Веб-версия Rocket.Chat

Тестовая стратегия

Версия 1.0

История изменений

|  |  |  |  |
| --- | --- | --- | --- |
| **Дата** | **Версия** | **Описание** | **Автор** |
| 05/07/2023 | 1.0 | Создание документации - Тестовая стратегия | Шабарова Ю.А. |
| 10/07/2023 | Тестирование | Шабарова Ю.А. |
|  |  |  |  |
|  |  |  |  |

Оглавление

[1. Введение 4](#_Toc139911900)

[2. Типы проводимого тестирования](#_Toc139911902) 6

[2.1 Функциональное тестирование](#_Toc139911903) 6

[2.2 Нефункциональное тестирование 7](#_Toc139911904)

[3. Части системы веб-версии Rocket.Chat 9](#_Toc139911918)

[4. Окружение для работы 10](#_Toc139911919)

[5. Виды тестовой документации 11](#_Toc139911920)

[6. Время проведения тестирования](#_Toc139911921) 13

Тестовая стратегия

1. Введение

Подготовка тестовой стратегии является важным этапом в процессе тестирования программного обеспечения. Тестовая стратегия определяет подходы, методы и ресурсы, которые будут использоваться для выполнения тестирования и достижения целей проекта.

В данном документе описываю методы и подходы которые используются для тестирования сайта и приложения. Данный документ может быть использована как тестировщиками, так и менеджерами, и разработчиками в рамках проекта.

**Объект тестирования:** веб-версия Rocket.Chat

**Цель тестирования:** включает в себя определение информации о проекте, разработку тестовой стратегии, выделение ресурсов и представление результатов. Она направлена на обеспечение качества программного обеспечения и удовлетворение требований проекта и ожиданий заинтересованных сторон.

**Краткое описание объекта:** веб-версия Rocket.Chat - это онлайн-чат-платформа, которая предоставляет возможность общения и совместной работы для пользователей. Пользователи могут создавать групповые и приватные чаты, обмениваться сообщениями, файлами, изображениями и документами, делиться экраном, а также настраивать свои профили и уведомления.

Это платформа с открытым исходным кодом. Благодаря этому мессенджер может быть настроен в соответствии с потребностями различных компаний, с возможностью локального, облачного или гибридного развертывания.

Встроенные в Rocket.Chat инструменты помогают компаниям и командам улучшить корпоративную коммуникацию, сотрудничество и производительность.

**Особенности веб-версия Rocket.Chat:**

* Многопользовательский чат: Rocket.Chat поддерживает одновременное участие множества пользователей в чате, позволяя им общаться в режиме реального времени.
* Групповые и приватные чаты: Пользователи могут создавать групповые чаты для сотрудничества внутри команд и приватные чаты для приватных общений.
* Обмен файлами и документами: Пользователи могут отправлять файлы различных форматов, включая изображения, документы, архивы и другие файлы, внутри чата.
* Настройки профиля и уведомления: Пользователи могут настраивать свои профили, включая фотографии, и устанавливать предпочтения уведомлений, чтобы получать информацию по своим предпочтениям.
* Интеграция с другими системами: Rocket.Chat может интегрироваться с другими приложениями и системами, такими как системы управления проектами, электронная почта или системы уведомлений.
* Многоплатформенность: Веб-версия Rocket.Chat доступна через веб-браузеры на различных платформах, включая компьютеры, смартфоны и планшеты.

**Исполнитель**: Шабарова Юлия Александровна, Junior QA Engineer, логин на платформе Школа 21: saxophon

**Навыки:**

|  |  |
| --- | --- |
| Тестирование: | Теория, методология, техники тест-дизайна |
| Документация: | Тестовая стратегия, тест-план, тест-кейс, баг-репорт, чек-лист, тестовый набор, сценарий использования, отчет о результатах |
| Инструменты: | Android Studio SDK, Appetize.io, Git, GitLab, DevTools, Fiddler Classic, Rocket.Chat, Notion, Postman, PowerShell, TestIT, Swagger UI,  VS Code, Webminal, Webinar |
| Основы Web: | CSS, HTML, JavaScript, XPath, XML |
| Основы работы с СУБД: | SQL  Client-Server Architecture |

# Типы проводимого тестирования

Тестирование веб-приложения Rocket.Chat может включать следующие типы проводимого тестирования: функциональное тестирование и нефункциональное тестирование.

