Version control

Дата	Кто изменил	Какие изменения внесены
15.11	**Михаил**	**Добавлены разделы 1-4**
18.11	**Артем**	**Добавлены пункты 5.2.1 - 5.2.4**
18.11	**Анна**	**Добавлены пункты 5.2.5 - 5.2.7**
23.11	**Артем**	**Расписаны подробнее процессы в пунктах**
25.11	**Михаил**	**Добавлено описание пунктов 5.2.8 - 5.2.12**
27.11	**Михаил**	**Добавлены разделы 6-7**
28.11	**Анна**	**Добавлены разделы 8-10**
5.12	**Артем**	**Пункт 7.1 Унифицированная схема отчета**
7.12	**Михаил**	**Пункт 7.1 Примеры артефактов**

Test Plan
(План тестирования)

1. Introduction (Введение)

[Введение представляет собой обзор на весь документ в целом и включает в себя следующие разделы - назначение, область применения, определения и аббревиатуры, ссылки и обзор.]

1.1 Purpose

[Укажите назначение данного документа.]

Планирование действий компании, связанных с различными видами и способами тестирования. Распланировать тесты, которые разработчики будут проектировать. Привести описание высокоуровневой модели тестирования внутри компании.

1.2 Scope (Область применения)

[Приведите краткое описание области применения данного документа, к какому(им) проекту(ам) он относится, кем будет использоваться и т.д.]

Документ описывает все виды тестирования системы “Spy+”. Применяется ко всем модулям системы.

1.3 Intended Audience (Предполагаемая аудитория)

[Укажите, для кого написан данный документ и в каких целях он будет использоваться.]
Документ предназначен для разработчиков и менеджера проекта. Разработчики будут

ориентироваться на модель тестирования при проектировании тестов, а менеджеру документ поможет определить сроки проведения и окончания тестирования.

1.4 Document Terminology and Acronyms (Терминология документа)

[Укажите значение терминов и аббревиатур, которые употребляются в данном документе. Возможно указание ссылки на Глоссарий проекта.]

ССЫЛКА на глоссарий, содержащий значение терминов и аббревиатур, которые употребляются в данном документе.

1.5 References (Ссылки)

[Перечислите списком названия документов, на которые ссылаетесь в данном, укажите их источники.]

1. Книга “Корпорация “Бросайте курить”, автор Стивен Кинг;
2. Vision;
3. Software Requirements Specification (SRS);
4. Use Case template (UC);
5. Risk List (RL);
6. SDP (Software Development Plan).
7. Глоссарий
8. SAD

1.6 Document Structure (Структура документа)

[Приведите краткое описание остальных разделов документа.]

1. *Цель и мотивы тестирования - краткое описание того, чего необходимо достичь при помощи проведения тестирования;*
2. *Целевые объекты тестирования - перечисление элементов, работоспособность которых будет проверяться при помощи тестов;*
3. *План тестов - перечисление видов тестов, которые будут входить в план тестирования;*
4. *Подход к тестированию - рекомендованные стратегии для разработки и выполнения обязательных тестов;*
5. *Критерии старта и окончания - условия, при которых можно начать тестирование, а также условия, при которых его можно считать успешно завершенным;*
6. *Ожидаемые результаты тестирования - перечисление артефактов, которые будут созданы в процессе тестирования;*
7. *Необходимое окружение для проведения тестирования - описание ресурсов, необходимых для выполнения плана тестирования;*
8. *Обязанности сотрудников - описание необходимых навыков, знаний и умений у людей, осуществляющих процесс тестирования;*
9. *Управление - описание различных мероприятий по управлению процессом тестирования;*
10. *Сообщение о тестовом покрытии - описание процесса рецензирования результатов тестирования;*
11. *Выявление, избегание и решение проблем - описание механизма учета проблем, возникших в ходе выполнения тестов, а также действий, необходимых для их решения;*
12. *Утверждение плана тестирования - описание процесса утверждения данного плана тестирования, а также списка лиц, участвовавших в нем.*

2. Evaluation Mission and Test Motivation (Цель и мотивы тестирования)

2.1 Background (Справочная информация)

[В данном разделе кратко опишите проект, какие цели он преследует, как будет использоваться, какова его архитектура. Уместны ссылки на другие документы.]

Spy+ - внутренняя информационная система корпорации “Бросайте курить!”, предназначенная для цифровизации основных/ключевых процессов: мониторинга клиентов на основе показаний с датчиков-IOT-устройств, быстрой регистрации зафиксированных нарушений условий договора со стороны клиента, создания и распределения среди сотрудников задач наказания и слежки, составления и изменения графиков сотрудников, а также создания/подписания договоров с клиентами в онлайн формате. Более подробно с функциональностью системы можно ознакомиться в документах, приведенных в пункте 1.5.

2.2 Evaluation Mission (Цели тестирования)

[В данном разделе укажите, с какой целью проводится тестирование проекта. Например, удовлетворить заказчика, найти как можно больше ошибок до окончательного завершения разработки, выявить самые главные проблемы системы и т.д.]

Цели, которых мы хотим достичь путем проведения полноценного тестирования системы:

1) *Получение гарантий корректной работы системы в целом (интеграционное тестирование) и/или ее отдельных частей (модульное тестирование);*
2) *Поиск возможных ошибок, предотвращение появления багов в итоговом продукте;*

2.3 Test Motivators (Мотивы (?) тестирования)

[Укажите, какие элементы будут служить источником информации для тестирования - функциональные и нефункциональные требования, ограничения системы, риски и т.д.]

