МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ОТЧЕТ

по учебной практике

Тема: Тестирование сайта Wildberries

Студентка гр. 3341 _____ Игнатьев К.А.

Студентка гр. 3341 _____ Ступак А.А.

Студент гр. 3344 _____ Фоминых Е.Г.

Руководитель _____ Шевелева А.М.

Санкт-Петербург
2025

ЗАДАНИЕ
НА УЧЕБНУЮ ПРАКТИКУ

Студент Игнатьев К.А. группы 3341

Студент Ступак А.А. группы 3341

Студент Фоминых Е.Г. группы 3344

Тема практики: Тестирование сайта Ozon

Задание на практику: Написать 10 тестов для взаимодействия с маркетплейсом Wildberries.

Сроки прохождения практики: 25.06.2024 – 08.07.2024

Дата сдачи отчета: 07.07.2024

Дата защиты отчета: 07.07.2024

Студент	_____	Игнатьев К.А.
Студентка	_____	Ступак А.А.
Студент	_____	Фоминых Е.Г.
Руководитель	_____	Шевелева А.М.

АННОТАЦИЯ

Задание направлено на развитие навыков автоматизированного тестирования веб-приложений с использованием Java, библиотеки Selenide, фреймворка JUnit и системы сборки Maven. В качестве объекта тестирования выбрана система Wildberries - сервис для онлайн покупок и доставки товаров повседневного спроса.

В рамках работы реализован набор из десяти тестов, охватывающих основные пользовательские сценарии по созданию, управлению и взаимодействию с пинами, включая поиск по ключевым запросам. Каждый тест подробно документирован в виде чек-листа с входными данными, пошаговым описанием действий и критериями ожидаемого результата.

SUMMARY

The assignment is aimed at developing the skills of automated testing of web applications using Java, the Selenide library, the JUnit framework and the Maven build system. The Wildberries system, a service for visual search and content organization, was chosen as the testing object.

As part of the work, a set of ten tests has been implemented, covering the main user scenarios for creating, managing, and interacting with pins, including keyword search. Each test is documented in detail in the form of a checklist with input data, a step-by-step description of actions and criteria for the expected result.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Целью данной работы является создание набора автоматизированных UI-тестов для веб-приложения с использованием Java, Selenide, JUnit и Maven, обеспечивающего надёжную проверку пользовательских сценариев и поддерживающего единый стандарт документирования и логирования результатов прогонов.

В рамках задачи предстоит разработать и описать подробные тестовые сценарии для десяти ключевых функций системы: Проверка отображения названия, цены и изображения товара, проверка добавления товара в корзину, проверка изменения количества товара в корзине, включая исчезновение, проверка выбора характеристик товара(цвет), проверка добавления товара в избранное, проверка перехода к оформлению заказа после нажатия кнопки “Купить в один клик”, проверка отображения рейтинга товара и сортировки по рейтингу, проверка отправки жалобы, проверка отображения информации о доставке, проверка корректности работы раздела “рекомендуемые товары”.

1. РЕАЛИЗУЕМЫЕ ТЕСТЫ

В данном разделе представлен перечень ключевых функциональных тестов для работы с пином на Wildberries. Для каждого случая описывается последовательность действий, осуществляемых пользователем для выполнения того или иного действия.

Реализуемые тесты представлены в табл. 1.

Таблица 1 – Реализуемые тесты

№	Название теста	Что делает тест
1	Проверка отображения названия, цены и изображения товара	Проверяет правильное отображение товара при вводе артикула
2	Проверка добавления товара корзину	Проверяет, что выбранный товар успешно добавляется в корзину и отображается в ней.
3	Проверка изменения количества товара, включая исчезновение	Проверяет изменение количества товара в корзине и его удаление при уменьшении количества до нуля.
4	Проверка выбора характеристик товара (цвет)	Проверяет возможность выбора характеристики товара (цвета) — конкретно, выбор чёрного цвета.
5	Проверка добавления товара в избранное	Проверяет, что товар успешно добавляется в избранное и отображается в соответствующем списке.
6	Проверка перехода к оформлению заказа после нажатия кнопки “Купить в 1 клик”	Проверяет, что при нажатии кнопки “Купить в 1 клик” происходит переход на страницу оформления заказа
7	Проверка отображения рейтинга товара и сортировки по рейтингу	проверяет отображение рейтинга товара и корректную работу сортировки по высокому рейтингу в каталоге.
8	Проверка отправки жалобы	проверяет возможность отправки жалобы на отзыв и отображение уведомления об успешной отправке.
9	Проверка отображения информации о доставке	Проверяет возможность поделиться пином через личное сообщение: после авторизации пользователь выбирает пин, вводит имя получателя и отправляет его.

10	Проверка корректности работы раздела “рекомендуемые товары”	Проверяет наличие и отображение раздела «Рекомендуемые товары» на странице выбранного товара.

2. ОПИСАНИЕ КЛАССОВ И МЕТОДОВ ЭЛЕМЕНТОВ СТРАНИЦ

Все классы элементов страниц наследуются от базового класса *BaseElement*, хранящего в себе *SelenideElement; baseElement* - DOM-элемент страницы, с которым происходят дальнейшие действия. Кроме этого в нем представлены методы: *isDisplayed()* - проверка, отображается ли элемент на странице, и *getAttribute(..)* - метод, возвращающий заданный аттрибут элемента. UML-диаграмма наследуемых от *BaseElement* элементов представлена на рис. А.1.

Все классы элементов имеют поля констант вида статические методы вида *by{param_name}(..)* возвращающие объект класса по заданным параметрам.

Базисные элементы:

1. *AElement* - класс элемента гиперссылки(<a>), наследуемый от *BaseElement*. В класс добавлены методы: *click()* - нажатие на ссылку, *getText()* - получение текста ссылки, *getAttribute(String s)* - получение аттрибута ссылки.
2. *ButtonElement* - класс элемента кнопки, наследуемый от *BaseElement*. В класс добавлены метод: *click()* - нажатие на ссылку.
3. *CanvasElement* - класс элемента Canvas изображения, наследуемый от *BaseElement*. В класс добавлены метод: *isLoaded()* - проверка загрузки изображения.
4. *DivElement* - класс элемента контейнера, наследуемый от *BaseElement*. В класс добавлены методы: *getText()* - получение текста в контейнере, *elementAppeared()* - проверка появления контейнера.
5. *HElement* - класс элемента заголовка первого уровня, наследуемый от *BaseElement*. В класс добавлены методы: *getText()* - получение текста в контейнере.
6. *HTwoElement* - класс элемента заголовка второго уровня, наследуемый от *BaseElement*. В класс добавлены методы: *getText()* - получение текста в контейнере.

7. *InputElement* - класс элемента ввода (кроме кнопок), наследуемый от BaseElement. Является основой для элементов ввода. Реализованы методы: *setValue()* - устанавливает значение, *getValue()* - получает текущее значение.
8. *SpanElement* – класс элемента строкового контейнера, наследуемый от *BaseElement*. В классе реализованы методы: *getText()* - для получения текста элемента, *click()* - для нажатия на элемент.

Композитные элементы:

AutocompliteElement - класс элемента автодополнения при поиске, наследуется от *BaseElement*. Содержит метод удаления себя *deleteAutocomplete().*

CardElement - класс элемента карточки товара, наследуется от *BaseElement*. Содержит методы получения имени товара *getCardName(),* переходав к продукту *goToProduct()*, получения рейтинга *getRating().*

DeliveryMethodElement - класс элемента формы выбора адреса доставки, наследуемый от *BaseElement*. Содержит метод *addAddress()* для перехода на страницу добавления адреса.

