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

ОТЧЕТ

По «UI-тестирование» практике

Тема: Тестирование Яндекс Маркета

Студенты гр. 3388 _____ Глебова В.С
Раутио И.А
Басик В.В

Руководитель _____ Шевелева А.М

Санкт-Петербург
2025

ЗАДАНИЕ

НА «UI-ТЕСТИРОВАНИЕ» ПРАКТИКУ

Студенты: Глебова В.С, Раутио И.А, Басик В.В

Группа: 3388

Тема практики: Тестирование Яндекс Маркета

Задание на практику:

Нужно написать 10 тестов по одному определенному блоку / функционалу системы. Например, работа с постами в вк – создание поста, поделиться постом на своей странице, добавить комментарий к посту, лайкнуть пост, поделиться постом в сообщении, удалить пост, закрепить пост, добавить в архив, отключить комментарии.

Сроки прохождения практики: 25.06.2025 – 08.07.2025

Дата сдачи отчета: 07.07.2025

Дата защиты отчета: 07.07.2025

Студенты _____ Глебова В.С, Раутио И.А, Басик В.В

Руководитель _____ Шевелева А.М

АННОТАЦИЯ

Целью данной практики является освоение автоматизации тестирования веб-приложений с использованием современных технологий: Java, Selenide (на основе Selenium), JUnit для написания тестов и Maven как системы управления проектом. В рамках практики разработано 10 автотестов на определенный функциональный блок выбранной системы (например, работа с постами в социальной сети). Работа ведётся в группах по три человека через GitHub, где каждый участник выполняет коммиты и несёт ответственность за определённую часть проекта. Тесты должны быть задокументированы в формате чеклиста с описанием действий, входных данных и ожидаемых результатов. Также предусмотрено логирование выполнения тестов. Итоговый отчет включает описание реализации, UML-диаграммы, демонстрацию работы тестов и заключение по результатам практики

SUMMARY

The goal of this practice is to master the automation of web application testing using modern technologies: Java, Selenide (based on Selenium), JUnit for writing tests, and Maven as a project management system. Within the scope of the practice, 10 automated tests have been developed for a specific functional block of a selected system (for example, working with posts in a social network). The work is carried out in groups of three people via GitHub, where each participant makes commits and is responsible for a certain part of the project. The tests must be documented in a checklist format, including descriptions of actions, input data, and expected results. Logging of test execution is also provided. The final report includes an implementation overview, UML diagrams, demonstration of test execution, and a conclusion based on the results of the practice.

СОДЕРЖАНИЕ

	Введение	5
1.	Первый раздел	6
1.1.	Чеклист	6
2.	Второй раздел	11
2.1.	Описание классов и методы	11
2.2.	UML-диаграмма	19
3.	Третий раздел	20
3.1.	Тестирование одного теста	20
3.2.	Успешная обработка всех тестов	24
	Заключение	25
	Список использованных источников	28
	Приложение А. Исходный код программы	29

ВВЕДЕНИЕ

Цель: Разработка автоматизированных UI-тестов для веб-приложения с использованием современных инструментов тестирования.

Задачи:

1. Написать 10 автоматизированных тестов для выбранной системы.
2. Использовать технологии: Java, Selenide (Selenium), JUnit 5, Maven, логирование.
3. Организовать работу в GitHub (репозиторий на группу из 3 человек с распределением задач в README.md).
4. Оформить чеклист в виде таблицы с описанием тестов.

Для тестирования был выбран сайт Яндекс Маркет [2]. Выбор обусловлен тем, что это популярная и функционально насыщенная торговая площадка с широким ассортиментом товаров и развитой системой поиска, фильтрации и навигации. Такие особенности позволяют протестировать разнообразные сценарии взаимодействия пользователя с веб-интерфейсом, включая авторизацию, поиск, фильтрацию, добавление товаров в корзину и сравнение.

1. ПЕРВЫЙ РАЗДЕЛ

1.1. Чеклист

Тест 1 «Проверка заголовка главной страницы»

Данный тест авторизуется на сайте Яндекс Маркета затем проверяет что заголовок сайта действительно содержит название Яндекс Маркет
Ожидаемый результат:

1. Заголовок сайта содержит слова Яндекс Маркет

Тест 2 «Проверка функциональности поиска»

Тест авторизуется на сайте яндекс маркет, затем воодит в поисковую строку текст Iphone 15 и проверяет что результаты описание выданных товаров содержит текст с название телефона.
Ожидаемый результат:

1. Заголовок вывода результатов содержит текст Iphone 15

Тест 3 «Проверка фильтра по цене»

Шаги теста:

Тест авторизуется на сайте Яндекс Маркета затем в поисковое поле вводит запрос – ноутбук, затем в поля для заполнения цен от\до вводит 10т.р и 50т.р соответственно после чего проверяет что цена первого выданного товара по данному запросу соответствует диапазону от 10 до 50 т.р.
Ожидаемый результат:

1. Цена первого товара после применения всех фильтров соответствует заданному диапазону от 10 до 50 т.р.

Тест 4 «Проверка релевантности поиска»

Тест авторизуется на сайте Яндекс Маркет затем в поисковую строку вводит текст – Iphon 15 после чего нажимает кнопку поиска,а затем проверяет что первый результат по данному запросу содержит в названии текст – Iphone

15

Ожидаемый результат:

1. Первый товар в списке содержит в названиии текст из изначального запроса в поисковой строке т.е Iphone 15.

Тест 5 «Удаление товара из корзины»

Шаги теста:

Тест авторизуется на сайте Яндекс Маркета, затем вводит запрос «ноутбук» после чего нажимает кнопку найти, затем переходит на карточку первого товара из списка и уже оттуда нажимает кнопку добавить в корзину, после чего проверяет наличие товаров в корзине и пишет их кол-во

Ожидаемый результат:

1. Наличие товара в корзине и вывод кол-ва товаров в целом

Тест 6 «Удаление товара из корзины»

Тест авторизуется на сайте Яндекс Маркета после чего переходит в корзину проверяет есть ли там товары и если их нет сообщяет что нельзя удалить товар когда их нет если же товары в корзине есть то нажимает на кннопку удалить все после чего подтверждает удаление

Ожидаемый результат:

1. Пустая корзина.