1) *SRS - содержит перечень всех функций и модулей и служит основой для создания модульных и интеграционных тестов;*
2) *Use Case - описывает ключевые бизнес-процессы и определяет правильную последовательность пользовательских действий и ожидаемые результаты тест кейсов.*
3) *Vision - контекст и проблематика проекта. Определяет ожидаемую ценность проекта, а также показатели, на которые необходимо ориентироваться при определении успешности реализации.*
4) *SDP - устанавливает итерации, этапы и сроки, а значит определяет, когда начинаются и заканчиваются циклы тестирования*
5) *Risk List - определяет потенциальные угрозы, которые могут помешать успешному завершению проекта. На основе этих данных строятся отдельные тестовые сценарии для верификации устойчивости и способности системы реагировать на ошибки и восстанавливаться после инцидентов.*

3. Target Test Items (Целевые объекты тестирования)

[Перечислите объекты тестирования, т.е. что именно будет проверяться с помощью тестов.]

1) *Веб-интерфейс системы (Frontend);*
2) *Серверная часть (Backend);*
3) *Исходный код системы;*

4) *Интеграция с IOT-устройствами;*
5) *База данных (PostgreSQL);*

4. Outline of Planned Tests (План тестов)

[В данном разделе перечислите все виды тестов, которые будут включены в процесс тестирования. Кратко опишите, что они будут проверять.]

4.1 Data and Database Integrity Testing
- *Доступность БД*
- *Корректность структуры данных*
- *Корректность выполнения различных операций над записями*

4.2 Function Testing
- *Проверка функциональности и работоспособности системы в целом*

4.3 Business Cycle Testing
- *Проверка на соответствие бизнес процессам*
- *Проверка на отсутствие ошибок и логических несоответствий*

4.4 User Interface Testing
- *Корректность работы элементов интерфейса*
- *Корректность отображения данных*
- *Удобство и предсказуемость пользовательского взаимодействия*

4.5 Performance Profiling
- *Качества использования ресурсов системой*
- *Соответствия производительности системы заявленным в документации целевым параметрам*
- *Стабильность и равномерность работы системы*

4.6 Load Testing
- *Поведение системы при увеличении количества одновременных запросов к системе*
- *Устойчивость системы в условиях высокой нагрузки*

4.7 Stress Testing
- *Работоспособность системы в условиях, превышающих номинальные нагрузки*

4.8 Volume Testing
- *Работоспособность системы при большом количестве записей в БД*

4.9 Security and Access Control Testing
- *Правильность прав доступа, отсутствия критических уязвимостей в системе*

5. Test Approach (Подход к тестированию)

[Данный раздел представляет рекомендованные стратегии для разработки и выполнения обязательных тестов. Не все виды тестов обязательно должны быть реализованы.]

5.1 Testing Techniques and Types (Техники тестирования)

[Для описания каждой из используемых техник тестирования рекомендуется воспользоваться следующей таблицей:]

Technique Objective: (Цель)	*[В чем состоит цель данного типа тестов, что он проверяет]*
Technique: (Описание процесса)	*[Пошаговое подробное описание процесса выполнения тестов]*

Oracles: (Источники)	*[На какой документ/элемент системы опираются тесты для проверки результата выполнения.]*
Required Tools: (Инструменты)	*[Инструменты, необходимые для проведения теста - сторонние программы, необходимое окружение пользователя и т.д.]*
Success Criteria: (Критерий успеха)	*[Опишите условия, при которых данный тип тестов считается пройденным.]*

5.2.1 Data and Database Integrity Testing (Тестирование базы данных)

Technique Objective: (Цель)	*[В чем состоит цель данного типа тестов, что он проверяет]* *Цель тестов заключается в том, чтобы убедиться, что любая подсистема, использующая хранение данных, обеспечивает корректность и согласованность информации при выполнении всех видов операций: создании, чтении, обновлении и удалении, а также при ошибках и/или прерывании транзакций.* *Проверяются:* *1) Корректность структуры данных и всех обязательных полей* *2) Корректность связей между сущностями (1:1, 1:N, M:N)* *3) Корректность работы транзакций, отката транзакций при ошибках* *4) Предсказуемость поведения базы данных при массовых изменениях или удалениях связанных сущностей*
Technique: (Описание процесса)	*[Пошаговое подробное описание процесса выполнения тестов]* 1) *Зафиксировать основные сценарии для тестирования данных* a) *Вставка корректных и некорректных данных (соответствие схеме бд)* b) *Проверка работы ограничений, индексов* i) *FK* ii) *UNIQUE* iii) *NULL / NOT NULL* c) *Удаление данных, связанных между собой (проверка работы триггеров и каскадных действий)* 2) *Подготовка SQL-скриптов для каждого подготовленного сценария.* 3) *Запуск скриптов через pgAdmin* 4) *Подведение итогов. Проверка:* a) *Наличия ошибок в случае ввода заведомо некорректных данных* b) *Отсутствия ошибок в случае корректных данных* c) *Корректность отката изменений при моделировании ошибки в ходе работы транзакции* 5) *Фиксация результатов тестирования в отчете формата .md*

Oracles: (Источники)	*[На какой документ/элемент системы опираются тесты для проверки результата выполнения.]* *SAD (модель данных), SAD (db diagram), UC (1,3,4 - операции с бд)*
Required Tools: (Инструменты)	*[Инструменты, необходимые для проведения теста - сторонние программы, необходимое окружение пользователя и т.д.]* *DBeaver, SQL-скрипты.*
Success Criteria: (Критерий успеха)	*[Опишите условия, при которых данный тип тестов считается пройденным.]* *Данные корректно создаются/обновляются/удаляются, все операции заканчиваются с ожидаемым результатом, нет потерь и нарушений связей.*