DropDownMenuElement — класс элемента выпадающего меню, наследуемый от *BaseElement*. Содержит методы *openDropDownMenu()* для открытия меню и *chooseDropDownVariant()* для выбора конкретного пункта меню.

FeedBackFormElement - класс элемента формы выбора жалобы, наследуемый от *BaseElement*. Внутри реализованы методы *select()* - для выбора жалобы и *send()* - для отправки жалобы.

LoadingElement - класс элемента анимации загрузки страницы, наследуемый от *BaseElement*. Имеет метод *waitUntilLoad()* - для ожидания загрузки.

MapSearchElement — поиска элемента на карте, наследуемый от *BaseElement*. Реализованы методы *setAddress()* для установки адреса и *findAddress()* для поиска адреса.

OverlayElement - класс элемента оверлея при поиске, наследуется от *BaseElement*. Содержит метод удаления себя *deleteOverlay().*

ProductInCartElement - класс элемента товара в корзине, наследуется от *BaseElement*. Содержит методы *increaseProductCount()* - для увеличения количества товара на 1, *decreaseProductCount()* - для уменьшения количества товаров на 1, *deleteProduct()* - для удаления товара из корзины, *getProductCount()* - для получения количества товара.

ProductInFavoritesElement - класс элемента товара в избранном, наследуется от *BaseElement*. Содержит метод проверки отображения *isProductInFavorites().*

RecomendedProductsElement - класс элемента набора карточек товаров, наследуется от *BaseElement.* Содержит методы проверки отображения карточки *cardIsVisible(),* перехода к товару на карточке *goToProduct(),* получения имени карточки *getCardName(),* пролистывания страницы до набора *scrollToRecommends().*

ReviewElement - класс элемента отзыва, наследуется от BaseElement. Содержит методы нажатия на кнопку жалобы feedBackButton(), нажатия на кнопку открытия формы жалобы feedBackButton().

SearchElement - класс элемента поисковой строки, наследуется от BaseElement. Содержит методы вставки запроса insertRequest(String request), удаления автодополнения autocompleteDelete(), удаления оверлея deleteOverlay(), нажатия на кнопку поиска pressSearchButton().

3. ОПИСАНИЕ КЛАССОВ И МЕТОДОВ СТРАНИЦ

Все классы страниц наследуются от базового класса *BasePage*, хранящего в себе *Class<? extends BasePage> pageClass* - класс страницы для возвращения объекта текущей страницы. Также внутри хранится *String expectedUrlPart* - часть URL, которая должна быть на конкретной странице. Кроме этого в нем представлены методы: *refresh()* - обновление страницы, *page(..)* - создание нового объекта класса. UML-диаграмма наследуемых от BasePage классов страниц представлена на рис. А.4.

BuyPage - класс страницы оформления заказа, наследуемый от *BasePage*. Содержит название товара, кнопку выбора способа доставки, форму выбора адреса доставки и элемент и выбранный адрес, а также методы: *openDeliveryMethod()* - открывает форму выбора адреса доставки, *addAddress()* - открывает страницу добавления нового адреса, *getDeliveryAddress()* - возвращает выбранный адрес, *isDisplayed()* - проверяет загрузку страницы.

CartPage - класс страницы корзины, наследуемый от *BasePage*. Содержит продукт корзины, а также методы: *increaseProductCount()* - увеличивает количество продукта на 1, *decreaseProductCount()* - уменьшает количество продукта в корзине на 1, *deleteProduct()* - удаляет продукт из корзины, *getProductCount*() - возвращает количество продукта, *isProductInCart()* - проверяет наличие продукта в корзине.

FavoritesPage - класс страницы избранного, наследуемый от *BasePage*. Хранит поле продукта в избранном, а также метод для проверки наличия товара в избранном *isProductInFavorites()*.

MainPage - класс главной страницы, наследуемый от *BasePage*. Содержит элементы поисковой строки, карточки товара, загрузки страницы, гиперссылки на авторизацию, поле ввода номера телефона, кнопки отправки кода, кнопки закрытия рекламы. Содержит методы проверки загрузки *isDisplayed()*, вставки поискового запроса *insertSearchRequest()*, нажатия на кнопку поиска *clickOnSearchButton()*, ожидания загрузки страницы

waitUntilLoad(), удаления оверлея *deleteOverlay()*, удаления автодополнения deleteAutocomplete(), откртия новой страницы *openNewPage()*, авторизации *login()*.

MapPage() - класс страницы поиска и выбора адреса, наследуемый от BasePage. Содержит элементы подтверждения выбора адреса, выбранного адреса доставки и поисковой строки данной страницы. Содержит методы вставки поискового запроса *setAddress()*, поиска адреса по запросу *searchAddress()*, получения выбранного адреса *getAddress()*, выбора адреса *selectAddress()*, подтверждения выбора адреса *confirmAddress()*.

ProductPage - класс страницы товара, наследуемый от BasePage. Содержит элементы названия товара, его цены, его изображения, кнопку добавления в корзину, гиперссылку для перехода в корзину, гиперссылку для изменения цвета товара, кнопку добавления в избранное, текстовый контейнер с рейтингом, гиперссылку для перехода в избранное, кнопку «Купить сейчас», набор реккомендованных товаров, гиперссылку для открытия отзывов. Содержит методы проверки отображения имени товара *isProductNameDisplayed()*, получения имени товара *getProductName()*, проверки отображения цены товара *isProductPriceDisplayed()*, получения цены товара *getProductPrice()*, проверку загрузки изображения товара *isCanvasImageLoaded()*, добавления товара в корзину *addToCart()*, перехода в корзину *goToCart()*, изменения цвета товара *changeColor()*, добавления товара в избранное *addToFavorites()*, перехода в избранное *goToFavorites()*, проверяет отображение рейтинга *isRatingDisplayed()*, получает рейтинг *getRating()*, переходит на страницу оформления заказа на этот товар *buyNow()*, открывает новую страницу *openNewPage()*, пролистывает страницу до набора рекомендаций *scrollToRecomends()*, проверяет отображение карточки товара *isCardVisible()*, проверяет отображение набора рекомендованных товаров *isRecommendedProductsVisible()*, переходит на страницу товара из рекомендаций *goToProduct()*, получает имя товара из рекомендаций *getCardName()*, открывает страницу отзывов *openReviews()*.

ReviewPage - класс страницы отзывов, наследуемый от BasePage. Хранит отзыв, форму выбора жалобы, контейнер с всплывающем уведомлением об отправке отзыва, а также методы: *feedBack()* и *openForm()* - два метода для открытия формы выбора жалобы, которые должны выполняться последовательно из-за особенности верстки, *selectElement()* - для выбора пункта из формы жалоб, *send()* - для отправки жалобы, isAppears — проверяет появление всплывающего уведомления в течение заданного времени.

SearchPage - класс страницы с результатами, наследуемый от BasePage. Имеет элемент всплывающего меню, а так же методы для открытия всплывающего меню *openDropDown()*, выбора режима сортировки *chooseFilter()*, и получения рейтингов первых 20-ти элементов *getRatings()*.

4. ОПИСАНИЕ КЛАССОВ И МЕТОДОВ ТЕСТОВ

Все классы тестов наследуются от базового класса BaseTest, хранящего в себе URL тестируемого сайта и артикул товара. Кроме этого в нем представлен метод с аннотацией @Before, что означает, что данный метод должен выполняться перед каждым тестом: init() - настройка браузера, в котором будет происходить UI-тестирование. Также представлен метод с аннотацией @After, что означает, что данный метод должен выполняться после каждого теста: tearDown() - закрытие браузера. Так же представлены методы: setUp(), который выполняет настройку браузера и вызывается в init(), start(), который выполняет загрузку главной страницы, авторизацию при необходимости, поиска по артикулу search(), и получение страницы товара getProductPage() UML-диаграмма наследуемых от BaseTest тестов представлена на рис. А.6.

