

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.

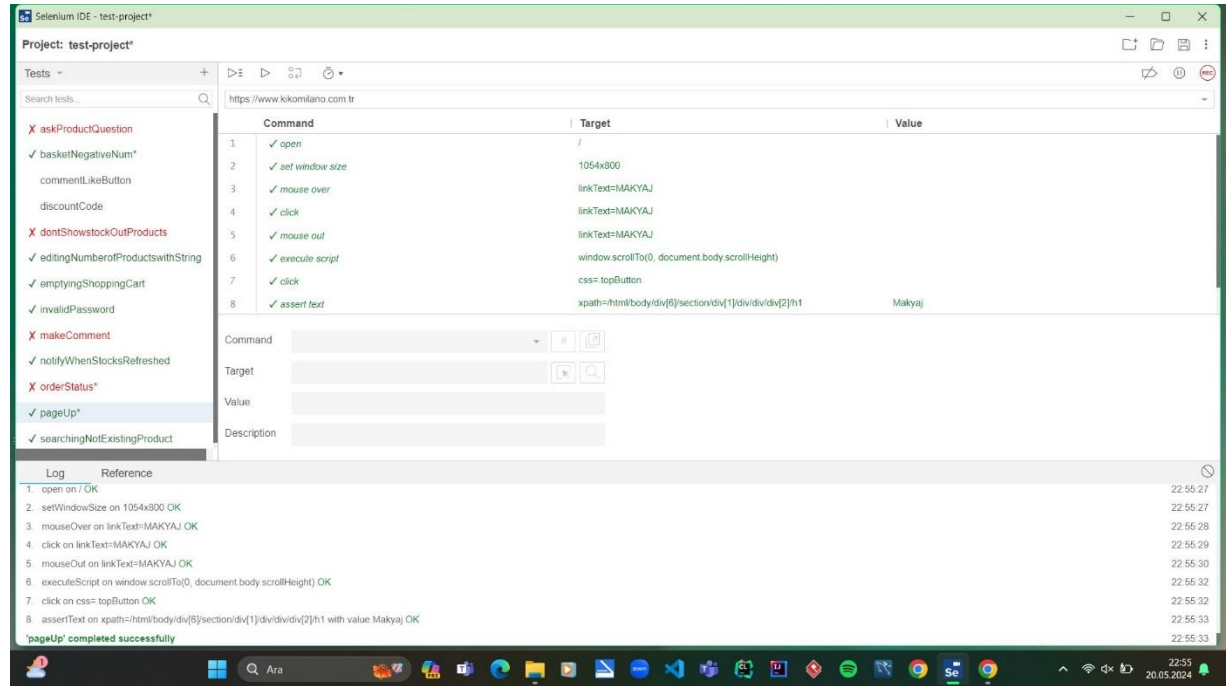
The screenshot displays the Selenium IDE interface for a project named 'test-project'. The URL is set to 'https://www.kikomilano.com.tr'. The test suite includes several steps, with the 'invalidPassword' test highlighted. The test steps are as follows:

Step	Command	Target	Value
5	type	xpath=html/body/div[6]/section/pz-tab-content/pz-tab-content-item[2]/pz-form/form/div[1]/pz-label/pz-input/input	yagmursabir35@gmail.com
6	click	xpath=html/body/div[6]/section/pz-tab-content/pz-tab-content-item[2]/pz-form/form/div[2]/pz-label/pz-input/input	
7	type	xpath=html/body/div[6]/section/pz-tab-content/pz-tab-content-item[2]/pz-form/form/div[2]/pz-label/pz-input/input	123456789
8	click	css=-w-full:nth-child(5)	
9	assert text	xpath=//*[id="LoginForm"]/form/label	E-posta veya şifre hatalı.