5.2.2 Function Testing (Функциональное тестирование)

Technique Objective: (Цель)	*[В чем состоит цель данного типа тестов, что он проверяет]* *Проверка функциональности всех критически важных модулей согласно SRS, UC и SAD.* *Проверяет конкретные функции системы, которые заложены логикой его работы, то есть работает ли система в целом.*
Technique: (Описание процесса)	*[Пошаговое подробное описание процесса выполнения тестов]* 1) *Подготовка тестовых данных (создание нужных учетных записей, заполнение основных таблиц)* 2) *Определить проверяемую функцию (н.п. создание договора, регистрация нарушения, вход в систему и т.д.)* 3) *Сформулировать входные условия (определить, что должно быть доступно перед тестом (н. п. пользователь зарегистрирован в системе, договор не создан))* 4) *Выполнения основного функционального действия (вызов нужной ручки и/или выполнение действия в UI системы)* 5) *Проверка результата - сравнить ожидаемый результат с реальным.* 1) *Определить основные функции каждого модуля, отвечающие за соответствие системы ее функциональным требованиям* 2) *На основе подготовленного списка, подготовить тестовые сценарии для каждого задействованного API- эндпоинта. Сюда входит:* a) *Определить проверяемое действие (создание договора, регистрация нарушения, изменение расписания и т.д.)* b) *Фиксация входных условий и характеристик окружения:* i) *Какие данные должны быть созданы перед тестом* ii) *Какие сервисы должны быть подняты/замоканы*

	iii) *Какие условия должны быть соблюдены (пользователь зарегистрирован в системе, договор не создан)* c) *Выполнение основного функционального действия (вызов конкретной ручки и/или выполнение действия в пользовательском интерфейсе системы)* d) *Валидация результата – сверка полученного результата с ожидаемым, анализ ошибок и предупреждения* 3) *Запуск подготовленных тестовых сценариев* 4) *Формирование отчета о результатах тестирования в формате .txt файла*
Oracles: (Источники)	*[На какой документ/элемент системы опираются тесты для проверки результата выполнения.]* *SRS (3.1 - функциональные требования)* *UC (1-5 - сценарии использования системы)* *SAD*
Required Tools: (Инструменты)	*[Инструменты, необходимые для проведения теста - сторонние программы, необходимое окружение пользователя и т.д.]* *Postman/curl - отправка запросов;* *Swagger UI - отправка запросов, понимание формата запросов / ответов;* *TablePlus - проверка состояния данных в бд.*
Success Criteria: (Критерий успеха)	*[Опишите условия, при которых данный тип тестов считается пройденным.]* *Все функции работают в соответствии с описанием (все проверенные функции выдают корректный результат при корректных входных данных). Непредусмотренные входные данные корректно обрабатываются, все проверки валидации возвращают ожидаемые коды ошибок и сообщения. Состояние БД соответствует ожидаемому, отсутствуют нарушения связи между сущностями, все FK соблюдаются. Системные логи не содержат некорректных/неожиданных ошибок/паник.*

5.2.3 Business Cycle Testing (Тестирование бизнес-цикла)

Technique Objective: (Цель)	*Проверяет работу системы в реальных сценариях бизнеса (соответствие бизнес-процессам, отсутствие ошибок и логических несоответствий)*
Technique: (Описание процесса)	*[Пошаговое подробное описание процесса выполнения тестов]* 1) *Зафиксировать тестируемые бизнес-циклы, а также конкретные use-кейсы, из которых они состоят.* 2) *Для каждого тестируемого сценария выполнить действия:*

	a) *Определить роли, которыми должен обладать участник тестового сценария (н.п. клиент, менеджер, сотрудник слежки и т.д.)* b) *Подготовить тестовое окружение и тестовые данные - создать необходимые учетные записи и сущности системы (н.п. активных/неактивных договоров)* c) *Подготовить Selenium скрипты, имитирующие определенный набор пользовательских действий в интерфейсе системы (опционально, для особо крупных сценариев. Может быть заменено ручным тестированием)* d) *Запустить подготовленные скрипты - выполнить шаги Use Case в определенной последовательности, используя интерфейс или API-вызовы.* 3) *В процессе тестирования необходимо осуществлять:* a) *Проверку промежуточных состояний системы - необходимо убедиться, что каждая фаза процесса тестирования приводит систему в определенное ожидаемое состояние* b) *Проверку реакции системы на отклонение входных данных от нормы - как она реагирует на изменение последовательности шагов тестирования или вводе заведомо ложных данных. В таких случаях система должна возвращать корректную ошибку, не останавливая работу других модулей системы.* 4) *Провалидировать итоговые данные и состояние системы - проверить базу данных и логи на предмет отсутствия ошибок и несогласованности данных (например, все договоры создались привязанными к клиентам и менеджерам)* 5) *Зафиксировать результаты тестирования в файле формата .md*
Oracles: (Источники)	*[На какой документ/элемент системы опираются тесты для проверки результата выполнения.]* *UC (1-5 - сценарии использования системы)* *SRS (3.1)* *SAD* *Vision (4.2 & 5)*
Required Tools: (Инструменты)	*[Инструменты, необходимые для проведения теста - сторонние программы, необходимое окружение пользователя и т.д.]* *Postman/curl* *Selenium - автоматизация сценариев, связанных с UI* *DBeawer/TablePlus*
Success Criteria: (Критерий успеха)	*[Опишите условия, при которых данный тип тестов считается пройденным.]*

	Все шаги сценария выполняются в правильной последовательности, а промежуточные статусы и конечный результат соответствуют UC и SRS. *При ошибках и/или нарушении последовательности действий система корректно разрешает ситуацию конфликта, при необходимости откатывает изменения и не нарушает состояние консистентности БД.*

