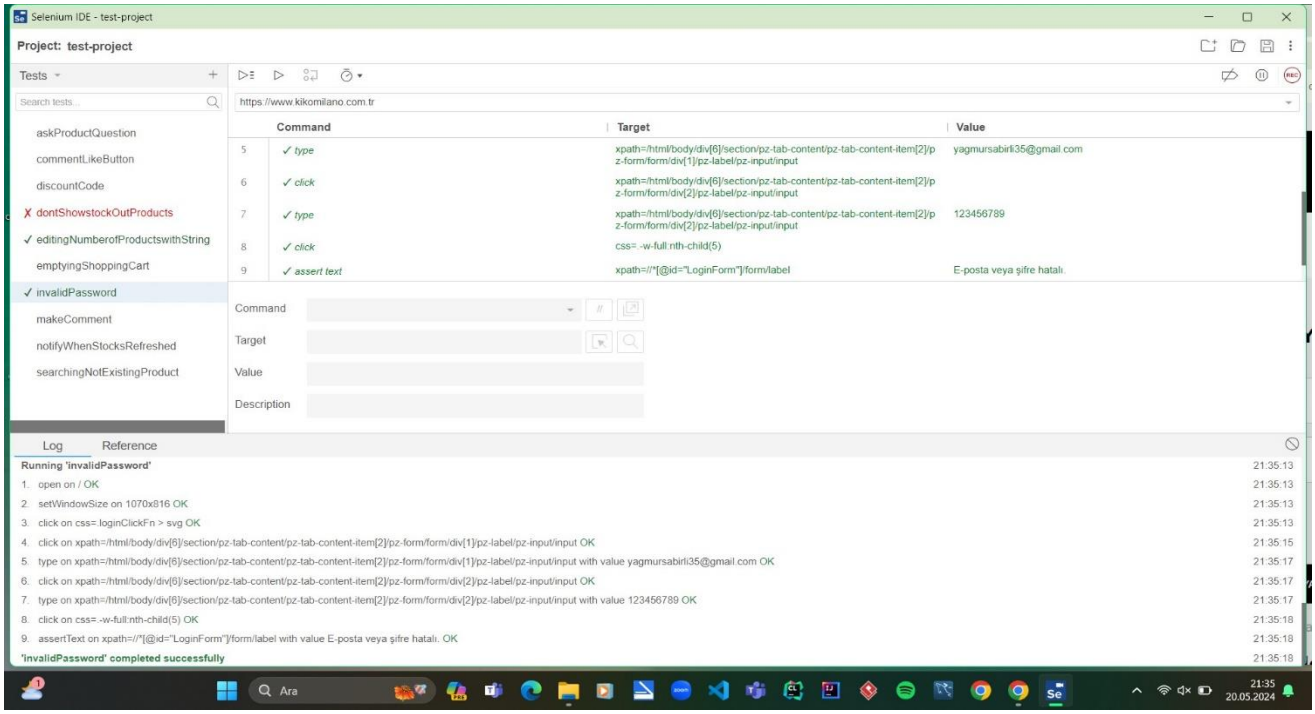
# SE 2226 Test Completion Report

## 1. Summary of testing performed

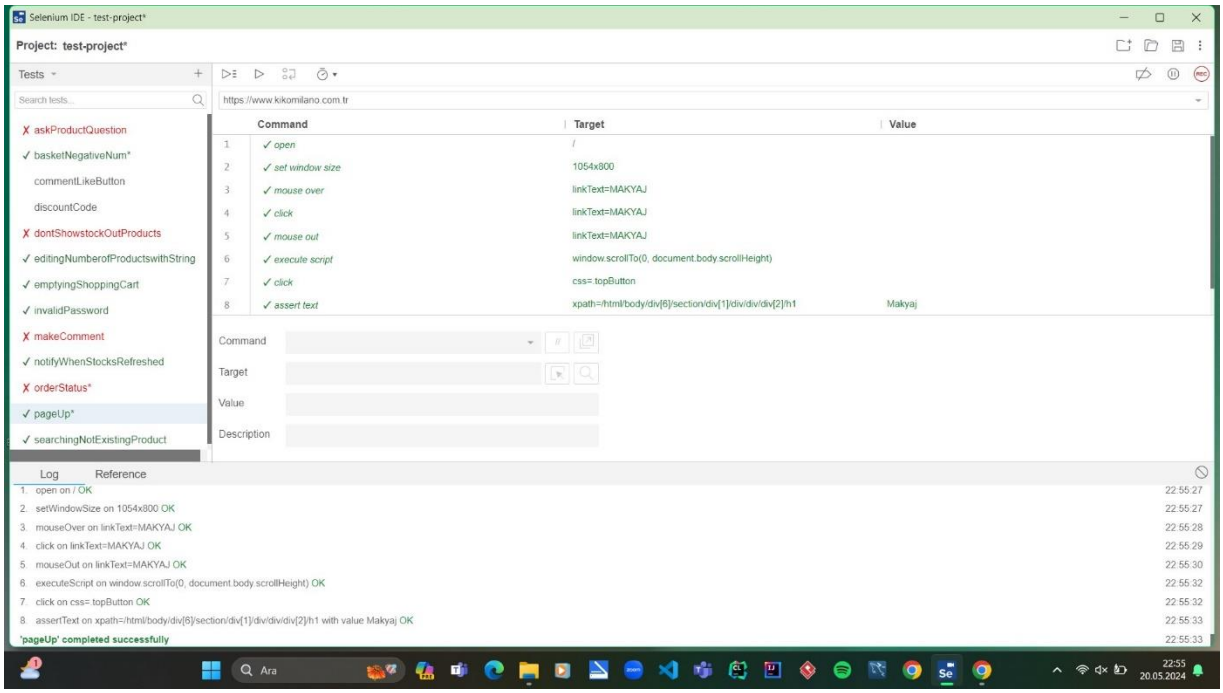
For this project, we tested the website of 'Kiko Milano' (https://www.kikomilano.com.tr/) using three different testing methods: Selenium IDE web extension, JMeter and Blackbox testing.