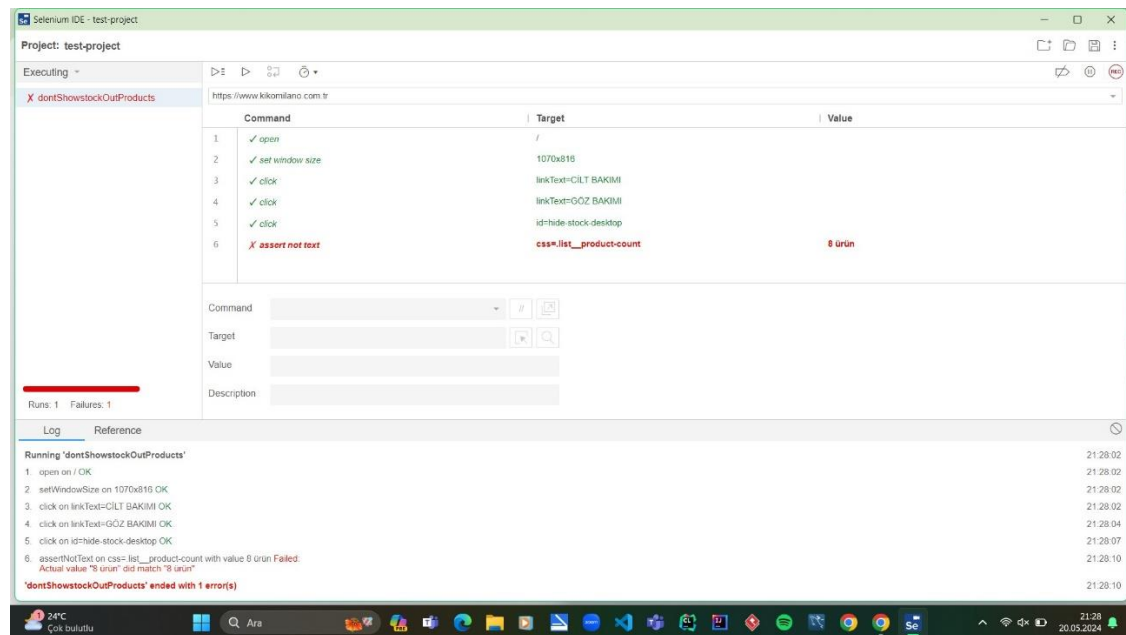
The log shows the execution of the 'invalidPassword' test, which completed successfully. The log entries are as follows:

```
Running 'invalidPassword'
1. open on / OK
2. setWindowSize on 1070x816 OK
3. click on css=-loginClick^n > svg OK
4. click on xpath=html/body/div[6]/section/pz-tab-content/pz-tab-content-item[2]/pz-form/form/div[1]/pz-label/pz-input/input OK
5. type on xpath=html/body/div[6]/section/pz-tab-content/pz-tab-content-item[2]/pz-form/form/div[1]/pz-label/pz-input/input with value yagmursabir35@gmail.com OK
6. click on xpath=html/body/div[6]/section/pz-tab-content/pz-tab-content-item[2]/pz-form/form/div[2]/pz-label/pz-input/input OK
7. type on xpath=html/body/div[6]/section/pz-tab-content/pz-tab-content-item[2]/pz-form/form/div[2]/pz-label/pz-input/input with value 123456789 OK
8. click on css=-w-full:nth-child(5) OK
9. assertText on xpath=//*[id="LoginForm"]/form/label with value E-posta veya şifre hatalı. OK
'invalidPassword' completed successfully
```

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.

The screenshot shows the Selenium IDE interface for a test project named 'test-project'. The test script is titled 'notifyWhenStocksRefreshed' and is located at 'https://www.kikomilano.com.tr'. The script consists of five commands: 'open', 'set window size', 'click', 'click', and 'type'. The 'click' commands target the 'loginClickFn > svg' and the 'xpath=//pz-form[@id=LoginForm]/form/div/pz-label/pz-input/input' respectively. The 'type' command targets the 'xpath=//pz-form[@id=LoginForm]/form/div/pz-label/pz-input/input' with the value 'yagnursabir135@gmail.com'. The execution log shows that the test completed successfully at 21:40:14.

Command	Target	Value
1. ✓ open	/	
2. ✓ set window size	1070x816	
3. ✓ click	css= loginClickFn > svg	
4. ✓ click	xpath=//pz-form[@id=LoginForm]/form/div/pz-label/pz-input/input	
5. ✓ type	xpath=//pz-form[@id=LoginForm]/form/div/pz-label/pz-input/input	yagnursabir135@gmail.com

Log Reference

3. click on css= loginClickFn > svg OK 21:39:45

4. click on xpath=//pz-form[@id=LoginForm]/form/div/pz-label/pz-input/input OK 21:39:48

5. type on xpath=//pz-form[@id=LoginForm]/form/div/pz-label/pz-input/input with value yagnursabir135@gmail.com OK 21:39:50

6. click on xpath=//pz-form[@id=LoginForm]/form/div/pz-label/pz-input/input OK 21:39:50

7. type on xpath=//pz-form[@id=LoginForm]/form/div/pz-label/pz-input/input with value Muhasebe77 OK 21:39:50

8. click on css= #full-nth-child(5) OK 21:39:50

9. click on linkText=GİLT BAKIMI OK 21:39:50

10. Trying to find linkText=GÖZ BAKIMI OK 21:39:51

11. click on css= listProducts:nth-child(11) pz-button__text OK 21:40:10

12. assertElementPresent on xpath=//html/body/div[6]/pz-modal/div OK 21:40:14

13. assertText on xpath=//div[@id=product-stock-alert-modal]/div/header/p with value Stok Alarmı OK 21:40:14

'notifyWhenStocksRefreshed' completed successfully 21:40:14

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.