Тест 7 «Навигация по категории»

Текст авторизуется на сайте Яндекс Маркета после чего кликает по «гамбургерному» меню после чего кликает по категории электроника ,а

дальше проверяет то что в URL действительно есть текст elektronika
Ожидаемый результат:

1. URL содержит все текст elekronika т.е. навигация удачна и мы перешли к разделу электроники.

Тест 8 «Проверка пустой корзины»

Тест авторизуемся на сайте Яндекс Маркета после чего переходит в корзину из главного меню и проверяет то что корзина пуста т.е. высвечивается текст – корзина пуста
Ожидаемый результат:

1. Сообщение "Корзина пуста" отображается (или аналогичный индикатор).
2. Отсутствуют карточки товаров в корзине.

Тест 9 «Проверка кнопки каталога»

Тест авторизуется на сайт Яндекс Маркета тест проверяет то что «гамбургерное» меню или меню каталога товаров отображается корректно на сайте

Ожидаемый результат:

1. Меню каталога отображается (например, заголовок "Электроника" или список товаров).

Тест 10 «Проверка элементов хедера»

Авторизация на сайте Яндекс Маркета после чего тест проверяет наличие и правильное отображения информации в заголовке сайта.

Ожидаемый результат:

1. Кнопка зарегистрированного пользователя отображается .
2. Кнопка корзины отображается (например, иконка корзины).
3. Поисковое поле доступно для ввода запроса.

2. ВТОРОЙ РАЗДЕЛ

1.1 Описание классов и методов

1. Методы класса ***YandexMarketTests***:

- testHomePageTitle()

1) Проверяет заголовок главной страницы
2) Использует authService.authenticate() для аутентификации
3) Проверяет, что заголовок содержит "Яндекс Маркет"

- testSearchFunctionality()

1) Тестирует функциональность поиска
2) Вводит текст поиска (Constants.SEARCH_PHONE)
3) Нажимает кнопку поиска
4) Проверяет отображение заголовка результатов

- testPriceFilter()

1) Тестирует фильтр по цене
2) Ищет ноутбуки (Constants.SEARCH_LAPTOP)
3) Применяет фильтр по цене (мин/макс из Constants)
4) Проверяет, что цены находятся в заданном диапазоне

- testSearchRelevance()

1) Проверяет релевантность поиска
2) Ищет телефон (Constants.SEARCH_PHONE)
3) Проверяет, что заголовки товаров содержат искомый текст

- testAddToCart()

1) Тестирует добавление товара в корзину (с обработкой нескольких вкладок)
2) Использует логирование через LoggerUtil
3) Ищет ноутбук, выбирает первый результат
4) Переключается на новую вкладку с товаром
5) Добавляет товар в корзину и проверяет его наличие

- testRemoveFromCart()

1) Тестирует удаление товара из корзины
2) Проверяет, что товар есть в корзине перед удалением
3) Удаляет товар и проверяет, что корзина пуста

- testCategoryNavigation()

1) Проверяет навигацию по категориям
2) Открывает категорию "Электроника"
3) Проверяет URL на соответствие категории

- testEmptyCart()

1) Проверяет состояние пустой корзины
2) Убеждается, что корзина пуста

- testCatalogButton()

1) Проверяет работу кнопки каталога
2) Открывает категорию "Электроника"
3) Проверяет отображение меню каталога

- testHeaderElements()

1) Проверяет элементы хедера
2) Проверяет отображение кнопки входа, корзины и поля поиска

2. Методы класса ***AuthService***:

- authenticate()

1) Логирует начало процесса (LoggerUtil.logInfo).
2) Открывает главную страницу Яндекс.Маркета (driver.get("https://market.yandex.ru")).
3) Переходит на страницу входа через кнопку в хедере (homePage.header().clickLoginButton()).
4) Вводит логин/пароль из Constants и выполняет вход (loginPage.login(Constants.USERNAME, Constants.PASSWORD)).
5) Возвращает HomePage после успешной авторизации.

3. Методы класса ***SearchResultsPage:***:

- isResultsHeaderDisplayed()

1) Ожидает появления заголовка результатов
2) Возвращает true, если заголовок отображается

- applyPriceFilter(int min, int max)

1) Вводит минимальную и максимальную цену из Constants
2) Ожидает обновления результатов после фильтрации

- arePricesInRange(int minPrice, int maxPrice)

1) Проверяет, что все цены товаров попадают в заданный диапазон
2) Преобразует текст цен в числа (удаляет нецифровые символы)

- selectFirstProduct()

1) Кликает на первый товар в списке
2) Ожидает кликабельности элемента

- doProductTitlesContain(String keyword)

1) Проверяет, что хотя бы один товар содержит искомое слово
2) Сравнивает без учета регистра

4. Методы класса ***ProductPage***:

- addToCart()

1) Нажимает кнопку добавления товара
2) Не возвращает значений (void)

- goToCart()

1) Ожидает кликабельности ссылки
2) Выполняет переход на страницу корзины
3) Возвращает экземпляр CartPage

- waitForCartIconUpdate(String expectedText)

1) Ожидает появления указанного текста в иконке
2) Используется для проверки успешного добавления товара

5. Методы класса ***LoginPage***:

   login(String username, String password)

   Основной метод для выполнения полного процесса входа.

   Последовательно вызывает:

   - clickMoreOptionsButton()

   1) Кликает по кнопке "Больше вариантов" после ожидания ее доступности.
   2) Логирует действие через LoggerUtil.logInfo.

   - selectLoginByUsername()

   1) Выбирает опцию входа по имени пользователя.
   2) Пытается найти элемент сначала по одному XPath, затем по альтернативному (если первый не сработал).
   3) В случае ошибки логирует проблему через LoggerUtil.logError и выбрасывает исключение.

   - enterUsername(String username)

   1) Вводит переданное имя пользователя в соответствующее поле после ожидания его видимости.
   2) Логирует действие.

   - clickSubmitLoginButton()

   1) Кликает по кнопке подтверждения логина после ожидания ее доступности.
   2) Логирует действие.

   - enterPassword(String password)

   1) Вводит переданный пароль в соответствующее поле после ожидания его видимости.
   2) Логирует действие.

   - clickSubmitPasswordButton()

   1) Кликает по кнопке подтверждения пароля после ожидания ее доступности.

2) Логирует действие и возвращает новый экземпляр HomePage.

6. Методы класса ***HomePage***:

- header()

Возвращает компонент хедера (HeaderComponent), чтобы взаимодействовать с его элементами (например, корзиной, профилем и т. д.).

- isTitleCorrect()

Проверяет, содержит ли заголовок страницы текст "Яндекс Маркет".

