

Додаток А

Графічний матеріал кваліфікаційної роботи

ГЮИК.501610.008

(позначення документу)

ЗАТВЕРДЖЕНО
ГЮИК.501610.008 – ЛЗ

Дослідження процесів контролю якості взаємодії користувачів з веб-додатком
інтернет-магазину

Графічний матеріал

ГЮИК. 501610.008 – ЛЗ

ЛИСТІВ 24

2021 р.

Міністерство освіти і науки України
Харківський Національний Університет Радіоелектроніки

«ЗАТВЕРДЖУЮ»
керівник кваліфікаційної роботи
проф. Мінухін С.В.

Дослідження процесів контролю якості взаємодії користувачів з веб-додатком
інтернет-магазину

Графічний матеріал

ЛИСТ ЗАТВЕРДЖЕННЯ

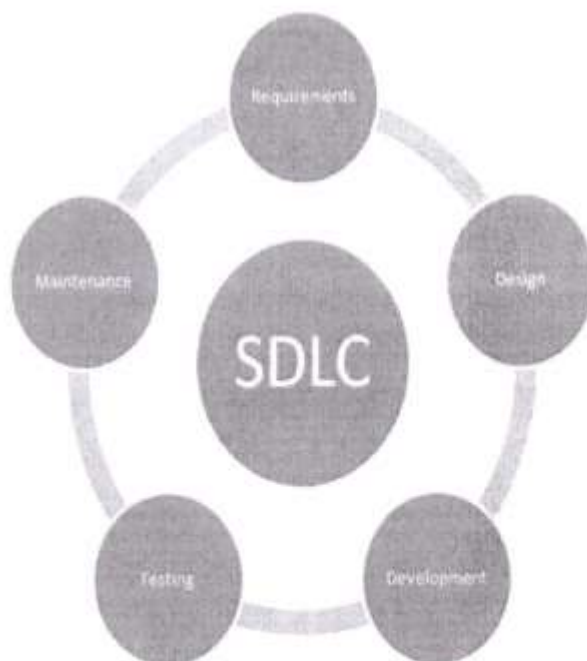
ГЮИК.501610.008 – ЛЗ

РОЗРОБИВ:
ст. гр. СПРМ-20-1
Мороз М.Ю.

2021 р.

ЖИТТЄВИЙ ЦИКЛ РОЗРОБКИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

1. Збір та аналіз вимог
2. Дизайн
3. Розробка
4. Тестування
5. Підтримка



| | | | | |
|------------------|----------------|-----------|--|------------------|
| <i>Розроб.</i> | Мороз М.Ю. | <i>МЮ</i> | Дослідження процесів контролю якості взаємодії користувачів з веб-додатком інтернет-магазину | |
| <i>Перевір.</i> | Мінухін С.В. | <i>СВ</i> | | |
| <i>Н. Контр.</i> | Мінухін С.В. | <i>СВ</i> | | |
| | | | <i>СПРМ-20-1</i> | <i>Лист 1</i> |
| <i>Затверд.</i> | Гребеннік І.В. | | <i>СТ</i> | <i>Листів 22</i> |

МОДЕЛЬ ЯКОСТІ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ



| | | | | |
|-----------|----------------|----------------|--|-----------|
| Розроб. | Мороз М.Ю. | <i>Мороз</i> | Дослідження процесів контролю якості взаємодії користувачів з веб-додатком інтернет-магазину | |
| Перевір. | Мінухін С.В. | <i>Мінухін</i> | | |
| Н. Контр. | Мінухін С.В. | <i>Мінухін</i> | | |
| | | | СПРМ-20-1 | Лист 2 |
| Затверд. | Гребеннік І.В. | | СТ | Листів 22 |

ПРОЦЕС ЗАБЕЗПЕЧЕННЯ ЯКОСТІ



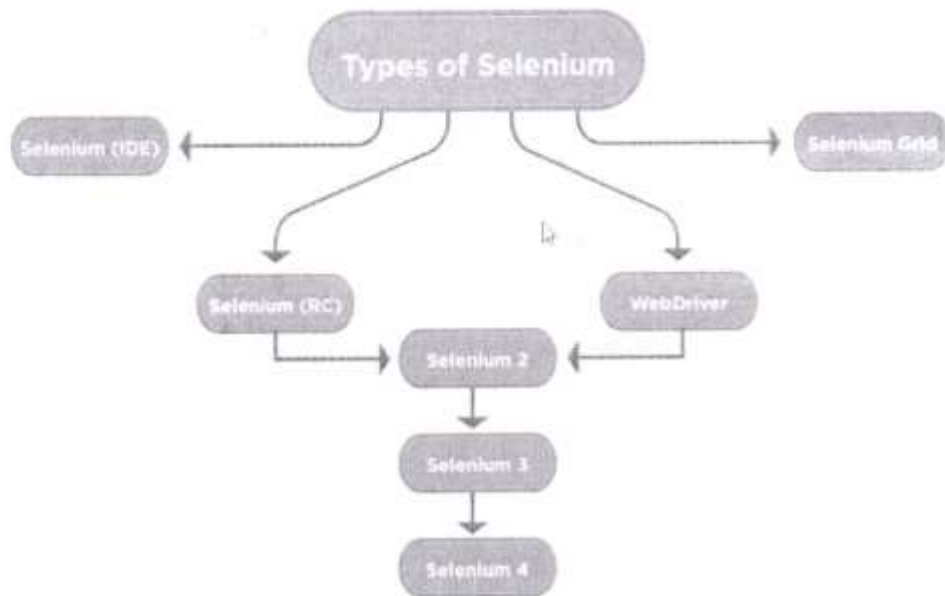
| | | | | | |
|------------------|----------------|-----------|--|--|------------------|
| <i>Розроб.</i> | Мороз М.Ю. | <i>МЮ</i> | | Дослідження процесів контролю якості взаємодії користувачів з веб-додатком інтернет-магазину | |
| <i>Перевір.</i> | Мінухін С.В. | <i>СВ</i> | | | |
| <i>Н. Коитр.</i> | Мінухін С.В. | <i>СВ</i> | | | |
| | | | | <i>СПРМ-20-1</i> | <i>Лист 3</i> |
| <i>Затверд.</i> | Гребеннік І.В. | | | <i>СТ</i> | <i>Листів 22</i> |

СТРУКТУРА ВЕБ-ДОДАТКУ



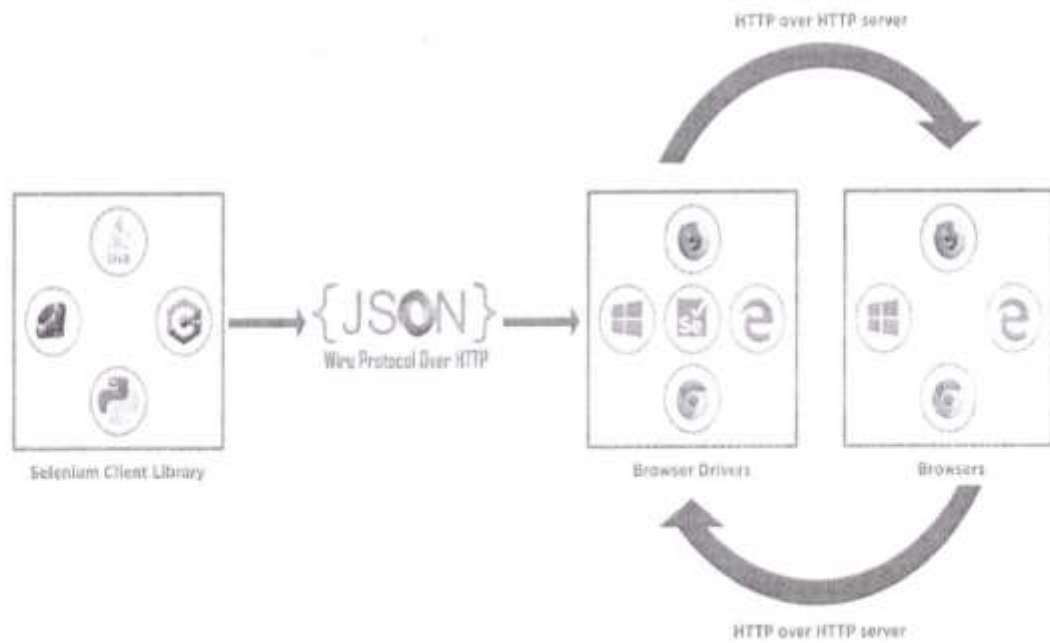
| | | | | |
|-----------|----------------|-----------|--|-----------|
| Розроб. | Мороз М.Ю. | <i>МЮ</i> | Дослідження процесів контролю якості взаємодії користувачів з веб-додатком інтернет-магазину | |
| Перевір. | Мінухін С.В. | <i>СВ</i> | | |
| Н. Коитр. | Мінухін С.В. | <i>СВ</i> | | |
| | | | СПРМ-20-1 | Лист 4 |
| Затверд. | Гребеннік І.В. | | СТ | Листів 22 |