The screenshot shows the Selenium IDE interface for a test project named 'test-project'. The test script is titled 'makeComment' and is located at 'https://www.kikomilano.com.tr'. The script consists of eight commands: 'click', 'type', 'click', 'type', 'click', 'click', 'type', and 'click'. The 'click' commands target the 'xpath=//pz-form[@id=LoginForm]/form/div/pz-label/pz-input/input', 'xpath=//pz-form[@id=LoginForm]/form/div/pz-label/pz-input/input', 'xpath=//pz-form[@id=LoginForm]/form/div/pz-label/pz-input/input', 'xpath=//pz-form[@id=LoginForm]/form/div/pz-label/pz-input/input', and 'css=#w-full:nth-child(5)'. The 'type' commands target the 'xpath=//pz-form[@id=LoginForm]/form/div/pz-label/pz-input/input' with the values 'yagnursabir135@gmail.com' and 'Muhasebe77'. The execution log shows that the test failed at 22:01:43 due to an 'assertElementNotPresent' error.

Command	Target	Value
4. ✓ click	xpath=//pz-form[@id=LoginForm]/form/div/pz-label/pz-input/input	
5. ✓ type	xpath=//pz-form[@id=LoginForm]/form/div/pz-label/pz-input/input	yagnursabir135@gmail.com
6. ✓ click	xpath=//pz-form[@id=LoginForm]/form/div/pz-label/pz-input/input	
7. ✓ type	xpath=//pz-form[@id=LoginForm]/form/div/pz-label/pz-input/input	Muhasebe77
8. ✓ click	css=#w-full:nth-child(5)	

Log Reference

9. click on css= loginClickFn > svg OK 22:00:55

10. click on linkText=Değerlendirmelerim OK 22:01:01

11. click on css= reviews-product:nth-child(1) showEva OK 22:01:06

12. click on xpath=//form[@id=evaluationsForm]/pz-label/pz-input/input OK 22:01:11

13. type on xpath=//html/body/div[6]/div[5]/div[2]/div[2]/pz-modal[1]/div/div[1]/form/pz-label/pz-input/input with value beğendim OK 22:01:15

14. click on xpath=//ul[@id=evastars]/li[4] OK 22:01:19

15. click on css=#evastarsCargo > star:nth-child(4) > fa OK 22:01:23

16. click on name=showName OK 22:01:27

17. click on id=kıkkıReview OK 22:01:31

18. click on id=aydinlatmaReview OK 22:01:35

19. click on css=button:nth-child(11) OK 22:01:39

20. assertElementNotPresent on xpath=//div[@id=evaluationsForm]/button Failed: Element with locator xpath=//div[@id=evaluationsForm]/button was found 22:01:43

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.

Selenium IDE - test-project*

Project: test-project*

Tests

Search tests...

askProductQuestion

commentLikeButton

discountCode

✗ dontShowStockOutProducts

✓ editingNumberOfProductsWithString

emptyingShoppingCart

✓ invalidPassword

makeComment

✓ notifyWhenStocksRefreshed*

✓ searchingNotExistingProduct*

https://www.kikomilano.com.tr

	Command	Target	Value
2	✓ set window size	1069x816	
3	✓ click	id=pz-form-input-AutocompleteInput	
4	✓ type	id=pz-form-input-AutocompleteInput	invalidSearch
5	✓ send keys	id=pz-form-input-AutocompleteInput	\$(KEY_ENTER)
6	✓ click	css=-empty	
7	✓ assert text	css=list__products-empty-content	Sonuç bulunamadı: \nAna Sayfa

Command

Target

Value

Description

Log Reference

Running 'searchingNotExistingProduct'

1. open on / OK 21:42:29

2. setWindowSize on 1069x816 with value OK 21:42:29

3. click on id=pz-form-input-AutocompleteInput OK 21:42:29

4. type on id=pz-form-input-AutocompleteInput with value invalidSearch OK 21:42:32