5.2.4 User Interface Testing (Тестирование интерфейса)

Technique Objective: (Цель)	*[В чем состоит цель данного типа тестов, что он проверяет]* *Цель - убедиться, что все элементы интерфейса работают корректно, отображают данные в соответствии с требованиями и обеспечивают удобное, предсказуемое взаимодействие для всех категорий пользователей*
Technique: (Описание процесса)	*[Пошаговое подробное описание процесса выполнения тестов]* *1) Подготовка тестовой среды:* *a) развертывание frontend + backend* *b) развертывание bd + создание тестовых пользователей* *2) Проверка структуры интерфейса* *a) Открытие всех ключевых страниц* *b) Проверка корректного отображения всех элементов интерфейса и подписей к ним* *i) Все кнопки находятся в пределах пользовательской доступности* *ii) Надписи не наезжают друг на друга, а также на другие элементы интерфейса* *iii) Изображения, таблицы, календари и прочие графические элементы прогружаются полностью* *c) Проверка адаптивности экрана* *3) Проверка интерактивных элементов* *a) Кнопки, поля ввода, фильтры, выпадающие списки работают и выполняют предназначенные им функции* *b) Работа кнопок и их состояние соответствует контексту (например, кнопка “подписать” недоступна, если договор уже был подписан* *4) Проверка реакций и обновлений* *a) После определенных действий пользователя интерфейс обновляется без перезагрузки страницы* *b) Индикаторы загрузки, ошибок и уведомлений отображаются корректно* *c) Проверить push-уведомления в режиме реального времени* *5) Проверка корректности отображения данных* *a) Данные, полученные от API отображаются корректно* *b) Проверка пагинации, сортировки и фильтрации списков и таблиц* *c) Сравнить отображаемые значения со значениями из бд* *6) Проверка доступности и UX*

	a) *Проверка навигации с клавиатуры (enter, tab, arrows, …)* b) *Орфография, размер текста, читаемость (все надписи должны быть доступны к прочтению без применения дополнительных инструментов (по типу экранной лупы))* c) *Все кнопки и поля, а также их назначение должны быть интуитивно понятны (подсказки, иконки, графические выделения)* d) *Проверить наличие, читаемость и интуитивность сообщений об ошибках* 7) *Проверка сценариев ролей* a) *Доступные действия соответствуют роли* b) *Отображаемые разделы и данные соответствуют роли* 8) *Проверка на устойчивость* a) *Поведение при потере соединения с сервером (данные отображаются, но действия недоступны)* b) *Возврат к работе после восстановления связи*
Oracles: (Источники)	*[На какой документ/элемент системы опираются тесты для проверки результата выполнения.]* *SAD* *SRS 3.2 Usability – описание требований к удобству и визуальному представлению интерфейса* *UC (поведение интерфейса в некоторых сценариях)*
Required Tools: (Инструменты)	*[Инструменты, необходимые для проведения теста - сторонние программы, необходимое окружение пользователя и т.д.]* 1) *Браузер Chrome - последняя версия* 2) *Инструменты автоматизации действий над UI – Selenium* 3) *Chrome DevTools* 4) *Postman/SwaggerUI* 5) *DBeawer/TablePlus* 6) *Jira - для фиксации найденных артефактов и дефектов*
Success Criteria: (Критерий успеха)	*[Опишите условия, при которых данный тип тестов считается пройденным.]* *Все элементы интерфейса отображаются корректно на всех разрешениях экрана. Все интерактивные элементы выполняют свои функции без ошибок. Данные на экране при любом состоянии системы соответствуют ответам API и состоянию бд. Все роли пользователей видят только разрешенные разделы и данные и могут выполнять только ряд разрешенных для этой роли действий.*

5.2.5 Performance Profiling (Тестирование производительности)

Technique Objective:	*[В чем состоит цель данного типа тестов, что он проверяет]*

(Цель)	*Определить и оценить, как система использует ресурсы при выполнении типичных операций и базовом взаимодействии между компонентами* *Выявить узкие места и подтвердить, что производительность системы соответствует целевым параметрам, зафиксированным в проектной документации* *Проанализировать стабильность, равномерность и эффективность работы архитектуры при стандартных сценариях*
Technique: (Описание процесса)	*[Пошаговое подробное описание процесса выполнения тестов]* *1) Подготовка тестовой среды* *a) Развернуть систему в изолированной среде* *b) Подготовить отдельный экземпляр базы данных* *i) Очистить все таблицы и загрузить минимальный объем данных (baseline)* *c) Запустить инструменты мониторинга* *i) Поднять Prometheus который будет читать метрики backend-а (/internal/metrics)* *ii) Поднять Grafana и подключить Prometheus в качестве источника данных о метриках* *iii) Убедиться, что собираются и визуализируются ключевые метрики* *(1) cpu_total* *(2) gc_metrics* *(3) latency, http_count, …* *2) Определить список эталонных операций, которые будут тестироваться в этом виде тестирования (н.п. создание договора, получение расписания сотрудника, регистрация нарушения и тд)* *3) Настройка тестовых сценариев в JMeter:* *a) Для каждой операции создается отдельный Thread Group:* *i) Method + body + header* *ii) Параметры нагрузки (threads, loop-count)* *4) Запуск тестов. Для каждого сценария:* *a) Фиксируем* *i) Среднее время отклика* *ii) max* *b) Смотрим и проверяем на соответствие требованиям* *i) CPU, RAM, Avg Response Time (Grafana & Prom)* *5) Подготовить отчет в формате .md, содержащий все метрики и сравнительные таблицы запусков, а также выводы по ним.*
Oracles: (Источники)	*[На какой документ/элемент системы опираются тесты для проверки результата выполнения.]* *SRS 2.1 Productional Functions – описание ключевых операций, по которым измеряется производительность* *Vision 9.3 Performance Requirements – содержит ожидаемые характеристики производительности системы, стабильность при стандартных нагрузках, предсказуемое время отклика*

