Lab 9: Final Test Sprint Plan

Project: ProjectWebsiteV9
Team: Josh and Colm
Date: 03/04/25

---

1. Unit Testing

Classes to Test (Located in `/classes`):

- `user.php` – Handles user registration, login, session management.
- `admin.php` – Manages product listings, admin privileges, backend actions.

Testing Tool:

- **PHPUnit** will be used for automated testing.

Test Plan:

- Ensure all public methods in both classes are covered.
- Add/Update unit tests for:
    - User login/logout
    - Registration and input validation
    - Admin CRUD actions (Create, Update, Delete products)
- All unit tests will reside or be added in the `/tests/` folder.

---

2. UI Testing

Current Implemented Features:

- Main navigation (index.php, about.php, contact.php, etc.)
- Product listing (product.php), Search (search.php)
- Checkout process (checkout.php, shipping.php)

Testing Focus:

- Minimum clicks from landing to purchase (Product → Cart → Checkout)
- Basic input validation feedback
- Use of forms and recoverability (e.g. navigating back, resubmitting forms)

To Be Added:

- Enhanced error messaging for invalid forms
- ARIA roles / accessibility features

- Confirmation messages after actions (e.g. successful checkout)

---

3. Requirements Testing

(Reference: `Assignment2-Requirement Documentation.pdf`)
Below is a checklist of core requirements:

Requirement	Status
User Registration/Login	✅ Completed
Product Browsing	✅ Completed
Product Search	✅ Completed
Add to Cart	✅ Completed
Checkout Process	✅ Completed
Order Summary	🔲 In Progress
Admin Product Management	✅ Completed
Contact Form	✅ Completed
Email Confirmation	🔲 Not Implemented

Additional functional or non-functional requirements should be listed similarly based on the PDF.

---

4. Basis Path Testing (White-box)

Sample Path Selection:
Function: `processCheckout()`
Decision Points:

- If user is logged in
- If cart is not empty
- If address and payment are valid
- Else return error

Cyclomatic Complexity Calculation:

- 4 decision nodes
- Formula: `V(G) = E - N + 2`
- Example: V(G) = 6 - 4 + 2 = **4**
- → **4 independent paths** to test.

Paths:

1. User logged in → Cart valid → Payment valid → Success
2. User not logged in → Error
3. User logged in → Empty cart → Error

4. User logged in → Cart valid → Payment invalid → Error

---

5. Equivalence Partitioning

Input Areas & Partitions:

Input Type	Valid	Invalid	Edge Cases
Search	“Shoes”	“#@$%”	Empty string
Email	test@email.com	test@.com	email@domain
Card Number	16-digit	15/17-digit	Empty
Quantity	1–100	0, -1, 1000	1

Plan: Test one input from each partition per field.

---

6. Validation Testing

Custom Validation Rules (not HTML5):

Field	Validation
Email	Must match `^[\w-.]+@([\w-]+\.)+[\w-]{2,4}$`
Phone	Digits only, 10–15 characters
Password	Minimum 8 characters, 1 uppercase, 1 number
Confirm Password	Must match Password
Quantity	Must be > 0

Each form input will have manual server-side checks in addition to client-side behavior.