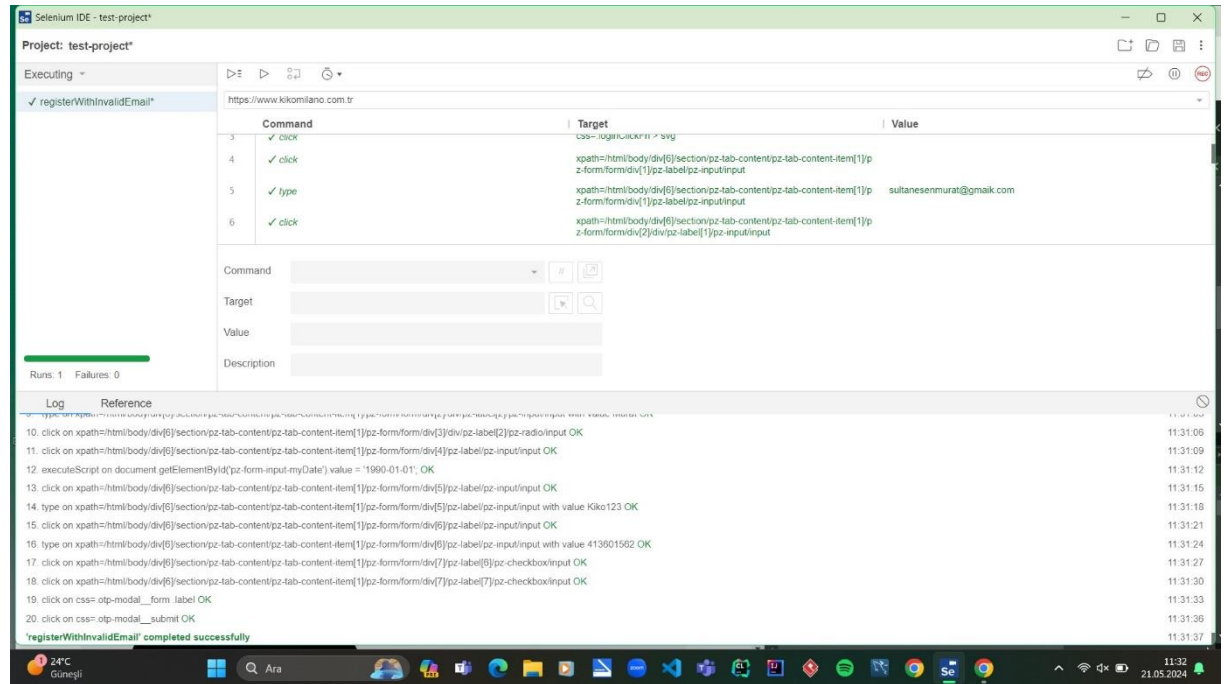
5. sendKeys on id=pz-form-input-AutocompleteInput with value \$(KEY_ENTER) OK 21:42:33

6. click on css=-empty OK 21:42:33

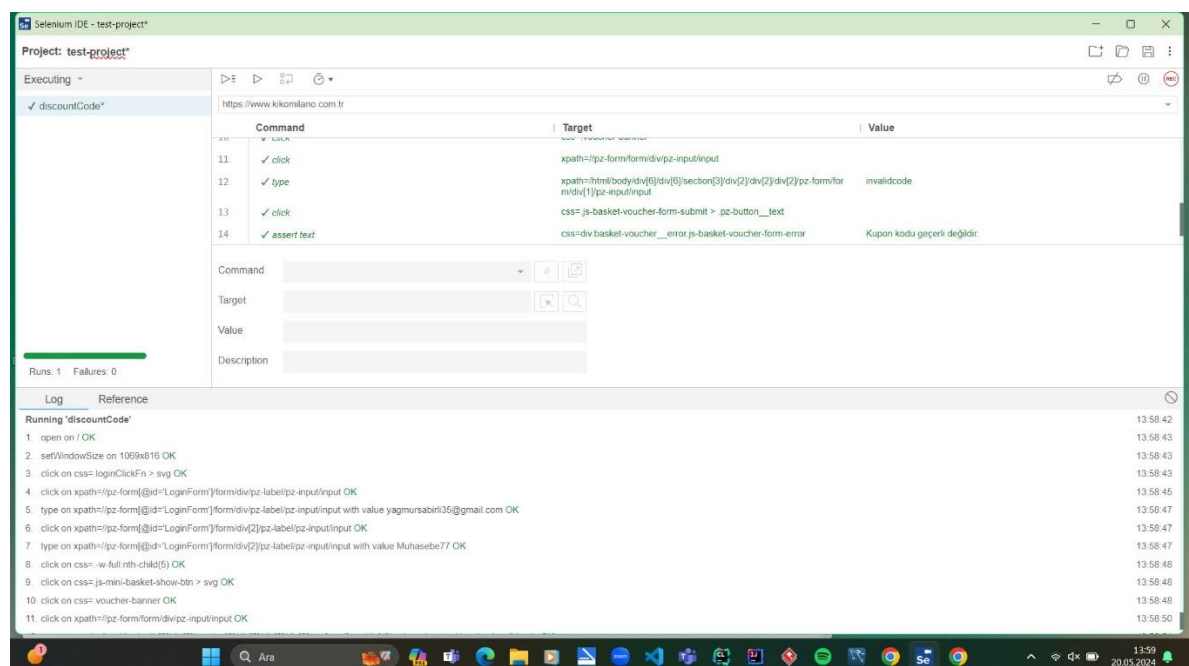
7. assertText on css=list__products-empty-content with value Sonuç bulunamadı: \nAna Sayfa OK 21:42:34