Required Tools: (Инструменты)	*[Инструменты, необходимые для проведения теста - сторонние программы, необходимое окружение пользователя и т.д.]* *Apache JMeter – нагрузочные сценарии для API* *Prometheus + Grafana – сбор и визуализация различных метрик приложения и состояния системы*
Success Criteria: (Критерий успеха)	*[Опишите условия, при которых данный тип тестов считается пройденным.]* *Время отклика API и запросов к бд остается стабильным при повторных итерациях и не демонстрирует деградации* *Ресурсы системы используются равномерно, нигде не наблюдается резких скачков нагрузки или зависаний* *Все операции выполняются без таймаутов*

5.2.6 Load Testing (Нагрузочное тестирование)

Technique Objective: (Цель)	*[В чем состоит цель данного типа тестов, что он проверяет].* *Проверить поведение системы при увеличении количество одновременных действий пользователей, запросов к API и событий от IOT-устройств.* *Проверить, выдерживает ли архитектура системы нагрузку, соответствующую предполагаемому масштабу эксплуатации, описанному в проектных документах, а также зафиксировать слабые места системы, условия, в которых начинают проявляться замедления или сбои*
Technique: (Описание процесса)	*[Пошаговое подробное описание процесса выполнения тестов]* 1) *Подготовка тестовой среды (аналогично 5.2.5)* 2) *Определение сценариев нагрузки в утилите JMeter с предельно допустимой нагрузкой.* 3) *Выполнение сценариев JMeter.* 4) *Мониторинг поведения системы – отслеживать ответы API, появление непредвиденных ошибок. Просмотр логов сервисов, брокеров и базы данных. Фиксировать любые ошибки, задержки или блокировки.* 5) *Завершение тестов – после прекращения нагрузки убедиться, что система корректно освобождает задействованные ресурсы и восстанавливает нормальное состояние (нет зависших процессов и утечек памяти)*
Oracles: (Источники)	*[На какой документ/элемент системы опираются тесты для проверки результата выполнения.]* *SRS 2.1 Production Functions – описание функций, которые должны корректно выполняться даже при одновременном обращении нескольких пользователей* *Vision 9.3 Performance Requirements – описывает целевые параметры производительности системы, используемые для проверки того, что система сохраняет стабильность при типовой и повышенной нагрузке.*

Required Tools: (Инструменты)	*[Инструменты, необходимые для проведения теста - сторонние программы, необходимое окружение пользователя и т.д.]* *Apache JMeter / k6/ Locust – инструменты нагрузочного тестирования* *Prometheus / Grafana* *Jira / Excel*
Success Criteria: (Критерий успеха)	*[Опишите условия, при которых данный тип тестов считается пройденным.]* *Система сохраняет работоспособность при ожидаемой нагрузке, соответствующей масштабам использования, описанным в проектной документации. При увеличении количества запросов не происходит неконтролируемого роста количества ошибок или отказа/падения сервисов. Все критические операции выполняются без дублирования и потери данных. Время отклика и стабильность системы не страдают от резкой деградации при длительной нагрузке. После прекращения тестов и уменьшения нагрузки система корректно освобождает ресурсы, возвращаясь к своему базовому состоянию без ошибок. В логах всех элементов системы отсутствуют неизвестные ошибки/исключения.*

5.2.7 Stress Testing (Стрессовое тестирование)

Technique Objective: (Цель)	*[В чем состоит цель данного типа тестов, что он проверяет]* *Проверить работоспособность системы в условиях, превышающих номинальные нагрузки* *Определить, сохранит ли система корректное поведение и консистентность данных при нехватке ресурсов, большом количестве одновременных запросов, сбоях сети и частых обращениях к бд.*
Technique: (Описание процесса)	*[Пошаговое подробное описание процесса выполнения тестов]* 1) *Подготовка тестовой среды (аналогично 5.2.6)* 2) *Определить контрольную точку нормальной нагрузки (на основании результатов пункта 5.2.6)* 3) *Формирование стрессовых сценариев (резкий прирост числа запросов к API в 2-3 раза по отношению к номинальному, рост числа выполнений операций, содержащих внутри себя поход в базу данных, рост объема данных, получаемых с IOT-устройств и т.д.)* 4) *Наблюдение и фиксация поведения* a) *Отслеживание время отклика API, частоту ошибок, задержки при записи и походе в базу данных* b) *Проверка типов появляющихся ошибок (timeout, out-of-memory, deadlock и прочие)* c) *Отслеживание факта сохранения данных при сбоях (наблюдается ли потеря данных)*

	5) Завершение тестов (аналогично пункту 5.2.6)
Oracles: (Источники)	*[На какой документ/элемент системы опираются тесты для проверки результата выполнения.]* *SRS 2.1 Production Functions – ключевые функции, которые должны сохранять корректность даже при аномальных нагрузках и сбоях* *Vision 9.3 Performance Requirements*
Required Tools: (Инструменты)	*[Инструменты, необходимые для проведения теста - сторонние программы, необходимое окружение пользователя и т.д.]* *JMeter / k6 / Locust* *Prometheus + Grafana*
Success Criteria: (Критерий успеха)	*[Опишите условия, при которых данный тип тестов считается пройденным.]* *Система не разрушается при перегрузке и корректно деградирует (частичные ошибки при которых отсутствует потеря данных)* *После прекращения нагрузки все сервисы восстанавливают свою работу без ручного вмешательства.* *Все данные в бд остаются целостными и согласованными.* *В логах отсутствуют критические ошибки, паники/fatal.*

