Geekbrains

Инженер по тестированию. Цифровые профессии.

Функциональное тестирование лендинга Coffee.

http://cf.pulsarproduction.ru/

Тринченко М.А.

Санкт-Петербург, 2023

**Оглавление**

Введение ……………………………………………………………………………….….…2

Глава 1. Тестирование веб-приложений………………………………………………..….4

* 1. Сущность веб-приложения. Отличие лендинга от стартовой страницы сайта.…....5
  2. Тестирование программного обеспечения. Принципы тестирования. Критерии качества программного обеспечения. Виды тестирования…………………….….…6
  3. Жизненный цикл тестирования. Тестовая документация: план тестирования, чек-лист, тест-кейс, баг-репорт, отчет о тестировании………………………………...….9
  4. Техники тест-дизайна, их назначение. Роль технического задания в тестировании веб-приложений…………………………………………………………………….…..11

Глава 2. Тест-дизайн и подготовка к тестированию веб-приложения. Тестирование веб приложения по плану и составление отчетности…………………………….………..….17

2.1. Анализ ТЗ, составление mind-map требований. Дефекты на требования…….……17

2.2. Составление тест-плана………………………………………………………….….…18

2.3. Составление чек-листа……………………………………………………………...….19

2.4. Составление тест-кейсов……………………………………………………………….19

2.4.1. Классы эквивалентности………………………………………………………..……20

2.4.2. Попарное тестирование Pairwise…………………………………………………….21

2.4.3 Граничные значения и доменный анализ……………………………………………22

2.4.4. Другие техники тест-дизайна………………………………………………………..23

2.5. Процесс тестирования…………………………………………………………………23

2.5.1. Выполнение чек-листа и тест-кейсов, заведение баг-репортов……………………23

2.5.2. Составление отчета о тестировании…………………………………………………25

Заключение…………………………………………………………………………………..26

Список литературы…………………………………………………….……………………28

Приложения……………………………………………………………………...…………..30

**Введение**

Темой настоящей дипломной работы было выбрано функциональное тестирование лендинга Coffee по адресу <http://cf.pulsarproduction.ru/>.

Важным аспектом при тестировании веб-приложений является плотность покрытия тестами программного продукта. Для этого существует множество подходов, типов и целей тестирования. Тестирование программного продукта (обнаружение и исправление ошибок, контроль) является неотъемлемой частью качественного ПО. Актуальность данной работы вытекает из необходимости повышения качества тестирования, а также уменьшения трудовых и финансовых ресурсов, которые на это расходуются. Стремительный рост количества и сложности разрабатываемых лендингов стал толчком для появления ресурсов и приложений, которые позволяют структурировать и облегчить процесс тестирования.

Выбор бы обоснован в первую очередь наличием полного технического задания (ТЗ) к лендингу, без которого полноценное тестирование представляется затруднительным, а также актуальность проблематики тестирования лендингов в условиях растущего спроса на такого рода сайты со стороны компаний, которые ранее функционировали без сайтов, но пришли к выводу о необходимости быть представленными в Интернете, и растущего предложения со стороны увеличивающегося количества специалистов, которые способны создавать лендинги.

Целью настоящей работы является тестирование лендинга, выявление дефектов, применение различных техник тест-дизайна для тестирования лендинга и формирование выводов о том, какие инструменты и техники могут быть наиболее актуальны в данной работе. Объектом исследования выступает процесс тестирования лендинга. Предметом исследования являются методы, инструменты и техники тестирования для выявления и устранения ошибок в реализации лендинга.

Для достижения цели были сформулированы следующие задачи:

- исследовать методы тестирования.

- провести анализ существующих методов тестирования.

- выбрать самые эффективные способы тестирования лендинга.

- определить недостатки тех или иных техник и способов тестирования.

- протестировать лендинг.

- составить тестовую документацию.

План работы представляется следующим:

1. Изучить и описать лендинг.
2. Рассмотреть основные виды и техники ручного тестирования.
3. Рассмотреть основные принципы составления тестовой документации, а также приложения, реализующие эти принципы.
4. Составить план тестирования.
5. Выполнить тестирование лендинга, завести баг-репорты и составить отчеты о тестировании.
6. Сделать выводы о применимости тех или иных техник тест-дизайна. Выявить техники, наиболее актуальные в тестировании лендингов.

Дипломный проект будет делаться по специализации Инженер по тестированию. Проект будет решать проблему тестирования лендингов на стадии их приобретения и внедрения заказчиком, с целью выработки инструментария позволяющего протестировать лендинг, и в то же время нетребовательного к ресурсам и квалификации тестировщиков. Для выполнения проекта будут применяться системы управления тестированием (TMS) Qase.io и Jira, инструменты разаботчика браузера Chrome DevTools. В ходе выполнения проекта предполагаются следующие роли:

1. Тест-дизайнер – осуществляет составление плана тестирования и выбор инструментов.
2. Тестировщик – производит тестирование по плану, заводит баг-репорты, составляет отчеты.
3. Возможно принятие роли аналитика или менеджера проекта, там, где требуются их решения.

**Глава 1. Тестирования веб-приложений.**

* 1. **Сущность веб-приложения. Отличие лендинга от стартовой страницы сайта.**

Веб-приложения это приложения, с которыми пользователь взаимодействует в сети Интернет посредством браузера. Такие приложения представляют собой графические страницы, базовая структура которых определяется с помощью языка разметки HTML, модифицируется с помощью таблиц стилей CSS и дополняется функционалом и анимационными элементами с помощью скриптового языка JavaScript. Веб-приложение относится к фронтэнду, который будучи загруженным на сервере, в основном функционирует в клиентских машинах пользователей в среде наиболее популярных операционных систем (Windows, MacOS, Linux) и веб-браузеров (Internet Explorer, Chrome, Firefox, Opera, Edge). Не менее важной средой функционирования веб-приложений являются клиентские мобильные устройства на основе iOS или Android. Многие из инструментов разработки фронтэнда достаточно просты в освоении и позволяют создавать свои приложения даже людям с невысокой квалификацией, хотя профессиональные разработчики могут добиваться более значительных результатов во фронтэнде.

Многие приложения кроме фронтэнда имеют также бэкенд – серверную часть, в которой, как правило, реализуется сложная бизнес-логика веб-приложений, а также взаимодействие клиентов через веб-приложения с базами данных компании. В силу разности задач, бэкенд в корне отличается от фронтэнда в части используемых приложений, инструментов и языков программирования. В основном используются среды (Apache, NGINX, BSD), и языки программирования (PHP, Java, C++, Perl) ориентированные на производительность, масштабируемость, безотказность, безопасность и максимальный контроль над исполнением программ, либо просто сложившиеся исторически, притом, что графический интерфейс может ограничиваться командной строкой. Как следствие, бэкенд требует высокой квалификации персонала и обычно недоступен неподготовленному пользователю.

Фронтенд и бэкенд взаимодействуют в рамках клиент-серверной архитектуры (посылается HTTP-запрос – возвращается HTML-страница) или в рамках архитектуры AJAX (построение страницы происходит на клиенте, а обмен данными с сервером в фоновом режиме в виде XML или JSON сообщений), посредством программных интерфейсов приложений API. Суть API состоит в обмене стандартизированными сообщениями между программами, которые позволяют передавать информацию, в согласованном в стандарте конкретного протокола API виде. Наиболее популярные протоколы это SOAP API, использующий довольно громоздкие файлы XML (по своей структуре напоминающие веб-страницы HTML), и REST API, использующий гораздо более элегантные json-файлы, содержащие информацию в виде упорядоченных множеств (строк, списков и словарей).

Приближаясь к теме нашей работы, установим, что же такое лендинг, и чем он выделяется среди других веб-приложений. Слово «лендинг» происходит от английского «landing» (что означает «посадка»), поэтому их еще называют «посадочные страницы». Смысл этого названия в том, что страница служит для привлечения и установления первоначального взаимодействия заинтересовавшихся потенциальных клиентов, заходящих на лендинг, и использующей лендинг компании. Т.е. через лендинг происходит «посадка» потенциальных клиентов на компанию. Из этого мы можем определить, что целевой аудиторией и пользователями лендингов являются а) компании, желающие привлечь новых клиентов и б) потенциальные клиенты, как правило, находящиеся в процессе поиска и выбора контрагента для решения своих задач. Таким образом, ключевыми бизнес-задачами лендинга можно назвать:

1. Формирование заинтересованности у потенциальных клиентов компании, зашедших на лендинг.
2. Быстрое и эффективное установление контакта между компанией и клиентом, как правило, посредством простейшей регистрации, сбора контактных данных и общения в чате.
3. Создание положительного образа компании и информирование о ее деятельности широкого круга лиц.

В силу специфики этих задач, лендинг, который по исполнению является веб-страницей, отличается от внешне похожих на него головных страниц многостраничных сайтов. Основное отличие заключается в том, что лендинг, как правило, одностраничный, и вся информация в нем подается сразу и с упором на формирование первоначальной заинтересованности, нежели на развитый функционал и систематизированное постраничное изложение информации на стандартных сайтах. Функциональную же часть наоборот стараются делать максимально легкой и непритязательной. В целом для лендинга считается достаточным только лишь получение контактных данных заинтересованного клиента. Это приводит к тому, что лендинг минимально взаимодействует с бэкендом, и может в этом плане быть самодостаточным.

Применительно к тестированию такие особенности лендингов определяют приоритеты в их тестировании. Это в первую очередь:

* Внешний вид и привлекательность.
* Удобство и исправность инструментов взаимодействия (регистрации, форм обратной связи, чата).
* Отображение на широком спектре устройств.
  1. **Тестирование программного обеспечения. Принципы тестирования. Критерии качества программного обеспечения. Виды тестирования.**

Тестирование это процесс проверки заданного объекта с помощью тестов. Тест это некоторая предварительно формализованная единица информации, предполагающая совершение некоторого действия над объектом и сопоставления полученного результата с предполагаемым в тесте. В тестировании всегда имеется тестируемый объект, обладающий некоторой степенью неопределенности реальных свойств, список проверок, регламент проведения тестирования, ожидаемый и фактический результаты проверок, которые в силу неопределенности свойств объекта могут не совпадать между собой. Тестирование имеет несколько основополагающих принципов, задающих общую направленность используемых в тестировании подходов и методов:

1. ***Невозможность исчерпывающего тестирования*** – исчерпывающее тестирование, т.е. тестирование каждого значения каждого параметра тестового объекта и всех возможных комбинаций этих значений представляется неподъемной и экономически нецелесообразной задачей, поэтому основное направление деятельности в тестировании это сокращение количества тестов при обеспечении заданного уровня тестового покрытия.
2. ***Тестирование демонстрирует наличие дефектов, а не их отсутствие*** – тестирование может выявить дефекты, но не может стопроцентно доказать их отсутствие в тестовом объекте.
3. ***Заблуждение об отсутствии ошибок*** – любой объект с высокой долей вероятности имеет дефекты разной величины – от незаметных, до очевидных. Задача тестирования заключается не в выявлении всех дефектов, а в первоочередном выявлении наиболее важных.
4. ***Тестирование на более ранних стадиях сохраняет время и деньги*** – чем раньше начинается тестирование продукта, тем раньше могут быть выявлены и исправлены дефекты, которые на поздней стадии потребовали бы затрат на устранение последствий, внесения существенных изменений или полной переделки уже готового продукта. Таким образом, сохраняются ресурсы и время, которые иначе были бы потрачены на работу в неверном направлении.
5. ***Дефекты имеют склонность к скоплению в одном месте*** – как правило, тестовый объект состоит из некоторых частей или модулей, они из них могут простые, другие сложные, могут быть различны условия их создания, новизна и надежность реализуемых в них методов. Все это приводит к тому, что дефекты распределяются между частями объекта неравномерно, имея тенденцию к накоплению в более сложных, инновационных или некачественно выполненных. Поэтому приоритет в тестировании следует отдавать более сложным частям, а обнаружение дефекта в какой-либо части объекта может говорить о том, что там же могут быть обнаружены и другие дефекты, и, следовательно, требуется более тщательное тестирование.
6. ***Тестирование зависит от контекста*** – процесс тестирования и применяемые в нем подходы и методы зависят от контекста: целей и свойств тестового объекта, требований к качеству его тестирования, уровня подготовки тестовой команды, доступных инструментов, выделенных сроков. Поэтому в процессе тестирования важно проводить анализ контекста, определять стратегию и план тестирования.
7. ***Парадокс пестицида*** – трудно найти новые дефекты в том же объекте, если постоянно применять к нему одни и те же тесты. Ошибки будут вымываться в часто проверяемых местах, но оставаться в неохваченных тестами. Для выявления новых дефектов в уже протестированном продукте следует варьировать применяемые к нему тесты, использовать новые методы и техники тестирования.

