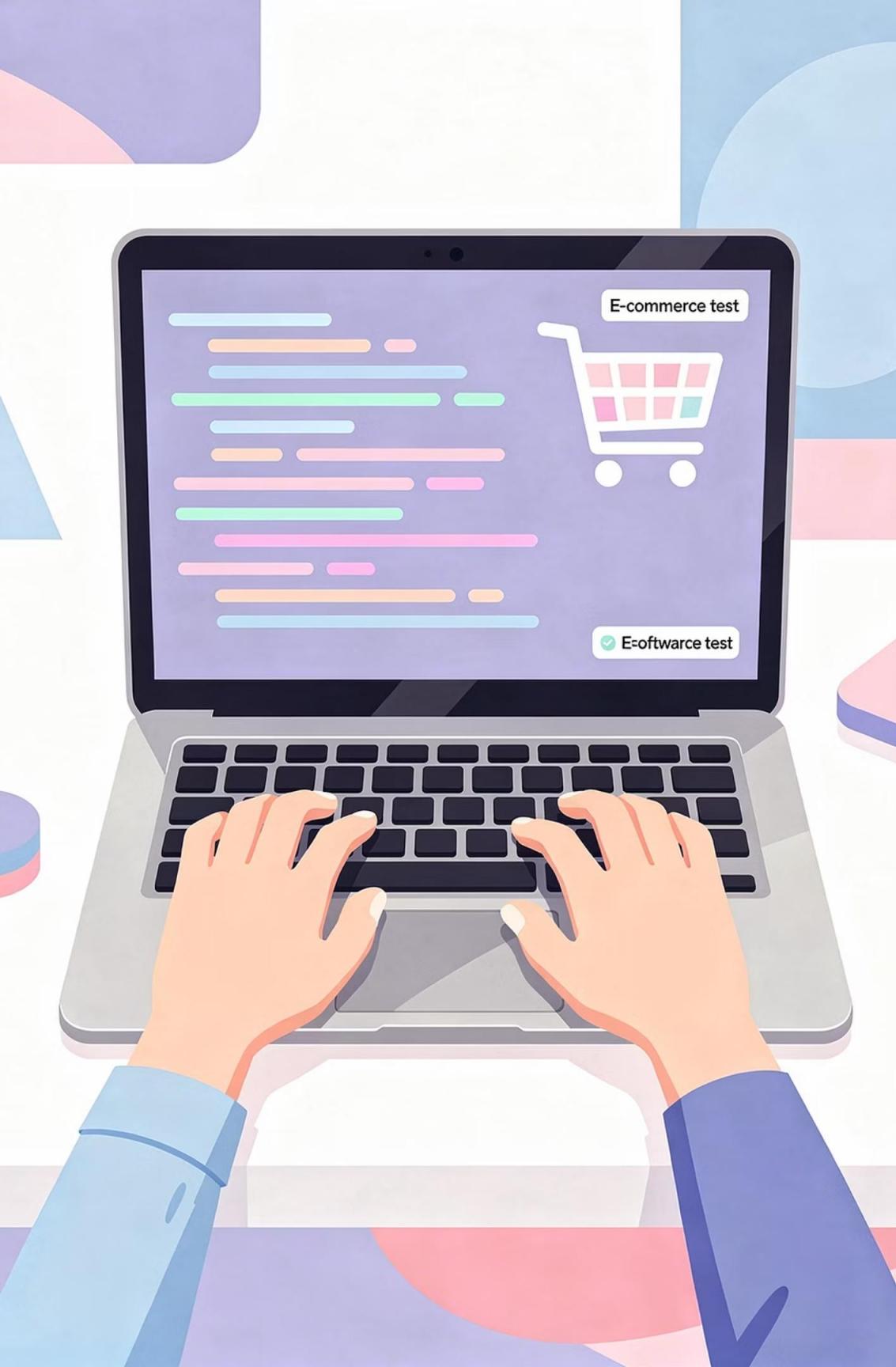


Graduation Project – Software Testing

OpenCart Demo – E-commerce Platform



Team 4 – Software Testing Track

- Nervana Isaac (Team Leader)
- Amira Samy
- Hadeer Mahmoud
- Youssef Khaled
- Yasser Mehair

