



PES UNIVERSITY

AUGUST – DECEMBER 2022 SEMESTER 5

SOFTWARE ENGINEERING LAB TASKS

Professor-in-charge: Dr. Jayashree R

Teaching Assistant: Aanchal Narendran (Sem VII)

We hope at this point in your semester you have achieved considerable progress in your Software Engineering Project. All of the tasks in this lab manual are geared towards improving and testing your project prior to your submissions.

TEAM NUMBER: 14

PES2UG20CS265: RAM SELVARAJ

PES2UG20CS320: KAMEEL

PES2UG20CS277: RIYA HURTIS

PES2UG20CS292: SAGARIKHA

Problem Statement – 1: Unit Testing

A unit is the smallest block of code that functions individually. The first level of testing is Unit testing and this problem statement is geared towards the same.

- Discuss with your teammates and demarcate units in your code base
 - Note: discuss why the code snippet you have chosen can be classified as a unit
- Develop test cases for both valid and invalid data
- Ideate how you could further modularize larger blocks of code into compact units with your teammates

```
<?php
include("connection/connect.php");
error_reporting(0);
session_start();
if(isset($_POST['submit']))
{
    $username = $_POST['username'];
    $password = $_POST['password'];

    if(!empty($_POST["submit"]))
    {
        $loginquery = "SELECT * FROM users WHERE username='$username' && password='".md5($password)."'"; //selecting matching records
        $result=mysqli_query($db, $loginquery); //executing
        $row=mysqli_fetch_array($result);

        if(is_array($row))
        {
            $_SESSION["user_id"] = $row['u_id'];
            header("refresh:1;url=index.php");
        }
        else
        {
            $message = "Invalid Username or Password!";
        }
    }
}
?>
```

Sr No.	Functional Test Cases	Type- Invalid/ Valid Test Case
1	Verify if a user will be able to login with a valid username and valid password.	Valid
2	Verify if a user cannot login with a valid username and an invalid password.	Invalid
3	Verify the login page for both, when the field is blank and the Submit button is clicked.	Invalid
4	Verify the messages for invalid login.	Valid
5	Verify if the data in the password field is either visible as asterisk or bullet signs.	Valid
6	Verify if a user is able to login with a new password only after he/she has changed the password.	Valid
7	Verify if the login page allows you to log in simultaneously with different credentials in a different browser.	Valid
8	Verify if the 'Enter' key of the keyboard is working correctly on the login page.	Valid

Sr No.	Security test cases	Type- Invalid/ Valid Test Case
		Valid
1	Verify if a user cannot enter the characters more than the specified range in each field (Username and Password).	
2	Verify if a user cannot enter the characters more than the specified range in each field (Username and Password).	Invalid
3	Verify the login page of the browser that it should not allow you to enter into the system once you log out.	Valid
4	Verify if a user should not be allowed to log in with different credentials from the same browser at the same time.	Valid
5	Verify if a user should be able to login with the same credentials in different browsers at the same time.	Invalid

Problem Statement – 2

Dynamic testing involves execution of your code to analyse errors found during execution. Some common techniques are Boundary Value Analysis and Mutation Testing.

Problem Statement –2.a:Boundary Value Analysis

When it comes to finding errors in your code base, they are often found at locations where a condition is being tested. Due to this, developers often use Boundary Value tests to reduce defect density.

- How would you define a boundary test?

Note: Simple relational conditions are a basic example

- Build your boundary test cases and execute them

1. How would you define a boundary test?

Boundary testing is a black-box testing technique that software developers often use to check the errors at the boundaries or extreme ends of a given input domain. An input domain comprises all the possible inputs available in a software program. Software developers utilize black-box testing to analyze the behavior of a software program and examine its functionality. The extreme ends in boundary testing might include start to end, lower to upper or minimum to maximum. Instead of focusing only on the center of the data, boundary testing helps detect errors occurring at the boundary values of valid or invalid partitions.