Возвращает true, если заголовок корректен, иначе false.

- enterSearchText(String text)

Очищает поле поиска и вводит переданный текст.

clickSearchButton()

Нажимает кнопку поиска (после ввода запроса).

- clickCatalogButton()

Нажимает кнопку каталога (в текущей реализации кликает на catalogMenu, что может быть ошибкой).

- isCatalogMenuDisplayed()

Проверяет, отображается ли меню каталога.

Возвращает true, если меню видно, иначе false.

- openElectronicsCategory()

Открывает каталог (кликает catalogButton), но в текущей реализации не выбирает конкретную категорию (закомментировано).

7. Методы класса ***CartPage***:

- isItemInCart()

1) Проверяет, есть ли товары в корзине.
2) Получает текст из cartItemCount и преобразует его в число.
3) Возвращает true, если количество товаров > 0, иначе false.

4) Логирует текущее количество товаров через LoggerUtil.logInfo.
5) В случае ошибки логирует проблему через LoggerUtil.logError и возвращает false.

- removeItem()

1) Удаляет товар из корзины:
2) Кликает кнопку удаления (removeButton).
3) Подтверждает удаление (submitRemove).
4) Ожидает появления сообщения о пустой корзине (emptyCartMessage).
5) В случае ошибки выбрасывает исключение RuntimeException.

- isEmpty()

1) Проверяет, пуста ли корзина:
2) Если отображается emptyCartMessage, возвращает true.
3) Если сообщение не найдено, проверяет текст cartItemCount.
4) Возвращает true, если количество товаров = 0 или текст пуст, иначе false.

- getCartItemsCount()

1) Возвращает количество товаров в корзине:
2) Если товары есть (isItemInCart()), парсит число из cartItemCount.
3) Если корзина пуста, возвращает 0.

8. Методы класса ***HeaderComponent***:

- clickLoginButton()

Кликает по кнопке входа (loginButton) и возвращает экземпляр LoginPage.

- clickCartButton()

Кликает по основной кнопке корзины (cartButton) и возвращает экземпляр CartPage.

- clickCartFromLobbyButton()

Кликает по альтернативной кнопке корзины (cartFromLobbyButton) и возвращает экземпляр CartPage.

- isLoginButtonDisplayed()

Проверяет, отображается ли кнопка/иконка профиля (AccButton).

Возвращает true, если элемент виден.

- isCartButtonDisplayed()

Проверяет, отображается ли альтернативная кнопка корзины (cartFromLobbyButton).

Возвращает true, если элемент виден.

- isSearchInputDisplayed()

Проверяет, отображается ли поле поиска (searchInput).

Возвращает true, если элемент виден.

9. Методы класса ***SearchFilterComponent***:

- setMinPrice(int price)

Устанавливает минимальную цену:

Очищает поле minPriceInput.

Вводит переданное значение цены.

- setMaxPrice(int price)

Устанавливает максимальную цену:

Очищает поле maxPriceInput.

Вводит переданное значение цены.

- applyFilters()

Кликает по кнопке применения фильтров (applyButton).

10. Методы класса ***CatalogComponent***:

- openCatalog() - кликает по кнопке каталога, открывая его

- selectElectronicsCategory() - выбирает категорию "Электроника" в меню каталога
- isCatalogMenuDisplayed() - проверяет, отображается ли меню каталога на странице (возвращает boolean)

11. Методы класса ***NotificationComponent***:

- closeNotificationIfPresent()

1) Ожидает появления всплывающего уведомления
2) Если уведомление появляется, нажимает кнопку "Закрыть"
3) Если уведомление не появляется (выбрасывается исключение), метод завершается без действий

- isNotificationDisplayed()

1) Проверяет, отображается ли уведомление на странице
2) Возвращает true, если уведомление видимо
3) Возвращает false, если уведомление не найдено или не видимо (перехватывает исключение)

12. Методы класса ***SearchComponent***:

- searchFor(String text)

1) Очищает поле поиска (searchInput.clear()).
2) Вводит переданный текст (searchInput.sendKeys(text)).
3) Нажимает кнопку поиска (searchButton.click()).
4) Назначение:
5) Выполняет поиск товаров по заданному тексту.

- isSearchInputDisplayed()

1) Проверяет, отображается ли поле поиска на странице (searchInput.isDisplayed()).
2) Возвращает true, если поле видимо, и false в противном случае.

Код программы [6]

2.2. UML-диаграмма

UML-диаграмма классов системы тестирования Яндекс.Маркета(рис.1)

Она отображает ключевые классы, используемые в автоматизированном тестировании:

- Базовые классы (*BaseTest*, *BasePage*, *BaseComponent*) с общими методами (настройка драйвера, логирование).
- Страницы приложения (*HomePage*, *LoginPage*, *CartPage*, *ProductPage*), включая их компоненты (*HeaderComponent*, *SearchComponent*).
- Тестовые классы (*YandexMarketTests*) и вспомогательные утилиты (*LoggerUtil*, *Constants*).
- Стрелки и связи между классами отражают зависимости и взаимодействия.