КОМПОНЕНТИ SELENIUM SUITE



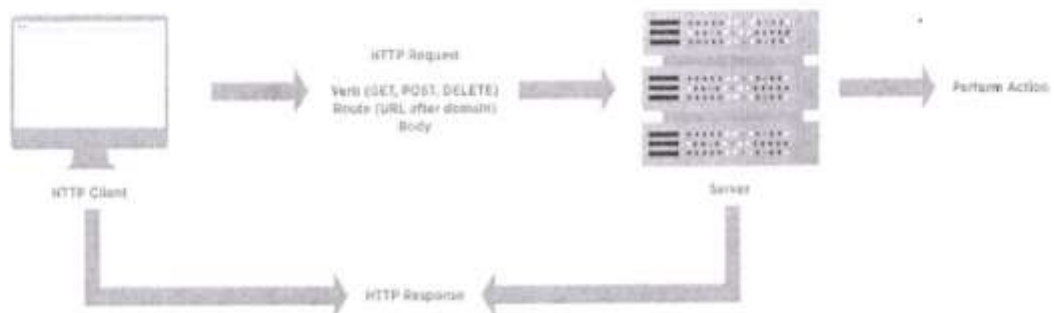
| | | | | | |
|-----------|----------------|--------------|--|--|-----------|
| Розроб. | Мороз М.Ю. | <i>M.Yu.</i> | | Дослідження процесів контролю якості взаємодії користувачів з веб-додатком інтернет-магазину | |
| Перевір. | Мінухін С.В. | <i>S.V.</i> | | | |
| Н. Контр. | Мінухін С.В. | <i>S.V.</i> | | | |
| | | | | СПРМ-20-1 | Лист 5 |
| Затверд. | Гребеннік І.В. | | | СТ | Листів 22 |

КОМПОНЕНТИ SELENIUM WEBDRIVER



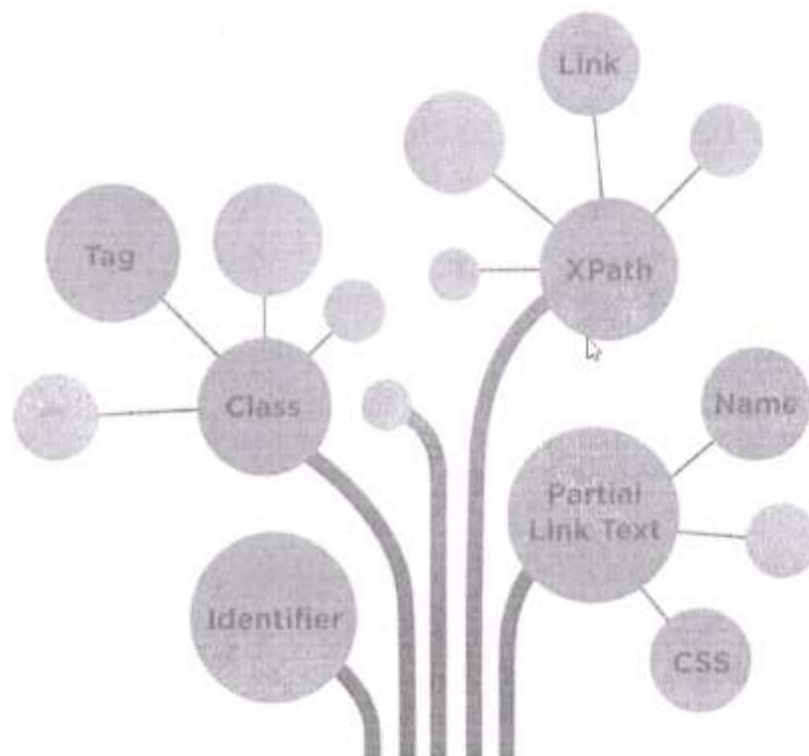
| | | | | |
|-----------|----------------|--------------------|--|-----------|
| Розроб. | Мороз М.Ю. | <i>[Signature]</i> | Дослідження процесів контролю якості взаємодії користувачів з веб-додатком інтернет-магазину | |
| Перевір. | Мінухін С.В. | <i>[Signature]</i> | | |
| Н. Коитр. | Мінухін С.В. | <i>[Signature]</i> | | |
| | | | СПРМ-20-1 | Лист 6 |
| Затверд. | Гребеннік І.В. | | СТ | Листів 22 |

РОБОЧИЙ ПРОЦЕС SELENIUM WEBDRIVER



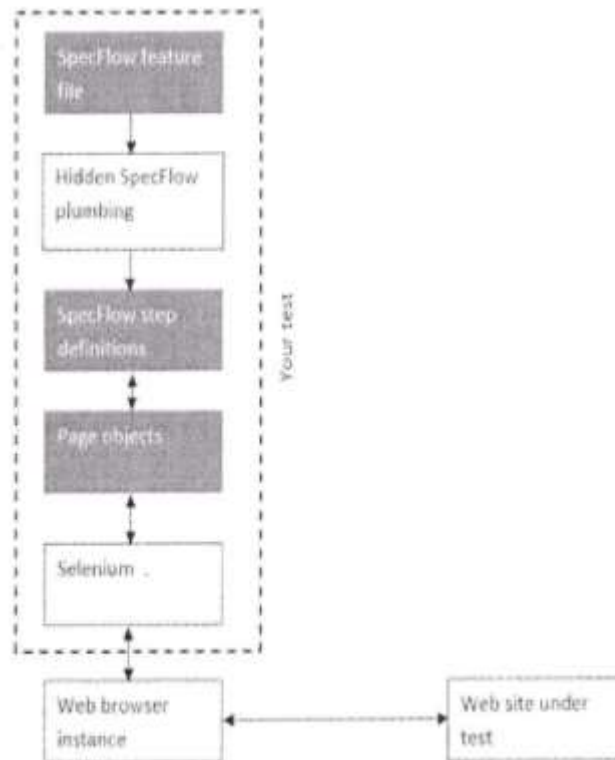
| | | | | | |
|-----------|----------------|--------------------|--|--|-----------|
| Розроб. | Мороз М.Ю. | <i>[Signature]</i> | | Дослідження процесів контролю якості взаємодії користувачів з веб-додатком інтернет-магазину | |
| Перевір. | Мінухін С.В. | <i>[Signature]</i> | | | |
| Н. Коитр. | Мінухін С.В. | <i>[Signature]</i> | | | |
| | | | | СПРМ-20-1 | Лист 7 |
| Затверд. | Гребеннік І.В. | | | СТ | Листів 22 |

ТИПИ ЛОКАТОРІВ



| | | | | |
|-----------|----------------|--------------------|--|-----------|
| Розроб. | Мороз М.Ю. | <i>[Signature]</i> | Дослідження процесів контролю якості взаємодії користувачів з веб-додатком інтернет-магазину | |
| Перевір. | Мінухін С.В. | <i>[Signature]</i> | | |
| Н. Коитр. | Мінухін С.В. | <i>[Signature]</i> | | |
| | | | СПРМ-20-1 | Лист 8 |
| Затверд. | Гребеннік І.В. | | СТ | Листів 22 |

ВЗАЄМОДІЯ SPECFLOW ТА BDD З ВЕБ-ДОДАТКОМ



| | | | | |
|-----------|----------------|-------------|--|-----------|
| Розроб. | Мороз М.Ю. | <i>eluf</i> | Дослідження процесів контролю якості взаємодії користувачів з веб-додатком інтернет-магазину | |
| Перевір. | Мінухін С.В. | <i>СВ</i> | | |
| Н. Контр. | Мінухін С.В. | <i>СВ</i> | | |
| | | | СПРМ-20-1 | Лист 9 |
| Затверд. | Гребеннік І.В. | | СТ | Листів 22 |

ТЕСТОВИЙ СЦЕНАРІЙ РЕЄСТРАЦІЇ У ВЕБ-ДОДАТОК

```

SignUp.feature  + X
1  Feature: Sign Up
2  As a user
3  I want to be able to sign in
4  So that I have an ability to do shopping
5
6  Background:
7  |   Given I navigate to the main page on the site
8
9  @Sign in
10 Scenario: As a user I want to be able to create an account
11 |   When I navigate to the authentication page
12 |   And I create a user with the following information
13 |   | Field | Value |
14 |   | Email | mikhailmoroz2@gmail.com |
15 |   | Personal First Name | Mikhail |
16 |   | Personal Last Name | Moroz |
17 |   | Password | Qqwerty1!! |
18 |   | Day of Birth | 18 |
19 |   | Month of Birth | May |
20 |   | Year of Birth | 1999 |
21 |   | Address First Name | Mikhail |
22 |   | Address Last Name | Moroz |
23 |   | Address | St Main |
24 |   | City | Kharkiv |
25 |   | State | California |
26 |   | Zip code | 62002 |
27 |   | Phone | 0991231232 |
28 |   Then I should see 'Mikhail Moroz' user is registered
29
30
  
```

| | | | | |
|-----------|----------------|---------------------|--|-----------|
| Розроб. | Мороз М.Ю. | <i>М.Ю. Мороз</i> | Дослідження процесів контролю якості взаємодії користувачів з веб-додатком інтернет-магазину | |
| Перевір. | Мінухін С.В. | <i>С.В. Мінухін</i> | | |
| Н. Коитр. | Мінухін С.В. | <i>С.В. Мінухін</i> | | |
| | | | СПРМ-20-1 | Лист 10 |
| Затверд. | Гребеннік І.В. | | СТ | Листів 22 |

ЗГЕНЕРОВАНИЙ КЛАС ВИЗНАЧЕНИХ КРОКІВ З ШАБЛОНАМИ МЕТОДІВ

```

1 using System;
2 using TechTalk.SpecFlow;
3
4 namespace SpecFlowProjectMikhailMoroz.Steps
5 {
6     [Binding]
7     public class AuthenticationsSteps
8     {
9         [When(@"I create a user with the following information")]
10        public void WhenICreateUserWithTheFollowingInformation(Table table)
11        {
12            ScenarioContext.Current.Pending();
13        }
14
15        [Then(@"I should see '(.)' user is registered")]
16        public void ThenIShouldSeeUserIsRegistered(string ps, Table table)
17        {
18            ScenarioContext.Current.Pending();
19        }
20    }
21
22

