

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Факультет

вычислительной техники

Кафедра

«Вычислительная техника»

Направление подготовки: 09.03.01 «Информатика и вычислительная техника»

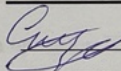
Профиль: «Программное обеспечение средств вычислительной
техники и автоматизированных систем»

БАКАЛАВРСКАЯ РАБОТА

на тему

АВТОМАТИЗИРОВАННОЕ ФУНКЦИОНАЛЬНОЕ
ТЕСТИРОВАНИЕ СОЗДАНИЯ ИНДИВИДУАЛЬНЫХ И
ГРУППОВЫХ ПРИКАЗОВ

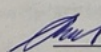
Студент

 17.05.2024 Скирдова Вероника Максимовна
(подпись, дата) (ФИО полностью)

Руководитель

 17.05.2024 А.Н. Токарев
(подпись, дата) (фамилия, инициалы)

Нормоконтролёр

 22.05.24 А.С. Бычков
(подпись, дата) (фамилия, инициалы)

Работа допущена к защите (протокол зас. кафедры от 17.05.24 № 11)

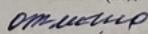
Заведующий кафедрой


(подпись)

М.А. Митрохин

(фамилия, инициалы)

Работа защищена с отметкой



(протокол заседания ГЭК от 04.06.24 № 3)

Секретарь ГЭК



Е.И. Гурин

(подпись)

(фамилия, инициалы)

Пенза, 2024

Федеральное государственное бюджетное образовательное учреждение высшего образования

«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Факультет

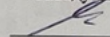
вычислительной техники

Кафедра

«Вычислительная техника»

«Утверждаю»

Заведующий кафедрой

 М.А. Митрохин

«19» 04 2024г.

ЗАДАНИЕ НА БАКАЛАВРСКУЮ РАБОТУ

1. Студент Скирдова Вероника Максимовна гр. 20ВВП2 факультета вычислительной техники.

2. Тема работы Автоматизированное функциональное тестирование создания индивидуальных и групповых приказов.

Тема утверждена приказом ПГУ № 405/0 от «19» 04. 2024 г.

3. Руководитель бакалаврской работы к.т.н., доцент кафедры ВТ ПГУ Токарев А. Н.

4. Задание на работу (назначение разработки, исходные данные и т.п.)

Разработать автоматизированное функциональное тестирование web-приложения для оптимизации процесса отладки. Язык программирования Java, среда разработки IntelliJ IDEA. Программа должна содержать широкий спектр проверок и использовать следующие технологии: Selenide, TestNG, Allure.

5. Перечень подлежащих разработке вопросов.

Требуется произвести анализ предметных областей и фреймворков автоматизированного тестирования. Выбрать технологии разработки и анализа автотестов.

6. Календарный график выполнения работы

№ п/п	Наименование этапов работы	Объем работы	Срок выполнения	Подпись руководителя
1	Анализ предметной области функционального тестирования	32	19.04.2024- 22.04.2024	
2	Анализ предметной области автоматизированного тестирования	32	23.04.2024- 27.04.2024	
3	Анализ фреймворков для автоматизации тестирования	65	28.04.2024- 02.05.2024	
4	Разработка системы функционального автоматизированного тестирования на основе проведенного анализа	94	03.05.2024- 23.05.2024	
5	Отладка системы функционального автоматизированного тестирования	49	24.05.2024- 27.05.2024	
6	Оформление выпускной работы	32	28.05.2024- 30.05.2024	

Дата выдачи задания "19" 04.2024 г.

Руководитель бакалаврской работы  19.04.2024 А.Н. Токарев
(подпись, дата)

Задание к исполнению приняла студент  19.04.2024 В.М. Скирдова
(подпись, дата)

Работу к защите допустить

Декан факультета  03.06.2024 Л.Р. Фионова
(подпись, дата)

Реферат

Бакалаврская работа содержит 66 листов, 26 рисунков, 2 таблицы, 15 источников и 3 приложения.

АВТОТЕСТЫ, АВТОМАТИЗИРОВАННОЕ ТЕСТИРОВАНИЕ, ТЕСТИРОВАНИЕ СИСТЕМЫ, JAVA, TESTNG, SELENIDE

Цель работы – изучение систем автоматизированного тестирования и разработка программной системы для реализации функциональных проверок автоматизированной информационной системы.

Объектом разработки является автоматизированное тестирование, осуществляющее функциональную проверку автоматизированной информационной системы. Данная система позволяет виртуально выполнять действия пользователя и генерирует отчет о проведенных проверках.

В результате проведенной работы разработаны автоматизированные тесты на базе фреймворков Selenide и TestNG на языке Java.

Результаты данной работы применены в научно-исследовательской работе ООО «Ай-Теко ВЦ» для организации рабочего процесса.

ПГУ 1.090301.36.001 ПЗ

Изм	Лист	№ докум.	Подпись	Дата			
Разраб.	Скирдова В.М.		24.05.24		Автоматизированное функциональное тестирование создания индивидуальных и групповых приказов Пояснительная записка	Литера	Лист
Пров.	Токарев А.Н.		24.05.24				Листов
							2
							114
Н. Контр.	Бычков А.С.		24.05.24		ФВТ, каф. ВТ гр. 20ВВП2		
Уме.	Митрохин М.А.		24.05.24				

Содержание

Перечень сокращений и обозначений	3
Введение	4
1 Анализ предметной области	5
1.1 Виды тестирования	6
1.1.1 Функциональное тестирование	7
1.1.2 Тестирование интерфейса пользователя	8
1.1.3 Тестирование удобства использования	9
1.1.4 Ручное тестирование	10
1.1.5 Регрессионное тестирование	11
1.1.6 Модульное тестирование	12
1.1.7 Smoke-тестирование	12
1.1.8 Тестирование безопасности	13
1.1.9 Интеграционное тестирование	14
1.2 Автоматизация тестирования	14
1.3 Возможности автоматизации тестирования	15
1.4 Объект тестирования	16
1.5 Постановка задачи	18
2 Выбор инструментов тестирования	25
2.1 Выбор языка программирования	25
2.1.1 Python	25
2.1.2 C#	25
2.1.3 Java	26
2.2 Выбор фреймворков для автоматизации тестирования	26
2.2.1 Selenium	26
2.2.2 Selenide	27
2.2.3 JUnit	27
2.2.4 TestNG	27
2.3 Выбор среды разработки	27

2.3.1 VSCode	28
2.3.2 NetBeans	28
2.3.3 Eclipse	29
2.3.4 IntelliJ IDEA	29
3 Проектирование системы	31
4 Развертывание системы проектирования	36
4.1 Подготовка окружения	36
4.1.1 Установка Java JDK	36
4.1.2 Установка IntelliJ IDEA	38
4.1.3 Подключение Selenide	39
4.1.4 Подключение TestNG	40
4.2 Разработка автоматизированных тестов	40
5 Результаты работы автоматизированного тестирования	44
5.1 Запуск тестирования	44
5.2 Получение результатов тестирования.....	46
5.3 Генерация отчетов Allure	49
6 Охрана труда.....	56
6.1 Потенциальные опасности при работе с ЭВМ.....	56
6.2 Эргономика рабочего места	57
Заключение	60
Список использованных источников	61
Приложение А – UML-диаграммы	63
Приложение А.1 – Диаграмма классов	64
Приложение А.2 – Диаграмма прецедентов.....	66
Приложение А.3 – Диаграмма последовательностей.....	67
Приложение А.4 – Диаграмма деятельности	68
Приложение Б – Листинг программы	69
Приложение В – Презентация.....	107

Перечень сокращений и обозначений

АИС – автоматизированная информационная система

АО – академический отпуск

БД – база данных

ПК – персональный компьютер

ПОО – профессиональная образовательная организация

ПО – программное обеспечение

ПЭВМ – персональная электронно-вычислительная машина

СПО – среднее профессиональное образование

ЭВМ – электронно-вычислительная машина

API (англ. Application Programming Interface) – передача состояния представления

HTTP (англ. Hyper Text Transfer Protocol) – протокол передачи гипертекста

IDE (англ. Integrated Development Environment) – интегрированная среда разработки

QA (англ. Quality Assurance) – обеспечение качества

UI (англ. User Interface) – пользовательский интерфейс

UML (англ. Unified Modeling Language) – унифицированный язык моделирования

UX (англ. User Experience) – пользовательский опыт

					ПГУ 1.090301.36.001 ПЗ	Лист
						3
Изм.	Лист	№ докум.	Подпись	Дата		

Введение

В ходе работы будут рассмотрены различные аспекты автоматизации тестирования, проанализированы существующие инструменты и фреймворки, а также разработана и протестирована система функционального тестирования.

В современном мире программное обеспечение занимает значительную роль в различных аспектах жизни, таких как образование, бизнес и повседневная деятельность. Одной из важнейших задач при разработке программного обеспечения является обеспечение его надежности и качества, достигающееся путем всестороннего тестирования. Особенно актуальной становится автоматизация тестирования, которая позволяет значительно ускорить процесс, улучшить точность и минимизировать человеческие ошибки.

Цель данной бакалаврской работы – разработка системы автоматизированного тестирования для автоматизированной информационной системы «Зачисление в учреждения профессионального образования». Эта система предназначена в том числе для создания индивидуальных и групповых приказов, являясь немаловажным компонентом процесса зачисления студентов.

Автоматизированное тестирование позволяет проводить комплексные проверки функциональности программного обеспечения, обеспечивая его соответствие заданным требованиям и спецификациям, включая проверку пользовательского интерфейса, производительности, безопасности и других аспектов, критичных для успешной работы системы. В ходе работы будет рассмотрен широкий спектр инструментов и фреймворков для автоматизации тестирования.

Работа также подразумевает анализ предметной области, выбор и обоснование используемых инструментов, разработку и реализацию автоматизированных тестов, а также анализ полученных результатов. В результате будет создана надежная и эффективная система тестирования, способная обеспечить высокое качество и устойчивость работы АИС «Зачисление в учреждения профессионального образования».

					ПГУ 1.090301.36.001 ПЗ	Лист
						4
Изм.	Лист	№ докум.	Подпись	Дата		

1 Анализ предметной области

Программное обеспечение (ПО) представляет собой совокупность программных компонентов, которые обеспечивают функциональность компьютерной системы или устройства. Тестировщики программного обеспечения – это специалисты, занимающиеся проверкой и обеспечением качества программного продукта. Они разрабатывают тестовые сценарии, выполняют тестирование и анализируют результаты, чтобы обеспечить соответствие программного продукта ожиданиям пользователей и требованиям.

Тестирование ПО – это процесс проверки соответствия между реальным и ожидаемым поведением программы, осуществляемый на конечном наборе тестов, выбранном определенным образом. Цель тестирования заключается в обеспечении пользователю качественного программного продукта. Важно, чтобы первое впечатление от продукта было положительным, поскольку в противном случае пользователи могут обратиться к альтернативным вариантам.

Тестирование программного обеспечения является неотъемлемой частью процесса разработки и играет ключевую роль для всех участников этого процесса. Для разработчиков тестирование служит инструментом для проверки правильности реализации функциональности и выявления возможных дефектов в коде. Они стремятся создать стабильный и надежный продукт, который будет соответствовать ожиданиям пользователей [1].

Для конечных пользователей тестирование обеспечивает гарантию того, что программное обеспечение работает корректно и безопасно. Пользователи ожидают получить качественный продукт, который будет выполнять свои функции без сбоев и ошибок, обеспечивая им уверенность в его безопасности и защите их данных.

Также тестирование значимо для менеджеров проектов и заказчиков, которые отвечают за успешное завершение проекта и удовлетворение потребностей клиентов. Они используют результаты тестирования для оценки качества и готовности продукта к выпуску на рынок, принимая обоснованные

решения о времени и способе его запуска. Тестирование позволяет минимизировать риски и обеспечивает успех проекта, обеспечивая высокое качество, надежность и безопасность программного продукта [2,3].

1.1 Виды тестирования

Виды тестирования можно классифицировать по самым разным критериям, например, по доступности кода (черный, серый, белый ящик), по объекту тестирования (интерфейс пользователя, локализация, скорость и надежность, безопасность, удобство использования, совместимость, функциональность), по сценарию использования (позитивное, негативное), по времени проведения (альфа, бета-тестирование, дымовое, регрессионное, приемочное тестирование, а также тестирование соответствия), по уровням тестирования (компонентное, интеграционное, системное), по степени автоматизации (ручное, автоматизированное, интуитивное), по субъекту (штатный, внештатный тестировщик) [4]. Данные виды тестирования отображены на рисунке 1.

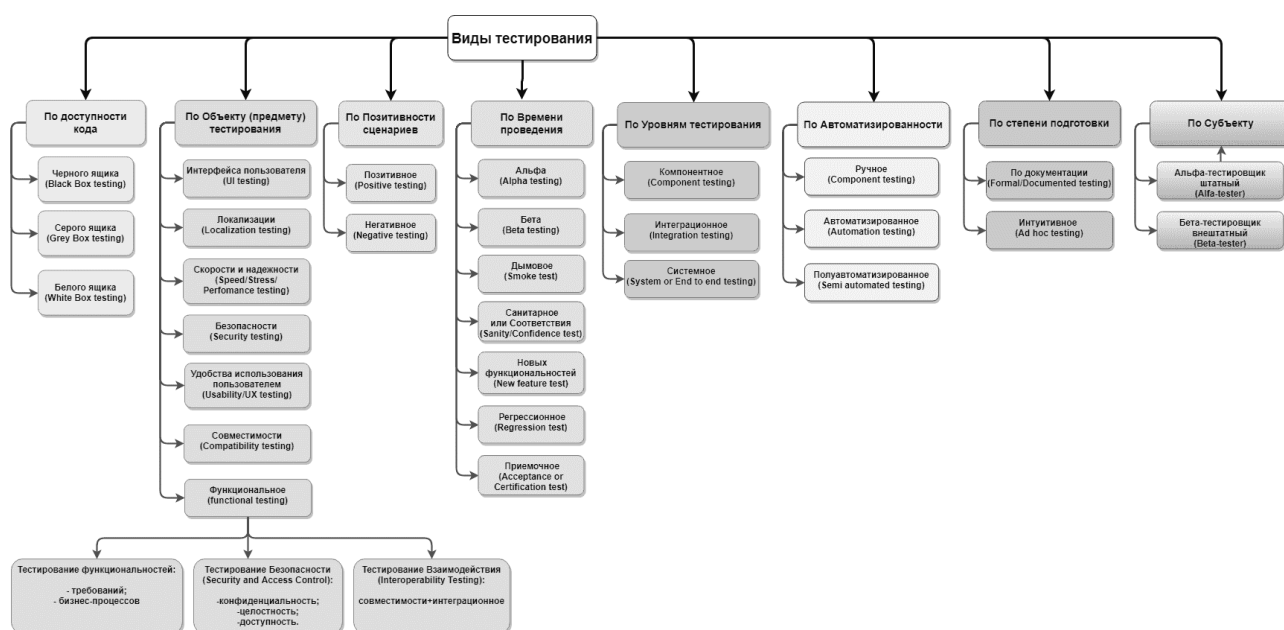


Рисунок 1 – Виды тестирования

Рассмотрим подробнее некоторые наиболее значимые виды тестирования.

1.1.1 Функциональное тестирование

Функциональное тестирование – это процесс проверки соответствия функциональных возможностей программного продукта его спецификациям, требованиям и ожиданиям пользователей. Основная цель функционального тестирования заключается в том, чтобы убедиться в том, что программа выполняет свои функции правильно и соответствует ожиданиям пользователей. Этот вид тестирования проверяет то, как программа реагирует на определенные входные данные, как она обрабатывает различные операции и какие результаты она возвращает [5].

Преимущество функционального тестирования заключается в гарантии соответствия программы заявленным требованиям и спецификациям. Он также помогает выявить ошибки и недоработки в реализации программного продукта, такие как неправильная обработка входных данных или некорректное поведение функций. Благодаря функциональному тестированию разработчики могут обнаруживать и исправлять ошибки на ранних этапах разработки, способствуя повышению общего качества программного продукта, а тестировщики своевременно сообщать команде о проблемах в функционале программы.

К основным этапам функционального тестирования можно отнести планирование, подготовку тестовых данных, выполнение тестов, анализ результатов и подготовка отчётности.

Планирование – это определение целей тестирования, выбор тестовых сценариев и разработка тестового плана.

Подготовка тестовых данных подразумевает создание набора тестовых данных, которые будут использоваться для проведения тестов.

Выполнение тестов – непосредственно запуск тестовых сценариев с использованием подготовленных тестовых данных и регистрация результатов.

Анализом результатов является оценка результатов тестирования, идентификация дефектов и их документирование для последующего исправления.

Последним этапом является отчетность – подготовка отчета (соответствующей документации) о проведенном тестировании, описание найденных дефектов и рекомендации по их устранению.

Так, функциональное тестирование представляет собой процесс проверки программного продукта на соответствие заявленным требованиям и спецификациям, а также на корректность работы основных функций и операций. Оно подразумевает несколько различных подходов и методов, каждый из которых направлен на проверку определенных аспектов функциональности программы.

1.1.2 Тестирование интерфейса пользователя

Тестирование интерфейса пользователя (или UI тестирование) является еще одним довольно важным видом функционального тестирования, поскольку пользовательский интерфейс – это основной способ взаимодействия пользователя с программой. Цель UI тестирования заключается в том, чтобы убедиться в корректности работы интерфейса, что легко воспринимаем и удобен для конечного пользователя.

Наиболее важными проверками тестирования интерфейса являются проверка внешнего вида, навигации, ввода данных, отображения данных, реакции на действия пользователя. Рассмотрим подробнее каждую из проверок.

Проверка внешнего вида – проверка соответствия дизайну, стилю и корпоративной идентичности. Например, проверка цветовой гаммы, шрифтов, размеров элементов управления.

Проверка навигации. Данная проверка помогает убедиться в том, что пользователь может легко перемещаться между различными разделами и функциями программы. К примеру, тестирование работы меню, кнопок навигации, гиперссылок.

Проверка ввода данных пользователем. Это может быть проверка работы полей ввода, выпадающих списков, флажков, радиокнопок.

Проверка корректности отображения данных в интерфейсе. Например, проверка таблиц, списков, графиков, диаграмм.

Проверка реакции на действия пользователя помогает убедиться, что программа корректно реагирует на действия пользователя. Например, работы кнопок, форм отправки данных, всплывающих окон.

Примером тестирования интерфейса пользователя может служить проверка работы онлайн-магазина. При тестировании можно проверить, что все кнопки «Добавить в корзину» корректно выполняют свои функции, что страницы категорий товаров отображаются правильно и что процесс оформления заказа проходит без каких-либо проблем.

Недостатки тестирования пользовательского интерфейса заключаются в том, что такой подход сосредотачивается главным образом на внешнем виде программы, игнорируя при этом более глубокие функциональные аспекты. Это может привести к упущению важных проблем в работе программы, которые не связаны с ее визуальным оформлением. Тестирование пользовательского интерфейса обычно требует значительных усилий на поддержку, особенно в случае изменений в дизайне или интерфейсе программы. Необходимо пересматривать и актуализировать существующие тест-кейсы, что может быть довольно трудоемким процессом. Также стоит учитывать тот факт, что автоматизация тестирования интерфейса может быть сложной из-за разнообразия элементов управления и их взаимодействия, и может сказаться на эффективности и скорости процесса тестирования.

1.1.3 Тестирование удобства использования

Тестирование удобства использования, также известное как тестирование пользовательского опыта или UX-тестирование, направлено на оценку того, насколько легко и удобно пользователи могут взаимодействовать с программным продуктом. Оно направлено на такие аспекты, как навигация, дизайн интерфейса, удобство взаимодействия с элементами пользовательского интерфейса и общее впечатление от использования продукта.

Основная цель тестирования удобства использования – улучшение пользовательского опыта и повышение удовлетворенности пользователей продуктом [6].

Примеры тестирования удобства использования могут варьироваться в зависимости от типа продукта. Так, для веб-сайтов это могут быть тесты на навигацию, проверка скорости загрузки страниц, доступность контента для пользователей с ограниченными возможностями и т.д. Для мобильных приложений же это могут быть тесты на удобство навигации по интерфейсу, удобство ввода данных, визуальное оформление и т.д.

Недостатки такого тестирования могут заключаться в том, что оценка удобства использования является в большей степени субъективной и может зависеть от индивидуальных предпочтений пользователей. Также трудно оценить эффективность тестирования удобства использования в условиях, когда пользователи имеют разные уровни опыта или различные потребности.

1.1.4 Ручное тестирование

Ручное тестирование – вид тестирования программного обеспечения без использования автоматизированных инструментов/скриптов. Проверки осуществляются вручную специалистами по тестированию (тестирующими или QA-инженерами).

Цель ручного тестирования заключается в проверке работоспособности и соответствии программного продукта его требованиям и ожиданиям пользователей [7].

Тестирование может проходить по заранее написанным тест-кейсам или чек-листам. Проверка по тест-кейсам является пошаговой, позволяя выяснить, на каком точном этапе произошла ошибка или несоответствие требованиям. Это, безусловно, является одним из плюсов ручного тестирования.

Несмотря на множество преимуществ автоматизированного тестирования, ручное тестирование имеет и некоторые недостатки. В частности, оно требует значительных временных и человеческих ресурсов, особенно для выполнения повторяющихся и объемных тестов. Ручное тестирование подвержено человеческим ошибкам (повторяя, например, 10 раз одни и те же шаги, можно попросту упустить незаметную ошибку) и может быть менее эффективным в

					ПГУ 1.090301.36.001 ПЗ	Лист
						10
Изм.	Лист	№ докум.	Подпись	Дата		

случае необходимости быстрой проверки больших объемов программного кода или функциональности.

1.1.5 Регрессионное тестирование

Регрессионное тестирование – это процесс проверки программного обеспечения на предмет того, что изменения, внесенные в код или функциональность программы, не привели к появлению новых ошибок или не повлияли на уже существующую функциональность. Цель регрессионного тестирования заключается в обеспечении стабильности и надежности программы после каждого внесения изменений.

Примером регрессионного тестирования может быть изменение модальных окон веб-приложения. Например, разработчики пересмотрели затемнение экрана при открытии модального окна из-за того, что раньше происходило «наслоение» экранов друг на друга, вызывая полное затемнение экрана. Необходимо убедиться, что изменения в работе модальных окон не повлияло на остальной функционал приложения. Например, что всплывающие сообщения-предупреждения не будут находиться за затемненным экраном при открытии различных типов модальных окон.

Другим примером может служить веб-приложение для онлайн-покупок. Предположим, что были внесены изменения в систему оплаты товаров. Перед выпуском обновления на продуктивный контур необходимо провести регрессионное тестирование, чтобы убедиться, что изменения в оплате не повлияли на другие функции приложения, такие как добавление товаров в корзину или просмотр каталога товаров.

Регрессионное тестирование можно автоматизировать. Такие тесты можно запускать после каждого обновления кода или функциональности.

Однако, регрессионное тестирование имеет и некоторые недостатки. Для проведения регрессионных тестов требуется значительное количество времени и ресурсов, особенно в случае больших и сложных проектов. С увеличением объема программного кода и функциональности, количество регрессионных тестов может быстро возрасти, что также затруднит их проведение и поддержку. Следует

					ПГУ 1.090301.36.001 ПЗ	Лист
						11
Изм.	Лист	№ докум.	Подпись	Дата		

учитывать, что проведение регрессионных тестов не гарантирует полного отсутствия ошибок в программе, поскольку некоторые проблемы могут остаться незамеченными из-за ограниченности времени или недостаточного охвата тестами.

1.1.6 Модульное тестирование

Модульное тестирование (блочное тестирование или юнит-тестирование) – это метод тестирования программного обеспечения, который сосредотачивается на проверке отдельных модулей, функций или методов программы. Основная идея состоит в том, чтобы изолировать каждый модуль от остальной части программы и протестировать его поведение в изоляции.

При модульном тестировании создаются тестовые случаи, которые проверяют различные аспекты работы конкретного модуля. Содержит проверку входных данных, ожидаемых результатов и побочных эффектов при различных условиях. Тесты обычно разрабатываются и запускаются автоматически, обеспечивая быструю обратную связь о работоспособности кода [8].

Преимущества модульного тестирования заключаются в повышении качества кода путем выявления дефектов на ранних этапах разработки, облегчение рефакторинга и изменений в коде благодаря наличию надежных тестов, а также ускорение процесса разработки за счет быстрой проверки отдельных компонентов.

Пример модульного тестирования может быть следующим. Допустим, что есть функция, которая вычисляет сумму двух чисел. В модульном тесте можно проверить, что функция корректно возвращает ожидаемый результат для различных комбинаций входных данных и не вызывает непредвиденных побочных эффектов.

Так, модульное тестирование является важным инструментом в разработке программного обеспечения, который помогает обеспечить надежность, стабильность и качество кода.

1.1.7 Smoke-тестирование

					ПГУ 1.090301.36.001 ПЗ	Лист
						12
Изм.	Лист	№ докум.	Подпись	Дата		

Smoke-тестирование (реже дымовое) представляет собой короткий и быстрый цикл тестирования, направленный на проверку ключевых функций приложения. Оно необходимо для того, чтобы убедиться в правильной работе основных компонентов программного продукта.

Частое обновление программного обеспечения может привести к появлению ошибок, которые могут оказать серьезное влияние на бизнес-процессы. Поэтому важно проводить проверку основной функциональности системы как перед передачей программного продукта на тестирование, так и сразу после сборки обновления.

В качестве примера smoke-тестирования может служить веб-приложение для онлайн-бронирования гостиниц. Перед каждым обновлением системы разработчики проводят smoke-тестирование, чтобы убедиться, что функционал поиска гостиниц, выбора даты проживания и оформления бронирования работает корректно.

Следует отметить, что smoke-тестирование не охватывает все аспекты приложения и может не выявить некоторые скрытые проблемы или ошибки, которые могут возникнуть при взаимодействии различных компонентов системы.

Регрессионное тестирование и smoke-тестирование являются двумя различными подходами к тестированию программного обеспечения, хотя и имеют некоторые сходства. Основное различие заключается в их целях и охвате. Регрессионное тестирование более полно исследует влияние изменений на всю систему, в то время как smoke-тестирование сконцентрировано на быстрой проверке ключевых функций для обеспечения минимального уровня работоспособности.

1.1.8 Тестирование безопасности

Тестирование безопасности (тестирование уязвимостей или тестирование безопасности информационной системы) – еще один важнейший вид тестирования программного обеспечения. Он направлен на выявление и устранение уязвимостей, угроз и рисков, которые могут быть использованы злоумышленниками для незаконного доступа к системе, поиску лазеек и слабых

мест ПО, утечки конфиденциальной информации или нарушения целостности данных.

Существуют основные принципы безопасности – конфиденциальность, целостность и доступность.

Все продукты должны быть защищены от утечек данных, несанкционированного доступа, взлома, хищения личных данных, намеренного удаления или изменения этих данных, перегрузки системы и так далее. Особенно это важно для продуктов компаний, занимающихся хранением огромного количества конфиденциальных данных. Нарушение принципов безопасности могут повлечь за собой как репутационные убытки, так и финансовые. Тестированием безопасности как правило занимается специалист ИБ (информационная безопасность) [9].

1.1.9 Интеграционное тестирование

Интеграционное тестирование – это процесс проверки взаимодействий между компонентами не только одной системы, но и между различными системами. В современном информационном мире множество ИТ-систем взаимодействуют между собой, и эффективное функционирование этих взаимосвязей играет ключевую роль для обеспечения бесперебойной работы.