5.2.8 Volume Testing (Объемное тестирование)

Technique Objective: (Цель)	*[В чем состоит цель данного типа тестов, что он проверяет]* *Оценить производительность приложения при увеличении объема данных в базе данных для выявления возможных узких мест, проблем производительности и оценки поведения системы при больших объемах информации.*
Technique: (Описание процесса)	*[Пошаговое подробное описание процесса выполнения тестов]* *Определение часто используемых таблиц, выявление наиболее узких мест:* ● *С помощью логирования запросов или мониторинга производительности PostgreSQL (pg_stat_statements) определить набор таблиц, которые используются в 80% всех запросов в рамках всех бизнес-кейсов* ● *Подготовка SQL-скриптов для записи большого объема данных в определенные ранее таблицы.*

	Для определенных на предыдущем этапе таблиц подготовить скрипты для записи тестовых данных в большом объеме. Возможно использование генераторов тестовых данных. Заполнение определенных таблиц тестовыми данными. • *Выполнение SQL-запросов, участвующих в бизнес-сценариях, анализ производительности*
Oracles: (Источники)	*[На какой документ/элемент системы опираются тесты для проверки результата выполнения.]* *SAD, SRS*
Required Tools: (Инструменты)	*[Инструменты, необходимые для проведения теста - сторонние программы, необходимое окружение пользователя и т.д.]* *DBeaver*
Success Criteria: (Критерий успеха)	*[Опишите условия, при которых данный тип тестов считается пройденным.]* *При увеличении объема данных, время операций чтения, записи и обновления в БД не превышает установленного лимита*

5.2.9 Security and Access Control Testing (Тестирование безопасности и прав доступа)

Technique Objective: (Цель)	*[В чем состоит цель данного типа тестов, что он проверяет]* • *Проверка 3rd party библиотек а также исходного кода на предмет уязвимостей* • *Проверка правильности реализации прав доступа.*
Technique: (Описание процесса)	*[Пошаговое подробное описание процесса выполнения тестов]* 1. *Подготовка тестовой среды* 2. *Проверить правильность ролевого доступа* a. *Определить список ролей в системе* b. *Определить RestAPI ресурсы и набор ролей, у которых должен быть доступ к каждому из ресурсов* c. *Развернуть серверную часть системы: БД и веб-сервер* d. *Создать тестовые учетные записи для каждой из ролей* e. *С помощью Postman/SwaggerUI для каждого ресурса проверить корректность доступа определенного набора ролей, предоставляя различные credentials* 3. *С помощью OWASP ZAP провести сканирование равзернутого веб-приложения* a. *Зафиксировать полученные предупреждения*
Oracles: (Источники)	*[На какой документ/элемент системы опираются тесты для проверки результата выполнения.]*

	SRS
Required Tools: (Инструменты)	*[Инструменты, необходимые для проведения теста - сторонние программы, необходимое окружение пользователя и т.д.]* *OWASP ZAP*
Success Criteria: (Критерий успеха)	*[Опишите условия, при которых данный тип тестов считается пройденным.]* *Корректный ролевой доступ к ресурсам системы.*

5.2.10 Failover and Recovery Testing (Тестирование на отказ и восстановление)
Тестирование на отказ и восстановление не проводится: система работает на одном узле, не является кластерной и отказоустойчивой.

5.2.11 Configuration Testing (Тестирование конфигурации)
Тестирование конфигурации не проводится: серверная часть будет развернута на одном оборудовании с единой конфигурацией Разработчиком

5.2.12 Installation Testing (Тестирование процесса установки)
Тестирование процесса установки не проводится ввиду отсутствия установки: для пользователей сервис представляет из себя веб-приложение, которое не требует установки какого-либо дополнительного ПО на клиентское устройство, помимо веб-браузера.

6. Entry and Exit Criteria (Критерии старта и окончания)

6.1. Test Plan Entry Criteria (Критерий старта)

[Укажите условие, при котором можно начать процесс тестирования]

Тестирование начинается при появлении новых объектов тестирования, например, исходного кода, реализующего некоторую функциональность, для которого применяется unit тестирование.

6.2 Test Plan Exit Criteria (Критерий окончания)

[Укажите условие, при котором процесс тестирования считается оконченным.]

- Тестирование заканчивается после соответствия системы указанным требованиям и успешном выполнении use cases после окончания разработки требуемых функций
- Обнаружены критические ошибки, которые блокируют дальнейшее тестирование

6.3 Suspension and Resumption Criteria (Критерий паузы и возобновления)

[Укажите условие, при котором необходимо приостановить процесс тестирования и при котором продолжить.]

Тестирование приостанавливается когда текущий функционал уже протестирован, но новый ещё не реализован. Тестирование возобновляется при появлении новой версии системы.

7. Deliverables (Ожидаемые результаты тестирования)

[В данном разделе перечислите артефакты, которые будут созданы в процессе тестирования.]

7.1 Test Evaluation Summaries (Результаты выполнения тестов)

[Опишите формат и содержание результатов выполнения тестирования]

Формат отчета представляет собой документ по унифицированной схема:

1) Общая информация (введение)
    a) Вид тестирования
    b) Дата проведения тестирования
    c) версия/сборка/тег/ветка
    d) Исполнитель
2) Охваченная область (что было протестировано)
    a) Модули/логика/экраны/бизнес-логика
    b) Объем данных
    c) профили нагрузки (при наличии)
    d) роли пользователей
3) Использованные инструменты и поднятое окружение
4) Сценарии тестирования
    a) Название и формулировка
    b) Алгоритм
    c) Ожидаемый результат
    d) Фактический результат
    e) Итоговый статус сценария
    f) Артефакты и комментарии
        i) Логи
        ii) Скриншоты
        iii) Графики
        iv) Краткие комментарии тестировщика
5) Метрики (свои для каждого вида тестирования)
    a) Функциональные тесты - время отклика и состояния системы
    b) Нагрузочные тесты - latency, throughput, errors, errors by endpoint, CPU, RAM
    c) UI - время отклика, рендера страниц и перехода между ними, apdex
    d) тестирование безопасности - severity найденных уязвимостей
    e) БД - время выполнения запросов
6) Выводы и рекомендации
    a) Общий статус (успешно/провал)
    b) Основные риски
    c) группы проблем
    d) рекомендации по исправлению

