Системное тестирование веб-версии Rocket.Chat

Тестовая стратегия

Версия<1.0>

История изменений

|  |  |  |  |
| --- | --- | --- | --- |
| **Дата** | **Версия** | **Описание** | **Автор** |
| 25.07.2023 | 1.0 | Разработана первая версия тестовой стратегии | Потапова Т.А. |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Оглавление

[1. Введение 4](#_Toc140505186)

[2. Типы проводимого тестирования 5](#_Toc140505187)

[2.1 Функциональное тестирование 5](#_Toc140505188)

[2.2 UI – тестирование интерфейса 5](#_Toc140505189)

[2.3 UX – тестирование (юзабилити-тестирование) 5](#_Toc140505190)

[2.4 Нефункциональное тестирование 5](#_Toc140505191)

[2.4.1 Тестирование производительности 5](#_Toc140505192)

[2.4.2 Тестирование юзабилити 6](#_Toc140505193)

[2.4.3 Тестирование безопасности 6](#_Toc140505195)

[2.4.4 Тестирование отказоустойчивости 6](#_Toc140505196)

[3. Части системы, которые будут протестированы (нужно описать все модули системы и за что каждый из них отвечает) 6-7](#_Toc140505197)

[4. Окружение для работы 7-8](#_Toc140505198)

[5. Виды тестовой документации, которые будут составляться в процессе тестирования, обоснование выбора. Какие техники тест-дизайна будут использоваться при формировании тест-кейсов 8](#_Toc140505199)

[6. Время проведения тестирования 9](#_Toc140505200)

Тестовая стратегия

# Введение

Платформа Rocket.Chat – это бесплатная, масштабируемая, гибко настраиваемая и безопасная платформа с открытым исходным кодом, которая позволяет вам общаться и сотрудничать с вашей командой, обмениваться файлами и общаться в режиме реального времени. Она является кроссплатформенной и работает в мобильных операционных системах Linux, Windows, macOS, Android и iOS.

Преимущества платформы Rocket.Chat:

- открытый исходный код. Благодаря этому мессенджер может быть настроен в соответствии с потребностями различных компаний, с возможностью локального, облачного или гибридного развертывания, расширения или добавления новых функций в соответствии с требованиями команды или бизнеса;

- есть возможно самостоятельно контролировать поток своих данных, пользователь сам решает где разместить экземпляр сервера и как сделать его безопасным;

- для обеспечения безопасной связи: сквозное шифрование (если эта функция включена, только участники чата могут расшифровывать и читать содержимое сообщения. Если отключена, есть возможность выбрать то, что остается приватным, а что может быть видно другим пользователям системы) , двухфакторная аутентификация(2FA) , белый список IP-адресов , групповую синхронизацию LDAP и единый вход . Это помогает обеспечить безопасность данных и разговоров пользователей;

- платформа доступна на компьютерах и телефонах, в виде веб- и мобильных приложений для iOS и Android. Так же возможна интеграция с другими приложениями, включая Jira, GitHub и Trello;

- пользователи платформы могут создавать закрытые и открытые частные группы для общения, а также переписываться тет-а-тет, возможна группировка чатов в потоки по темам, проектам,есть функция поиска для быстрой навигации сообщений;

- есть встроенная опция в каждый из чатов для обмена файлами любого типа и размера;

- предоставление множества функций для комуникации: видеозвонков и аудиоконференций, обмена сообщениями и файлами, совместного использования экрана. Пользователи могут коллективно работать над документами, презентациями и другим контентом в режиме реального времени.

Rocket.Chat изначально поддерживает две основные системы видеоконференций: Jitsi и Big Blue Button. Обе имеют схожие преимущества с Rocket.Chat:

- Открытый источник

- Полный контроль для администраторов

- Прозрачность того, как используются ваши персональные данные

Таким образом, встроенные в Rocket.Chat инструменты помогают компаниям и командам улучшить корпоративную коммуникацию, сотрудничество и производительность.

***Автор данного задания*** - студент Школы 21 Потапова Татьяна Александровна (логин –granvilr).

Работала со следующими проектами: Git, VS Code, GitLab, TestIT, Notion, Swagger, Postman, DevTools, Rocket.Chat, Android Studio SDK, EventLog, PowerShell, Webminal, SQL.