Интеграционное тестирование нацелено на корректность обмена данными между различными модулями системы. Например, проверка того, что удаленные из почтового ящика письма помещаются в другой компонент системы как «Корзина». В таком случае важна не просто проверка функций удаления, а именно обмена данными с другими составляющими системы.

Интеграционное тестирование помогает минимизировать риски возникновения ошибок при внедрении новых компонентов или обновлении существующих систем, способствуя повышению качества программного обеспечения в целом.

1.2 Автоматизация тестирования

					ПГУ 1.090301.36.001 ПЗ	Лист
						14
Изм.	Лист	№ докум.	Подпись	Дата		

Автоматизированное тестирование – это процесс, при котором тестовые сценарии и проверки выполняются автоматически с использованием специальных инструментов и скриптов, вместо ручного выполнения тестов. Оно осуществляется с целью ускорения и улучшения качества процесса тестирования. Обычно автоматизированные тесты разрабатываются инженерами по тестированию или автоматизации. Но автоматизацию могут проводить и разработчики, особенно если речь идет о модульном тестировании или тестировании интеграции.

Автоматизированное тестирование полезно для команд разработки программного обеспечения, поскольку оно позволяет повысить эффективность тестирования, сократить время на выполнение тестовых сценариев и улучшить общее качество продукта. Автоматизация тестирования также очень важна в выявлении дефектов на ранних этапах разработки [10].

Основные этапы автоматизации тестирования содержат анализ и выбор тестов для автоматизации, разработку тестовых скриптов или сценариев, их запуск и отладку, интеграцию с системами управления версиями и непрерывной интеграции, а также поддержку и обновление автоматизированных тестов в процессе развития проекта.

Код для автоматизированных тестов должен быть хорошо структурирован и поддерживаем, чтобы облегчить понимание и поддержку тестов в будущем другими тестировщиками. Он также должен быть гибким и адаптивным, чтобы легко обновляться в случае изменений в приложении или его интерфейсе.

1.3 Возможности автоматизации тестирования

Каждая команда разработчиков сталкивается с тем, что времени на внедрение передовых практик остается недостаточно. Такая же ситуация характерна и для стратегии тестирования, которая, как деятельность, не всегда ставится в приоритет. Поэтому на пути приходится искать баланс и принимать взвешенные решения в зависимости от характера разрабатываемого приложения, ограничений по времени, используемого ПО для тестирования и наличия

					ПГУ 1.090301.36.001 ПЗ	Лист
						15
Изм.	Лист	№ докум.	Подпись	Дата		

ресурсов [11,12]. Рассмотрим подробнее некоторые виды тестов, нуждающиеся в автоматизации.

Модульное тестирование, в котором упор делается на тестирование отдельных компонентов кода. Это позволяет разработчикам получить более полное представление о работе созданных функций.

Приоритетные функции, где акцент делается на тестировании функций, вероятность сбоев которых оценивается как высокая.

Регрессионные и интеграционные тесты направлены на проверку работоспособности приложения после внесенных изменений. Их автоматизация экономит значительное количество времени.

Нагрузочные тесты и тесты производительности, где необходимо моделировать множество пользователей, использующих приложение в различных условиях.

Повторяющиеся и часто выполняющиеся тестовые сценарии. Например, тестирование работы функции входа в систему.

Базовая функциональность, также известная как «дымовые тесты», обеспечивает проверку основных функций приложения, например, доступности системы для пользователей.

Тем не менее, не все тесты можно и нужно автоматизировать. Так, например, удобство использования невозможно автоматизировать, для этого обязательно нужен человек, то есть ручной тестировщик. Нецелесообразно автоматизировать и продукты на ранней стадии разработки, функциональность с низкой важностью или тесты, результаты которых недостаточно ясны [13].

1.4 Объект тестирования

Объектом тестирования является автоматизированная информационная система (АИС) «Зачисление в учреждения профессионального образования».

АИС предназначена для:

– автоматизации процессов оказания услуг по приему заявлений о поступлении в профессиональные образовательные организации (ПОО);

					ПГУ 1.090301.36.001 ПЗ	Лист
						16
Изм.	Лист	№ докум.	Подпись	Дата		

- снижения трудозатрат при управлении контингентом ПОО;
- обеспечения оперативного информирования сотрудников, ответственных за контроль исполнения процесса управления контингентом ПОО;
- учета и хранения юридически значимых сведений процесса обучения по образовательным программам среднего профессионального образования (СПО);
- обеспечения возможности формирования аналитической отчетности.

В качестве пользователя будет выступать «Оператор ПОО» – специально обученный сотрудник образовательной организации.

Данная роль может выполнять следующие функции:

- работа (просмотр, ввод, редактирование, создание, удаление, выгрузка) с заявлениями, приказами, личными делами, образовательными программами, учебными группами, экзаменационными группами, реестрами, отчетами;
- просмотр статических файлов (инструкции и руководства);
- заполнения атрибутов на форме личного дела;
- просмотр данных на форме группового помещения на карантин.

В данной работе автоматизация тестирования будет проводиться с разделом работы с приказами.

Приказы бывают как индивидуальными (по одному человеку), так и групповые (по нескольким людям).

В ходе развития системы количество приказов увеличивается. На текущий момент существуют следующие типы приказов:

- приказ о зачислении;
- приказ об отчислении;
- приказ о восстановлении;
- приказ о предоставлении академического отпуска;
- приказ о выпуске;
- приказ о переводе из группы в группу;
- приказ о переводе с курса на курс;
- приказ о переводе на программу эффективного учебного плана;

– приказ о переводе на программу базового учебного плана.

Каждый приказ может находиться в ряде статусов:

- «проект приказа»;
- «на рассмотрении»;
- «отклонен»;
- «выпущен»;
- «удален».

Все созданные приказы находятся в разделе «Книга учета приказов». Так как количество приказов может достигаться сотен и тысяч, оператору предоставляется возможности отображения по нужному количеству приказов, поиска по номеру приказа, сортировки, фильтрации (тип, дата, статус). Также пользователю доступен следующий функционал: выгрузка данных списка приказов (экспорт в Excel), переход к приказу и удаление приказа из меню дополнительных действий.

1.5 Постановка задачи

Необходимо разработать функциональное автоматизированное тестирование создания и выпуска индивидуальных и групповых приказов для обучающихся профессионально-технических образовательных учреждений.

В данной программе будет реализована часть автоматизированного тестирования системы, отвечающая за создание приказов о зачислении, восстановлении, об отчислении, переводе из группы в группу, с курса на курс, на программу базового учебного плана (БУП), на программу эффективного учебного плана (ЭУП).

В качестве примера выбрана проверка работы создания группового приказа об отчислении студента из образовательной организации. Автотест будет основываться на ручном тест-кейсе, шаги и ожидаемый результат которого описаны в таблице 1.

Таблица 1 – Тест-кейс «Групповой приказ отчисления»

					ПГУ 1.090301.36.001 ПЗ	Лист
						18
Изм.	Лист	№ докум.	Подпись	Дата		

Шаг №	Шаг теста	Ожидаемый результат
1	Авторизоваться в системе пользователем с ролью «Оператор ПОО»	Пользователь авторизован, отображен рабочий стол
2	По меню перейти в «Создание приказа» раздела «Приказы» и выбрать тип «Приказ об отчислении»	<p>Отобрана страница создания приказа с элементами:</p> <ul style="list-style-type: none"> - Блок «Сведения о групповом приказе» - Блок «Обучающийся на отчисление» - Кнопка «Корзина» (неактивна, до тех пор, пока в списке только один обучающийся) - Кнопка «Добавить еще обучающегося» (неактивна до тех пор, пока не будут заполнены обязательные поля) - Кнопка «Отмена» - Кнопка «Сохранить» (неактивна до тех пор, пока не будут заполнены все обязательные поля)
3	Заполнить все обязательные поля и нажать «Добавить еще обучающегося»	Отобразен дополнительный блок «Обучающийся на отчисление»
4	Заполнить все поля и нажать «Отмена»	Все поля заполнены, выдано модальное окно модальное окно «Прекращение работы» с текстом: «Вы уверены, что хотите прекратить работу? Внесенные данные не будут сохранены» и кнопками «Крестик», «Отмена», «Прекратить»
5	Нажать «Отмена»	Модальное окно закрыто. Отобрана страница создания

Шаг №	Шаг теста	Ожидаемый результат
		приказа в режиме редактирования
6	Очистить любое обязательное поле (обязательность поля отмечена символом «*»)	Отобрана подсказка об обязательности заполнения поля. Кнопка «Создать приказ» неактивна
7	Заполнить обязательное поле и нажать «Создать приказ»	Выдано уведомление «Групповой приказ №[номер приказа] успешно создан». Отобрана страница приказа. В шапке отображены цепочка навигации (Приказы Книга учета приказов Групповой приказ об отчислении), заголовок «Групповой приказ об отчислении», бейдж «проект приказа», кнопка «Действия»
8	Нажать кнопку-карандаш блока «Сведения о групповом приказе»	Открыто модальное окно «Редактирование сведений приказа»
9	Очистить обязательное поле	Отобрана подсказка об обязательности заполнения поля. Кнопка «Сохранить» неактивна
10	Нажать «Отмена»	Выдано модальное окно модальное окно «Прекращение работы» с текстом: «Вы уверены, что хотите прекратить работу? Внесенные данные не будут сохранены» и кнопками «Крестик», «Отмена», «Прекратить»
11	Нажать «Отмена» на модальном окне	Модальное окно закрыто
12	Заполнить обязательное поле и нажать «Сохранить»	Модальное окно закрыто. Выдано уведомление «Изменения в приказе успешно сохранены».

Шаг №	Шаг теста	Ожидаемый результат
13	Нажать кнопку-карандаш одного из обучающихся в блоке «Список обучающихся на отчисление»	Открыто модальное окно «Редактирование сведений об отчисляемом студенте»
14	Очистить обязательное поле	Отобразена подсказка об обязательности заполнения поля. Кнопка «Сохранить» неактивна
15	Нажать «Отмена»	Выдано модальное окно модальное окно «Прекращение работы» с текстом: «Вы уверены, что хотите прекратить работу? Внесенные данные не будут сохранены» и кнопками «Крестик», «Отмена», «Прекратить»
16	Нажать «Отмена» на модальном окне	Модальное окно закрыто
17	Заполнить обязательное поле и нажать «Сохранить»	Модальное окно закрыто. Выдано уведомление «Изменения в приказе успешно сохранены»
18	Нажать на кнопку «Корзина» любого обучающегося	Отображено модальное окно «Удаление обучающегося из приказа» с текстом: «Вы уверены, что хотите удалить обучающегося [ФИО обучающегося] из приказа?» и кнопками «Отмена», «Удалить»
19	Нажать «Отмена» на модальном окне	Модальное окно закрыто
20	Нажать на кнопку «Корзина» любого обучающегося, в модальном окне нажать «Удалить»	Модальное окно закрыто, выдано уведомление «Обучающийся успешно удален из приказа»
21	Нажать «Действия» и выбрать	Выдано модальное окно «Вы уверены, что хотите выпустить

Шаг №	Шаг теста	Ожидаемый результат
	«Выпустить приказ»	приказ №[номер приказа]?» с кнопками «Отмена» и «Выпустить»
22	Нажать «Выпустить»	Выдано уведомление с текстом: «Групповой приказ №[номер приказа] успешно выпущен». Модальное окно закрыто. Отобрана страница приказа. В шапке отображены цепочка навигации (Приказы Книга учета приказов Групповой приказ об отчислении), заголовок «Групповой приказ об №[номер приказа]» и данными приказа («От» и «На основании ... от ...»), бейдж «Выпущен», кнопка «Отклонить» Блок «Список обучающихся на отчисление» отображен в режиме просмотра с кнопками «Свернуть/развернуть»
23	Нажать на кнопку «Развернуть» одного из обучающихся	Отображены сведения в режиме просмотра
24	Нажать «Отклонить»	Открыто модальное окно «Отклонение приказа №[номер приказа]»
25	Нажать на выпадающий список поля «Причина отклонения»	Выдан список с вариантами «Ошибка оператора», «Изменение обстоятельств, связанных с выпуском приказа»
26	Выбрать любой вариант и нажать «Отклонить»	Выдано уведомление «Групповой приказ №[номер приказа] успешно отклонен и переведен в статус «Проект приказа». Отобрана страница приказа с бейджем «проект приказа»

Шаг №	Шаг теста	Ожидаемый результат
27	Нажать «Действия» и выбрать «Удалить приказ»	Выдано модальное окно «Удаление проекта приказа» с текстом «Вы уверены, что хотите удалить приказ №[номер приказа]? Это действие будет необратимо» и кнопками «Удалить» и «Отмена»
28	Нажать «Удалить»	Выдано уведомление с текстом: «Групповой приказ №[номер приказа] успешно удалён». Модальное окно закрыто. Отображена страница приказа с бейджем «удален»
29	Нажать на значок профиля в правом верхнем углу и выбрать в меню «Выход»	Отображена страница авторизации

В результате выполнения тест-кейса будет создан и выпущен приказ об отчислении, после чего приказ отклоняется и удаляется.

Пользователь (Оператор ПОО) имеет возможность:

- перейти по меню;
- создать новый приказ (выбрать тип приказа, добавить и/или удалить учащегося, редактировать сведения, прекратить работу с приказом);
- отклонить и удалить приказ;
- просмотреть приказ.

Основные возможности данной проверки отражены на UML-диаграмме вариантов использования (рисунок 2).

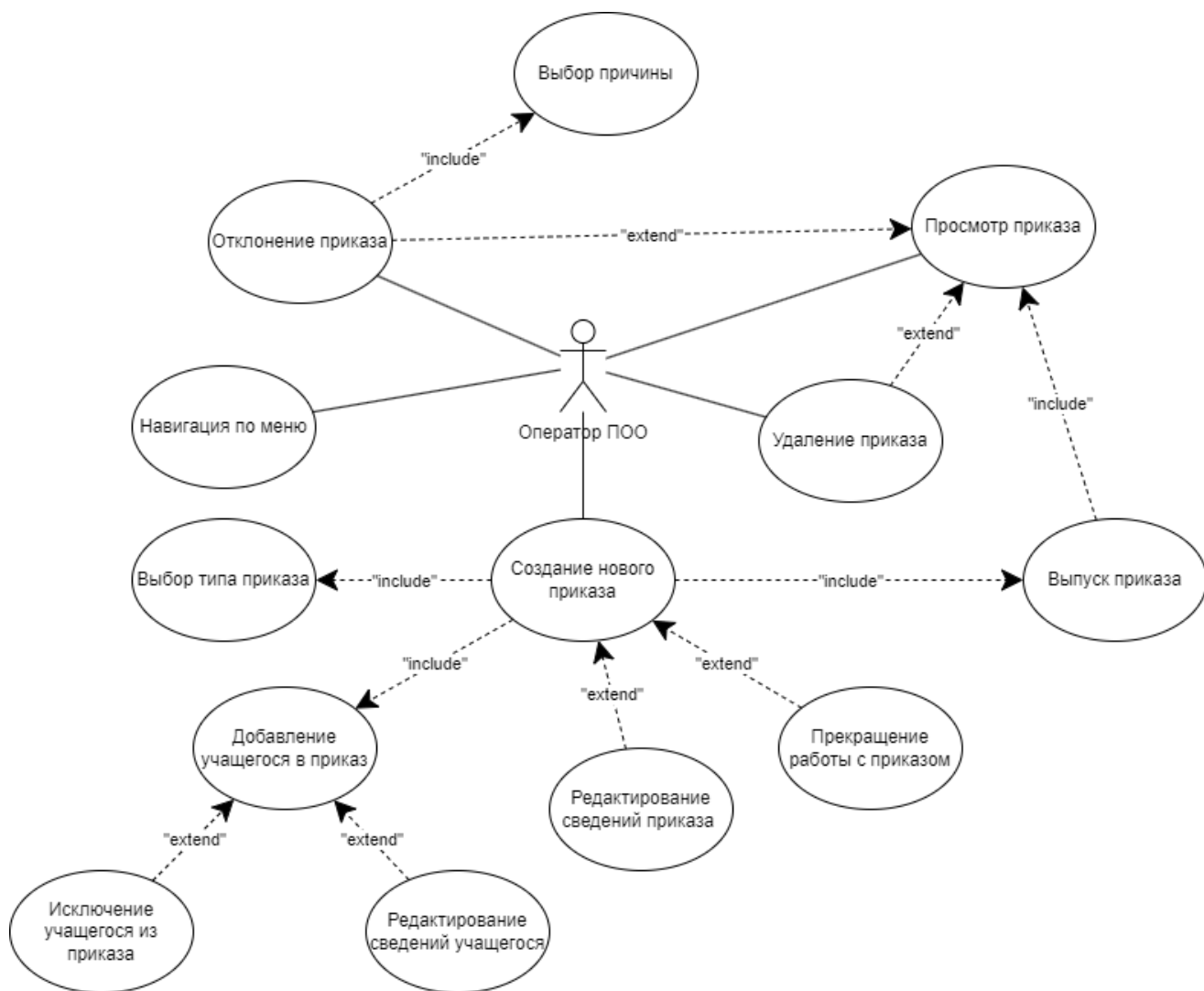


Рисунок 2 – UML-диаграмма вариантов использования «Создание группового приказа об отчислении»

2 Выбор инструментов тестирования

Перед началом любой разработки, будь то написание приложения или автоматизация тестирования, встает вопрос о выборе языка программирования, среды разработки, целесообразности использования фреймворков и плагинов для среды разработки. Существует множество мнений по поводу выбора инструментов разработки. Так, что-то, что считается устаревшим, довольно часто используется разработчиками. Это мнение довольно субъективное и должно опираться непосредственно на цели разработки, ее предназначение и команду внутри проекта.

2.1 Выбор языка программирования

Для разработки автотестов для веб-сервиса необходимо выбрать наиболее подходящий язык программирования. Рассмотрим некоторые из них.

2.1.1 Python

Python является высокоуровневым языком программирования, который используется в веб-разработке, создании мобильных и десктопных приложений и тестировании, анализе данных и машинном обучении. Обладает очень минималистичным синтаксисом, понятен без обращения к документации. Язык поддерживает множество библиотек и фреймворков для автоматизации тестирования, например Selenium, Appium, PyUnit, Pytest и другие. В качестве недостатка можно выделить более низкую скорость работы и высокое потребление памяти, если сравнивать код с аналогичным на другом языке.

2.1.2 C#

C# (C Sharp) – объектно-ориентированный язык программирования от компании Microsoft. Используется для разработки web-приложений, клиент-серверных приложений, автоматизированного тестирования и другого. В некотором роде является аналогом Java.

Синтаксис языка напоминает C++ и Java. Поддерживает полиморфизм, перегрузку операторов. Язык легко интегрируется с продуктами Microsoft.

					ПГУ 1.090301.36.001 ПЗ	Лист
						25
Изм.	Лист	№ докум.	Подпись	Дата		

Поддерживает библиотеки для автоматизации тестирования, например, Selenium WebDriver, NUnit, MSTest. Существенным минусом является отсутствие кроссплатформенности.

2.1.3 Java

Java является популярным объектно-ориентированным языком программирования для разработки мобильных приложений, десктопных и web-приложений, а также автоматизированных тестов.

Язык программирования кроссплатформенным, то есть может быть запущен на разных ОС. поддерживает множество фреймворков, таких как Selenium WebDriver, TestNG, JUnit, сборщик проектов Maven. Соблюдается строгая типизация, что очень важно для крупных проектов, обладает лаконичным синтаксисом. Так как Java является популярным языком программирования с активным сообществом разработчиков, за годы существования он оброс обширной документацией [14,15].

Исходя из достоинств и недостатков вышеперечисленных языков программирования, для разработки автоматизированных тестов был выбран язык Java.

2.2 Выбор фреймворков для автоматизации тестирования

В данном разделе будут рассмотрены различные фреймворки для автоматизации тестирования, их особенности, преимущества и недостатки для выбора наиболее подходящего инструмента для разработки автотестирования.

2.2.1 Selenium

Selenium (или Selenium WebDriver) является одним из главных open source инструментов для тестирования приложений. Данная библиотека уже много лет пользуется популярностью среди разработчиков. Представляет собой семейство драйверов различных браузеров (Firefox, Safari, Opera, Google Chrome). Также поддерживает мобильные браузеры для Android и IOS. Легко интегрируется с другими инструментами тестирования.

					ПГУ 1.090301.36.001 ПЗ	Лист
						26
Изм.	Лист	№ докум.	Подпись	Дата		

2.2.2 Selenide

Selenide представляет собой еще одну библиотеку для автоматизации тестирования, созданный на основе Selenium WebDriver. Содержит в себе все преимущества Selenium, добавляя новые. Так, есть возможности автоматического сохранения скриншотов при провале теста, простое подключение через зависимости и импорты, довольно лаконичный код, поддержка паттерна проектирования Page Object Model. Нет необходимости долго искать причину падения тестов – нужное место подсветится в коде и отобразится информация о сбое. Используется автоматическое ожидание элементов на странице.

Взвесив все достоинства и недостатки, для разработки автотестов было решено выбрать Selenide.

2.2.3 JUnit

JUnit является фреймворком для автоматизированного тестирования. Поддерживает разные языки программирования, например, PHP, C#, JavaScript, Java. Поддерживает аннотации (специальные методы перед тестовыми методами), позволяет легко проверять и запускать тестовые сценарии. Довольно просто подключается к IDE через добавление зависимостей.

2.2.4 TestNG

TestNG – фреймворк для тестирования приложений на Java. Основан на JUnit. Позволяет проводить функциональные и интеграционные тесты, генерировать отчеты о прохождении тестирования. Поддерживает только один язык программирования – Java. Легко интегрируется в среду разработки. Фреймворк содержит тест-сюиты, отдельные тесты и аннотации, возможность параллельного запуска автотестов, пропуска и объединения тестов.

Для управления тестированием выбор сделан в пользу фреймворка TestNG.

2.3 Выбор среды разработки

Выбор интегрированной среды разработки является необходимым шагом для создания любой разрабатываемой системы. Среда разработки позволяет

					ПГУ 1.090301.36.001 ПЗ	Лист
						27
Изм.	Лист	№ докум.	Подпись	Дата		

разработчикам писать, проверять, отлаживать код, тестировать программы, а также вести совместную разработку крупных проектов.

Существует множество различных IDE. Их можно разделить на разные категории. Так, существуют бесплатные, условно-бесплатные и полностью платные среды. Бесплатные, как правило, выбирают начинающие разработчики. Зачастую функционал таких IDE ограничен, но достаточен для изучения программирования, учебы или небольших проектов. Платные среды разработки используют уже более опытные программисты и команды разработки. Программы обладают довольно широким функционалом, лучшей поддержкой, но могут быть более требовательны к знаниям пользователей, характеристикам компьютера и занимаемому месту. Существуют одноязычные и мультязычные среды разработки. Здесь выбор зависит от требуемого языка или языков программирования. Не все IDE совместимы с известными операционными средами. Как правило, в крупных компаниях заранее известна ОС и подобрана наиболее подходящая под нее среда разработки. Еще одним важным критерием может быть удобство использования IDE. Например, удобный графический интерфейс, подсветка синтаксиса, автоматические сохранения, возможность установки плагинов и, наконец, наличие контроля версий, без которого не обойдется ни одна команда разработки.

Рассмотрим некоторые IDE, чтобы осуществить выбор рабочей среды для создания системы автоматизированного тестирования.

2.3.1 VSCode

VSCode является весьма популярным легким редактором кода, разработанным компанией Microsoft для различных операционных систем. Поддерживает множество расширений для добавления языков программирования, легко кастомизируется, имеет возможность использования контроля версий. Распространяется бесплатно.

2.3.2 NetBeans

NetBeans или Apache NetBeans – интегрированная среда разработки, поддерживающая различные языки программирования. IDE распространяется

					ПГУ 1.090301.36.001 ПЗ	Лист
						28
Изм.	Лист	№ докум.	Подпись	Дата		

бесплатно, легко интегрируется с разными фреймворками, доступна на самых известных операционных системах. Используется в создании web-приложений, десктопных и мобильных приложений. Изначально создавалась для языка Java. Также она имеет возможность создания и запуска автотестов. В качестве минусов можно выделить ограниченную производительность, медленную загрузку и реакцию на действия пользователя. Среда разработки недостаточно поддерживается, допускает ошибки редактирования кода и подсветки синтаксиса, имеет не самый приятный интерфейс.

2.3.3 Eclipse

Eclipse – интегрированная среда разработки. Она поддерживает различные языки программирования использованием плагинов, например, C/C++, Java, PHP. Преимущественно используется для разработки на Java. IDE является бесплатной, но без ограничения функционала, как бывает с большинством сред разработки. Также она поддерживает различные операционные системы, являясь кроссплатформенной. Так как Eclipse долгое время занимала лидирующие позиции используемых сред разработки под Java, для нее есть обширная документация, в которой можно найти все интересующие вопросы. Большое количество плагинов является как плюсом, так и минусом, потому что в них легко запутаться начинающему разработчику, а несовместимые модули могут вовсе положить систему, после чего потребуется переустановка среды разработки. Также выделяют такой недостаток, как неинтуитивный интерфейс, хотя этот момент является довольно спорным среды сообщества разработчиков.

2.3.4 IntelliJ IDEA

IntelliJ IDEA – одна из самых популярных сред разработки. Поддерживает Java, Kotlin, Groovy и другие языки программирования. Является кроссплатформенной. Существуют две версии программы – бесплатная Community edition и платная Ultimate edition с расширенной функциональностью. Поддерживает контроль версий, установку плагинов, хорошо интегрируется с различными фреймворками разработки и тестирования. Также присутствуют инструменты для работы с базами данных. IDE привлекает удобным и

настраиваемым интерфейсом, качественной подсветкой синтаксиса, автодополнением кода, анализом ошибок и предложениям по их исправлению.

В качестве среды разработки для разработки автоматизированных тестов была выбрана IntelliJ IDEA.

					ПГУ 1.090301.36.001 ПЗ	Лист
						30
Изм.	Лист	№ докум.	Подпись	Дата		

3 Проектирование системы

Для решения поставленной задачи необходимо разработать программный модуль на выбранном языке программирования, а именно Java, в котором должна быть реализована логика работы автотестирования.

Для удовлетворения всех потребностей пользователя, представленных на диаграмме прецедентов (приложение А.1), было принято решение реализовать класс OrderTest с нижеописанными методами.

createGroupAcademicLeaveOrder() – метод, описывающий групповой приказ о предоставлении академического отпуска.

createGroupEnrollmentOrder() – метод, описывающий групповой приказ о зачислении в образовательную организацию. Происходит выборка заявлений на зачисление студентов по общей специальности/профессии и зачисление в учебную группу.

createGroupExpulsionOrder() – метод, описывающий групповой приказ об отчислении из образовательной организации. Указывается причина отчисления, от которой зависит возможность дальнейшего восстановления студента.

createGroupRecoveryOrder() – метод, описывающий групповой приказ о восстановлении. Доступен студентам, находящимся в академическом отпуске или отчисленным по определенным причинам (например, по семейным обстоятельствам).

createGroupReleaseOrder() – метод, описывающий групповой приказ о выпуске из образовательной организации по причине завершения обучения.

createGroupTransferCourseOrder() – метод, описывающий групповой приказ о переводе с курса на курс. Например, со третьего на второй.

createGroupTransferOrder() – метод, описывающий групповой приказ о переводе из группы в группу. Может быть как в рамках одной образовательной программы, так и разных. Например, из группы «20ПК» образовательной программы «Повар, кондитер» в группу «20ГД» образовательной программы «Графический дизайнер».

