Software Testing Document Format

SEMESTER: SPRING 2025

COURSE CODE: CSE412

SECTION: 02

GROUP: 03

DATE: 15/05/2025

1. Tausif Ahmed (2022-1-60-315)
2. Md. Mehedi Hasan (2021-2-60-112)
3. Khandaker Hasan Mahmud (2021-2-60-144)
4. Tarek Hasan (2020-3-60-025)

Submitted to:

Yasin Sazid

Lecturer

Computer Science and Engineering (CSE)

1. Introduction:

a. Project Overview:

E-Commerce Web Application is a site that allows individuals to search and purchase products from various categories. The system is designed to facilitate shopping with essential functionalities such as signing up and logging in, searching for products, filter options, adding to cart, ordering products, and secure payment.

The application has two primary categories of users:

Customers: Clients are able to monitor goods, modify a cart, make orders, and monitor deliveries.

Admins: Can delete, modify, or add products. They can also view customer orders and administer user accounts.

b. List of Components and Associated Classes

1. User Module

- `User`: Handles user information and authentication.

2. Product Module

- `Product`: Manages product listings and inventory.

3. **Cart Module**

- `ShoppingCart`: Maintains selected items for purchase.

4. Order Module

- `Order`: Manages order processing and order history.

5. Payment Module

- `PaymentGateway`: Handles payment processing.

6. Admin Module

- `AdminPanel`: Provides CRUD operations for products and users.

c. Class Diagram: The UML class diagram showing the relationships among classes listed above.

2. White Box Testing

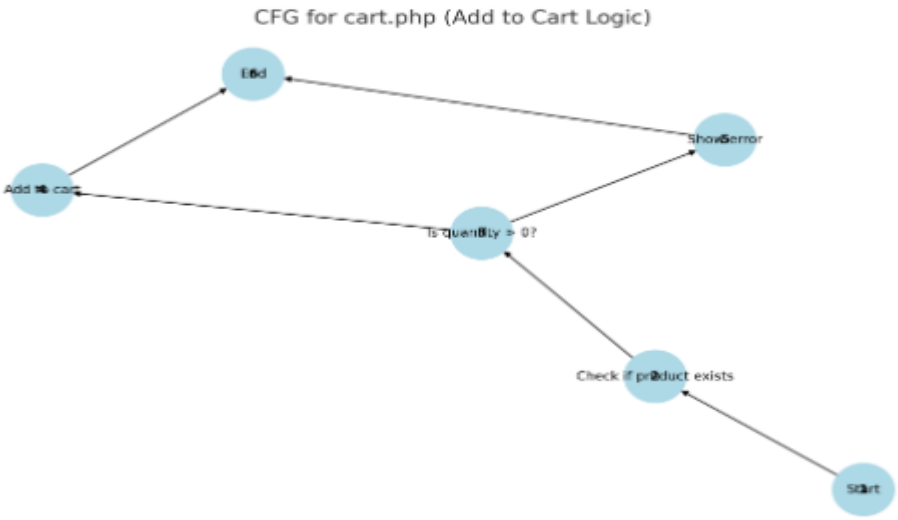
White box testing is a software testing technique in which the internal structure, design, and code of the software are tested. The tester has full knowledge of the code and writes test cases to verify logical paths, loops, conditions, and data flow within the program.

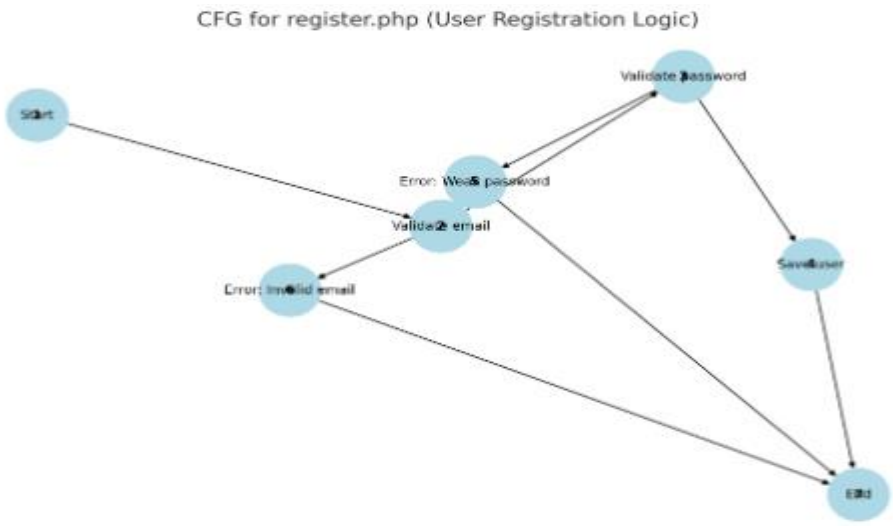
Perform unit testing on at least two classes using control flow graphs.