# Типы проводимого тестирования

## Функциональное тестирование

Функциональное тестирование – это тестирование, которое будет проверять платформу Rocket.Chat на соответствие функциональным требованиям.

Этапы проведения функционального тестирования:

1. Определение функционала, которое необходимо для тестирования;
2. Изучение частей системы: как должна работать каждая система и как пользователь (администратор) будет ее использовать;
3. Описание сценариев использования веб-версии Rocket.Chat;
4. Написание тест-кейсов на базе сценариев использования;
5. Подготовка тестовых данных максимально приближенных к тем, что могут использовать пользователи;
6. Проведение тестирования – сравнение фактического результата и ожидаемого;
7. Составление отчета по результатам тестирования.

## UI – тестирование интерфейса

UI ー это User Interface, «пользовательский интерфейс». При проведении теста интерфейса необходимо имитировать действия пользователя платформы. Задача такого тестирования ー убедиться, что все компоненты системы правильно взаимодействуют друг с другом.

Тестирование проводится с помощью тест-кейсов на базе случаев использования (User Case). По каждому случаю использования разработан один тест-кейс

## UX – тестирование (юзабилити-тестирование)

UX – тестирование – пользовательский опыт - метод наблюдения за респондентами, для выявления проблемных мест и точек роста интерфейса. Взаимодействие пользователя с интерфейсом и удобство его использования.

Шаги проведенияUX – тестирования:

1. Определение проблемы, которую нужно решить при помощи тестирования;
2. Поиск ЦА платформы, поиск респондентов;
3. Создание пользовательских сценариев;
4. Определение вида (качественные/количественные) и методики тестирования.

## Нефункциональное тестирование

Нефункциональное тестирование необходимо для проверки нефункциональных требований платформы. С помощью нефункционального тестирования необходимо проверить соответствует ли поведение системы требованиям по всем аспектам, не охваченные функциональным тестированием.

### **Тестирование производительности**

Тестирование производительности – это процесс, направленный на определение скорости, масштабируемости, стабильности и эффективности работы приложения или системы в различных условиях нагрузки. Основная цель такого тестирования заключается в том, чтобы определить, как система будет работать в реальном мире с большим количеством пользователей или при большом объеме данных.

Типы тестирования производительности:

1. Нагрузочное тестирование. Цель – проверить как система ведет себя при постоянной нагрузке;
2. Стресс – тестирование. Цель - выявить поведение приложения, вынужденного работать за пределами нормальной или пиковой нагрузки;
3. Тестирование выносливости. Цель - убедиться, что использование ресурсов остается на том же уровне при имитации стандартной нагрузки в течение длительного времени;
4. Пиковое тестирование. Цель - измерить способность системы обращаться с кривой экстремального спроса, удерживать высокую нагрузку короткое время, и эффективно восстановить ресурсы после этого всплеска.

Инструмент, при помощи которого проводится проверка и отладка – DevTools,

### **Тестирование****юзабилити**

### Тестирование с целью определения степени понятности, легкости в изучении и использовании, учитывается уровень комфорта при взаимодействии с программным продуктом, оценивается обратная связь (на этом уровне обращают внимание на визуальное оформление, навигацию, логичность и др.).

### **Тестирование безопасности**

Тестирование безопасности приложений (AST) - это процесс повышения устойчивости приложений к угрозам безопасности путем выявления слабых мест и уязвимостей в исходном коде.

Этапы тестирования безопасности:

1. Сбор информации об элементах, используемых для разработки веб-приложений, с учетом операционной системы, аппаратного обеспечения и технологий;
2. Список потенциальных рисков и уязвимостей;
3. Подготовка профиля рисков;
4. Автоматическое тестирование;
5. Разработка отчетности.