`createGroupTransferToBupOrder()` – метод, описывающий групповой приказ на программу БУП (базовый учебный план).

`createGroupTransferToEupOrder()` – метод, описывающий групповой приказ на программу ЭУП (эффективный учебный план).

`createIndividualExpulsionOrder()` – метод, описывающий индивидуальный приказ об отчислении. Возможен только из личного дела обучающегося.

`orderAcademicLeave()` – метод, описывающий индивидуальный приказ о предоставлении академического отпуска. Возможен только из личного дела обучающегося.

`orderRealise()` – метод, описывающий индивидуальный приказ о выпуске. Возможен только из личного дела обучающегося.

`orderRecovery()` – метод, описывающий индивидуальный приказ о восстановлении. Возможен только из личного дела обучающегося.

`orderTransfer()` – метод, описывающий индивидуальный приказ о переводе из группы в группу. Возможен только из личного дела обучающегося.

`orderTransferBUP()` – метод, описывающий индивидуальный приказ о переводе на программу БУП. Возможен только из личного дела обучающегося.

`orderTransferEUP()` – метод, описывающий индивидуальный приказ о переводе на программу ЭУП. Возможен только из личного дела обучающегося.

`orderBudgetFinance()` – метод, описывающий индивидуальный приказ о смене финансирования. Например, с бюджета на договор или наоборот. Возможен только из личного дела обучающегося.

`orderEnroll()` – метод, описывающий индивидуальный приказ о зачислении. Возможен только из страницы заявления обучающегося.

`recoveryFromAcademicLeave()` – метод, описывающий автоматический приказ о восстановлении из академического отпуска при наступлении определенной даты.

Класс `OrderTest` наследуется от `BaseSelenideTest`, в конструкторе которого происходит необходимая настройка конфигурации веб-браузера, в котором будет реализован функционал дочернего класса `OrderTest`.

Также для удобства дальнейшей разработки программного модуля было решено использовать аннотацию Epic, чтобы было легче вести документацию кода.

Более подробное описание взаимодействия трех вышеописанных классов отображено на рисунке 3.

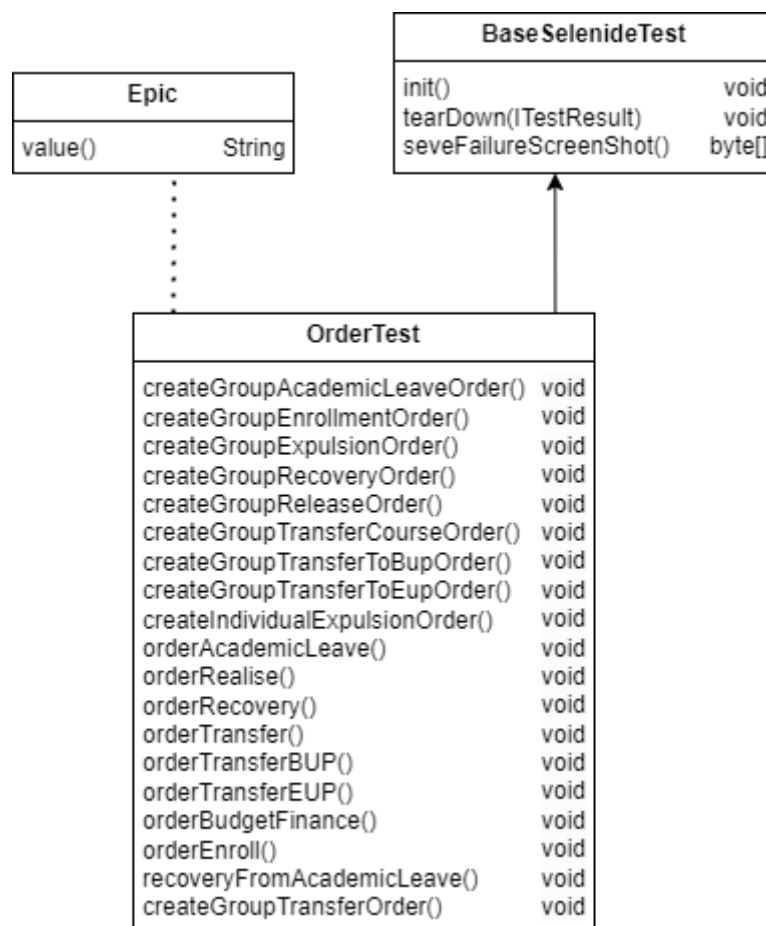


Рисунок 3 – UML-диаграмма классов «Приказы»

После описания всех основных сущностей разрабатываемого продукта следует описать жизненный цикл работы программного модуля. Ниже описаны основные шаги работы автотеста в части создания приказов.

При успешной авторизации в систему пользователь, переходя к созданию приказа, выбирает тип приказа, отправляя http-запрос. Далее происходит запрос к БД (базе данных) на поиск данных. Выполнив поиск, БД выдает результат в виде

списка типов приказов, которые с успешным http-ответом будут выданы пользователю в виде выпадающего списка типов приказов.

У каждого приказа существует своя форма. Чтобы получить данные, пользователь выбирает тип приказа. Далее действия выполняются аналогично запросу типов приказов.

В результате действий пользователя в БД будет добавлена (и обновлена) запись о новом приказе с соответствующим отображением его в интерфейсе.

UML-диаграмма последовательности предоставлена на рисунке 4.

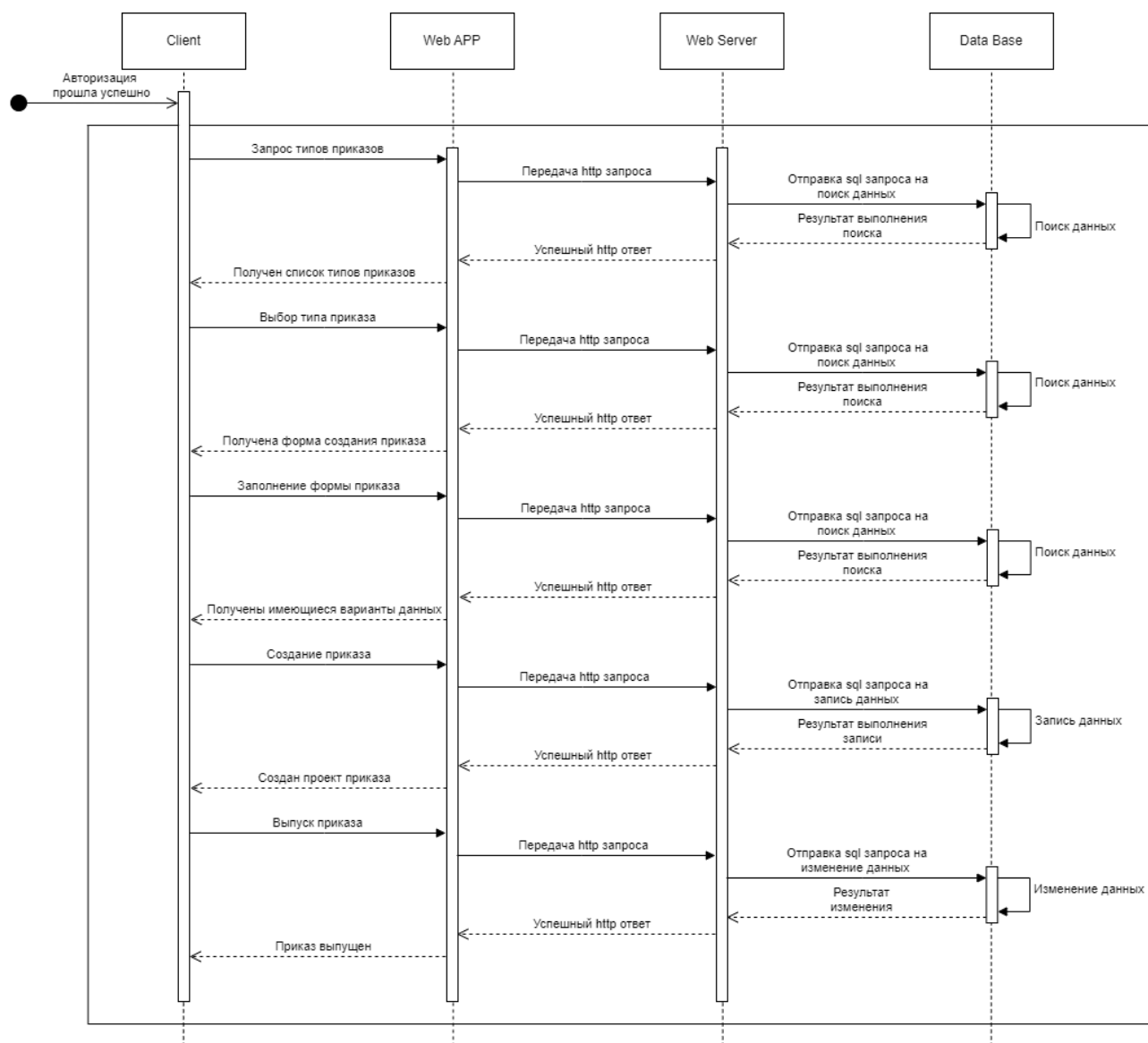


Рисунок 4 – UML-диаграмма последовательности «Создание и выпуск приказа»

Все UML-диаграммы данной работы отображены в приложении А.

4 Развертывание системы проектирования

Развертывание системы проектирования представляет собой процесс установки и настройки необходимых компонентов программного обеспечения для использования в рамках проектов.

Первым шагом является установка JDK – обязательного компонента для разработки и запуска Java-приложений. Затем рассматривается выбор среды разработки, обеспечивающей удобную и эффективную рабочую среду для написания кода и выполнения автотестов. Последним шагом устанавливаются и настраиваются выбранные фреймворки автоматизации и плагины для среды разработки.

Далее приведено описание процесса развертывания системы.

4.1 Подготовка окружения

4.1.1 Установка Java JDK

Для разработки приложений на языке Java, запуска автотестов и генерации отчетов тестирования Allure необходимо установить JDK (Java Development Kit).

Перед установкой необходимо перейти на официальный сайт Oracle для загрузки JDK и скачать нужную версию JDK для Windows (рисунок 5).

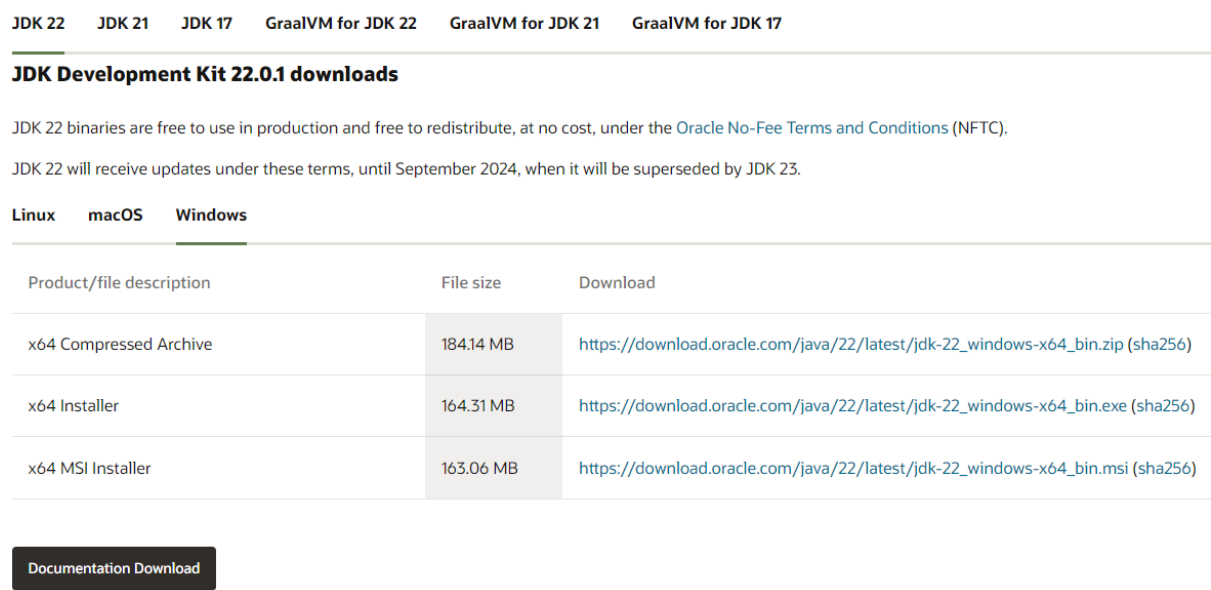


Рисунок 5 – Официальный сайт Oracle

Далее нужно извлечь содержимое архива в папку C:\Program Files\Java.

Следующим шагом стоит добавить переменную среды JAVA_HOME – необходимая переменная для указания пути к установленному JDK. Переменная PATH используется для добавления путей к исполняемым файлам в системную переменную, чтобы командная строка и другие программы могли найти исполняемые файлы JDK без необходимости указывать полный путь каждый раз.

Для добавления переменной среды необходимо перейти в настройки ПК и нажать на «Дополнительные параметры системы». В открывшемся окне «Свойства системы» нужно нажать на кнопку «Переменные среды» (рисунок 6).

Далее нажать на кнопку «Создать». В окне «Новая пользовательская переменная» ввести имя переменной JAVA_HOME, в поле значение – путь до папки с установленным JDK. Перезапустить компьютер.

Наконец, проверить успешность установки можно командой «java -version» – в консоли появится сообщение с информацией о версии Java, установленной на компьютере (рисунок 7).

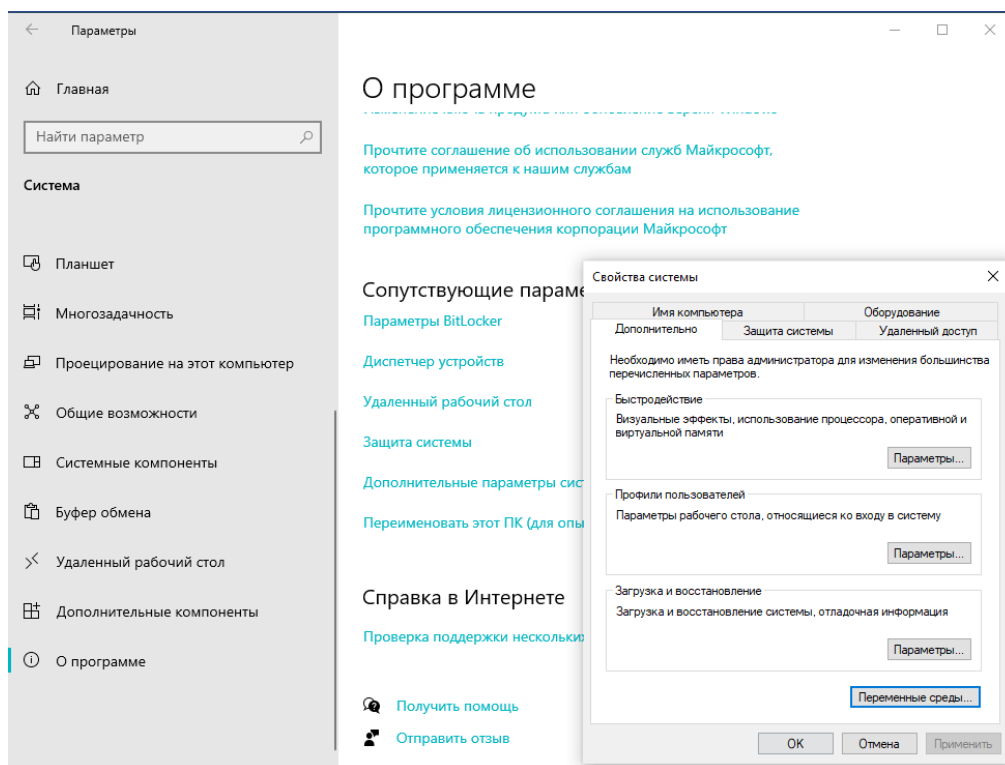


Рисунок 6 – Настройки компьютера

```
C:\Users\verka>java -version
java version "22.0.1" 2024-04-16
Java(TM) SE Runtime Environment (build 22.0.1+8-16)
Java HotSpot(TM) 64-Bit Server VM (build 22.0.1+8-16, mixed mode, sharing)
```

Рисунок 7 – Проверка установки Java

После завершения установки JDK необходимо установить и настроить среду разработки.

4.1.2 Установка IntelliJ IDEA

Для установки IntelliJ IDEA необходимо перейти на официальный сайт JetBrains и скачать бесплатную версию IntelliJ IDEA Community Edition (рисунок 8).

После окончания загрузки нужно запустить установочный файл, выбрать необходимые компоненты и папку установки, дождаться завершения установки программы и переходить к созданию проекта (рисунок 9).

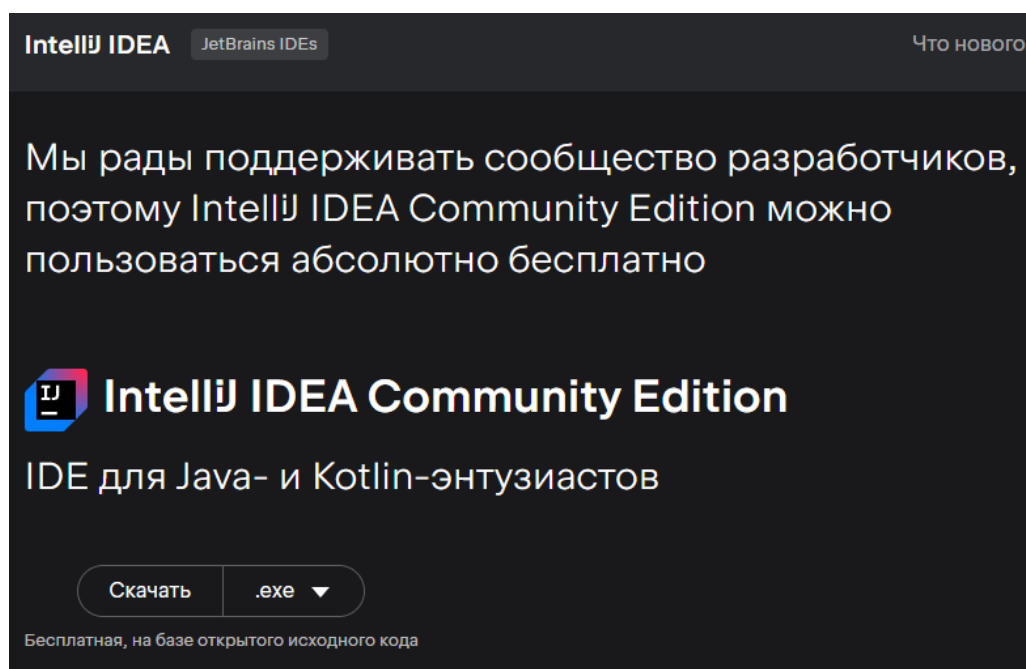


Рисунок 8 – Официальный сайт JetBrains

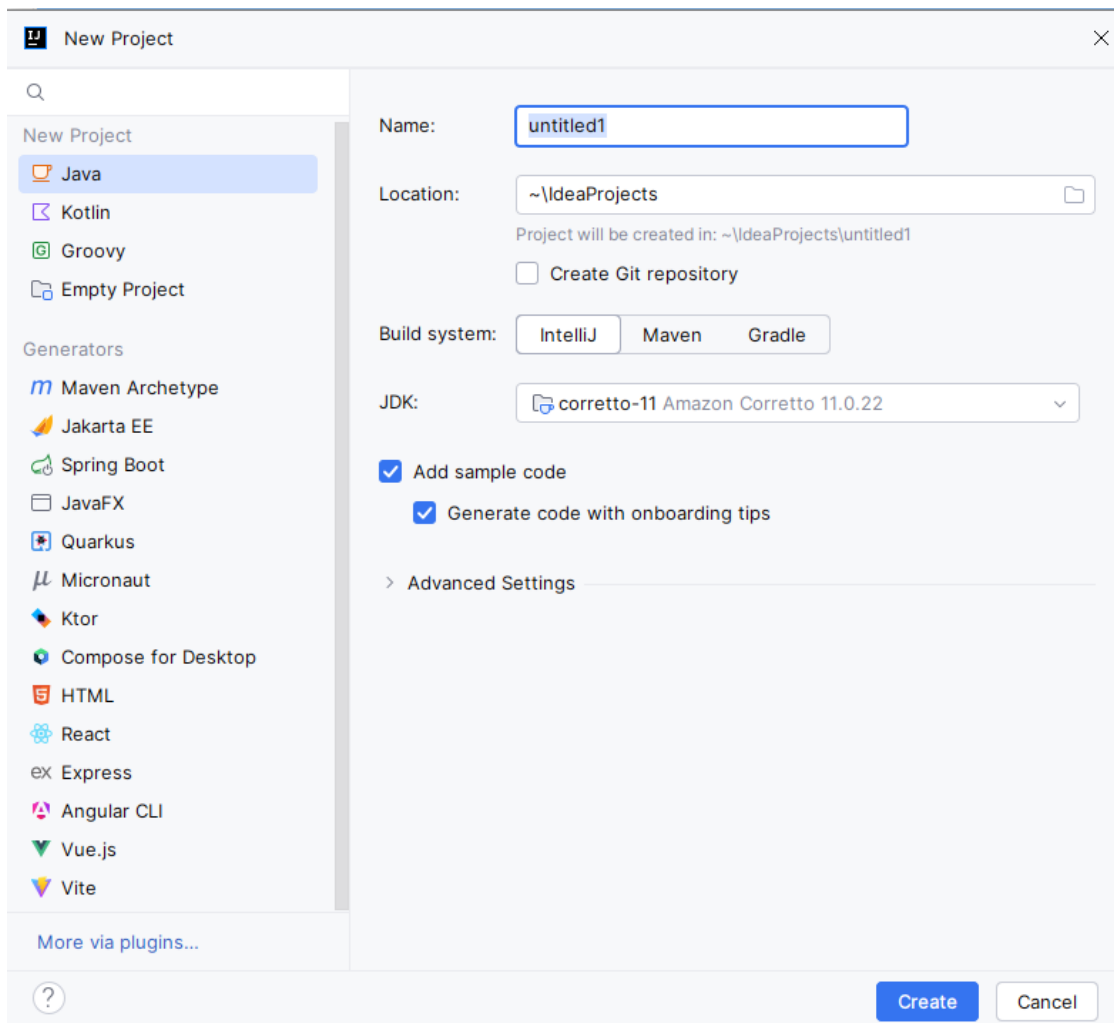


Рисунок 9 – Создание нового проекта в IntelliJ IDEA

4.1.3 Подключение Selenide

Для добавления TestNG в проект необходимо перейти в файл pom.xml и прописать зависимости:

```
<dependency>
  <groupId>com.codeborne</groupId>
  <artifactId>selenide</artifactId>
  <version>7.3.1</version>
  <scope>test</scope>
</dependency>
```

Далее необходимо прописать импорты в разрабатываемый Java-файл:

```
import static com.codeborne.selenide.Selenide.*;
import static com.codeborne.selenide.Condition.*;
```

4.1.4 Подключение TestNG

TestNG подключается аналогично Selenide – необходимо прописать зависимости в файл pom.xml:

```
<dependency>
    <groupId>org.testng</groupId>
    <artifactId>testng</artifactId>
    <version>7.5</version>
</dependency>
```

Также добавить импорт:

```
import org.testng.annotations.Test;
```

4.2 Разработка автоматизированных тестов

Для написания автотестов был использован фреймворк TestNG. С его помощью были созданы описания тестовых методов, включение/выключение тестов и их авторство.

Например, описания тестовых методов могут выглядеть следующим образом:

@Test(description = "Приказы / Групповой приказ об отчислении / Роль Оператор ПОО").

Атрибут enabled отвечает за то, будет ли проигнорирован тест или нет. Пример отключенного метода:

@Test(enabled = false).

В рамках разработки автотестов были проигнорированы 3 метода ввиду их неактуальности в АИС.

Атрибут @Owner используется для указания авторства:

@Owner(Authors.SKIRDOVA_VM).

Указание авторства может быть полезно для:

- указания тестирующего, написавшего тестовый метод;
- упрощения обновления методов, поиска более опытных разработчиков при возникновении проблем и их отладке.

Также TestNG позволяет задавать приоритет атрибутом priority, например:

					ПГУ 1.090301.36.001 ПЗ	Лист
						40
Изм.	Лист	№ докум.	Подпись	Дата		

@Test(priority = 5).

Таким образом можно выставить любое целое число. Тесты будут выполняться по указанному порядку.

Данный тест (OrderTest) наследуется от класса BaseSelenideTest:

@Epic("Приказы")

public class OrderTest extends BaseSelenideTest {

...

}

Полный код автотестов представлен в приложении Б.

В качестве примера взят автоматизированный тест проверки создания группового приказа о переводе из группы в группу.

@Test(description = "Приказы / Групповой приказ о переводе из группы в группу/ Роль Оператор ПОО") – это аннотация TestNG, которая помечает метод как тестовый. Аннотация содержит описание теста, указывающее на его назначение и роль в системе.

@Owner(Authors.SKIRDOVA_VM) – аннотация TestNG, которая определяет владельца теста, указывая на ответственного за его разработку.

DateTimeFormatter dateTimeFormatter =
DateTimeFormatter.ofPattern("ddMMyyyy") – создание объекта DateTimeFormatter, который будет использоваться для форматирования даты в нужный формат.

LocalDate today = LocalDate.now() – получение текущей даты в виде объекта LocalDate.

Map<String, String> infoPart = new HashMap<>() – создание объекта HashMap, который будет использоваться для хранения информации о приказе.

infoPart.put(«orderNumber», «Автотест» +
Rand.getRandomRuWord(5).toLowerCase()) – генерация уникального номера приказа и добавление в infoPart.

infoPart.put(«orderFrom», today.format(dateTimeFormatter)) – формирование текущей даты в соответствии с установленным форматом и добавление в infoPart.

Map<String, String> groupListPart = new HashMap<>() – создание объекта HashMap, который будет использоваться для хранения информации о списке групп.

groupListPart.put(«transferFrom»,
today.minusDays(1).format(dateTimeFormatter)) – формирование даты, предшествующей текущей на один день, и добавление в groupListPart.

groupListPart.put(«fromGroupSelect", "19ГД-146") – исходная группа.

groupListPart.put(«inGroupSelect", "23ПЖ") – целевая группа для перевода.

PageSteps.auth(Users.getUserByName(Users.adm1)) – авторизация пользователем с ролью «Администратор приложения».

createOrderPage.checkInitElementsHeaderFooter() – проверка инициализации элементов хедэра и футера страницы.

NavBar().openCreateOrder() – открытие страницы создания приказа через навигационную панель.

createOrderPage.checkInitElements() – проверка инициализации элементов страницы создания приказа.

createOrderPage.selectOrderTypeTransfer() – выбор типа приказа о переводе.

transferOrderPart.checkInitElements() – проверка инициализации элементов страницы создания приказа о переводе.

transferOrderPart.getInformationPart().fillFields(infoPart) – заполнение информации о приказе.

transferOrderPart.getGroupListPart().fillFields(groupListPart) – заполнение списка групп.

createOrderPage.clickCreateOrderBtn().checkSuccessCreate() – нажатие на кнопку создания приказа и проверка успешного создания.