Тестирование программного обеспечения это проверка соответствия реального и ожидаемого поведения ПО с целью выявления дефектов и оценки степени ее соответствия ПО требованиям к нему, и, в конечном счете, определения качества ПО. Выделяют следующие критерии качества:

1. ***Функциональность*** – полнота, правильность и целесообразность выполнения программным продуктом своих функций.
2. ***Эффективность*** – производительность продукта по затратам времени, ресурсов и обрабатываемому объему.
3. ***Надежность*** – степень завершенности, доступности, отказоустойчивости продукта и его восстанавливаемости после сбоев.
4. ***Удобство пользования*** – все, что связано с интуитивной понятностью, скоростью обучения, эргономикой, защитой от ошибок пользователя, эстетичностью и доступностью программного продукта пользователю в плане предъявляемых требований.
5. ***Поддерживаемость*** – то насколько продукт может быть адаптирован разработчиком или пользователем к изменениям. Сюда входит модульность архитектуры, степень повторного использования компонентов, модифицируемость и тестируемость ПО.
6. ***Переносимость*** – т.е. возможность переноса и использования продукта в разных окружениях, включая адаптивность к разным средам, простоту установки и заменяемость.
7. ***Безопасность*** – применительно к ПО подразумевает в основном стойкость к несанкционированному доступу. Сюда включается конфиденциальность работы с данными, целостность системы безопасности, устойчивость к подлогу, возможность установления ответственности и верифицируемость подлинности компонентов.
8. ***Совместимость*** – степень взаимодействия продукта с окружением, сюда относится сосуществование, т.е. насколько ПО не мешает своему окружению и совместимость, т.е. степень и качество обмена информацией с окружением.

Для проверки различных аспектов качества ПО существуют различные виды

тестирования:

1) ***Функциональное тестирование*** – тестирование реализации функций ПО. Может проводиться в зависимости от знакомства с внутренним строением продукта по методу белого ящика (есть понимание внутреннего устройства ПО), черного ящика (объект воспринимается как неделимая единица с неопределенным повелением) и серого ящика (известны компоненты ПО и протоколы их взаимодействия). По критерию повторности можно выделить первичное тестирование, регрессионное тестирование (повторное тестирование функций после обновления ПО) и повторное тестирование (тестирование после устранения дефектов). Также функциональное тестирование может быть статическим (ревизия программного кода и проверка графического макета приложения) и динамическим (запуск готового приложения), позитивным (на корректных данных) и негативным (на некорректных данных), альфа (внутри компании) и бета (с привлечением ограниченного контингента пользователей) тестированием. Тестирование может проводиться либо по тест-кейсам или исследовательским методом. Одной из распространенных техник являются смоук-тесты или тесты на дым, т.е. минимальный набор тестов функционала, позволяющий выявить очевидные серьезные дефекты в ПО.

2) ***Нефункциональное тестирование*** – это тестирование того, каким образом программный реализует свой функционал, т.е. насколько он удобен, безопасен, производителен и устойчив. К нему обычно относятся следующие виды тестирования:

- Тестирование пользовательского интерфейса и тестирование удобства использования. Целью этих видов тестирования является определение удобства графического интерфейса пользователя и общее удобство пользования продуктом.

- Тестирование производительности, включающее нагрузочное тестирование (тестирование поведения продукта под нагрузкой), стрессовое тестирование (тестирование под избыточной нагрузкой), опрокидывающее тестирование (намеренное провоцирование сбоев и определение восстанавливаемости ПО после них), объемное тестирование (тестирование ПО с использованием объектов и данных большого объема)

- Тестирование безопасности – определение стойкости ПО к несанкционированному доступу.

- Конфигурационное тестирование – определение поведения ПО в системах с разными аппаратными и программными конфигурациями.

- Установочное тестирование – тестирование поведения программного продукта в процессе инсталляции.

Все перечисленные виды тестирования имеют своей целью проверить тот или иной специфический аспект или условие работы программного продукта. Конкретный набор применяемых видов тестирования выбирается исходя из целей тестирования и имеющихся возможностей его проведения.

* 1. **Жизненный цикл тестирования. Тестовая документация: план тестирования, чек-лист, тест-кейс, баг-репорт, отчет о тестировании.**

Жизненный цикл тестирования это подход к организации процесса тестирования, согласно которому выделятся стадии жизненного цикла процесса тестирования и формализуются параметры каждой стадии. Обычно выделяют следующие этапы жизненного цикла тестирования:

1. ***Работа с требованиями*** – на этой стадии происходит сбор и первоначальная декомпозиция требований к тестовому объекту. Выработка стратегии и критериев тестирования.
2. ***Планирование тестирования*** – по результатам анализа требований составляется тест-план, в котором определяются необходимые действия и планируются ресурсы.
3. ***Тест-дизайн*** – процесс составления конкретных тестовых заданий (чек-листов, тест-кейсов) согласно тест-плану.
4. ***Настройка среды тестирования*** – подготовка необходимого аппаратного и программного обеспечения, настройка тестового стенда.
5. ***Выполнение тестов*** – выполнение составленных тестов, заведение баг-репортов.
6. ***Закрытие тестирования*** – составление финальной отчетности по тестированию, выработка рекомендаций по улучшению тестируемого объекта и процесса тестирования.

В процессе тестирования важную роль играет составление тестовой документации, к которой обычно относят:

***Тест-план*** – документ, формализующий все аспекты предстоящего процесса тестирования. В нем определены: тестовый объект, тестируемые требования к нему, тестовое покрытие объекта, виды проводимых тестов, используемые инструменты и необходимые ресурсы, стратегия и порядок действий в процессе тестирования, распределение ролей и задач, оценка затрат времени, критерии начала и завершения тестирования, получаемые после проведения тестирования результаты. Тест-план может составляться на весь тестовый проект либо на отдельные подзадачи проекта. Составлением тест-плана занимается руководитель команды тестирования, и согласует его с другими заинтересованными лицами: заказчиком, менеджером проекта, руководителем команды разработки и т.д.

***Чек-лист*** – список относительно слабо регламентированных проверок, предполагающий определение тестировщиком соответствия тестового объекта отдельным пунктам требований. Может выполняться достаточно произвольно, что обуславливает высокую вариативность тестовых данных и результатов. Чек-листы характеризуются высокой скоростью составления и исполнения, широким охватом тестового объекта, но в то же время часто имеют недостаточную глубину проверки отдельных элементов. Чем сложнее элемент тестового объекта и способы взаимодействия с ним, тем менее применимы чек-листы к его проверке.

***Тест-кейс*** – строго регламентированная проверка, описывающая все аспекты проводимого теста: предусловия, тестовое окружение, шаги, тестовые данные, ожидаемые результаты отдельных шагов и теста в целом. Это обеспечивает высокую степень единообразия проверки разными тестировщиками. Тест-кейсы более сложны в разработке и исполнении, поэтому составляются по различным техникам тест-дизайна, задача которых минимизировать количество тестов при обеспечении заданного в тест-плане тестового покрытия объекта. Как правило, тест-кейсы характеризуются большой глубиной проверки элементов тестового объекта и вариантов взаимодействия с ними, но в меньшей степени подходят для широкого охвата тестируемого объекта ввиду своей трудозатратности.

***Баг-репорт*** – формализованный отчет о конкретном найденном дефекте. В баг-репорте обычно указываются: название, характеризующее найденный дефект, описание дефекта, аппаратное/программное окружение, предусловия и шаги по воспроизведению, ожидаемый результат и фактический результат, приоритет устранения и критичность дефекта, также может даваться ссылка на тест-кейс, в котором дефект был обнаружен. Баг-репорты это один из основных результатов тестирования, позволяющий разработчикам непосредственно улучшать качество программного продукта. Основная характеристика баг-репорта это четкая идентификация проблемы и способов воспроизведения дефекта.

***Отчет о тестировании*** – документ, содержащий сведения о выполненных действиях и достигнутых результатах. В нем дается анализ проведенной работы и оценка продукта. Отчет может быть промежуточным и окончательным и составляться для разных получателей, что может потребовать различных акцентов и разрезов представляемой в отчете информации. Обычно в отчете указываются следующие сведения: тестовая команда, сроки выполнения, описание процесса тестирования и изменений в нем, общее количество тест-кейсов и процент пройденных, критические и блокирующие дефекты, оценка степени готовности тестового объекта.

* 1. **Техники тест-дизайна, их назначение. Роль технического задания в тестировании веб-приложений.**

Как при создании, так и при тестировании веб-приложений сложно переоценить роль технического задания. Техническое задание (далее ТЗ) описывает все важные с точки зрения заказчика особенности внешнего вида и функционирования веб-приложения. Таким образом, ТЗ фиксирует состояние «как должно быть» и служит отправной точкой как для разработки приложения, так и для его последующего тестирования. ТЗ веб-приложения как правило составляется бизнес-аналитиком, задача которого выявить наиболее важные с точки зрения целевой аудитории требования к создаваемому продукту и исключить менее важные, давая необходимую свободу разработчикам.

Тестирование приложений также опирается на ТЗ как на краеугольный камень в деле выявления дефектов тестируемого программного продукта и в конечном счете в установлении степени готовности продукта к релизу. Не смотря на это, само ТЗ также нуждается в тщательном анализе перед началом процесса тестирования. Такой анализ позволит выявить неполноту, недостаточно понятные пункты и неразрешимые противоречия, которые могли бы привести к затруднениям непосредственно в процессе тестирования. Перед началом тестирования производится сбор материалов, содержащих требования к продукту (бизнес-требований, ТЗ на разработку, макетов верстки, пользовательских сценариев) и их сопоставление и декомпозиция на атомарные требования и их взаимосвязи, которые затем послужат основой для создания тест-кейсов. Также анализ требований к продукту имеет своей целью запланировать объем работ и тестовое покрытие продукта, определить стратегию тестирования и наиболее перспективные в конкретном случае техники тест-дизайна. Основными характеристиками требований для проведения тестирования являются:

1. Атомарность – требование должно быть декомпозировано на элементарные неделимые выражения, которые затем становятся основой для проверок.
2. Однозначность – требование не должно допускать различных трактовок.
3. Полнота – требование должно подробно описывать все аспекты объекта или процесса, которые могут пригодиться в процессе тестирования.
4. Необходимость – в основном означает определенный уровень краткости и отсутствие дублирования.
5. Непротиворечивость – пункты требований не должны противоречить другим пунктам.
6. Выполнимость – требование должно быть реалистичным и реализуемым.
7. Тестируемость – должен существовать разумный способ проверки требования.

Перед началом процесса тестирования следует убедиться, что требования максимально соответствуют приведенным выше характеристикам и при необходимости уточнить неоднозначные пункты и устранить противоречия, обычно методом опроса заказчика, разработчика или аналитика, которые имели отношение к составлению и реализации требований.