```

| | | | | |
|-----------|----------------|-----------|--|-----------|
| Розроб. | Мороз М.Ю. | <i>МЮ</i> | Дослідження процесів контролю якості взаємодії користувачів з веб-додатком інтернет-магазину | |
| Перевір. | Мінухін С.В. | <i>СВ</i> | | |
| Н. Контр. | Мінухін С.В. | <i>СВ</i> | | |
| | | | СПРМ-20-1 | Лист 11 |
| Затверд. | Гребеннік І.В. | | СТ | Листів 22 |

ЛОКАТОРИ ЕЛЕМЕНТІВ ДЛЯ РЕЄСТРАЦІЇ

```

using BDD;

namespace SpecFlowProjectMikhailMoroz.POM
{
    /// 
    /// Class AuthenticationPage
    /// 
    public class AuthenticationPage
    {
        public static string emailAddressInputField = "//input[@id='email_create']";
        public static string createAnAccountbutton = "//button[@id='submit_create']";
        public static string genderMale = "//input[@id='id_genders']";
        public static string customerFirstName = "//input[@id='customer_firstname']";
        public static string customerLastName = "//input[@id='customer_lastname']";
        public static string password = "//input[@id='passwd']";
        public static string addressFirstName = "//input[@id='firstname']";
        public static string addressLastName = "//input[@id='lastname']";
        public static string address = "//input[@id='address1']";
        public static string city = "//input[@id='city']";
        public static string zipCode = "//input[@id='postcode']";
        public static string country = "//select[@id='id_country']/option[text()='united states']";
        public static string mobilePhone = "//input[@id='phone_mobile']";
        public static string registerAccount = "//button[@id='submit_account']";
        public static string daysCombobox = "//select[@id='days']";
        public static string monthCombobox = "//select[@id='months']";
        public static string yearsCombobox = "//select[@id='years']";
        public static string loginEmailField = "//form[@id='login_form']/input[@id='email']";
        public static string loginPasswordField = "//form[@id='login_form']/input[@id='passwd']";
        public static string loginSignInButton = "//form[@id='login_form']/button[@id='submit_login']";

        /// 
        /// usernameOrHeader(string username) => $"/div[@class='header_user_info']/a/span[text()='(username)']";
        /// 
        public static string usernameOrHeader(string username) => $"/div[@class='header_user_info']/a/span[text()='(username)']";
        /// 
        /// dayOfBirth(string day) => $"/select[@id='days']/option[contains(text(), '{day}')]";
        /// 
        public static string dayOfBirth(string day) => $"/select[@id='days']/option[contains(text(), '{day}')]";
        /// 
        /// monthOfBirth(string month) => $"/select[@id='months']/option[contains(text(), '{month}')]";
        /// 
        public static string monthOfBirth(string month) => $"/select[@id='months']/option[contains(text(), '{month}')]";
        /// 
        /// yearOfBirth(string year) => $"/select[@id='years']/option[contains(text(), '{year}')]";
        /// 
        public static string yearOfBirth(string year) => $"/select[@id='years']/option[contains(text(), '{year}')]";
        /// 
        /// state(string state) => $"/select[@id='id_state']/option[contains(text(), '{state}')]";
        /// 
        public static string state(string state) => $"/select[@id='id_state']/option[contains(text(), '{state}')]";
    }
}

```

| | | | | |
|-----------|----------------|-----------|--|-----------|
| Розроб. | Мороз М.Ю. | <i>МЮ</i> | Дослідження процесів контролю якості взаємодії користувачів з веб-додатком інтернет-магазину | |
| Перевір. | Мінухін С.В. | <i>СВ</i> | | |
| Н. Коитр. | Мінухін С.В. | <i>СВ</i> | | |
| | | | СПРМ-20-1 | Лист 12 |
| Затверд. | Гребеннік І.В. | | СТ | Листів 22 |

РЕАЛІЗОВАНІ МЕТОДИ КЛАСУ AUTHENTICATIONPAGE

```

@reference
public AuthenticationPage SendTextToEmailField(string emailAddress)
{
    _authenticationMap.SendTextToInput(emailAddressInputField, emailAddress);
    return this;
}

@reference
public AuthenticationPage ClickOnCreateAnAccount()
{
    _authenticationMap.ClickButton(createAnAccountButton);
    return this;
}

@reference
public AuthenticationPage SelectGender()
{
    _authenticationMap.ClickButton(genderMale);
    return this;
}

@reference
public AuthenticationPage SendTextToPersonalFirstNameField(string firstName)
{
    _authenticationMap.SendTextToInput(customerFirstName, firstName);
    return this;
}

@reference
public AuthenticationPage SendTextToPersonalLastNameField(string lastName)
{
    _authenticationMap.SendTextToInput(customerLastName, lastName);
    return this;
}

@reference
public AuthenticationPage SendTextToPasswordField(string userPassword)
{
    _authenticationMap.SendTextToInput(password, userPassword);
    return this;
}




@reference
public AuthenticationPage SelectDateOfBirth(string day, string month, string year)
{
    _authenticationMap.ClickButton(dayOfBirth(day));
    _authenticationMap.ClickButton(monthOfBirth(month));
    _authenticationMap.ClickButton(yearOfBirth(year));
    return this;
}

@reference
public AuthenticationPage SendTextToAddressFirstNameField(string firstName)
{
    _authenticationMap.SendTextToInput(addressFirstName, firstName);
    return this;
}

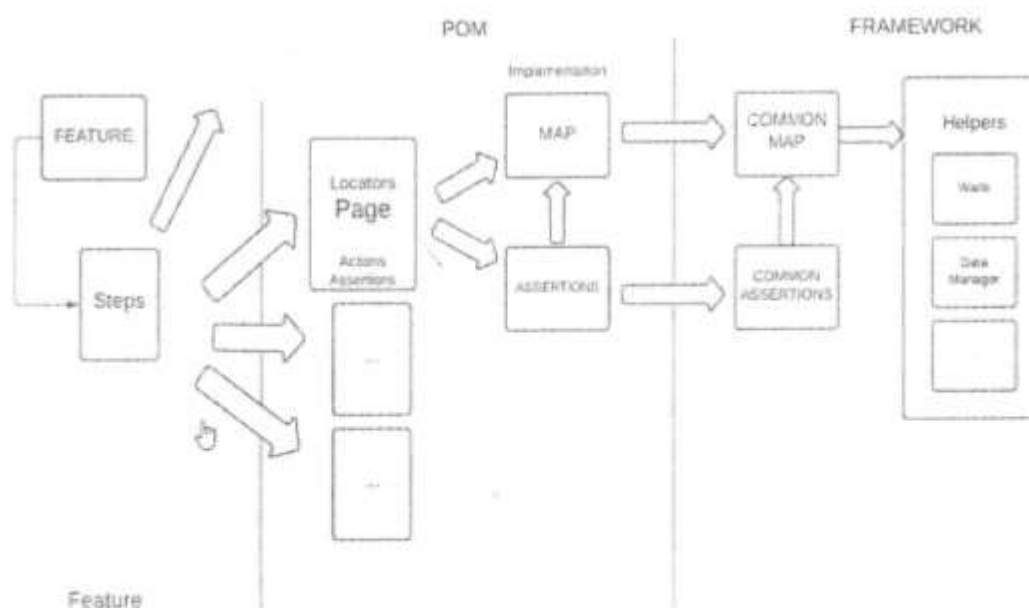
@reference
public AuthenticationPage SendTextToAddressLastNameField(string lastName)
{
    _authenticationMap.SendTextToInput(addressLastName, lastName);
    return this;
}

@reference
public AuthenticationPage SendTextToAddressField(string addressName)
{
    _authenticationMap.SendTextToInput(address, addressName);
    return this;
}

```

| | | | | |
|-----------|----------------|---|--|-----------|
| Розроб. | Мороз М.Ю. |  | Дослідження процесів контролю якості взаємодії користувачів з веб-додатком інтернет-магазину | |
| Перевір. | Мінухін С.В. |  | | |
| Н. Контр. | Мінухін С.В. |  | | |
| | | | СПРМ-20-1 | Лист 13 |
| Затверд. | Гребеннік І.В. | | СТ | Листів 22 |

ДІАГРАМА СТРУКТУРИ ТА ВЗАЄМОДІЇ КОМПОНЕНТІВ



| | | | | |
|-----------|----------------|-----------|--|-----------|
| Розроб. | Мороз М.Ю. | <i>МЮ</i> | Дослідження процесів контролю якості взаємодії користувачів з веб-додатком інтернет-магазину | |
| Перевір. | Мінухін С.В. | <i>СВ</i> | | |
| Н. Коитр. | Мінухін С.В. | <i>СВ</i> | | |
| | | | СПРМ-20-1 | Лист 15 |
| Затверд. | Гребеннік І.В. | | СТ | Листів 22 |

РЕАЛІЗАЦІЯ КЛАСУ DATAMANAGER

```

1  using Bo01;
2  using System;
3  using System.Collections.Generic;
4  using System.Text;
5
6  namespace SpecFlowProjectMikhailMoroz.Helpers
7  {
8      [reference]
9      class DataManager
10     {
11         private IObjectContainer _objectContainer;
12         [reference]
13         public DataManager(IObjectContainer objectContainer)
14         {
15             _objectContainer = objectContainer;
16         }
17
18         [reference]
19         public void SetData(object data, string name)
20         {
21             _objectContainer.RegisterInstanceAs<object>(data, name);
22         }
23
24         [reference]
25         public object GetData(string name)
26         {
27             return _objectContainer.Resolve<object>(name);
28         }
29
30         [reference]
31         public void SetList(list<string> data, string name)
32         {
33             _objectContainer.RegisterInstanceAs<list<string>>(data, name);
34         }
35
36         [reference]
37         public list<string> GetList(string name)
38         {
39             return _objectContainer.Resolve<list<string>>(name);
40         }
41
42         [reference]
43         public void SetListofDict(list<dictionary<string, string>> data, string name)
44         {
45             _objectContainer.RegisterInstanceAs<list<dictionary<string, string>>>(data, name);
46         }
47
48         [reference]
49         public list<dictionary<string, string>> GetListofDict(string name)
50         {
51             return _objectContainer.Resolve<list<dictionary<string, string>>>(name);
52         }
53
54         [reference]
55         public void SetDict(dictionary<string, string> data, string name)
56         {
57             _objectContainer.RegisterInstanceAs<dictionary<string, string>>(data, name);
58         }
59
60         [reference]
61         public dictionary<string, string> GetDict(string name)
62         {
63             return _objectContainer.Resolve<dictionary<string, string>>(name);
64         }
65     }
66 }