Этапы проведения контроля доступа:

1. Проверка средств аутентификации и авторизации;
2. Определение требований парольной политики;
3. Тестирование подбора учетных данных;
4. Тестирование восстановления учетной записи;
5. Тестирование функций идентификации учетной записи;
6. Проверка полномочий и прав доступа;
7. Исследования сессии (время жизни, сессионный токены, признаки, попытки одновременной работы и т.д.);

### **Тестирование отказоустойчивости**

Тестирование на отказоустойчивость (FaultToleranceTesting) — это процесс проверки надежности и стабильности программного обеспечения при возникновении сбоев, ошибок или непредвиденных ситуаций. Цель данного вида тестирования — определить способность системы продолжать работу и восстанавливаться после возникновения проблем.

Этапы тестирования отказоустойчивости:

1. Определение требований к отказоустойчивости системы;
2. Разработка тестовых сценариев, которые проверяют работу системы в различных условиях;
3. Подготовка среды, в которой будет проводиться тестирование (включая аппаратное и программное обеспечение);
4. Выполнение тестов согласно тестовым сценариям;
5. Анализ результатов тестов и их документация.

## 2.4.4 Тестирование совместимости

Программное обеспечение должно быть установлено на всех версиях Windows, на любых веб-браузерах. Для смартфонов и планшетов существуют версии Rocket.Chat в Google Play или App Store.

# Части системы, которые будут протестированы (нужно описать все модули системы и за что каждый из них отвечает)

1. Панель пользователя

1.1 Имя и текущий статус. Отображает имя пользователя и текущий статус, выбранный пользователем

1.2 Статус. Отображает параметры статуса, доступные для выбора (онлайн, отошел, занят, офлайн). Так же можно установить настраиваемый статус.

1.3 Тема. Установка темы рабочего пространства

1.4 Моя учетная запись. Управлять настройками своей учетной записи через браузер, настольные и мобильные приложения

1. Главная страница

2.1 Добавление пользователей. Пригласить и добавить участников в рабочую область, чтобы начать общение

2.2 Создание канала. Создать общедоступный канал, к которому могут присоединиться новые участники рабочей области.

2.3 Присоединиться к комнатам (к существующим каналам). Открыть общедоступные каналы и команды в каталоге рабочего пространства.

2.4 Мобильные приложения. Установить Rocket.Chat как мобильное приложение.

2.5 Настольные приложения. Установить Rocket.Chat на предпочитаемую настольную платформу.

2.6 Документация. Информация о больших возможностях Rocket.Chat.

2.7 Настройка внешнего вида рабочего пространства.

1. Поисковая строка. Поиск каналов по введенным буквам (латиница) или цифрам
2. Каталог. Открытие перечня каталогов, пользователей и команд
3. Редактирование внешнего вида списка чатов. Их сортировка и группировка
4. Редактирование и удаление каналов. Возможность создать новый канал, с последующим его удалением
5. Создание команд. Возможность способствовать сотрудничеству между разными командами с заданными параметрами
6. Создание личной переписки. Возможность создания разовых разговоров, звуковых сообщений
7. Создание обсуждения, а также редактирование и удаление
8. Администрирование. Руководство по пробному периоду
9. Приложения. Настроить пользовательские приложения, которые будут использоваться для открытой аутентификации.
10. Аудит сообщений и его журнал. Просмотр журнала аудита сообщений с указанием периода
11. Настройка веб-приложения

13.1 Изменение логотипа, иконки и многого другого для своего рабочего пространства

13.2 Настроить общие параметры рабочего пространства

13.3 Просмотр, как пользователи взаимодействуют с вашим рабочим пространством

13.4 Выгрузка пользовательских данных. Конфигурации, позволяющие или запрещающие участникам рабочей области загружать данные рабочей области

13.5 Безопасность. Конфиденциальная беседа.

13.6 Настроить загрузку и хранение файлов

13.7 Управление устройствами. Настроить политику безопасности и контроля доступа

13.8 Хранилище пользовательских звуков и эмодзи.