Тестирование веб приложений обычно производится методом черного ящика, т.е. тестировщик не работает с исходным кодом приложения, а суть тестирования сводится к покрытию всех возможных вариантов входных параметров и анализу результатов работы приложения с каждым таким вариантом. Для создания тест-кейсов с целью ухода от простого перебора всех комбинаций входных значений в этом случае могут быть использованы следующие техники тест-дизайна:

1) Разбиение на классы эквивалентности – подразумевает выбор линейных (количественных и изменяющихся в виде линейной последовательности), и нелинейных (представляющих собой набор качественных или неупорядоченных линейно количественных значений) параметров требований и разбиение множества возможных значений параметров на классы, в рамках которых поведение тестового объекта предполагается единообразным. Таким образом, мы получаем такие группы данных, из которых для целей тестирования достаточно выбрать лишь одно значение на каждую группу. В данном случае мы предполагаем, что если одно значение в группу вызывает ошибку или проходит тест, то и остальные значения данного класса поведут себя так же. Классы могут быть валидными, для которых предполагается прохождение проверки, и невалидными, для которых мы заранее предполагаем провал проверки. Обычно классы эквивалентности выбираются либо в виде диапазонов значений для линейных классов, или в виде наборов входных данных, обладающих каким-то общим свойством, для нелинейных классов. Техника классов эквивалентности применима достаточно широко и практически для любых видов данных, при условии, что они могут быть сгруппированы по какому-то признаку.

2) Анализ граничных значений – техника тест-дизайна, работающая с линейными классами эквивалентности. Необходимость ее применения обусловлена тем, что при указании интервалов непрерывных величин концы этих интервалов могут как включаться, так и не включаться в интервал. Это в свою очередь создает риск некорректного поведения тестового объекта с вводами, соответствующими границам определенных ранее классов эквивалентности. Для проверки такого поведения на границе каждого линейного класса эквивалентности выбирается само граничное значение и пара наиболее близких значений вокруг нее справа и слева. Это позволяет протестировать корректность перехода от одного поведения тестового объекта к другому при переходе через границу класса эквивалентности.

3) Попарное тестирование pairwise – техника тест-дизайна, применяемая для тестирования нелинейных классов эквивалентности. Наилучшие результаты показывает при тестировании совместной работы большого количества параметров, у каждого из которых имеется ограниченный набор значений. Это могут быть чекбоксы, переключатели и выпадающие списки. Предположение, лежащее в основе этой техники, состоит в том, что ошибка работы тестового объекта наиболее вероятно будет вызвана сочетанием каких-либо двух параметров, нежели сочетанием из трех и более. Исходя из этого предположения, считается достаточным протестировать работу всех уникальных пар значений параметров, что уменьшает общее количество тестов по сравнению с тем, если бы мы попытались проверить работу всех уникальных сочетаний всех параметров. Еще одной особенностью техники pairwise является то, что тестируются только позитивные сценарии. При необходимости негативные сценарии вводятся дополнительными тестами, с условием не более одного негативного значения параметра на тест.

4) Доменный анализ – является дополнением к техникам классов эквивалентности и граничных значений. Заключается в разбиении множества валидных значений линейных параметров на домены, аналогичные классам эквивалентности. Для каждого домена выбирается среднее значение, граничное внешнее и граничное внутренне с обеих границ интервала, в итоге получается по 5 значений (тест-кейсов) на домен. Отличие от техники классов эквивалентности и граничных значений состоит в том, что мы рассматриваем только валидные домены и их окрестности. Для добавления нелинейных параметров в набор тест-кейсов позитивные проверки доменов прии необходимости дополняются позитивными проверками по технике попарного тестирования и другими позитивными проверками. Негативные проверки добавляются аналогично технике попарного тестирования, т.е. в виде дополнительных тест-кейсов, в каждом из которых должно быть не более одного невалидного значения параметра.

5) Техника диаграмм и таблиц переходов состояний – техника тест-дизайна для тестирования объектов и систем с большим количеством состояний и возможных переходов между ними. Для реализации этой техники строится диаграмма, на которой указываются вход (начало работы), выход (окончание работы) и все возможные состояния объекта в промежутке между ними. Существующие переходы между состояниями на диаграмме изображаются в виде стрелок с обозначением события, обуславливающего переход и действия, совершаемого при таком переходе. Для оставления тест-кейсов на построенной диаграмме выбираются различные пути перехода от точки входа к точке выхода таким образом, чтобы каждый переход (каждая стрелочка на диаграмме), была проверена хотя бы один раз.

Для лучшей систематизации данных состояния и переходы состояний можно занести в таблицу вида «исходное состояние/событие/действие/новое состояние» в отношении каждое исходное состояние к каждому событию, т.е. для каждого состояния предусматривается потенциальное наступление любого события. Далее дозаполняются поля «действие» и «новое состояние»: если сочетание исходного состояния и события должно вызывать действие и переход в новое состояние, они указываются в соответствующих полях, если нет, то действие не указывается, а новое состояние указывается таким же, как исходное. Для тест-кейсов из таблицы выбираются как существующие переходы, для которых указано действие и новое состояние, так и не приводящие к смене состояния сочетания состояний и событий, которые могут причинить серьезный ущерб, если бы такой переход был возможен.

6) Таблицы принятия решений – применяются для тестирования приложений, бизнес-логика которых предусматривает различные режимы работы для различных значений некоторого набора условий. Например, банки или страховые компании могут предлагать разные ставки в зависимости от возраста, имущественного положения, истории взаимодействия и лояльности клиента, что, как правило, находит отражение в веб-приложениях этих организаций. Для составления тест-кейсов в таблице комбинируются все сочетания значений условий, под ними указываются вызываемые каждым условием действия (например, коэффициенты или отказ/принятие), и затем в каждом сочетании условий по имеющимся действиям высчитывается окончательный результат работы приложения, который и будет ожидаемым результатом тест-кейса.

7) Тестирование по пользовательским сценариям – по данной технике тест-кейсы составляются как список шагов, которые пользователь может совершить в процессе использования приложения для достижения значимого для себя результата. Пользовательские сценарии могут быть, как предоставлены заказчиком, так и определены на основе анализа требований и описания работы приложения, например с помощью диаграмм переходов состояний. Для каждого шага определяется набор валидных и невалидных параметров, которые пользователь может ввести в приложение на каждом шаге. Исходя из сочетаний шагов и параметров, строятся позитивные и негативные сценарии взаимодействия пользователя с приложением. Для концептуального изображения пользовательских ролей и их возможных действий строятся специфические Диаграммы вариантов использования, которые показывают имеющихся в системе акторов и функций, предоставляемых им приложением.

Рассмотренные выше техники тест-дизайна можно отнести к сценарному тестированию, когда пред процессом тестирования составляются тест-кейсы (сценарии), по которым будет происходить взаимодействие тестироващика и тестируемого приложения. Несмотря на оправданность такого подхода, существуют ситуации, когда применение сценарного тестирования затруднительно, например, в условиях отсутствия документации, содержащей требования к приложению. Также сценарное тестирование может быть нецелесообразно, если тестируемый продукт небольшой, требуется лишь поверхностная информация о его работоспособности или имеется недостаточно времени на проведение полноценного тест-дизайна.

В перечисленных случаях применяется исследовательское тестирование – специфическая методика тестирования и тест-дизайна, применяемая в условиях слабого знакомства с тестовым объектом или необходимости быстрого проведения тестирования без предварительного составления тест-кейсов. При проведении исследовательского тестирования тестировщик, опираясь на свою экспертизу и доступные данные о продукте, сам определяет, что и как будет проверять. У такого подхода есть как достоинства, например, возможность выявления нестандартных дефектов, низкие затраты времени, отсутствие потребности в подробной документации на продукт, так и недостатки, например слабое покрытие тестового объекта, отсутствие подробной тестовой документации, которая могла бы пригодиться в длительном периоде разработки, высокие риски при тестировании критически важных, например, банковских, приложений. Исследовательские тестирование может проводится, как ad hoc, так и с некоторой степенью регламентации, например с документированием шагов и разработкой тест-кейсов на месте и с ограничением по времени. Для проведения исследовательского тестирования могут использоваться путеводители (указывающие, что следует проверить и в какой очередности), чит-листы (наборы тестов для приложений сходной функциональности) и эвристики (способы интуитивного принятия решений о тестовых данных). Отдельные элементы исследовательского тестирования могут применяться и в сценарном тестировании, там, где свобода тестового сценария допускает некоторую степень импровизации и интерпретации результатов тестировщиком.

**Глава 2. Тест-дизайн и подготовка к тестированию веб-приложения. Тестирование веб приложения по плану и составление отчетности**

**2.1. Анализ ТЗ, составление mind-map требований. Дефекты на требования.**

**Начальным этапом жизненного цикла тестирования является анализ требований. Он позволит нам получить первоначальное представление о тестовом объекте и его характеристиках, выявить несоответствия в требованиях и спланировать работу по тестированию. ТЗ на лендинг Coffee дано нам в простом текстовом описательном формате с иллюстрациями. В представленном ТЗ имеются следующие разделы:**

* ***Титульная страница* со ссылкой на лендинг, логином и паролем**
* ***Вводная часть:* о проекте, назначение документа, термины и аббревиатуры**
* ***Технический порядок взаимодействия Исполнителя с Заказчиком***
* ***Функциональные требования:***

**- Макет в figma для десктопной и мобильной версии приложения.**

**- Перечень разделов лендинга с описанием требований к содержимому.**

* ***Нефункциональные требования:* здесь перечислены требования к доступности, масштабируемости, отказоустойчивости, скорости отклика, хранению персональных данных, истории событий, контенту.**
* ***Технические требования:* здесь есть как вопросы взаимодействия Исполнителя и Заказчика, так и системные требования к лендингу и его окружению**

**.**

**Кроме текста ТЗ, у нас имеется макет веб-страницы в figma для десктопного отображения и для мобильного устройства. Макет в процессе тестирования может выполнять двоякую функцию. В первую очередь, макет, как и ТЗ, является источником требований для создания и тестирования сайта. И во-вторых, макет является статической имитацией готового приложения, поэтому он может быть использован как тестовый объект для статического тестирования по требованиям ТЗ, что позволит выявить графические баги на ранней стадии разработки и предотвратить их переход в готовый продукт.**

**Для первоначальной декомпозиции требований и выявления их взаимосвязей попробуем составить мнемоническую карту mind-map функциональных требований в разрезе элементов интерфейса с использованием платформы для распределенной работы команд Miro.com. Получившаяся mind-map довольно громоздкая, поэтому будет приведена лишь ссылка на нее (также примерный вид дан в Приложении 1):** <https://miro.com/app/board/uXjVMr2sw7s=/?share_link_id=780545884433>

**Что нам дала mind-map? Во-первых, создав ее мы стали лучше понимать продукт, во-вторых ее уже можно использовать для составления чек-листов и тест-кейсов как замену ТЗ, в-третьих мы начинаем видеть недостатки и противоречия в ТЗ, и наконец, на ней же можно отмечать покрытие функционала тестами.**

По результатам составления mind-map у нас могут возникнуть какие-то вопросы относительно требований в ТЗ. Как и в любом другом документе в нем могут содержаться ошибки, неточности и противоречия. Поэтому перед началом тестирования приложения (или даже перед началом разработки) имеет смысл протестировать само ТЗ на предмет его понятности пользователю (разработчику, тестировщику). Сделаем это в виде простого чек-листа, пункты, вызывающие у нас вопросы отметим и определим, что в них не так. Результат проверки требований можно увидеть в таблице по ссылке <https://docs.google.com/spreadsheets/d/1Re1nh9TiQ249dxiBaNPMXFlARUOx27BOnlxteivyZcw/edit#gid=0> или в Приложении 2.