AddProductToFavoritesTest - класс, наследуемый от BaseTest, хранящий в себе метод addProductToFavoritesTest(), объединяющий последовательность действий: открытие страницы товара, добавление товара в избранное, переход на страницу избранного и проверку добавления товара в избранное.

AddToCartTest - класс, наследуемый от BaseTest, хранящий в себе метод addToCartTest(), объединяющий последовательность действий: открытие страницы товара, добавление товара в корзину, переход на страницу корзины, и проверку добавления товара в корзину.

BuyNowTest - класс, наследуемый от BaseTest, хранящий в себе метод buyNowTest(), объединяющий последовательность действий: открытие страницы товара, нажатие на кнопку “Купить сейчас”, открытие страницы оформления товара, проверку загрузки страницы.

ChangeProductColorTest - класс, наследуемый от BaseTest, хранящий в себе поле anotherColorProductName, которое хранит название товара, на который нужно перейти. Метод changeProductColorTest(), объединяющий последовательность действий: открытие страницы товара, нажатие на

гиперссылку с другим цветом товара, переход на страницу другого товара, сравнение ожидаемого названия и реального.

ChangeProductCountTest - класс, наследуемый от BaseTest, хранящий в себе поля: expectedProductCountAfterEncrease, которое хранит ожидаемое количество товара после увеличения, expectedProductCountAfterDecrease, которое хранит ожидаемое количество товара после уменьшения. Методы:

1. changeProductCountTest(), который открывает страницу товара, добавляет его в корзину, нажимает на кнопку перехода в корзину, открывает страницу корзины, увеличивает число элементов, сравнивает ожидаемое число с реальным.
2. decreaseProductCountTest(), который открывает страницу товара, добавляет его в корзину, нажимает на кнопку перехода в корзину, открывает страницу корзины, уменьшает число элементов, сравнивает ожидаемое число с реальным.
3. deleteProductTest(), который открывает страницу товара, добавляет его в корзину, нажимает на кнопку перехода в корзину, открывает страницу корзины, удаляет товар, проверяет удаление товара.

DeliveryAddressTest - класс, наследуемый от BaseTest, хранящий в себе поле originalAddress, которое хранит адрес, который метод будет вводить в поле поиска. Метод deliveryAddressTest(), объединяющий последовательность действий: открытие страницы товара, нажатие на кнопку “Купить сейчас”, переход на страницу оформления заказа, открытие формы выбора адреса, нажатие на кнопку добавления адреса, переход на страницу поиска адреса, установку originalAddress в поле поиска адреса, нажатие на кнопку поиска адреса, получение ближайшего адреса к указанному, сохранение его значения для сравнения в переменную selectedAddress, подтверждение выбора адреса, переход на страницу оформления заказа, получение выбранного адреса, сравнение его с selectedAddress.

FindNamePriceImgTest - класс, наследуемый от BaseTest, хранящий в себе поля: name, которое хранит ожидаемое название товара, price, которое

хранит ожидаемую цену товара, width и height, которые хранят размеры canvas изображения. Методы:

1. findName(), который открывает страницу товара, проверяет отображение названия товара и его имя.
2. findPrice(), который открывает страницу товара, проверяет отображение цены товара и ее значение.
3. findImg(), который открывает страницу товара, проверяет загрузку изображения.

RatingTest - класс, наследуемый от BaseTest, хранящий в себе поле: searchStr, которое хранит поисковой запрос. Методы:

1. productRatingTest(), который открывает страницу товара, проверяет отображение рейтинга товара и его значение.
2. filterRatingTest() - открывает главную страницу, вставляет поисковой запрос в поисковую строку, нажимает кнопку поиска, открывает страницу с результатами поиска, открывает выпадающее меню, выбирает пункт “По рейтингу”, собирает первые 20 рейтингов из карточек, проверяет их последовательно с помощью ratingInRange().
3. ratingInRange() - проверяет, находится ли рейтинг в диапазоне от 0 до 5.

RecommendationsTest - класс, наследуемый от BaseTest, хранящий в себе методы:

1. recommendationsVisibleTest() - открывает страницу товара, пролистывает страницу до раздела с рекомендованными товарами, проверяет отображение раздела
2. cardVisibleTest() - открывает страницу товара, пролистывает страницу до раздела с рекомендованными товарами, проверяет карточки в разделе.
3. CardOpenTest - открывает страницу товара, пролистывает страницу до раздела с рекомендованными товарами, сохраняет в cardName название товара из карточки, открывает страницу товара из карточки, сравнивает cardName и реальное название товара.

ReviewTest - класс, наследуемый от BaseTest, хранящий в себе метод reviewTest(), объединяющий последовательность действий: открытие страницы товара, нажатие на гиперссылку открытия страницы отзывов, переход на страницу отзывов, открытие формы выбора жалоы, выбор пункта жалобы, отправка жалобы, проверка появления всплывающего уведомления об успешной отправке жалобы.

5. ОПИСАНИЕ ДОПОЛНИТЕЛЬНЫХ КЛАССОВ И МЕТОДОВ

GetNoty — класс, выполняющий перехват уведомления и извлечение кода авторизации из него. Содержит методы:

1. *getNoty()*
    1. Получает код из системных уведомлений путем мониторинга DBus событий.
    2. Запускает процесс dbus-monitor, анализирует вывод в реальном времени
    3. Извлекает цифровой код по шаблону и возвращает его.
2. *startDbusMonitorProcess()*
    1. Создает и запускает процесс dbus-monitor для отслеживания уведомлений.
3. *startOutputProcessingThread()*
    1. Запускает поток для обработки вывода процесса в реальном времени.
4. *processOutputLines()*
    1. Обрабатывает поток вывода построчно, фильтруя и анализируя строки.
5. *checkForCodeMatch()*
    1. Проверяет строку на соответствие шаблону и извлекает код при совпадении.
    2. При обнаружении кода сохраняет его, выводит строку и завершает процесс.
6. *waitForProcessTermination()*
    1. Ожидает завершения процесса с ограничением по времени.

5. UML-ДИАГРАММЫ СВЯЗИ КЛАССОВ И МЕТОДОВ

Разработанный программный код представлен в [1].