'searchingNotExistingProduct' completed successfully 21:42:34

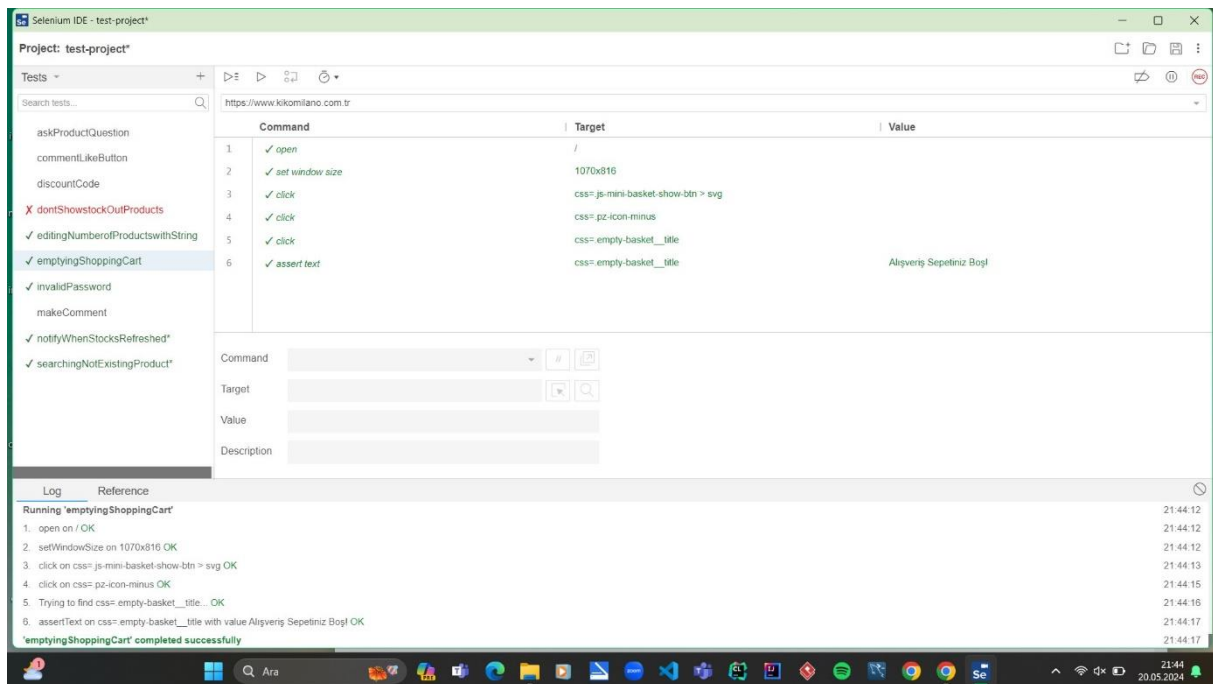
7. **registerWithInvalidEmail:** This test attempts to register a new user account using an invalid email format. It checks if the website validates the email and displays appropriate error message. This test should fail but, it did not. We could be able to register with an invalid email.