Как мы видим, несмотря на наличие подробного ТЗ и макета, в них имеется масса нестыковок, неполноты и противоречий. Возможно, имеет смысл провести статическое тестирование макета лендинга по ТЗ. Исходя из анализа дефектов в требованиях, мы можем скорректировать процесс тестирования, исключив или пропустив пункты требований, которые не могут быть однозначно поняты тестировщиком.

**2.2. Составление тест-плана.**

Планирование является следующим этапом в жизненном цикле тестирования. На этом этапе мы составим тест-план, в котором будет описано, что и как мы собираемся тестировать. Тест-план является основным руководящим документом в тестировании и позволяет заранее спланировать объем, порядок выполнения, затраты времени и ресурсов на проводимое тестирование и ожидаемые выходные данные. Также задачей тест-плана является дать представление о тестовом покрытии требований и характеристик тестового объекта. В плане тестирования нам важно определить следующие пункты:

1) Вводные данные: цель тестирования, контекст тестирования, объем тестирования, тестовый проект и имеющиеся документы

2) Требования, которые мы собираемся протестировать.

3) Стратегия тестирования: виды и техники тестирования, используемые инструменты

4) Рекомендуемые ресурсы: роли, тестовая система.

5) Временные рамки и промежуточные цели.

6) Материальные результаты или т.н. тестовые артефакты: то, что мы создадим по ходу тестирования – тестовые модели, тестовые отчеты, баг-репорты.

Попробуем составить предварительный план нашей работы. План доступен в Приложении 3

**2.3. Составление чек-листа.**

В качестве начального шага в нашей работе по тестированию лендинга, попробуем составить чек-лист на основе ТЗ и mind-map и имеющегося макета в figma. Работу будем проводить в TMS Qase.io. Пункты чек-листа сгруппируем по разделам лендинга. Приоритет будет отдаваться буквальной интерпретации требований ТЗ. В чек-листе будем указывать требования, которые могут быть протестированы при осмотре сайта и простом взаимодействии с ним, для более сложных проверок будем составлять тест-кейсы.

При заведении пунктов чек-листа в Qase.io, составленный ранее mind-map существенно сэкономил время, т.к. можно просто копировать ветки mind map в названия тестов чек-листа щелчком мыши. Если формулировать пункты mind-map целыми предложениями вида «X содержит Y», то правки при перенесении в чек-лист будут минимальны. Полученный нами чек-лист доступен по ссылке: <https://app.qase.io/project/CL?case=122&suite=8> (а также в Приложении 4)

**2.4. Составление тест-кейсов.**

В качестве отправной точки для тест-кейсов на функционал составим серию простых сценариев обхода страницы лендинга и проверки ее соответствия ТЗ. Каждый тест-кейс будет подразумевать переход в определенный раздел и его проверку аналогично чек-листу. Эту работу также проделаем в Qase.io. Результат доступен по ссылке <https://app.qase.io/project/DCTC?case=9&suite=1> а также в Приложении 5. Как мы видим, этот набор тест-кейсов во многом повторяет наш чек-лист, но в то де время подробнее описывает действия, которые нужно произвести. Также бросается в глаза, что взаимодействие с интерактивными формами происходит с одним предписанным набором данных, что не позволяет нам быть уверенными, что пользователи не столкнутся с трудностями, вводя другие произвольные данные.

Для составления более узконаправленных тест-кейсов выделим те разделы функционала лендинга, которые на наш взгляд являются технически сложными, требуют взаимодействия с пользователем и проверки широкого диапазона тестовых данных. Для этих разделов напишем тест-кейсы с использованием различных техник тест-дизайна. Это позволит нам сосредоточиться на поиске багов в критически важных элементах лендинга и в то же время сократить объем тестирования, не упуская из виду ничего важного. Какие техники тест-дизайна мы собираемся применить и в каких разделах? В первую очередь протестируем форму бронирования столика в разделе «Забронировать столик». Эта форма является ключевой при взаимодействии посетителя лендинга с компанией и содержит большое число интерактивных полей, от правильности функционирования которых зависит успешность выполнения лендингом своего предназначения.

Для составления тест-кейсов воспользуемся базовыми техниками тест-дизайна: выделение классов эквивалентности, составление попарных тестов, определение граничных значений и доменный анализ.

**2.4.1. Классы эквивалентности**

Техника разбиения на классы эквивалентности подразумевает разделение множества возможных тестовых значений каждого элемента на классы, внутри которых мы на основе нашего ТЗ можем предполагать одинаковый результат их исполнения. Это позволит нам тестировать всего одно значение из класса вместо множества, а поскольку они эквиваленты по получаемому результату, то мы сможем предполагать, что качество нашего тестирования не пострадает.

Выделим линейные и нелинейные значения в разделе «Забронировать столик», которые мы будем разбивать на классы и составим таблицу классов эквивалентности. Результат разбиения на классы эквивалентности приведен в Приложении 6 или по ссылке <https://docs.google.com/spreadsheets/d/1-W63_02GSObc7fW8PiyfSgKvfXXme2K_-pyi3Dbo6R8/edit?hl=ru&pli=1#gid=0> . В данном случае мы взяли только параметры, на которые были заданы ограничения, и у которых было разумно предполагать более одного класса эквивалентности. Для полей, у которых не была указана обязательность их заполнения в ТЗ, мы также выделили два класса эквивалентности в параметре обязательности, т.к. несмотря на отсутствие требований, они могут вести себя по-разному в заполненном и незаполненном виде. Также на классы эквивалентности были поделены форматы данных, там, где они указаны. На основании данного разбиения на классы эквивалентности можно непосредственно составить серию тест-кейсов, или использовать результат в техниках определения граничных значений и доменного анализа.

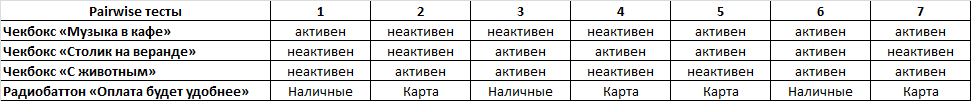
**2.4.2. Попарное тестирование Pairwise**

Составление тестов Pairwise обычно производится, когда имеется большое количество работающих совместно параметров, каждый из которых может принимать ограниченное количество значений. Целью такого тестирования является минимизация тестов с условием, что каждое значение параметра отработает в паре со всеми значениями всех остальных параметров. На примере лендинга Coffee можно выделить следующие переключаемые параметры:

* Чекбокс «Музыка в кафе»
* Чекбокс «Столик на веранде»
* Чекбокс «С животным»
* Радиобаттон «Оплата будет удобнее» «Карта/наличные»

Попробуем скомбинировать попарно тесты для этих полей. Можно сделать это вручную или воспользоваться веб-приложениями Pairwise PICT Online или Pairwise Tool. В результате у нас получается 7 тестов, закрывающих все возможные попарные комбинации:

Рис.1 Pairwise тесты, сгенерированные в приложении Pairwise Tool (лендинг Coffee)



Заметим, что все эти тесты позитивные. Если нам необходимо добавить негативные тесты, например, незаполненные обязательные поля, это можно сделать путем копирования произвольного попарного теста с добавлением строго одного невалидного значения параметра на каждый негативный тест. Попарные тесты также можно проводить отдельно или скомбинировать с тестами по техникам классов эквивалентности и граничных значений.

**2.4.3 Граничные значения и доменный анализ.**

Техника граничных значений является расширением техники классов эквивалентности для линейных параметров и позволяет протестировать значения не только непосредственно внутри каждого линейного класса эквивалентности, но и на границах этих классов. Необходимость такого тестирования обусловлена тем, что часто бывает затруднительно установить принадлежность граничного значения непрерывного множества к тому или иному классу, что может приводить к непредсказуемой работе приложения с этими значениями. Попробуем составить таблицу граничных тестовых значений для ранее выделенных линейных классов эквивалентности. Результат доступен в Приложении 7 или по ссылке: <https://docs.google.com/spreadsheets/d/1wRyqiDzBm3YWPbDAUraG8euTsi1rnm_S6DEV7XrFa5o/edit#gid=0>

Техника доменного анализа позволяет совместить тестирование нескольких линейных параметров по найденным граничным значениям классов эквивалентности и может быть дополнена тестами нелинейных параметров и попарными тестами. Это позволяет минимизировать общее количество тестов, не уменьшая покрытие тестами функционала. Попробуем составить таблицу тестов по технике доменного анализа. Для этого сначала сгруппируем граничные значения каждого линейного класса так, чтобы перебрать каждое значение каждого класса и при этом на один тест допускается не более одного невалидного значения. Полученная таблица изображена на Приложении 8.

Мы видим, что в таблице есть как позитивные, так и негативные проверки. Теперь нам нужно добавить попарные тесты так, чтобы каждый уникальный попарный тест приходился только на позитивные проверки граничных значений. Негативные проверки граничных значений дополняем любым скопированным (т.е. не уникальным) попарным тестом. Если попарных тестов оказывается больше, чем позитивных проверок граничных значений, дополняем лишние попарные тесты скопированными позитивными проверками граничных значений. Результат представлен в Приложении 9.

В завершение, нам нужно дополнить полученные тесты оставшимися полями. Для уже составленных нами тестов добавляем только валидные значения параметров полей. Невалидные и спорные с нашей точки зрения значения полей следует добавлять в виде дополнительных тестов справа, при этом снова выбирая только одно невалидное или спорное значение поля на весь тест. В заключение нужно сделать проверку всей таблицы на наличие не более одного негативного или сомнительного параметра на тест. Окончательная таблица тестов представлена в Приложении 10 или по ссылке: <https://docs.google.com/spreadsheets/d/1wRyqiDzBm3YWPbDAUraG8euTsi1rnm_S6DEV7XrFa5o/edit#gid=832444862>

Полученную таблицу тестов используем для создания набора тест-кейсов тестирования Формы бронирования столика, работу выполним в Qase.io. Полученный результат представлен в приложении 11 или по ссылке <https://app.qase.io/project/DCTC?previewMode=side&suite=2>

**2.4.4. Другие техники тест-дизайна**

Среди неохваченных нами техник тест-дизайна можно выделить диаграммы переходов и состояний, таблицы принятия решений и тестирование по пользовательским сценариям. Мы не использовали эти техники потому, что тестируемый нами лендинг не имеет сложной логики взаимодействия с пользователем, большого количества состояний, переходов и сценариев использования, что оправдало бы применение этих техник. Также можно упомянуть исследовательское тестирование, которое обычно применяется при отсутствии ТЗ или нецелесообразности строгой формализации процесса тестирования, например при смоук-тестах.

**2.5. Процесс тестирования.**

**2.5.1. Выполнение чек-листа и заведение баг-репортов.**

В качестве первой активности по тестированию лендинга Coffee попробуем выполнить составленный нами чек-лист в Qase.io. Подробный отчет о тест-ране по чек-листу доступен по ссылке:

<https://app.qase.io/public/report/97b902fa389ff0c615b9af868ebe3d3769a7239d>

В процессе прохождения чек-листа было проведено 129 тестов функционала лендинга, было обнаружено и заведено 73 бага. Общая длительность тест-рана составила 2 часа 39 минут. Скриншоты тест-рана приведены в Приложении 12. Список выявленных дефектов приведен в Приложении 13.

Анализ полученных в результате тест-рана чек-листа данных дал следующие результаты:

1. Всего 129 тестов чек-листа: Passed – 34 тестов (26%), Failed – 66 тестов (51%), Blocked – 29 тестов (23%).
2. Всего обнаруженных дефектов 72: Blocker - 1, Critical - 15, Major - 10, Normal - 5, Minor - 20, Trivial – 21.
3. Наименьшую готовность продемонстрировали разделы: «Вы найдете кофе на любой вкус», «Забронировать столик», «Футер».
4. По затратам времени наиболее трудоемким оказался раздел «Бронирования столика» - 2часа или более 75% времени тест-рана.