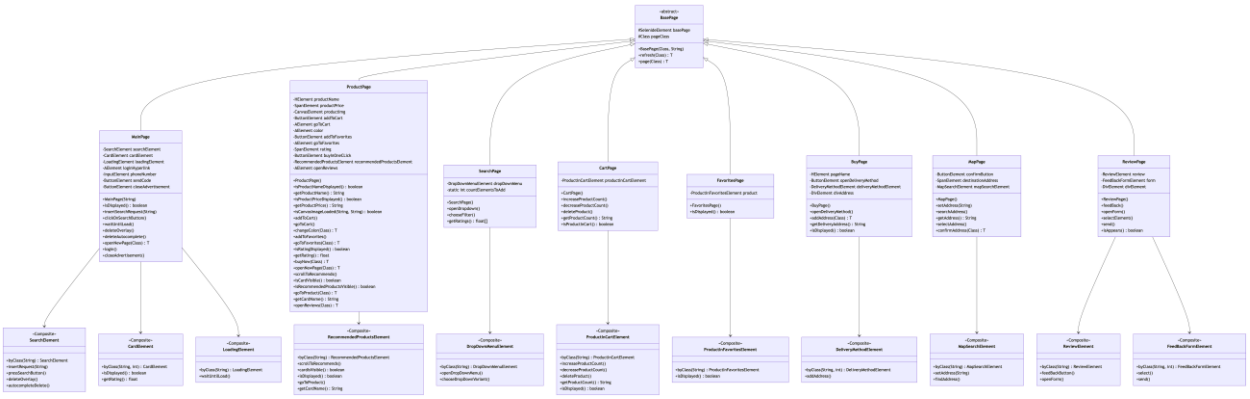

Рисунок 1 - UML-диаграмма страниц

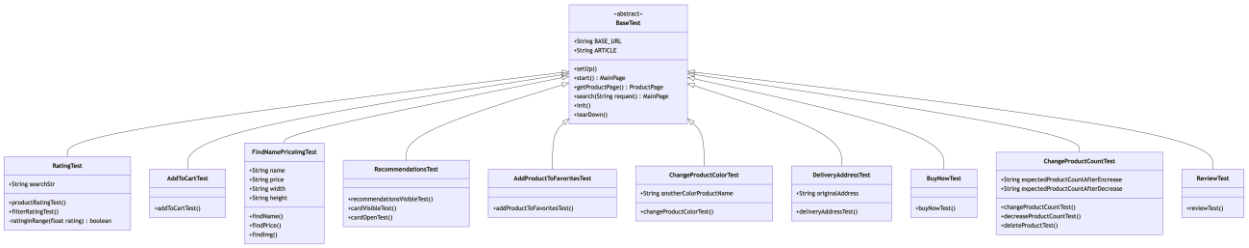

Рисунок 2 - UML-диаграмма тестов

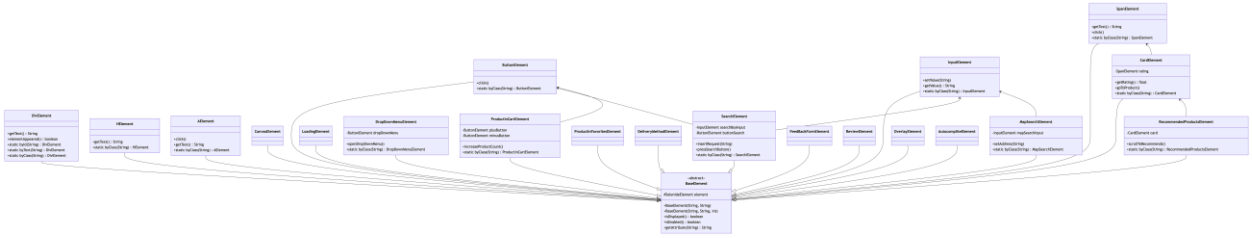

Рисунок 3 — UML-диаграмма элементов

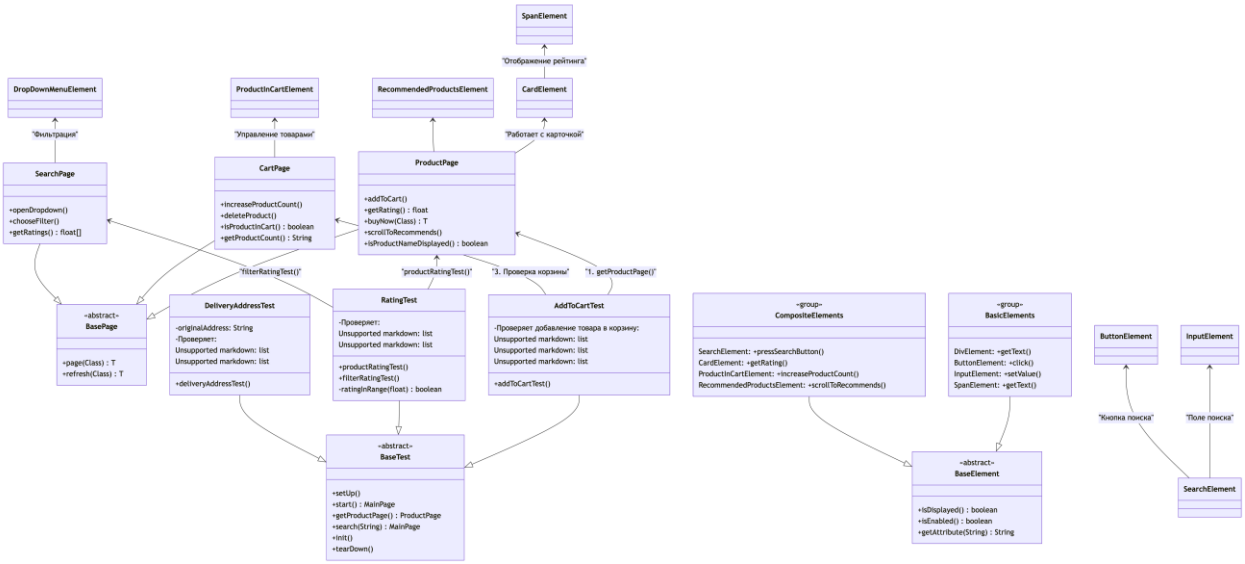

Рисунок 4 – UML-диаграмма общая

6. ТЕСТИРОВАНИЕ

6.1. Тестирование Добавления товара в корзину

Работа теста на примере теста добавления товара в корзину.

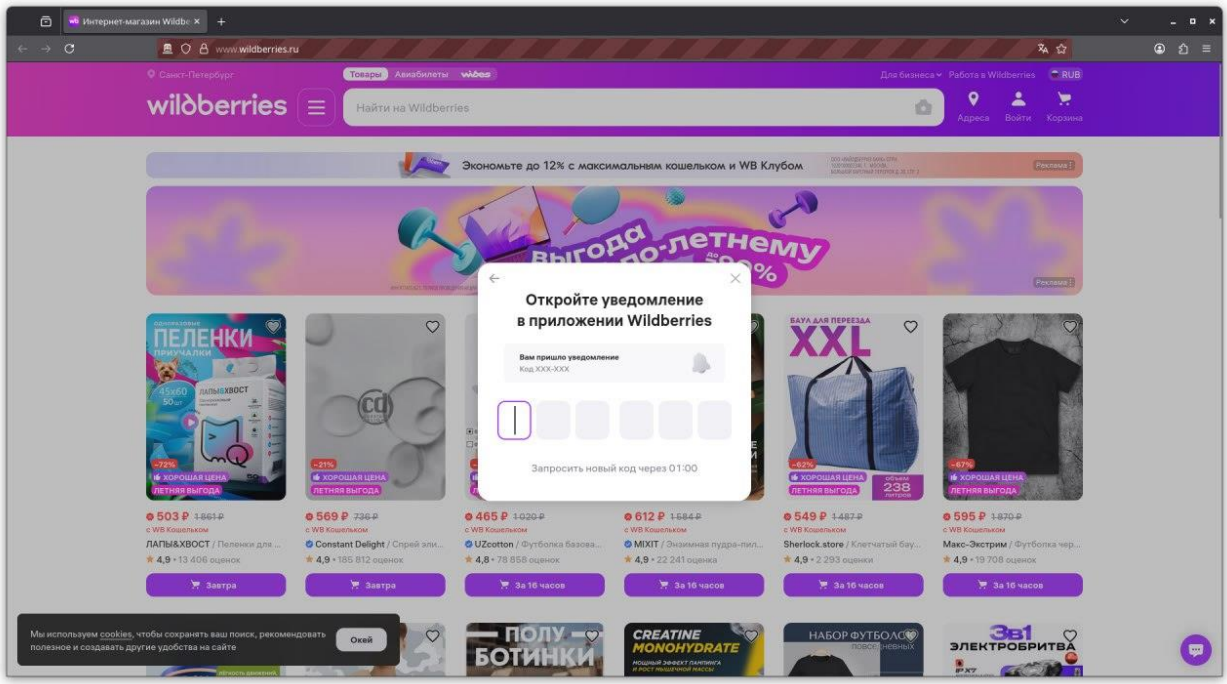

Рисунок 5 – Вход в учетную запись.