Agenda

01

Project Overview

02

Project Objectives

03

Test Scope & Out of Scope

04

Testing Strategy

05

Tools Used

06

Manual Testing Overview

07

Database Testing Overview

08

API Testing Overview

09

Automation Testing Overview

10

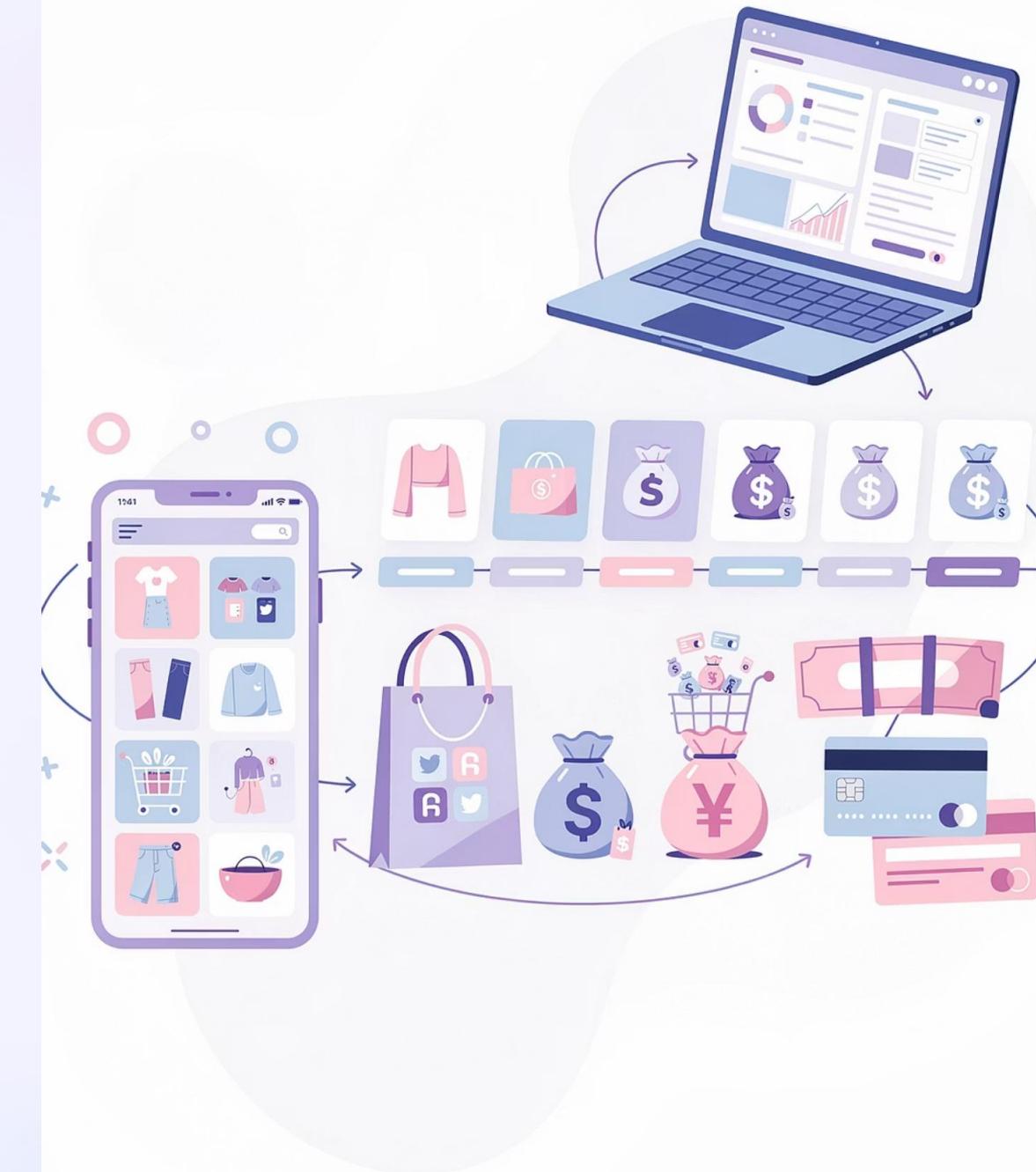
Test Results Summary

11

Final Conclusion

Project Overview

- OpenCart Demo simulates a full online shopping system for users and admins.
- End-to-end testing covered both the customer interface and the Admin Panel.
- API and automation tests were done on **Automation Exercise** due to OpenCart's limitations.
- Manual and database tests were performed on **OpenCart**.



Project Objectives

Ensure full functional coverage of critical workflows

Validate UI functionality and customer journey

Verify database consistency after key actions

Test API behavior and responses

Build an automation suite for selected scenarios

Document and log defects

Deliver complete documentation and a final test summary

TEST SCOPE

In-Scope:



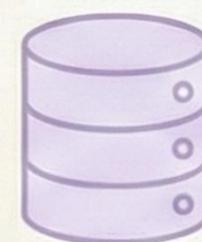
Customer Frontend



Admin Area



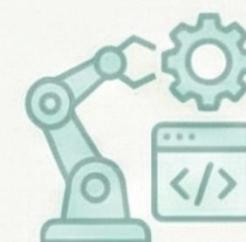
Manual Functional Testing



Database Testing



API Testing
(via Automation Exercise)



Automation Testing
(via Automation Exercise)

Tools Used

Java

Selenium WebDriver

IntelliJ IDEA

TestNG

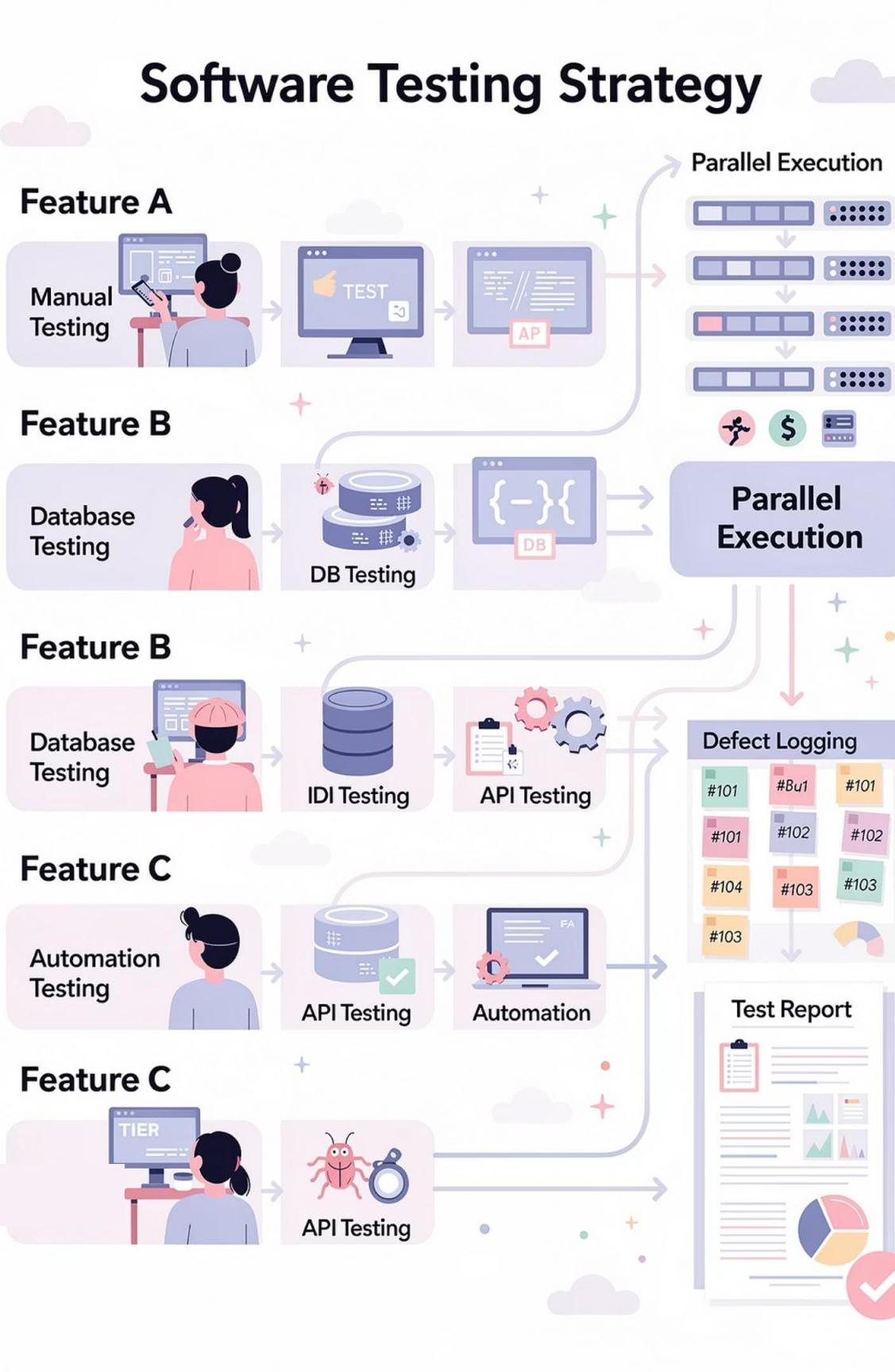
Postman

MySQL

Excel / Google Sheets

JIRA (Defect Logging)

