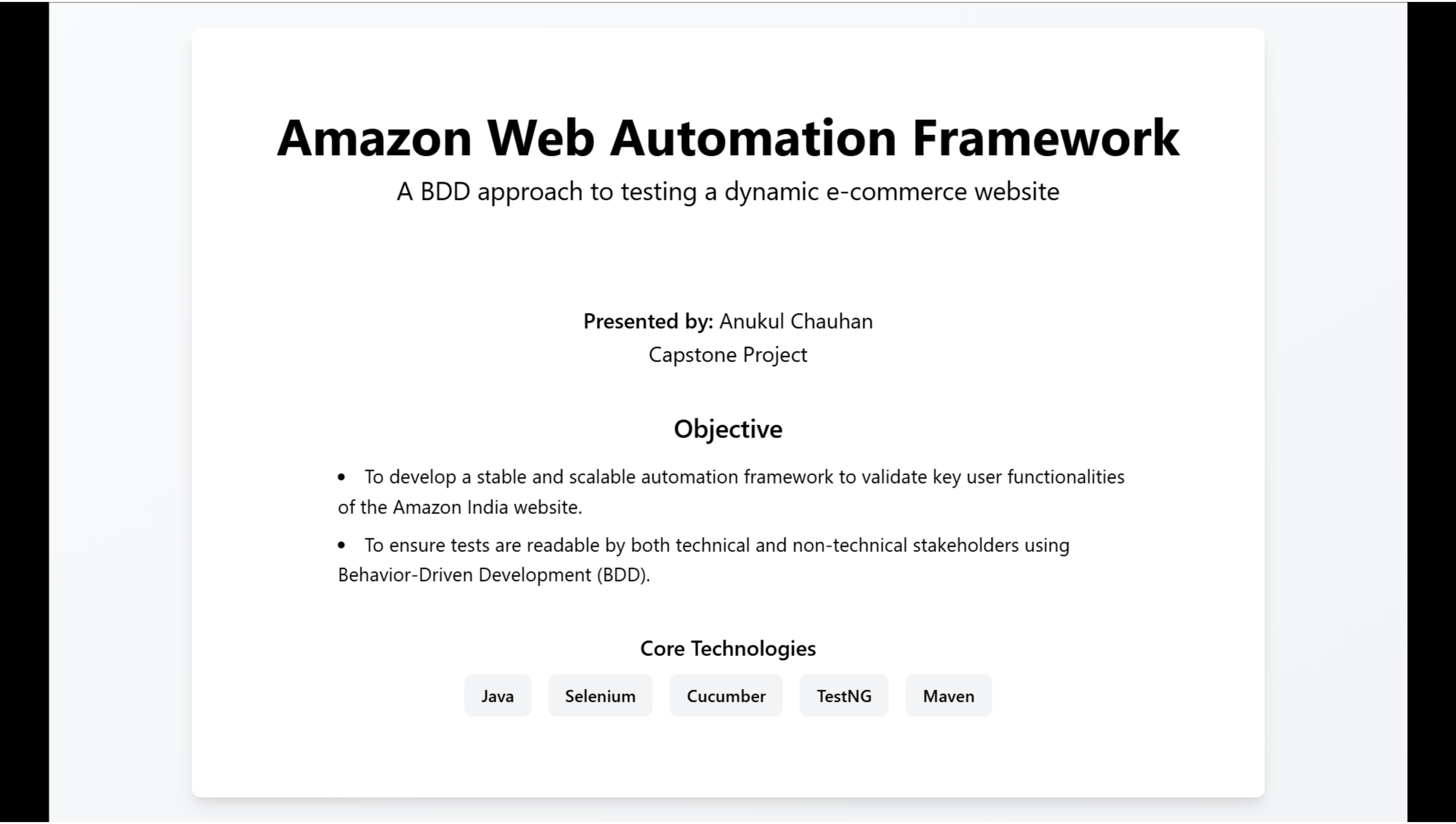


Amazon Web Automation Framework

A BDD approach to testing a dynamic e-commerce website

Presented by: Anukul Chauhan
Capstone Project

Objective

- To develop a stable and scalable automation framework to validate key user functionalities of the Amazon India website.
- To ensure tests are readable by both technical and non-technical stakeholders using Behavior-Driven Development (BDD).