```

| | | | | |
|-----------|----------------|-----------|--|-----------|
| Розроб. | Мороз М.Ю. | <i>МЮ</i> | Дослідження процесів контролю якості взаємодії користувачів з веб-додатком інтернет-магазину | |
| Перевір. | Мінухін С.В. | <i>СВ</i> | | |
| Н. Коитр. | Мінухін С.В. | <i>СВ</i> | | |
| | | | СПРМ-20-1 | Лист 16 |
| Затверд. | Гребеннік І.В. | | СТ | Листів 22 |

РЕАЛІЗАЦІЯ КЛАСУ WAITHELPER

```

WaitHelper.cs
SpecFlowProjectMikhailMoroz
SpecFlowProjectMikhailMoroz\Helpers\WaitHelper

10
11 namespace SpecFlowProjectMikhailMoroz.Helpers
12 {
13     // reference
14     class WaitHelper
15     {
16         private IObjectContainer _objectContainer;
17         private DriverManager _driverManager;
18         private WebDriver _webdriver;
19         private WebDriverWait wait;
20
21         // reference
22         public WaitHelper(IObjectContainer objectContainer)
23         {
24             _objectContainer = objectContainer;
25             _driverManager = new DriverManager(_objectContainer);
26             _webdriver = _driverManager.GetDriver();
27             wait = new WebDriverWait(new SystemClock(),
28                 _webdriver,
29                 TimeSpan.FromSeconds(10),
30                 sleepInterval: TimeSpan.FromMilliseconds(100));
31
32         // reference
33         public WebDriverElement ClickWait(string locator)
34         {
35             WebDriverElement element = wait.Until(SeleniumExtras.WaitHelpers.ExpectedConditions.ElementExists(By.XPath(locator)));
36             element = wait.Until(SeleniumExtras.WaitHelpers.ExpectedConditions.ElementIsVisible(By.XPath(locator)));
37             ScrollIntoView(element);
38             element = wait.Until(SeleniumExtras.WaitHelpers.ExpectedConditions.ElementToBeClickable(By.XPath(locator)));
39             return element;
40         }
41
42         // reference
43         public WebDriverElement ClickInNewWindowWait(string buttonLocator, string viewLocator)
44         {
45             ScrollIntoView(buttonLocator, viewLocator);
46             return ClickWait(buttonLocator);
47         }
48
49         // reference
50         public WebDriverElement SendTextWait(string locator)
51         {
52             WebDriverElement element = wait.Until(SeleniumExtras.WaitHelpers.ExpectedConditions.ElementExists(By.XPath(locator)));
53             element = wait.Until(SeleniumExtras.WaitHelpers.ExpectedConditions.ElementIsVisible(By.XPath(locator)));
54             ScrollIntoView(element);
55             return element;
56         }
57
58         // reference
59         public WebDriverElement ExistWait(string locator)
60         {
61             WebDriverElement element = wait.Until(SeleniumExtras.WaitHelpers.ExpectedConditions.ElementExists(By.XPath(locator)));
62             return element;
63         }
64
65         // reference
66         public void NavigateWait(string url)
67         {
68             string shortUrl = url.Replace("https:", "").Replace("http:", "");
69             wait.Until(SeleniumExtras.WaitHelpers.ExpectedConditions.UrlContains(shortUrl));
70             return;
71         }
72     }
73 }

```

| | | | | |
|-----------|----------------|------------|--|-----------|
| Розроб. | Мороз М.Ю. | <i>MJM</i> | Дослідження процесів контролю якості взаємодії користувачів з веб-додатком інтернет-магазину | |
| Перевір. | Мінухін С.В. | <i>СВ</i> | | |
| Н. Коитр. | Мінухін С.В. | <i>СВ</i> | | |
| | | | СПРМ-20-1 | Лист 17 |
| Затверд. | Гребеннік І.В. | <i>ІВ</i> | СТ | Листів 22 |

РЕАЛІЗАЦІЯ КЛАСУ COMMONPAGEMAP

```

CommonPageMap.cs
SpecFlowProjectMikhailMoroz
7 references
abstract class CommonPageMap
{
    private IObjectContainer _objectContainer;
    private DriverManager _driverManager;
    private IWebDriver _webDriver;
    private WaitHelper _waitHelper;
    19 references
    public CommonPageMap(IObjectContainer objectContainer)
    {
        21
        _objectContainer = objectContainer;
        22
        _driverManager = new DriverManager(_objectContainer);
        23
        _waitHelper = new WaitHelper(_objectContainer);
        24
        _webDriver = _driverManager.GetDriver();
        25
    }
    26
    27
    28 references
    public void ClickTextButton(string locator, string text, bool waitForJavaScriptLoadAfterClick = false)
    {
        29
        string formattedLocator = string.Format(locator, text);
        30
        IWebElement element = _waitHelper.ClickWait(formattedLocator);
        31
        element.FindElement(By.XPath(formattedLocator)).Click();
        32
        33
        if (waitForJavaScriptLoadAfterClick) _waitHelper.WaitForJavaScriptLoad();
        34
    }
    35
    36
    37 references
    public void ClickButton(string locator, bool waitForJavaScriptLoadAfterClick = false)
    {
        38
        IWebElement element = _waitHelper.ClickWait(locator);
        39
        element.FindElement(By.XPath(locator)).Click();
        40
        if (waitForJavaScriptLoadAfterClick) _waitHelper.WaitForJavaScriptLoad();
        41
    }
    42
    43
    44 references
    public void ClickButtonOnView(string buttonLocator, string viewLocator, bool waitForJavaScriptLoadAfterClick = false)
    {
        45
        IWebElement element = _waitHelper.ClickButtonOnViewWait(buttonLocator, viewLocator);
        46
        element.FindElement(By.XPath(buttonLocator)).Click();
        47
        if (waitForJavaScriptLoadAfterClick) _waitHelper.WaitForJavaScriptLoad();
        48
    }
    49
    50
    51 references
    public IWebElement GetElementAvoidStaleElementReferenceException(string locator)
    {
        52
        int attempts = 0;
        53
        while (attempts < 3)
        54
        {
        55
            try
        56
            {
        57
                IWebElement element = _waitHelper.ClickWait(locator);
        58
                return element;
        59
            }
        60
            catch (StaleElementReferenceException)
        61
            {
        62
                attempts++;
        63
            }
        64
        }
        65
        throw new StaleElementReferenceException(locator);
        66
    }
    67
    68
}

```

| | | | | |
|-----------|----------------|---------------------|--|-----------|
| Розроб. | Мороз М.Ю. | <i>dm</i> | Дослідження процесів контролю якості взаємодії користувачів з веб-додатком інтернет-магазину | |
| Перевір. | Мінухін С.В. | <i>С.В. Минухин</i> | | |
| Н. Контр. | Мінухін С.В. | <i>С.В. Минухин</i> | | |
| | | | СПРМ-20-1 | Лист 18 |
| Затверд. | Гребеннік І.В. | | СТ | Листів 22 |

РЕАЛІЗАЦІЯ КЛАСУ COMMONPAGEASSERTIONS

```

CommonPageAssertions.cs - X
SpecFlowProject\MichaelMoroz.CommonPageAssertions
SpecFlowProject\MichaelMoroz.CommonPageAssertions

10 namespace SpecFlowProject\MichaelMoroz.CommonPage
11 {
12     [TestFixture]
13     class CommonPageAssertions
14     {
15         private IObjectContainer _objectContainer;
16         private DriverManager _driverManager;
17         private IWebDriver _webdriver;
18
19         [SetUp]
20         public CommonPageAssertions(IObjectContainer objectContainer)
21         {
22             _objectContainer = objectContainer;
23             _driverManager = new DriverManager(_objectContainer);
24             _webdriver = _driverManager.GetDriver();
25         }
26
27         [Test]
28         public void AreEqual(object expected, object actual, string assertionMessage)
29         {
30             Assert.AreEqual(expected, actual, assertionMessage);
31         }
32
33         [Test]
34         public void IsTrue(bool condition, string assertionMessage)
35         {
36             Assert.IsTrue(condition, assertionMessage);
37         }
38
39         [Test]
40         public void IsFalse(bool condition, string assertionMessage)
41         {
42             Assert.IsFalse(condition, assertionMessage);
43         }
44
45         [Test]
46         public void IsEmpty(string actual, string assertionMessage)
47         {
48             Assert.IsEmpty(actual, assertionMessage);
49         }
50
51         [Test]
52         public void Contains(object expected, ICollection actual, string message)
53         {
54             Assert.Contains(expected, actual, message);
55         }
56
57         [Test]
58         public void NotContains(object expected, ICollection actual, string message)
59         {
60             collectionAssert.DoesNotContain(actual, expected, message);
61         }
62
63         [Test]
64         public void StartsWith(string expected, string actual, string assertionMessage)
65         {
66             StringAssert.StartsWith(expected, actual, assertionMessage);
67         }
68
69         [Test]
70         public void PageTitleIsEqualWithDelay(string expectedText, string assertionMessage, int afterSeconds = 5)
71         {
72             Assert.That(() => _webdriver.Title, Is.EqualTo(expectedText).After(afterSeconds).Seconds.PollEvery(100), assertionMessage);
73         }
74     }