Software Testing Strategy



Testing Strategy

We divided the project into features, and each team member was responsible for testing their assigned features using:

- Manual Testing
- Database Testing
- API Testing
- Automation Testing

Additional strategy points:

- Parallel execution
- Severity-based defect logging
- Final consolidation into reports

Testing Strategy

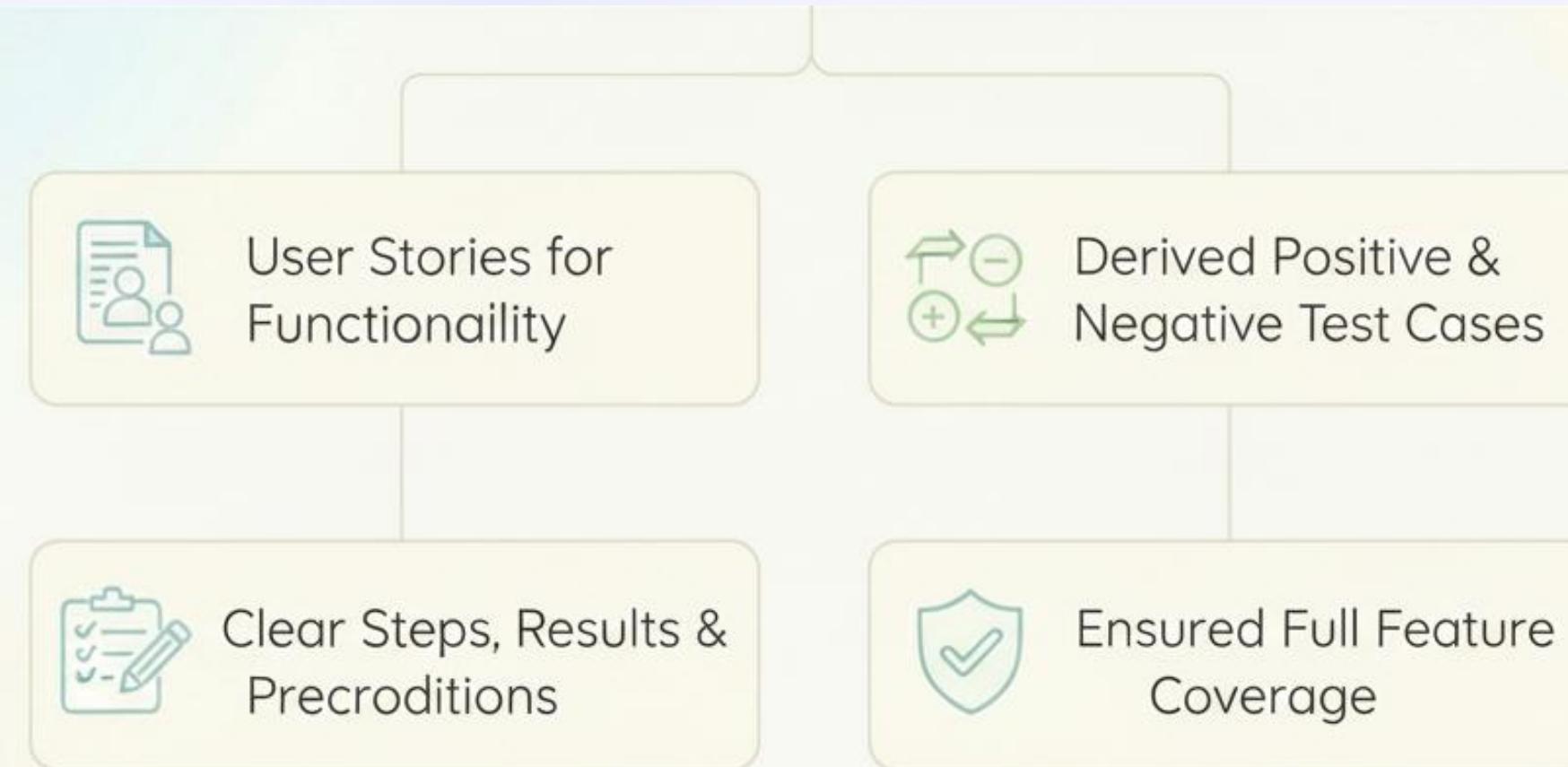
OpenCart Online Shopping (Customer Page) Features:

- User Registration
- User Login
- Account Management
- Product Browsing
- Cart Management
- Wishlist
- Checkout Process

Admin & Management Features:

- Catalog Management
- Admin Reporting
- Storefront Design

Manual Testing Approach



customer Login



Description

Normal text | **B** *I* ... | A | := | + | Write

As a registered user,
I want to log in using my username and password or reset my password if I forget it,
So that I can securely access my account and recover it when needed.

Acceptance Criteria:

1. User can log in successfully with a valid username and valid password.
2. User sees an appropriate error message when using invalid username and/or password.
3. User can use the "Forgot Password" feature to request a password reset via registered email.
4. System sends a password reset link to the registered email.
5. User can set a new password and log in successfully.
6. Appropriate error messages are displayed if the email is not registered or invalid.

Save

Cancel

Linked work items



blocks

GD-81 login with valid user name and valid password	PASSED		
---	--------	--	--

is blocked by

GD-97 login with valid user name and invalid password	PASSED		
GD-88 login with invalid user name and valid password	PASSED		

relates to

GD-114 Login with invalid username and invalid password	PASSED		
GD-115 Login with empty username and/or password	PASSED		
GD-116 Forgot Password functionality	PASSED		
GD-117 Verify password reset via email link	FAILED		

Manual Testing Execution

Execution Summary:

- **Total Test Cases Designed:** 102
- **Executed:** 95

Execution covered:

- **Customer Page Features** (Registration, Login, Account Management, Product Browsing, Cart, Wishlist, Checkout)
- **Admin & Management Features** (Catalog, Admin Reporting, Storefront Design)

Defect Management:

- Failed test cases were logged as defects with evidence
- Cross-review ensured consistent execution results across the team

Login fails after password reset — error page displayed



Description

Steps to Reproduce

1. Open the **Customer Login** page.
2. Click “Forgot Password?”.
3. Enter a registered email address and submit the reset request.
4. Open the received password reset email and click the password reset link.
5. Complete the **create new password** flow (enter and submit a new password).
6. After being redirected to the **Login** page, attempt to log in using the newly created password.

Expected Result

- After clicking the reset link the user is taken to a valid password-creation page.
- The user can create a new password, then log in successfully with that new password and access their account.

Actual Result

- The user is able to create a new password and is redirected to the Login page.
- However, when attempting to log in with the newly created password, an error page appears and the account cannot be accessed.