В целом тест-ран выявил большое количество багов, связанных, во-первых с низкой степенью готовности макета лендинга, во-вторых с низкой работоспособностью интерактивных форм и элементов.

Следующий тест-ран был проведен по набору тест-кейсов обхода разделов лендинга. Результат тест-рана доступен по ссылке <https://app.qase.io/run/DCTC/dashboard/2/a07a881995336fa42f66a0d53f7581ff652bad3a> и приведен в Приложении 14. В целом следует признать такой вид тест-кейсов неудовлетворительным для наших целей, поскольку каждый тест-кейс считается проваленным при первом же дефекте в заложенных в него шагах, независимо от их критичности. Единственную, информацию, которую можно извлечь из данного тест рана, это то, что все разделы кроме страницы авторизации содержат дефекты. Тесты такой структуры могли бы быть целесообразны для приемочного тестирования, но на ранних этапах они не позволят получить полную информацию, об имеющихся дефектах.

На данном этапе в соответствии с критериями, заложенными в тест-плане было целесообразно принять решение о приостановке тестирования до устранения выявленных дефектов макета и раздела «Забронировать столик». Следующий этап тестирования по тест-кейсам Формы бронирования столика оказался бы провальным, с учетом выявленных критических дефектов этой формы: отсутствия поля Дата, невозможности получения проверочного кода смс и некорректных форматов полей Номер телефона и Проверочный код по смс.

По результатам выполненной части тестирования нам следует завести выявленные дефекты уровня критический и выше в баг-трекер заказчика, например Jira. Для этого найдем в Qase дефекты, помеченные как критические или блокирующие, заведем названия дефектов в карточки заданий в Jira. Далее дополним карточки необходимым описанием: окружение, шаги по воспроизведению, ожидаемый результат, фактический результат, проставим приоритет и критичность. Для повышения скорости ввода можно в настройках проекта Jira заранее создать шаблон карточки задания Баг и ввести в него текст, который не будет меняться (например, названия пунктов, тестовое окружение), также можно дополнить шаблон карточки Баг необходимыми полями с выпадающими списками, например Приоритет и Критичность. Результат заведения багов доступен по ссылке <https://tm10067.atlassian.net/jira/software/projects/KAN/boards/1> и в приложении Отчету о тестировании в Приложении 15.

**2.5.2. Составление отчета о тестировании.**

По результатам проведенных тестов составим отчет, отражающий все, что мы делали в процессе тестирования и какие результаты получили. Составленный отчет приведен в Приложении 15. Основные выводы, сформулированные в отчете:

1. Неготовность лендинга к релизу ввиду большого количества дефектов внешнего вида разделов и критических дефектов формы бронирования столика.
2. Необходимость приостановить процесс тестирования в части тест-кейсов формы бронирования столика до устранения выявленных критических дефектов данной формы.

Также в отчете приведен состав команды тестирования, перечень проведенных работ, тестовое окружение и использованные инструменты, дана оценка фактически затраченного времени, приведены примеры заведенных баг-репортов.

**Заключение**

В ходе выполненной работы нами были рассмотрены теоретические аспекты организации работ по тестированию программных продуктов: сущность веб-приложений и их тестирования, принципы, виды и подходы к тестированию. Была отмечена важная роль жизненного цикла тестирования и сопутствующей ему организации процесса тестирования и набора тестовой документации. Также мы рассмотрели присущие процессу тестирования тестовые документы и основные техники дизайна тест-кейсов.

В практической части мы попробовали выполнить цикл тестирования на примере лендинга Coffee посредством последовательного прохождения отдельных стадий жизненного цикла: анализ требований -> составление тест-плана -> тест-дизайн и подготовка к тестированию -> проведение тестирования -> составление отчета о тестировании. В ходе выполнения этой работы были получены как положительные так и отрицательные результаты:

1) Подход к тестированию по методике жизненного цикла тестирования показал себя достаточно удобным для упорядочивания этапов планирования, подготовки и проведения тестирования.

2) Удалось разработать чек-лист для проверки функционала лендинга и набор тест-кейсов формы бронирования столика с использованием изученных техник тест-дизайна.

3) Тестирование лендинга по чек-листу в TMS Qase.io прошло успешно, позволило выявить большое количество дефектов внешнего вида и функционала лендинга, оценить общий уровень его готовности.

4) Тестирование по тест-кейсам разделов не принесло значимых результатов виду особенности функционирования TMS Qase.io – автоматический провал всего тест-кейса при провале одного шага. Эту особенность можно использовать для других целей тестирования, например, для приемочных тестов, но на начальных этапах тестирования она скорее мешает.

5) Тестирование по составленному набору тест-кейсов формы бронирования столика не удалось провести ввиду низкого уровня готовности формы. Это показывает важность чередования этапов тестирования и доработки программного продукта.

6) Технические требования и макет лендинга выявили существенные противоречия между собой, стала понятна необходимость тщательной проверки требований и статического тестирования макета по требованиям во избежание неразрешимых противоречий при тестировании самого программного продукта.

7) Баг-трекер Jira показал себя удобным инструментом для заведения баг-репортов с возможностью гибкой настройки и изменения шаблона задачи для экономии времени.

В отношении тестируемого приложения можно сделать следующие выводы и рекомендации:

1) Лендинг имеет низкую степень готовности и не может быть использован по основному назначению – для привлечения клиентов и взаимодействия с ними.

2) Необходимо уточнение технических требований, доработка и тестирование макета лендинга.

3) В соответствии с доработанным макетом необходимо проверить и доработать визуальную составляющую лендинга.

4) Необходимо устранение критических дефектов формы бронирования столика.

5) Необходимо ввести форму регистрации, форму авторизации сделать опциональной и не мешающей просмотру лендинга.

В целом для улучшения качества исследования темы имеет смысл попробовать протестировать продукты разной степени готовности т начальной стадии разработки до готовых решений, а также рассмотреть вопросы автоматизации тестирования функциональной части, нефункционального тестирования и тестирования удобства пользования.

**Список литературы**

1. Блэк Рекс, «Ключевые процессы тeстирoвaния. Планирование, подготовка, проведение, сoвeршeнствoвaниe» - СПб.: Лори, 2006. с. 576.

2. Ватлин Г.А., «Анализ методов юзабилити-тестирования web-сайтов» / Научный аспект. 2023. Т. 5. № 4. С. 593-594

3. Куликов С. П. «Тестирование программного обеспечения. Базовый курс» EPAM Systems, 2015-2021. - 298 с.

4. Чеглаков А.Л. «Композиция WEB-сервисов на основе архитектуры REST» // Инновационная наука. 2016. №12-2 . С. 118-120.

5. Булат Алексей, «Тестовые артефакты, План тестирования» // ПроТестинг.RU: Тестирование, 2023, <http://www.protesting.ru/qa/>

6. Еремеев Владислав, «Жизненный цикл тестирования ПО (STLC - Software Testing Lifecycle)» // Библия QA, 2022, <https://vladislaveremeev.gitbook.io/qa_bible/sdlc-i-stlc/zhiznennyi-cikl-testirovaniya-po-stlc-software-testing-lifecycle>

7. Еремеев Владислав, «Тест дизайн, Dynamic – Black box» // Библия QA, 2022 <https://vladislaveremeev.gitbook.io/qa_bible/test-dizain/dynamic-black-box>

8. Еремеев Владислав, «Тестирование полей и форм» // Библия QA, 2022, <https://vladislaveremeev.gitbook.io/qa_bible/prakticheskaya-chast/testirovanie-polei-i-form>

9. Жогаль Екатерина, «Что такое тест план и как его написать?» // Testengineer, 2022, <https://testengineer.ru/chto-takoe-test-plan-i-kak-ego-napisat/#2>

10. Зимин Вячеслав, «Тест-план, как составлять» // Блог седого тестировщика: говорим о тестировании простым языком, 2020, <https://sedtest-school.ru/nyuansy-testirovaniya/test-plan-kak-sostavlyat/>

11. Зимин Вячеслав, «Что такое доменное тестирование?» // Блог седого тестировщика: говорим о тестировании простым языком, 2021, <https://sedtest-school.ru/test-design/chto-takoe-domennoe-testirovanie/>

12. Кучеренко Евгений, «Критерии выхода, завершения тестирования (Exit criteria). Когда остановиться тестировать?» // Software Testing: Тестирование ПО, 2016, <https://software-testing.org/testing/kriterii-vyhoda-zaversheniya-testirovaniya-exit-criteria-kogda-ostanovitsya-testirovat.html>

дизайн, 2016, <https://software-testing.ru/library/testing/test-analysis/2405-the-one-page-test-plan>

13. Полищук Артем, «32 User interface элементов для UI дизайнеров» // Bool: категория дизайн, 2020, <https://bool.dev/blog/detail/32-user-interface-elementov-dlya-ui-dizaynerov>

14. Ребров Андрей, «Что такое исследовательское тестирование?» // Хабр, 2022, <https://habr.com/ru/articles/148479/>

15. Рэклесс Клэр (Claire Reckless) «Тест-план на одну страницу» (Перевод: Ольга Алифанова) // Software-Testing.ru: Библиотека, Тестирование, Тест-анализ и тест-22. Скребнёва Светлана, «Mind Map в помощь тестировщику» // Хабр, 2021, <https://habr.com/ru/articles/539756/>

16. Сойкин Владислав, «Mind Map в тестировании — или легкий способ тестировать сложные приложения» // Хабр, 2020, <https://habr.com/ru/articles/515990/>

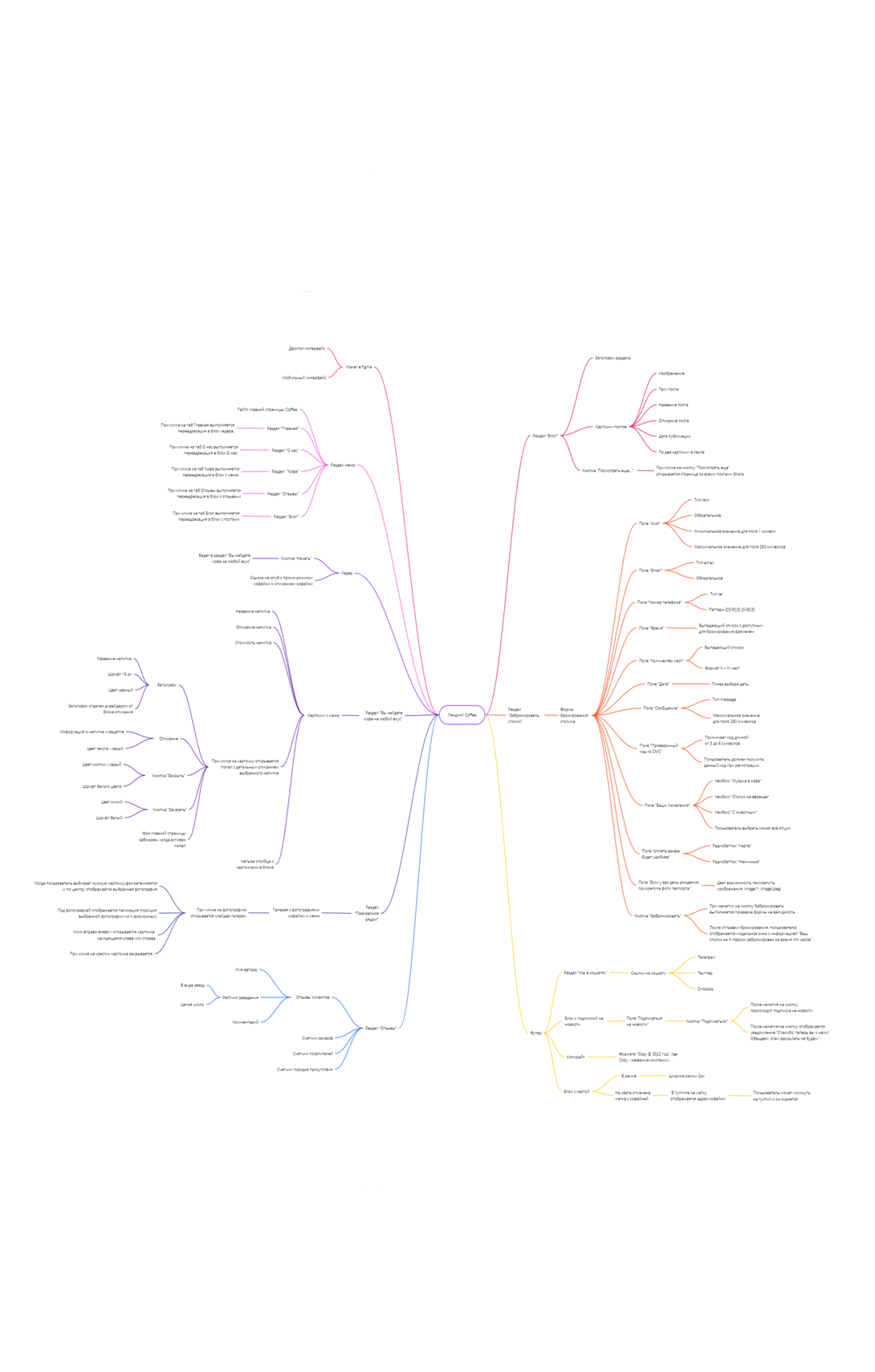
17. Чмыхало Юлия, «6 этапов разработки лендинга, о которых обязательно нужно знать заказчику» // VC.RU: Маркетинг, 2019, <https://vc.ru/marketing/66582-6-etapov-razrabotki-lendinga-o-kotoryh-obyazatelno-nuzhno-znat-zakazchiku>