* 1. **Функциональное тестирование:**

**Проверка основных функций:**

* Функциональное тестирование включает проверку основных функций приложения для убеждения в их корректной работе. Это может включать функции входа в систему, регистрации, отправки сообщений, управления данными и другие.

**Верификация и валидация данных**:

* Важной частью функционального тестирования является проверка верификации и валидации данных, включая проверку правильности обработки пользовательского ввода, проверку целостности данных и соответствия заданным форматам и структурам.

**Проверка взаимодействия компонентов:**

* Функциональное тестирование также включает проверку взаимодействия различных компонентов и модулей приложения. Это может включать проверку передачи данных между компонентами, правильность вызова функций и проверку согласованности данных.

**Тестирование различных сценариев использования:**

* Функциональное тестирование проводится с использованием различных сценариев использования приложения, чтобы убедиться, что оно работает правильно и предоставляет ожидаемый функционал в различных ситуациях. Это включает проверку типичных, альтернативных и исключительных сценариев.

**UI-тестирование (тестирование пользовательского интерфейса):**

* UI-тестирование является частью функционального тестирования и фокусируется на проверке правильности отображения элементов пользовательского интерфейса и их взаимодействия с пользователем.

Цель UI-тестирования состоит в проверке соответствия пользовательского интерфейса приложения дизайну и спецификациям, а также взаимодействия с пользователем в соответствии с ожиданиями.

*Примеры UI-тестирования включают проверку правильности расположения элементов интерфейса, цветовой схемы, работу кнопок, форм и других элементов, а также проверку переходов между различными экранами.*

**UX-тестирование (тестирование пользовательского опыта):**

* UX-тестирование фокусируется на оценке пользовательского опыта и удовлетворенности пользователей при использовании приложения.

Цель UX-тестирования заключается в понимании, насколько приложение удовлетворяет потребностям и ожиданиям пользователей, а также в выявлении проблем и улучшении пользовательского опыта.

*Примеры UX-тестирования включают пользовательские исследования, тестирование прототипов или ранних версий приложения, анализ пользовательских потоков и другие методы для оценки эмоционального восприятия, удобства использования и общего впечатления от приложения.*

Регрессионное тестирование является важным компонентом стратегии тестирования для готового веб-приложения. Поскольку разработка и внесение изменений в приложение могут привести к возникновению ошибок или нарушению ранее работающей функциональности, регрессионное тестирование поможет убедиться, что новые изменения не повлияли на уже проверенные и работающие части приложения. Оно позволяет выявить и исправить потенциальные проблемы и обеспечить стабильность приложения.

* 1. **Нефункциональное тестирование:**

**Тестирование производительности:**

* Нагрузочное тестирование: проверка производительности приложения под ожидаемой нагрузкой и большим количеством пользователей.
* Тестирование масштабируемости: проверка способности приложения масштабироваться при увеличении числа пользователей или объема данных.

*Проверка производительности поможет определить, как приложение справляется с ожидаемыми нагрузками и обрабатывает большое количество пользователей и сообщений. Это позволяет выявить возможные узкие места и улучшить производительность, чтобы обеспечить плавную работу приложения даже при высокой нагрузке.*

**Тестирование безопасности:**

* Тестирование на проникновение: проверка уязвимостей приложения и его защиты от несанкционированного доступа.
* Тестирование аутентификации и авторизации: проверка механизмов аутентификации и авторизации, чтобы убедиться, что доступ к функциям приложения ограничен правильно и безопасно.
* Тестирование обработки данных: проверка, как приложение обрабатывает и хранит данные, чтобы обеспечить их безопасность и целостность.