```

| | | | | |
|-----------|----------------|------------|--|-----------|
| Розроб. | Мороз М.Ю. | <i>MJM</i> | Дослідження процесів контролю якості взаємодії користувачів з веб-додатком інтернет-магазину | |
| Перевір. | Мінухін С.В. | <i>SV</i> | | |
| Н. Коитр. | Мінухін С.В. | <i>SV</i> | | |
| | | | СПРМ-20-1 | Лист 19 |
| Затверд. | Гребеннік І.В. | | СТ | Листів 22 |

РЕАЛІЗАЦІЯ КЛАСУ DRIVERMANAGER

```

DriverManager.cs
SpecFlowProjectMikhailMoroz
7 using System.IO;
8 using NUnit.Framework;
9 using OpenQA.Selenium.Remote;
10
11 [assembly: Parallelizable(ParallelScope.Fixtures)]
12 [assembly: LevelOfParallelism(1)]
13 namespace SpecFlowProjectMikhailMoroz.Drivers
14 {
15     [Binding]
16     class DriverManager
17     {
18         private readonly IObjectContainer _objectContainer;
19         private IWebDriver _driver;
20         private DriverFactory driverFactory;
21         private static string _browser;
22
23         3 references
24         public DriverManager(IObjectContainer objectContainer)
25         {
26             _objectContainer = objectContainer;
27             driverFactory = new DriverFactory();
28
29         }
30
31         [BeforeTestRun]
32         0 references
33         public static void SetEnvironment()
34         {
35             var settings = ConfigurationLoader.Settings;
36             _browser = settings.Browser;
37
38         }
39
40         [BeforeScenario]
41         0 references
42         public void SelectBrowser()
43         {
44             Initialize();
45
46         }
47
48         1 reference
49         public void Initialize()
50         {
51             _driver = driverFactory.InitLocalDriver(_browser);
52             IAllowsFileDetection allowsDetection = (IAllowsFileDetection) _driver;
53             allowsDetection.FileDetector = new LocalFileDetector();
54             _objectContainer.RegisterInstanceAs<IWebDriver>(_driver, "driver");
55
56         }
57
58         3 references
59         public IWebDriver GetDriver()
60         {
61             _driver = _objectContainer.Resolve<IWebDriver>("driver");
62             return _driver;
63
64         }
65
66         [AfterScenario]
67         0 references
68         public void Cleanup()
69         {
70             _driver.Quit();
71
72         }
73     }

```

| | | | | |
|-----------|----------------|-----------|--|-----------|
| Розроб. | Мороз М.Ю. | <i>МЮ</i> | Дослідження процесів контролю якості взаємодії користувачів з веб-додатком інтернет-магазину | |
| Перевір. | Мінухін С.В. | <i>СВ</i> | | |
| Н. Коитр. | Мінухін С.В. | <i>СВ</i> | | |
| | | | СПРМ-20-1 | Лист 20 |
| Затверд. | Гребеннік І.В. | | СТ | Листів 22 |

РЕАЛІЗАЦІЯ КЛАСУ DRIVERFACTORY

```

DriverFactory.cs
SpecFlowProjectMikhailMoroz
SpecFlowProjectMikhailMoroz.Drivers.DriverFactory

1 using OpenQA.Selenium;
2 using OpenQA.Selenium.Chrome;
3 using OpenQA.Selenium.Edge;
4 using OpenQA.Selenium.Firefox;
5 using System;
6 using System.Collections.Generic;
7 using System.IO;
8 using System.Text;
9
10 namespace SpecFlowProjectMikhailMoroz.Drivers
11 {
12     class DriverFactory
13     {
14         private WebDriver _driver;
15         private OptionsFactory options = new OptionsFactory();
16         public WebDriver InitLocalDriver(string browser)
17         {
18             var webDriverServerPath = Path.GetDirectoryName(System.Reflection.Assembly.GetExecutingAssembly().Location);
19             switch (browser)
20             {
21                 case "chrome":
22                     _driver = new ChromeDriver(webDriverServerPath, options.getChromeOptions());
23                     _driver.Manage().Window.Maximize();
24                     return _driver;
25                 case "firefox":
26                     _driver = new FirefoxDriver(webDriverServerPath, options.getFirefoxOptions());
27                     _driver.Manage().Window.Maximize();
28                     return _driver;
29                 case "edge":
30                     _driver = new EdgeDriver(webDriverServerPath, options.getEdgeOptions());
31                     _driver.Manage().Window.Maximize();
32                     return _driver;
33                 default:
34                     _driver = new ChromeDriver(webDriverServerPath, options.getChromeOptions());
35                     _driver.Manage().Window.Maximize();
36                     return _driver;
37             }
38         }
39     }
40 }
41

```

| | | | | |
|-----------|----------------|-------------|--|-----------|
| Розроб. | Мороз М.Ю. | <i>M.M.</i> | Дослідження процесів контролю якості взаємодії користувачів з веб-додатком інтернет-магазину | |
| Перевір. | Мінухін С.В. | <i>S.V.</i> | | |
| Н. Коитр. | Мінухін С.В. | <i>S.V.</i> | | |
| | | | СПРМ-20-1 | Лист 21 |
| Затверд. | Гребеннік І.В. | | СТ | Листів 22 |

РЕАЛІЗАЦІЯ КЛАСУ OPTIONSFACTORY

```

OptionsFactory.cs
SpecFlowProjectMikhailMoroz
1 using OpenQA.Selenium.Chrome;
2 using OpenQA.Selenium.Edge;
3 using OpenQA.Selenium.Firefox;
4 using System;
5 using System.Collections.Generic;
6 using System.Text;
7
8 namespace SpecFlowProjectMikhailMoroz.Drivers
9 {
10     2 references
11     class OptionsFactory
12     {
13         2 references
14         public ChromeOptions getChromeOptions()
15         {
16             string[] arguments = {"--start-maximized", "--ignore-certificate-errors",
17             "--disable-popup-blocking", "--no-sandbox", "--disable-dev-shm-usage"};
18             ChromeOptions chromeOptions = new ChromeOptions();
19             chromeOptions.AcceptInsecureCertificates = true;
20             chromeOptions.AddArguments(arguments);
21
22             return chromeOptions;
23         }
24         1 reference
25         public FirefoxOptions getFirefoxOptions()
26         {
27             FirefoxOptions firefoxOptions = new FirefoxOptions();
28             firefoxOptions.AcceptInsecureCertificates = true;
29
30             return firefoxOptions;
31         }
32         1 reference
33         public EdgeOptions getEdgeOptions()
34         {
35             EdgeOptions edgeOptions = new EdgeOptions();
36             edgeOptions.UseChromium = true;
37             edgeOptions.AcceptInsecureCertificates = true;
38
39             return edgeOptions;
40         }
41     }
42 }

```

| | | | | |
|-----------|----------------|-----------|--|-----------|
| Розроб. | Мороз М.Ю. | <i>ММ</i> | Дослідження процесів контролю якості взаємодії користувачів з веб-додатком інтернет-магазину | |
| Перевір. | Мінухін С.В. | <i>СВ</i> | | |
| Н. Коитр. | Мінухін С.В. | <i>СВ</i> | | |
| | | | СПРМ-20-1 | Лист 22 |
| Затверд. | Гребеннік І.В. | | СТ | Листів 22 |

Додаток Б

Текст програми

ГЮИК.501610.008 – 01 12 01

(позначення документу)

ЗАТВЕРДЖЕНО
ГЮИК.501610.008 – 01 12 01 – ЛЗ

Дослідження процесів контролю якості взаємодії користувачів з веб-додатком
Інтернет-магазину

Текст програми

ГЮИК.501610.008 – 01 12 01 – ЛЗ

ЛИСТІВ 21

2021 р.

Міністерство освіти і науки України
Харківський Національний Університет Радіоелектроніки

«ЗАТВЕРДЖУЮ»
керівник кваліфікаційної роботи
проф. Мінухін С. В.

Дослідження процесів контролю якості взаємодії користувачів з веб-додатком
Інтернет-магазину

Текст програми

ЛИСТ ЗАТВЕРДЖЕННЯ

ГЮИК.501610.008 – 01 12 01 – ЛЗ

РОЗРОБИВ:
ст. гр. СПРМ-20-1
Мороз М.Ю.

2021 р.

