
        $files = array_diff( scandir( $_SERVER['DOCUMENT_ROOT'] . $val['picture']),
array(".", "..") ); //get list of image files found in the post's directory
        $temp_array = array('directory' => $val['picture'],'images' => $files);
        array_push($search_data_pictures, $temp_array);
        //[Code Review by Syed Sumair] for $files & $temp_array No object should exist
longer than necessary
        }
        }
        }
        unset($val);
        }
        else{
        $message = "No Data Found";
        }
        }
        else{
```

```
        $message = "Search input needs to be alphanumeric and less than 40 characters";
        }
        }

// necessary to forward to the twig template so that the "create post" button only
appears when a user is logged in
        $check_auth = $session->get('is_authenticated');



        /* Setting Template Variable, $page_data */
        if(!empty($search_data)){
        $page_data = ['title' => $title, 'body' => $body, 'errors'=> $errors, 'message'
=> $message, 'result' => $search_data,
        'search_data_pictures' => $search_data_pictures, 'Doc_root' =>
$_SERVER['DOCUMENT_ROOT'], 'check_auth' => $check_auth, 'session' => $session];
        }
        else{
        $page_data = ['title' => $title, 'body' => $body, 'errors'=> $errors, 'message'
=> $message, 'check_auth' => $check_auth, 'session' => $session];
        }
        try {
        echo $template->render($temp_name, $page_data);
        } catch (\Exception $ex) {
        $errors[] = "Template render error: " . $ex->getMessage();
        var_dump($errors);
        }
```

5. Self-check on Security Best Practices

Major Assets being protected

1. **Server Login credentials**
2. **Database**
3. **Deployed Application Code**

Major threats for each Asset above and how they are protected

1. **Server Login credentials**
    a. Private Key Compromise - Private key is only accessible to one administrator who manages the server.
    b. Unauthorized SSH Access - SSH credentials to server is only known to one administrator.
2. **Database**
    a. Privilege abuse – Database has only one user who administers the database deployed in server.
    b. SQL Injection – SQL injection has been mitigated by using data sanitizing filtering for all search, signup, login, product and message posting forms inputs.
    c. Exploitation of Vulnerable, Misconfigured Databases – Database can not be accessed from the web or without first logging in using the SSH credentials of the server.
    d. Denial of Service – Server is deployed using Amazon web services which prevents the Denial-of-Service vulnerability.
    e. Storage media exposure – No media stored in the server can be accessed from outside the application interface which is ensured by the Apache 2.0 configuration installed in the server.
3. **Deployed Application Code**
    a. Security Issues of Programming Language used – Usage of PHP 7 in server ensures better security enhancements, e.g. a filtered unserialize function or a set of functions cryptographically secure random numbers (random_bytes() and random_init()). Testing applications and finding hidden bugs becomes simpler due to scalar type and return type declarations.
    b. Directory traversal attack – It is prevented by avoiding passing of user-supplied input to filesystem APIs altogether.

Confirmation of User Information Security: All user supplied passwords are saved as encrypted values in the database.

Confirmation of Input data validation: FuldaMarkt codes validates,

1. Signup form – Emails for suffix, character limits and data type.
2. Login form – Validity of User Email and Password.
3. Product posting – Character limits for each form text inputs.
4. Search bar – Search inputs up to 40 alphanumeric characters.

6. Adherence to original Non-functional specs

No.	Non-Functional Requirement	Status	Notes (If any)
1	Application shall be developed, tested and deployed using tools and servers approved by Class CTO and as agreed in M0 (some may be provided in the class, some may be chosen by the student team but all tools and servers have to be approved by class CTO).	DONE	
2	Application shall be optimized for standard desktop/laptop browsers e.g. must render correctly on the two latest versions of two major browsers	DONE	
3	All or selected application functions must render well on mobile devices	ON TRACK	
4	Data shall be stored in the database on the team's deployment server.	DONE	
5	No more than 50 concurrent users shall be accessing the application at any time	DONE	
6	Privacy of users shall be protected and all privacy policies will be appropriately communicated to the users.	DONE	
7	The language used shall be English (no localization needed)	DONE	
8	Application shall be very easy to use and intuitive	DONE	
9	Application should follow established architecture patterns	DONE	
10	Application code and its repository shall be easy to inspect and maintain	DONE	
11	Google analytics shall be used (optional)	ISSUE	No, it has not been implemented. It was optional.
12	No e-mail clients shall be allowed. Interested users can only message to sellers via in-site messaging.	DONE	
13	Pay functionality, if any (e.g. paying for goods and services) shall not be implemented nor simulated in UI.	DONE	
14	Site security: basic best practices shall be applied (as covered in the class) for main data items	DONE	
15	Media formats shall be standard as used in the market today	DONE	
16	Modern SE processes and practices shall be used as specified in the class, including collaborative and continuous SW development	DONE	
17	The application UI (WWW and mobile) shall prominently display the following exact text on all pages "Fulda University Software Engineering Fall 2020. For Demonstration Only" at the top of the WWW page. (Important so as to not confuse this with a real application).	ON TRACK	