*Безопасность является критическим аспектом для любого приложения, особенно для приложений, работающих с конфиденциальными данными и личной информацией пользователей. Проверка на уязвимости, аутентификацию, авторизацию и другие аспекты безопасности помогут обнаружить и решить потенциальные угрозы и защитить приложение и данные пользователей.*

**Тестирование совместимости:**

* Тестирование на разных платформах и браузерах: проверка, как приложение работает и отображается на различных операционных системах, браузерах и устройствах.
* Тестирование на разных разрешениях экрана: проверка, как приложение адаптируется к различным разрешениям экрана, чтобы убедиться, что пользовательский интерфейс остается доступным и функциональным.

**Тестирование надежности:**

* Тестирование восстановления после сбоев: проверка, как приложение восстанавливается и восстанавливает данные после сбоев или ситуаций, требующих перезапуска.
* Тестирование стабильности: проверка стабильности и надежности приложения в течение продолжительного времени работы или при длительной нагрузке.

**Тестирование доступности:**

* Проверка, насколько приложение доступно для пользователей с ограниченными возможностями, такими как пользователи с ограничениями зрения или подвижности.

*Относительно платформ, на которых следует проводить тестирование, это зависит от целевой аудитории и требований проекта. Веб-приложение Rocket.Chat может быть доступно на различных платформах, включая различные операционные системы (например, Windows, macOS, Linux), браузеры (например, Chrome, Firefox, Safari) и устройства (например, настольные компьютеры, мобильные устройства). Стратегия тестирования должна включать покрытие наиболее популярных платформ, используемых целевой аудиторией, чтобы обеспечить соответствие и качество работы приложения в разных окружениях.*

# Части системы, которые будут протестированы

В настоящее время, приложение находится на стадии эксплуатации, модульное и интеграционное тестирование успешно завершено.

Типичный общий сценарий:

1) Модульное тестирование

2) Интеграционное тестирование

3) **Системное тестирование**

Таким образом, цель настоящей стратегии направлена на проверку соответствия исходных данных и проведение системного тестирование.

1. Каналы

* Создание;
* Редактирование;
* Удаление;

1. Команды

* Создание;
* Редактирование;
* Удаление;

1. Личные переписки

* Создание;
* Редактирование;
* Удаление;

1. Обсуждения

* Создание;
* Редактирование;
* Удаление;

1. Переход по ссылкам на главной странице

* Переход на Mobile apps;
* Переход на Desktop apps;

1. Присоединение к существующим каналам;
2. Работа с каталогом;
3. Работа чата

* Отправка текстовых сообщений;
* Отправка голосовых сообщений;
* Загрузка файлов;
* Создание трайдов;
* Добавление реакций;
* Цитирование;
* Работа с участниками;
* Работа с уведомлениями;

1. Профиль

* Просмотр профиля;
* Редактирование профиля;

1. Статус

* Просмотр статуса;
* Редактирование статуса;

1. Создание звонков
2. Панель: Учетная запись

* Настройки:
  + Вкладка Язык;
  + Вкладка Общее;
  + Вкладка Присутствие пользователя;
  + Вкладка Уведомления;
  + Вкладка Сообщение;
  + Вкладка Подсветка сообщений;
  + Вкладка Звуковые оповещения;
  + Вкладка Мои данные.
* Профиль;
* Безопасность;
* Токены для личного доступа;
* Настройки Omnichannel;
* Manage Devices.

1. Панель: Администрирование:

* Администрирование;
* Приложение;
* Аудит.

# Окружение для работы. Описание операционных систем, версии приложения и браузеров

Сведения о ПК: операционной системой **Windows 10**

Сведения о браузере: **Google Chrome** Версия 114.0.5735.199 (Официальная сборка), (64 бит)

Сведения о веб-версия: **Rocket.Chat 6.2.10** от 14.07.2023

# Виды тестовой документации, которые будут составляться в процессе тестирования.

При тестировании веб-приложения Rocket.Chat могут использоваться следующие **тестовые артефакты:**