OrderTransferGroupPage transferGroupPage = new OrderTransferGroupPage() – инициализация страницы перевода группы.

transferGroupPage.checkOrderStatus("проект приказа") – проверка приказа на статус «проект приказа».

transferGroupPage.clickGroupChangeButton() – нажатие на кнопку изменения группы.

transferGroupPage.getTransferGroupPart().checkInitElements() – проверка инициализации элементов страницы перевода группы.

transferGroupPage.getTransferGroupPart().changeTransferFrom() – изменение даты перевода.

transferGroupPage.getTransferGroupPart().deleteStudent(1) – удаление одного студента.

transferGroupPage.getTransferGroupPart().hideTable() – скрытие таблицы.

transferGroupPage.getTransferGroupPart().clickSaveButton() – нажатие на кнопку сохранения.

transferGroupPage.clickActionButton().clickDeleteOrder() – нажатие на кнопку «Действие» и кнопку удаления приказа.

transferGroupPage.getModalWindow().clickDelete() – подтверждение удаления приказа.

transferGroupPage.checkSuccessDelete() – проверка успешного удаления приказа.

PageSteps.logout() – выход из системы.

5 Результаты работы автоматизированного тестирования

По завершении разработки программы необходимо провести автоматизированное тестирование и проанализировать полученные результаты. В этой главе будет рассмотрен процесс запуска автотестов и анализа результатов тестирования.

5.1 Запуск тестирования

В IntelliJ IDEA есть несколько способов запуска автотестов с использованием TestNG.

Для запуска целого цикла тестирования необходимо нажать на кнопку «Run» рядом с классом «OrderTest» (рисунок 10). Тогда тесты будут выполняться один за другим в порядке их приоритета.

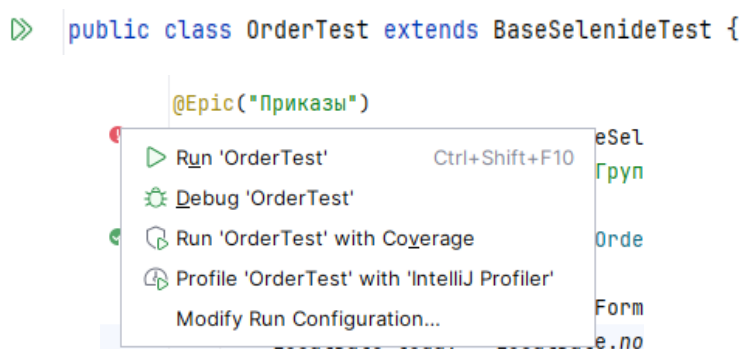


Рисунок 10 – Запуск полного тестирования

Также можно запускать отдельные автотесты, нажатием на кнопку «Run» рядом с соответствующим методом (рисунок 11).

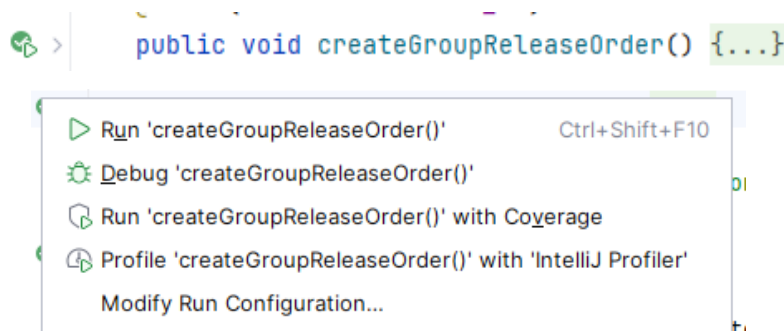


Рисунок 11 – Запуск одного теста

Имеется и возможность запуска последних тестов в блоке «Recent Configurations» (рисунок 12). Это позволяет экономить время на поиске нужных методов, если их запуск был выполнен недавно.

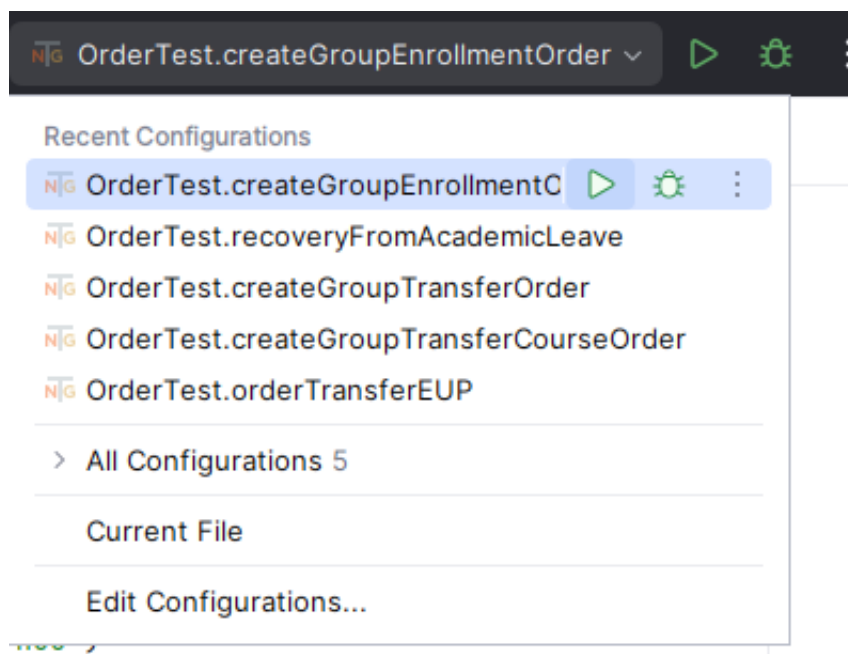


Рисунок 12 – Запуск последних методов

После запуска целого цикла или отдельных тестов можно перезапустить выполнение целиком (рисунок 13) или только проваленных тестов (рисунок 14).

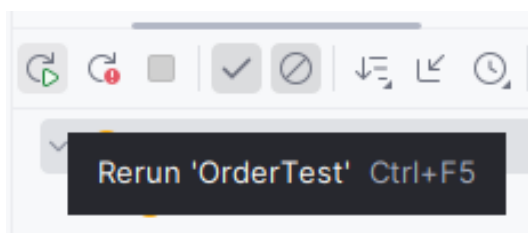


Рисунок 13 – Перезапуск всей группы тестов

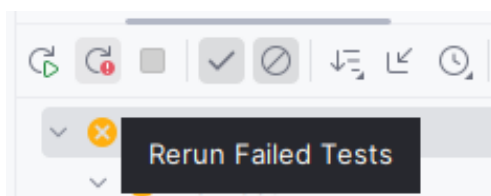


Рисунок 14 – Перезапуск проваленных тестов

5.2 Получение результатов тестирования

В ходе разработки было составлено 19 автоматизированных тестов групповых и индивидуальных приказов. 3 автотеста являются неактуальными на сегодняшний день, поэтому были отключены.

Результаты выполнения автотестов показаны в таблице 2.

Таблица 2 – Отчет о прохождении тестирования

Метод	Описание	Результат
createGroupAcademicLeaveOrder()	Групповой о предоставлении АО	Успех
createGroupEnrollmentOrder()	Групповой о зачислении	Сбой
createGroupExpulsionOrder()	Групповой об отчислении	Успех
createGroupRecoveryOrder()	Групповой о восстановлении	Сбой
createGroupReleaseOrder()	Групповой о выпуске	Успех
createGroupTransferCourseOrder()	Групповой о переводе с курса на курс	Сбой
createGroupTransferToBupOrder()	Групповой о переводе на БУП	Успех
createGroupTransferToEupOrder()	Групповой о переводе на ЭУП	Успех
createIndividualExpulsionOrder()	Индивидуал. об отчислении	Успех
orderAcademicLeave()	Индивидуал. о предоставлении АО	Успех
orderRealise()	Индивидуал. о выпуске	Успех
orderRecovery()	Индивидуал. о восстановлении	Сбой
orderTransfer()	Индивидуал. о переводе из группы в группу	Сбой
orderTransferBUP()	Индивидуал. о переводе на БУП	Отключен
orderTransferEUP()	Индивидуал. о переводе на ЭУП	Успех

Метод	Описание	Результат
orderBudgetFinance()	Индивидуал. приказ о смене финансир-я	Отключен
orderEnroll()	Индивидуал. о зачислении	Отключен
recoveryFromAcademicLeave()	Автомат. о восстановлении из АО	Сбой
createGroupTransferOrder()	Групповой о переводе из группы в группу	Сбой

Таким образом, было запущено и выполнено 16 тестов (рисунок 15).

Default Suite

Total tests run: 16, Passes: 9, Failures: 7, Skips: 0

Рисунок 15 – Количество успешных и проваленных тестов

Результаты тестирования продемонстрированы на рисунке 16.

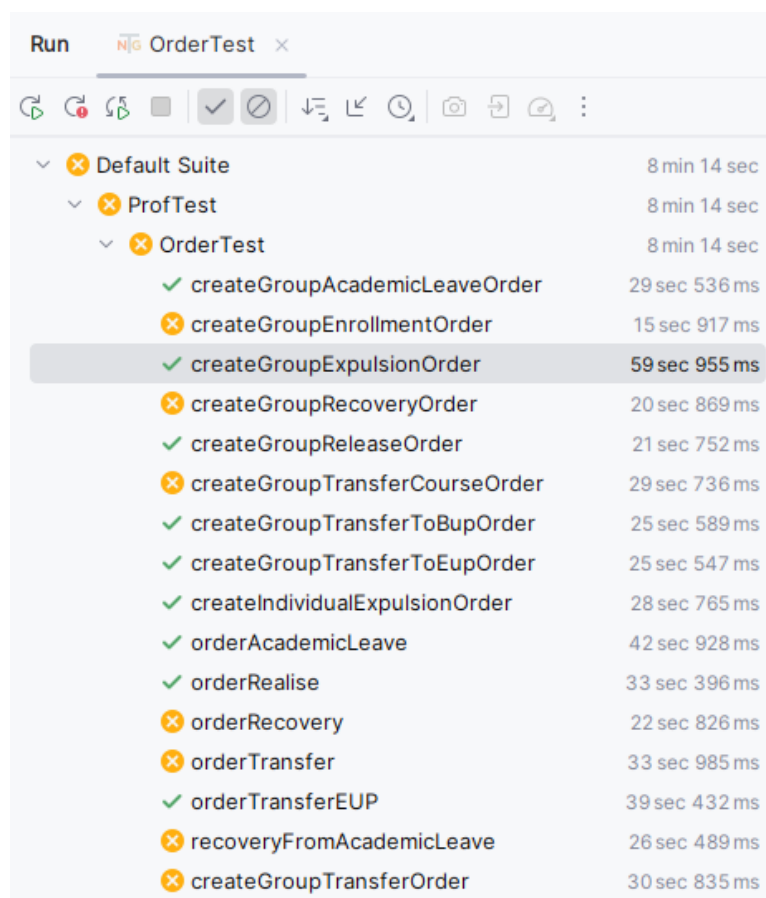


Рисунок 16 – Результаты прохождения тестирования

Исходя из результатов тестирования, видно, что практически половина тестов были выполнены с результатом Assertion Failure (желтый значок). Например, метод `createGroupTransferOrder()` не был завершен корректно из-за таймаута по причине, что не была найдена требуемая группа (рисунок 17).

java.lang.AssertionError: Create breakpoint : Элемент `//*[contains(@class, 'react-select__option') and text()='19ГД-147']` не найден за 15 секунд

Рисунок 17 – Сообщение об ошибке

Большинство тестов не были завершены из-за отсутствия требуемых тестовых данных. Например, после добавления в тест правильных значений перезапуск теста может быть выполнен успешно (Test Passed).

Так, после смены группы «19ГД-147» на, например, существующую группу «23ПЖ» тест `createGroupTransferOrder()` был выполнен успешно (рисунок 18).

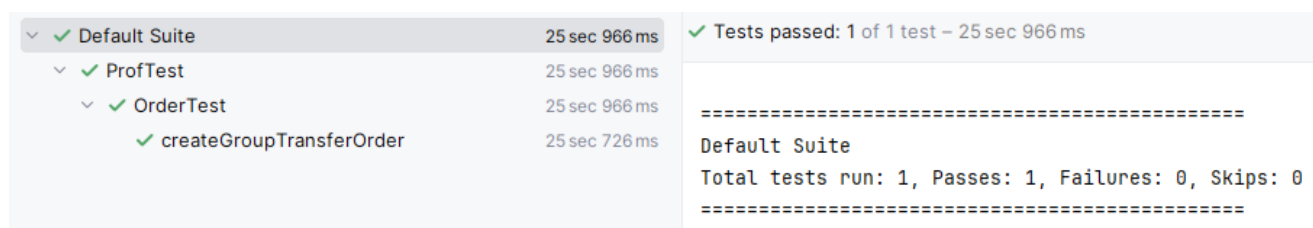


Рисунок 18 – Тест после перезапуска

Скриншоты результатов тестирования сохраняются по адресу `C:/Users/verka/IdeaProjects/ProfTest/build/reports/tests` (рисунок 19).

Например, тест `createGroupEnrollmentOrder()` не был выполнен из-за серверной ошибки на странице приказа. Скриншот был сохранен под именем «1715364756087.2.png» (рисунок 20).

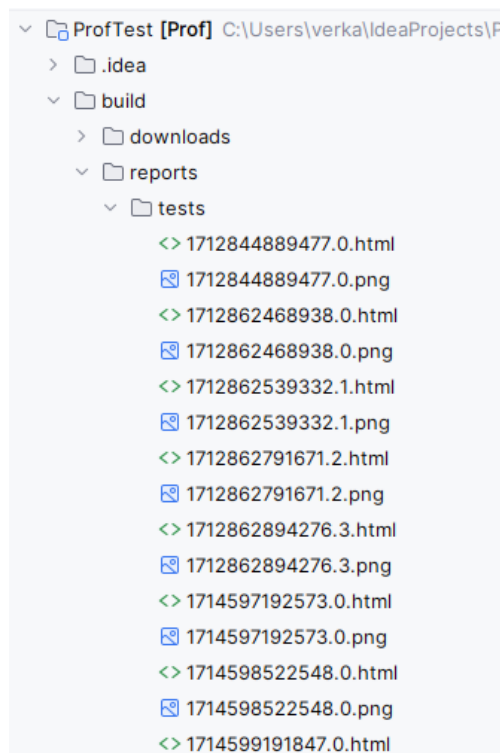


Рисунок 19 – Скриншоты прохождения тестирования

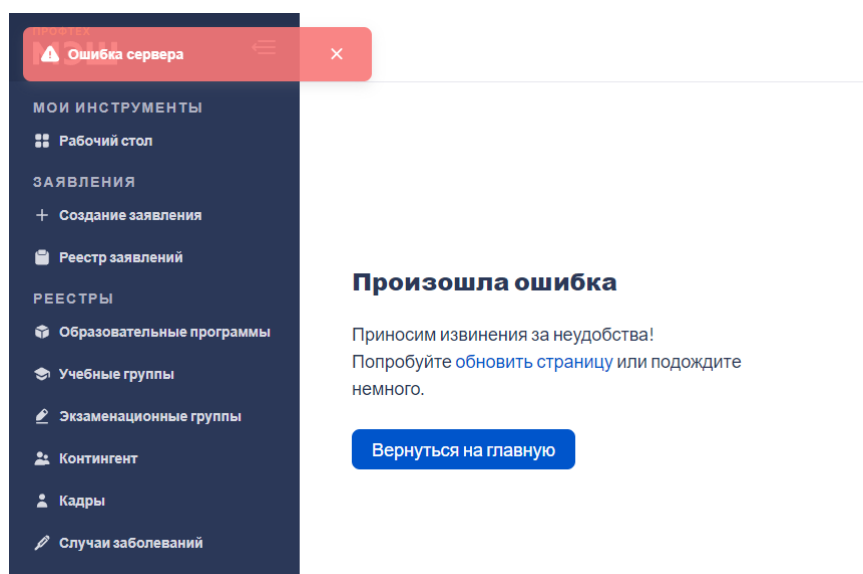


Рисунок 20 – Скриншот ошибки

5.3 Генерация отчетов Allure

Помимо просмотра результатов тестирования непосредственно из среды разработки есть возможность генерации отчетов Allure.

Allure представляет собой открытый фреймворк для генерации отчетов о результатах автоматизированного тестирования, позволяющий создавать детализированные и наглядные отчеты. Этот инструмент обеспечивает визуализацию данных тестирования в удобном и интерактивном интерфейсе. Это значительно облегчает анализ и выявление ошибок.

Среди языков программирования, которые поддерживает Allure, — Java, Python, JavaScript/TypeScript, Ruby, PHP, C# и Kotlin. Например, для Java Allure работает с фреймворками JUnit, TestNG и Spock. В Python поддерживаются Pytest, Unittest и Nose. В экосистеме JavaScript/TypeScript Allure интегрируется с Mocha и Jest. Ruby поддерживается через RSpec и Cucumber, PHP — через Codeception и PHPUnit, а C# — через NUnit и SpecFlow.

Allure создает детализированные и визуально привлекательные отчеты, охватывающие различные аспекты тестирования, такие как шаги выполнения, скриншоты и логи, делая анализ результатов тестирования удобным и информативным.

Перед запуском генерации отчетов необходимо выполнить команду `allure:clean`, чтобы удалить старые логи тестов, если такие имеются (рисунок 21). Далее запускается тест или группа тестов. После окончания тестирования просмотреть отчеты можно, используя команду `allure:serve`, которая запускает временный сервер и открывает отчет в браузере. После выполнения этой команды откроется браузер с сгенерированным отчетом Allure, где можно будет просмотреть все результаты тестирования в удобном и наглядном виде.

Allure интерфейс содержит несколько ключевых разделов (рисунок 22).

Overview (Обзор) – главная страница отчета, где представлены сводные данные о выполненных тестах, количество успешных, неудачных и пропущенных тестов.

Suites (Наборы тестов) – раздел, отображающий структуру тестов в виде иерархии, позволяя детально просматривать результаты каждого набора и отдельных тестов.

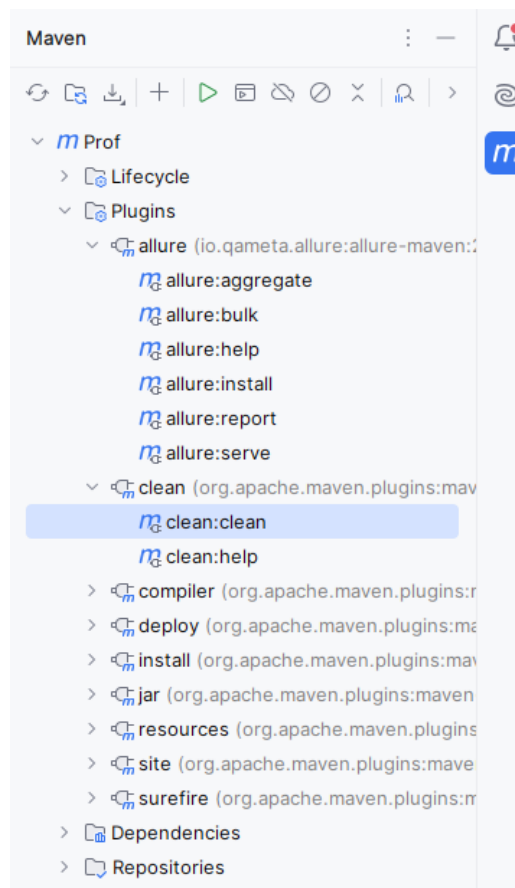


Рисунок 21 – Запуск отчета Allure

Graphs (Графики) – визуализация данных тестирования в виде графиков и диаграмм, что помогает быстро оценить динамику выполнения тестов.

Timeline (Таймлайн) – хронологическая шкала, показывающая последовательность выполнения тестов и время, затраченное на каждый тест.

Behaviors (Поведения) – раздел, отображающий тесты в контексте поведения системы, группируя их по функциональным характеристикам.

Packages (Пакеты) – организация тестов по пакетам, что полезно для проектов с модульной структурой.

Пример прохождения набора тестов «Приказы» продемонстрирован на рисунке 23.

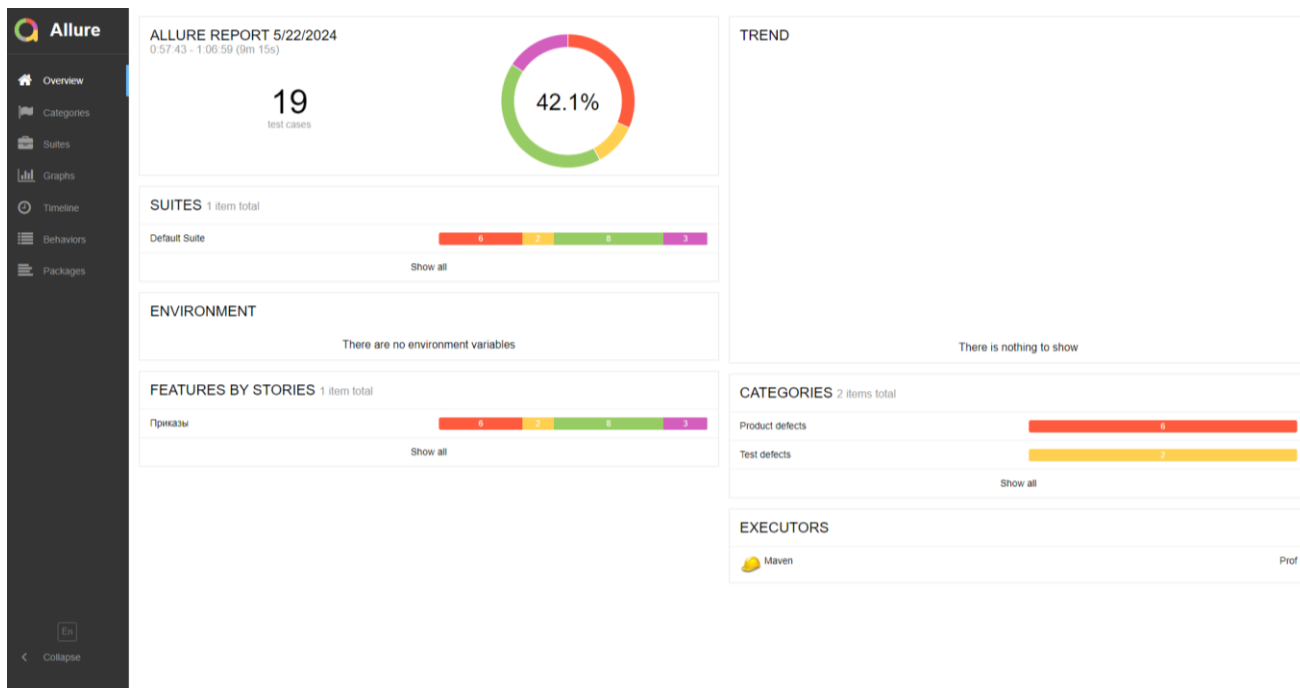


Рисунок 22 – Интерфейс Allure

Suites					
order	name	duration	status	Filter by status: 6 2 8 0 3	
▼	Default Suite			6	2 8 0 3
▼	ProfTest			6	2 8 0 3
▼	orders.OrderTest			6	2 8 0 3
✖	#7 Приказы/ групповой приказ о восстановлении / Роль Оператор ПОО	22s 406ms			
✖	#18 Приказы/ Автомат. приказ о восстановлении из академ.отпуска/ Роль Оператор ПОО	30s 370ms			
✔	#8 Приказы/ Групповой приказ о выпуске/ Роль Оператор ПОО	25s 703ms			
✖	#5 Приказы/ Групповой приказ о зачислении/ Роль Оператор ПОО	52s 348ms			
✔	#10 Приказы/ Групповой приказ о переводе на БУП/ Роль Оператор ПОО	31s 142ms			
✔	#11 Приказы/ Групповой приказ о переводе на ЭУП/ Роль Оператор ПОО	30s 395ms			
✖	#9 Приказы/ Групповой приказ о переводе с курса на курс/ Роль Оператор ПОО	31s 560ms			
✔	#4 Приказы/ Групповой приказ о предоставлении академического отпуска/ Роль Оператор ПОО	28s 654ms			
✔	#6 Приказы/ Групповой приказ об отчислении/ Роль Оператор ПОО	48s 729ms			
✔	#12 Приказы/ Индивидуальный приказ об отчислении/ Роль Оператор ПОО	34s 684ms			
✔	#19 Приказы/Групповой приказ о переводе из группы в группу/ Роль Оператор ПОО	29s 025ms			
✖	#15 Приказы/Индивидуальный приказ о восстановлении/Роль Оператор ПОО	9s 622ms			
?	#2 Приказы/Индивидуальный приказ зачисления/Роль Оператор ПОО	0s			
!	#14 Приказы/Индивидуальный приказ о выпуске/Роль Оператор ПОО	17s 157ms			
!	#16 Приказы/Индивидуальный приказ о переводе из группы в группу/Роль Оператор ПОО	16s 644ms			
?	#3 Приказы/Индивидуальный приказ о переводе на БУП/Роль Оператор ПОО	0s			
✔	#17 Приказы/Индивидуальный приказ о переводе на ЭУП/Роль Оператор ПОО	1m 05s			
✖	#13 Приказы/Индивидуальный приказ о предоставлении академического/Роль Оператор ПОО	1m 01s			
?	#1 Приказы/Индивидуальный приказ о смене финансирования/Роль Оператор ПОО	2ms			

Рисунок 23 – Отчет Allure

Возможные результаты тестов в отчете Allure:

- Failed. Тест завершился с ошибкой, одно или несколько проверяемых условий не выполнены;
- Passed. Тест выполнен успешно, все проверяемые условия выполнены;
- Broken. Тест прерван из-за ошибки в самом тесте или в тестовой среде, не связанной с проверяемым функционалом;
- Skipped. Тест был пропущен и не выполнялся, возможно, из-за настроек условий выполнения или зависимостей от других тестов;
- Unknown. Тест неизвестен. Это может быть вызвано ошибкой или неправильным использованием адаптера Allure.

В ходе выполнения наборов тестов «Приказы» были получены результаты в статусах, продемонстрированных на рисунке 24.

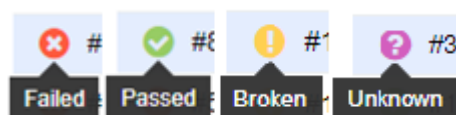


Рисунок 24 – Виды статусов

Пример успешно выполненного теста отображен на рисунке 25, а проваленного – на рисунке 26.

