Национальная научно-образовательная корпорация ИТМО

Факультет программной инженерии и компьютерной техники

ЛАБОРАТОРНАЯ РАБОТА №3

по дисциплине

«Тестирование программного обеспечения»

Вариант megamarket.ru

Выполнил:

Студент группы P3309

Черноморов Кирилл Александрович

Преподаватель:

Наумова Надежда Александровна

Санкт-Петербург, 2025

## Задание:

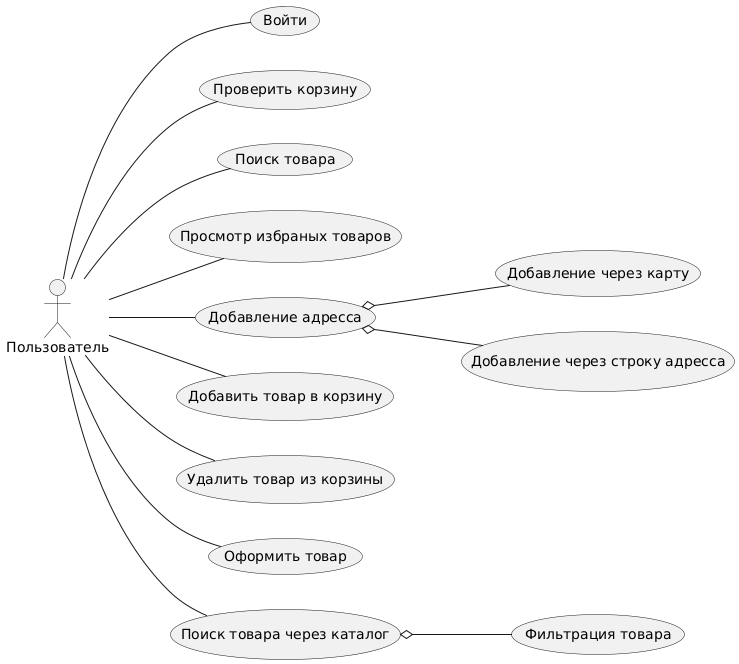
**Требования к выполнению работы:**

1. Тестовое покрытие должно быть сформировано на основании набора прецедентов использования сайта.
2. Тестирование должно осуществляться автоматически - с помощью системы автоматизированного тестирования [Selenium](http://docs.seleniumhq.org/" \t "_blank).
3. Шаблоны тестов должны формироваться при помощи Selenium IDE и исполняться при помощи Selenium RC в браузерах Firefox и Chrome.
4. Предполагается, что тестируемый сайт использует динамическую генерацию элементов на странице, т.е. выбор элемента в DOM должен осуществляться не на основании его ID, а с помощью [XPath](http://ru.wikipedia.org/wiki/XPath" \t "_blank).

**Требования к содержанию отчёта:**

1. Текст задания.
2. UseCase-диаграмму с прецедентами использования тестируемого сайта.
3. CheckList тестового покрытия.
4. Описание набора тестовых сценариев.
5. Результаты тестирования.
6. Выводы.

Диаграмма прецедентов



**Прецеденты**

**Тестирование авторизации**

Главный актер: Пользователь

1. Пользователь открывает сайт megamarket

2. Нажимает на кнопку войти

3. Удостоверяется в наличии предложенных способах авторизации

**Поиск товара**

Главный актер: Пользователь

1. Пользователь открывает сайт megamarket

2. Нажимает на кнопку поиска

3. Вводит realme gt 6 и нажимает enter

4. Удостоверяется в наличии телефонов

**Просмотр избранных товаров**

Главный актер: Пользователь

1. Пользователь открывает сайт megamarket

2. Пользователь нажимает на сердечко

**Просмотр корзины**

Главный актер: Пользователь

1. Пользователь открывает сайт megamarket

2. Пользователь нажимает на корзину

**Добавления адреса через карту**

Главный актер: Пользователь

1. Пользователь открывает сайт megamarket

2. Нажимает на кнопку адреса

3. Нажимает на карту

4. Нажимает кнопку добавить

**Добавить адрес через поиск**

Главный актер: Пользователь

1. Пользователь открывает сайт megamarket

2. Нажимает на кнопку адреса

3. Вводит адрес

4. Нажимает добавить

**Добавить продукт в корзину из главной страницы**

Главный актер: Пользователь

1. Пользователь открывает сайт megamarket

2. Скролит вниз

3. Нажимает на добавления товара

4. В случаи если товар от определенного магазина выбирает магазин

5. Проверяет что продукт добавился в корзину

**Удаление товара из корзины**

Главный актер: Пользователь

1. Пользователь открывает сайт megamarket

2. Добавляет товар по предыдущему прецеденту

3. Заходит на megamarket.ru/multicart

4. Удаляет товар

5. Проверяет что товаров нет

**Оформляет товар из корзины**

Главный актер: Пользователь

1. Пользователь открывает сайт megamarket

2. Добавляет товар по предыдущему прецеденту

3. Заходит на megamarket.ru/multicart

4. Нажимает на кнопку оформить товар

5. Проверяет предложение об авторизации

**Поиск товара в каталоге**

Главный актер: Пользователь

1. Пользователь открывает сайт megamarket/catalog

2. Пользователь проверяет наличие категорий

3. Пользователь выбирает товары для дачи

4. Пользователь выбирает цветы

5. Пользователь добавляет цветы в корзину

6. Пользователь проверяет наличие товара в корзине

**Проверка фильтров**

Главный актер: Пользователь

1. Пользователь открывает сайт megamarket/catalog

2. Пользователь проверяет наличие категорий

3. Пользователь выбирает товары для дачи

4. Пользователь выбирает цветы

5. Ставит высокий рейтинг

6. Скролит страницу вниз

7. Выбирает продавца и проверяет что все товары от него

8. Ставит минимальную цену и проверяет что все товары не дешевле

## Выполнение

### <https://github.com/Gallade901/tpo-3>

### Вывод по работе

Я разработал тестовые сценарии на основе UseCase-диаграммы для сайта megaMarket, охватив ключевые прецеденты: поиск билетов, оформление бронирования и выбор отелей. С помощью Selenium WebDriver создал шаблоны тестов, организовал их автоматическое исполнение в браузерах Firefox и Chrome. Для работы с динамически генерируемыми элементами использовал XPath, что позволило корректно взаимодействовать с интерфейсом. В процессе познакомился с инструментами автоматизации, кросс-браузерным тестированием и методами адаптации скриптов под изменяемую структуру DOM. Это позволило углубить навыки в тестировании веб-приложений с динамическим контентом.