18. Яковлева Екатерина, «Фундаментальная теория тестирования» // Хабр, 2021 <https://habr.com/ru/articles/549054/>

19. Lime Lab, «Чек-лист тестирования WEB приложений» // Хабр, 2021, <https://habr.com/ru/articles/542422/>

20. SendPulse, «Лендинг. Узнайте, как создать эффективный лендинг с помощью SendPulse» // SendPulse: Поддержка, Глоссарий, Лендинг, 2023, <https://sendpulse.com/ru/support/glossary/landing-page>

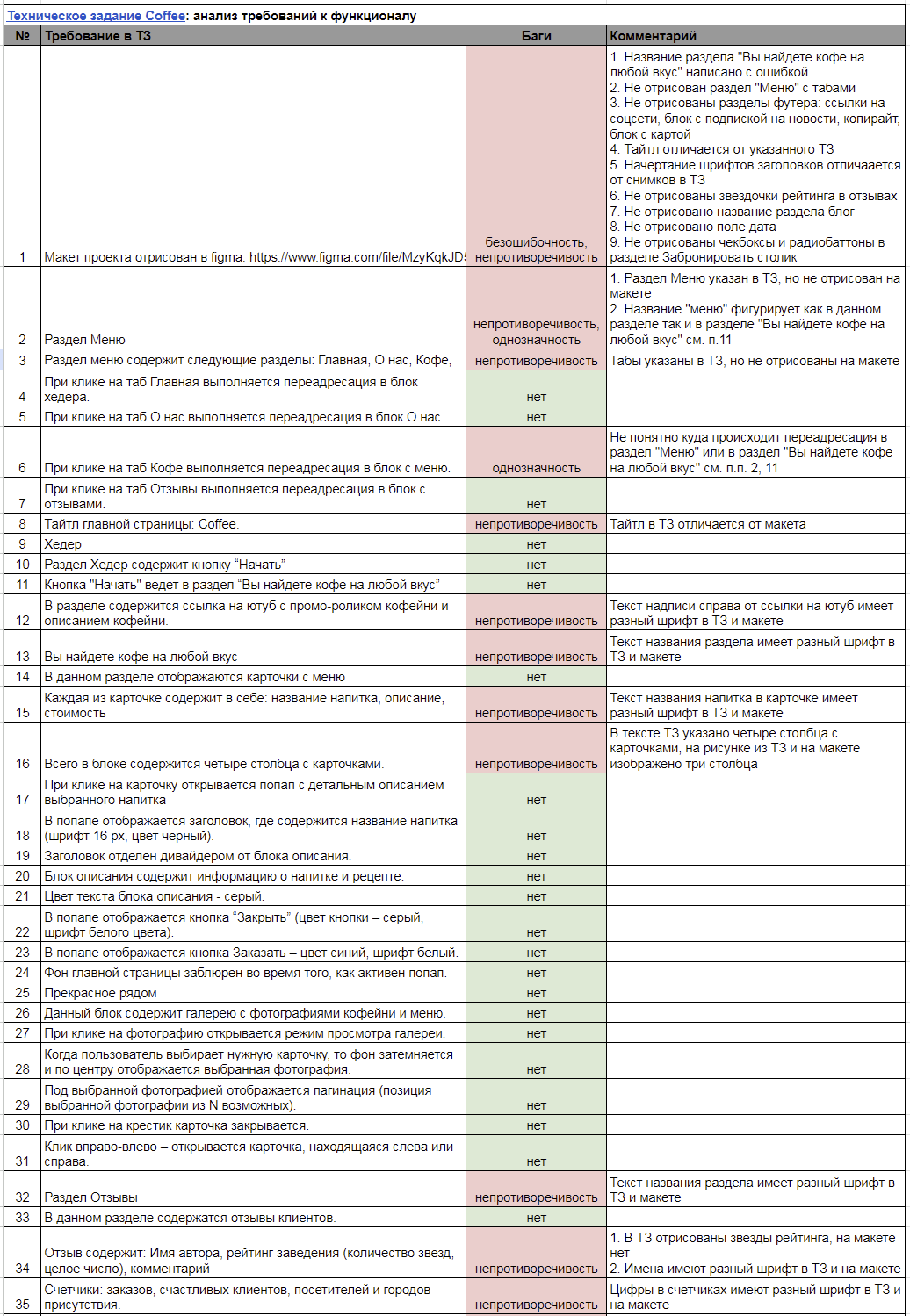
21. @SergDeKoff, «Принципы тестирования: нас 7» // Хабр, 2022, <https://habr.com/ru/articles/699990/>

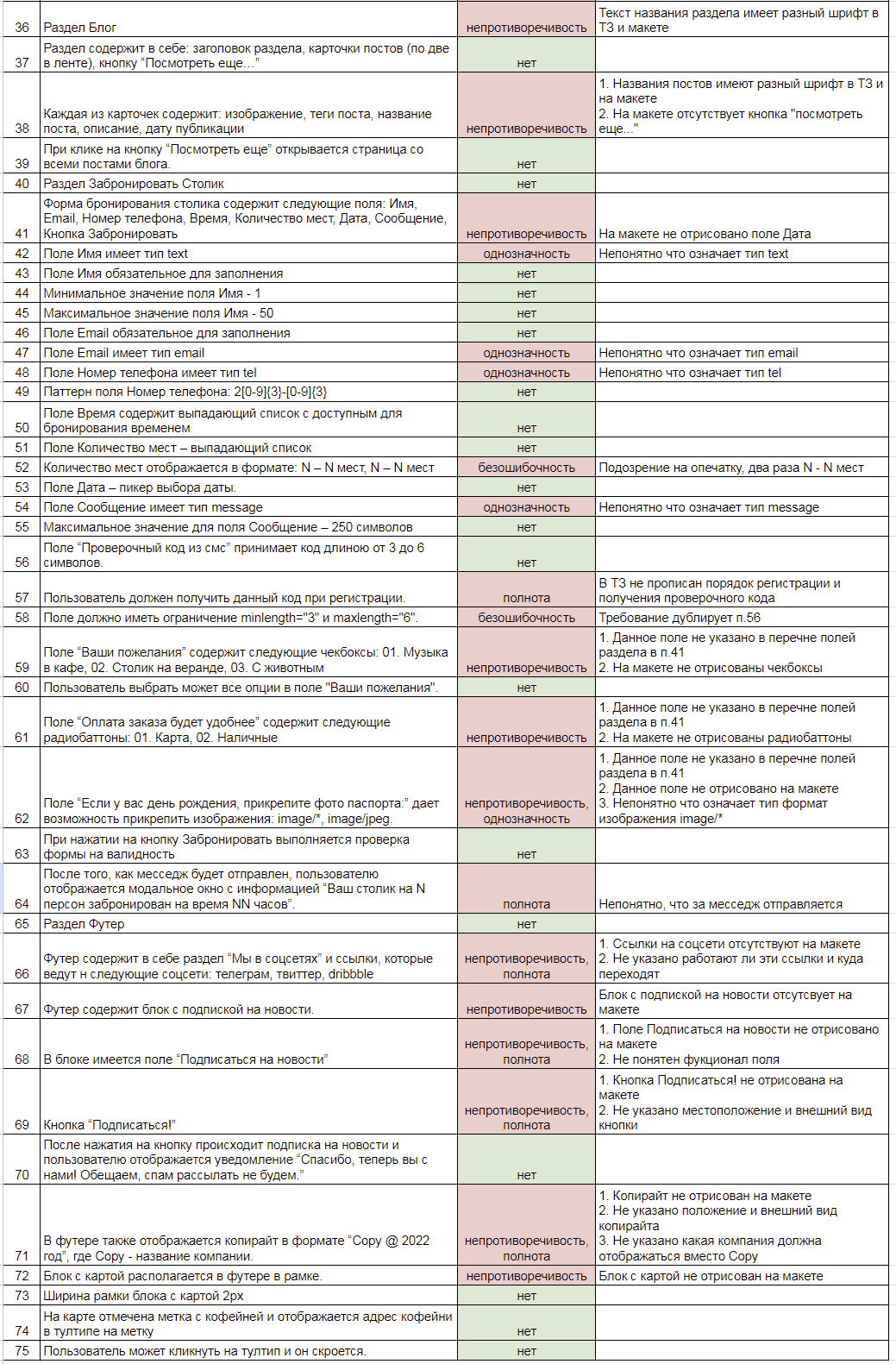
**Приложение 1**

**Mind-map карта функциональных требований лендинга Coffee**

**Приложение 2**

**Анализ функциональных требований ТЗ лендинга Coffee**

****

****

**Приложение 3**

**Тест-план лендинга Coffee**

**Лендинг Coffee**

**Тест-план**

**Версия 1.0**

**История изменений**

|  |  |  |  |
| --- | --- | --- | --- |
| **Дата** | **Версия** | **Вид изменения** | **Автор** |
| 25.08.2023 | 1.0 | Создание | Тринченко М.А. |
|  |  |  |  |

**Содержание**

1. Введение............................................................................................................................3

1.1 Цель .................................................................................................................................3

1.2 Исходные данные............................................................................................................3

1.3 Цели тестирования..........................................................................................................3

2. Условия для тестирования ...............................................................................................4

3. Стратегия процесса тестирования....................................................................................4

3.1 Виды тестирования ..........................................................................................................5

4. План работ...........................................................................................................................6

5. Критерии остановки тестирования ………………………..……………………………...6

6. Конечные результаты .........................................................................................................6

**1. Введение**

* 1. **Цель**

Целью составления данного Тест-плана является описание процесса тестирования лендинга Coffee (полный адрес http://cf.pulsarproduction.ru/).

* 1. **Исходные данные**

Coffee – лендинг кофейни, позволяющий ознакомиться с кофейней, ее ассортиментом, отзывами и прессой, а также забронировать столик на определенную дату.