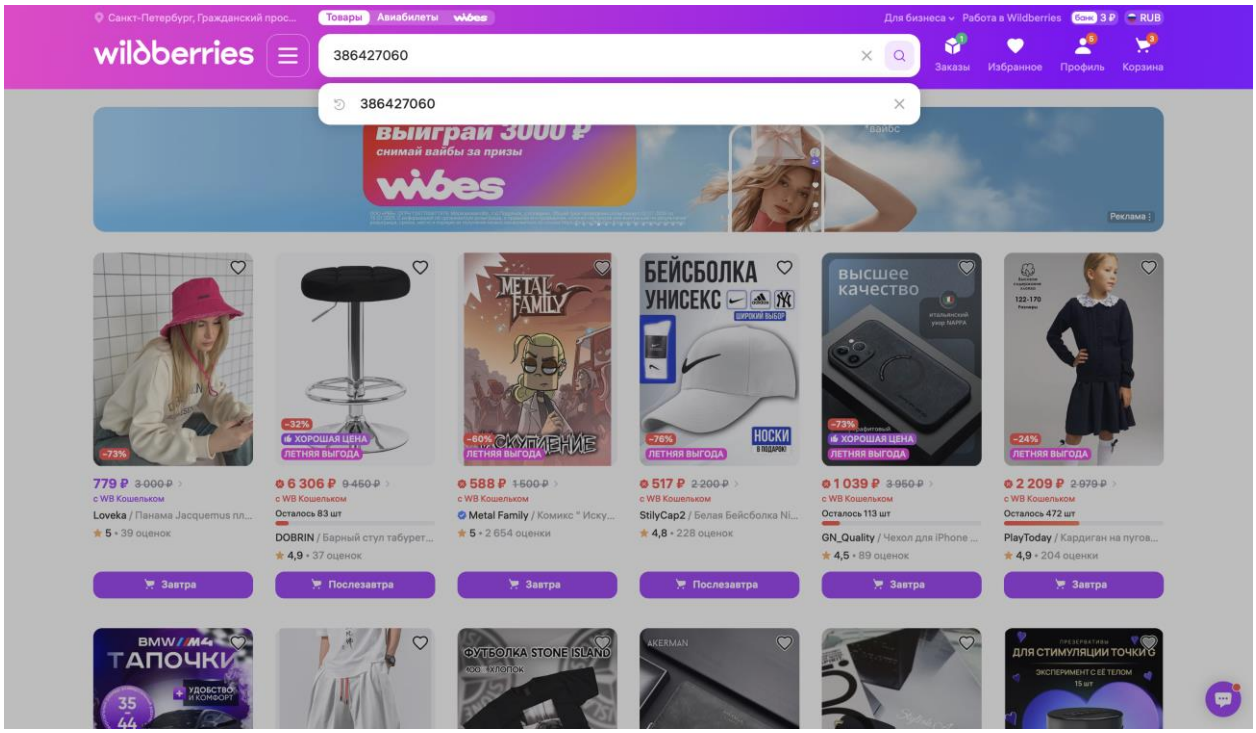

Рисунок 6 – Ввод артикула в строку поиска.

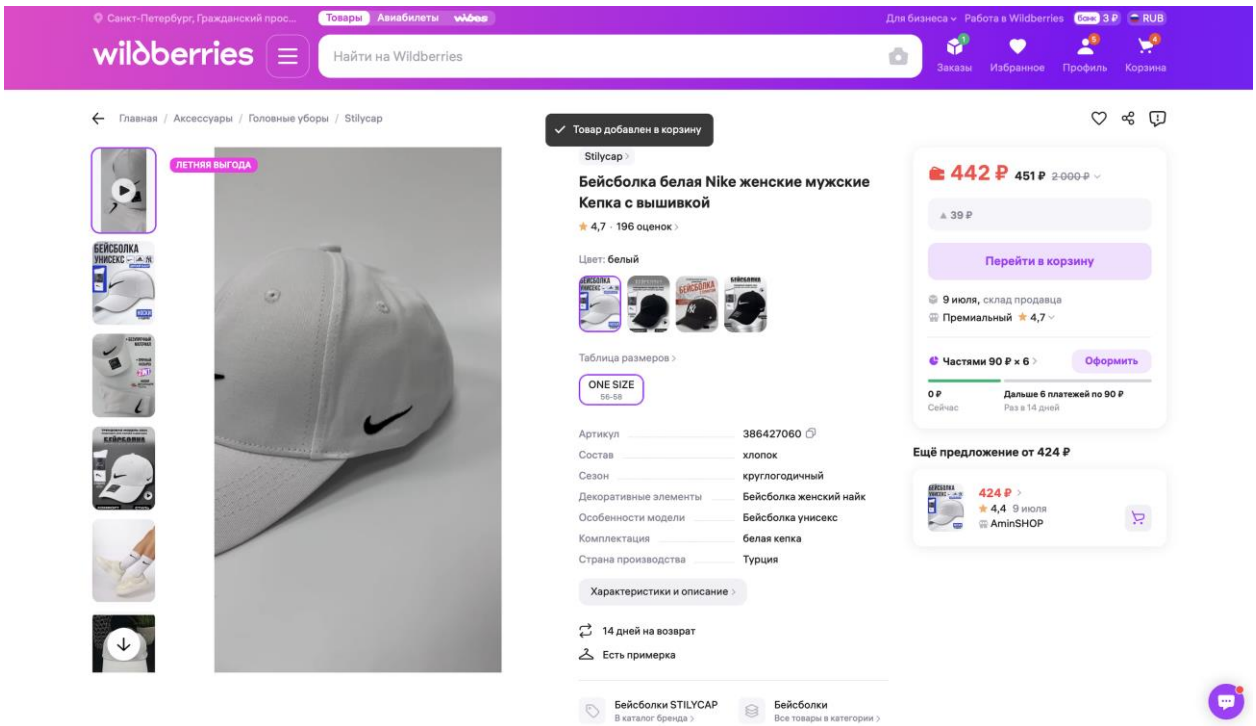

Рисунок 7 – нажатие кнопки “Добавить в корзину”