Core Technologies

Java

Selenium

Cucumber

TestNG

Maven

Project Overview

This project delivers a robust and scalable BDD-based automation framework for Amazon India. It ensures essential site features are reliably tested, and scenarios are easy to read for both technical and non-technical stakeholders.

Key Benefits

Reduces Manual Effort

Manually testing a large site like Amazon is slow and error-prone.

Handles a Dynamic UI

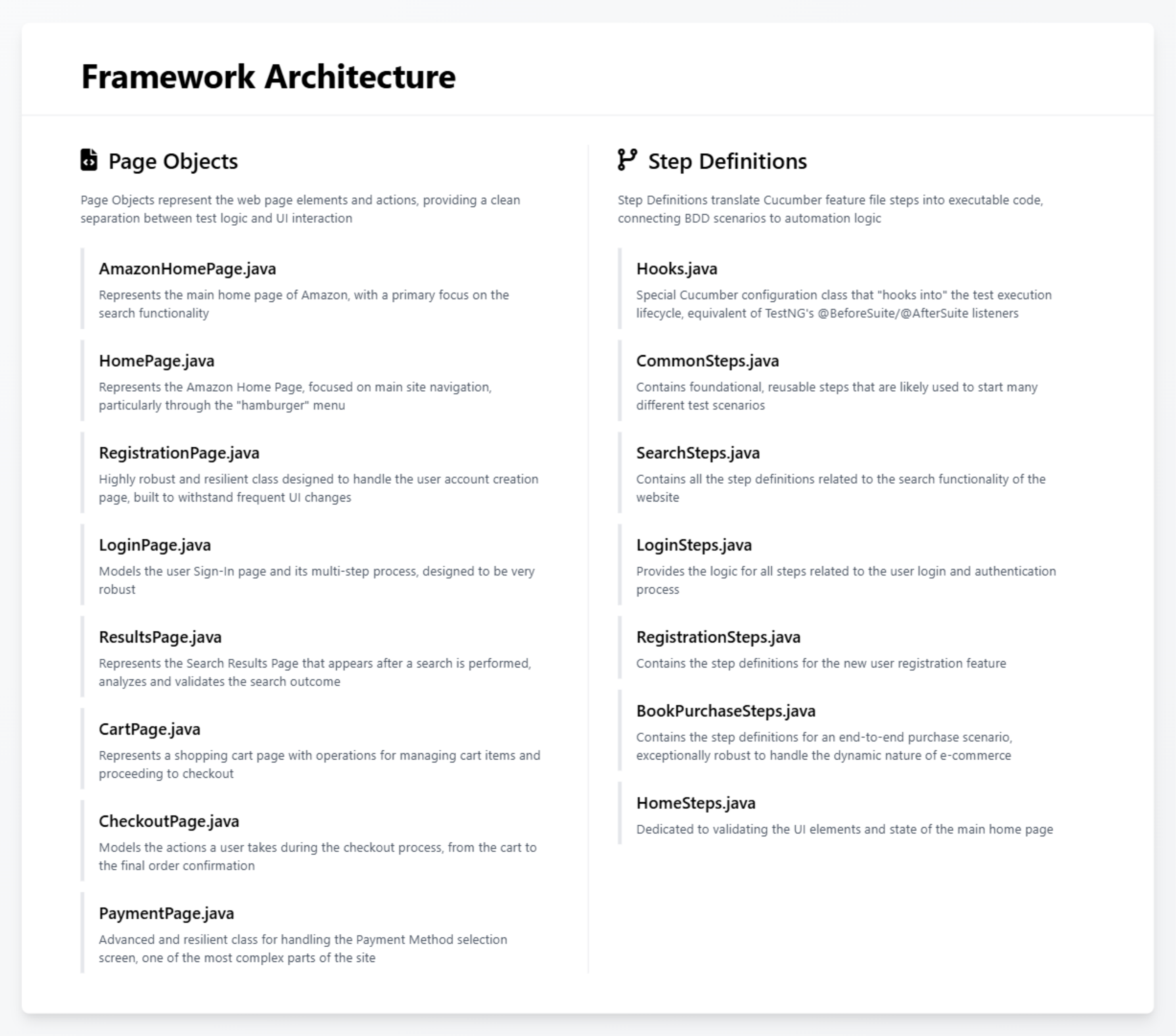
Amazon's website changes constantly, which can easily break simple test scripts.

Improves Collaboration

Traditional test code is hard for business analysts to read.