Артефакты тестирования будут отличаться в зависимости от вида тестирования

1) **Ручное тестирование БД, UI, API**

Артефактом тестирования будет таблица со списком тестовых сценариев и корректностью результата их прохождения, а также информации о возникших ошибках

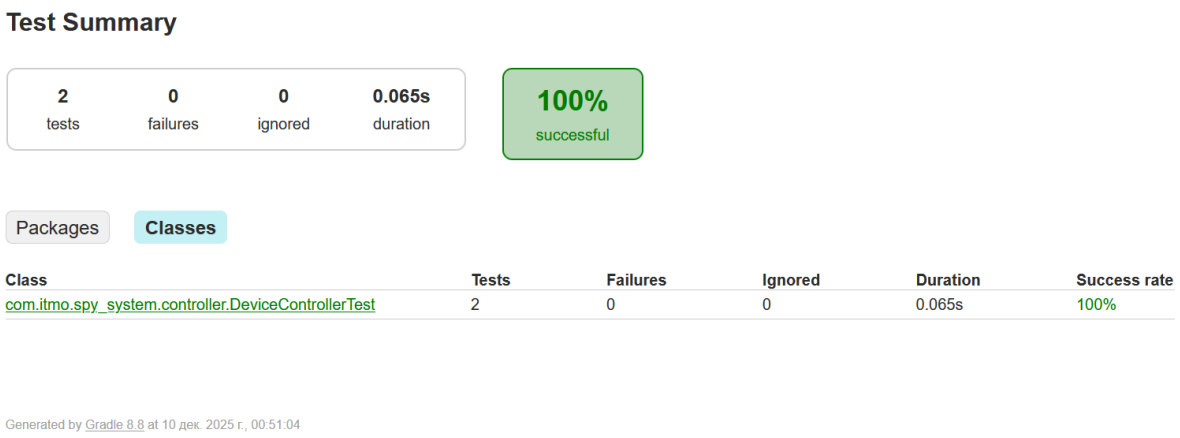
Пример:

Тестовый сценарий	Описание	Результат	Возникшие ошибки
Корректность работы навигационной панели	Описание действий	Пройден	нет
Соответствие дизайн-макету страницы авторизации	Описание действий	Не пройден	*ссылка на подробную информацию об ошибке*
Соответствие дизайн-макету страницы "Панель управления пользователя"		Пройден	нет
Соответствие дизайн-макету страницы "Панель управления менеджера"		Не пройден	*ссылка на подробную информацию об ошибке*
Соответствие дизайн-макету страницы "Панель управления сотрудника наказаний"		Не пройден	*ссылка на подробную информацию об ошибке*
Соответствие дизайн-макету страницы "Панель управления сотрудника слежки"		Пройден	нет
Проверка работы модального окна редактирования информации о пользователе	Описание действий	Пройден	нет
Проверка работы модального окна редактирования расписания сотрудника		Не пройден	*ссылка на подробную информацию об ошибке*
Проверка работы модального окна редактирования задания		Не пройден	*ссылка на подробную информацию об ошибке*

2) Unit Тестирование

Артефакт – автогенерируемый html документ, содержащий статистику и результаты выполнения тестов

Пример:



3) Тестирование производительности, нагрузочное тестирование и стресс-тестирование

Артефакт – отчеты инструмента JMeter в формате HTML, содержащие статистику о проведенных тестах.

Пример:

Test and Report informations

File:	"results-lt.csv"
Start Time:	"20/10 18:45:13"
End Time:	"20/10 20:46:04"
Filter for display:	"((^AjoutPanier)|(^Visite)|(^ClickAnneeTicket)|(^ClickDetailCommande)|(^ClickDetailTicket)|(^ClickMaCarteMaison)|(^ClickMesCoordonnees)|(^ClickMesTicketsCaisse)|(^ClickMonCompte)|(^ClickMonMagasin)|(^ClickMonSuiviCommande)|(^ClickOngletCommunaute)|(^ClickOngletConseils)|(^ClickPlusMoins)|(^CommunautePagination)|(^Continuer)|(^ContinuerMaCommande)|(^FicheProduit)|(^HP)|(^Ma[DonneesPerso)|(^MeConnecter)|(^PasserMaCommande)|(^PasserPro)|(^PositionnerModeLivraison)|(^PositionnerModeRelais)|(^PositionnerModeRetrait)|(^RechercheParCp)|(^RechercheParCritere)|(^RechercheParRef)|(^SaisieCP)|(^SelectionJour)|(^SelectionMagasin)|(^SelectionSecteurActivite)|(^ValiderAdresseLivPart)|(^ValiderAdresseLivPro)|(^VoirPanier)|(^autoComplete)|(^autoCompleteStore)|(^choixMagasin)|(^choixMagasinParCodePostal)|(^clickCours)|(^clickFamille)|(^clickFiltreCours)|(^clickHistoCommandeAnnee)|(^clickLienAutoComplete)|(^clickLienService)|(^clickMagasinsEtServices)|(^clickPagination)|(^clickRecherche)|(^clickSousUnivers)|(^clickUnivers)|(^cours-de-bricolage)|(^familleTriParUrl)|(^location-vehicules)|(^paginationFamille)|(^rechercheMagParGps)|(^sc1-recherche)|(^sc2-descente-produit)|(^sc3-commande-lad)|(^sc3-commande-retrait-mag)|(^sc3-commande-retrait-relais)|(^sc4-compte-internaute)|(^sc5-services)|(^selectionPointRelais)|(^showConcept)|(^triResultatsRecherche))(-success|-failure)?"