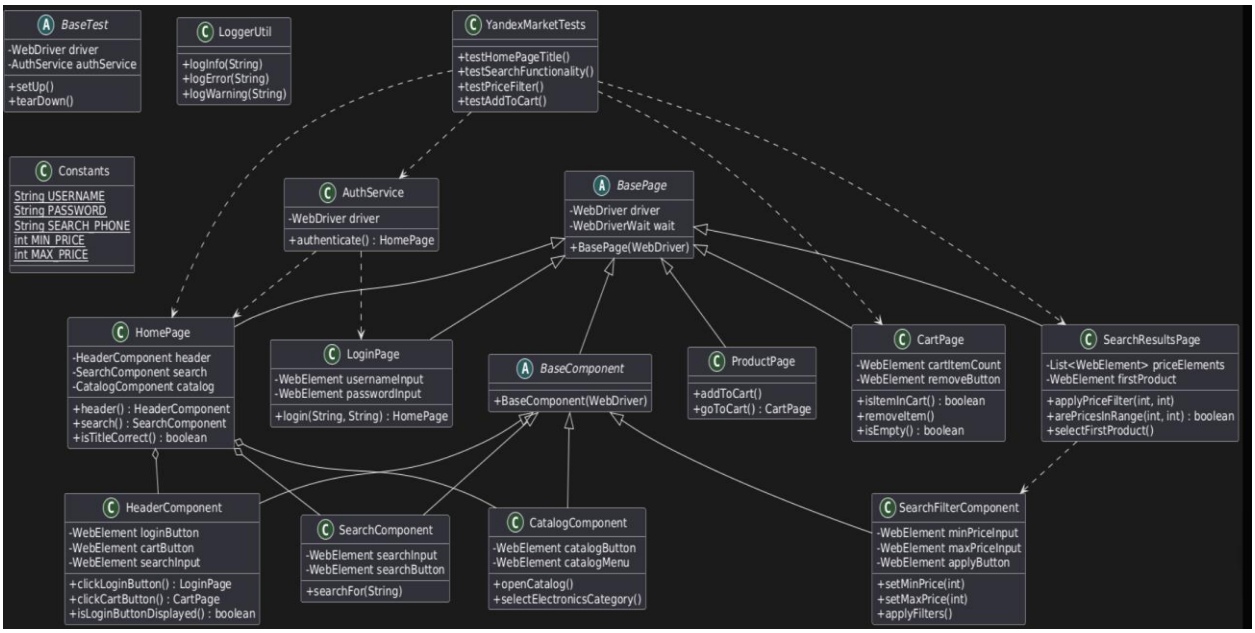

Рисунок 1 – UML-диаграмма классов тестов

3. ТРЕТИЙ РАЗДЕЛ

1.1. Тестирование одного теста

Для описания выбран тест «Добавление товара в корзину»

1. Программа заходит на сайт Яндекс маркет и проходит регистрацию на сайте. Выбирает войти по логину , вводит почту и пароль(рис.2-рис.4).