2. The boundary testing will be applied to create an Account
 - a) The user's password count should not exceed a value of 222
 - b) The minimum value of password count should be 6
 - c) The maximum value of password count should be 222

Problem Statement –2.b:Mutation Testing

- Using your isolated units from the first problem statement, ideate with your team mates on how to mutate the code
- Develop at least 3 mutants of the functioning code and test all 4 code bases using the test case from the first problem statement

Mutation testing is a fault-based testing technique where variations of a software program are subjected to the test dataset. This is done to determine the effectiveness of the test set in isolating the deviations.

1)Original:

```
$item_total = 0;

foreach ($_SESSION["cart_item"] as $item)
{
    ?>

    <div class="title-row">
    <?php echo $item["title"]; ?><a href="dishes.php?res_id=<?php echo $_GET['res_id']; ?>&action=remove&id=<?php echo $item["d_id"]; ?>" >
    <i class="fa fa-trash pull-right"></i></a>
    </div>

    <div class="form-group row no-gutter">
        <div class="col-xs-8">
            <input type="text" class="form-control b-r-0" value=<?php echo "$".$item["price"]; ?> readonly id="exampleSelect1">
        </div>
        <div class="col-xs-4">
            <input class="form-control" type="text" readonly value='<?php echo $item["quantity"]; ?>' id="example-number-input"> </div>
    </div>

    <?php
    $item_total += ($item["price"]*$item["quantity"]);
}
}
```

Mutant 1:

```
$item_total = 0;

foreach ($_SESSION["cart_item"] as $item)
{
    ?>

    <div class="title-row">
    <?php echo $item["title"]; ?><a href="dishes.php?res_id=<?php echo $_GET['res_id']; ?>&action=remove&id=<?php echo $item["d_id"]; ?>" >
    <i class="fa fa-trash pull-right"></i></a>
    </div>

    <div class="form-group row no-gutter">
        <div class="col-xs-8">
            <input type="text" class="form-control b-r-0" value=<?php echo "$".$item["price"]; ?> readonly id="exampleSelect1">
        </div>
        <div class="col-xs-4">
            <input class="form-control" type="text" readonly value='<?php echo $item["quantity"]; ?>' id="example-number-input"> </div>
    </div>

    <?php
    $item_total = ($item["price"]*$item["quantity"]);
}
}
```

Adding item to the Cart and checking the price

Your Cart

Yorkshire Lamb Patties

\$14.00

2

Lobster Thermidor

\$36.00

1

TOTAL

\$36

Free Delivery!

Checkout

As a new item is added the total value does not get added to the original value but gets replaced

2) Original:

```

<?php
include("connection/connect.php");
error_reporting(0);
session_start();
if(isset($_POST['submit']))
{
    $username = $_POST['username'];
    $password = $_POST['password'];

    if(!empty($_POST["submit"]))
    {
        $loginquery = "SELECT * FROM users WHERE username='$username' && password='".md5($password)."'"; //selecting matching records
        $result=mysqli_query($db, $loginquery); //executing
        $row=mysqli_fetch_array($result);

        if(is_array($row))
        {
            $_SESSION["user_id"] = $row['u_id'];
            header("refresh:1;url=index.php");
        }
        else
        {
            $message = "Invalid Username or Password!";
        }
    }
}
}
?>

```

Mutant 2:

```

<?php
include("connection/connect.php");
error_reporting(0);
session_start();
if(isset($_POST['submit']))
{
    $username = $_POST['username'];
    $password = $_POST['password'];

    if(empty($_POST["submit"]))
    {
        $loginquery = "SELECT * FROM users WHERE username='$username' && password='".md5($password)."'"; //selecting matching records
        $result=mysqli_query($db, $loginquery); //executing
        $row=mysqli_fetch_array($result);

        if(is_array($row))
        {
            $_SESSION["user_id"] = $row['u_id'];
            header("refresh:1;url=index.php");
        }
        else
        {
            $message = "Invalid Username or Password!";
        }
    }
}
}
?>