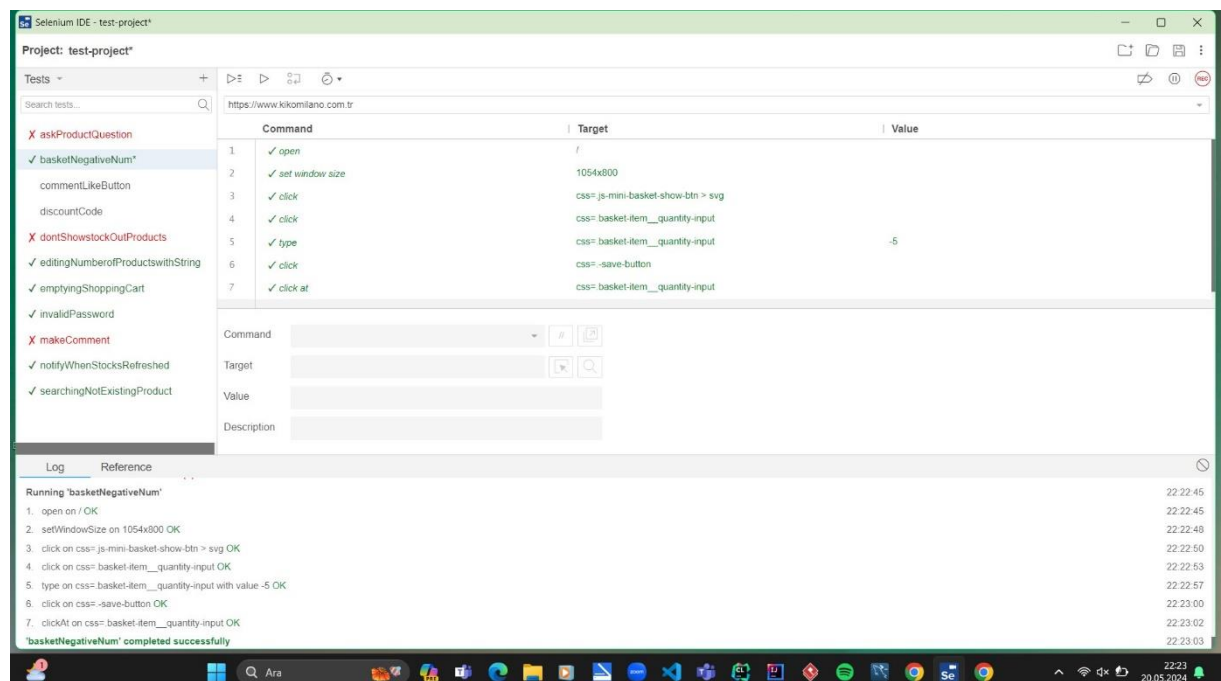
8. **discountCodeTest:** This test enters an invalid discount code during checkout and checks if the website displays an error message.



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 entering a negative number. It checks if the website handles invalid input correctly. This test should fail. A negative number of product in the shopping cart cannot exist.



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 fail. A non-numeric string as product quantity in the shopping cart cannot exist.

The screenshot displays the Selenium IDE interface for a test project named 'test-project'. The test being executed is 'editingNumberOfProductsWithString' at the URL 'https://www.kikomilano.com.tr'. The test consists of seven steps, all of which are marked as successful (green checkmarks). The steps are as follows:

Step	Command	Target	Value
1	open	/	
2	set window size	1073x816	
3	click	id=Path_7	
4	click	css=basket-item__quantity-input	
5	type	css=basket-item__quantity-input	abc
6	click	css=pz-icon-check	
7	click at	css=pz-button__icon pz-icon-check	

Below the command table, there are input fields for 'Command', 'Target', 'Value', and 'Description'. The 'Log' tab at the bottom shows the execution details for each step, confirming that all steps completed successfully. The final status is 'editingNumberOfProductsWithString' completed successfully.