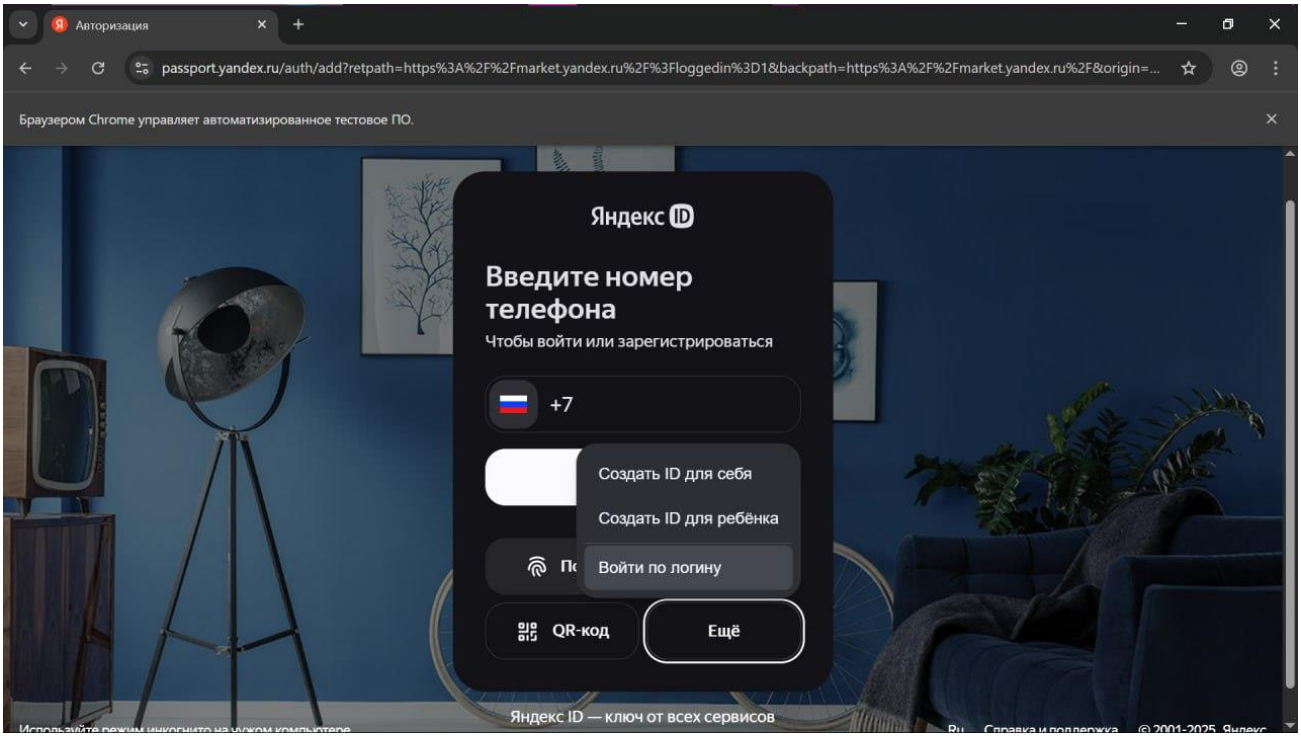

Рисунок 2 – Авторизация

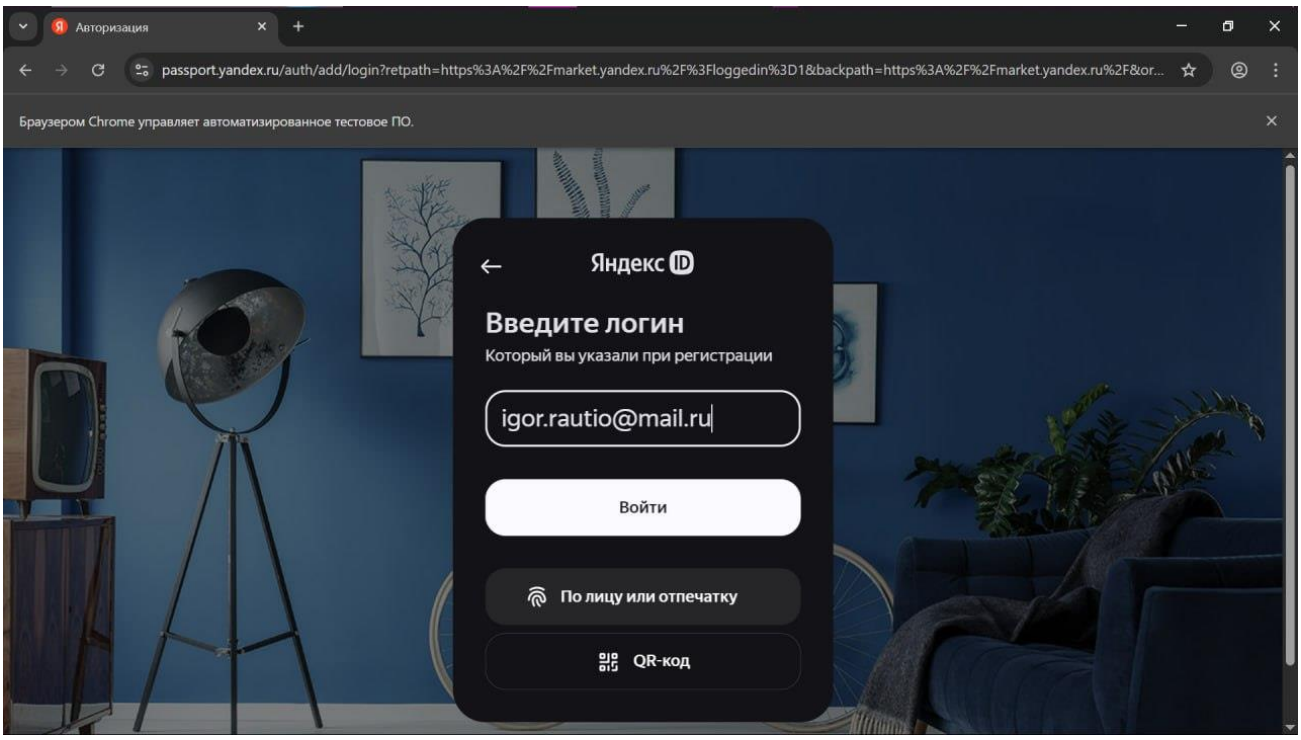

Рисунок 3 – Ввод логина

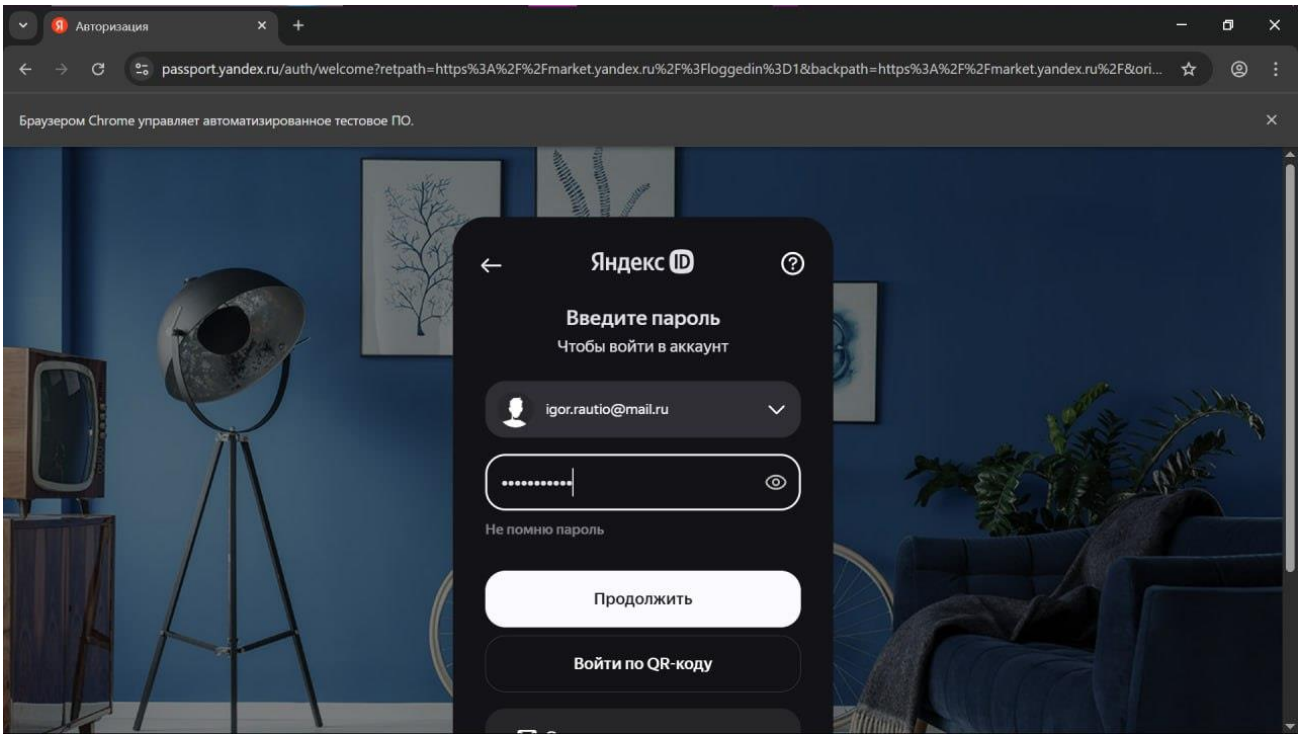

Рисунок 4 – Ввод пароля

2. Далее в поисковой строке вводится запрос «ноутбук», клик по кнопке «Найти», выбор товара по запросу(рис.5)