**Selenium IDE**: We performed 12 specific tests using this tool:

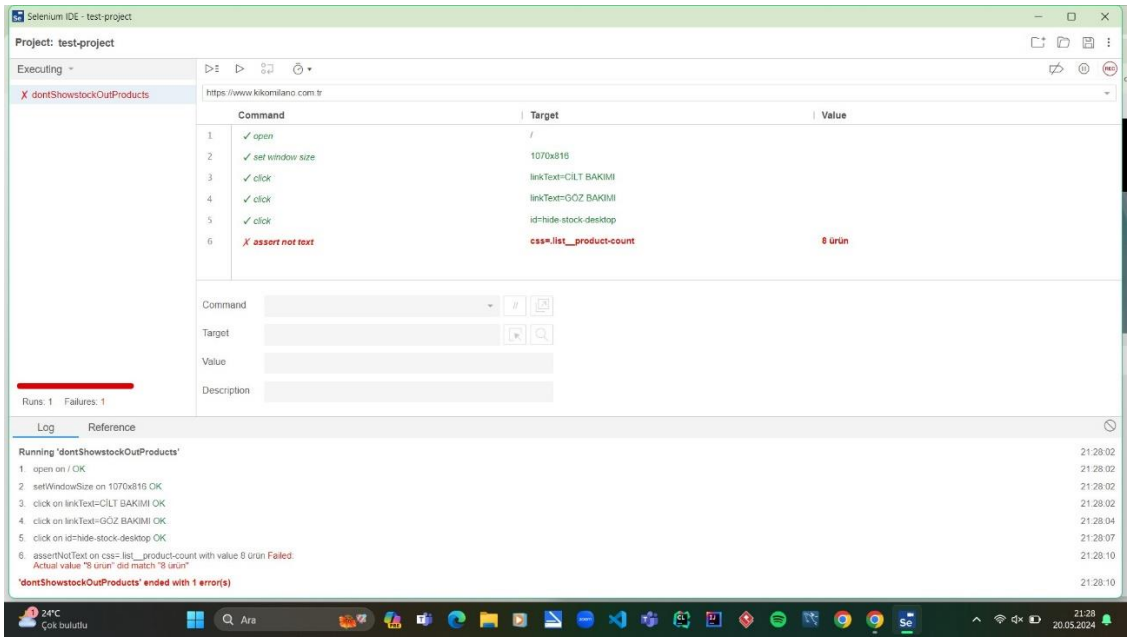
1. **invalidPassword:** This test attempts to log in with an incorrect password. It verifies if the website displays an error message.



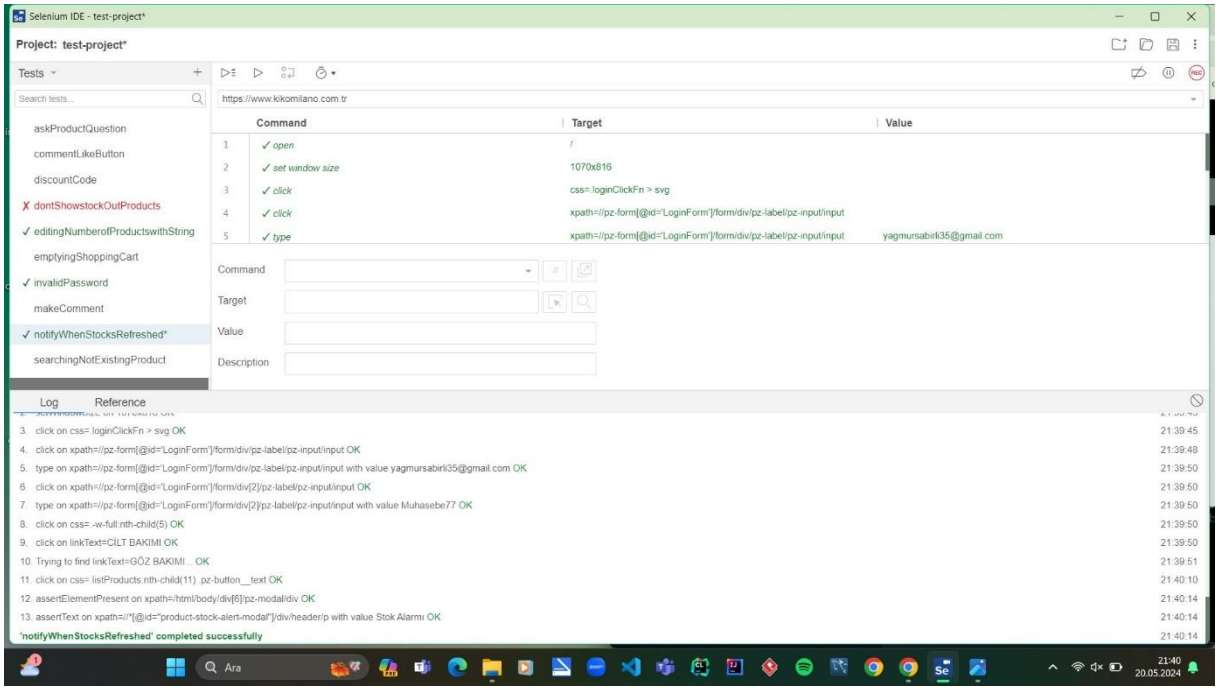
2. **pageUp:** This test verifies the functionality of the ‘Page Up’ button on this webpage. It ensures that the page scrolls up correctly when the button is used.



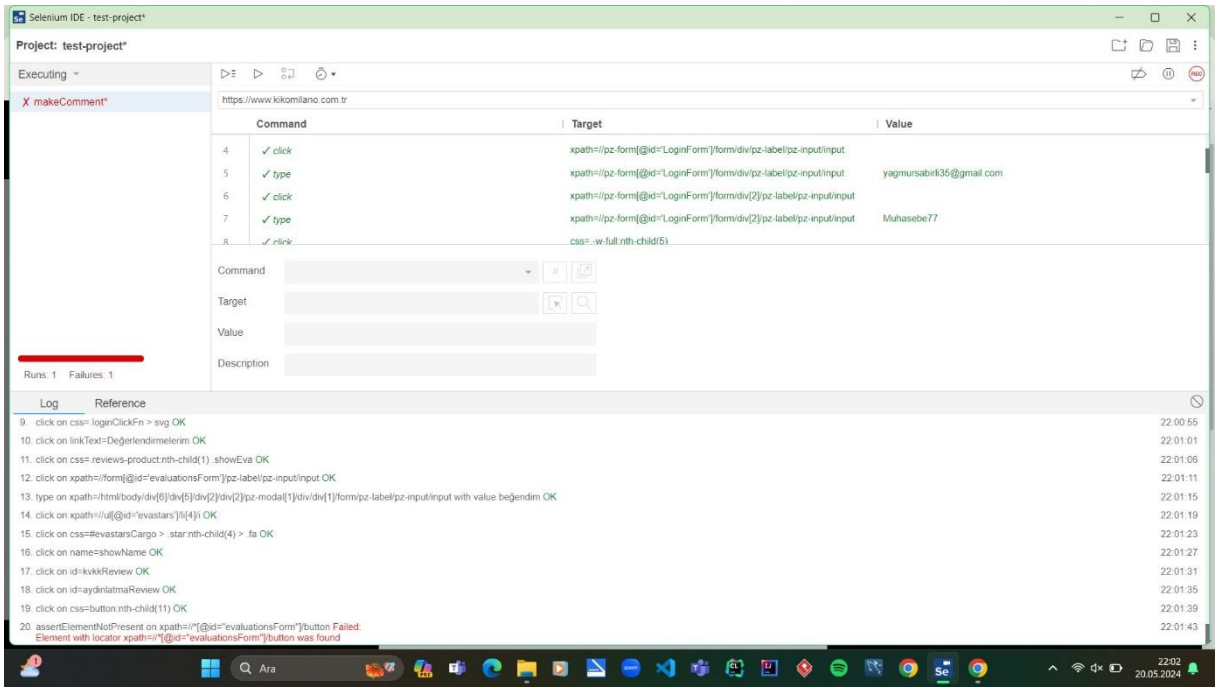

3. **dontShowStockOutProducts:** This test checks if the website correctly displays the number of products in stock and ensures that ‘out of stock’ products are not shown. This test failed because when the ‘Stokta Olmayanları Gösterme’ button is used the part that says number of products did not change.