```

As we removed the ! in the if statement. The page does not fetch any user with given username and password

Login to your account

Username

Password

Login

Not registered? [Create an account](#)

Original:

```
if($_POST['password'] != $_POST['cpassword']){
    echo "<script>alert('Password not match');</script>";
}
elseif(strlen($_POST['password']) < 6)
{
    echo "<script>alert('Password Must be >=6');</script>";
}
elseif(strlen($_POST['phone']) < 10)
{
    echo "<script>alert('Invalid phone number!');</script>";
}
```

Mutant 3:

```

if($_POST['password'] != $_POST['cpassword']){
    echo "<script>alert('Password not match');</script>";
}
elseif(strlen($_POST['password']) < 5)
{
    echo "<script>alert('Password Must be >=6');</script>";
}

```

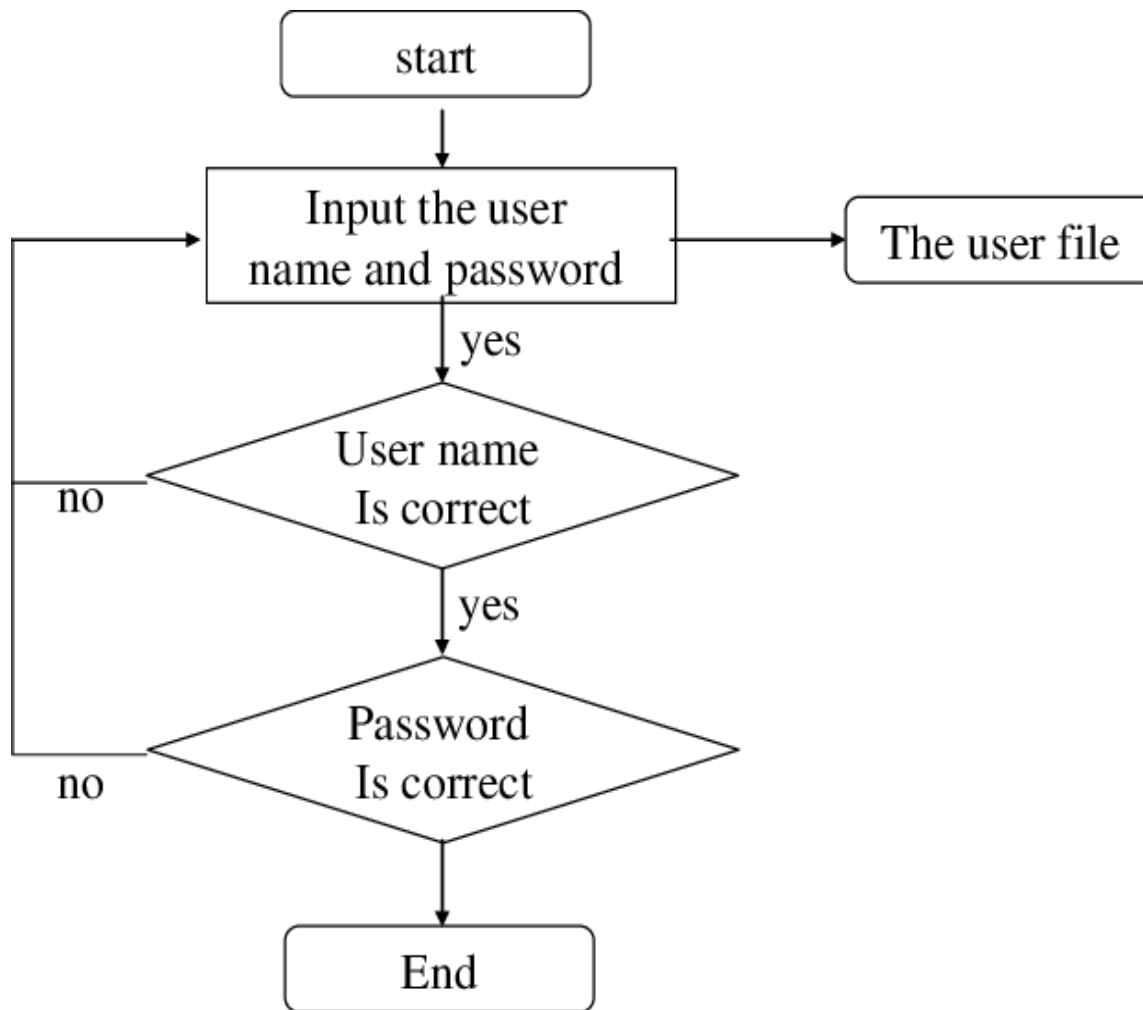
The password should exceed a count of 5



Problem Statement – 3: Static Testing

Static testing involves validating your code without any execution. Under this problem statement, you will be expected to analyse and calculate the cyclomatic complexity of your code.

- Using the unit you selected in the first problem statement as an example, develop the control flow graph of your problem statement.
- Using the Control flow graph, calculate the cyclomatic complexity of your code.
- Using the cyclomatic complexity as an indicator, Ideate and code your unit again to reduce complexity



SOLUTION:

$$M = E - N + 2P$$

where,

M = cyclomatic complexity

E = the number of edges in the control flow graph
N = the number of nodes in the control flow graph
P = the number of connected components

$$E = 7$$

$$N = 6$$

$$P = 1$$

$$M = 7 - 6 + 2(1)$$

$$M = 3$$

Therefore, the cyclomatic complexity of our above code is 3.

Problem Statement – 4: Acceptance Testing

Assume your neighbouring team is the client for your code. Give them an idea of what your product is and the software requirements for the product.

- Exchange your code base and test each other's projects to see if it meets user requirements
- If you identify a bug in the project you are testing, inform the opposing team of the bug
- As a team, based in clients experience, ideate modifications to the existing project that could improve client experience.

Solution:

Since the onset of the pandemic, customers are preferring minimal contact services. Due to this, the restaurants have been actively making an effort to prevent any contact through the physical menus, as it is tedious to sanitize them after every customer uses it. With this in mind, we chose to implement online Food Delivery System that enables the customer to order with minimal contact. This project will have a GUI platform which will serve as the front-end framework. The front-end will contain the complete menu of the restaurant, that will allow the customers to browse through the menu and place their order. The backend framework will be handled using python programming language.

The online food delivery platform enables users to choose from a wide variety of cuisines. It has an Admin Dash which provides information about the pending orders, cancelled orders, on-delivery orders, update orders etc. It also has a User Dash which shows details about present and past orders, re-order etc. It also shows recommended food items

Software Requirements for the product:

Functional Requirements:

The system will:

- Allow Customers to scroll through the menu and select the dishes he/she wants.
- Show the customer the total cost of each category of the menu.
- Allow the Customer to reset the order.
- Give the subtotal of the orders.
- Calculate the total cost of the order along with the taxes.
- Allow the user to save the order.
- Print the order receipt.
- Allow Customers to request for a bill.
- Send the customer the final bill.
- Allow customers to save the bill.

REQ-1: The system must notify the hotel that a customer has requested a bill.

REQ-2: The system must notify the hotel that a customer has paid a bill.

Nonfunctional Requirements:

Performance Requirements:

As the main purpose of this system is to make ordering food and managing the restaurant easier, the system must be interactive, easy to use with clear instructions and the delays should be as low as possible. The main action being scrolling through the menu and

selecting and adding the orders there should be no more than 2 seconds before the items get added in. After adding the items into the cart, the subtotal and the final total need to be calculated without any kind of delay. The bill should be displayed with proper calculation of the amount. The receipt should be ready to be sent to the user. Deletion / Reset of the order should be done without much delay.

Safety Requirements:

The software is completely environmentally friendly and does not cause any safety violations. The menu will have a flexible font with a suitable color in the background and GUI components so as to not strain the eyes while watching the menu.