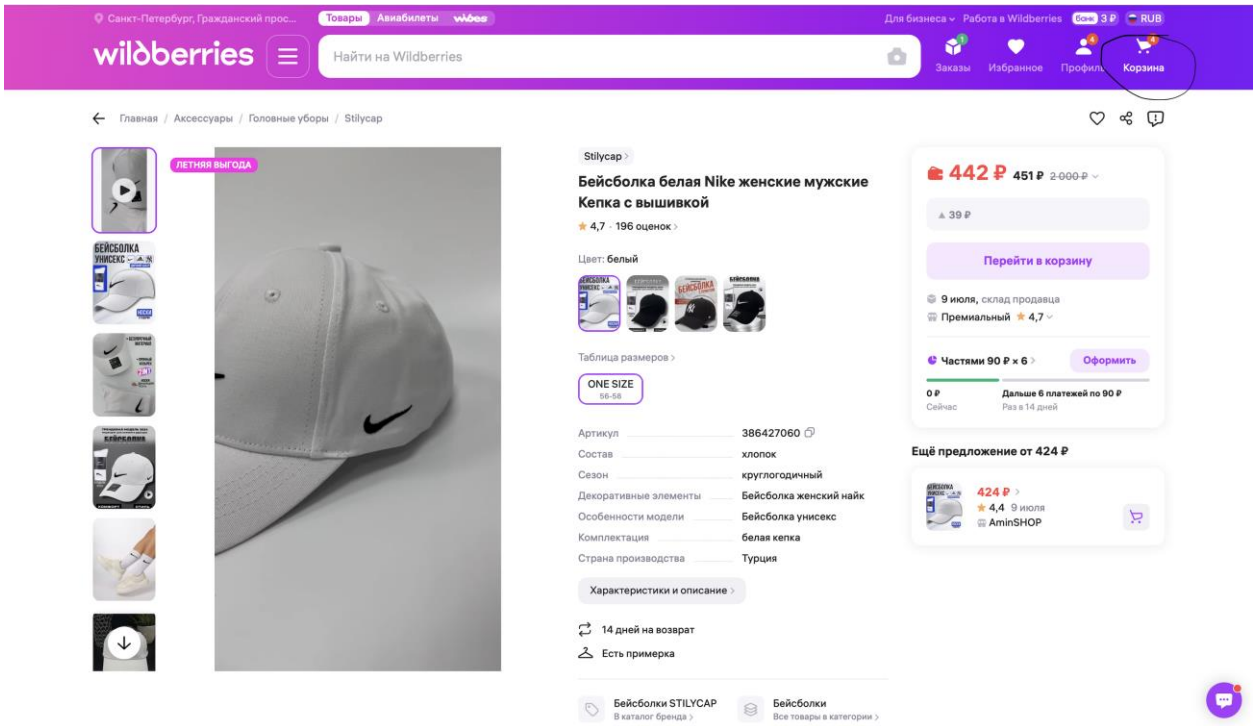

Рисунок 8 – Нажатие на значек “Корзина”