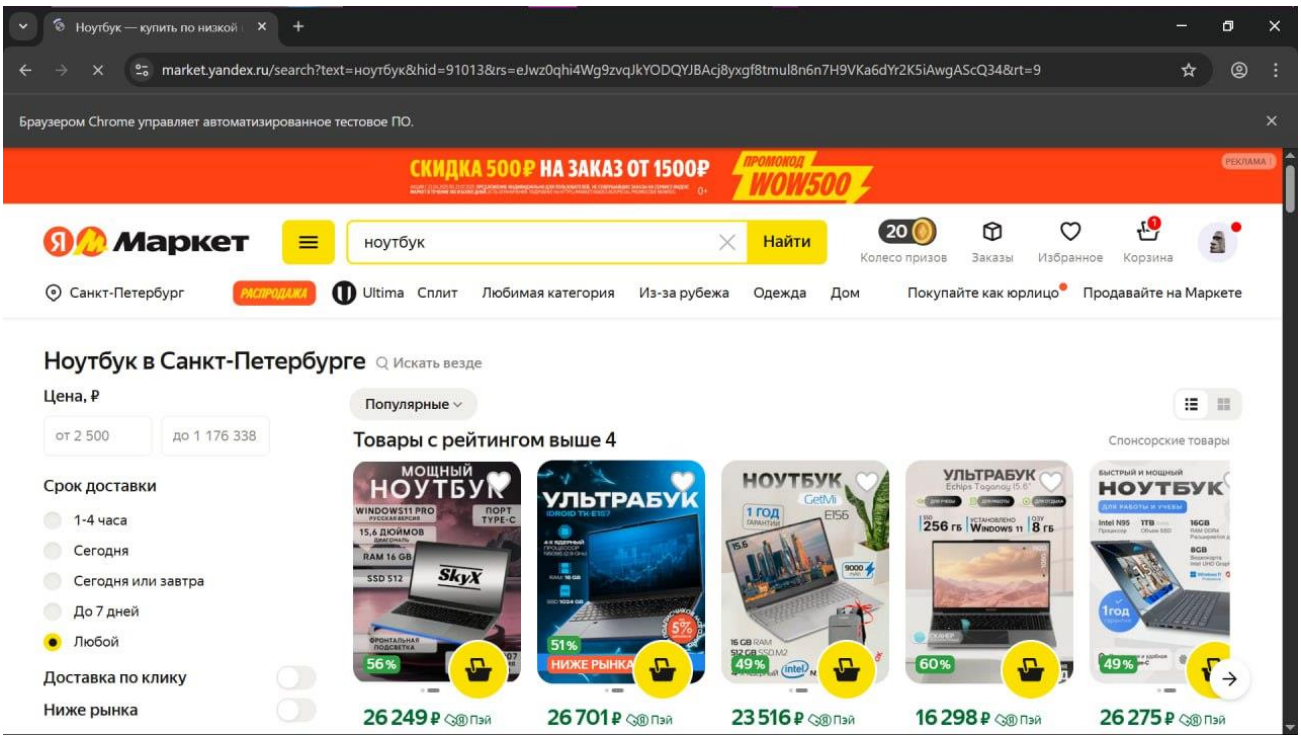

Рисунок 5 – Ввод товара в строку поиска

3. Программа переходит к карточке выбранного товара и добавляет его в корзину(рис.6)

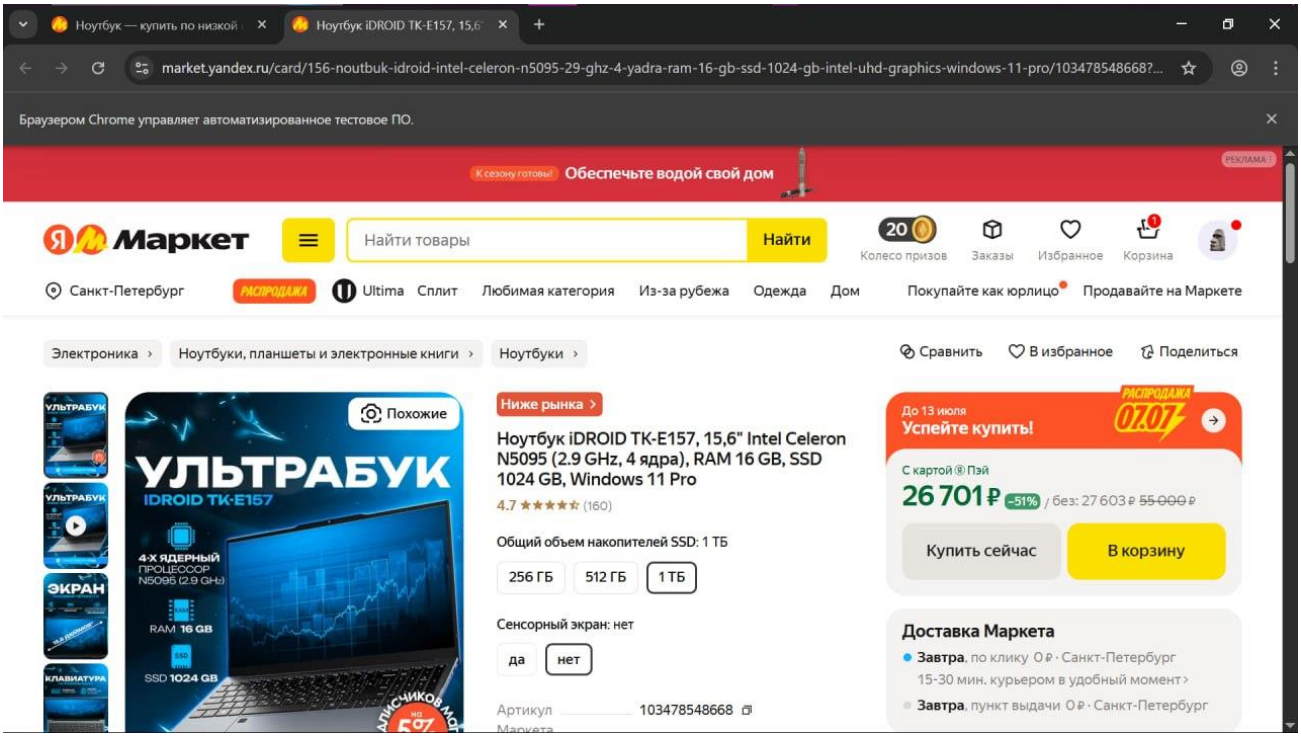

Рисунок 6 – Добавление товара в корзину

4. Происходит проверка, что товар добавлен в корзину(рис.7)