Attachments 1



What happened?

The owner of this website (www.openvert.com)
has banned you temporarily from accessing
this website.
Please see
<https://devopsers.cloudflare.com/support/troubleshooting#status-when-cloudflare-has-temporarily-banned>

WhatsApp Ima... 61f.jpg

27 Nov 2025, 11:38 AM

Linked work items



relates to

GD-113 customer Login

TO DO ▾



MA

Verify password reset via email link



Test Data

Registered Email: nervanaisaac1@gmail.com

Pre-condition

User is registered with valid email; Login page accessible

Description

1. Open Login page
2. Click "Forgotten Password?" link
3. Enter registered email
4. Submit request
5. Open email and click the password reset link
6. Enter new password and confirm
7. Submit new password
8. Try to log in using new password

Expected Result

- System displays the message: "Password reset successfully"
- User should be able to log in immediately using the new password without any errors

Actual Result

- System displays the message: "Password reset successfully"
- But when trying to log in with the new password, the user is redirected to an error page instead of logging in

Linked work items



relates to

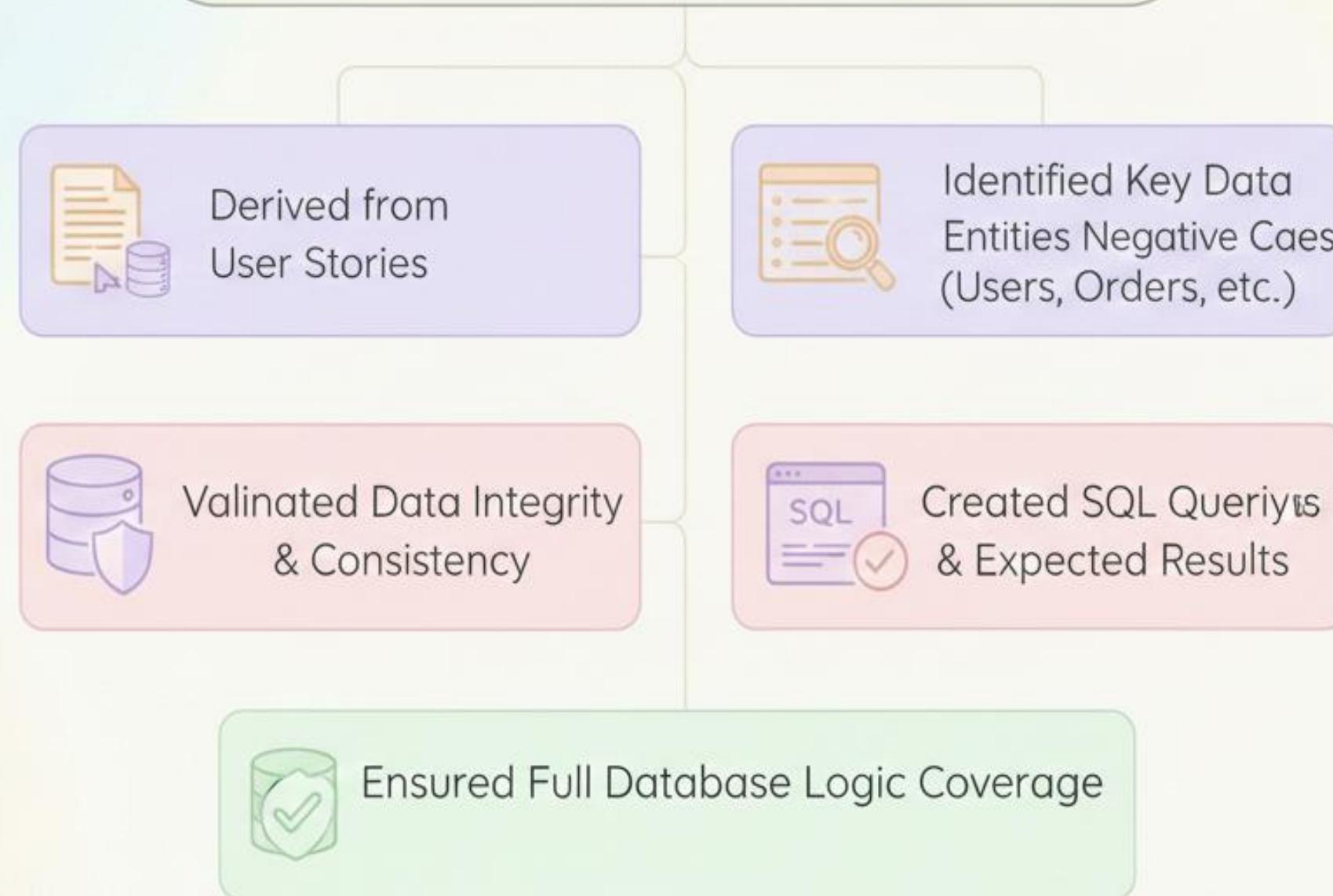
GD-113 customer Login

TO DO ▾



Database Testing Approach

Key Principles:



G4-DEPI ...

Summary Timeline Backlog Board Calendar List Forms All work Development Code Archived work items Pages

GD-10

Database Testing - QA Cart

+

Description

Add a description...

Child work items

... + 0% Done

Work	Priority	Assignee	Status
</> GD-20 CATALOG-DB-001 - Verify Product Added	= Medium	Unassigned	PASSED
</> GD-21 CATALOG-DB-002 - Verify Product Edit	^ High	Unassigned	PASSED
</> GD-23 CATALOG-DB-003 - Delete Product	= Medium	Unassigned	PASSED
</> GD-31 CATALOG-DB-005 - Add New Category	= Medium	Unassigned	PASSED
</> GD-34 REP-DT-001 - Verify Out of Stock Products in Database	= Medium	Unassigned	PASSED
</> GD-40 REP-M-002 - Verify Online Visitors in Database	▼ Low	Unassigned	PASSED
</> GD-49 Product Browsing Database Search	^ High	Unassigned	PASSED
</> GD-57 Verify the product By Category Database	^ High	Unassigned	PASSED
</> GD-72 Search in database by Keywords (Product Name,Sub Name,product id ,product not exist)	= Medium	Unassigned	PASSED
</> GD-101 add,remove and review Wish List	= Medium	Unassigned	FAILED

Made with GAMMA

Database Testing Execution

Execution Summary:

- Total designed Database Test Cases: 31
- Executed: 31

Execution covered:

- Product and category updates
- Search and browsing validations in DB
- Customer registration records
- Wishlist and cart data integrity
- Reviews insertion, approval, and linkage
- Orders, coupons, and reporting data accuracy

Defect Management:

- Failed test cases were logged with SQL evidence
- Cross-validation ensured accurate and consistent results

CATALOG-DB-001 - Verify Product Added



Test Data

Product Name: Test Product

Model: SCREEN

Price: 2500

Quantity: 1

Pre-condition

Admin successfully added a new product with Model: SCREEN

Description

Test Steps:

1. Open phpMyAdmin and select the opencart_dt database.
2. Execute the SQL query:

```
"SELECT * FROM oc_product WHERE model = 'SCREEN';
```

Expected Result

Product record exists in the database with correct Name, Model, Price, and Quantity.