1. Возможности работы чата

14.1 Просмотр информации о чате. Взаимодействие пользователей

14.2 Просмотр тредов и обсуждений чата. Создание отдельной переписки в чате

14.3 Просмотр параметров чата: настройки уведомлений, упоминания, отмеченные сообщения, прикрепленные сообщения, экспорт сообщений, горячие клавиши, удалить сообщения

14.4 Просмотр участников чата

14.5 Поиск сообщений и файлов в чате

14.6 Отправка сообщений текстовых и голосовых. Опция написать и отправить сообщение в текстовом и голосовом виде

14.7 Отправка видеосообщения. Способность записывать видеосообщение

14.8 Прикрепление файлов (документов). Возможность отправки файловых документов

1. Создание звонков. Возможность создавать звонки, в т.ч. видеоконференции

# Окружение для работы

Выпуск Windows 10 Pro

Версия 22H2

Сборка ОС 19045.3208

Яндекс Браузер Версия 23.7.0.2530

# Виды тестовой документации, которые будут составляться в процессе тестирования, обоснование выбора. Какие техники тест-дизайна будут использоваться при формировании тест-кейсов

**Виды тестовой документации, которые будут составляться в процессе тестирования:**

- *Сценарии использования* — это сценарий взаимодействия пользователя (или пользователей) с программным продуктом для достижения конкретной цели. Сценарии использования используются для анализа, проектирования и тестирования системы. Они помогают понять, как система будет использоваться в реальных условиях и какие функциональные требования должны быть удовлетворены. Пошаговое подробное взаимодействие пользователя с системой.

- *Тест - кейсы* – это описанный алгоритм, который описывает шаги и условия для выполнения определенного тестового сценария. Он содержит информацию о входных данных, ожидаемых результатах и ожидаемом поведении системы при выполнении определенных действий. Тест-кейсы используются для тестирования программного обеспечения с целью проверки его работоспособности, соответствия требованиям и выявления возможных ошибок или дефектов.

- *Тест - план* – это документ, который определяет общий подход и стратегию тестирования программного обеспечения

- *Отчет о тестировании* – это документ, который содержит информацию о результатах проведенных тестов программного обеспечения. Он предоставляет подробную информацию об объектах тестирования, целях и задачах, а также методах и ресурсах, необходимых для проведения тестирования. Отчет о тестировании также может включать информацию о покрытии тестами, времени, затраченном на каждый тестовый сценарий, и оценку качества продукта на основе результатов тестирования.

**Техники тест - дизайна, которые будут использоваться при формировании тест-кейсов:**

- *Классы эквивалентности (эквивалентное разбиение)*. Эта техника позволяет минимизировать число тестов, не создавая сценарий для каждого возможного значения. Техника используется для разделения входных данных на группы (классы) схожих значений, которые должны вести себя одинаково. В контексте этой техники, каждый класс эквивалентности содержит входные данные, которые должны привести к одному и тому же поведению программы. При тестировании выбираются представители каждого класса эквивалентности в качестве тестовых данных для выполнения тестов

- *Причинно-следственный анализ*. Этот метод позволяет проверить все возможности системы, а также обнаружить баги и улучшить техническую документацию продукта. Суть этой техники заключается в том, чтобы исследовать влияние определенных входных данных или действий на поведение программы и опередить, какие входные данные или действия могут привести к определенным результатам или ожидаемому поведению программы.

- *Попарное тестирование*. Этот подход основан на большом количестве входных параметров. Чем больше параметров, тем больше вероятность ошибки. Цель как специалиста по тестированию — сократить количество тест-кейсов до оптимального (позволяет обнаружить максимум ошибок без избыточных проверок). Суть его заключается в том, что каждое значение каждого проверяемого параметра должно быть протестировано на взаимодействие с каждым значением всех остальных параметров.

- *Предугадывание ошибок*. Техника используется для предварительной оценки и обнаружения потенциальных проблем пользовательского интерфейса или дизайна. Цель этой техники - предположить, какие ошибки или непроизводительные детали могут возникнуть при использовании продукта, и предпринять меры для их предотвращения или устранения.

# Время проведения тестирования

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Наименование работ** | | **Время (час)** |
| 1 | Определение и изучение частей системы для тестирования | | 4 |
| 2 |  | |  |
| 3 | Создание тестовых артефактов | сценарии тестирования | 8 |
| 4 | тест - кейсы | 26 |
| 5 | Проведение тестирования | | 4 |
| 6 | Фиксация результатов | | 2 |
| 7 | Составление отчета по результатам тестирования | | 2 |
| **Всего** | | | **46** |