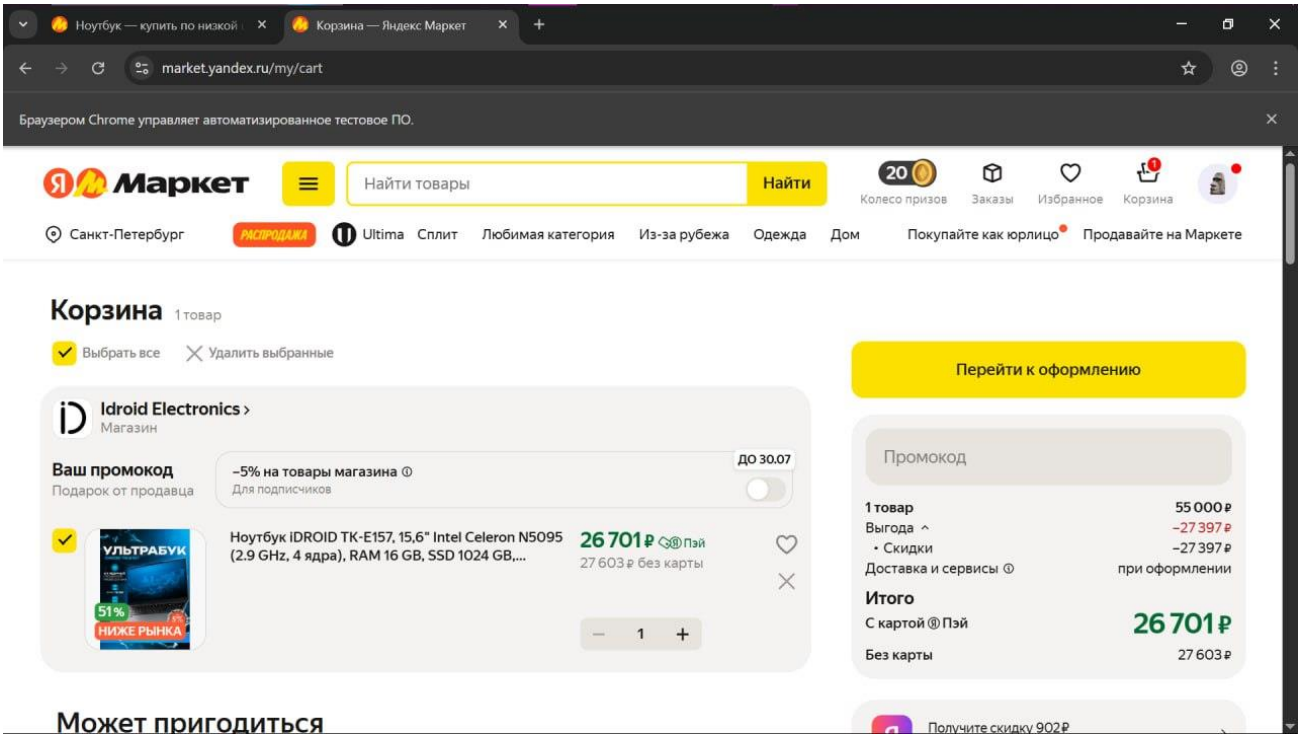

Рисунок 7 – Переход в корзину

5. На рисунке 8 представлены поэтапные шаги.

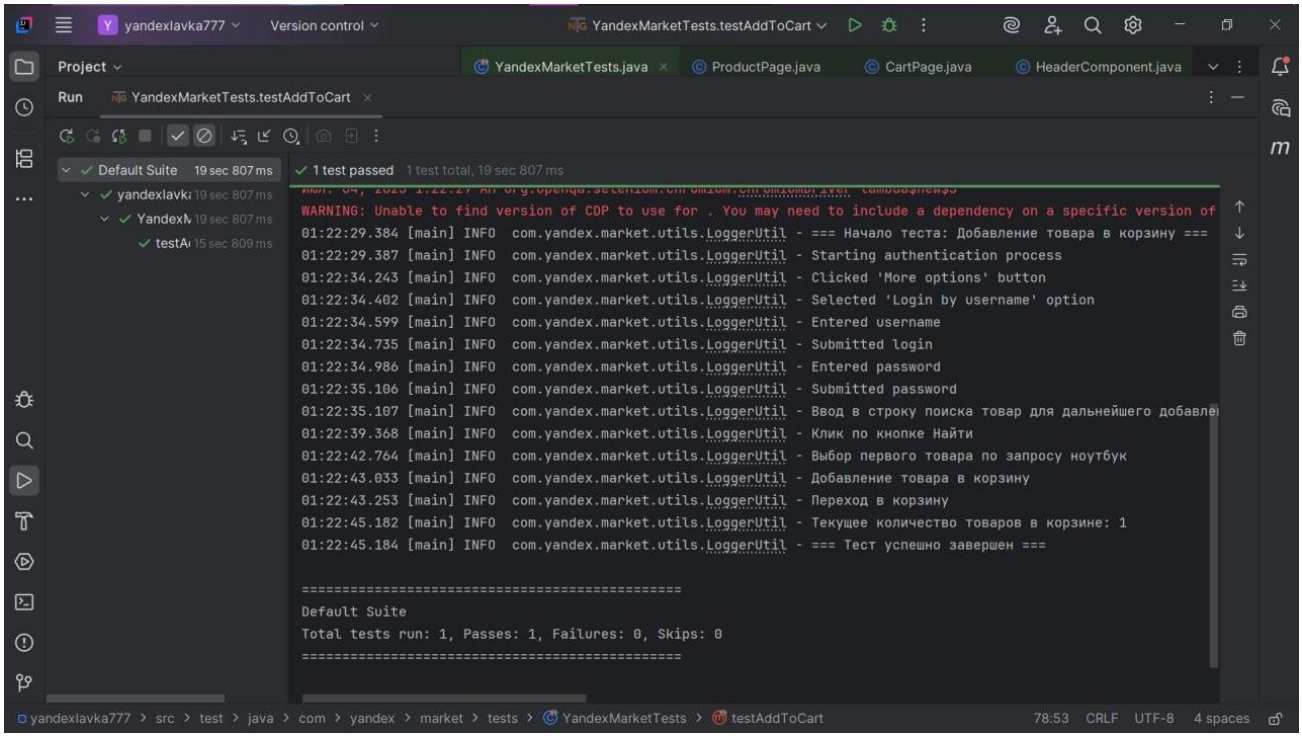

Рисунок 8 – Логгер для тестов

3.2. Успешная обработка всех тестов

Скриншот (рис. 9) демонстрирует успешное прохождение всех тестов: все проверки завершены без ошибок, о чём свидетельствуют зелёные индикаторы и итоговый статус «Passes: 10, Failures: 0».