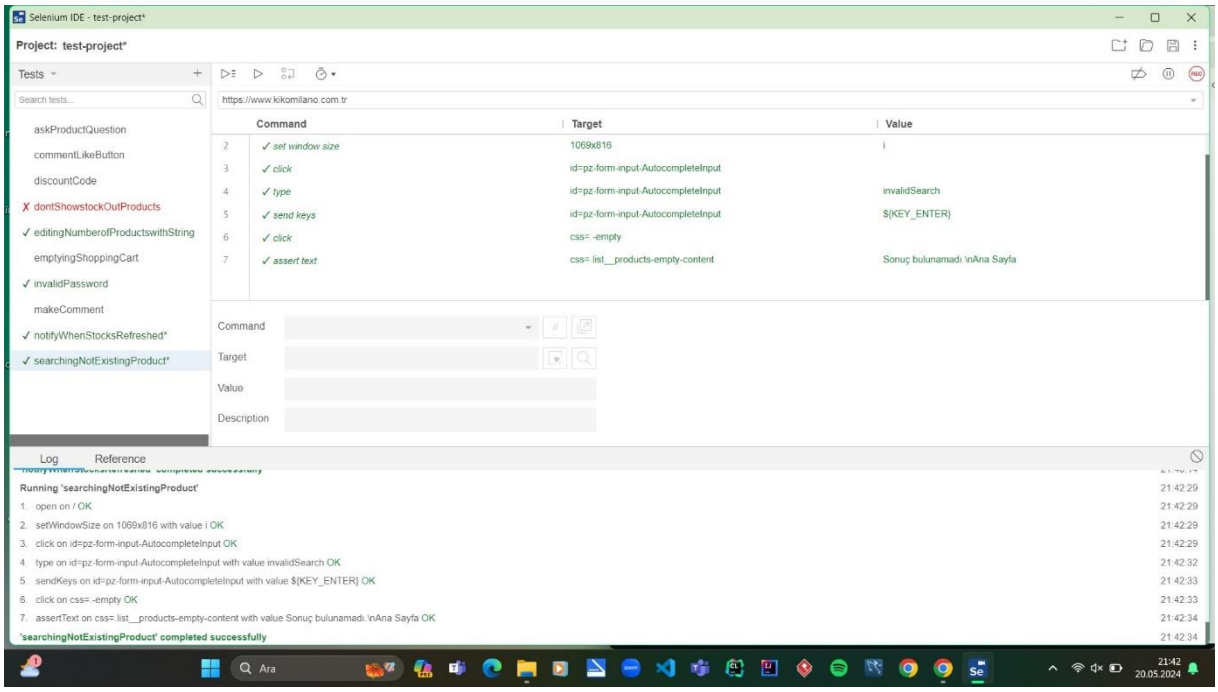
4. **notifyStocksRefreshed:** This test checks if the ‘Stoğu Gelince Haber Ver’ button works properly.



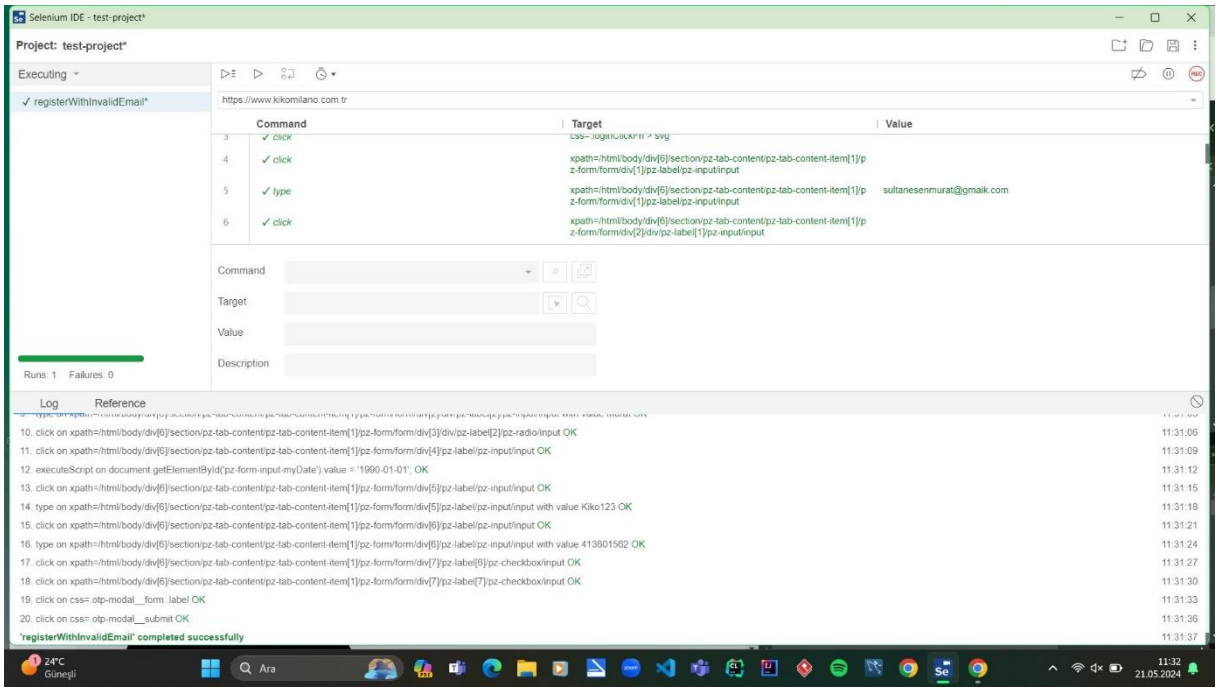

5. **makeComment:** This test tries to add a comment to a product. This test fails because after writing the comment and filling the necessary parts the ‘Gönder’ button does not work.