Passed**Приказы/Групповой приказ о переводе из группы в группу/
Роль Оператор ПОО**

Overview History Retries

Severity: normal

Duration: 29s 025ms

Owner

Скирдова Вероника Максимовна

Execution

> Set up

✓ Test body

> Авторизация - autotest-adm / Qwerty123 1 parameter, 12 sub-steps	7s 938ms
> Проверка header и footer элементов главной страницы 19 sub-steps	6s 383ms
> Переход из меню в раздел 'Создание приказа' 5 sub-steps	472ms
> Проверка элементов страницы "Создание приказа" 10 sub-steps	487ms
> Выбор типа создаваемого приказа 10 sub-steps	329ms
> Проверка загрузки элементов страницы "Создание приказа" при выборе приказа о переводе из группы в группу	48 sub-steps 3s 626ms
> Заполнение поля ввода даты «от:» значением «22052024» 1 parameter	512ms
> Заполнение поля ввода даты «Перевести с» значением «21052024» 1 parameter	389ms
> Заполнение элемента с выпадающим списком «Из группы:» текстом "19ГД-146" 1 parameter, 13 sub-steps	732ms
> Заполнение элемента с выпадающим списком «Из группы:» текстом "23ПЖ" 1 parameter, 13 sub-steps	658ms
✓ Нажатие на кнопку «Создать приказ»	156ms
> Проверка уведомление о создании приказа 8 sub-steps	1s 336ms
> Проверка статуса приказа 1 parameter	29ms
✓ Нажатие на кнопку со значком карандаша группы в разделе "Список обучающихся на перевод из группы в группу"	163ms
> Проверка элементов модального окна "Редактирование сведений о переводе из группы в группу" 31 sub-steps	1s 740ms
> Изменение значения поля "Перевести с" 1 sub-step	442ms
> Исключение студента из списка 1 parameter	261ms
✓ Свернуть таблицу со студентами	93ms
> Нажатие на кнопку "Сохранить" 7 sub-steps	794ms

Рисунок 25 – Пример успешного выполнения теста

Изм.	Лист	№ докум.	Подпись	Дата

ПГУ 1.090301.36.001 ПЗ

Лист

54

orders.OrderTest.createGroupTransferCourseOrder

Failed

Приказы/ Групповой приказ о переводе с курса на курс/ Роль Оператор ПОО

Overview

History

Retries

Элемент `//*[contains(@class, 'react-select__option')] and text()='21ГД-2з']` не найден за 15 секунд

Categories: Product defects

Severity: normal

Duration: 31s 560ms

Owner

Скирдова Вероника Максимовна

Execution

> Set up

> Test body

> Авторизация - autotest-adm / Qwerty123 1 parameter, 12 sub-steps4s 417ms

> Проверка header и footer элементов главной страницы 19 sub-steps6s 298ms

> Переход из меню в раздел 'Создание приказа' 5 sub-steps454ms

> Проверка элементов страницы "Создание приказа" 10 sub-steps499ms

> Выбор типа создаваемого приказа 10 sub-steps336ms

> Проверка загрузки элементов страницы "Создание приказа" при выборе приказа о переводе с курса на курс 47 sub-steps2s 656ms

> Заполнение поля ввода даты «от:» значением «22052024» 1 parameter357ms

> Заполнение поля ввода даты «Перевести с» значением «21052021» 1 parameter281ms

> Заполнение элемента с выпадающим списком «Из группы:» текстом "21ГД-2з" 1 parameter, 11 sub-steps15s 933ms

text21ГД-2з

> Ожидание элемента в DOM `//div[text()='Сведения о переводе']/../..//label[text()='Перевести из группы БУП']/../div[contains(@class, 'react-select__control')]` 1 parameter9ms

> Скролл к элементу `//div[text()='Сведения о переводе']/../..//label[text()='Перевести из группы БУП']/../div[contains(@class, 'react-select__control')]` 1 parameter299ms

> Ожидание элемента `//div[text()='Сведения о переводе']/../..//label[text()='Перевести из группы БУП']/../div[contains(@class, 'react-select__control')]` 1 parameter21ms

> Выбор опции элемента с выпадающим списком «Из группы:» - «21ГД-2з» 1 parameter, 7 sub-steps15s 604ms

Рисунок 26 – Пример проваленного теста

6 Охрана труда

В данном разделе будут рассмотрены вопросы, касающиеся охраны труда пользователей, долгое время работающих за компьютером, работников, чья профессия связана с длительным взаимодействием с ПК, потенциальные опасности и меры, предпринимаемые для исключения негативного воздействия на организм человека.

6.1 Потенциальные опасности при работе с ЭВМ

С каждым годом сферы общественной жизни, использующие персональные электронно-вычислительные машины (ПЭВМ), расширяются. Образование, медицина, банковские структуры уже невозможно представить без использования компьютеров. Теперь, когда весь мир пользуется великим достижением человечества, появилась масса потенциальных опасностей для физического и психологического здоровья людей.

Большинство работников сферы информационных технологий так или иначе сталкиваются с проблемами со здоровьем. Сидячий образ жизни и недостаток физической активности может привести к множеству проблем. Они могут быть выражены в виде головной боли, сухости и усталости глаз из-за слишком близкого расположения монитора, постоянного фокусирования взгляда на одну точку, недостаточного увлажнения глаз. Боли в шее, плечах могут быть вызваны постоянным напряжением мышц из-за продолжительного нахождения в одной позе. Следовательно, из-за проблем с шеей происходит ухудшение кровообращения мозга и развитие проблем сердечно-сосудистой системы. Постоянный набор текста, неправильное расположение клавиатуры и компьютерной мыши может привести к болям в кистях и развитию туннельного синдрома. Страдает так же плечевой пояс, возникает разница в положении плеч. Из-за длительного несоблюдения рекомендаций работы с компьютером страдает и спина. Происходит искривление позвоночника, его ослабление, образование кифоза. Неправильное расположение ног приводит к растяжению коленного

					ПГУ 1.090301.36.001 ПЗ	Лист
						56
Изм.	Лист	№ докум.	Подпись	Дата		

сустава, болям в ногах. Частые и регулярные разминки помогут предотвратить большинство проблем со здоровьем.

Постоянный стресс, частые перенапряжения также сказываются на психическом здоровье человека, выражаясь в нервных срывах, быстром и хроническом переутомлении, раздражительности, ослаблении памяти и нарушении сна.

Для обеспечения комфорта и здоровья сотрудника или простого пользователя ПК необходимо создать комфортные условия труда, подобрать эргономичную и качественную мебель, технику, продумать освещение помещения и рабочего пространства, положительно относиться к регулярным упражнениям и проводить их.

Соблюдение всех мер безопасности и предосторожности, забота о собственном здоровье и здоровье сотрудников минимизирует риски ухудшения работоспособности организма, предотвратит различные профессиональные травмы и заболевания, повысит производительность труда, понизит шанс профессионального выгорания и, наконец, обеспечит работника комфортом и хорошими условиями труда.

Далее будут рассмотрены меры, направленные на предотвращение проблем со здоровьем пользователей ПК и эргономику рабочего места.

6.2 Эргономика рабочего места

Эргономика рабочего места – сведения о безопасности и здоровье при работе с ПЭВМ. Правильная организация рабочего пространства помогает снизить физическое напряжение, предотвратить профессиональные заболевания и повысить продуктивность.

Человек большую часть рабочего дня проводит за столом, соответственно, необходимо выбрать подходящий рабочий стол. Высота стола должна быть такой, чтобы руки пользователя были на уровне с запястьями. Поверхность стола должна быть достаточно большой для расположения всех необходимых устройств, таких

как монитор, клавиатура, мышь и других аксессуаров, не образуя беспорядка на столе.

Клавиатура должна быть расположена на удобной высоте, чтобы пользователю не приходилось поднимать или опускать руки слишком высоко. Рекомендуется использовать эргономичную клавиатуру, которая снижает нагрузку на запястья. Мышь следует размещать в пределах досягаемости от клавиатуры, чтобы не пришлось тянуться к ней. Также существуют множество современных вариантов эргономичных компьютерных мышей, которые помогают уменьшить напряжение в запястьях и руках и повысить комфортность.

На второе место по значимости встает выбор эргономичного кресла. В первую очередь, оно должно подходить по размеру к выбранному столу. Удобным вариантом будут кресла с регулируемой высотой и подлокотниками. Существуют множество современных вариантов компьютерных кресел, но подойти может и обычный стул. Главное – правильное положение. Ноги должны упираться на пол, колени быть согнутыми под углом в 90 градусов. Нельзя также закидывать ноги друг на друга. Спина должна быть выпрямлена, но не выгнута. Спинка кресла должна быть удобной, но работник не должен «утопать» в ней. Правильно выбранное кресло поможет предотвратить искривление осанки, развитие болезней шеи и спины.

Далее необходимо правильно подобрать и расположить монитор. Его следует располагать прямо перед пользователем на расстоянии вытянутой руки (примерно 50 см). Чем больше размер экрана, тем дальше он должен быть от глаз. Верхняя часть экрана должна быть на уровне глаз или чуть ниже, чтобы предотвратить напряжение в шее. Угол наклона монитора должен быть таким, чтобы на экране не было бликов и не было необходимости наклонять голову вверх или вниз. Регулирование яркости и контрастности монитора помогает уменьшить нагрузку на глаза. Излучаемый экраном синий свет плохо сказывается на здоровье человека, его зрению, поэтому была изобретена технология Low Blue Light (у разных производителей разные названия). Она помогает снизить излучение синего цвета, изображение как бы становится более теплым.

					ПГУ 1.090301.36.001 ПЗ	Лист
						58
Изм.	Лист	№ докум.	Подпись	Дата		

Существенный минус такой технологии заключается в том, что он искажает цвета на экране. Это может быть критично для тех, кто работает с обработкой фотографий и видео. Также дисплей не должен мерцать. Широтно-импульсная модуляция (ШИМ) вызывает головные боли при работе с такими экранами.

Рабочее место должно быть хорошо освещено, но без прямого попадания света на монитор. Хорошо, если в помещении будет как искусственный свет, так и естественный. При нехватке света на рабочем столе отличным решением будет использование настольной лампы. Существуют удобные варианты ламп, которые крепятся к краю стола и не занимают место на рабочем пространстве.

Подставки для ног могут помочь поддерживать правильное положение ног и улучшить кровообращение. Использование подставок под запястья для клавиатуры и мыши снижает нагрузку на запястья и предотвращает развитие туннельного синдрома.

Самое важное – регулярные перерывы и физические упражнения. Они способствуют поддержанию здоровья при работе с компьютером. Организму обязательно нужно менять свое положение в течении рабочего дня. Рекомендуется каждые 10-15 минут делать разминку глаз, а каждые 30-60 минут делать короткие перерывы для разминки, чтобы снять напряжение с мышц и улучшить кровообращение. Простые упражнения, такие как растяжка, вращение запястий и плеч, помогают предотвратить усталость и болезненные ощущения. Хорошо, если есть возможность встать и походить, чтобы размяться.

Эргономика рабочего места – это не только комфорт и удобство, но профилактика профессиональных заболеваний и травм. Правильно организованное рабочее пространство способствует поддержанию здоровья, повышает продуктивность и улучшает общее качество работы.

Заключение

В ходе выполнения работы была проанализирована предметная область тестирования и автоматизации тестирования, рассмотрены различные его виды и возможности. Был проведен анализ и выбор инструментов разработки, таких как язык программирования, фреймворки автотестирования и среда разработки. Спроектирована система автоматизированного функционального тестирования с учетом выбора инструментов. Разработана и развернута система. Получены и разобраны результаты выполнения тестирования, сгенерирован отчет.

Успешно разработанная система показала свою эффективность, а полученные результаты подтверждают актуальность выбранной темы – автоматизации тестирования, соответственно, могут быть использованы как база для дальнейшего исследования, разработки и улучшения автотестов для web-приложений.

					ПГУ 1.090301.36.001 ПЗ	Лист
						60
Изм.	Лист	№ докум.	Подпись	Дата		

Список использованных источников

1. Bret Pettichord, James Bach, Cem Kaner, Lessons Learned in Software Testing: A Context-Driven Approach, 2002. – С. 45-50
2. Дж. Уиттакер, Дж. Арбон, Дж. Каролло Как тестируют в Google. — СПб.: Питер, 2014. – С. 109-114.
3. Куликов, С. С Тестирование программного обеспечения. Базовый курс : практ. пособие. / С. С. Куликов. — Минск: Четыре четверти, 2015. — 294 с.
4. Савин Р. Тестирование Дот Ком, или Пособие по жестокому обращению с багами в интернет-стартапах. — М.: Дело, 2007. — С. 139-169.
5. В. Н. Пероцкая, Д. А. Градусов Основы тестирования программного обеспечения: учеб. пособие: Владим. гос. ун-т им. А. Г. и Н. Г. Столетовых. — Владимир: Изд-во ВлГУ, 2017. — С. 13-15.
6. Ron Patton, Software Testing. – 2000. С. 175.
7. Гленфорд Майерс, Том Баджетт, Кори Сандлер Искусство тестирования программ / Пер. с англ. под ред. Б. А. Позина. – М.: Финансы и статистика, 1982. – С. 54-60.
8. Хориков Владимир Принципы юнит-тестирования. — СПб.: Питер, 2021. – С. 40-62.
9. James A. Whittaker How to break software security: effective techniques for security testing, – 2004. С. 102-117.
10. Элфрид Дастин, Джефф Рэшка, Джон Пол Автоматизированное тестирование программного обеспечения, 2014. с. 330-170.
11. Лопатина К. И., Темникова Е. А. Ручное и автоматизированное тестирование программного обеспечения отличия и взаимозаменяемость //Вестник современных исследований. – 2017. – №. 10-1. – С. 75-77.
12. Караваева О. В., Борисова К. В. Разработка системы автоматизированного тестирования //Вестник науки и образования. – 2018. – №. 13 (49). – С. 45-48.

13. Матвеева Н. А., Герасимов В. В., Игнатъева Д. О. Исследование технологий модульного тестирования на платформе Java //Системні технології. – 2016. – №. 1. – С. 49-55.

14. Тобиас Клейн Дневник охотника за ошибками. Путешествие через джунгли проблем безопасности программного обеспечения. Пер. с англ. Киселев А. Н. – М.: ДМК Пресс, 2013. – 240с.

15. Загидуллин Р. А. ЯЗЫК ПРОГРАММИРОВАНИЯ JAVA //Редакционный совет. – 2017. – Т. 43. – №. 3. – С. 889.

					ПГУ 1.090301.36.001 ПЗ	Лист
						62
Изм.	Лист	№ докум.	Подпись	Дата		

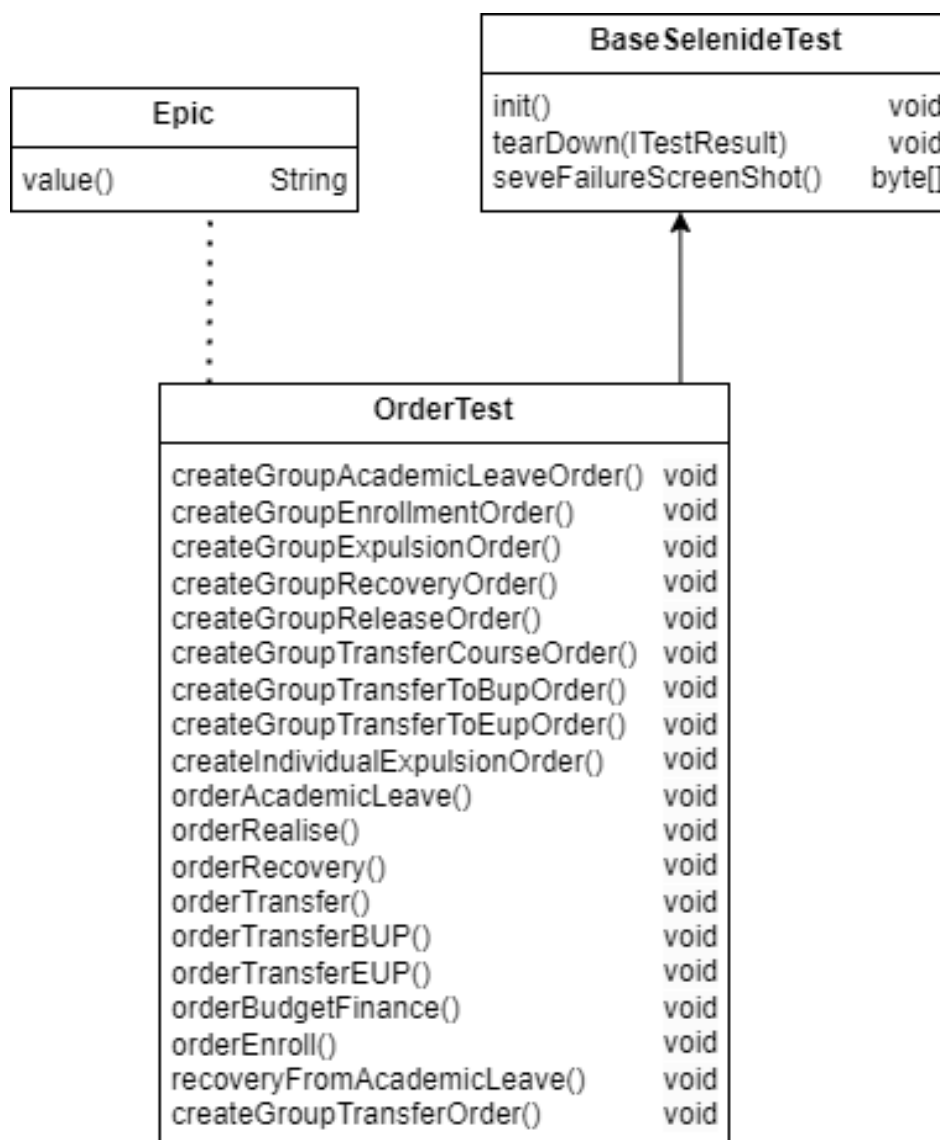
Приложение А

(Обязательное)

Автоматизированное функциональное тестирование создания индивидуальных
и групповых приказов
UML-диаграммы

					ПГУ 1.090301.36.001 ПЗ	Лист
						63
Изм.	Лист	№ докум.	Подпись	Дата		

Приложение А.1 – Диаграмма классов



					ПГУ 1.090301.36.002 Д11			
Изм.	Лист	№ докум.	Подпись	Дата	Аттестованное функциональное тестирование создано индивидуальными и групповыми приказами декана факультета	Лит.	Лист	Листов
Разработ.		Сидорова В.М.						
Провер.		Токарев А.Н.						
И. центр.		Былков А.С.				ФВТ, каф. ВТ дл. 20ВВГ2		

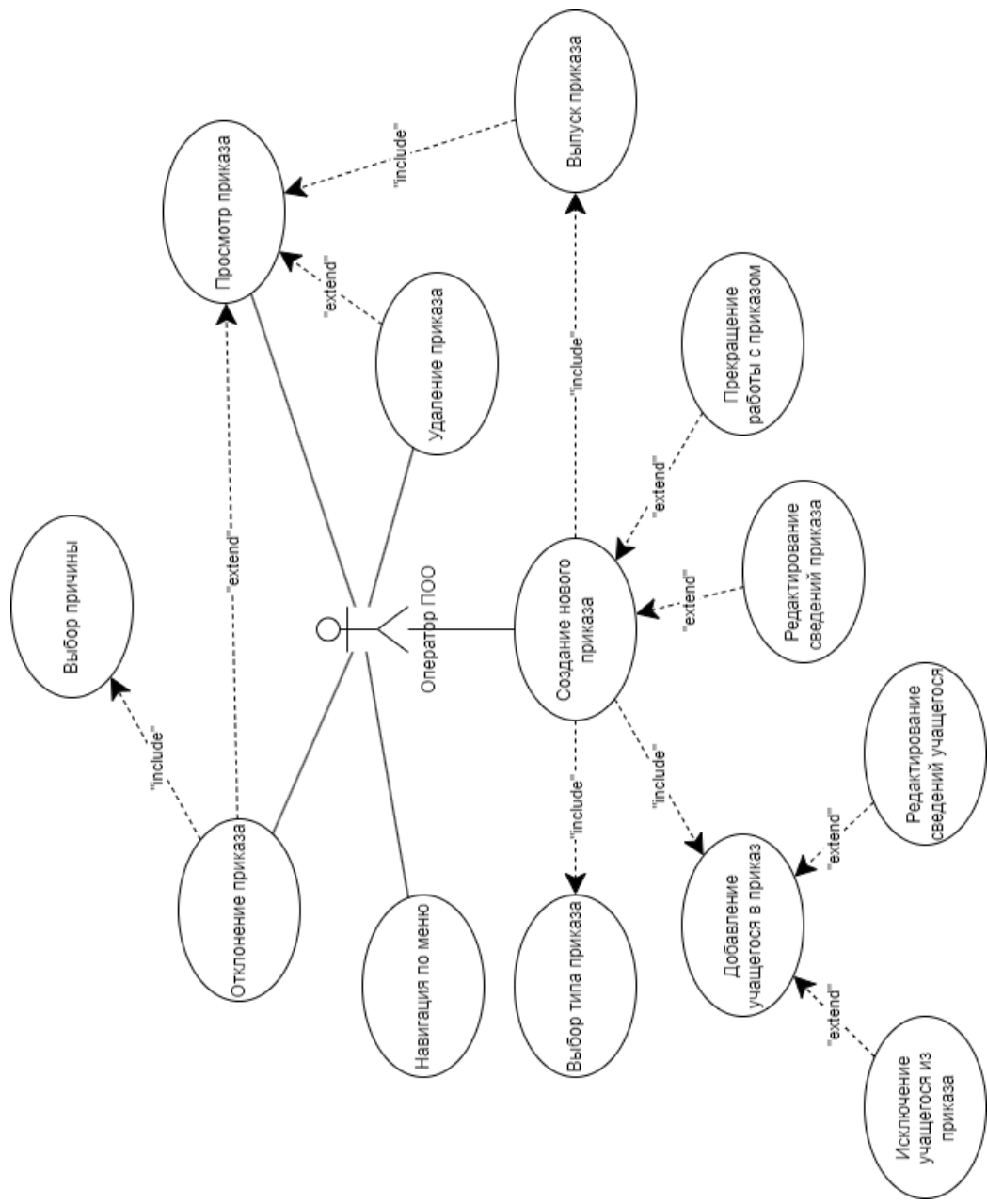
Изм.	Лист	№ докум.	Подпись	Дата

ПГУ 1.090301.36.001 ПЗ

Лист

65

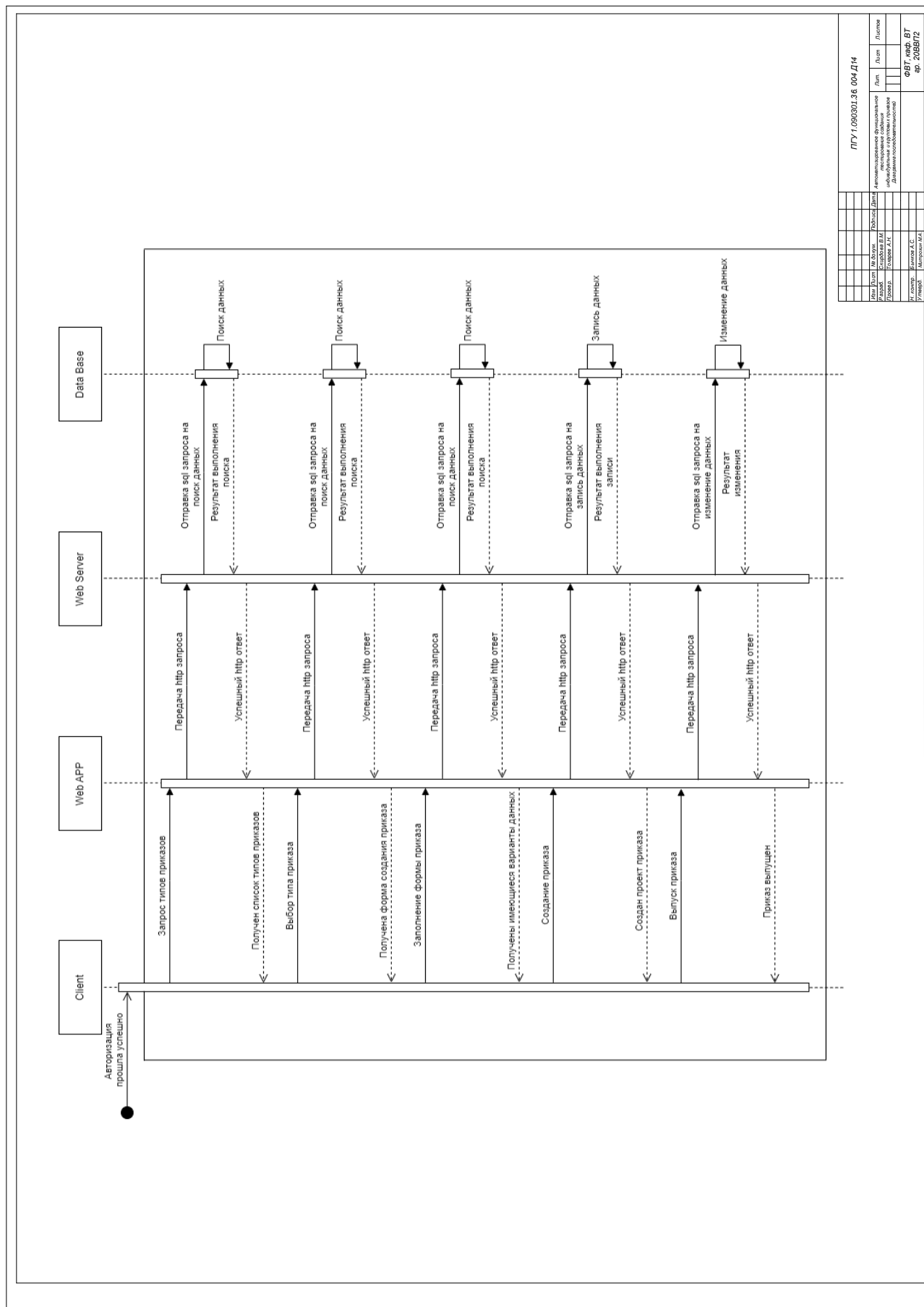
Приложение А.2 – Диаграмма прецедентов



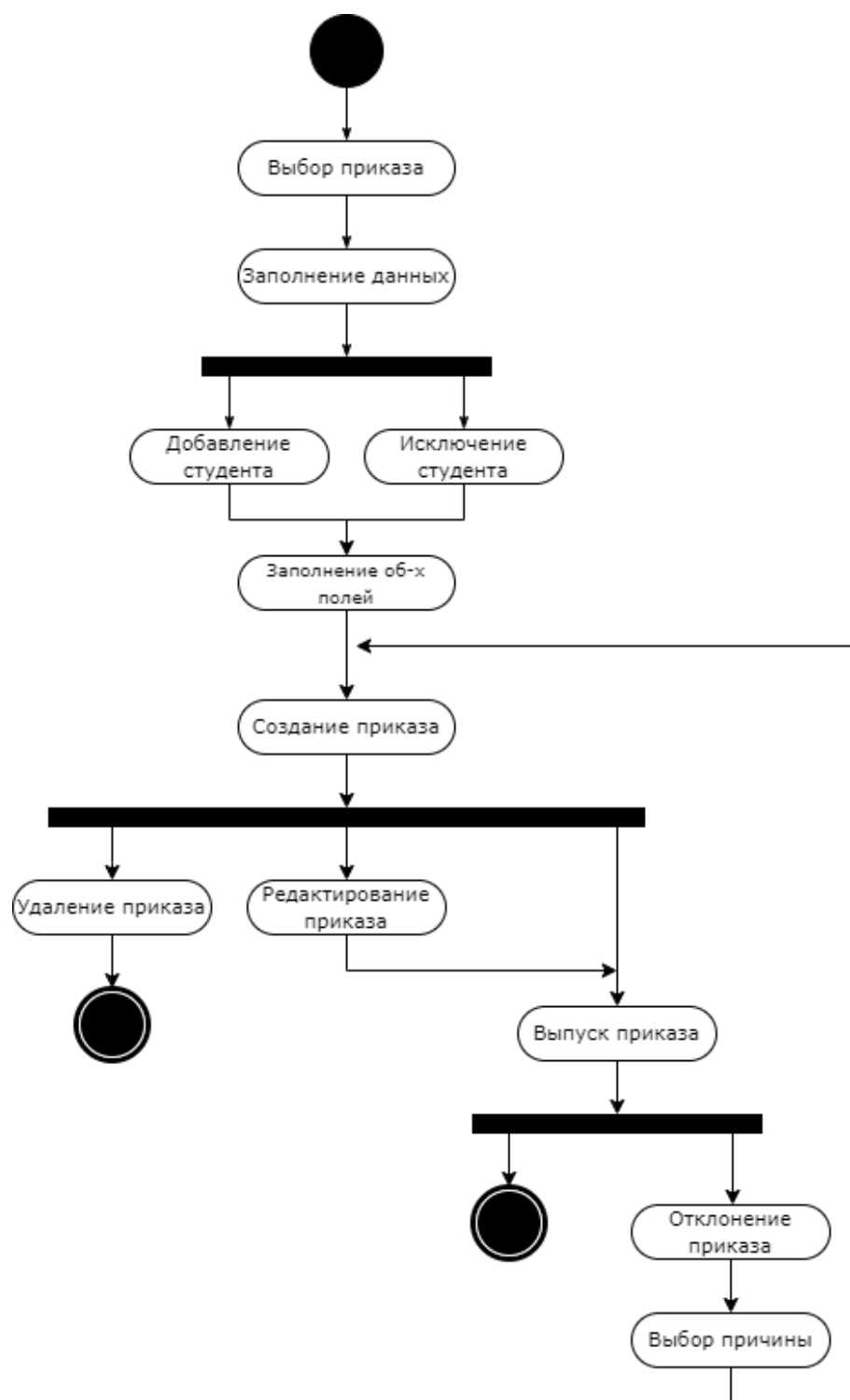
ПГУ 1.090301.36.001 ПЗ									
Изм.	Лист	№ докум.	Подпись	Дата	Исполнитель	Проверенный	Утвержденный	Согласованный	Согласованный
					Создана В.М.	Проверен А.Н.	Утвержден	Согласован	Согласован
ФЭТ, ред. В.Т. № 2008/72									

Изм.	Лист	№ докум.	Подпись	Дата

Приложение А.3 – Диаграмма последовательностей



Приложение А.4 – Диаграмма деятельности



					ПГУ 1.090301.36.005 Д17				
Изм.	Лист	№ докум.	Подпись	Дата	Автоматизированное функциональное тестирование создания индивидуальных и групповых приказов Дирекции деятельности	Лит.	Лист	Листов	ФВТ, каф. ВТ вр. 20ВВГ72
Разработчик	Смирнов В.М.	Проверен	Госарев А.И.						
И. экз.	Бачков А.С.	Утвержден	Мирошкин М.А.						

