## ****Знакомство:****

Привет, меня зовут Елена.

Я джуниор QA-инженер с горящими глазами. Ищу удаленную работу!

**Обо мне:**

Я начинающий QA инженер, ручной тестировщик с опытом тестирования веб-приложений. Прошла обучение по курсу "Инженер по тестированию" в Skypro, а также завершила обучение по автоматизации тестирования на Python.

### ****Связаться со мной:****

<https://mozhaysk.hh.ru/resume/2e831257ff0ee080fe0039ed1f7079335a4f30>

Мобильный телефон — предпочитаемый способ связи

[+7 (916) 226-80-93](https://tel:+79162268093)

### Языки и инструменты:

Postman, SQL, Python, test it.

Владею:

Составление тестовой документации: Test-case, Check-list и Cheat-list, Bug report.

Системы хранения документации: Jira, Confluence, Trello, Qase.io, Sitechko, Checkvist, Miro, Allure.

БД: pgAdmin, SQL.

Api: HTTP/HTTPS, Chrome DevTools, Postman, Swagger, curl.

### Пример проектов в портфолио:

<https://lenokzvereva.atlassian.net/wiki/x/DQBP>

[06 12\_17\_40672b5e64efac89.98064364skyengрасписаниеКурсоваяработа.postman\_collection.json](/api/attachments.redirect?id=2d72ef09-109a-4056-a8ca-fa518b638f65)[06 12\_17\_40672b5e64efaf78.51388895ТесткейсыКР2.xlsx](/api/attachments.redirect?id=9323e012-eb56-4e9a-b4b7-16d20ab6f15b)[06 12\_17\_40672b5e64efc175.50711394skyengрасписаниеКурсоваяработа.postman\_test\_run.json](/api/attachments.redirect?id=52f82a33-8378-458c-972d-debeeeca7ead)

В портфолио собраны проекты, выполненные во время обучения по специальности [Инженер по тестированию](https://praktikum.yandex.ru/qa-engineer) в SKYPRO.

## Тестирование веб-приложений

### Задание

1. Анализ требований

Изучи обновлённые требования к сервису Кинопоиск. Команда успела сделать первую версию продукта.

2. Выбор конфигураций для кроссбраузерного тестирования

Выбери операционные системы, браузеры и разрешения, в которых нужно провести тесты. В этой задаче примени парное тестирование, чтобы уменьшить количество комбинаций. Поддерживаемые ОС, браузеры и разрешения описаны в требованиях.

3. Тестовая документация для вёрстки

Проанализируй требования к вёрстке и определи объекты тестирования: это все визуальные элементы на макете, у которых нужно проверить внешний вид и корректное расположение.

Напиши чек-лист, чтобы протестировать вёрстку. Не забудь проверить: соответствие макетам (визуально — не «пиксель в пиксель»); как отображается результат расчёта стоимости и времени маршрута при длинных значениях; орфографию.

Напиши чек-лист в таблице «Чек-лист и результаты выполнения тестов: тестирование вёрстки и логики интерфейса».

4. Тестовая документация для логики интерфейса

Проанализируй требования к логике интерфейса и составь тестовую документацию.

Напиши тест-кейсы на отображение результата расчёта стоимости и времени по всем видам транспорта. Чтобы составить тесты на отображение результатов, применяй технику «Таблица принятия решений». В тестировании используй только адреса из таблицы в уроке «Требования».

Важно: разработчики ещё не до конца реализовали расчёт стоимости поездки на бэкенде, поэтому логика расчёта суммы и времени может быть некорректной. Её тестировать не нужно.

Составь чек-листы, которые помогут протестировать: отрисовку и скрытие точек на карте; отрисовку и скрытие маршрута на карте; масштабирование карты; перемещение карты. В требованиях есть ограниченный список адресов — тестировать можно только по ним.

Дополни чек-лист из задания 3. Оформи тест-кейсы в таблице «Тест-кейсы: логика интерфейса».

5. Тестирование и заведение баг-репортов

Протестируй сервис Кинопоиска

У тебя уже есть набор конфигураций, в которых нужно проверить сервис. Но на тестирование осталось не так много времени, поэтому протестируй сервис на ограниченном наборе: в твоей текущей операционной системе в Яндекс.Браузере при разрешении экрана 800x600, а в Firefox — при 1280х720.