Actual Result

Product record exists in the database with correct Name, Model, Price, and Quantity.

Linked work items

relates to



GD-7 Admin Product Catalog Management

Database Testing Execution

Dashboard

Catalog

- Categories
- Products
- Subscription Plans
- Filters
- Attributes
- Options
- Manufacturers
- Identifiers
- Downloads
- Reviews
- Information

CMS

Extensions

Design

Sales

Customers

Product List

	Image	Product Name ^	Model	Price	Quantity	Action
<input type="checkbox"/>		Test Product	SCREEN	\$250.00	1	<button> </button>

Showing 21 to 21 of 21 (3 Pages)

| < < 1 2 3

Filter

Product Name

Model

Categories

Manufacturer

Price

From - To

Quantity

From - To

Status

Activate Windows

Go to Settings to activate Windows.

← Server: 127.0.0.1:3307 » Database: opencart_dt » Table: oc_product

Browse Structure SQL Search Insert Export Import Privileges Operations Tracking Triggers

Run SQL query/queries on database opencart_dt: [?](#)

```
1 SELECT * FROM oc_product WHERE model = 'SCREEN';
```

Bind parameters [?](#)

Bookmark this SQL query:

Delimiter: ; Show this query here again Retain query box Rollback when finished Enable foreign key checks

Showing rows 0 - 0 (1 total, Query took 0.0003 seconds.)

```
SELECT * FROM oc_product WHERE model = 'SCREEN';
```

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Show all | Restore column order | Number of rows: 25 Filter rows:

Extra options

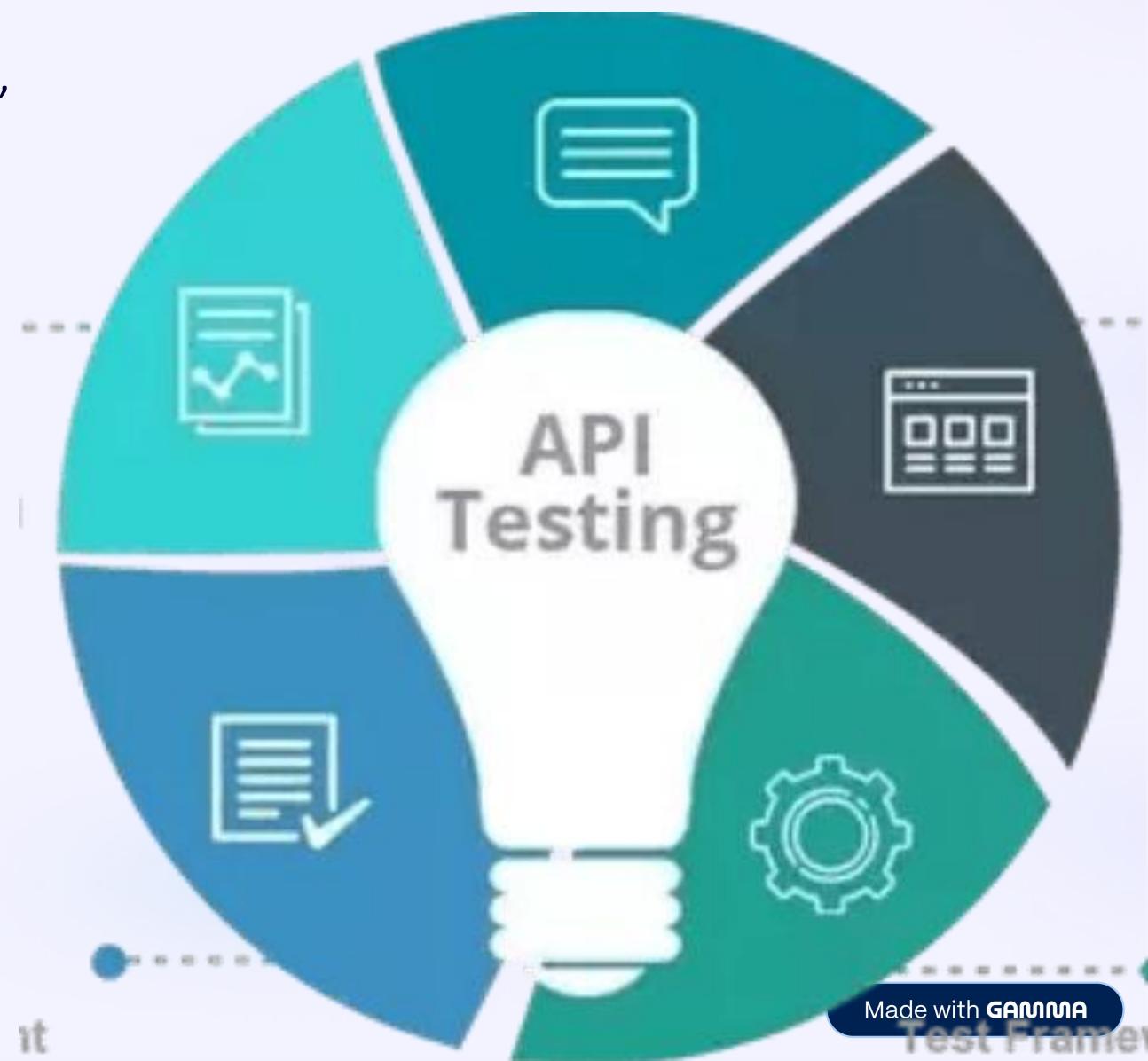
← T →	product_id	master_id	model	price	quantity	date_added	date_available	length	width	height	upc	sku	ean	jan	isbn	n
<input type="checkbox"/> Edit Copy Delete	52	0	SCREEN	250.0000	1	2025-11-14 16:21:50	2025-11-14	2.00000000	2.00000000	2.00000000	NULL	NULL	NULL	NULL	NULL	N

Activate Windows
Go to Settings to activate Windows.