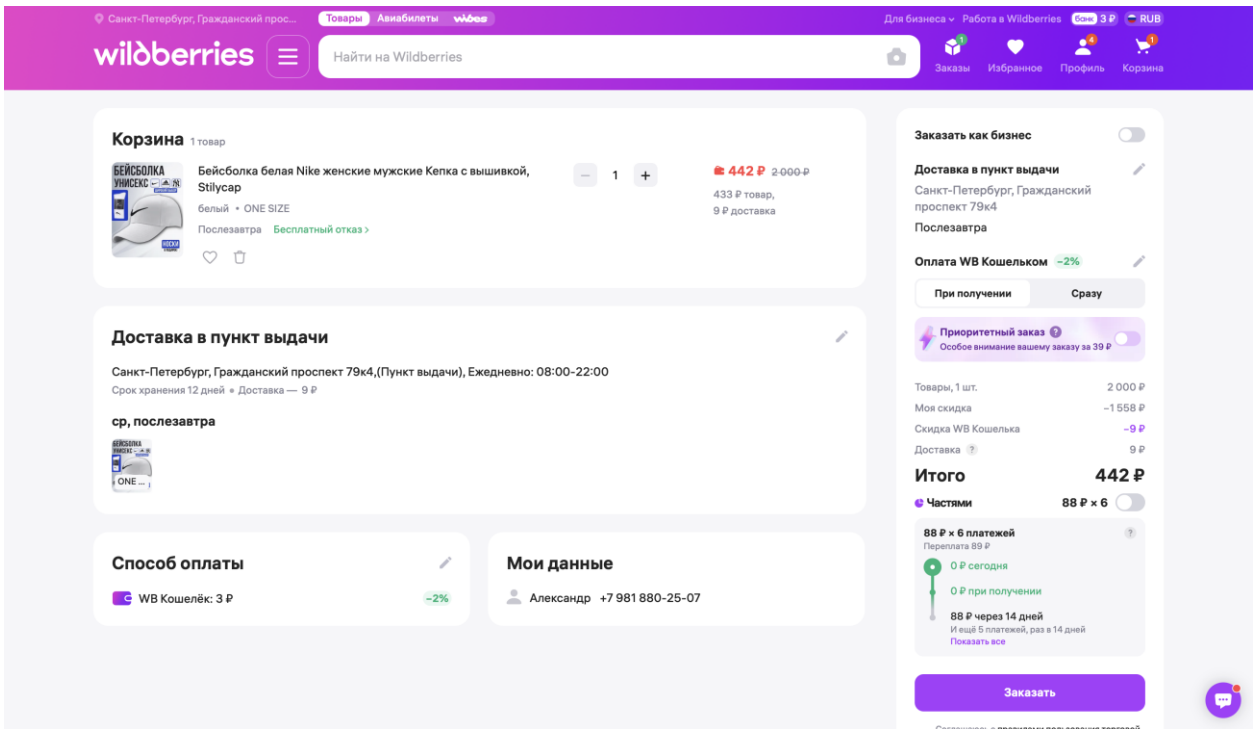

Рисунок 9 – Проверка наличия товара в корзине

6.2. Скриншоты успешного выполнения тестов

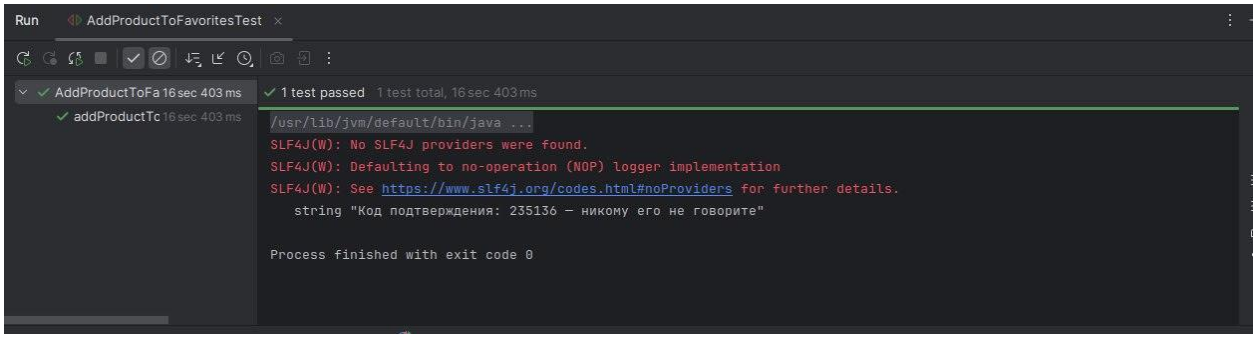

Рисунок 12 – AddProductToFavoritesTest

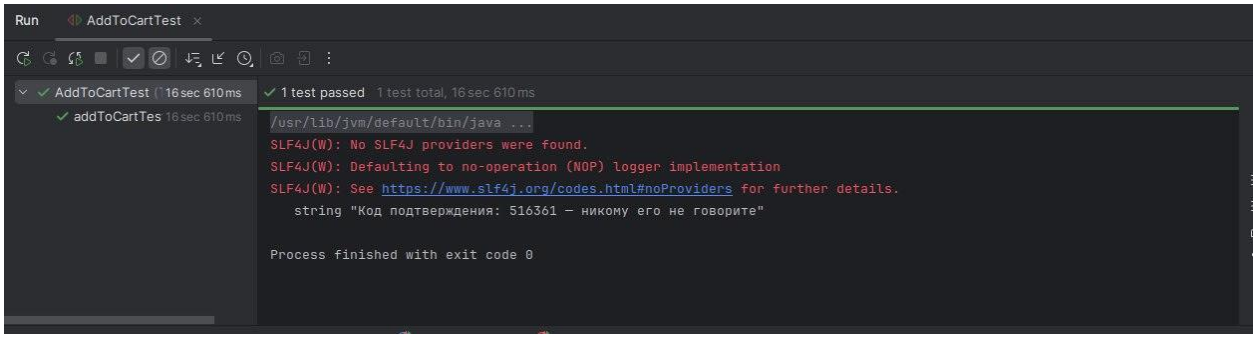

Рисунок 13 – AddToCartTest

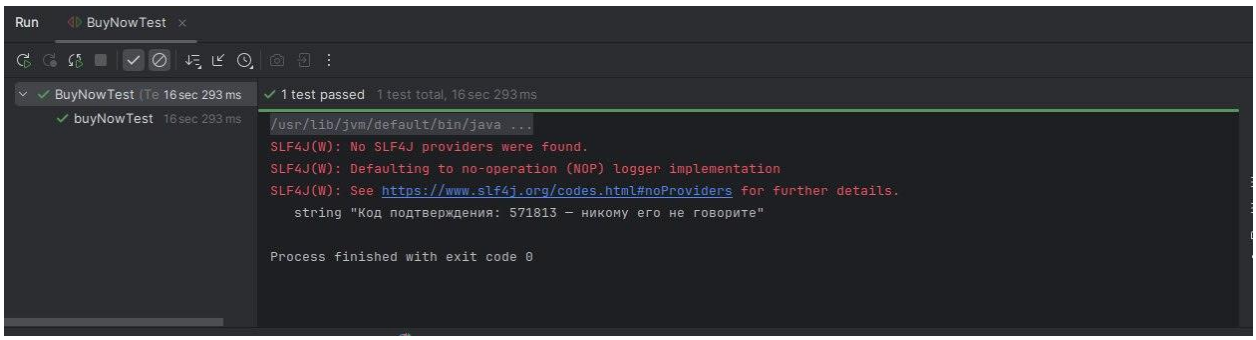

Рисунок 14 – BuyNowTest

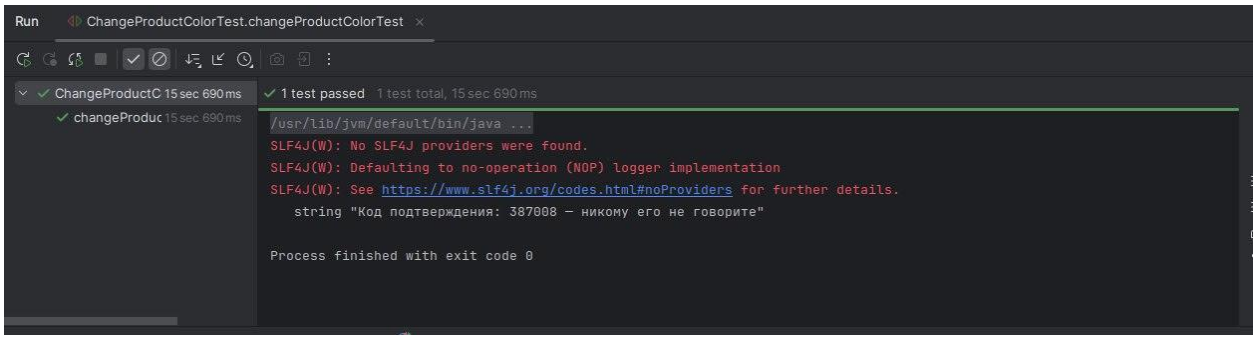

Рисунок 15 – ChangeProductColorTest

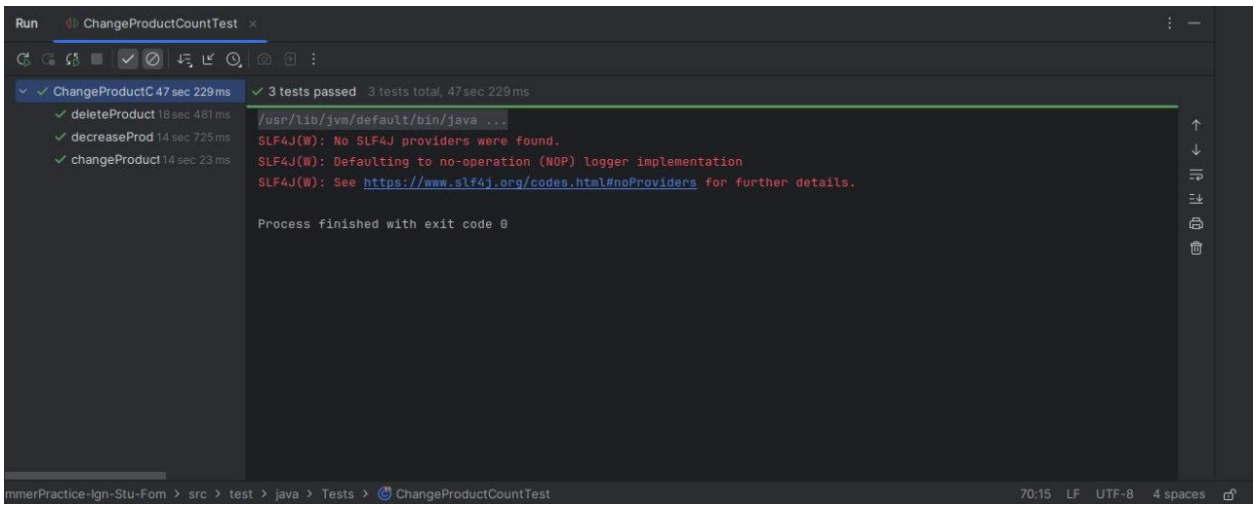

Рисунок 16 – ChangeProductCountTest

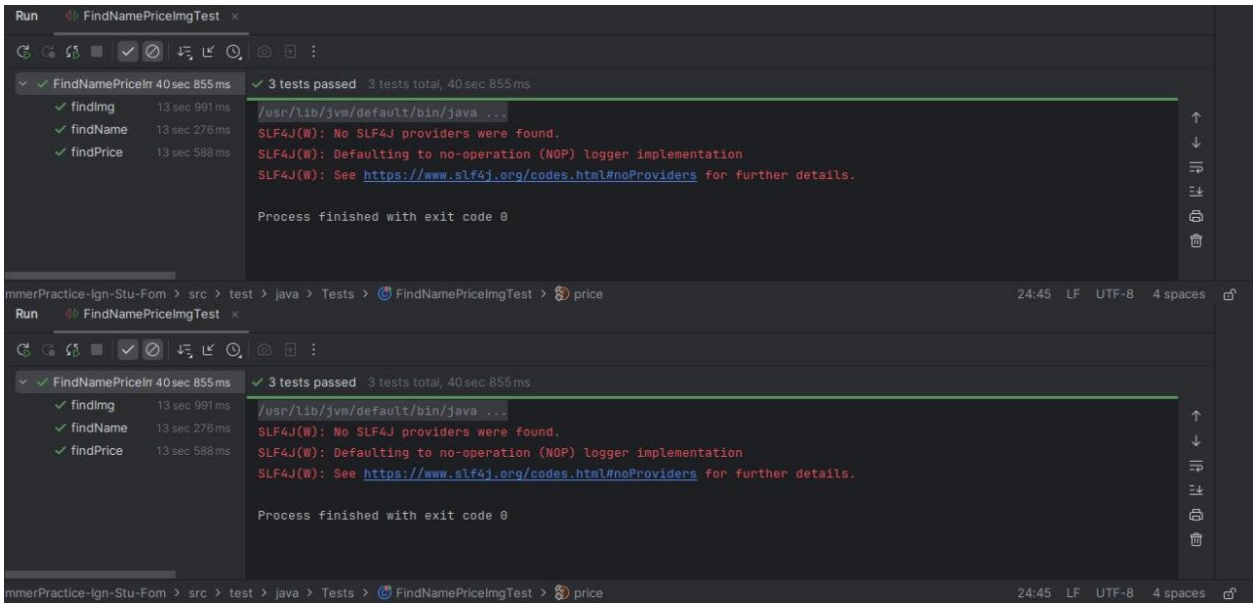

Рисунок 17 – FindNamePriceImgTest

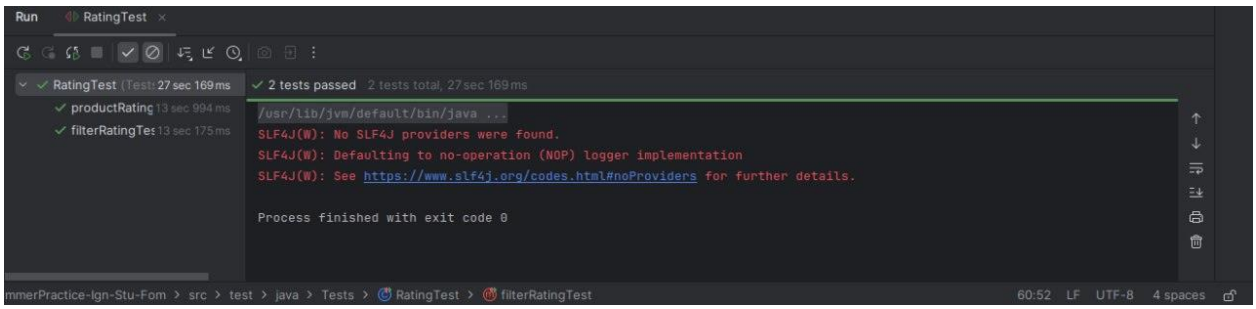

Рисунок 18 – RatingTest

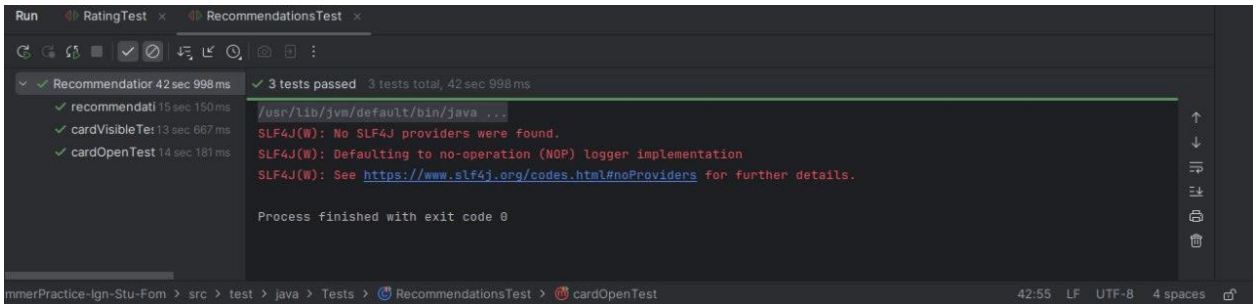

Рисунок 19 – RecomendationsTest

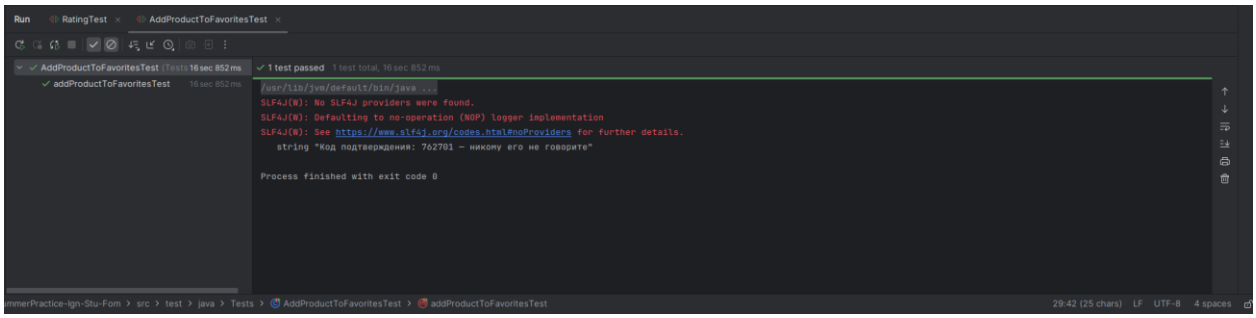

Рисунок 20 – AddProductToFavoritesTest

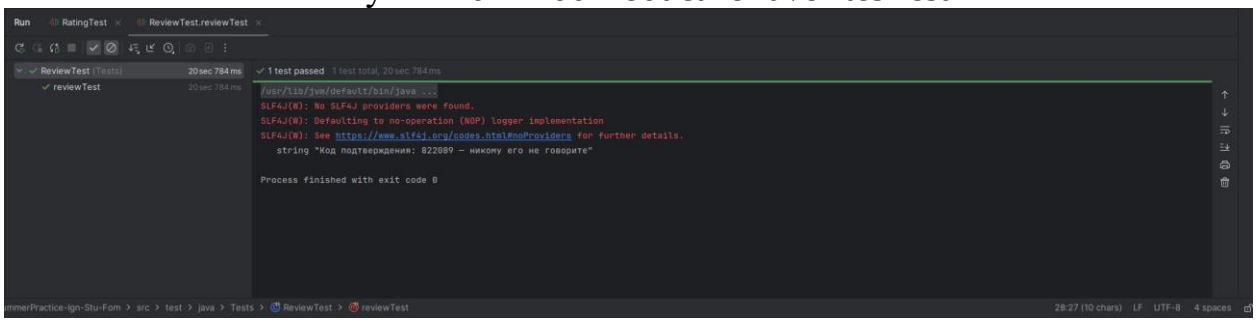

Рисунок 21 – ReviewTest

ЗАКЛЮЧЕНИЕ

В ходе выполнения работы была успешно создана и документирована система автоматизированного тестирования веб-приложения Wildberries с использованием Java, библиотеки Selenide, фреймворка JUnit и системы сборки Maven. В качестве основы был разработан и реализован комплекс UI-тестов, охватывающих ключевые пользовательские сценарии: от поиска товаров и фильтрации по рейтингу до добавления товаров в корзину и избранное, оформления доставки, смены характеристик товара, работы с отзывами и блоком рекомендаций. Каждый тест оформлен в виде чек-листа с чётко заданными входными данными, последовательностью действий и критериями ожидаемого результата, что обеспечивает прозрачность воспроизводимости тестов и надёжность выявления дефектов.

При реализации фреймворка использован подход **Page Object Model**. Для инкапсуляции элементов интерфейса создана универсальная обёртка BaseElement, а для моделирования страниц — базовый класс BasePage. Такой подход обеспечивает структурированность, повторное использование кода и удобство расширения функционала при изменениях интерфейса Wildberries. Классы элементов разделены на базовые и композитные, что позволяет гибко моделировать как простые, так и сложные составные элементы, например, карточки товара, блоки отзывов.

Интеграция с **Maven** и **JUnit** обеспечивает удобную сборку и запуск тестов, а также автоматическое формирование подробных отчётов о результатах. Использование возможностей **Selenide** по гибким ожиданиям гарантирует устойчивость тестов к сетевым задержкам и динамической загрузке элементов интерфейса.

Результатом работы является гибкий и поддерживаемый тестовый фреймворк для Wildberries, который может быть расширен новыми

сценариями без значительных затрат времени и ресурсов. Созданная архитектура обеспечивает удобство поддержки тестов при обновлениях сайта и служит основой для развития CI/CD процессов автоматизированного тестирования на реальных проектах.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Гитхаб репозиторий // URL: https://github.com/IgnatevKirill3341/SummerPractice-Ign-Stu-Fom.git