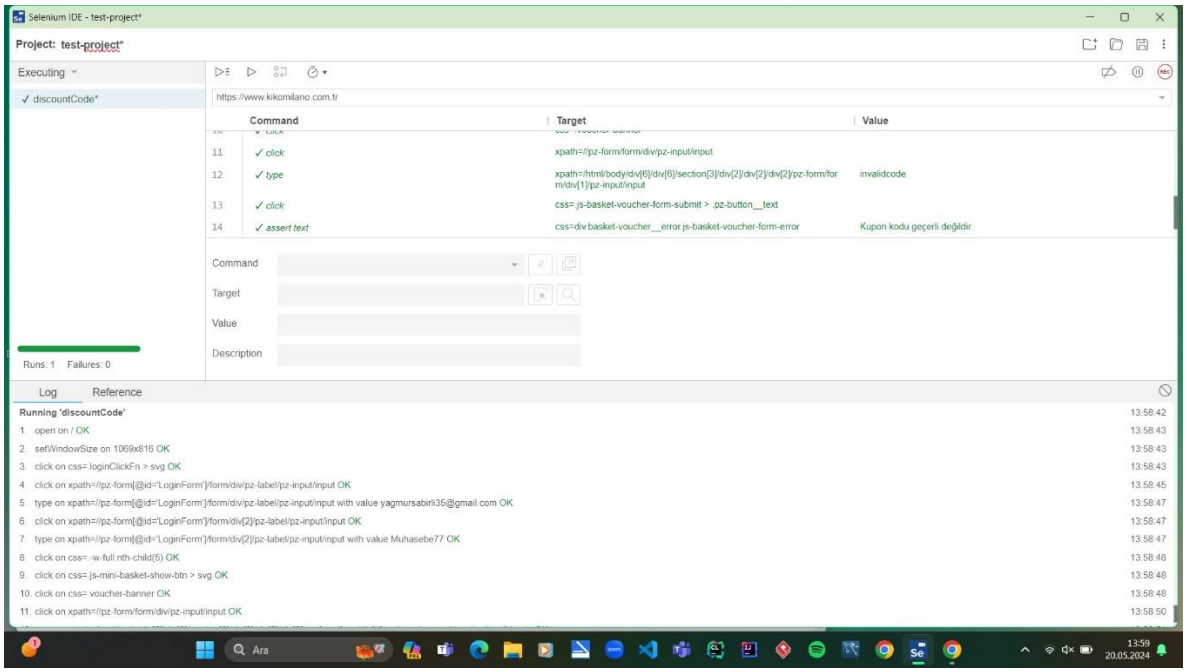
6. **SearchingNotExistingProduct:** This test performs a search using invalid or nonsensical search terms. It checks if the website handles invalid searches correctly and displays appropriate message.



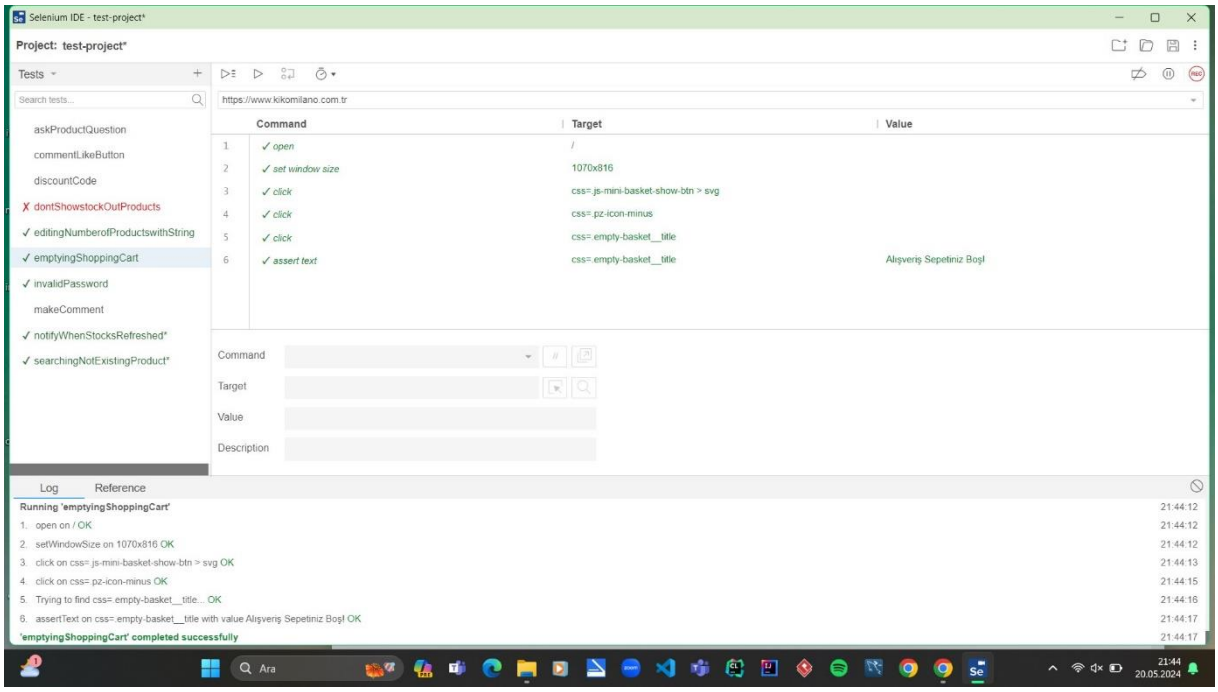
7. **registerWithInvalidEmail:** This test attempts to register a new user account using an invalid email format. It checks if the website validates the email and displays appropriate error message. This test shoul be fail but, it did not. We could be able to register with an invalid email.



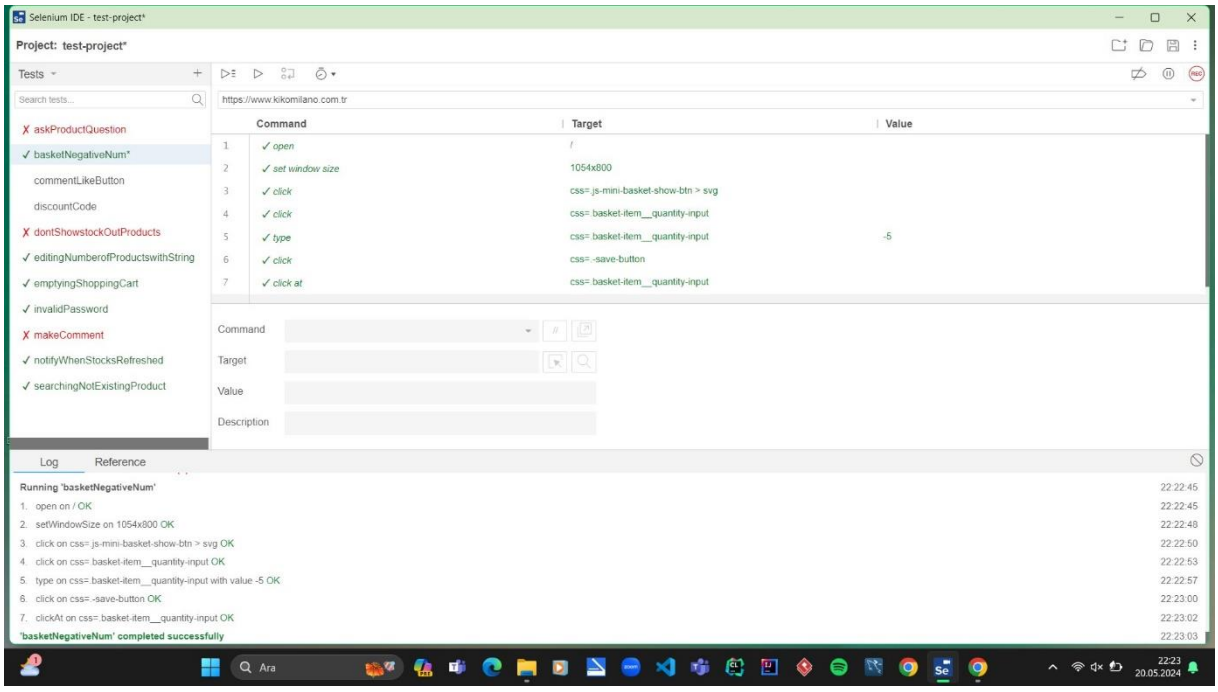

8. **discountCodeTest:** This test enters an invalid discount code during checkout and checks if the website displays an error message.