Чтобы проверить длинные строки в интерфейсе, примени DevTools — в нём можно заменить HTML; или Fiddler/Charles — в них можно изменить ответ от сервера.

В процессе тестирования отмечай результаты выполнения теста: PASSED или FAILED. Если тест со статусом FAILED, заведи баг-репорт в Jira; в очереди BUG и вписывай ID в соответствующую таблицу результатов.

Тестирование вёрстки нужно провести в обоих окружениях. Тесты на логику интерфейса можно сделать только в одном разрешении. В отчёте поставь статус N/A (Not Applicable, не применимо) около такого теста.

## Проекты

## Финальный проект по Web тестированию

## Курсовая работа по тестированию API

<https://lenokzvereva.atlassian.net/wiki/spaces/~712020adec5a97ad93410a9e675c15a28eab30/pages/5177357/1-2>

## Финальный проект по ручному тестированию

<https://blackid.yonote.ru/doc/sajt-kinopoisk-test-plan-deUhKOUDUg>

<https://github.com/FedorovaEP-1985/Finale_progect>

## ****Описание проектов:****

**Проект: Курсовая работа по тестированию API**

**Проект: Расписание личного кабинета для учителей Skyeng.**

Описание: в расписание учителей добавлена новая функциональность — личные события. Преподаватель может использовать личные события для собственных встреч, которые не привязаны к ученикам платформы Skyeng. Необходимо протестировать бэкенд на основе функционального чек-листа по личным событиям. Для тестирования была составлена коллекция запросов в Postman. По завершении тестирования был доработан отчет к первому проекту.

Цель проекта: протестировать новую функциональность и вынести решение о готовности продукта к релизу.

Были проведены такие виды тестирования, как:

• Тестирование документации,

• Smoke-тестирование,

• Функциональное тестирование,

• Регрессионное тестирование,

• Приёмочное тестирование.

Были применены такие инструменты, как:

• Miro - для создания декомпозиции новой функциональности;

• Swagger - для чтения документации API на проекте;

• DevTools - для воспроизведения запросов в рабочей среде;

• Qase, Sitechko (Checkvist) - для написания тестовой документации;

• Confluence - для ведения тест-плана и отчета;

• Jira - для фиксирования баг-репортов.

**Проект: Финальный проект по Web тестированию**

**Проект: WEB-приложения**

Описание: тестировщик-студент подготовил несколько веб-страниц на следующие темы: «Работа с командной строкой Windows», «Работа с терминалом MacOS», «Команды Git». Необходимо найти ошибки и исправить их.

Цель проекта: найти ошибки на страницах: опечатки, отсутствие ссылок на главную страницу, несоответствие переходов и названия кнопок, ошибки в описании команд и другие. Записать ошибки в файл defects.txt и исправить найденные ошибки.

Были применены такие инструменты, как:

• GitHub — сайт для размещения IT-проектов, основанный на системе контроля версий

**Проект: Финальный проект по ручному тестированию**

**Проект: Кинопоиск** - сайт для выбора фильмов, новинок киноиндустрии, а также другого контента.

Описание: www.kinopoisk.ru – сайт, позволяющий пользователю узнать больше о новинках в киноиндустрии, просматривать их, в соответствии выбранного жанра, узнавать новости о новинках, просмотр онлайн кинотеатра, возможность подключить  других операторов кинопросмотра, купить онлайн билет в кино еще пользоваться обратной связью. Оформлять подписки.

Цель проекта: провести тестирование выбранного перечня функций приложения и составить отчёт о проделанной работе с результатами, выводом, рекомендациями и заключением

Были проведены такие виды тестирования, как:

• Функциональное тестирование;

• Тестирование доступности;

• Кроссбраузерное тестирование;

• Тестирование адаптивности веб-версии на мобильных устройствах;

• Тестирование API;

• UI/UX тестирование.

Были применены такие инструменты, как:

• Miro;

• Jira;

• Qase.io;

• Sitechko;

• DewTools;

• Postman;

• WAVE;

• LambdaTest;

[Наверх](https://github.com/FedorovaEP-1985/Portfolio)

© [Кинопоиск](https://www.kinopoisk.ru) & [Елена Федорова](https://github.com/FedorovaEP-1985/Portfolio), 2025