Изм.	Лист	№ докум.	Подпись	Дата

ПГУ 1.090301.36.001 ПЗ

Лист

68

Приложение Б

(Обязательное)

Автоматизированное функциональное тестирование создания индивидуальных
и групповых приказов
Листинг программы

Файл OrderTest.java

```
package orders;

import base.BaseSelenideTest;
import common.Rand;
import constant.Authors;
import constant.Users;
import io.qameta.allure.Epic;
import io.qameta.allure.Owner;
import org.testng.annotations.Test;
import
pages.applications.newcreation.pages.CreateApplicationPage;
import
pages.applications.registry.pages.RegistryApplicationsPage;
import pages.main.MainPage;
import pages.order.creation.pages.CreateOrderPage;
import
pages.order.creation.pages.group.academic.leave.CreateAcademicLeaveO
rderPage;
import
pages.order.creation.pages.group.enrollment.CreateEnrollmentOrderPag
e;
import
pages.order.creation.pages.group.expulsion.CreateExpulsionGroupOrder
Page;
import
pages.order.creation.pages.group.recovery.CreateRecoveryOrderPage;
import
pages.order.creation.pages.group.release.CreateReleaseOrderPage;
import
pages.order.creation.pages.group.transfer.CreateTransferOrderPage;
import
pages.order.creation.pages.group.transfer.bup.CreateTransferToBupOrd
erPage;
import
pages.order.creation.pages.group.transfer.course.CreateTransferCours
eOrderPage;
import
pages.order.creation.pages.group.transfer.eup.CreateTransferToEupOrd
erPage;
import
pages.order.creation.pages.individual.expulsion.CreateExpulsionIndiv
idualOrderPage;
import
pages.order.creation.pages.individual.vacationrecovery.CreateVacatio
nRecoveryOrderPage;
import pages.order.orders.pages.OrderBookPage;
import
pages.order.pages.academic.leave.OrderAcademicLeaveGroupPage;
import pages.order.pages.enrollment.OrderEnrollmentGroupPage;
import
pages.order.pages.expulsion.group.OrderExpulsionGroupPage;
```

					ПГУ 1.090301.36.001 ПЗ	Лист
						70
Изм.	Лист	№ докум.	Подпись	Дата		

```

import
pages.order.pages.expulsion.individual.OrderExpulsionIndividualPage;
import pages.order.pages.graduation.OrderGraduationGroupPage;
import pages.order.pages.recovery.OrderRecoveryGroupPage;
import pages.order.pages.transfer.OrderTransferGroupPage;
import
pages.order.pages.transfer.bup.OrderTransferToBupGroupPage;
import
pages.order.pages.transfer.course.OrderTransferCourseGroupPage;
import
pages.order.pages.transfer.eup.OrderTransferToEupGroupPage;
import
pages.order.pages.vacationrecovery.OrderVacationRecoveryPage;
import
pages.order.parts.editing.enrollment.EditingEnrollmentGroupDetailsPart;
import
pages.order.parts.editing.transfer.course.EditingTransferCourseGroupPart;
import pages.registries.contingent.PersonalFilePage;
import pages.registries.contingent.RegistriesContingentPage;
import
pages.registries.contingent.academicleave.AcademicLeavePage;
import pages.registries.contingent.enrolle.CreateEnrollPage;
import pages.registries.contingent.enrolle.EnrollPage;
import
pages.registries.contingent.financetype.AppChangeFinanceTypePage;
import
pages.registries.contingent.financetype.ChangeFinanceType;
import
pages.registries.contingent.financetype.CreateAppChangeFinanceTypePage;
import
pages.registries.contingent.financetype.CreateChangeFinanceTypePage;
import pages.registries.contingent.graduate.GraduationPage;
import
pages.registries.contingent.graduate.CreateGraduationPage;
import pages.registries.contingent.parts.InfoAboutOrderPart;
import pages.registries.contingent.parts.ModalWindows;
import
pages.registries.contingent.academicleave.CreateAcademicLeavePage;
import pages.registries.contingent.recovery.CreateRecoveryPage;
import pages.registries.contingent.recovery.RecoveryPage;
import
pages.registries.contingent.tranfers.anothergroup.CreateTransferInGroupPage;
import
pages.registries.contingent.tranfers.anothergroup.TransferInGroupPage;
import
pages.registries.contingent.tranfers.bup.CreateTransferToBUPPage;
import
pages.registries.contingent.tranfers.bup.TransferToBUPPage;

```

```

import
pages.registries.contingent.tranfers.eup.CreateTransferToEUPPage;
import
pages.registries.contingent.tranfers.eup.TransferToEUPPage;
import parts.NavBar;
import parts.notification.CancelWorkNotification;
import parts.notification.SuccessNotification;
import steps.page.PageSteps;

import java.time.LocalDate;
import java.time.format.DateTimeFormatter;
import java.util.HashMap;
import java.util.Map;

@Epic("Приказы")
public class OrderTest extends BaseSelenideTest {
    @Test(description = "Приказы/ Групповой приказ о выпуске/
Роль Оператор ПОО")
    @Owner(Authors.SKIRDOVA_VM)
    public void createGroupReleaseOrder() {
        DateTimeFormatter dateTimeFormatter =
DateTimeFormatter.ofPattern("ddMMyyyy");
        LocalDate today = LocalDate.now();

        Map<String, String> infoPart = new HashMap<>();
        infoPart.put("orderNumber", "Автотест" +
Rand.getRandomRuWord(5).toLowerCase());
        infoPart.put("orderFrom",
today.format(dateTimeFormatter));

        Map<String, String> groupListPart = new HashMap<>();
        groupListPart.put("deductFrom",
today.format(dateTimeFormatter));
        groupListPart.put("fromGroupSelect", "19ГД-4з");

        PageSteps.auth(Users.getUserByName(Users.adm1))
            .steps.checkInitElementsHeaderFooter();
        CreateOrderPage createOrderPage = new
NavBar().openCreateOrder();
        createOrderPage
            .checkInitElements();
        CreateReleaseOrderPage createReleaseOrderPage =
createOrderPage.selectOrderTypeRelease();
        createReleaseOrderPage
            .checkInitElements();
        createReleaseOrderPage.getInformationPart()
            .fillFields(infoPart);
        createReleaseOrderPage.getReleaseGroupsListPart()
            .fillFields(groupListPart);
        createOrderPage
            .clickCreateOrderBtn()
            .checkSuccessCreate();
    }
}

```

					ПГУ 1.090301.36.001 ПЗ	Лист
						72
Изм.	Лист	№ докум.	Подпись	Дата		

```

        OrderGraduationGroupPage orderGraduationGroupPage = new
OrderGraduationGroupPage();
        orderGraduationGroupPage
            .checkOrderStatus("проект приказа")
            .clickOrderInformationChangeButton();
        orderGraduationGroupPage.getEditingOrderDetails()
            .checkInitElements()
            .changeOrderNum("Автотест" +
Rand.getRandomRuWord(5).toLowerCase());
            .clickSaveButton();
        orderGraduationGroupPage
            .clickGroupChangeButton();

orderGraduationGroupPage.getEditingGraduationGroupDetailsPart()
            .checkInitElements()
            .changeDeductFrom()
            .deleteStudent(1)
            .hideTable()
            .clickSaveButton();
        orderGraduationGroupPage
            .clickActionButton()
            .clickDeleteOrder();
        orderGraduationGroupPage.getModalWindow()
            .clickDelete();
        orderGraduationGroupPage
            .checkSuccessDelete();
        PageSteps.logout();
    }

@Test(description = "Приказы/Групповой приказ о переводе из
группы в группу/ Роль Оператор ПОО")
@Owner(Authors.SKIRDOVA_VM)
public void createGroupTransferOrder() {
    DateTimeFormatter dateTimeFormatter =
DateTimeFormatter.ofPattern("ddMMyyyy");
    LocalDate today = LocalDate.now();

    Map<String, String> infoPart = new HashMap<>();
    infoPart.put("orderNumber", "Автотест" +
Rand.getRandomRuWord(5).toLowerCase());
    infoPart.put("orderFrom",
today.format(dateTimeFormatter));

    Map<String, String> groupListPart = new HashMap<>();
    groupListPart.put("transferFrom",
today.minusDays(1).format(dateTimeFormatter));
    groupListPart.put("fromGroupSelect", "19ГД-146");
    groupListPart.put("inGroupSelect", "19ГД-147");

    PageSteps.auth(Users.getUserByName(Users.adm1))
        .steps.checkInitElementsHeaderFooter();
    CreateOrderPage createOrderPage = new
NavBar().openCreateOrder();

```

```

        createOrderPage
            .checkInitElements();
        CreateTransferOrderPage transferOrderPart =
createOrderPage.selectOrderTypeTransfer();
        transferOrderPart
            .checkInitElements();
        transferOrderPart.getInformationPart()
            .fillFields(infoPart);
        transferOrderPart.getGroupListPart()
            .fillFields(groupListPart);
        createOrderPage
            .clickCreateOrderBtn()
            .checkSuccessCreate();
        OrderTransferGroupPage transferGroupPage = new
OrderTransferGroupPage();
        transferGroupPage
            .checkOrderStatus("проект приказа")
            .clickGroupChangeButton();
        transferGroupPage.getTransferGroupPart()
            .checkInitElements()
            .changeTransferFrom()
            .deleteStudent(1)
            .hideTable()
            .clickSaveButton();
        transferGroupPage
            .clickActionButton()
            .clickDeleteOrder();
        transferGroupPage.getModalWindow()
            .clickDelete();
        transferGroupPage
            .checkSuccessDelete();
        PageSteps.logout();
    }

```

```

@Test(description = "Приказы/ групповой приказ о
восстановлении / Роль Оператор ПОО")
@Owner(Authors.SKIRDOVA_VM)
public void createGroupRecoveryOrder() {

    DateTimeFormatter dateTimeFormatter =
DateTimeFormatter.ofPattern("ddMMyyyy");
    LocalDate today = LocalDate.now();

    Map<String, String> infoPart = new HashMap<>();
    infoPart.put("orderNumber", "Автотест" +
Rand.getRandomRuWord(5).toLowerCase());
    infoPart.put("orderFrom",
today.format(dateTimeFormatter));
    PageSteps.auth(Users.getUserByName(Users.adm1))
        .steps.checkInitElementsHeaderFooter();
    CreateOrderPage createOrderPage = new
NavBar().openCreateOrder();
    createOrderPage

```

```

        .checkInitElements();
        CreateRecoveryOrderPage createRecoveryOrderPage =
createOrderPage.selectOrderTypeRecovery();
        createRecoveryOrderPage
            .checkInitElements();
        createRecoveryOrderPage.getInformationPart()
            .fillFields(infoPart);
        createRecoveryOrderPage.getGroupsListPart()

.setDateForAllGroups(today.minusDays(1).format(dateTimeFormatter));
        createOrderPage
            .clickCreateOrderBtn()
            .checkSuccessCreate();
        OrderRecoveryGroupPage recoveryGroupPage = new
OrderRecoveryGroupPage();
        recoveryGroupPage.getModalWindow()
            .clickCross();
        recoveryGroupPage
            .checkOrderStatus("проект приказа")
            .chooseGroup()
            .clickGroupChangeButton();
        recoveryGroupPage.getRecoveryGroupDetailsPart()
            .checkInitElements()
            .changeRecoverFrom()
            .clickSaveButton();
        recoveryGroupPage
            .chooseAnotherGroup()
            .clickGroupCartButton();
        recoveryGroupPage
            .clickDelete();
        recoveryGroupPage
            .clickActionButton()
            .clickDeleteOrder();
        recoveryGroupPage.getModalWindow()
            .clickDelete();
        recoveryGroupPage
            .checkSuccessDelete();
        PageSteps.logout();
    }

```

@Test(description = "Приказы/ Групповой приказ о переводе
на ЭУП/ Роль Оператор ПОО")

@Owner(Authors.SKIRDOVA_VM)

public void createGroupTransferToEupOrder() {

```

        DateTimeFormatter dateTimeFormatter =
DateTimeFormatter.ofPattern("ddMMyyyy");
        LocalDate today = LocalDate.now();

```

```

        Map<String, String> infoPart = new HashMap<>();
        infoPart.put("orderNumber", "Автотест" +
Rand.getRandomRuWord(5).toLowerCase());

```

					ПГУ 1.090301.36.001 ПЗ	Лист
						75
Изм.	Лист	№ докум.	Подпись	Дата		

```

        infoPart.put("orderFrom",
today.format(dateTimeFormatter));

        Map<String, String> groupListPart = new HashMap<>();
        groupListPart.put("transferFrom",
today.minusDays(1).format(dateTimeFormatter));
        groupListPart.put("fromGroupSelect", "21ИИФ2-1");
        groupListPart.put("inGroupSelect", "21ИИФ2-1.");

        Map<String, String> secondGroupListPart = new
HashMap<>();
        secondGroupListPart.put("transferFromAddGroup",
today.minusDays(1).format(dateTimeFormatter));
        secondGroupListPart.put("fromGroupSelectAddGroup", "СТ-
112");
        secondGroupListPart.put("inGroupSelectAddGroup",
"Группа test");

        PageSteps.auth(Users.getUserByName(Users.adm1))
                .steps.checkInitElementsHeaderFooter();
        CreateOrderPage createOrderPage = new
NavBar().openCreateOrder();
        createOrderPage
                .checkInitElements();
        CreateTransferToEupOrderPage transferToEupOrderPage =
createOrderPage.selectOrderTypeTransferToEup();
        transferToEupOrderPage
                .checkInitElements();
        transferToEupOrderPage.getInformationPart()
                .fillFields(infoPart);
        transferToEupOrderPage.getGroupsListPart()
                .fillFields(groupListPart);
        createOrderPage
                .clickCreateOrderBtn()
                .checkSuccessCreate();
        OrderTransferToEupGroupPage transferToEupGroupPage =
new OrderTransferToEupGroupPage();
        transferToEupGroupPage
                .checkOrderStatus("проект приказа")
                .clickOrderInformationChangeButton();
        transferToEupGroupPage.getOrderDetailsPart()
                .checkInitElements()
                .clearOrderNum()
                .clickCancelButton();
        transferToEupGroupPage.getModalWindow()
                .clickCancel();
        transferToEupGroupPage.getOrderDetailsPart()
                .changeOrderNum("Автотест" +
Rand.getRandomRuWord(5).toLowerCase())
                .clickSaveButton();
        transferToEupGroupPage
                .clickGroupChangeButton();

```

```

transferToEupGroupPage.getTransferToEupGroupDetailsPart()
    .checkInitElements()
    .clearTransferFrom()
    .clickCancelButton();
transferToEupGroupPage.getModalWindow()
    .clickCancel();

transferToEupGroupPage.getTransferToEupGroupDetailsPart()
    .fillFormField("transferFrom",
today.format(dateTimeFormatter))
    .clickSaveButton();
transferToEupGroupPage
    .clickAddGroupButton();

transferToEupGroupPage.getTransferToEupGroupDetailsPart()
    .fillFields(secondGroupListPart)
    .clickCancelButton();
transferToEupGroupPage.getModalWindow()
    .clickCancel();

transferToEupGroupPage.getTransferToEupGroupDetailsPart()
    .clickAddButton();
transferToEupGroupPage
    .clickGroupChangeButton(2);

transferToEupGroupPage.getTransferToEupGroupDetailsPart()
    .deleteStudent(1)
    .clickSaveButton();
transferToEupGroupPage
    .clickGroupDeleteButton(2);
transferToEupGroupPage
    .clickDelete();
transferToEupGroupPage
    .clickActionButton()
    .clickDeleteOrder();
transferToEupGroupPage.getModalWindow()
    .clickDelete();
transferToEupGroupPage
    .checkSuccessDelete();
PageSteps.logout();
}

@Test(description = "Приказы/ Групповой приказ о переводе
на БУП/ Роль Оператор ПОО")
@Owner(Authors.SKIRDOVA_VM)
public void createGroupTransferToBupOrder() {

    DateTimeFormatter dateTimeFormatter =
DateTimeFormatter.ofPattern("ddMMyyyy");
    LocalDate today = LocalDate.now();

    Map<String, String> infoPart = new HashMap<>();

```

					ПГУ 1.090301.36.001 ПЗ	Лист
						77
Изм.	Лист	№ докум.	Подпись	Дата		

```

        infoPart.put("orderNumber", "Автотест" +
Rand.getRandomRuWord(5).toLowerCase());
        infoPart.put("orderFrom",
today.format(dateTimeFormatter));

        Map<String, String> groupListPart = new HashMap<>();
        groupListPart.put("transferFrom",
today.minusDays(1).format(dateTimeFormatter));
        groupListPart.put("fromGroupSelect", "Группа test");
        groupListPart.put("inGroupSelect", "СТ-112");

        Map<String, String> secondGroupListPart = new
HashMap<>();
        secondGroupListPart.put("transferFromAddGroup",
today.minusDays(1).format(dateTimeFormatter));
        secondGroupListPart.put("fromGroupSelectAddGroup",
"21ИИФ2-1.");
        secondGroupListPart.put("inGroupSelectAddGroup",
"21ИИФ2-1");

        PageSteps.auth(Users.getUserByName(Users.adm1))
                .steps.checkInitElementsHeaderFooter();
        CreateOrderPage createOrderPage = new
NavBar().openCreateOrder();
        createOrderPage
                .checkInitElements();
        CreateTransferToBupOrderPage transferToBupOrderPage =
createOrderPage.selectOrderTypeTransferToBup();
        transferToBupOrderPage
                .checkInitElements();
        transferToBupOrderPage.getInformationPart()
                .fillFields(infoPart);
        transferToBupOrderPage.getGroupsListPart()
                .fillFields(groupListPart);
        createOrderPage
                .clickCreateOrderBtn()
                .checkSuccessCreate();
        OrderTransferToBupGroupPage transferToBupGroupPage =
new OrderTransferToBupGroupPage();
        transferToBupGroupPage
                .checkOrderStatus("проект приказа")
                .clickOrderInformationChangeButton();
        transferToBupGroupPage.getOrderDetailsPart()
                .checkInitElements()
                .clearOrderNum()
                .clickCancelButton();
        transferToBupGroupPage.getModalWindow()
                .clickCancel();
        transferToBupGroupPage.getOrderDetailsPart()
                .changeOrderNum("Автотест" +
Rand.getRandomRuWord(5).toLowerCase())
                .clickSaveButton();
        transferToBupGroupPage

```

```

        .clickGroupChangeButton();

transferToBupGroupPage.getTransferToBupGroupDetailsPart()
        .checkInitElements()
        .clearTransferFrom()
        .clickCancelButton();
transferToBupGroupPage.getModalWindow()
        .clickCancel();

transferToBupGroupPage.getTransferToBupGroupDetailsPart()
        .fillFormField("transferFrom",
today.format(dateTimeFormatter))
        .clickSaveButton();
transferToBupGroupPage
        .clickAddGroupButton();

transferToBupGroupPage.getTransferToBupGroupDetailsPart()
        .fillFields(secondGroupListPart)
        .clickCancelButton();
transferToBupGroupPage.getModalWindow()
        .clickCancel();

transferToBupGroupPage.getTransferToBupGroupDetailsPart()
        .clickAddButton();
transferToBupGroupPage
        .clickGroupChangeButton(2);

transferToBupGroupPage.getTransferToBupGroupDetailsPart()
        .deleteStudent(1)
        .clickSaveButton();
transferToBupGroupPage
        .clickGroupDeleteButton(2);
transferToBupGroupPage
        .clickDelete();
transferToBupGroupPage
        .clickActionButton()
        .clickDeleteOrder();
transferToBupGroupPage.getModalWindow()
        .clickDelete();
transferToBupGroupPage
        .checkSuccessDelete();
PageSteps.logout();
}

@Test(description = "Приказы/ Групповой приказ о переводе с
курса на курс/ Роль Оператор ПОО")
@Owner(Authors.SKIRDOVA_VM)
public void createGroupTransferCourseOrder() {

    DateTimeFormatter dateTimeFormatter =
DateTimeFormatter.ofPattern("ddMMyyyy");
    LocalDate today = LocalDate.now();

```

```

        Map<String, String> infoPart = new HashMap<>();
        infoPart.put("orderNumber", "Автотест" +
Rand.getRandomRuWord(5).toLowerCase());
        infoPart.put("orderFrom",
today.format(dateTimeFormatter));

        Map<String, String> groupListPart = new HashMap<>();
        groupListPart.put("transferFrom",
today.minusDays(1).minusYears(3).format(dateTimeFormatter));
        groupListPart.put("fromGroupSelect", "21ГД-2з");
        groupListPart.put("inGroupSelect", "21ГД-3з_test");

        PageSteps.auth(Users.getUserByName(Users.adm1))
                .steps.checkInitElementsHeaderFooter();
        CreateOrderPage createOrderPage = new
NavBar().openCreateOrder();
        createOrderPage
                .checkInitElements();
        CreateTransferCourseOrderPage transferOrderPage =
createOrderPage.selectOrderTypeTransferCourse();
        transferOrderPage
                .checkInitElements();
        transferOrderPage.getInformationPart()
                .fillFields(infoPart);
        transferOrderPage.getGroupsListPart()
                .fillFields(groupListPart);
        createOrderPage
                .clickCreateOrderBtn()
                .checkSuccessCreate();
        OrderTransferCourseGroupPage transferCourseGroupPage =
new OrderTransferCourseGroupPage();
        transferCourseGroupPage
                .checkOrderStatus("проект приказа")
                .clickActionButton()
                .clickReleaseOrder();
        transferCourseGroupPage.getModalWindow()
                .clickRelease();
        transferCourseGroupPage
                .checkErrorDateRelease();
        transferCourseGroupPage
                .clickGroupChangeButton();
        EditingTransferCourseGroupPart
editingTransferCourseGroupPart = new
EditingTransferCourseGroupPart();
        editingTransferCourseGroupPart
                .changeTransferFrom()
                .deleteStudent(1)
                .deleteStudent(0)
                .hideTable()
                .clickSaveButton();
        transferCourseGroupPage
                .clickActionButton()
                .clickDeleteOrder();

```

```

        transferCourseGroupPage.getModalWindow()
            .clickDelete();
        transferCourseGroupPage
            .checkSuccessDelete();
        PageSteps.logout();
    }

    @Test(description = "Приказы/ Групповой приказ о
зачислении/ Роль Оператор ПОО")
    @Owner(Authors.SKIRDOVA_VM)
    public void createGroupEnrollmentOrder() {

        DateTimeFormatter dateTimeFormatter =
DateTimeFormatter.ofPattern("ddMMyyyy");
        LocalDate today = LocalDate.now();

        Map<String, String> infoPart = new HashMap<>();
        infoPart.put("orderNumber", "Автотест" +
Rand.getRandomRuWord(5).toLowerCase());
        infoPart.put("orderFrom",
today.format(dateTimeFormatter));

        Map<String, String> groupListPart = new HashMap<>();
        groupListPart.put("transferFrom",
today.minusDays(1).minusYears(3).format(dateTimeFormatter));
        groupListPart.put("fromGroupSelect", "21ГД-2з");
        groupListPart.put("inGroupSelect", "21ГД-3з_test");

        PageSteps.auth(Users.getUserByName(Users.adm1))
            .steps.checkInitElementsHeaderFooter();
        CreateOrderPage createOrderPage = new
NavBar().openCreateOrder();
        createOrderPage
            .checkInitElements();
        CreateEnrollmentOrderPage createEnrollmentOrderPage =
createOrderPage.selectOrderTypeEnrollment();
        createEnrollmentOrderPage
            .checkInitElements();
        createEnrollmentOrderPage.getInformationPart()
            .fillFields(infoPart);
        createEnrollmentOrderPage.getGroupsListPart()

.setDateForAllGroups(today.minusYears(3).format(dateTimeFormatter)
, today.format(dateTimeFormatter));
        createOrderPage
            .clickCreateOrderBtn()
            .checkSuccessCreate();
        OrderEnrollmentGroupPage enrollmentGroupPage = new
OrderEnrollmentGroupPage();
        enrollmentGroupPage.getModalWindow()
            .clickCross();
        enrollmentGroupPage