Value Proposition

- ✓ To automate the core customer experience, including user login.
- ✓ To automate critical user journeys like **Product Search, User Authentication, and the Checkout process.**
- ✓ Simplifies test creation and maintenance



Framework Architecture

Page Objects

Page Objects represent the web page elements and actions, providing a clean separation between test logic and UI interaction

AmazonHomePage.java

Represents the main home page of Amazon, with a primary focus on the search functionality

HomePage.java

Represents the Amazon Home Page, focused on main site navigation, particularly through the "hamburger" menu

RegistrationPage.java

Highly robust and resilient class designed to handle the user account creation page, built to withstand frequent UI changes

LoginPage.java

Models the user Sign-In page and its multi-step process, designed to be very robust

ResultsPage.java

Represents the Search Results Page that appears after a search is performed, analyzes and validates the search outcome

CartPage.java

Represents a shopping cart page with operations for managing cart items and proceeding to checkout

CheckoutPage.java

Models the actions a user takes during the checkout process, from the cart to the final order confirmation

PaymentPage.java

Advanced and resilient class for handling the Payment Method selection screen, one of the most complex parts of the site

Step Definitions

Step Definitions translate Cucumber feature file steps into executable code, connecting BDD scenarios to automation logic

Hooks.java

Special Cucumber configuration class that "hooks into" the test execution lifecycle, equivalent of TestNG's @BeforeSuite/@AfterSuite listeners

CommonSteps.java

Contains foundational, reusable steps that are likely used to start many different test scenarios

SearchSteps.java

Contains all the step definitions related to the search functionality of the website

LoginSteps.java

Provides the logic for all steps related to the user login and authentication process

RegistrationSteps.java

Contains the step definitions for the new user registration feature

BookPurchaseSteps.java

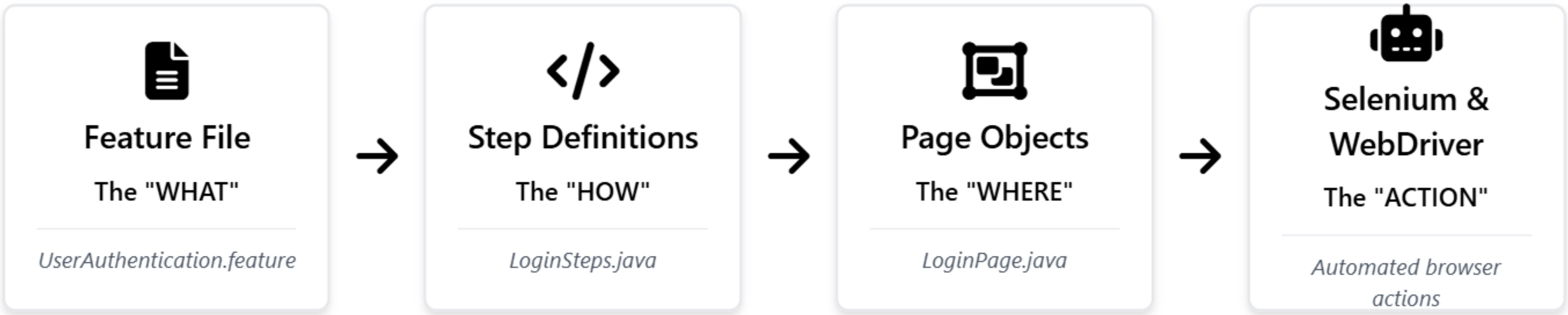
Contains the step definitions for an end-to-end purchase scenario, exceptionally robust to handle the dynamic nature of e-commerce

HomeSteps.java

Dedicated to validating the UI elements and state of the main home page

The Test Automation Flow

The automation follows this process:



Feature File

The "WHAT"

UserAuthentication.feature

→

Step Definitions

The "HOW"

LoginSteps.java

→

Page Objects

The "WHERE"

LoginPage.java

→

Selenium & WebDriver

The "ACTION"

Automated browser actions

Example Workflow:

- **Feature File:** UserAuthentication.feature defines "Given I am on login page, When I enter valid credentials..."
- **Step Definition:** LoginSteps.java translates the English steps into Java code
- **Page Object:** LoginPage.java interacts with the web elements on Amazon's login page
- **WebDriver:** Executes browser actions to fill the login form and submit it

Advanced Reporting

Effective reporting is crucial for test automation. Our framework provides comprehensive evidence and insights through two complementary approaches:

Extent Reports

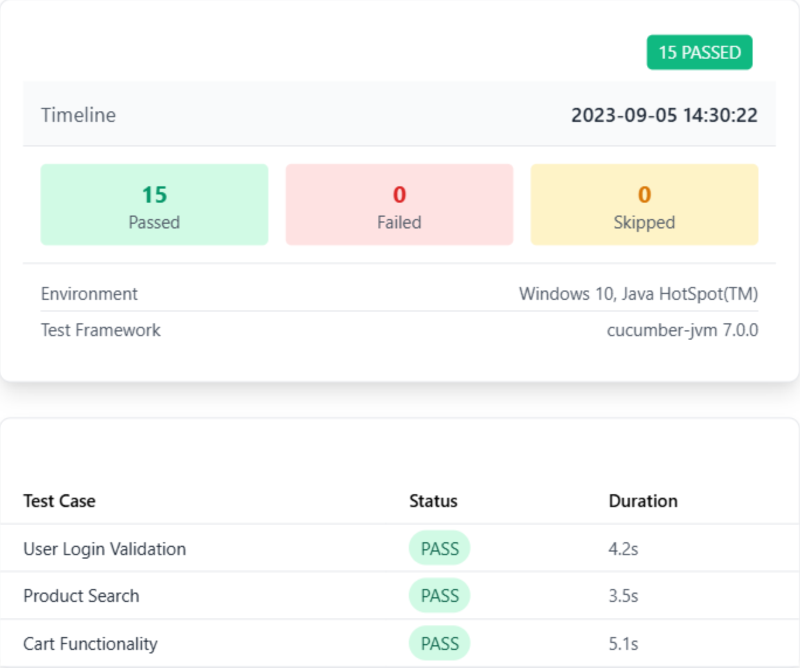
Provides visual, detailed technical logs with:

- Step-by-step execution details
- Automatic screenshots for failures
- Test execution timelines
- Environment information
- Interactive dashboard for analysis

Excel Summary

Offers a simple, quick overview with:

- Pass/fail statistics at a glance
- Test execution duration
- Test case categorization
- Perfect for non-technical stakeholders



15 PASSED

Timeline	2023-09-05 14:30:22

15	0	0
Passed	Failed	Skipped

Environment	Windows 10, Java HotSpot(TM)
Test Framework	cucumber-jvm 7.0.0

Test Case	Status	Duration
User Login Validation	PASS	4.2s
Product Search	PASS	3.5s
Cart Functionality	PASS	5.1s

Feature File Structure

Behavior-Driven Development (BDD)

BDD is a collaborative approach that makes test scenarios clear and accessible to all stakeholders, including non-technical team members.

Given-When-Then Format

Given

Establishes the initial context or preconditions

"Given I am on the Amazon homepage"

When

Describes the action being taken

"When I search for 'headphones'"

Then

Validates the expected outcome

"Then I should see search results"

And

Extends Given, When, or Then with additional context

"And the results should include items from Electronics category"

```
Feature: User Authentication

  As a customer
  I want to log into my account
  So that I can view my personal information

  Scenario: User login with valid credentials
    Given I am on the Amazon login page
    When I enter valid email and password
    And I click on sign-in button
    Then I should be logged in successfully
    And I should see my account name

  Scenario: User login with invalid credentials
    Given I am on the Amazon login page
    When I enter invalid email and password
    And I click on sign-in button
    Then I should see an error message
    And I should remain on the login page
```

UserAuthentication.feature

Test Execution Examples

Our framework demonstrates robust capabilities in handling various test scenarios, from simple searches to complex authentication flows.

Key Test Scenarios

User Authentication Tests

Validates login functionality with both valid and invalid credentials

- ✓ Invalid email validation
- ✓ Incorrect password handling
- ✓ Two-factor authentication flows

Product Search Tests

Verifies search functionality and results accuracy

- ✓ Keyword search validation
- ✓ Search results filtering
- ✓ Product details verification

End-to-End Purchase Flow

Tests complete customer journey from search to checkout

Test: Invalid Login Attempt Failed

```
Error: The email you entered was not found in our system.
```

LoginSteps.java:126 Duration: 3.42s

Test: Search "mosquito all out" Passed

```
Found 24 results for "mosquito all out"
Verified product titles contain search terms
```

SearchSteps.java:85 Duration: 2.17s

Robust against UI changes