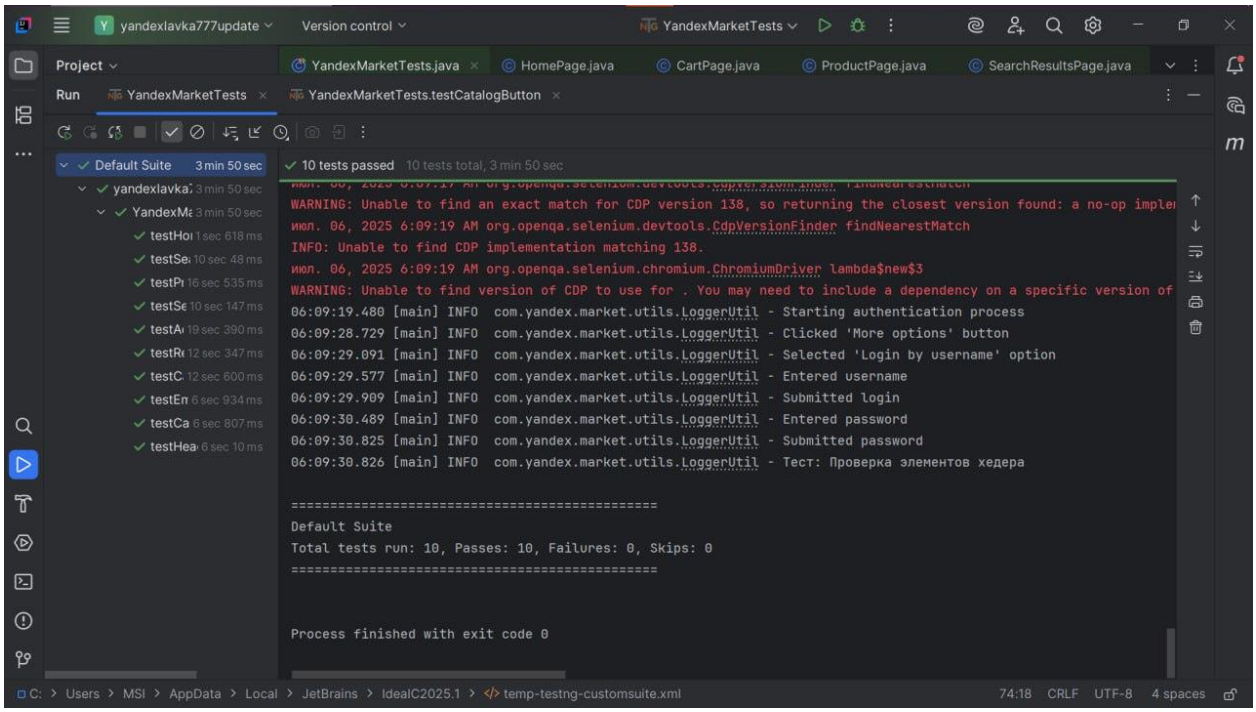

Рисунок 9 – Успешное выполнение всех тестов

ЗАКЛЮЧЕНИЕ

В ходе выполнения практики по UI-тестированию была проведена работа по разработке и реализации автоматизированных тестов для веб-приложения Яндекс.Маркет с использованием современных технологий, таких как Java, Selenide (на основе Selenium), JUnit 5, Maven и инструменты логирования. Было создано 10 автотестов, охватывающих ключевые функциональные блоки системы, включая авторизацию, поиск товаров, фильтрацию по цене, добавление и удаление товаров из корзины, проверку элементов интерфейса и навигацию по каталогу.

Цель практики — освоить методики автоматизации тестирования веб-приложений и применить их на реальном проекте — была успешно достигнута. В процессе работы были закреплены навыки написания чистого, поддерживаемого кода, структурирования проекта, организации совместной работы через GitHub, а также документирование тестовых сценариев в формате чеклиста. Все этапы практики были выполнены в соответствии с заданными требованиями и сроками.

Одним из ключевых результатов стало создание модульной и гибкой архитектуры тестовой системы. Проект был организован с использованием принципов объектно-ориентированного программирования: были выделены абстрактные базовые классы `*BasePage*` и `*BaseComponent*`, от которых наследуются конкретные страницы и компоненты. Это позволило минимизировать дублирование кода, повысить его читаемость и упростить сопровождение. Также были реализованы специализированные сервисы, такие как `*AuthService*` для управления аутентификацией, `*LoggerUtil*` для централизованного логирования, `Constants` для хранения глобальных переменных и параметров. Такой подход способствовал созданию устойчивой и легко расширяемой тестовой среды.

Большое внимание было уделено качеству документирования. Чеклист с описанием всех тестов, шагов, входных данных и ожидаемых результатов обеспечил прозрачность и воспроизводимость тестирования. Каждый тестовый сценарий был тщательно продуман и протестирован на соответствие бизнес-логике приложения. Например, тесты «Поиск товара» и «Релевантность поиска» позволяют убедиться в том, что система правильно обрабатывает пользовательские запросы и возвращает релевантные результаты. Тесты «Добавление товара в корзину» и «Удаление товара из корзины» проверяют корректную работу одного из самых важных пользовательских сценариев — покупки товара.

Также была реализована система логирования, которая позволяет фиксировать все действия во время выполнения тестов. Это помогает быстро находить и анализировать ошибки, а также отслеживать прогресс выполнения тестового набора. Логирование выполнялось с помощью утилитарного класса `*LoggerUtil*`, который предоставляет удобный интерфейс для вывода информации, предупреждений и сообщений об ошибках.

Помимо технической реализации, значительное внимание было уделено организации командной работы. Работа велась в группе из трех человек через систему контроля версий GitHub, где каждый участник нес ответственность за определенную часть проекта. Это способствовало развитию навыков совместной разработки, эффективного взаимодействия и использования Git-стратегий.

Все тесты были успешно запущены и показали высокую степень покрытия проверяемой функциональности. На этапе демонстрации работы тестов были представлены скриншоты, подтверждающие успешное выполнение каждого сценария. Было проверено состояние корзины,

работоспособность поиска, корректность фильтрации и отображение основных элементов интерфейса.

Таким образом, результаты практики показали, что разработанные автотесты полностью соответствуют поставленным задачам. Они обеспечивают надежное покрытие ключевых функций Яндекс.Маркета, могут быть легко адаптированы к изменениям в интерфейсе и расширены для проверки новых возможностей системы. Полученный опыт и навыки позволили не только углубить знания в области автоматизации тестирования, но и получить практические навыки работы с современными инструментами разработки, которые будут полезны в дальнейшей профессиональной деятельности.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Васильев А.Н. «Программирование на Java»
2. Яндекс Маркет. – URL: https://market.yandex.ru
3. Официальная документация Selenium по тестированию пользовательского интерфейса. – URL: https://www.selenium.dev/documentation/webdriver/ui_testing/
4. Мэйвороно, Энтони Test-Driven Development with Selenium — Packt Publishing, 2021. (Подробное описание работы с UI-тестами через Selenium и Selenide
5. Официальные материалы по функционалу Яндекс Маркета, полезны для понимания логики интерфейса. – URL: https://yandex.ru/support/partners/market/index.html
6. Исходный код программы. – URL: https://github.com/zkarajaz/Summer_internship/tree/main/yandexlavka777update/yandexlavka777