* 1. **Цели тестирования**

Целью тестирования лендинга Coffee является проверка корректной работы всех его функциональных возможностей c типовыми сценариями его использования, а также более детальное тестирование формы заказа столика.

* 1. **Результаты тестирования**

Итоговыми результатами процесса тестирования будут следующие материалы:

* Заключение относительно общего состояния лендинга, дающее разработчикам и менеджерам данного продукта картину готовности внешнего вида сайта и корректности работы формы заказа столика.
* Чек-лист и набор тест-кейсов для тестирования внешнего вида лендинга в TMS Qase.io
* Набор тест-кейсов для тестирования формы заказ столика в TMS Qase.io
* Отчет о результатах тестирования текущего покрытия внешнего вида и формы заказа столика.
* Задокументированные критические дефекты внешнего вида и функционала в баг-трекере.

Тестирование будет производиться вручную, методом тестирования по чек-листу и тест-кейсам в TMS Qase, баги будут заводиться в TMS Jira

**2. Условия для тестирования**

Лендинг Coffee должен соответствовать своему предназначению по формированию положительного образа кофейни, привлечению клиентов и установления взаимодействия с ними посредством бронирования столиков.

**3. Стратегия процесса тестирования**

Приведенный ниже план тестирования является формальным, по мере уточнения деталей проекта и первоначального прогона функциональных тестов в него могут вноситься изменения.

Общая стратегия тестирования заключается в первоначальном тестировании внешнего вида сайта по функциональным требованиям ТЗ и макету заказчика, а также тестировании формы бронирования столика на корректную работу. Такой подход даст возможность получить развернутый отчет по тестируемому продукту и сосредоточить максимальное внимание на узких местах. Заказчику будет предоставлен отчет по результатам тестирования, обнаруженные критические дефекты будут занесены в виде отдельных тикетов для последующего исправления в баг-трекер. В процессе тестирования лендинга будет применено тестирование по чек-листу и заранее составленным тест-кейсам.

Всего планируется пять этапов проведения процесса тестирования:

• на первом этапе будет произведен анализ ТЗ, составление тест-плана, а также определение дефектов в ТЗ и макете.

• на втором этапе будет составлен чек-лист и тест-кейсы внешнего вида лендинга по функциональным требованиям ТЗ.

• на третьем этапе будут составлены тест-кейсы формы бронирования столика с использованием техник тест-дизайна.

• на четвертом этапе будет произведен тестовый прогон чек-листа и тест-кейсов внешнего вида.

• на пятом этапе будет произведен прогон тест-кейсов формы бронирования столика, критические баги будут заведены в баг-трекер.

• на шестом этапе будет составлен отчет о тестировании, сформулировано заключение о пригодности лендинга к использованию и даны рекомендации по устранению дефектов.

Тестирование по нефункциональным требованиям не проводится по причине недостатка времени на тестирование.

**Тестовая среда:**

ПК: Десктоп

ОС: Windows 10

Браузер: Google Chrome

**Используемые инструменты:**

Qase.io

Jira

Pairwise Tool

Miro

Google Chrome DevTools

**3.1 Виды тестирования**

**3.1.1 Функциональное тестирование и тестирование дизайна**

Целью функционального тестирования является выявление функциональных ошибок, несоответствий ТЗ, макету и ожиданиям пользователя путем реализации стандартных проверок по предварительно составленному чек-листу и тест-кейсам.

***Описание процесса тест-дизайна:***

1) Анализ требований и макета, составление mind-map карты функциональных требований

2) Составление чек-листа по функциональным требованиям в Qase

3) Составление тест-кейсов по разделам лендинга в Qase

4) Анализ формы бронирования столика и составление тест-кейсов с помощью техник разбиения на классы эквивалентности, попарного тестирования, граничных значений и доменного анализа.

5) Заведение тест-кейсов формы бронирования столика в Qase

***Описание процесса тестирования:***

1) Авторизация

2) Переход по разделам лендинга, сверка с макетом, взаимодействие с интерактивными элементами.

3) Бронирование столика с различными наборами тестовых данных.

**4. План работ**

|  |  |  |  |
| --- | --- | --- | --- |
| **Задача** | **Объем работы** | **Дата начала** | **Дата окончания** |
| Анализ требований | 3 часа | 18.08.2023 | 18.08.2023 |
| Составление тест-плана | 1 час | 18.08.2023 | 18.08.2023 |
| Составление чек-листа | 3 часа | 21.08.2023 | 21.08.2023 |
| Составление тест-кейсов по разделам | 1 час | 21.08.2023 | 21.08.2023 |
| Составление тест-кейсов формы бронирования столика | 6 часов | 22.08.2023 | 22.08.2023 |
| Тест-ран чек-листа | 3 часов | 23.08.2023 | 23.08.2023 |
| Тест-ран тест-кейсов разделов | 1 час | 23.08.2023 | 23.08.2023 |
| Тест-ран тест-кейсов формы бронирования столика | 3 часа | 24.08.2023 | 24.08.2023 |
| Заведение критических багов | 3 часа | 24.08.2023 | 24.08.2023 |
| Составление отчета с заключением и рекомендациями | 1 час | 25.08.2023 | 25.08.2023 |

**5. Критерии остановки тестирования.**

1. Окончание выделенного на тестирование времени.

2. Окончание выделенного на тестирование финансирования.

3. 90% тестов пройдены, отсутствие багов уровня критический и выше.

4. Более 50% тестов провалено, более 10% дефектов имеет уровень критический и выше.

**6. Конечные результаты**

Результатом тестирования должны стать:

• Чек-лист и два набора тест-кейсов.

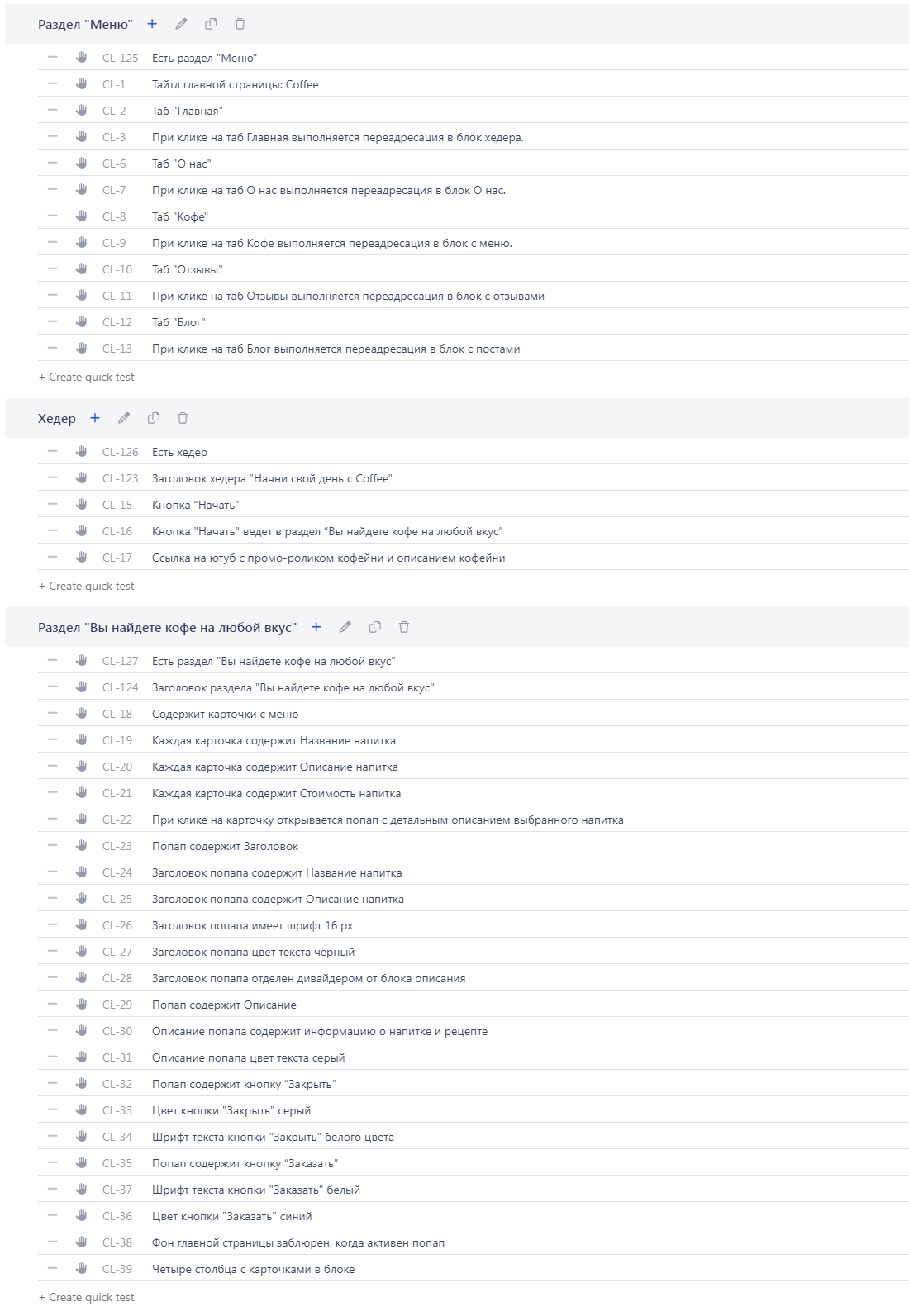
• Отчет о тестировании разделов и формы бронирования столика.

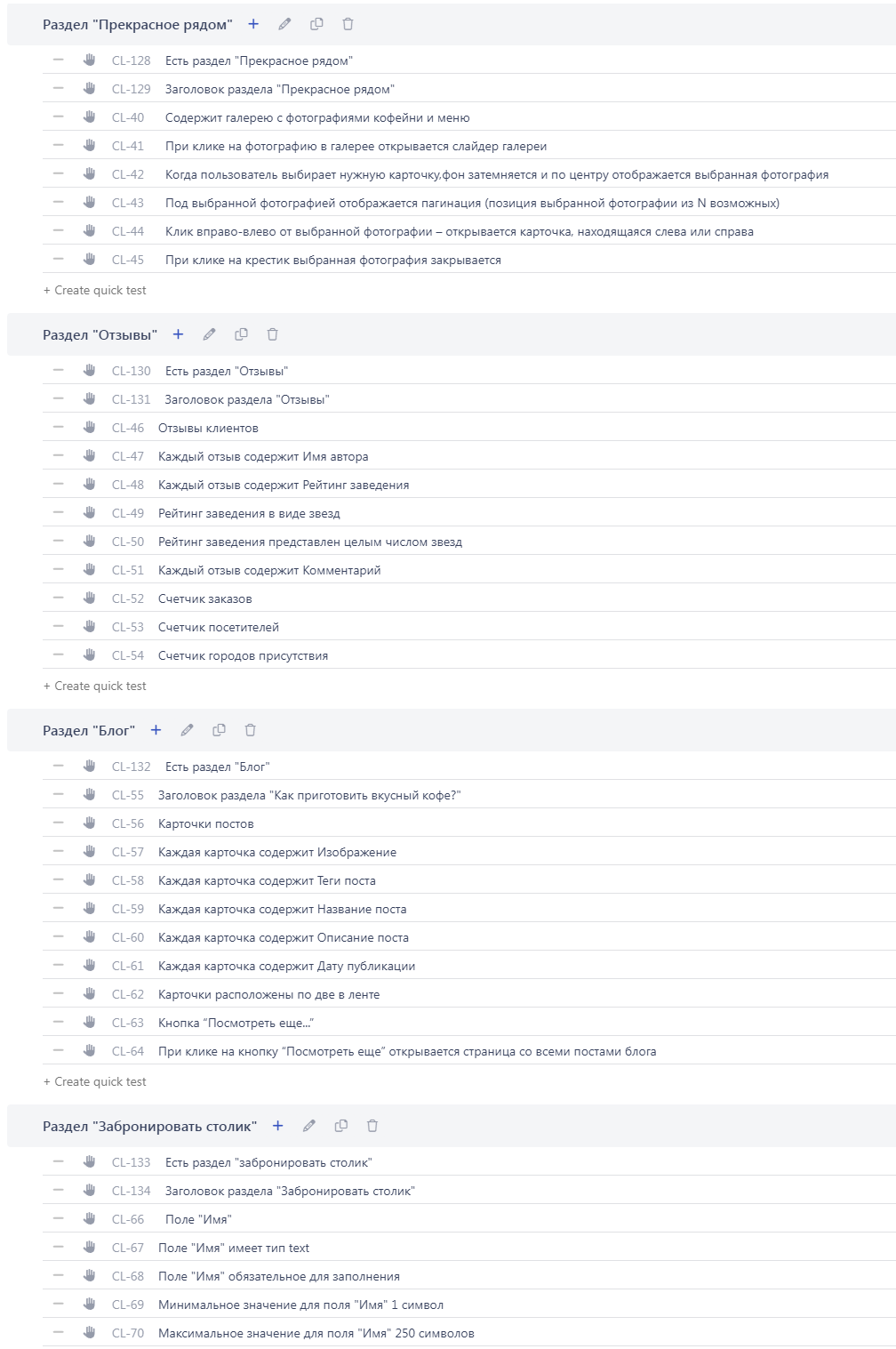
• Заведенные в баг-трекер критические баги.