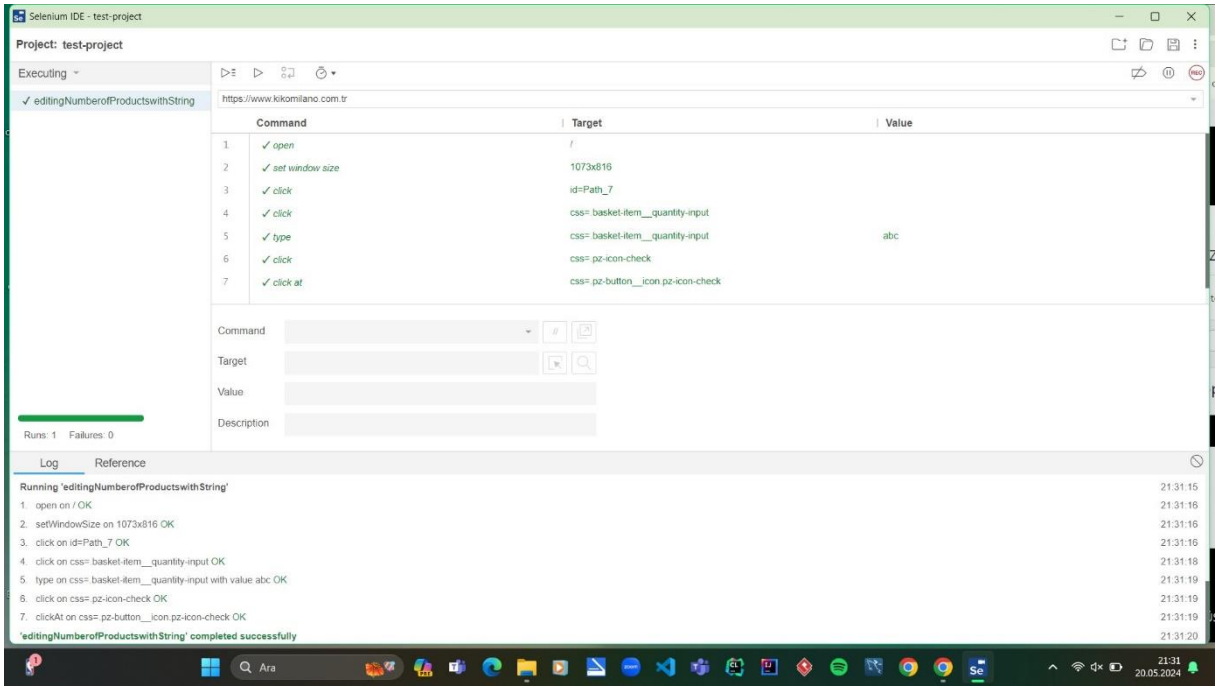
9. **emptyingShoppingCart:** This test clicks on the minus icon when there is one product in the shopping cart and checks if the shopping cart is empty.



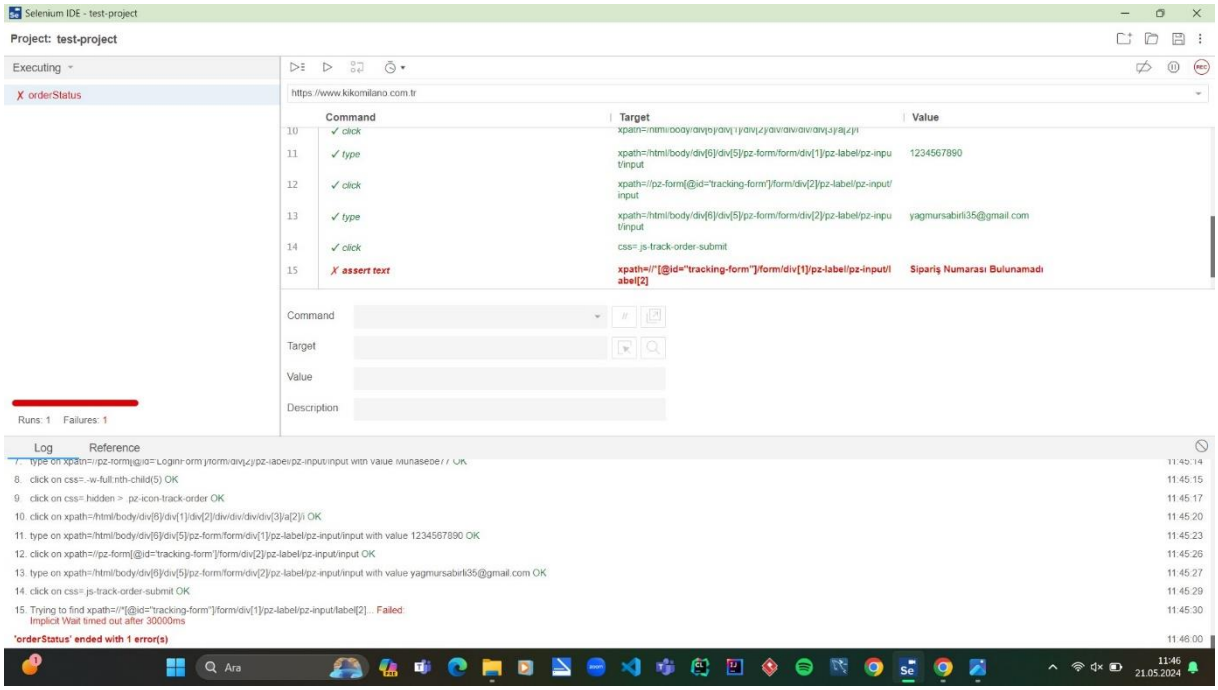

10. **basketNegNum:** This test tries to change the quantity of a product in the shopping cart by enterin a negative number. It checks if the website handles invalid input correctly. This test should be fail. A negative number of product in the shopping cart cannot be exists.



11. **editingProductNumberWithString:** This test tries to change the quantity of a product in the shopping cart by entering a non-numeric string. It checks if the website handles invalid input correctly. This test should be fail. A non-numeric string as product quantity in the shopping cart cannot be exists.



12. **orderStatu:** This test enters a invalid order number and checks if the website displays an error message. The test should not be fail. It failed because the website does not display an error message.




**Blackbox Testing:** Conducted test from a user's perspective without knowledge of the internal code structure. These tests focused on user scenarios and overall functionality.

1. **Login Functionality**

   The tests covered: •Valid and invalid email and password combinations.