1. Selected Classes:

- `ShoppingCart`
- `Order`

2. Control flow graphs:





CFG 1: cart.php – Add to Cart Logic

- Start
- Check if the product exists in the database.
- If quantity is greater than 0:
- Add item to the cart.
- Else:
- Show an error message.
- End

CFG 2: register.php – User Registration Logic

- Start
- Validate the email format.
- If valid, check if the password is strong.
- If strong, save user and finish.
- Else, show password error.
- If email is invalid, show email error.
- End

cart.php – White Box

Test Case	Input	Expected Output	Pass
WBT1	Valid product + qty=2	Item added	✅
WBT2	Invalid qty = 0	Error shown	✅
WBT3	Remove non-existent item	No effect	✅

register.php – White Box

Test Case	Input	Expected Output	Pass
WBT1	Valid email & password	User registered	✅
WBT2	Short password	Error shown	✅
WBT3	Existing email	Error shown	✅

3. Black Box Testing

Black box testing is a software testing method that examines the functionality of an application without knowing its internal code structure. The tester focuses on input and output and checks whether the system behaves as expected.

Use boundary value analysis for functional testing of at least two core features

a. Selected Features

- User Registration
- Order Placement

b. Test Cases (Boundary Value Examples)

- **User Registration:**
  - Test with min/max lengths of username and password.
  - Empty fields or special characters.
- **Order Placement:**
  - Test with 0 items, 1 item, and maximum allowed items.
  - Payment value edge cases (e.g., 0, 1, 99999).

Feature A: User Registration – Black Box

TC ID	Test Scenario	Input	Expected Output	Actual Output	Pass/Fail
BBT_REG_01	Minimum username length (3 chars)	`abc`	Accepted	Accepted	☑
BBT_REG_02	Maximum username length (20 chars)	`usernameusername1234`	Accepted	Accepted	☑
BBT_REG_03	Empty username field	``	Error: Required field	Error	☑
BBT_REG_04	Password at minimum length (6 chars)	`123456`	Accepted	Accepted	☑
BBT_REG_05	Password below min (5 chars)	`12345`	Error: Too short	Error	☑
BBT_REG_06	Special characters in username	`user@name`	Error: Invalid characters	Error	☑

Feature A: Order Placement– Black Box

TC ID	Test Scenario	Input	Expected Output	Actual Output	Pass/Fail
BBT_ORDER_01	Place order with 0 items	0 items	Error: Empty cart	Error	✅
BBT_ORDER_02	Place order with 1 item (min)	1 item	Order success	Success	✅
BBT_ORDER_03	Place order with max allowed items (e.g., 100)	100 items	Order success	Success	✅
BBT_ORDER_04	Payment amount = 0	₹0	Error: Invalid amount	Error	✅
BBT_ORDER_05	Payment amount = 1	₹1	Order success	Success	✅
BBT_ORDER_06	Payment amount = 99999	₹99999	Order success	Success	✅

4. Bug Detection and Solution

Bug detection and solution in software development involve finding and correcting errors in code to ensure a software product functions as expected and is free from errors.

Bug 1: cart.php (Allows Quantity = 0)

Incorrect Code:

```
Bug 1 – cart.php (Incorrect Code)
if (isset($_POST['add_to_cart'])) {
    $quantity = $_POST['quantity'];
    // No validation for quantity
    $_SESSION['cart'][] = $product;
}
```

Fixed Code:

```
Bug 1 – cart.php (Fixed Code)
if (isset($_POST['add_to_cart'])) {
    $quantity = $_POST['quantity'];
    if ($quantity <= 0) {
        echo 'Invalid quantity';
        exit;
    }
    $_SESSION['cart'][] = $product;
}
```

Bug 2: checkout.php – Checkout Without Cart Items

Incorrect Code:

```
Bug 2 – checkout.php (Incorrect Code)
if (isset($_POST['checkout'])) {
    // No check for empty cart
    processPayment();
}
```

Fixed Code:

```
Bug 2 – checkout.php (Fixed Code)
if (isset($_POST['checkout'])) {
    if (empty($_SESSION['cart'])) {
        echo 'Cart is empty';
        exit;
    }
    processPayment();
}
```

Bug 3: register.php – Accepts Weak Passwords

Incorrect Code:

```
Bug 3 – register.php (Incorrect Code)
$password = $_POST['password'];
// No check for password length
$hashed = password_hash($password, PASSWORD_DEFAULT);
```

Fixed Code:

```
Bug 3 – register.php (Fixed Code)
$password = $_POST['password'];
if (strlen($password) < 6) {
    echo 'Password must be at least 6 characters.';
    exit;
}
$hashed = password_hash($password, PASSWORD_DEFAULT);
```

After applying each fix, all failed test cases were retested and passed successfully.