Security Requirements:

The system doesn't take in important customer details as it mainly involves ordering of the food within the restaurant. The bill is downloaded through the device. The receipt is provided to the customer.

User Requirements:

1. Customer Interface

The customer interface will contain one screen. Along with a calculator for bill verification and a bill generation button. The various buttons will provide the customer the option to add items or completely reset the order. There are buttons to save/print and send the bill.

1.1. Place Order

In this screen, the system shows a list of cards (UI Elements) of dishes. Each dish will have an image, its price per serving. The order will be sent to the kitchen for preparation. The customer must press send, to send the order to the kitchen.

1.2. Billing

There will be a button to request a bill. The calculator will allow the customer to cross verify the amount. The generated bill can be saved or printed. This is done to save paper and make this whole experience more contactless.

As a client, the product meets all the user requirements. We have done user acceptance testing. User acceptance testing is used to determine whether the product is working for the user correctly. Specific requirements which are quite often used by the customers are primarily picked for the testing purpose. This is also termed as End-User Testing. The product is tested against the SRS provided by the product manager.

Bugs found:

- Input validation
- Encryption of password

As a team, based on clients experience we would suggest a few modifications in the product.

- The product can have an input validation
- The product can have a use of encrypted password.

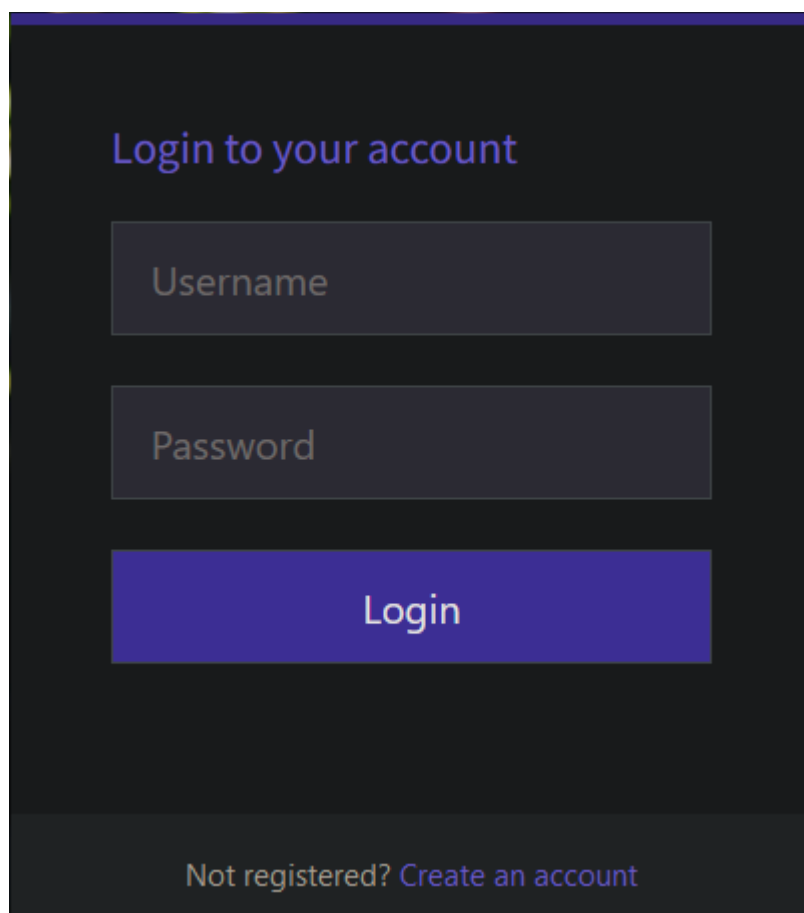
Problem Statement – 5:

Maintenance Activities Once a product is completed, it is handed off to a service-based company to ensure all maintenance activities are performed without the added expenditure of skilled developers. However, a few tasks are performed by the maintenance team to gauge the product better. In this problem statement, you will be asked to experiment with your code.