   •Boundary value analysis for email and password lengths.

| Equivalence Classes | |
|---|---|
| E1 | 3<=email_address<=254 |
| U1 | 3>email_address |
| U2 | email_address>254 |
| E2 | 4<= password<=100 |
| U3 | Password<3 |
| U4 | password>100 |

## Equivalence Class

| Equivalence Classes | Test # | Test Input | Expected Input |
|---|---|---|---|
| E1E2 | 1 | “esmur@gmail.com”, “Ye1234” | Valid |
| E1U3 | 2 | “esmur@gmail.com”, “12” | Error |
| E1U4 | 3 | “esmur@gmail.com”, "Loremipsumdolorsitametconsecteturadipiscingelit Seddoeiusmodtemporincididuntutlaboreetdolorem agnaaliquaUtenimadminimveniamquisnostrudexer citationullamcolaborisnisiutaliquipexeacommodo consequat" | Error |
| E2U1 | 4 | “y@gmail.com”, “SecurePass35” | Error |
| E2U2 | 5 | "invalidemailaddress12345678901234567890123 456789012345678901234567890123456789012345 678901234567890123456789012345678901234567 890123456789012345678901234567890123456789 012345678901234567890123456789012345678901 234567890123456789012345678901234567890123 456789012345678901234567890123456789012345 678901234567890123456789012345678901234567 890123456789012345678901234567890123456789 012345678901234567890123456789012345678901 234567890123456789012345678901234567890 @example.com", “YagmurEsen.35” | Error |
| U1U3 | 6 | “z@hotmail.com”, “23c” | Error |
| U1U4 | 7 | “20@outlook.com”, “123456789012345678901234 567890123456789012345678901234567890123456 789012345678901234567890123456789012345678 901234567890123456789012345678901234567890 123456789012345678901234567890123456789012 345678901234567890123456789012345678901234 567890” | Error |
| U2U3 | 8 | "toolongemailaddress123456789012345678901234 567890123456789012345678901234567890123456 789012345678901234567890123456789012345678 901234567890123456789012345678901234567890 123456789012345678901234567890123456789012 345678901234567890123456789012345678901234567 | Error |

| | | | |
|---|---|---|---|
| | | 8901234567890123456789012345678<br>9012345678901234567890123456789012345678<br>9012345678901234567890123456789012345678<br>0123456789012345678901234567890123456789<br>0123456789012345678901234567890123456789<br>0123456789012345678901234567890123456789<br>@example.com", “4” | |
| U2U4 | 9 | "toolongemailaddress1234567890123456789012<br>3456789012345678901234567890123456789012345<br>6789012345678901234567890123456789012345678<br>9012345678901234567890123456789012345678<br>9012345678901234567890123456789012345678<br>9012345678901234567890123456789012345678<br>9012345678901234567890123456789012345678<br>9012345678901234567890123456789012345678<br>@example.com",<br>"Loremipsumdolorsitametconsecteturadipiscingelit<br>Seddoeiusmodtemporincididuntutlaboreetdolorem<br>agnaaliquaUtenimadminimveniamquisnostrudexer<br>citationullamcolaborisnisiutaliquipexeacommodo<br>consequat" | Error |

## Boundary Value Analysis

| Equivalence Classes | | Test # | Test Input | Expected Input |
|---|---|---|---|---|
| E1E2 | 3<=email_address<=254<br>4<=password<=100 | 1 | “esm@gmail.com”, “Ye12” | Valid |
| | | 2 | “esm@gmail.com”, “u”*100 | Valid |
| | | 3 | “esm@gmail.com”, “yamur” | Valid |
| | | 4 | “esm@gmail.com”, “u”*99 | Valid |
| | | 5 | “esmu@gmail.com”, “esen” | Valid |
| | | 6 | “esmu@gmail.com”, “yamur” | Valid |
| | | 7 | “esmu@gmail.com”, “e”*99 | Valid |
| | | 8 | “esmu@gmail.com”, “y”*100 | Valid |
| | | 9 | “e”*253+”@gmail.com”, “esen” | Valid |
| | | 10 | “e”*253+”@gmail.com”, “yamur” | Valid |
| | | 11 | “e”*253+”@gmail.com”, “e”*99 | Valid |
| | | 12 | “e”*253+”@gmail.com”, “y”*100 | Valid |
| | | 13 | “e”*254+”@gmail.com”, “esen” | Valid |
| | | 14 | “e”*254+”@gmail.com”, “yamur” | Valid |
| | | 15 | “e”*253+”@gmail.com”, “e”*99 | Valid |
| | | 16 | “e”*253+”@gmail.com”, “y”*100 | Valid |
| U1U3 | email_address<3<br>Password<3 | 17 | “ey@gmail.com” , “ ey” | Error |
| U1U4 | email_address<3<br>password>100 | 18 | “ey@gmail.com” , “e”*101 | Error |
| U2U3 | email_address>254<br>Password<3 | 19 | “e”*255+”@gmail.com”, “ya” | Error |
| U2U4 | email_address>254<br>password>100 | 20 | “e”*255+”@gmail.com”, “y”*101 | Error |
| E1U3 | 3<=email_address<=254<br>Password<3 | 21 | “esm@gmail.com”, “ya” | Error |

| | | | | |
|---|---|---|---|---|
| | | 22 | “esmu@gmail.com”, “ya” | Error |
| | | 23 | “e”*253+”@gmail.com”, “ya” | Error |
| | | 24 | “e”*254+”@gmail.com”, “ya” | Error |
| E1U4 | 3<=email_address<=254<br>password>100 | 25 | “esm@gmail.com”, “y”*101 | Error |
| | | 26 | “esmu@gmail.com”, “y”*101 | Error |
| | | 27 | “e”*253+”@gmail.com”,<br>“y”*101 | Error |
| | | 28 | “e”*254+”@gmail.com”,<br>“y”*101 | Error |
| | | | | |
| E2U1 | 4<= password<=100<br>3>email_address | 29 | “ya@gmail.com”, “yamu” | Error |
| | | 30 | “ya@gmail.com”, “yamur” | Error |
| | | 31 | “ya@gmail.com”, “y”*99 | Error |
| | | 32 | “ya@gmail.com”, “y”*100 | Error |
| E2U2 | 4<= password<=100<br>email_address>254 | 33 | “y”*255+”@gmail.com”, “yamu” | Error |
| | | 34 | “y”*255+”@gmail.com”,<br>“yamur” | Error |
| | | 35 | “y”*255+”@gmail.com”, “y”*99 | Error |
| | | 36 | “y”*255+”@gmail.com”,<br>“y”*100 | Error |

2. **Search Functionality**

The test covered: • Valid and invalid search queries based on character length.
• Boundary value analysis fort he search input.

| Equivalence Classes | |
|---|---|
| E1 | 0<=search<=4094 |
| U1 | Search>4094 |

### Equivalence class

| Equivalence Classes | Test # | Test Input | Expected Output |
|---|---|---|---|
| E1 | 1 | “ruj” | Valid |
| U1 | 2 | “u”*4095 | error |