Runs: 1 Failures: 0

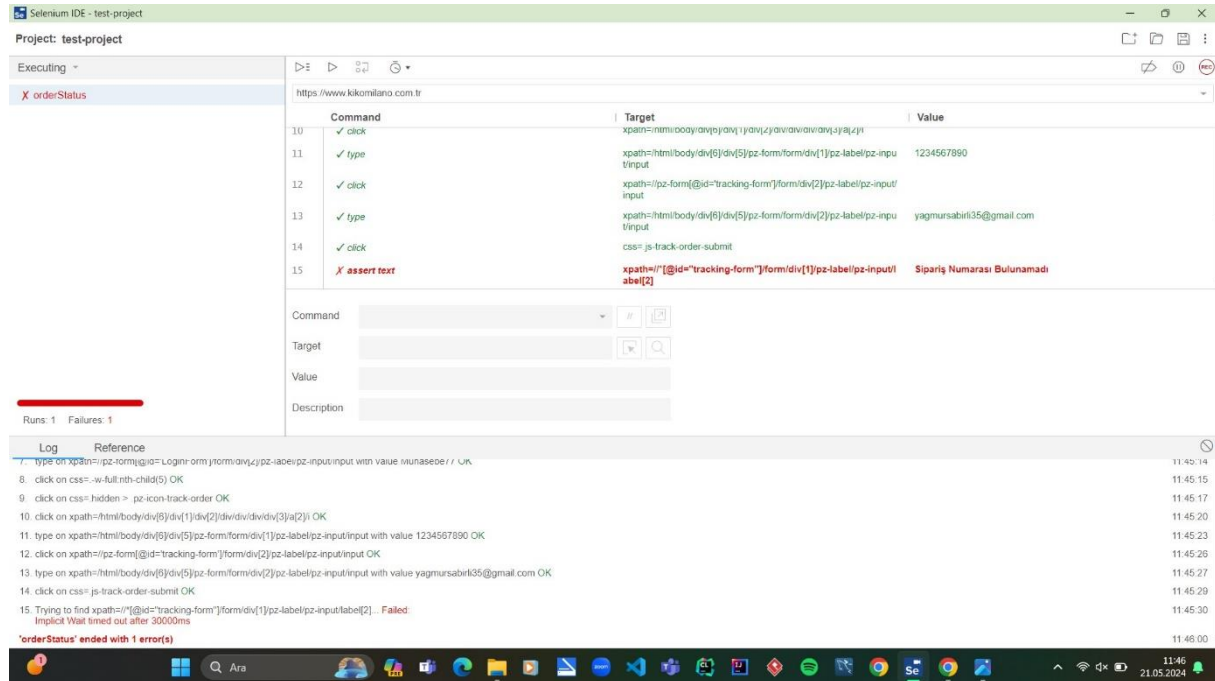
Log Reference

Running 'editingNumberOfProductsWithString'

- 1. open on / OK 21:31:15
- 2. setWindowSize on 1073x816 OK 21:31:16
- 3. click on id=Path_7 OK 21:31:16
- 4. click on css=basket-item__quantity-input OK 21:31:16
- 5. type on css=basket-item__quantity-input with value abc OK 21:31:18
- 6. click on css=pz-icon-check OK 21:31:19
- 7. clickAt on css=pz-button__icon pz-icon-check OK 21:31:19

'editingNumberOfProductsWithString' completed successfully 21:31:20

12. **orderStatus:** This test enters an invalid order number and checks if the website displays an error message. The test should not 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.

Boundary Value Analysis

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

		22	" esmu@gmail.com ", "ya"	Error
		23	" @gmail.com ", "ya"	Error
		24	" @gmail.com ", "ya"	Error
E1U4	3<=email_address<=254 password>100	25	" esm@gmail.com ", "y"*101	Error
		26	" esmu@gmail.com ", "y"*101	Error
		27	" @gmail.com ", "y"*101	Error
		28	" @gmail.com ", "y"*101	Error
E2U1	4<= password<=100 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 email_address>254	33	" @gmail.com ", "yamu"	Error
		34	" @gmail.com ", "yamur"	Error
		35	" @gmail.com ", "y"*99	Error
		36	" @gmail.com ", "y"*100	Error

2. Search Functionality

The test covered: • Valid and invalid search queries based on character length.

- Boundary value analysis for the 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 for the 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 for the quantity input.

Equivalence Classes	
E1	$0 \leq \text{length}(\text{cart}) < 100$
U1	$\text{Length} < 0$
U2	$\text{Length} > 100$

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 input	Expected output
E1	$0 \leq \text{length}(\text{cart}) < 100$	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 decision table checks if the customer bought more than 3 products, are they lipstick or eyeshadow, if the customer 'kiko me' member or not and if the customer has a discount code. According to these conditions, discounts are 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.