* **Тест-план** **(Test plan):** Документ, описывающий общую стратегию и подход к тестированию. В нем могут быть указаны цели тестирования, основные функциональные и нефункциональные требования, спецификации окружения и тестовые данные, а также расписание и ресурсы, выделенные для тестирования.
* **Сценарий тестирования (User case):** Описание последовательности шагов, которые должен выполнить тестировщик, чтобы проверить определенный функционал или сценарий использования веб-приложения. Сценарии тестирования могут включать в себя входные данные, ожидаемые результаты и шаги по воспроизведению конкретных действий пользователей.
* **Тест-кейсы** **(Test cases):** Детальное описание тестовых сценариев, включающее шаги, ожидаемые результаты и ожидаемое поведение системы. Тест-кейсы могут быть структурированы по модулям или функциональности и предназначены для полного охвата возможных вариантов использования веб-приложения.
* **Отчет о тестировании** **(Test report):** Документ, содержащий результаты выполненных тестов, выявленные проблемы и рекомендации по улучшению. Отчет может включать описание проведенных тестов, их статус (пройдены, не пройдены, пропущены), сводную информацию о найденных дефектах и общую оценку качества приложения.
* **Баг-репорты** **(Bug reports):** Документы, содержащие описание выявленных проблем или дефектов в веб-приложении. Каждый баг-репорт обычно содержит информацию о симптомах, шагах для воспроизведения, ожидаемом и фактическом поведении системы, а также другие детали, которые помогают разработчикам понять и исправить проблему.
* **Чек-лист** **(Checklist):** Список пунктов, который помогает тестировщику проверять функциональность или особенности приложения. Чек-лист может включать конкретные шаги или проверки, которые нужно выполнить для каждого аспекта приложения.
* **Тестовый набор (Test suite):** Группа связанных тест-кейсов, объединенных по определенному критерию. Например, можно создать тестовый набор для базовой функциональности, набор для тестирования чат-комнат, набор для тестирования безопасности и т. д.

**Техники тест-дизайна**, которые могут использоваться в тестировании Rocket.Chat:

В тестировании программного обеспечения существуют различные техники тест-дизайна, которые помогают тестировщиками создавать эффективные и полные наборы тестов.

* **Классы эквивалентности (Equivalence Partitioning):**

Техника классов эквивалентности предполагает разделение входных данных на группы или классы, где каждый класс имеет одинаковое поведение программы. Затем тестирование выполняется по одному представителю из каждого класса. Это позволяет сократить количество тестовых случаев, при этом охватывая все классы эквивалентности.

* **Граничные значения (Boundary Value Analysis):**

Техника граничных значений основывается на том, что ошибки чаще всего возникают на границах допустимых значений. При использовании этой техники тестирования выбираются тестовые случаи, которые проверяют поведение программы на границах и около них. Это позволяет выявить ошибки, связанные с неправильной обработкой минимальных и максимальных значений, а также с переполнением или недостатком ресурсов.

* **Диаграмма состояний (State Diagram):**

Диаграмма состояний представляет собой графическую модель, которая описывает все возможные состояния системы и переходы между ними в ответ на события. Эта техника тест-дизайна позволяет тестировщикам визуализировать различные состояния приложения и проверить правильность переходов между состояниями и обработку событий.

* **Попарное тестирование (Pairwise Testing):**

Техника попарного тестирования заключается в проверке всех возможных комбинаций параметров или входных данных, чтобы обнаружить возможные проблемы взаимодействия между ними. Вместо тестирования всех возможных комбинаций, попарное тестирование генерирует минимальный набор тестов, который покрывает все возможные пары параметров. Это позволяет снизить объем тестирования, сохраняя при этом высокую степень покрытия.

* **Причинно-следственный анализ (Cause-Effect Analysis):**

Причинно-следственный анализ помогает выявить основные причины и факторы, которые могут вызывать определенные проблемы или дефекты в приложении. Эта техника позволяет идентифицировать основные причины проблем и сосредоточить усилия на их устранении. Причинно-следственный анализ может быть использован для идентификации уязвимостей, проблем производительности или ошибок взаимодействия между компонентами системы.

# Время проведения тестирования.

Время начала тестирования: 05/07/2023

Время окончания тестирования: 15/07/2023