Console Check all With selected

API Testing Approach

- Testing executed on [Automation Exercise](#) environment.
- API test cases derived from core application functionalities: Products, Brands, Search, Login, User Account management
- Requests configured in Postman with necessary Body data and organized into a [Postman Collection](#)
- Each test case included expected outcomes and preconditions to ensure proper validation



▼ Automation Exercise

GET Get All Products List**POST** POST To All Products List**GET** Get All Brands List**PUT** PUT To All Brands List**POST** POST To Search Product**POST** POST To Search Product without...**POST** POST To Verify Login with valid ...**POST** POST To Verify Login without e...**DEL** DELETE To Verify Login**POST** POST To Verify Login with invalid...**POST** POST To Create/Register User A...**DEL** DELETE METHOD To Delete Use...**PUT** PUT METHOD To Update User A...**GET** GET user account detail by email

API Testing Execution

Summary:

- Total API Test Cases: 14
- Executed: 14 → all passed

Coverage:

- Positive and negative scenarios for all endpoints
- Captured requests and responses for evidence
- Key endpoints tested include: Product listing, Brand management, Search
Login verification, User account creation/update/deletion

Home Workspaces API Network

Search Postman Ctrl K

My Workspace By martian-space-279024 NOT VERIFIED

New Import

Overview POST POST To Verify Login v +

HTTP Automation Exercise / POST To Verify Login with valid details

POST https://automationexercise.com/api/verifyLogin

Overview Params Authorization Headers (8) Body Scripts Settings

Body form-data

Key	Type	Description
email	Text	d4rkstri3@gmail.com
password	Text	123

Cookies

Send

Automation Exercise

- GET Get All Products List
- POST POST To All Products List
- GET Get All Brands List
- PUT PUT To All Brands List
- POST POST To Search Product
- POST POST To Search Product without search_product parameter
- POST POST To Verify Login with valid details
- POST POST To Verify Login without email parameter
- DEL DELETE To Verify Login
- POST POST To Verify Login with invalid details
- POST POST To Create/Register User Account
- DEL DELETE METHOD To Delete User Account
- PUT PUT METHOD To Update User Account
- GET GET user account detail by email

Contract Testing

Functional testing

Integration Testing

Intro to Writing Tests

Mock Data Generation

Performance Testing

Regression Testing

Test Collection of APIs

Body Cookies Headers (18) Test Results

200 OK 1.11 s 851 B Save Response

{ } JSON Preview Visualize

```
1 {  
2   "responseCode": 200,  
3   "message": "User exists!"  
4 }
```

Online Find and replace Console

Postbot Runner Start Proxy Cookies Vault Trash

Automation Testing Approach

Automation testing was carried out on [Automation Exercise](#).

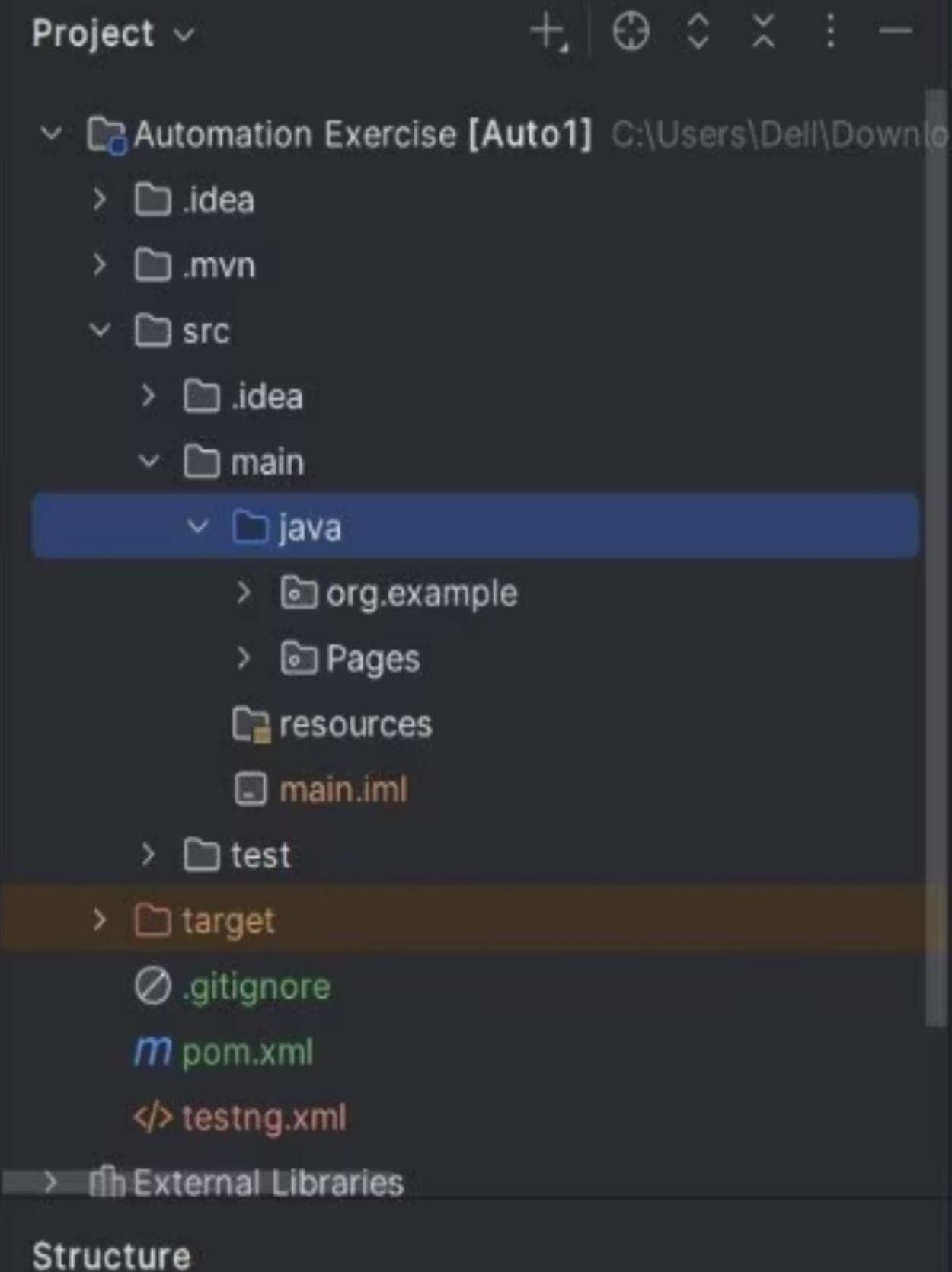
A [Page Object Model \(POM\)](#) architecture was implemented to ensure a clean, maintainable, and scalable test structure. Each application screen was represented by a dedicated [Page Class](#) containing:

- Structured element locators
- Defined user actions
- Reusable interaction methods

Scenario logic was developed within independent [Test Classes](#), ensuring clear separation between test flow and page structure.

Automation Framework Structure

The automation framework was organized into structured layers to ensure clarity, reusability, and ease.



1 — pom.xml Layer:

Manages all project dependencies, including libraries like Selenium and TestNG, ensuring consistent versions across different environments.