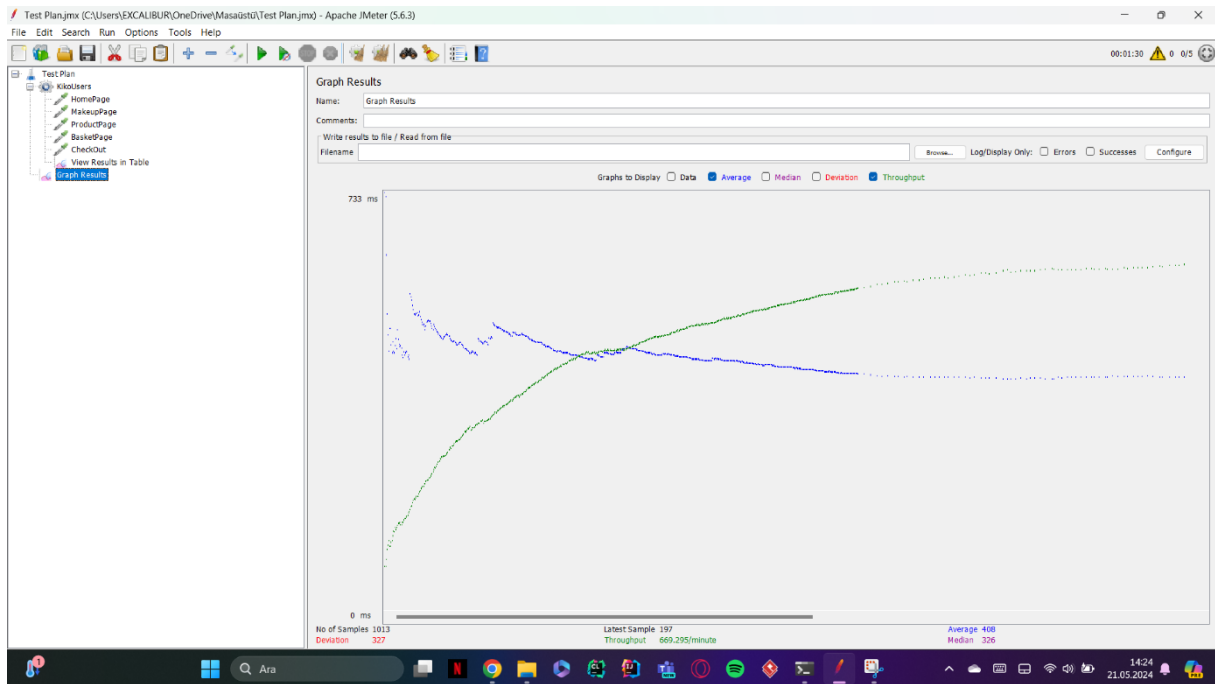
Postcondition: User gets information about the order status.

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

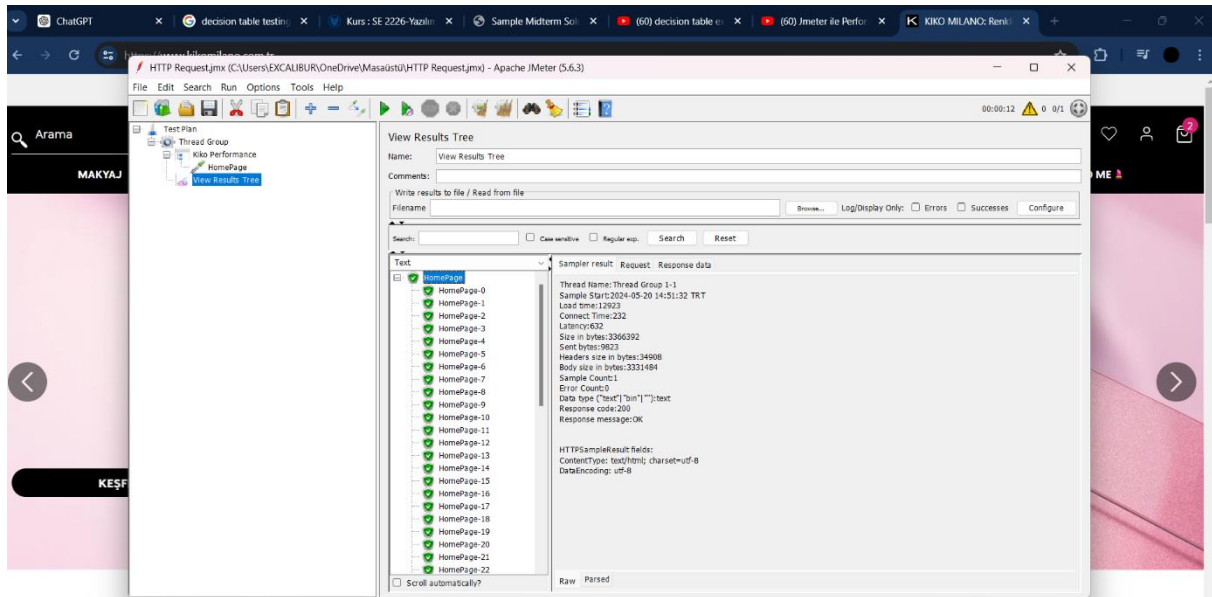
Load Test:

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