APDEX (Application Performance Index)

Apdex	T (Toleration threshold)	F (Frustration threshold)	Label
0.957	**2 sec**	**8 sec**	**Total**
0.984	2 sec	8 sec	AjoutPanier
0.984	2 sec	8 sec	autoComplete
0.999	2 sec	8 sec	autoComplete-1
0.999	2 sec	8 sec	autoComplete-2
1.000	2 sec	8 sec	autoCompleteStore
1.000	2 sec	8 sec	choixMagasin
0.994	2 sec	8 sec	ClickAnneeTicket
1.000	2 sec	8 sec	clickCours
0.946	2 sec	8 sec	ClickDetailTicket
0.998	2 sec	8 sec	clickFamille
0.988	2 sec	8 sec	clickFiltreCours
0.992	2 sec	8 sec	clickLienAutoComplete
1.000	2 sec	8 sec	clickLienService

Requests Summary

7.2 Perceived Quality Reports (Оценка качества)

[Опишите формат и содержание отчета о качестве разрабатываемой системы]

Численные критерии:

- Процент покрытия строк кода тестами (Code coverage)
- Количество успешных/неуспешных тестов к общему количеству
- Количество выявленных ошибок

7.3 Incident Logs and Change Requests (Журналы ошибок и изменений)

[Опишите, каким образом будут фиксироваться найденные ошибки в системе, а также изменения, сделанные с целью их исправить.]

Описание проблем, приоритет, статус решения ведутся в Google Spreadsheet. Решение проблемы разработчики должны представить в виде слитого PullRequest'а, в Google Spreadsheet заносится ссылка на PullRequest.

8. Environmental Needs (Необходимое окружение для проведения тестирования)

[Данный раздел содержит описание ресурсов (за исключением людей), необходимых для выполнения плана тестирования.]

8.1 Base System Hardware (Базовое аппаратное обеспечение)

[Опишите в таблице, приведенной ниже, конфигурацию систем(ы), на которой будут запускаться тесты]

Остальные виды тестирования могут проводиться не персональных рабочих компьютерах участников команды, которые должны удовлетворять минимальным требованиям:

Resource (Ресурс)	Quantity (Количество)	Name and Type (Название и тип)
ПК	1	8GB ram / 128GB space / Intel Core i5, 4 cores

8.2 Base Software Elements in the Test Environment (Базовые программы тестового окружения)

[Опишите в таблице, приведенное ниже, какие программы должны быть установлены на тестовой(ых) системе(ах).]

Software Element Name (Название)	Version (Версия)	Type (Тип)
OpenJDK	17	Среда выполнения серверной части системы
PostgreSQL	14	База данных
Chrome	142.0.0+	Браузер

8.3 Productivity and Support Tools (Вспомогательные инструменты)

[Опишите в таблице, приведенное ниже, какие программы будут полезны для проведения тестирования.]

Tool Category or Type (Тип программы)	Tool Brand Name (Название)	Vendor (Производитель)	Version (Версия)
Среда разработки	IntelliJ IDEA	JetBrains	2024+
Инструмент для нагрузочного и стресс-тестирования	Apache Jmeter	Apache Foundation	5+
Инструмент для администрирования БД	DBeaver	OpenSource	25.2.5+
Инструмент для тестирования API	Postman	Postman, Inc.	11.73.0+
Инструмент для	Selenium	Software Freedom	4.38.0+

тестирования UI		Conservancy	

9. Responsibilities, Staffing, and Training Needs (Обязанности сотрудников)

[В данном разделе описываются необходимые навыки и знания людей, осуществляющих процесс тестирования.]

9.1 People and Roles (Люди и роли)

Role (Роль)	Minimum Resources Recommended (Минимально необходимое количество людей)	Specific Responsbilities (Обязанности)
Тестировщик	1	Ручное тестирование. Разработка и исполнение интеграционных тестов. Разработка автотестов.
Разработчик	1	Разработка unit тестов.
Менеджер	1	Управление процессом тестирования - постановка задач, прием результатов

10. Management Process and Procedures (Управление)

[Данный раздел содержит описание различных мероприятий по управлению процессом тестирования]

10.1 Reporting on Test Coverage (Сообщение о тестовом покрытии)

[Опишите процесс рецензирования результатов тестирования.]

Тестировщик проводит отведенный ему вид тестирования, в результате которого пишет отчет о тестировании.

После проведения тестирования проводится анализ полученных отчетов на соответствие заявленным требования, при несоответствии заявленным требованиям формируются задачи для исправления.

10.2 Problem Reporting, Escalation, and Issue Resolution (Выявление, избегание и решение проблем)

[Опишите, каким образом будет вестись учет проблем, возникших во время выполнения тестов, и какие действия нужно предпринять для их решения.]

При выявлении недочета системы тестировщик обязан его задокументировать, описав некорректное поведение как можно детальнее, уточнить у менеджера приоритет и заносит в Google Spreadsheet.

Разработчики в свою очередь осуществляют выполнение задач, связанных с недочетами, в соответствии с установленным планом.

При планировании очередного цикла работ менеджер выделяет наиболее значимые проблемы и заносит их в план. План работ получают разработчики.

10.3 Approval and Signoff (Утверждение плана тестирования)

[Опишите процесс утверждения данного плана тестирования, а также укажите список лиц, участвующих в нём.]

В процессе утверждения плана тестирования участвуют Тестировщик, Разработчик, Менеджер.

Тестировщик заводит документирует известные ошибки в системе и назначает им приоритеты.

Разработчик определяет сроки выполнения и порядок исправлений в зависимости от неисправной функциональности.

Менеджер утверждает план тестирования на ближайший цикл работ в зависимости от глобального приоритета команды на этот цикл, желаний заказчика и задач на расширение функциональности.