### Boundary value analysis

| Equivalence Classes | | Test # | Test Input | Expected Output |
|---|---|---|---|---|
| E1 | 0<=search<=4094 | 1 | “ ” | Valid |
| | | 2 | “a” | Valid |
| | | 3 | “ruj” | Valid |
| | | 4 | “a” *4093 | Valid |
| | | 5 | “a” * 4094 | Valid |
| U1 | Search>4094 | 6 | “a” * 4095 | Error |

3. **Discount Code Functionality**
   The test covered: • Valid and invalid discount codes based on character length.
   • Boundary value analysis fort he discount code input.

| Equivalence Classes | |
|---|---|
| E1 | 5<= Discount Code <=20 |
| U1 | Discount Code >20 |
| U2 | Discount Code<5 |

## Equivalence Class

| Equivalence Classes | Test # | Test Input | Expected Output |
|---|---|---|---|
| E1 | 1 | “YASAR25” | Valid |
| U1 | 2 | “YASARUNIVERSITELILEREOZEL25” | error |
| U2 | 3 | Y2 | error |

## Boundary Value Analysis

| Equivalence C | | Test # | Test Input | Expected Output |
|---|---|---|---|---|
| E1 | 5<=DiscountCode<=20 | 1 | “ YASAR” | Valid |
| | | 2 | “YASAR1” | Valid |
| | | 3 | “YASARUNIVERSITEOZEL” | Valid |
| | | 4 | “YASARUNIVERSITEOZEL1” | Valid |
| U1 | DiscountCode>20 | 5 | “YASARUNIVERSITEOZEL10” | Error |
| U2 | Discount Code<5 | 6 | “YASA” | Error |

4. **Shopping Cart Functionality**
   The test covered: • Valid and invalid quantities based on character length.

   • Boundary value analysis fort he quantity input.

| Equivalence Classes | |
|---|---|
| E1 | 0<=lenght(cart)<100 |
| U1 | Lenght < 0 |
| U2 | Lenght >100 |

**Equivalence Classes**

| Equivalence Classes | Test # | Test Input | Expected Output |
|---|---|---|---|
| E1 | 1 | “123” | Valid |
| E1 | 2 | “-5” | error |
| E1 | 3 | “♣” | error |
| E1 | 4 | “abcde” | error |
| U1 | 5 | | error |
| U2 | 6 | “a”*101 | error |

**Boundary value analysis**

| Equivalence Classes | | Test # | Test ınput | Expected output |
|---|---|---|---|---|
| E1 | 0<=lenght(cart)<100 | 1 | “ “ | Valid |

| | | | | |
|---|---|---|---|---|
| | | 2 | “1” | Valid |
| | | 3 | “1”*101 | Valid |
| | | 4 | “-5” | error |
| | | 5 | “-abc” | error |
| | | 6 | “-◆” | error |
| | | 7 | “-◆abc” | error |
| | | 8 | “-◆abc1” | error |
| | | 9 | “abc” | error |
| U1 | Lenght < 0 | 10 | | error |
| U2 | Lenght >100 | 11 | ‘2’*101 | error |

5. **Discount Functionality**
   This decison table checks if the custmer bought more than 3 products, are they lipstick or eyeshadow, if the customer ‘kiko me’ member or not and if the customer have a discount code. According to the this conditions discounts applied.

| Conditions | R1 | R2 | R3 | R4 | R5 | R6 | R7 | R8 | R9 | R10 | R11 | R12 | R13 | R14 | R15 | R16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bought 3 or more products | T | T | T | T | T | T | T | T | F | F | F | F | F | F | F | F |
| Kiko Me Member | T | T | T | T | F | F | F | F | T | T | T | T | F | F | F | F |
| Lipstick or eyeshadow | T | T | F | F | T | T | F | F | T | T | F | F | T | T | F | F |
| Discount Coupon | T | F | T | F | T | F | T | F | T | F | T | F | T | F | T | F |
| Actions | | | | | | | | | | | | | | | | |
| 25% Discount | ✓ | ✓ | | | ✓ | ✓ | | | | | | | | | | |
| 10% Discount | | | ✓ | | ✓ | | ✓ | | ✓ | | ✓ | | ✓ | | ✓ | |
| 5% Discount | ✓ | ✓ | ✓ | ✓ | | | | | ✓ | ✓ | ✓ | ✓ | | | | |
| Total Discount | 30 % | 30 % | 15 % | 5 % | 25 % | 25 % | 10 % | 0 % | 15 % | 5 % | 15 % | 5 % | 10 % | 0 % | 10 % | 0 % |

## Use Case Scenario

**Use Case 1:** Where Is My Order

**Description :** Allows users to check the status of their orders without logging in or being a member.

**Actor:** User

**Precondition**: User has placed an order on the website and has received an order tracking number.

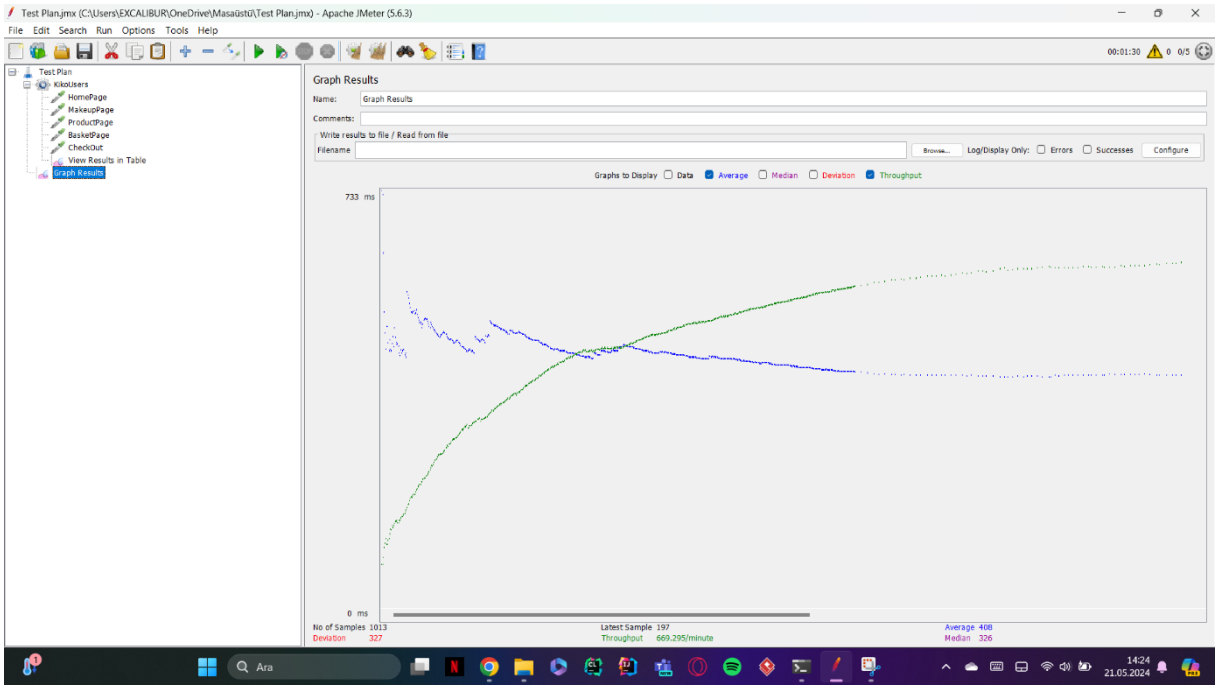
**Postcondition:** User gets information about the order status.