DriverFactory.cs

```
using OpenQA.Selenium;
using OpenQA.Selenium.Chrome;
using OpenQA.Selenium.Edge;
using OpenQA.Selenium.Firefox;
using System.IO;

namespace SpecFlowProjectMikhailMoroz.Drivers
{
    class DriverFactory
    {
        private IWebDriver _driver;
        private OptionsFactory options = new OptionsFactory();
        public IWebDriver InitLocalDriver(string browser)
        {
            var webDriverServerPath =
                Path.GetDirectoryName(System.Reflection.Assembly.GetExecutingAssembly().Location);
            switch (browser)
            {
                case "chrome":
                    _driver = new ChromeDriver(webDriverServerPath,
options.getChromeOptions());
                    _driver.Manage().Window.Maximize();
                    return _driver;
                case "firefox":
                    _driver = new FirefoxDriver(webDriverServerPath, options.getFirefoxOptions());
                    _driver.Manage().Window.Maximize();
                    return _driver;
                case "edge":
                    _driver = new EdgeDriver(webDriverServerPath, options.getEdgeOptions());
                    _driver.Manage().Window.Maximize();
                    return _driver;
                default:
                    _driver = new ChromeDriver(webDriverServerPath,
options.getChromeOptions());
                    _driver.Manage().Window.Maximize();
                    return _driver;
            }
        }
    }
}
```

DriverManager.cs

```
using TechTalk.SpecFlow;
using BoDi;
using OpenQA.Selenium;
using NUnit.Framework;
using OpenQA.Selenium.Remote;

[assembly: Parallelizable(ParallelScope.Fixtures)]
[assembly: LevelOfParallelism(1)]
namespace SpecFlowProjectMikhailMoroz.Drivers
{
    [Binding]
    class DriverManager
    {
        private readonly IObjectContainer _objectContainer;
        private IWebDriver _driver;
        private DriverFactory driverFactory;
        private static string _browser;

        public DriverManager(IObjectContainer objectContainer)
        {
            _objectContainer = objectContainer;
            driverFactory = new DriverFactory();
        }

        [BeforeTestRun]
        public static void setEnvironment()
        {
            var settings = ConfigurationLoader.Settings;
            _browser = settings.Browser;
        }

        [BeforeScenario]
        public void SelectBrowser()
        {
            Initialize();
        }

        public void Initialize()
        {
            _driver = driverFactory.InitLocalDriver(_browser);
            IAllowsFileDetection allowsDetection = (IAllowsFileDetection)_driver;
            allowsDetection.FileDetector = new LocalFileDetector();
            _objectContainer.RegisterInstanceAs<IWebDriver>(_driver, "driver");
        }
    }
}
```

```

public IWebDriver getDriver()
{
    _driver = _objectContainer.Resolve<IWebDriver>("driver");
    return _driver;
}

[AfterScenario]
public void CleanUp()
{
    _driver.Quit();
}
}
}

```

OptionsFactory.cs

```

using OpenQA.Selenium.Chrome;
using OpenQA.Selenium.Edge;
using OpenQA.Selenium.Firefox;

namespace SpecFlowProjectMikhailMoroz.Drivers
{
    class OptionsFactory
    {
        public ChromeOptions getChromeOptions()
        {
            string[] arguments = {"--start-maximized", "--ignore-certificate-errors",
                "--disable-popup-blocking", "--no-sandbox", "--disable-dev-shm-usage"};
            ChromeOptions chromeOptions = new ChromeOptions();
            chromeOptions.AcceptInsecureCertificates = true;
            chromeOptions.AddArguments(arguments);

            return chromeOptions;
        }
        public FirefoxOptions getFirefoxOptions()
        {
            FirefoxOptions firefoxOptions = new FirefoxOptions();
            firefoxOptions.AcceptInsecureCertificates = true;

            return firefoxOptions;
        }
        public EdgeOptions getEdgeOptions()
        {
            EdgeOptions edgeOptions = new EdgeOptions();
            edgeOptions.UseChromium = true;
            edgeOptions.AcceptInsecureCertificates = true;

            return edgeOptions;
        }
    }
}

```

```
    }  
  }  
}  
  
                CommonPageAssertions.cs  
  
using BoDi;  
using NUnit.Framework;  
using OpenQA.Selenium;  
using SpecFlowProjectMikhailMoroz.Drivers;  
using System.Collections;  
  
namespace SpecFlowProjectMikhailMoroz.CommonPage  
{  
    class CommonPageAssertions  
    {  
  
        private IObjectContainer _objectContainer;  
        private DriverManager _driverManager;  
        private IWebDriver _webDriver;  
  
        public CommonPageAssertions(IObjectContainer objectContainer)  
        {  
            _objectContainer = objectContainer;  
            _driverManager = new DriverManager(_objectContainer);  
            _webDriver = _driverManager.getDriver();  
        }  
  
        public void AreEqual(object expected, object actual, string assertionMessage)  
        {  
            Assert.AreEqual(expected, actual, assertionMessage);  
        }  
  
        public void IsTrue(bool condition, string assertionMessage)  
        {  
            Assert.IsTrue(condition, assertionMessage);  
        }  
  
        public void IsFalse(bool condition, string assertionMessage)  
        {  
            Assert.IsFalse(condition, assertionMessage);  
        }  
        public void IsEmpty(string actual, string assertionMessage)  
        {  
            Assert.IsEmpty(actual, assertionMessage);  
        }  
    }  
}
```

```

public void Contains(object expected, ICollection actual, string message)
{
    Assert.Contains(expected, actual, message);
}

public void NotContains(object expected, ICollection actual, string message)
{
    CollectionAssert.DoesNotContain(actual, expected, message);
}

public void StartsWith(string expected, string actual, string assertionMessage)
{
    StringAssert.StartsWith(expected, actual, assertionMessage);
}

public void PageTitleIsEqualWithDelay(string expectedText, string assertionMessage, int
afterSeconds = 5)
{
    Assert.That(() => _webDriver.Title,
Is.EqualTo(expectedText).After(afterSeconds).Seconds.PollEvery(100), assertionMessage);
}
}
}

```

CommonPageMap.cs

```

using BoDi;
using OpenQA.Selenium;
using SpecFlowProjectMikhailMoroz.Drivers;
using SpecFlowProjectMikhailMoroz.Helpers;
using System;
using System.Collections.Generic;
using System.Collections.ObjectModel;
using System.Linq;

namespace SpecFlowProjectMikhailMoroz.CommonPage
{
    abstract class CommonPageMap
    {
        private IObjectContainer _objectContainer;
        private DriverManager _driverManager;
        private IWebDriver _webDriver;
        private WaitHelper _waitHelper;
        public CommonPageMap(IObjectContainer objectContainer)
        {
            _objectContainer = objectContainer;
            _driverManager = new DriverManager(_objectContainer);
            _waitHelper = new WaitHelper(_objectContainer);
            _webDriver = _driverManager.getDriver();
        }
    }
}

```

```

    public void ClickTextButton(string locator, string text, bool
waitForJavaScriptLoadAfterClick = false)
    {
        string formattedLocator = string.Format(locator, text);
        IWebElement element = _waitHelper.ClickWait(formattedLocator);
        element.FindElement(By.XPath(formattedLocator)).Click();

        if (waitForJavaScriptLoadAfterClick) _waitHelper.WaitForJavaScriptLoad();
    }

    public void ClickButton(string locator, bool waitForJavaScriptLoadAfterClick = false)
    {
        IWebElement element = _waitHelper.ClickWait(locator);
        element.FindElement(By.XPath(locator)).Click();
        if (waitForJavaScriptLoadAfterClick) _waitHelper.WaitForJavaScriptLoad();
    }

    public void ClickButtonOnView(string buttonLocator, string viewLocator, bool
waitForJavaScriptLoadAfterClick = false)
    {
        IWebElement element = _waitHelper.ClickButtonOnViewWait(buttonLocator,
viewLocator);
        element.FindElement(By.XPath(buttonLocator)).Click();
        if (waitForJavaScriptLoadAfterClick) _waitHelper.WaitForJavaScriptLoad();
    }

    public IWebElement GetElementAvoidStaleElementReferenceException(string locator)
    {
        int attempts = 0;
        while (attempts < 3)
        {
            try
            {
                IWebElement element = _waitHelper.ClickWait(locator);
                return element;
            }
            catch (StaleElementReferenceException)
            {
                attempts++;
            }
        }
        throw new StaleElementReferenceException(locator);
    }

```

```
public void SendTextToInput(string locator, string text, bool
waitForJavaScriptLoadAfterInput = false)
{
    IWebElement element = _waitHelper.SendTextWait(locator);
    element.FindElement(By.XPath(locator)).SendKeys(text);
    if (waitForJavaScriptLoadAfterInput) _waitHelper.WaitForJavaScriptLoad();
}

public void ClearInputAndSendText(string locator, string text)
{
    IWebElement element = _waitHelper.SendTextWait(locator);
    element.SendKeys(Keys.Control + "a");
    element.SendKeys(text);
}

public string GetText(string locator)
{
    IWebElement element = _waitHelper.SendTextWait(locator);
    return element.FindElement(By.XPath(locator)).Text;
}

public void Navigate(string url, bool isWaitNeeded = true)
{
    _webDriver.Navigate().GoToUrl(url);
    if (isWaitNeeded) _waitHelper.NavigateWait(url);
}

public void Refresh()
{
    _webDriver.Navigate().Refresh();
}

public IWebElement GetElement(string locator)
{
    return _waitHelper.GetTextWait(locator).FindElement(By.XPath(locator));
}

public IWebElement GetInvisibleElement(string locator)
{
    return _waitHelper.ExistsWait(locator).FindElement(By.XPath(locator));
}

public ReadOnlyCollection<IWebElement> FindElementsInFrame(string locatorFrame,
string locatorElement)
{
    _waitHelper.WaitForJavaScriptLoad();
}
```

```

        IWebElement frame =
        _waitHelper.GetTextWait(locatorFrame).FindElement(By.XPath(locatorFrame));
        return
        _webDriver.SwitchTo().Frame(frame).FindElements(By.XPath(locatorElement));
    }
    public IWebElement GetElementByFormattedLocator(string locator, string text)
    {
        string formattedLocator = string.Format(locator, text);
        return
        _waitHelper.GetTextWait(formattedLocator).FindElement(By.XPath(formattedLocator));
    }

    public string GetElementValue(string locator)
    {
        return
        _waitHelper.GetTextWait(locator).FindElement(By.XPath(locator)).GetAttribute("value");
    }

    public ReadOnlyCollection<IWebElement> GetElements(string locator)
    {
        _waitHelper.GetTextWait(locator);
        return _webDriver.FindElements(By.XPath(locator));
    }

    public IWebElement GetElementWithoutWait(string locator)
    {
        return _webDriver.FindElement(By.XPath(locator));
    }

    public ReadOnlyCollection<IWebElement> GetElementsWithoutWait(string locator)
    {
        return _webDriver.FindElements(By.XPath(locator));
    }

    public string GetPageUrl()
    {
        return _webDriver.Url;
    }

    public List<String> GetListOfValues(string locator)
    {
        return _webDriver.FindElements(By.XPath(locator)).Select(e =>
e.GetAttribute("value")).ToList();
    }

    public List<String> GetListOfTitles(string locator)
    {