Handles the build process and test execution, making it easy to maintain the project and quickly set up the framework for new team members.



maven
pom.xml

Project

+ ⊕ ◁ × : -

m pom.xml (Auto1) ×

Automation Exercise [Auto1] C:\Users\Dell\Downloaded

- > .idea
- > .mvn
- src
 - > .idea
 - main
 - > java
 - > org.example
 - > Pages
 - resources
 - main.iml
 - test
 - > java
 - test.iml
 - > target
- .gitignore
- m pom.xml
- </> testng.xml

> External Libraries

≡ Scratches and Consoles

```
1   <?xml version="1.0" encoding="UTF-8"?>
2   <project xmlns="http://maven.apache.org/POM/4.0.0"
3           xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4           xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-
5           pom-4.0.0.xsd">
6       <modelVersion>4.0.0</modelVersion>
7
8       <groupId>org.example</groupId>
9       <artifactId>Auto1</artifactId>
10      <version>1.0-SNAPSHOT</version>
11
12      <properties>
13          <maven.compiler.source>21</maven.compiler.source>
14          <maven.compiler.target>21</maven.compiler.target>
15          <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
16          <selenium.version>4.15.0</selenium.version>
17          <testng.version>7.8.0</testng.version>
18      </properties>
19
20      <dependencies>
21          <dependency>
22              <groupId>org.seleniumhq.selenium</groupId>
23              <artifactId>selenium-java</artifactId>
24              <version>${selenium.version}</version>
25          </dependency>
26
27          <dependency>
28              <groupId>org.testng</groupId>
29              <artifactId>testng</artifactId>
```

Automation Framework Structure

2- Pages Layer

Contains all Page Classes (Login, Register, Product, Cart....).



Element
Locators



User
Actions



Reusable Helper
Helper Methods

Project ▾

> org.example

Pages

AccountCreatedPage

AccountDeletedPage

AccountInfoPage

AddReview

BasePAGE

Cart

Checkout

ContactUs

Home

Login

Payment

ProductDetails

Products

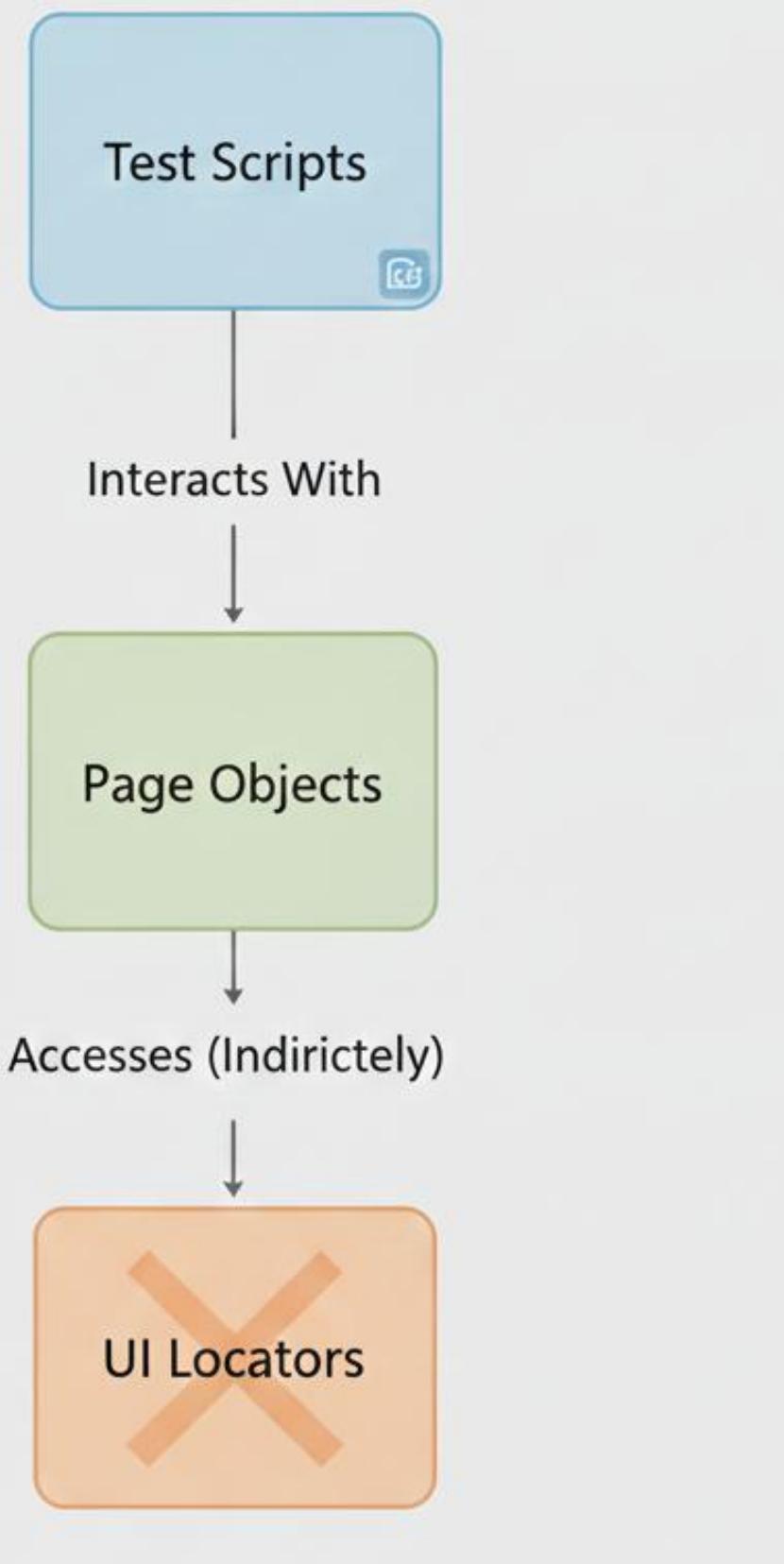
Signup

resources

Structure

© Home.java ×

```
1 package Pages;
2
3 > import ...
4
5
6 public class Home extends BasePAGE { new *
7
8     public Home(WebDriver driver) { new *
9         super(driver);
10        PageFactory.initElements(driver, page: this);
11    }
12
13
14
15
16
17
18 @FindBy(xpath = "//div[@class='logo pull-left']") 1 usage
19 WebElement homeLogo;
20
21 @FindBy(xpath = "//*[@id=\"header\"]//div/div/div[2]/div/ul/li[8]/a") 1 usage
22 WebElement contactUsButton;
23
24 @FindBy(xpath = "//button[text()='Continue Shopping']") 2 usages
25 WebElement continueShoppingButton;
```



Automation Framework Structure

3-Test Layer:

Includes TestNG test classes representing the actual test scenarios. Test scripts interact with Page Objects only, without directly accessing UI locators

Project Automation Exercise [Auto1] C:\Users\Hossam\Downloads

Structure

SuccessfulRegistration.java

```
1 import Pages.*;
2 import org.openqa.selenium.WebDriver;
3 import org.openqa.selenium.chrome.ChromeDriver;
4 import org.testng.Assert;
5 import org.testng.annotations.BeforeClass;
6 import org.testng.annotations.AfterClass;
7 import org.testng.annotations.Test;
8
9 public class SuccessfulRegistration { new*
10     WebDriver driver; 10 usages
11
12 @BeforeClass new*
13     public void setup() {
14         driver = new ChromeDriver();
15         driver.manage().window().maximize();
16         driver.get("https://automationexercise.com/");
17     }
18
19 @Test new*
20     public void RegisterUser() {
21         Home home = new Home(driver);
22         Assert.assertTrue(home.isHomePageVisible());
23
24         home.clickSignupLogin();
25
26         Signup signup = new Signup(driver);
27         Assert.assertTrue(signup.isNewUserTextVisible());
28
29         signup.enterNameAndEmail( name: "Youssef", email: "youssef" + System.currentTimeMillis() + "@test.com")
30         signup.clickSignup();
31
32         AccountInfoPage accountInfo = new AccountInfoPage(driver);
```

Activate Window

Go to Settings to...

Project

Home.java

Signup.java

SuccessfulRegistration.java

SuccessfulLogin.java

ExistingSignup.java

⚠ 1 ✅ 1 ^ ⌂

Automation Exercise [Auto1] C:\Users\Dell\Download

- > .idea
- > .mvn
- ✓ src
 - > .idea
 - > main
 - > java
 - org.example
 - Pages
 - resources
 - main.iml
 - > test
 - > java
 - AddReviewTest
 - ContactUsForm
 - ExistingSignup
 - IncorrectUserLogin
 - PlaceOrderTest
 - RemoveProductsFromCart
 - SearchProduct
 - SearchProductNotExist
 - SuccessfulLogin
 - SuccessfulRegistration
 - VerifyAllProducts
 - VerifyProductquantityInCart
 - test.iml
 - > target

Structure

SuccessfulRegistration

```
9  public class SuccessfulRegistration { new *
10    public void RegisterUser() {
11
12      signup.enterNameAndEmail( name: "Youssef", email: "youssef" + System.currentTimeMillis() + "@test.com");
13      signup.clickSignup();
14
15      AccountInfoPage accountInfo = new AccountInfoPage(driver);
16      accountInfo.fillAccountInfo( pass: "123456");
17      accountInfo.fillAddressInfo();
18      accountInfo.clickCreateAccount();
19
20      AccountCreatedPage created = new AccountCreatedPage(driver);
21      Assert.assertTrue(created.isAccountCreatedVisible());
22      created.clickContinue();
23
24      Login logged = new Login(driver);
25      Assert.assertTrue(logged.isLoggedInVisible());
26    }
27
28    @AfterClass new *
29    public void teardown() {
30      if (driver != null) {
31        driver.quit();
32      }
33    }
34  }
```

Activate Win
Go to Settings



Automation Framework Structure

4-TestNG.xml:

Used for grouping and organizing test executions, enabling selective or combined runs.

Automation Testing Execution

Execution Summary:



Executed: **12**



Passed: **12**



Failed: **0**

Execution Coverage:

- Core customer flows automated:
- Registration: SuccessfulRegistration, Existing Signup
- Login: SuccessfulLogin, IncorrectUserLogin
- Product Search: Search Product, Search ProductNotExist, ViewCategoryProducts
- Cart Management: Remove ProductsFromCart, Verify ProductQuantityInCart
- Product Listing: Verify AllProducts
- Contact Form: ContactUsForm
- Reviews: AddReview Test
- Checkout & Orders: PlaceOrder Test

Automation Testing Execution

Defect Management:

- No critical failures occurred during execution
- All automation scripts passed successfully

AD AccountDeletedPage.java ▾

main ▾

Current File ▾

Project ▾

Run C:/Users/Amira/Desktop/Grad - Group4/Grad - Group4/... ▾

n.xml (Auto1) BasePAGE.java Cart.java Checkout.java ContactUs.java Signup.java testng.xml

22 tests passed 22 tests total, 4 min 33 sec

WARNING: Unable to find CDP implementation matching 143
Dec 07, 2025 8:18:24 PM org.openqa.selenium.chromium.ChromiumDriver Lambda\$new\$5
WARNING: Unable to find version of CDP to use for 143.0.7499.40. You may need to include a dependency on a specific version of the CDP usin
STEP 3: Verify Home Page is visible successfully
STEP 4: Add products to cart
STEP 5: Click 'Cart' button
STEP 6: Verify that cart page is displayed
STEP 7: Click 'X' button corresponding to particular product
STEP 8: Verify that product is removed from the cart
Closing browser
Dec 07, 2025 8:18:40 PM org.openqa.selenium.devtools.CdpVersionFinder findNearestMatch
WARNING: Unable to find CDP implementation matching 143
Dec 07, 2025 8:18:40 PM org.openqa.selenium.chromium.ChromiumDriver Lambda\$new\$5
WARNING: Unable to find version of CDP to use for 143.0.7499.40. You may need to include a dependency on a specific version of the CDP usin
Dec 07, 2025 8:18:58 PM org.openqa.selenium.devtools.CdpVersionFinder findNearestMatch
WARNING: Unable to find CDP implementation matching 143
Dec 07, 2025 8:18:58 PM org.openqa.selenium.chromium.ChromiumDriver Lambda\$new\$5
WARNING: Unable to find version of CDP to use for 143.0.7499.40. You may need to include a dependency on a specific version of the CDP usin
=====
AutomationExerciseSuite
Total tests run: 22, Passes: 22, Failures: 0, Skips: 0
=====

Process finished with exit code 0

Automation Exercise > testng.xml

1:1 CRLF UTF-8 4 spaces

Type here to search

Test Results Summary

Testing Type	Written	Executed	Passed	Failed
Manual	102	95	87	7
Database	31	31	25	6
API	14	14	14	0
Automation	12	12	12	0

Final Conclusion

This project gave us extensive, hands-on experience in:

- Manual Testing
- Database & API Testing
- Automation Testing
- Defect Management
- Documentation

We also strengthened our skills in:

- Analyzing requirements
- Designing effective test cases
- Executing tests systematically
- Reporting findings clearly

Overall, we successfully simulated a complete QA lifecycle, reflecting real-world industry standards and enhancing both our technical expertise and teamwork skills.

Thank You



Repository Link: https://github.com/amirasamy393-maker/CAI3_SWD6_G3_Group4