- Exchange code bases with your neighboring teams and reverse engineer a block of code in order to understand it's functionality

- After understanding the code block, Re-Engineer the code. Ideate how to refactor the code and the portion of the code base you would have to change. Discuss how the new changes would impact the time and space complexity of the project during execution
- After Reverse Engineering and Re-Engineering, the code, perform acceptance testing between the teams.

Software Re-Engineering is the examination and alteration of a system to reconstitute it in a new form. The principle of Re-Engineering when applied to the software development process is called software re-engineering. It affects positively software cost, quality, service to the customer, and speed of delivery. In Software Re-engineering, we are improving the software to make it more efficient and effective



Login to your account

Username

Password

Login

Not registered? [Create an account](#)

A user can be allowed to enter his username and password to login And also create an account if the account doesn't exist

User-Name

First Name

Last Name

Email Address

Phone number

Password

Confirm password

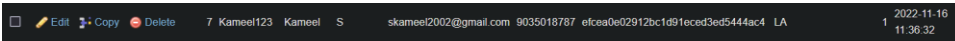
Delivery Address

Register

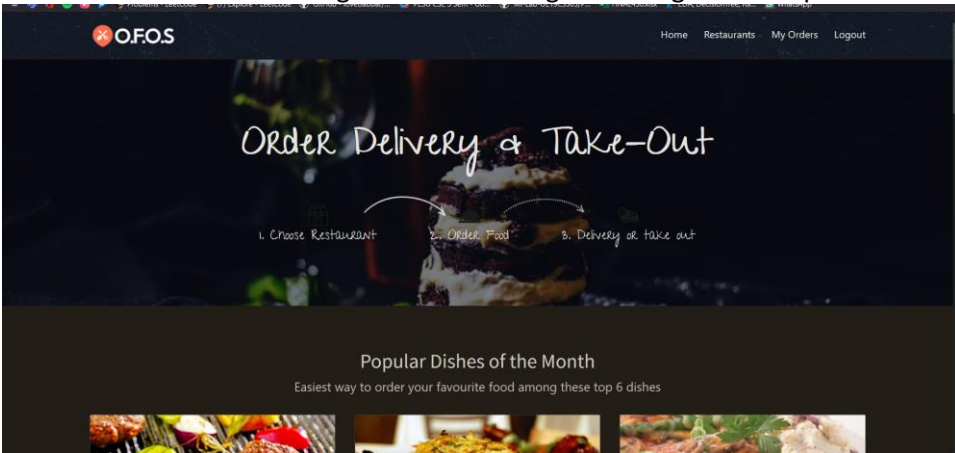
In Create account the user is supposed to enter all the details required to create an account. The portion of the code that we would like to change:

```
include("connection/connect.php");
if(isset($_POST['submit']))
{
    if(empty($_POST['firstname']) ||
        empty($_POST['lastname']) ||
        empty($_POST['email']) ||
        empty($_POST['phone']) ||
        empty($_POST['password']) ||
        empty($_POST['cpassword']) ||
        empty($_POST['cpassword']))
    {
        $message = "All fields must be Required!";
    }
    else
    {
        $check_username= mysqli_query($db, "SELECT username FROM users where username = '".$_POST['username']."' ");
        $check_email = mysqli_query($db, "SELECT email FROM users where email = '".$_POST['email']."' ");
```

All the options can be provided to the user after a Creating Account. The time complexity remains the same but the space complexity increases due to the introduction of a new feature to improve user experience.



We would like the user to login and then Navigate through all the options available on the website



As we can see there is no login page. Now the user can only see the options after logging in

As a client, the product meets all the user requirements. We have done user acceptance testing. User acceptance testing is used to determine whether the product is working for the user correctly. Specific requirements which are quite often used by the customers are primarily picked for the testing purpose. This is also termed as End-User Testing. The product is tested against the SRS provided by the product manager. No bug is found in the new product. The new product will have the following features discussed during the acceptance testing:

- Encrypted password
- Input validation