```

```

        return _webDriver.FindElements(By.XPath(locator)).Select(e =>
e.GetAttribute("title")).ToList();
    }

    public List<String> GetListOfTexts(string locator)
    {
        return _webDriver.FindElements(By.XPath(locator)).Select(e => e.Text).ToList();
    }

```

DataManager.cs

```

using BoDi;
using System.Collections.Generic;

namespace SpecFlowProjectMikhailMoroz.Helpers
{
    class DataManager
    {
        private IObjectContainer _objectContainer;
        public DataManager(IObjectContainer objectContainer)
        {
            _objectContainer = objectContainer;
        }

        public void SetData(object data, string name)
        {
            _objectContainer.RegisterInstanceAs<object>(data, name);
        }

        public object GetData(string name)
        {
            return _objectContainer.Resolve<object>(name);
        }

        public void SetList(List<string> data, string name)
        {
            _objectContainer.RegisterInstanceAs<List<string>>(data, name);
        }

        public List<string> GetList(string name)
        {
            return _objectContainer.Resolve<List<string>>(name);
        }

        public void SetListofDict(List<Dictionary<string, string>> data, string name)
        {
            _objectContainer.RegisterInstanceAs<List<Dictionary<string, string>>>(data, name);
        }
    }

```

```

    public List<Dictionary<string, string>> GetListOfDict(string name)
    {
        return _objectContainer.Resolve<List<Dictionary<string, string>>>(name);
    }

    public void SetDict(Dictionary<string, string> data, string name)
    {
        _objectContainer.RegisterInstanceAs<Dictionary<string, string>>(data, name);
    }

    public Dictionary<string, string> GetDict(string name)
    {
        return _objectContainer.Resolve<Dictionary<string, string>>(name);
    }
    public void SetListInt(List<int> data, string name)
    {
        _objectContainer.RegisterInstanceAs<List<int>>(data, name);
    }

    public List<int> GetListInt(string name)
    {
        return _objectContainer.Resolve<List<int>>(name);
    }
}

```

WaitHelper.cs

```

using BoDi;
using OpenQA.Selenium;
using OpenQA.Selenium.Interactions;
using OpenQA.Selenium.Support.UI;
using SpecFlowProjectMikhailMoroz.Drivers;
using System;
using System.Threading;

namespace SpecFlowProjectMikhailMoroz.Helpers
{
    class WaitHelper
    {
        private IObjectContainer _objectContainer;
        private DriverManager _driverManager;
        private IWebDriver _webDriver;
        private WebDriverWait wait;

        public WaitHelper(IObjectContainer objectContainer)
        {
            _objectContainer = objectContainer;

```

```

        _driverManager = new DriverManager(_objectContainer);
        _webDriver = _driverManager.getDriver();
        wait = new WebDriverWait(new SystemClock(),
                                _webDriver,
                                TimeSpan.FromSeconds(10),
                                sleepInterval: TimeSpan.FromMilliseconds(100));
    }

    public IWebElement ClickWait(string locator)
    {
        IWebElement element =
        wait.Until(SeleniumExtras.WaitHelpers.ExpectedConditions.ElementExists(By.XPath(locator
        )));
        element =
        wait.Until(SeleniumExtras.WaitHelpers.ExpectedConditions.ElementIsVisible(By.XPath(locator
        )));
        ScrollToView(element);
        element =
        wait.Until(SeleniumExtras.WaitHelpers.ExpectedConditions.ElementToBeClickable(By.XPath
        h(locator)));
        return element;
    }

    public IWebElement ClickButtonOnViewWait(string buttonLocator, string viewLocator)
    {
        ScrollIntoView(buttonLocator, viewLocator);
        return ClickWait(buttonLocator);
    }

    public IWebElement SendTextWait(string locator)
    {
        IWebElement element =
        wait.Until(SeleniumExtras.WaitHelpers.ExpectedConditions.ElementExists(By.XPath(locator
        )));
        element =
        wait.Until(SeleniumExtras.WaitHelpers.ExpectedConditions.ElementIsVisible(By.XPath(locator
        )));
        ScrollToView(element);
        return element;
    }

    public IWebElement ExistsWait(string locator)
    {
        IWebElement element =
        wait.Until(SeleniumExtras.WaitHelpers.ExpectedConditions.ElementExists(By.XPath(locator
        )));
        return element;
    }

```

```

    }

    public void NavigateWait(string url)
    {
        string shortURL = url.Replace("https:", "").Replace("http:", "");
        wait.Until(SeleniumExtras.WaitHelpers.ExpectedConditions.UrlContains(shortURL));
        return;
    }

    public IWebElement GetTextWait(string locator)
    {
        IWebElement element =
        wait.Until(SeleniumExtras.WaitHelpers.ExpectedConditions.ElementExists(By.XPath(locator
        )));
        element =
        wait.Until(SeleniumExtras.WaitHelpers.ExpectedConditions.ElementIsVisible(By.XPath(locator
        )));
        ScrollToView(element);
        return element;
    }

```

AuthenticationPage.cs

using BoDi;

namespace SpecFlowProjectMikhailMoroz.POM

```

{
    class AuthenticationPage
    {
        public static string emailAddressInputField = "//input[@id='email_create']";
        public static string createAnAccountButton = "//button[@id='SubmitCreate']";
        public static string genderMale = "//input[@id='id_gender1']";
        public static string customerFirstName = "//input[@id='customer_firstname']";
        public static string customerLastName = "//input[@id='customer_lastname']";
        public static string password = "//input[@id='passwd']";
        public static string addressFirstName = "//input[@id='firstname']";
        public static string addressLastName = "//input[@id='lastname']";
        public static string address = "//input[@id='address1']";
        public static string city = "//input[@id='city']";
        public static string zipCode = "//input[@id='postcode']";
        public static string country = "//select[@id='id_country']/option[text()='United States']";
        public static string mobilePhone = "//input[@id='phone_mobile']";
        public static string registerAccount = "//button[@id='submitAccount']";
        public static string daysCombobox = "//select[@id='days']";
        public static string monthCombobox = "//select[@id='months']";
        public static string yearsCombobox = "//select[@id='years']";
        public static string loginEmailField = "//form[@id='login_form']//input[@id='email']";
        public static string loginPasswordField =
        "//form[@id='login_form']//input[@id='passwd']";
    }
}

```

```

    public static string loginSignInButton =
    "-//form[@id='login_form']//button[@id='SubmitLogin']";
    public static string usernameOnHeader(string username) =>
    "$//div[@class='header_user_info']/a/span[text()=' {username} ]";
    public static string dayOfBirth(string day) =>
    "$//select[@id='days']/option[contains(text(), '{ day })]";
    public static string monthOfBirth(string month) =>
    "$//select[@id='months']/option[contains(text(), '{ month })]";
    public static string yearOfBirth(string year) =>
    "$//select[@id='years']/option[contains(text(), '{ year })]";
    public static string state(string state) =>
    "$//select[@id='id_state']/option[contains(text(), '{ state })]";

    private AuthenticationMap _authenticationMap;
    private AuthenticationAssertions _authenticationAssertions;

    public AuthenticationPage(IObjectContainer objectContainere, AuthenticationAssertions
authenticationAssertions)
    {
        _authenticationMap = new AuthenticationMap(objectContainere);
        _authenticationAssertions = authenticationAssertions;
    }

    public AuthenticationPage SendTextToEmailField(string emailAddress)
    {
        _authenticationMap.SendTextToInput(emailAddressInputField, emailAddress);
        return this;
    }
    public AuthenticationPage ClickOnCreateAnAccount()
    {
        _authenticationMap.ClickButton(createAnAccountButton);
        return this;
    }
    public AuthenticationPage SelectGender()
    {
        _authenticationMap.ClickButton(genderMale);
        return this;
    }
    public AuthenticationPage SendTextToPersonalFirstNameField(string firstName)
    {
        _authenticationMap.SendTextToInput(customerFirstName, firstName);
        return this;
    }
    public AuthenticationPage SendTextToPersonalLastNameField(string lastName)
    {