```

```

        .clickActionButton();
        .clickReleaseOrder();
enrollmentGroupPage.getModalWindow()
        .clickRelease();
enrollmentGroupPage
        .checkMessage("отсутствует идентификатор МЭШ");
enrollmentGroupPage
        .clickGroupChangeButton();
EditingEnrollmentGroupDetailsPart editingEnrollment =
new EditingEnrollmentGroupDetailsPart();
        editingEnrollment
                .checkInitElements()
                .changeEnrollFrom()
                .deleteStudent(1)
                .clickSaveButton();
enrollmentGroupPage
        .chooseAnotherGroup()
        .clickGroupCartButton()
        .clickDelete();
enrollmentGroupPage
        .clickActionButton()
        .clickDeleteOrder();
enrollmentGroupPage.getModalWindow()
        .clickDelete();
enrollmentGroupPage
        .checkSuccessDelete();
PageSteps.logout();

}

@Test(description = "Приказы/ Автомат. приказ о
восстановлении из академ.отпуска/ Роль Оператор ПОО")
@Owner(Authors.SKIRDOVA_VM)
public void recoveryFromAcademicLeave() {

        DateTimeFormatter dateTimeFormatter =
DateTimeFormatter.ofPattern("ddMMyyyy");
        LocalDate today = LocalDate.now();

        Map<String, String> infoPart = new HashMap<>();
        infoPart.put("onCourse", "1 курс");
        infoPart.put("inGroup", "КСК-112Д");

        PageSteps.auth(Users.getUserByName(Users.adm1))
                .steps.checkInitElementsHeaderFooter();
        MainPage mainPage = new MainPage();
        mainPage
                .checkInitElements()
                .chooseQuickAccessPoint(0);
        OrderBookPage orderBookPage = new OrderBookPage();
        orderBookPage
                .checkInitElements();
        orderBookPage.getTablePart()

```

```

        .goToOrderByValue(
            "Индивидуальный приказ о восстановлении
из академического отпуска");
        CreateVacationRecoveryOrderPage
createVacationRecoveryOrderPage = new
CreateVacationRecoveryOrderPage();
        createVacationRecoveryOrderPage
            .checkInitElements()
            .fillFields(infoPart)
            .clickSaveButton();
        OrderVacationRecoveryPage vacationRecoveryPage = new
OrderVacationRecoveryPage();
        vacationRecoveryPage
            .clickOrderInformationChangeButton();
        vacationRecoveryPage.getVacationRecoveryOrderPart()
            .checkInitElements()

.changeTransferFrom(today.format(dateTimeFormatter))
            .clickCancelButton();
        vacationRecoveryPage.getModalWindow()
            .clickCancel();
        vacationRecoveryPage.getVacationRecoveryOrderPart()
            .clickSaveButton();
        vacationRecoveryPage
            .clickActionButton()
            .clickDeleteOrder();
        vacationRecoveryPage.getModalWindow()
            .clickDelete();
        vacationRecoveryPage
            .checkSuccessDelete();
        PageSteps.logout();

    }

    @Test(description = "Приказы/ Индивидуальный приказ об
отчислении/ Роль Оператор ПОО")
    @Owner(Authors.SKIRDOVA_VM)
    public void createIndividualExpulsionOrder() {

        DateTimeFormatter dateTimeFormatter =
DateTimeFormatter.ofPattern("ddMMyyyy");
        LocalDate today = LocalDate.now();

        Map<String, String> infoPart = new HashMap<>();
        infoPart.put("expelFrom",
today.format(dateTimeFormatter));
        infoPart.put("reason", "По собственному желанию");

        PageSteps.auth(Users.getUserByName(Users.adm1))
            .steps.checkInitElementsHeaderFooter();
        RegistriesContingentPage rcPage = new
NavBar().openRegistriesContingent();
        rcPage

```

```

        .checkInitElements()
        .stepsPersonalFile.goToPersonalFile(7);
    PersonalFilePage personalFilePage = new
PersonalFilePage();
    personalFilePage
        .checkInitElements()
        .clickActions()
        .clickExpel();
    CreateExpulsionIndividualOrderPage
expulsionIndividualOrderPage
    = new CreateExpulsionIndividualOrderPage();
    expulsionIndividualOrderPage
        .checkInitElements()
        .fillFields(infoPart);
    expulsionIndividualOrderPage.getInformationPart()
        .fillFormField("orderNumber", "Автотест" +
Rand.getRandomRuWord(5).toLowerCase());
    expulsionIndividualOrderPage
        .clickCancelBtn();
    expulsionIndividualOrderPage.getModalWindow()
        .clickCancel();
    expulsionIndividualOrderPage
        .clickCreateOrderBtn();
    OrderExpulsionIndividualPage expulsionIndividualPage =
new OrderExpulsionIndividualPage();
    expulsionIndividualPage
        .checkOrderStatus("проект приказа")
        .clickOrderInformationChangeButton();
    expulsionIndividualPage.getEditingExpulsionOrderPart()
        .checkInitElements()
        .clearOrderNum()
        .clickCancelButton();
    expulsionIndividualPage.getModalWindow()
        .clickCancel();
    expulsionIndividualPage.getEditingExpulsionOrderPart()
        .changeOrderNum("Автотест" +
Rand.getRandomRuWord(5).toLowerCase())
        .clickSaveButton();
    expulsionIndividualPage
        .clickActionButton()
        .clickDeleteOrder();
    expulsionIndividualPage.getModalWindow()
        .clickDelete();
    expulsionIndividualPage
        .checkSuccessDelete();
    PageSteps.logout();
}

@Test(description = "Приказы/ Групповой приказ о
предоставлении академического отпуска/ Роль Оператор ПОО")
@Owner(Authors.SKIRDOVA_VM)
public void createGroupAcademicLeaveOrder() {

```

```

        DateTimeFormatter dateTimeFormatter =
DateTimeFormatter.ofPattern("ddMMyyyy");
        LocalDate today = LocalDate.now();

        Map<String, String> infoPart = new HashMap<>();
        infoPart.put("orderNumber", "Автотест" +
Rand.getRandomRuWord(5).toLowerCase());
        infoPart.put("orderFrom",
today.format(dateTimeFormatter));

        Map<String, String> groupListPart = new HashMap<>();
        groupListPart.put("basedOn", null);
        groupListPart.put("basedOnFrom", null);
        groupListPart.put("academicPeriod", null);
        groupListPart.put("startEnd",
today.format(dateTimeFormatter) +
today.plusMonths(1).format(dateTimeFormatter));
        groupListPart.put("fromGroupSelect", "19ГД-136");

        PageSteps.auth(Users.getUserByName(Users.adm1))
                .steps.checkInitElementsHeaderFooter();
        CreateOrderPage createOrderPage = new
NavBar().openCreateOrder();
        createOrderPage
                .checkInitElements();
        CreateAcademicLeaveOrderPage
createAcademicLeaveOrderPage =
createOrderPage.selectOrderTypeAcademicLeave();
        createAcademicLeaveOrderPage
                .checkInitElements(groupListPart);
        createAcademicLeaveOrderPage.getInformationPart()
                .fillFields(infoPart);
        createAcademicLeaveOrderPage.getGroupsListPart()
                .fillFields(groupListPart)
                .fillFormField("person", "Калита Павел
Андреевич");
        createOrderPage
                .clickCancelBtn();
        createOrderPage.getModalWindow()
                .checkNotification("Вы уверены, что хотите
прекратить работу?" +
                                " Внесенные данные не будут
сохранены.");
        createOrderPage
                .clickCancel();
        createAcademicLeaveOrderPage.getInformationPart()
                .clearOrderNumber();
        createOrderPage
                .checkFieldHint();
        createAcademicLeaveOrderPage.getInformationPart()
                .fillFormField("orderNumber", "Автотест" +
Rand.getRandomRuWord(5).toLowerCase());
        createOrderPage
                .clickCreateOrderBtn();

```

```

        createOrderPage.getModalWindow()
            .clickContinue();
        createOrderPage
            .checkSuccessCreate();
        OrderAcademicLeaveGroupPage academicLeaveGroupPage =
new OrderAcademicLeaveGroupPage();
        academicLeaveGroupPage
            .checkOrderStatus("проект приказа")
            .clickOrderInformationChangeButton();
        academicLeaveGroupPage.getOrderDetailsPart()
            .checkInitElements()
            .clearOrderNum()
            .checkFieldHint()
            .clickCancelButton();
        academicLeaveGroupPage.getModalWindow()
            .clickCancel();
        academicLeaveGroupPage.getOrderDetailsPart()
            .changeOrderNum("Автотест" +
Rand.getRandomRuWord(5).toLowerCase());
        academicLeaveGroupPage
            .clickSaveButton();
        academicLeaveGroupPage
            .clickActionButton()
            .clickDeleteOrder();
        academicLeaveGroupPage.getModalWindow()
            .clickDelete();
        academicLeaveGroupPage
            .checkSuccessDelete();

        PageSteps.logout();
    }

    @Test(description = "Приказы/ Групповой приказа об
отчислении/ Роль Оператор ПОО ")
    @Owner(Authors.SKIRDOVA_VM)
    public void createGroupExpulsionOrder() {

        DateTimeFormatter dateTimeFormatter =
DateTimeFormatter.ofPattern("ddMMyyyy");
        LocalDate today = LocalDate.now();

        Map<String, String> infoPart = new HashMap<>();
        infoPart.put("orderNumber", "Автотест" +
Rand.getRandomRuWord(5).toLowerCase());
        infoPart.put("orderFrom",
today.format(dateTimeFormatter));

        Map<String, String> firstGroupListPart = new
HashMap<>();
        firstGroupListPart.put("basedOn", null);
        firstGroupListPart.put("basedOnFrom", null);
        firstGroupListPart.put("expelFrom",
today.format(dateTimeFormatter));
        firstGroupListPart.put("fromGroupSelect", "19ИСП-429");
    }

```

```

firstGroupListPart.put("reason", "По собственному
желанию");

PageSteps.auth(Users.getUserByName(Users.adm1))
    .steps.checkInitElementsHeaderFooter();
CreateOrderPage createOrderPage = new
NavBar().openCreateOrder();
createOrderPage
    .checkInitElements();
CreateExpulsionGroupOrderPage
createExpulsionGroupOrderPage =
createOrderPage.selectOrderTypeExpulsion();
createExpulsionGroupOrderPage
    .checkInitElements(firstGroupListPart);
createExpulsionGroupOrderPage.getInformationPart()
    .fillFields(infoPart);
createExpulsionGroupOrderPage.getGroupsListPart(1)
    .fillFields(firstGroupListPart)
    .fillFormField("person", "Акимов Дмитрий
Александрович");
createExpulsionGroupOrderPage.getManager()
    .clickAddStudent();
createExpulsionGroupOrderPage
    .checkInitElements(firstGroupListPart);
createExpulsionGroupOrderPage.getGroupsListPart(2)
    .fillFields(firstGroupListPart)
    .fillFormField("person", "random");
createOrderPage
    .clickCancelBtn();
createOrderPage.getModalWindow()
    .checkNotification("Вы уверены, что хотите
прекратить работу?" +
                        " Внесенные данные не будут
сохранены.")
    .clickCancel();
createExpulsionGroupOrderPage.getInformationPart()
    .clearOrderNumber();
createOrderPage
    .checkFieldHint();
createExpulsionGroupOrderPage.getInformationPart()
    .fillFormField("orderNumber", "Автотест" +
Rand.getRandomRuWord(5).toLowerCase());
createOrderPage
    .clickCreateOrderBtn();
OrderExpulsionGroupPage expulsionGroupPage = new
OrderExpulsionGroupPage();
expulsionGroupPage
    .checkOrderStatus("проект приказа")
    .clickOrderInformationChangeButton();
expulsionGroupPage.getEditingOrderDetails()
    .clearOrderNum()
    .checkFieldHint()
    .clickCancelButton();

```

```

expulsionGroupPage.getModalWindow()
    .clickCancel();
expulsionGroupPage.getEditingOrderDetails()
    .changeOrderNum("Автотест" +
Rand.getRandomRuWord(5).toLowerCase())
    .clickSaveButton();
expulsionGroupPage
    .clickStudentsChangeButton();
expulsionGroupPage.getEditingGroupExpulsionOrderPart()
    .checkInitElements()
    .clearReason()
    .checkWarning()
    .clickCancelButton();
expulsionGroupPage.getModalWindow()
    .clickCancel();
expulsionGroupPage.getEditingGroupExpulsionOrderPart()
    .changeReason()
    .clickSaveButton();
expulsionGroupPage
    .clickStudentsCartButton();
expulsionGroupPage.getModalWindow()
    .checkNotification("Вы уверены, что хотите
удалить обучающегося")
    .clickButton("Отмена");
expulsionGroupPage
    .clickStudentsCartButton();
expulsionGroupPage.getModalWindow()
    .clickDelete();
expulsionGroupPage
    .clickActionButton()
    .clickReleaseOrder();
expulsionGroupPage.getModalWindow()
    .clickRelease();
expulsionGroupPage = new OrderExpulsionGroupPage();
expulsionGroupPage
    .checkSuccessRelease()
    .checkOrderStatus("выпущен")
    .clickStudentsUnwrapButton();
expulsionGroupPage
    .checkField("На основании")
    .checkField("Отчислить с")
    .checkField("Обучение по специальности")
    .checkField("Причина");
expulsionGroupPage
    .clickRejectButton();
expulsionGroupPage.getCancelingOrderPart()
    .selectReason("Ошибка оператора")
    .clickRejectButton();
expulsionGroupPage
    .clickActionButton()
    .clickDeleteOrder();
expulsionGroupPage.getModalWindow()
    .clickDelete();

```

```

        expulsionGroupPage
            .checkSuccessDelete();

        PageSteps.logout();
    }

    @Test(description = "Приказы/Индивидуальный приказ о
выпуске/Роль Оператор ПОО")
    @Owner(Authors.SKIRDOVA_VM)
    public void orderRealise() {
        PageSteps.auth(Users.getUserByName(Users.adm1))
            .checkInitElements();
        CreateGraduationPage createGraduationPage = new
CreateGraduationPage();
        GraduationPage graduationPage = new GraduationPage();
        InfoAboutOrderPart infoAboutOrderPart = new
InfoAboutOrderPart(InfoAboutOrderPart.OrderDateLabels.GRADUATION.get
Value());

        ModalWindows modalWindows = new ModalWindows();
        String orderNumber = "pt-4018";

        Map<String, String> orderParameters = new HashMap<>();
        orderParameters.put("basedOnInput", orderNumber);
        orderParameters.put("orderNumberInput", orderNumber);
        orderParameters.put("orderDate", "01.02.2024");
        orderParameters.put("basedOnDate", "01.02.2024");
        orderParameters.put("graduationDate", "01.02.2024");

        String fio = "Басимов Дмитрий Фанилевич";
        new NavBar()
            .openRegistriesContingent()
            .searchByFio(fio)
            .checkInitElements()
            .stepsPersonalFile.goToPersonalFile(1);
        new PersonalFilePage().checkInitElements()
            .clickActions()
            .clickRelease();

        createGraduationPage.checkInitElements();
        infoAboutOrderPart
            .checkHintLabel()
            .fillFields(orderParameters)
            .clickCancel();
        new CancelWorkNotification()
            .checkNotification()
            .clickCancelBtn();
        createGraduationPage.checkInitElements();
        infoAboutOrderPart

        .clearField(infoAboutOrderPart.getOrderNumberInput())
            .checkIsActiveCreateOrder();
        infoAboutOrderPart
            .fillFormField("orderNumberInput", orderNumber)

```

```

        .clickCreateOrder();

        new
        SuccessNotification().checkCreateOrderMessage(orderNumber);

        graduationPage.checkProjectInitElements();
        infoAboutOrderPart
            .clickActions()
            .checkContextButtons()
            .clickActions()
            .clickOnPencil();

        .checkInitEditOrderElements(InfoAboutOrderPart.OrderDateLabels.GRADU
        ATION.getValue())
            .clearField("orderNumber")
            .clickCancel();

        new CancelWorkNotification()
            .checkNotification()
            .clickCancelBtn();
        orderNumber += "1";
        new
        InfoAboutOrderPart(modalWindows.getModalContainer(), "")
            .checkInputElements()
            .fillFormField("orderNumberInput",
orderNumber);
        modalWindows
            .clickSave();
        infoAboutOrderPart
            .checkEditedOrderNotification()
            .clickActions()
            .clickGraduateOrder();

        modalWindows
            .checkInitConfirmGraduateElements(orderNumber)
            .clickRelease()
            .checkGraduateCheckCard()
            .clickCancel();
        graduationPage.checkRealisedInitElements(orderNumber);
        infoAboutOrderPart
            .clickReject()
            .checkInitRejectElements(orderNumber)
            .selectRandomReason()
            .clickReject();

        new
        SuccessNotification().checkRejectOrderMessage(orderNumber);

        graduationPage.checkInitElements();
        infoAboutOrderPart
            .clickActions()
            .clickDeleteOrder()
            .checkInitDeleteOrderElements(orderNumber)
            .clickDelete();

```

```

        new
        SuccessNotification().checkDeleteOrderMessage(orderNumber);
        graduationPage.checkDeletedInitElements(orderNumber);

        PageSteps.logout();
    }

    @Test(description = "Приказы/Индивидуальный приказ о
переводе из группы в группу/Роль Оператор ПОО")
    @Owner(Authors.SKIRDOVA_VM)
    public void orderTransfer() {
        PageSteps.auth(Users.getUserByName(Users.adm1))
            .checkInitElements();
        CreateTransferInGroupPage createOrderPage = new
        CreateTransferInGroupPage();
        TransferInGroupPage transferInGroupPage = new
        TransferInGroupPage();
        InfoAboutOrderPart infoAboutOrderPart = new
        InfoAboutOrderPart(InfoAboutOrderPart.OrderDateLabels.TRANSFER.getVa
        lue());

        ModalWindows modalWindows = new ModalWindows();
        String orderNumber = "pt-4059";
        String groupName = "19ГД-147";

        Map<String, String> orderParameters = new HashMap<>();
        orderParameters.put("basedOnInput", orderNumber);
        orderParameters.put("orderNumberInput", orderNumber);
        orderParameters.put("orderDate", "01.02.2024");
        orderParameters.put("basedOnDate", "01.02.2024");
        orderParameters.put("graduationDate", "01.02.2024");
        orderParameters.put("toGroupBUP", groupName);

        String fio = "Басимов Дмитрий Фанилевич";
        new NavBar()
            .openRegistriesContingent()
            .searchByFio(fio)
            .checkInitElements()
            .stepsPersonalFile.goToPersonalFile(1);
        new PersonalFilePage().checkInitElements()
            .clickActions()
            .clickInAnotherGroup();

        createOrderPage.checkInitElements();
        infoAboutOrderPart
            .checkHintLabel()
            .fillFields(orderParameters)
            .clickCancel();
        new CancelWorkNotification()
            .checkNotification()
            .clickCancelBtn();
        infoAboutOrderPart

        .clearField(infoAboutOrderPart.getOrderNumberInput())

```

```

        .checkIsActiveCreateOrder();
infoAboutOrderPart
        .fillFormField("orderNumberInput", orderNumber)
        .clickCreateOrder();
new
SuccessNotification().checkCreateOrderMessage(orderNumber);

transferInGroupPage.checkProjectInitElements();
infoAboutOrderPart
        .clickActions()
        .checkContextButtons()
        .clickActions()
        .clickOnPencil()

.checkInitEditOrderElements(InfoAboutOrderPart.OrderDateLabels.GRADU
ATION.getValue())
        .clearField("orderNumber")
        .clickCancel();

new CancelWorkNotification()
        .checkNotification()
        .clickCancelBtn();
orderNumber += "1";
new
InfoAboutOrderPart(modalWindows.getModalContainer(), "")
        .checkInputElement()
        .fillFormField("orderNumberInput",
orderNumber);
modalWindows
        .clickSave();
infoAboutOrderPart
        .checkEditedOrderNotification()
        .clickActions()
        .clickGraduateOrder();

modalWindows
        .checkInitConfirmGraduateElements(orderNumber)
        .clickRelease()
        .checkTransferCheckCard(groupName)
        .clickCancel();

transferInGroupPage.checkRealisedInitElements(orderNumber);
infoAboutOrderPart
        .clickReject()
        .checkInitRejectElements(orderNumber)
        .selectRandomReason()
        .clickReject();
new
SuccessNotification().checkRejectOrderMessage(orderNumber);

transferInGroupPage.checkInitElements();
infoAboutOrderPart
        .clickActions()

```

```

        .clickDeleteOrder()
        .checkInitDeleteOrderElements(orderNumber)
        .clickDelete();

        new
SuccessNotification().checkDeleteOrderMessage(orderNumber);

transferInGroupPage.checkDeletedInitElements(orderNumber);

        PageSteps.logout();
    }

    @Test(description = "Приказы/Индивидуальный приказ о
предоставлении академического/Роль Оператор ПОО")
    @Owner(Authors.SKIRDOVA_VM)
    public void orderAcademicLeave() {
        PageSteps.auth(Users.getUserByName(Users.adm1))
            .checkInitElements();
        CreateAcademicLeavePage createOrderPage = new
CreateAcademicLeavePage();
        AcademicLeavePage academicLeavePage = new
AcademicLeavePage();
        InfoAboutOrderPart infoAboutOrderPart = new
InfoAboutOrderPart();
        ModalWindows modalWindows = new ModalWindows();
        String orderNumber = "pt-4017";

        Map<String, String> orderParameters = new HashMap<>();
        orderParameters.put("basedOnInput", orderNumber);
        orderParameters.put("orderNumberInput", orderNumber);
        orderParameters.put("orderDate", "01.02.2024");
        orderParameters.put("basedOnDate", "01.02.2024");

        String fio = "Басимов Дмитрий Фанилевич";
        new NavBar()
            .openRegistriesContingent()
            .searchByFio(fio)
            .checkInitElements()
            .stepsPersonalFile.goToPersonalFile(1);
        new PersonalFilePage().checkInitElements()
            .clickActions()
            .clickAcademicLeave();

        createOrderPage.checkInitElements();
        infoAboutOrderPart
            .checkHintLabel()
            .fillFields(orderParameters)
            .selectDatePeriod("01.02.2024", "01.03.2024")
            .clickCancel();
        new CancelWorkNotification()
            .checkNotification()
            .clickCancelBtn();
        infoAboutOrderPart

```

```

.clearField(infoAboutOrderPart.getOrderNumberInput())
        .checkIsActiveCreateOrder();
    infoAboutOrderPart
        .fillFormField("orderNumberInput", orderNumber)
        .clickCreateOrder();

    new
SuccessNotification().checkCreateOrderMessage(orderNumber);

    academicLeavePage.checkProjectInitElements();
    infoAboutOrderPart
        .clickActions()
        .checkContextButtons()
        .clickActions()
        .clickOnPencil();

    .checkInitEditOrderElements(InfoAboutOrderPart.OrderDateLabels.GRADU
ATION.getValue())
        .clearField("orderNumber")
        .clickCancel();

    new CancelWorkNotification()
        .checkNotification()
        .clickCancelBtn();
    orderNumber += "1";
    new
InfoAboutOrderPart(modalWindows.getModalContainer(), "")
        .checkInputElements()
        .fillFormField("orderNumberInput",
orderNumber);
    modalWindows
        .clickSave();
    infoAboutOrderPart
        .checkEditedOrderNotification()
        .clickActions()
        .clickGraduateOrder();

    modalWindows
        .checkInitConfirmGraduateElements(orderNumber)
        .clickRelease()
        .checkAcademicLeaveCheckCard()
        .clickCancel();

    academicLeavePage.checkRealisedInitElements(orderNumber);
    infoAboutOrderPart
        .clickReject()
        .checkInitRejectElements(orderNumber)
        .selectRandomReason()
        .clickReject();

    new
SuccessNotification().checkRejectOrderMessage(orderNumber);

    academicLeavePage.checkInitElements();

```

```

        infoAboutOrderPart
            .clickActions()
            .clickDeleteOrder()
            .checkInitDeleteOrderElements(orderNumber)
            .clickDelete();

        new
SuccessNotification().checkDeleteOrderMessage(orderNumber);

academicLeavePage.checkDeletedInitElements(orderNumber);

        PageSteps.logout();
    }

    @Test(description = "Приказы/Индивидуальный приказ о смене
финансирования/Роль Оператор ПОО", enabled = false)
    @Owner(Authors.SKIRDOVA_VM)
    public void orderBudgetFinance() { //TODO нет требуемого
студента! (Нет предусловия)
        PageSteps.auth(Users.getUserByName(Users.adm1))
            .checkInitElements();

        CreateChangeFinanceTypePage createOrderPage = new
CreateChangeFinanceTypePage();
        ChangeFinanceType changeFinanceType = new
ChangeFinanceType();
        InfoAboutOrderPart infoAboutOrderPart = new
InfoAboutOrderPart("", "");
        ModalWindows modalWindows = new ModalWindows();

        String orderNumber = "pt-3982";
        String groupName = "21ГД-29";
        String orderDate = "02.02.2024";
        Map<String, String> orderParameters = new HashMap<>();
        orderParameters.put("orderDate", orderDate);
        orderParameters.put("basedOnDate", "02.02.2024");
        orderParameters.put("transferToStudyGroup", groupName);

        new NavBar()
            .openRegistriesContingent()
            .checkInitElements()
            .filterPart.clickExpandReg();
        new RegistriesContingentPage()
            .selectFinanceAcademicLeaveFilter()
            .filterPart.clickExpandReg();

        int i = 1;
        do {
            new
RegistriesContingentPage().checkInitElements().stepsPersonalFile.goT
oPersonalFile(i);
            new PersonalFilePage()
                .checkInitElements()
                .clickActions()

```

```

        .clickChangeFinanceType();
        i++;
    } while (new PersonalFilePage().isEffectPlanMessage());

    new
CreateAppChangeFinanceTypePage().checkInitElements();
    infoAboutOrderPart
        .checkAppHintLabel()
        .fillFields(orderParameters)
        .clickCancel();
    new CreateApplicationPage()
        .checkCancelModal()
        .clickCancelBtnInCancelModal();
    infoAboutOrderPart
        .clearField(infoAboutOrderPart.getOrderDate())
        .checkIsActiveCreateApp();
    infoAboutOrderPart
        .fillFormField("orderDate", orderDate)
        .clickCreateApp();

    new AppChangeFinanceTypePage().checkInitElements();
    infoAboutOrderPart
        .clickActions()
        .checkAppContextButtons()
        .clickActions()
        .clickOnPencil()
        .checkInitEditAppElements()
        .clearField(infoAboutOrderPart.getOrderDate())
        .clickCancel();

    new CancelWorkNotification()
        .checkNotification()
        .clickCancelBtn();
    orderDate = "01.02.2024";
    new
InfoAboutOrderPart(modalWindows.getModalContainer(), "")
        .checkChangeFinanceTypeElements()
        .fillFormField("orderDate", orderDate);
    modalWindows
        .clickSave();
    infoAboutOrderPart
        .checkEditedAppNotification()
        .clickActions()
        .clickChangeFinanceType();

    createOrderPage.checkInitElements();
    infoAboutOrderPart
        .checkHintLabel()
        .fillFormField("orderNumberInput", orderNumber)
        .clickCancel();
    new CancelWorkNotification()
        .checkNotification()

```

```

        .clickCancelBtn();
        infoAboutOrderPart

.clearField(infoAboutOrderPart.getOrderNumberInput())
        .checkIsActiveCreateOrder();
        infoAboutOrderPart
        .fillFormField("orderNumberInput", orderNumber)
        .clickCreateOrder();

        new
SuccessNotification().checkCreateOrderMessage(orderNumber);

        changeFinanceType.checkProjectInitElements();
        infoAboutOrderPart
        .clickActions()
        .checkContextButtons()
        .clickActions()
        .clickOnPencil()
        .checkInitFinanceEditOrderElements("от")
        .clearField("orderNumber")
        .clickCancel();
        new CancelWorkNotification()
        .checkNotification()
        .clickCancelBtn();
        orderNumber += "1";
        new
InfoAboutOrderPart(modalWindows.getModalContainer(), "")
        .checkInputFinanceElements()
        .fillFormField("orderNumberInput",
orderNumber);
        modalWindows
        .clickSave();
        infoAboutOrderPart
        .checkEditedOrderNotification()
        .clickActions()
        .clickGraduateOrder();

        modalWindows
        .checkInitConfirmGraduateElements(orderNumber)
        .clickRelease();
        PageSteps.logout();
    }

    @Test(description = "Приказы/Индивидуальный приказ
восстановлении/Роль Оператор ПОО")
    @Owner(Authors.SKIRDOVA_VM)
    public void orderRecovery() { //TODO нет требуемого
студента! (Нет предусловия)
        PageSteps.auth(Users.getUserByName(Users.adm1))
        .checkInitElements();

        CreateRecoveryPage createOrderPage = new
CreateRecoveryPage();
        RecoveryPage recoveryPage = new RecoveryPage();