| | |
|---|---|
| **Main Flow** | 1-The user navigates to the website's homepage.<br>2- The user clicks on the "Order Tracking" link found on the homepage or in the menu.<br>3-The system prompts the user to enter the order number and verification details such as email address or phone number.<br>4-The user enters the order number and verification details.<br>5-The system verifies the entered information and displays the current status of the order.<br>6-The user views the order status. |
| **Alternative Flow 1: Invalid Order Number** | 1-The system verifies the entered information.<br>2-If the information is incorrect, the system shows an error message and prompts the user to re-enter the details. |
| **Exceptional Flow 1: System Error** | 1-The user enters order details.<br>2-The system displays an error message due to a system issue.<br>3-The user may refresh the page or try again later. |

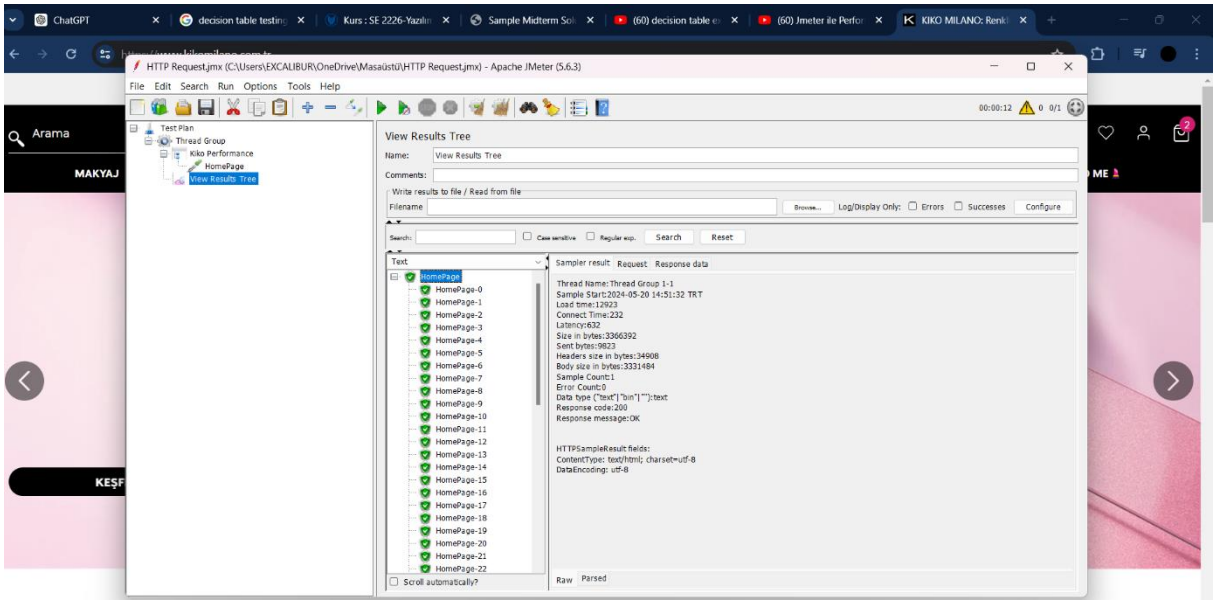
**Load Test:**

We performed load testing using JMeter. In this test, we simulated 100 users logging in, browsing products, and adding items to their cart.

We generated average response time and throughput graphs to analyze the performance under load. Below are the graphs that illustrate the results of our load testing.




We tested the load time of the homepage to measure how long it takes for the entire page, including all images and elements, to fully load. The average load time for the homepage was 12923 seconds.




## 2. Deviations from planned testing

**Like Button:** During our Selenium testing, we had a test case that verified if the "like" count increased when a comment on a product was liked. Initially, this test was functioning correctly**.** However, the Kiko website later removed the comments section from the product pages. As a result, our test case became invalid since it could no longer locate the comments to perform the "like" action.

**Ask a Question:** One of our Selenium tests involved testing the "Ask a Question" section under the product details. However, this section was inconsistently visible on the page, appearing at times and not at others. Due to this inconsistency, our test began to fail as it could not locate the "Ask a Question" section on the page**.**

## 3. Test completion evaluation

The test completion evaluation assesses the extent to which the testing objectives were met. In our project, we utilized Selenium IDE, JMeter, and Blackbox testing methods to comprehensively evaluate the functionality, performance, and user experience of the Kiko Milano website.

**Selenium IDE:**

Total Tests Conducted: 12

Tests Passed: 6

Tests Failed: 6

**JMeter:**

Load Testing: Simulated 100 users

Average Response Time: 12923 seconds

Throughput: 669.295/minute

**Blackbox Testing:**

Functional Scenarios Covered: Login functionality, shopping cart operations, discount code validation, etc.

Boundary Value Analysis and Equivalence Class Partitioning: Implemented for critical input fields.

Overall, the test completion evaluation reveals that while a significant number of test cases passed, there were notable failures that highlighted critical issues in the website's handling of invalid inputs and error messages. The load testing indicated potential performance bottlenecks under high user load.

## 4. Factors that blocked progress

One big problem we faced during testing was not being able to access the source code from Kiko. We tried reaching out and asking for access to the code repository multiple times, but we didn't get any response from Kiko's team. So, without access to the source code, we couldn't do thorough testing. We were limited in what kinds of analysis and checks we could do. As a result, we couldn't achieve the level of testing coverage we wanted.

## 5. Test Measures:

The test measures collected during the testing process include:

Test Cases Executed: A total of 12 specific tests were performed using Selenium IDE, along with multiple tests using Blackbox testing methods.

Defects Identified: Several defects were identified during the testing, such as:

Incorrect display of stock information (dontShowStockOutProducts test)

Comment functionality failure (makeComment test)

Registration with invalid email (registerWithInvalidEmail test)

Issues with shopping cart product quantities (basketNegNum and editingProductNumberWithString tests)

Order status error handling (orderStatu test)

Test Coverage: The tests covered various functionalities including login, registration, shopping cart management, discount code application, order tracking, and search functionalities.

Load Testing: The resource consumption was measured during load testing using JMeter. Key metrics included average response time and throughput.

## 6. Test deliverables: We have included all test documentation, including test cases and descriptions, in response to the first question.

## 7. Lessons learned:

Throughout the testing process, we learned several valuable lessons:

1) Automating repetitive test cases saved significant time and allowed us to focus on more complex testing scenarios.
2) Early and detailed planning of test cases and scenarios helped in identifying potential issues early in the development cycle.
3) Flexibility and adaptability were essential when facing unexpected changes, such as the removal of features from the application.

Overall, this project will guide us in future testing projects to improve our processes and outcomes

**21070006039 – Sultan Esen Murat**

**21070006017 – Yağmur Sabırlı**