```

```
    _authenticationMap.SendTextToInput(customerLastName, lastName);
    return this;
}
public AuthenticationPage SendTextToPasswordField(string userPassword)
{
    _authenticationMap.SendTextToInput(password, userPassword);
    return this;
}

public AuthenticationPage SelectDateOfBirth(string day, string month, string year)
{
    _authenticationMap.ClickButton(dayOfBirth(day));
    _authenticationMap.ClickButton(monthOfBirth(month));
    _authenticationMap.ClickButton(yearOfBirth(year));
    return this;
}

public AuthenticationPage SendTextToAddressFirstNameField(string firstName)
{
    _authenticationMap.SendTextToInput(addressFirstName, firstName);
    return this;
}

public AuthenticationPage SendTextToAddressLastNameField(string lastName)
{
    _authenticationMap.SendTextToInput(addressLastName, lastName);
    return this;
}

public AuthenticationPage SendTextToAddressField(string addressName)
{
    _authenticationMap.SendTextToInput(address, addressName);
    return this;
}

public AuthenticationPage SendTextToCityField(string cityName)
{
    _authenticationMap.SendTextToInput(city, cityName);
    return this;
}

public AuthenticationPage SelectCountry()
{
    _authenticationMap.ClickButton(country);
    return this;
}

public AuthenticationPage SelectState(string stateName)
{
```

```
        _authenticationMap.ClickButton(state(stateName));
        return this;
    }

    public AuthenticationPage SendTextToZipCodeField(string zipCodeName)
    {
        _authenticationMap.SendTextToInput(zipCode, zipCodeName);
        return this;
    }

    public AuthenticationPage SendTextToMobilePhoneField(string mobilePhoneName)
    {
        _authenticationMap.SendTextToInput(mobilePhone, mobilePhoneName);
        return this;
    }

    public AuthenticationPage RegisterAccount()
    {
        _authenticationMap.ClickButton(registerAccount);
        return this;
    }

    public AuthenticationPage VerifyUserIsRegistered(string userName)
    {
        _authenticationAssertions.AreEqual(userName,
        _authenticationMap.GetText(usernameOnHeader(userName)), $"{userName} is incorrect");
        return this;
    }

    public AuthenticationPage SendTextToLoginEmailField(string email)
    {
        _authenticationMap.SendTextToInput(loginEmailField, email);
        return this;
    }

    public AuthenticationPage SendTextToLoginPasswordField(string password)
    {
        _authenticationMap.SendTextToInput(loginPasswordField, password);
        return this;
    }
    public AuthenticationPage SignIn()
    {
        _authenticationMap.ClickButton(loginSignInButton);
        return this;
    }
}
}
```

AuthenticationMap.cs

```

using BoDi;
using OpenQA.Selenium;
using SpecFlowProjectMikhailMoroz.CommonPage;
using SpecFlowProjectMikhailMoroz.Drivers;
using SpecFlowProjectMikhailMoroz.Helpers;

namespace SpecFlowProjectMikhailMoroz.POM
{
    class AuthenticationMap : CommonPageMap
    {
        IObjectContainer _objectContainer;
        private DriverManager _driverManager;
        private IWebDriver _webDriver;
        private WaitHelper waitHelper;
        public AuthenticationMap(IObjectContainer objectContainer) : base(objectContainer)
        {
            _objectContainer = objectContainer;
            _driverManager = new DriverManager(_objectContainer);
            waitHelper = new WaitHelper(_objectContainer);
            _webDriver = _driverManager.getDriver();
        }

        public void ChooseElementInComboBoxList(string ValueInComboBoxListField, string
value)
        {
            ClickTextButton(ValueInComboBoxListField, value);
        }
    }
}

```

AuthenticationAssertions.cs

```

using BoDi;
using OpenQA.Selenium;
using SpecFlowProjectMikhailMoroz.CommonPage;
using SpecFlowProjectMikhailMoroz.Drivers;
using SpecFlowProjectMikhailMoroz.Helpers;
using System;
using System.Collections.Generic;
using System.Text;

namespace SpecFlowProjectMikhailMoroz.POM
{
    class AuthenticationAssertions : CommonPageAssertions
    {
        private IObjectContainer _objectContainer;
        private DriverManager _driverManager;
    }
}

```

```

    private IWebDriver _webDriver;
    private WaitHelper waitHelper;
    public AuthenticationAssertions(IObjectContainer objectContainer) :
base(objectContainer)
    {
        _objectContainer = objectContainer;
        _driverManager = new DriverManager(_objectContainer);
        waitHelper = new WaitHelper(_objectContainer);
        _webDriver = _driverManager.getDriver();
    }
}

```

AuthenticationSteps.cs

```

using SpecFlowProjectMikhailMoroz.POM;
using TechTalk.SpecFlow;

namespace SpecFlowProjectMikhailMoroz.Steps
{
    [Binding]
    class AuthenticationSteps
    {
        private readonly ScenarioContext _scenarioContext;
        private AuthenticationPage _authenticationPage;

        public AuthenticationSteps(AuthenticationPage authenticationPage)
        {
            _authenticationPage = authenticationPage;
        }

        [When(@"I create a user with the following information")]
        public void WhenICreateAUserWithTheFollowingInformation(Table userDetailsTable)
        {
            var detailsGrid = userDetailsTable.Rows;
            _authenticationPage.SendTextToEmailField(detailsGrid[0]["Email"])
                .ClickOnCreateAnAccount()
                .SelectGender()
                .SendTextToPersonalFirstNameField(detailsGrid[0]["Personal First Name"])
                .SendTextToPersonalLastNameField(detailsGrid[0]["Personal Last Name"])
                .SendTextToPasswordField(detailsGrid[0]["Password"])
                .SelectDateOfBirth(detailsGrid[0]["Day of Birth"], detailsGrid[0]["Month of
Birth"], detailsGrid[0]["Year of Birth"])
                .SendTextToAddressIFirstNameField(detailsGrid[0]["Address First Name"])
                .SendTextToAddressILastNameField(detailsGrid[0]["Address Last Name"])
                .SendTextToAddressField(detailsGrid[0]["Address"])
                .SendTextToCityField(detailsGrid[0]["City"])
        }
    }
}

```

```

        .SelectCountry()
        .SelectState(detailsGrid[0]["State"])
        .SendTextToZipCodeField(detailsGrid[0]["Zip code"])
        .SendTextToMobilePhoneField(detailsGrid[0]["Phone"])
        .RegisterAccount();
    }

```

```

[Then(@"I should see '(*)' user is registered")]
public void ThenIShouldSeeUserIsRegistered(string userName)
{
    _authenticationPage.VerifyUserIsRegistered(userName);
}

```

```

[When(@"I Login as a user with '(*)' email and '(*)' password")]
public void WhenILoginAsAUserWithEmailAndPassword(string email, string password)
{
    _authenticationPage.SendTextToLoginEmailField(email)
        .SendTextToLoginPasswordField(password)
        .SignIn();
}

```

```

[Then(@"I should see '(*)' user is logged in")]
public void ThenIShouldSeeUserIsLoggedIn(string userName)
{
    _authenticationPage.VerifyUserIsRegistered(userName);
}

```

```

    }
}

```

appsettings.json

```

{
  "Settings": {
    "Browser": "chrome"
  }
}

```

Feature: Sign Up*As a user**I want to be able to sign in**So that I have an ability to do shopping***Background:**

Given I navigate to the main page on the site

@Sign in

Scenario: As a user I want to be able to create an account

When I navigate to the authentication page

And I create a user with the following information

| | | | | |
|---------------------|----------------------------|---------------------------|---------------------------|--------------------------|
| <i>Email</i> | <i>Personal First Name</i> | <i>Personal Last Name</i> | <i>Password</i> | |
| <i>Day of Birth</i> | <i>Month of Birth</i> | <i>Year of Birth</i> | <i>Address First Name</i> | <i>Address Last Name</i> |
| <i>Address</i> | <i>City</i> | <i>State</i> | <i>Zip code</i> | <i>Phone</i> |
| | mikhailmoroz2@gmail.com | Mikhail | Moroz | Qqwerty1!! |
| 18 | May | 1999 | Mikhail | Moroz |
| | St Main | Kharkiv | California | |
| 62002 | 0991231232 | | | |




Then I should see '*Mikhail Moroz*' user is registered

Додаток В

Відомість кваліфікаційної роботи

ГЮИК.501610.008

(позначення документу)

| № | Позначення | | | | Назва | Дол. відомості | |
|-------------------|--------------------------|----------------|---|-------------|---|---------------------------------|--------|
| | | | | | Текстові документи | | |
| 1. | ГЮНК.501610.008 ПЗ | | | | Пояснювальна записка | 113 с. | |
| 2. | ГЮНК.501610.008-01 12 01 | | | | Текст програми | 19 с. | |
| 3. | ГЮНК.501610.008 | | | | Специфікація | 2 с. | |
| | | | | | | | |
| | | | | | Графічні документи | | |
| 4. | ГЮНК.501610.008 С10 | | | | Життєвий цикл розробки програмного забезпечення | Включено до ПЗ | |
| 5. | | | | | Модель якості програмного забезпечення | Включено до ПЗ | |
| 6. | | | | | Процес забезпечення якості | Включено до ПЗ | |
| 7. | | | | | Структура веб-додатку | Включено до ПЗ | |
| 8. | | | | | Компоненти Selenium | Включено до ПЗ | |
| 9. | | | | | Архітектурні компоненти Selenium | Включено до ПЗ | |
| 10. | | | | | Робочий процес Selenium WebDriver | Включено до ПЗ | |
| 11. | | | | | Типи локаторів | Включено до ПЗ | |
| 12. | | | | | Взаємодія SpecFlow та BDD з Веб-додатком | Включено до ПЗ | |
| 13. | | | | | Тестовий сценарій реєстрації у Веб-додаток | Включено до ПЗ | |
| 14. | | | | | Згенерований клас визначених кроків з шаблонами методів | Включено до ПЗ | |
| 15. | | | | | Локатори елементів для реєстрації | Включено до ПЗ | |
| 16. | | | | | Реалізовані методи класу AuthenticationPage | Включено до ПЗ | |
| | | | | | ГЮНК.504310.008 ДЗ | | |
| <i>Зм.</i> | <i>Лист</i> | <i>№ докум</i> | <i>Підп.</i> | <i>Дата</i> | | | |
| <i>Розроб.</i> | | Мороз М.Ю. |  | | <i>Дослідження процесі контролю якості взаємодії користувачів з веб-додатком інтернет-магазину.</i> | Лист | Листів |
| <i>Перевірів.</i> | | Мінухін С.В. |  | | | 1 | 2 |
| <i>Н. Контр.</i> | | Мінухін С.В. |  | | | <i>ХНУРЕ Кафедра СТ</i> | |
| <i>Затверд.</i> | | Гребеннік І.В. | | | | | |