```

```

        InfoAboutOrderPart infoAboutOrderPart = new
InfoAboutOrderPart("", "");
        ModalWindows modalWindows = new ModalWindows();
        RegistryApplicationsPage applicationsPage = new
RegistryApplicationsPage();

        String orderNumber = "pt-4014";
        String orderDate = "02.02.2024";
        Map<String, String> orderParameters = new HashMap<>();
        orderParameters.put("basedOnInput", orderNumber);
        orderParameters.put("orderNumberInput", orderNumber);
        orderParameters.put("orderDate", orderDate);
        orderParameters.put("basedOnDate", "02.02.2024");
        orderParameters.put("graduationDate", "02.02.2024");

        new NavBar()
            .openRegistryApplication()
            .checkInitElements()
            .filterPart.clickExpandReg();
        applicationsPage.cbReceived.setActive();
        applicationsPage.filterPart
            .applyFilters()
            .clickExpandReg();
        applicationsPage
            .clickFio(1);
        new PersonalFilePage()
            .checkAppPage()
            .clickActions()
            .clickRecovery();

        createOrderPage.checkInitElements();
        infoAboutOrderPart
            .checkHintLabel()
            .fillFields(orderParameters)
            .clickCancel();
        new CancelWorkNotification()
            .checkNotification()
            .clickCancelBtn();
        infoAboutOrderPart

        .clearField(infoAboutOrderPart.getOrderNumberInput())
            .checkIsActiveCreateOrder();
        infoAboutOrderPart
            .fillFormField("orderNumberInput", orderNumber)
            .clickCreateOrder();

        new
SuccessNotification().checkCreateOrderMessage(orderNumber);

        recoveryPage.checkProjectInitElements();
        infoAboutOrderPart
            .clickActions()
            .checkContextButtons()
            .clickActions()

```

					ПГУ 1.090301.36.001 ПЗ	Лист
						98
Изм.	Лист	№ докум.	Подпись	Дата		

```

        .clickOnPencil()

        .checkInitEditOrderElements(InfoAboutOrderPart.OrderDateLabels.RECOVER.getValue())

        .clearField("orderNumber")
        .clickCancel();

        new CancelWorkNotification()
            .checkNotification()
            .clickCancelBtn();
        orderNumber += "1";
        new
InfoAboutOrderPart(modalWindows.getModalContainer(), "")
            .checkInputElements()
            .fillFormField("orderNumberInput",
orderNumber);
        modalWindows
            .clickSave();
        infoAboutOrderPart
            .checkEditedOrderNotification()
            .clickActions()
            .clickGraduateOrder();

        modalWindows
            .checkInitConfirmGraduateElements(orderNumber)
            .clickRelease()
            .checkRecoveryCheckCard()
            .clickCancel();
        recoveryPage.checkRealisedInitElements(orderNumber);
        infoAboutOrderPart
            .clickReject()
            .checkInitRejectElements(orderNumber)
            .selectRandomReason()
            .clickReject();

        new
SuccessNotification().checkRejectOrderMessage(orderNumber);

        recoveryPage.checkInitElements();
        infoAboutOrderPart
            .clickActions()
            .clickDeleteOrder()
            .checkInitDeleteOrderElements(orderNumber)
            .clickDelete();

        new
SuccessNotification().checkDeleteOrderMessage(orderNumber);
        recoveryPage.checkDeletedInitElements(orderNumber);

        PageSteps.logout();
    }

    @Test(description = "Приказы/Индивидуальный приказ
зачислении/Роль Оператор ПОО", enabled = false)
    @Owner(Authors.SKIRDOVA_VM)

```

```

        public void orderEnroll() { //TODO нет требуемого студента!
(Нет предусловия)
            PageSteps.auth(Users.getUserByName(Users.adm1))
                .checkInitElements();

            CreateEnrollPage createOrderPage = new
CreateEnrollPage();
            EnrollPage enrollPage = new EnrollPage();
            InfoAboutOrderPart infoAboutOrderPart = new
InfoAboutOrderPart("", "");
            ModalWindows modalWindows = new ModalWindows();
            RegistryApplicationsPage applicationsPage = new
RegistryApplicationsPage();

            String orderNumber = "pt-4014";
            String orderDate = "02.02.2024";
            Map<String, String> orderParameters = new HashMap<>();
            orderParameters.put("basedOnInput", orderNumber);
            orderParameters.put("orderNumberInput", orderNumber);
            orderParameters.put("orderDate", orderDate);
            orderParameters.put("basedOnDate", "02.02.2024");
            orderParameters.put("graduationDate", "02.02.2024");

            new NavBar()
                .openRegistryApplication()
                .checkInitElements()
                .filterPart.clickExpandReg();
            applicationsPage.cbRecommendedForEnrolment.setActive();
            applicationsPage.filterPart
                .applyFilters()
                .clickExpandReg();
            applicationsPage
                .clickFio(1);
            new PersonalFilePage()
                .checkAppPage()
                .clickActions()
                .clickRecovery();

            createOrderPage.checkInitElements();
            infoAboutOrderPart
                .checkHintLabel()
                .fillFields(orderParameters)
                .clickCancel();
            new CancelWorkNotification()
                .checkNotification()
                .clickCancelBtn();
            infoAboutOrderPart

            .clearField(infoAboutOrderPart.getOrderNumberInput())
                .checkIsActiveCreateOrder();
            infoAboutOrderPart
                .fillFormField("orderNumberInput", orderNumber)
                .clickCreateOrder();

```

					ПГУ 1.090301.36.001 ПЗ	Лист
						100
Изм.	Лист	№ докум.	Подпись	Дата		

```

        new
        SuccessNotification().checkCreateOrderMessage(orderNumber);

        enrollPage.checkProjectInitElements();
        infoAboutOrderPart
            .clickActions()
            .checkContextButtons()
            .clickActions()
            .clickOnPencil()

        .checkInitEditOrderElements(InfoAboutOrderPart.OrderDateLabels.ENROL
        L.getValue())

            .clearField("orderNumber")
            .clickCancel();

        new CancelWorkNotification()
            .checkNotification()
            .clickCancelBtn();
        orderNumber += "1";
        new
        InfoAboutOrderPart(modalWindows.getModalContainer(), "")
            .checkInputElements()
            .fillFormField("orderNumberInput",
orderNumber);
        modalWindows
            .clickSave();
        infoAboutOrderPart
            .checkEditedOrderNotification()
            .clickActions()
            .clickGraduateOrder();

        modalWindows
            .checkInitConfirmGraduateElements(orderNumber)
            .clickRelease()
            .checkEnrollCheckCard()
            .clickCancel();
        enrollPage.checkRealisedInitElements(orderNumber);
        infoAboutOrderPart
            .clickReject()
            .checkInitRejectElements(orderNumber)
            .selectRandomReason()
            .clickReject();

        new
        SuccessNotification().checkRejectOrderMessage(orderNumber);

        enrollPage.checkInitElements();
        infoAboutOrderPart
            .clickActions()
            .clickDeleteOrder()
            .checkInitDeleteOrderElements(orderNumber)
            .clickDelete();

        new
        SuccessNotification().checkDeleteOrderMessage(orderNumber);

```

```

        enrollPage.checkDeletedInitElements (orderNumber);

        PageSteps.logout();
    }

    @Test(description = "Приказы/Индивидуальный приказ о
переводе на ЭУП/Роль Оператор ПОО")
    @Owner (Authors.SKIRDOVA_VM)
    public void orderTransferEUP() {
        PageSteps.auth (Users.getUserByName (Users.adm1))
            .checkInitElements();

        CreateTransferToEUPPage createOrderPage = new
CreateTransferToEUPPage();
        TransferToEUPPage transferToEUPPage = new
TransferToEUPPage();
        InfoAboutOrderPart infoAboutOrderPart = new
InfoAboutOrderPart (InfoAboutOrderPart.OrderDateLabels.TRANSFER.getVa
lue());

        ModalWindows modalWindows = new ModalWindows();

        String orderNumber = "pt-4021";
        String orderDate = "02.02.2024";
        Map<String, String> orderParameters = new HashMap<>();
        orderParameters.put ("orderNumberInput", orderNumber);
        orderParameters.put ("basedOnInput", orderNumber);
        orderParameters.put ("orderDate", orderDate);
        orderParameters.put ("basedOnDate", "02.02.2024");
        orderParameters.put ("graduationDate", "02.02.2024");

        String fio = "Малиновский Елеферий Васильевич";
        new NavBar()
            .openRegistriesContingent()
            .searchByFio(fio)
            .checkInitElements()
            .stepsPersonalFile.goToPersonalFile(1);
        new PersonalFilePage().checkInitElements()
            .clickActions()
            .clickToEUP();

        createOrderPage.checkInitElements();
        infoAboutOrderPart
            .checkHintLabel()
            .fillFields (orderParameters)
            .getToGroupEUP().selectRandomOption();
        infoAboutOrderPart
            .clickCancel();
        new CancelWorkNotification()
            .checkNotification()
            .clickCancelBtn();
        infoAboutOrderPart

        .clearField (infoAboutOrderPart.getOrderNumberInput())

```

```

        .checkIsActiveCreateOrder();
    infoAboutOrderPart
        .fillFormField("orderNumberInput", orderNumber)
        .clickCreateOrder();

    new
    SuccessNotification().checkCreateOrderMessage(orderNumber);

    transferToEUPPage.checkProjectInitElements();
    infoAboutOrderPart
        .clickActions()
        .checkContextButtons()
        .clickActions()
        .clickOnPencil()
        .checkInitEditOrderElements("Перевести с")
        .clearField("orderNumber")
        .clickCancel();
    new CancelWorkNotification()
        .checkNotification()
        .clickCancelBtn();
    orderNumber += "1";
    new
    InfoAboutOrderPart(modalWindows.getModalContainer(), "")
        .checkInputFinanceElements()
        .fillFormField("orderNumberInput",
orderNumber);
    modalWindows
        .clickSave();
    infoAboutOrderPart
        .checkEditedOrderNotification()
        .clickActions()
        .clickGraduateOrder();

    modalWindows
        .checkInitConfirmGraduateElements(orderNumber)
        .clickRelease()
        .checkTransferToEUPCheckCard()
        .clickCancel();

    transferToEUPPage.checkRealisedInitElements(orderNumber);
    infoAboutOrderPart
        .clickReject()
        .checkInitRejectElements(orderNumber)
        .selectRandomReason()
        .clickReject();

    new
    SuccessNotification().checkRejectOrderMessage(orderNumber);

    transferToEUPPage.checkInitElements();
    infoAboutOrderPart
        .clickActions()
        .clickDeleteOrder()
        .checkInitDeleteOrderElements(orderNumber)
        .clickDelete();

```

```

        new
        SuccessNotification().checkDeleteOrderMessage(orderNumber);

        transferToEUPPage.checkDeletedInitElements(orderNumber);

        PageSteps.logout();
    }

    @Test(description = "Приказы/Индивидуальный приказ о
переводе на БУП/Роль Оператор ПОО", enabled = false)
    @Owner(Authors.SKIRDOVA_VM)
    public void orderTransferBUP() { //TODO Нет нужных учеников
из предусловия
        PageSteps.auth(Users.getUserByName(Users.adm1))
            .checkInitElements();

        CreateTransferToBUPPage createOrderPage = new
        CreateTransferToBUPPage();
        TransferToBUPPage transferToBUPPage = new
        TransferToBUPPage();
        InfoAboutOrderPart infoAboutOrderPart = new
        InfoAboutOrderPart(InfoAboutOrderPart.OrderDateLabels.TRANSFER.getVa
        lue());

        ModalWindows modalWindows = new ModalWindows();

        String orderNumber = "pt-4021";
        String orderDate = "02.02.2024";
        Map<String, String> orderParameters = new HashMap<>();
        orderParameters.put("orderNumberInput", orderNumber);
        orderParameters.put("basedOnInput", orderNumber);
        orderParameters.put("orderDate", orderDate);
        orderParameters.put("basedOnDate", "02.02.2024");
        orderParameters.put("graduationDate", "02.02.2024");

        String fio = "Моисеева Александра Андреевна";
        new NavBar()
            .openRegistriesContingent()
            .searchByFio(fio)
            .checkInitElements()
            .stepsPersonalFile.goToPersonalFile(1);
        new PersonalFilePage().checkInitElements()
            .clickActions()
            .clickToBUP();

        createOrderPage.checkInitElements();
        infoAboutOrderPart
            .checkHintLabel()
            .fillFields(orderParameters)
            .getToGroupBUP().selectRandomOption();
        infoAboutOrderPart
            .clickCancel();
        new CancelWorkNotification()
            .checkNotification()

```

```

        .clickCancelBtn();
infoAboutOrderPart

.clearField(infoAboutOrderPart.getOrderNumberInput())
        .checkIsActiveCreateOrder();
infoAboutOrderPart
        .fillFormField("orderNumberInput", orderNumber)
        .clickCreateOrder();

new
SuccessNotification().checkCreateOrderMessage(orderNumber);

transferToBUPPage.checkProjectInitElements();
infoAboutOrderPart
        .clickActions()
        .checkContextButtons()
        .clickActions()
        .clickOnPencil()
        .checkInitEditOrderElements("Перевести с")
        .clearField("orderNumber")
        .clickCancel();
new CancelWorkNotification()
        .checkNotification()
        .clickCancelBtn();
orderNumber += "1";
new
InfoAboutOrderPart(modalWindows.getModalContainer(), "")
        .checkInputFinanceElements()
        .fillFormField("orderNumberInput",
orderNumber);
modalWindows
        .clickSave();
infoAboutOrderPart
        .checkEditedOrderNotification()
        .clickActions()
        .clickGraduateOrder();

modalWindows
        .checkInitConfirmGraduateElements(orderNumber)
        .clickRelease()
        .checkTransferToBUPCheckCard()
        .clickCancel();

transferToBUPPage.checkRealisedInitElements(orderNumber);
infoAboutOrderPart
        .clickReject()
        .checkInitRejectElements(orderNumber)
        .selectRandomReason()
        .clickReject();

new
SuccessNotification().checkRejectOrderMessage(orderNumber);

transferToBUPPage.checkInitElements();
infoAboutOrderPart

```

```

        .clickActions()
        .clickDeleteOrder()
        .checkInitDeleteOrderElements(orderNumber)
        .clickDelete();

        new
SuccessNotification().checkDeleteOrderMessage(orderNumber);

transferToBUPPage.checkDeletedInitElements(orderNumber);
        PageSteps.logout();
    }
}

```

					ПГУ 1.090301.36.001 ПЗ	Лист
						106
Изм.	Лист	№ докум.	Подпись	Дата		

Приложение В

(Обязательное)

Автоматизированное функциональное тестирование создания индивидуальных
и групповых приказов

Презентация

					ПГУ 1.090301.36.001 ПЗ	Лист
						107
Изм.	Лист	№ докум.	Подпись	Дата		

Тема работы:
«Автоматизированное функциональное
тестирование создания индивидуальных и
групповых приказов»

Автор работы: студент группы 20ВВП2 Скирдова В.М.

Научный руководитель: к.т.н., доцент кафедры ВТ ПГУ Токарев А.Н.

Пенза, 2024

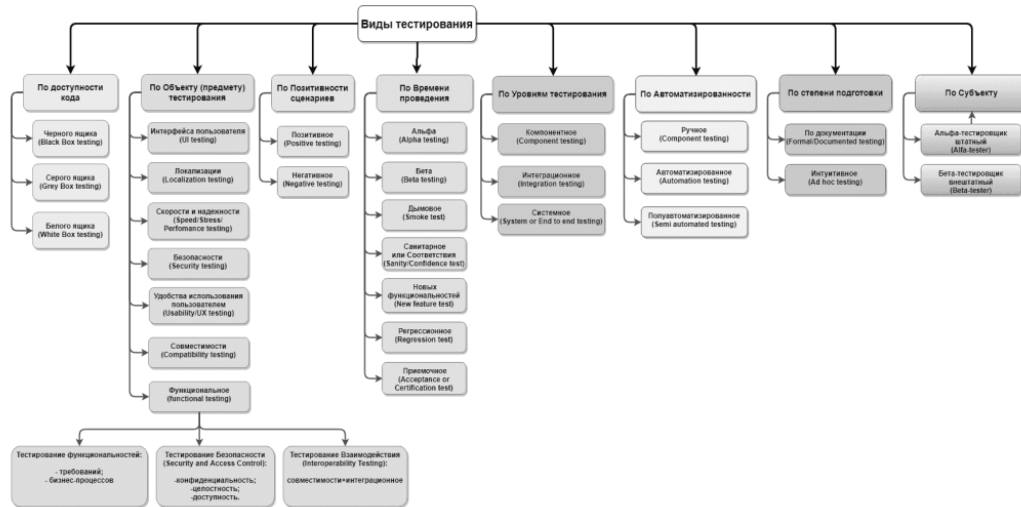
Цель работы и объект работы

Цель работы – разработать функциональное автоматизированное тестирование создания и выпуска индивидуальных и групповых приказов для обучающихся профессионально-технических образовательных учреждений.

Объект работы – web-приложение – автоматизированная информационная система (АИС) «Зачисление в учреждения профессионального образования».

					ПГУ 1.090301.36.001 ПЗ	Лист
						108
Изм.	Лист	№ докум.	Подпись	Дата		

Направления тестирования



Автоматизированное функциональное тестирование

- Функциональное тестирование — вид тестирования, при котором проверяются функциональные требования, согласно спецификации тестируемого программного обеспечения.
- Автоматическое, или автоматизированное тестирование – это способ проверки продукта с помощью специальных программ, называемых автотестами.

Стек технологий



Selenide
CONCISE UI TESTS IN JAVA



Allure Report



TestNG

Диаграмма вариантов использования

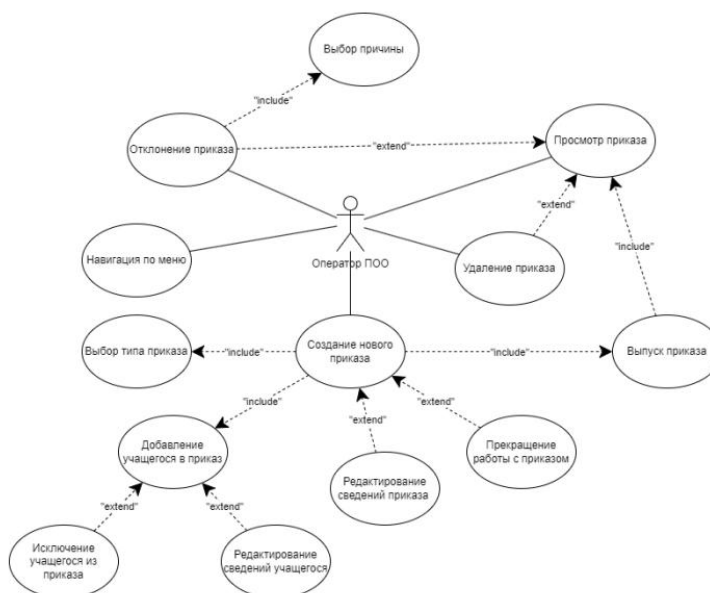


Диаграмма последовательности

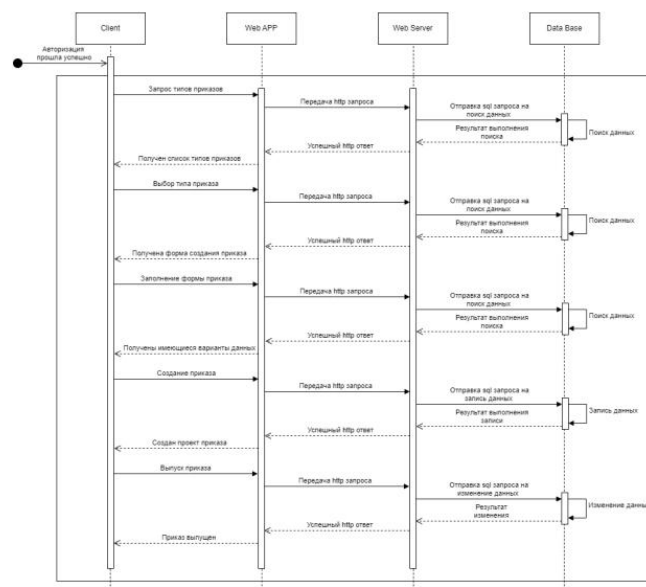
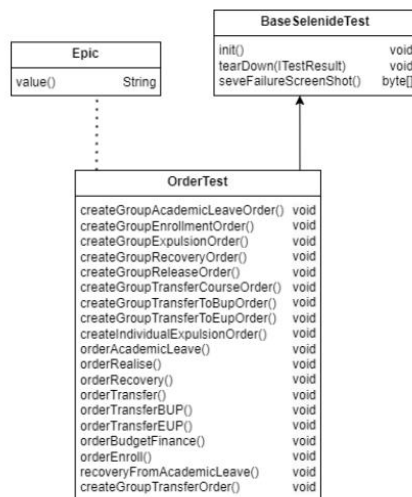
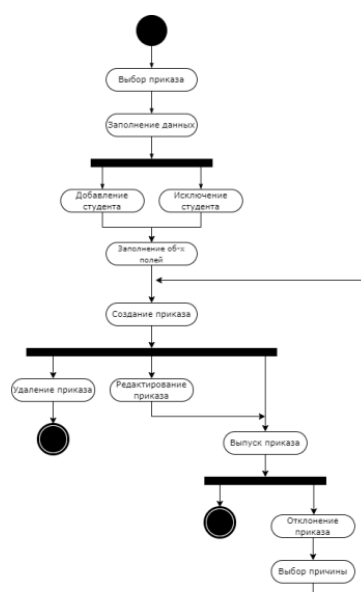
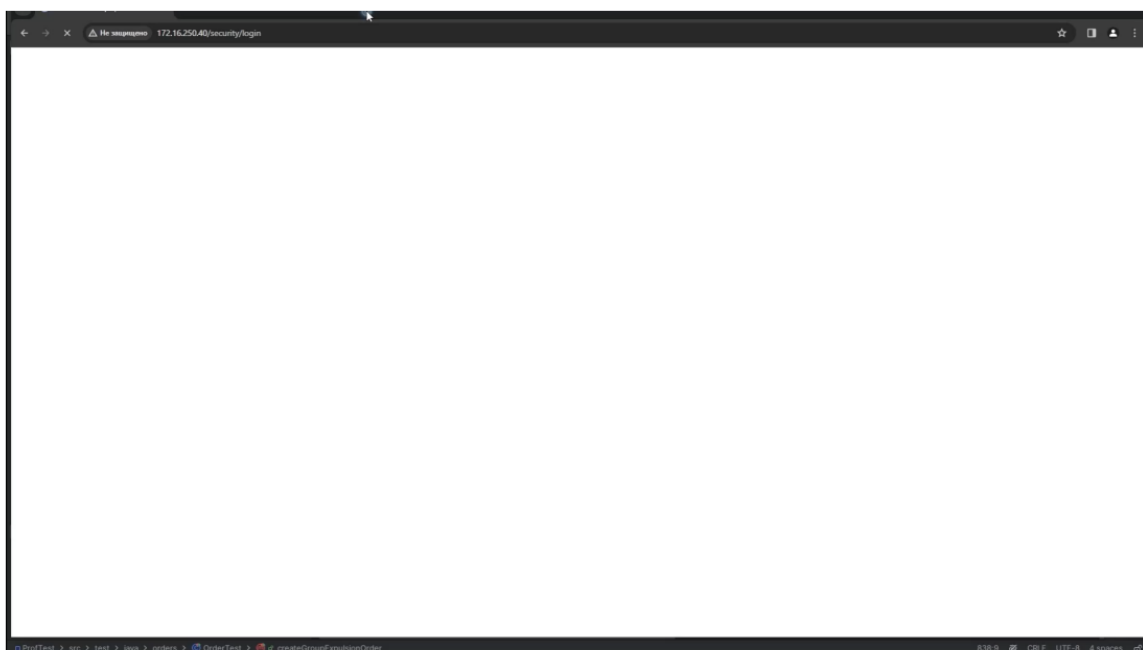


Диаграмма классов







Результаты работы

Suites				
order	name	duration	status	Filter by status
▼ Default Suite				
▼ ProffTest				
▼ orders OrderTest				
✖ #7	Приказы/ Групповой приказ о восстановлении / Роль Оператор ПОО	22s 406ms		
✖ #18	Приказы/ Автомат: приказ о восстановлении из академ отпуска/ Роль Оператор ПОО	30s 370ms		
✔ #8	Приказы/ Групповой приказ о выпуске/ Роль Оператор ПОО	25s 703ms		
✖ #5	Приказы/ Групповой приказ о зачислении/ Роль Оператор ПОО	52s 348ms		
✔ #10	Приказы/ Групповой приказ о переводе на БУ/П/ Роль Оператор ПОО	31s 142ms		
✔ #11	Приказы/ Групповой приказ о переводе на ЭУ/П/ Роль Оператор ПОО	30s 396ms		
✖ #9	Приказы/ Групповой приказ о переводе с курса на курс/ Роль Оператор ПОО	31s 560ms		
✔ #4	Приказы/ Групповой приказ о предоставлении академического отпуска/ Роль Оператор ПОО	28s 654ms		
✔ #6	Приказы/ Групповой приказ об отчислении/ Роль Оператор ПОО	48s 729ms		
✔ #12	Приказы/ Индивидуальный приказ об отчислении/ Роль Оператор ПОО	34s 684ms		
✔ #19	Приказы/Групповой приказ о переводе из группы в группу/ Роль Оператор ПОО	29s 025ms		
✖ #15	Приказы/Индивидуальный приказ восстановления/Роль Оператор ПОО	9s 622ms		
?	Приказы/Индивидуальный приказ зачисления/Роль Оператор ПОО	0s		
!	Приказы/Индивидуальный приказ о выпуске/Роль Оператор ПОО	17s 157ms		
!	Приказы/Индивидуальный приказ о переводе из группы в группу/Роль Оператор ПОО	16s 644ms		
?	Приказы/Индивидуальный приказ о переводе на БУ/П/Роль Оператор ПОО	0s		
✔ #17	Приказы/Индивидуальный приказ о переводе на ЭУ/П/Роль Оператор ПОО	1m 05s		
✖ #13	Приказы/Индивидуальный приказ о предоставлении академического/Роль Оператор ПОО	1m 01s		
?	Приказы/Индивидуальный приказ о смене финансирования/Роль Оператор ПОО	2ms		

Заключение

В ходе выполнения выпускной квалификационной работы были разработаны автоматизированные тесты для проверки web-приложения на языке Java с использованием фреймворков Selenide и TestNG.

СПАСИБО ЗА ВНИМАНИЕ!

					ПГУ 1.090301.36.001 ПЗ	Лист
						114
Изм.	Лист	№ докум.	Подпись	Дата		