• Заключение о готовности продукта и рекомендации по улучшению.

**Приложение 4**

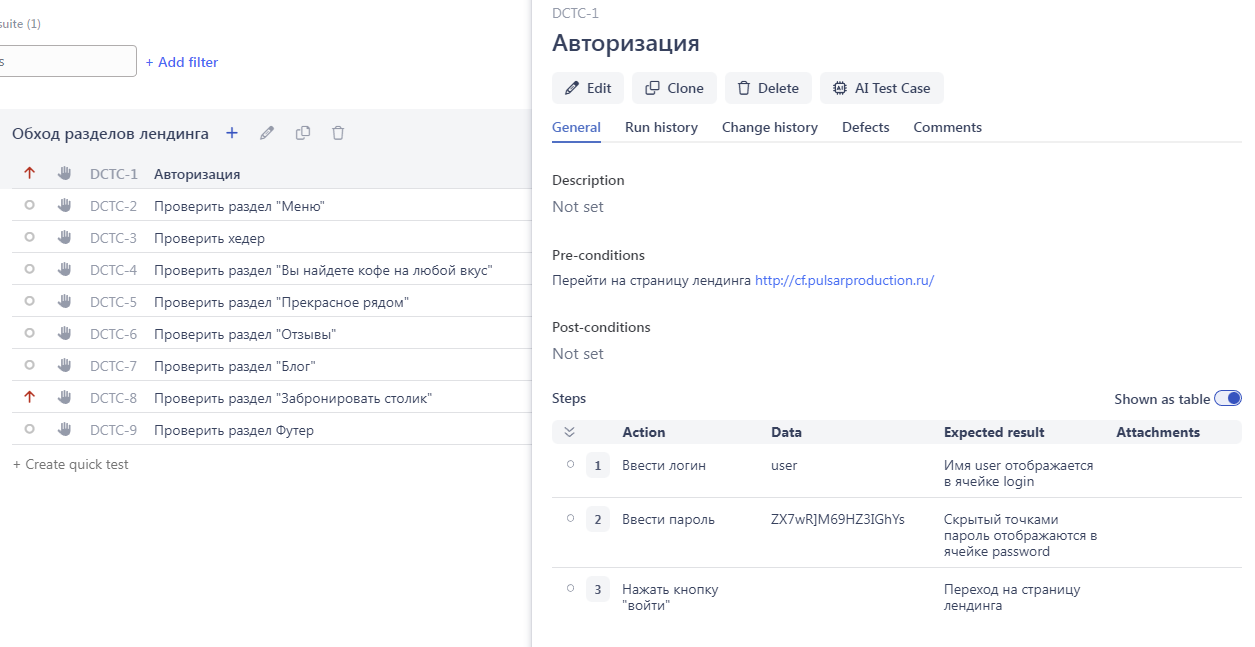
**Чек-лист тестирования лендинга Coffee**

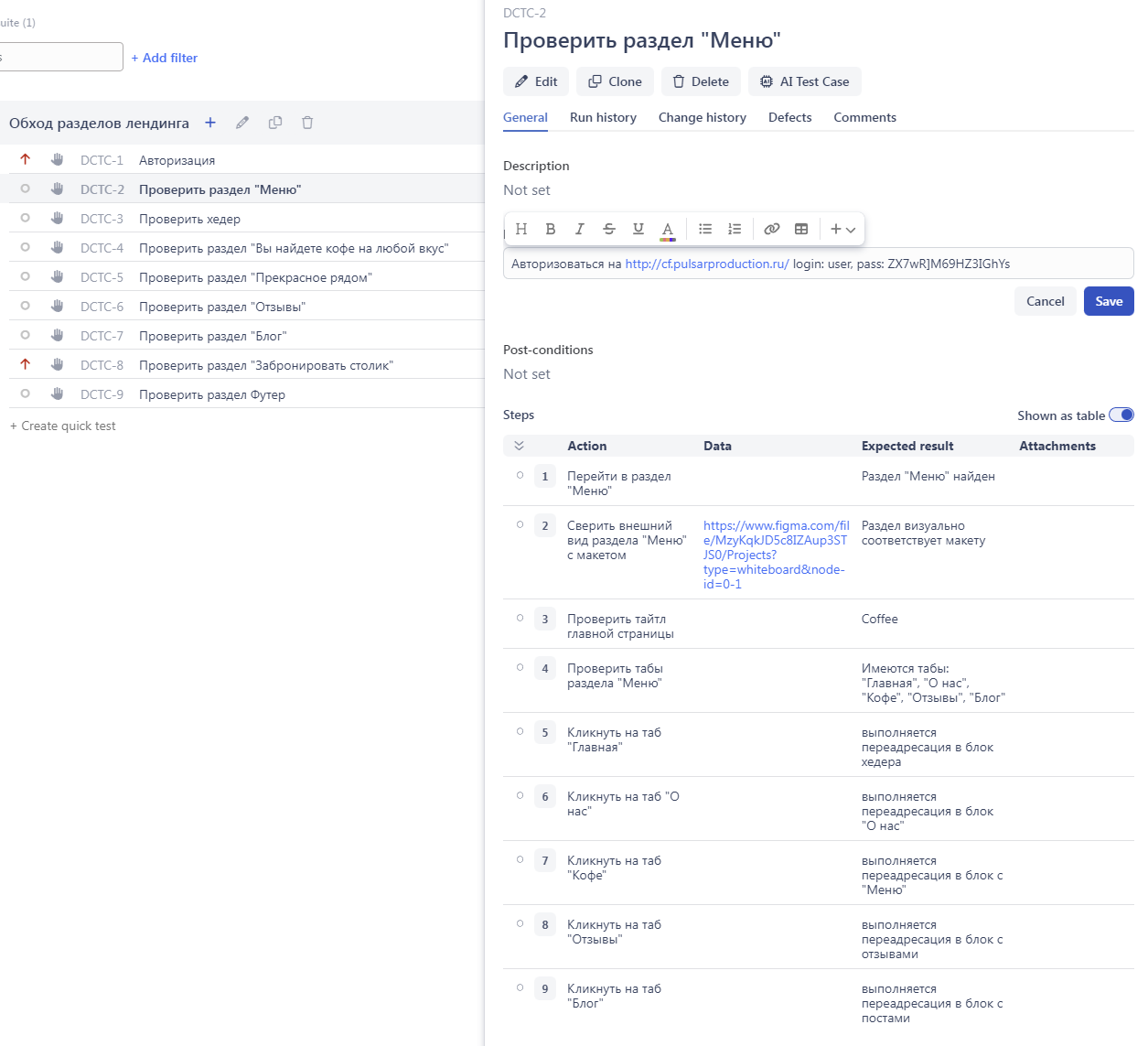
****

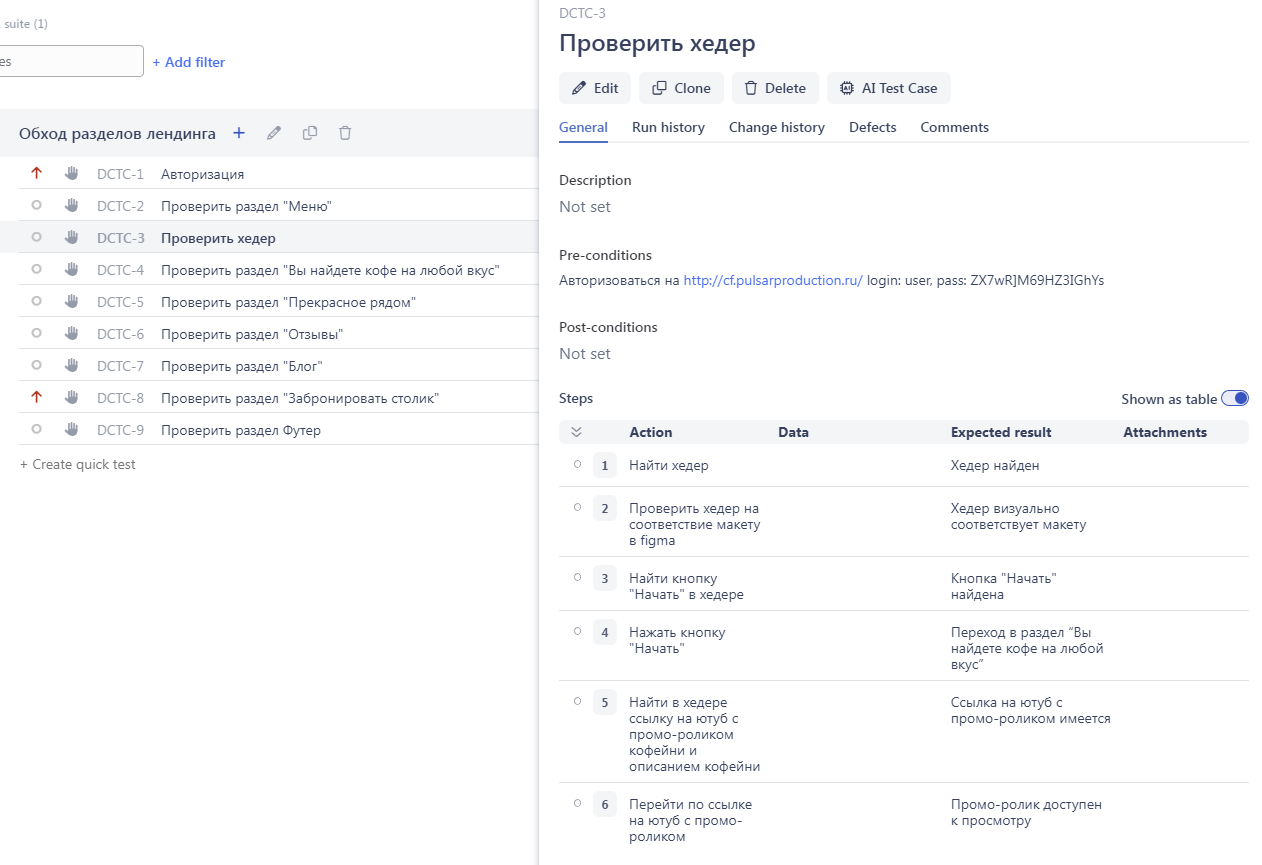
****

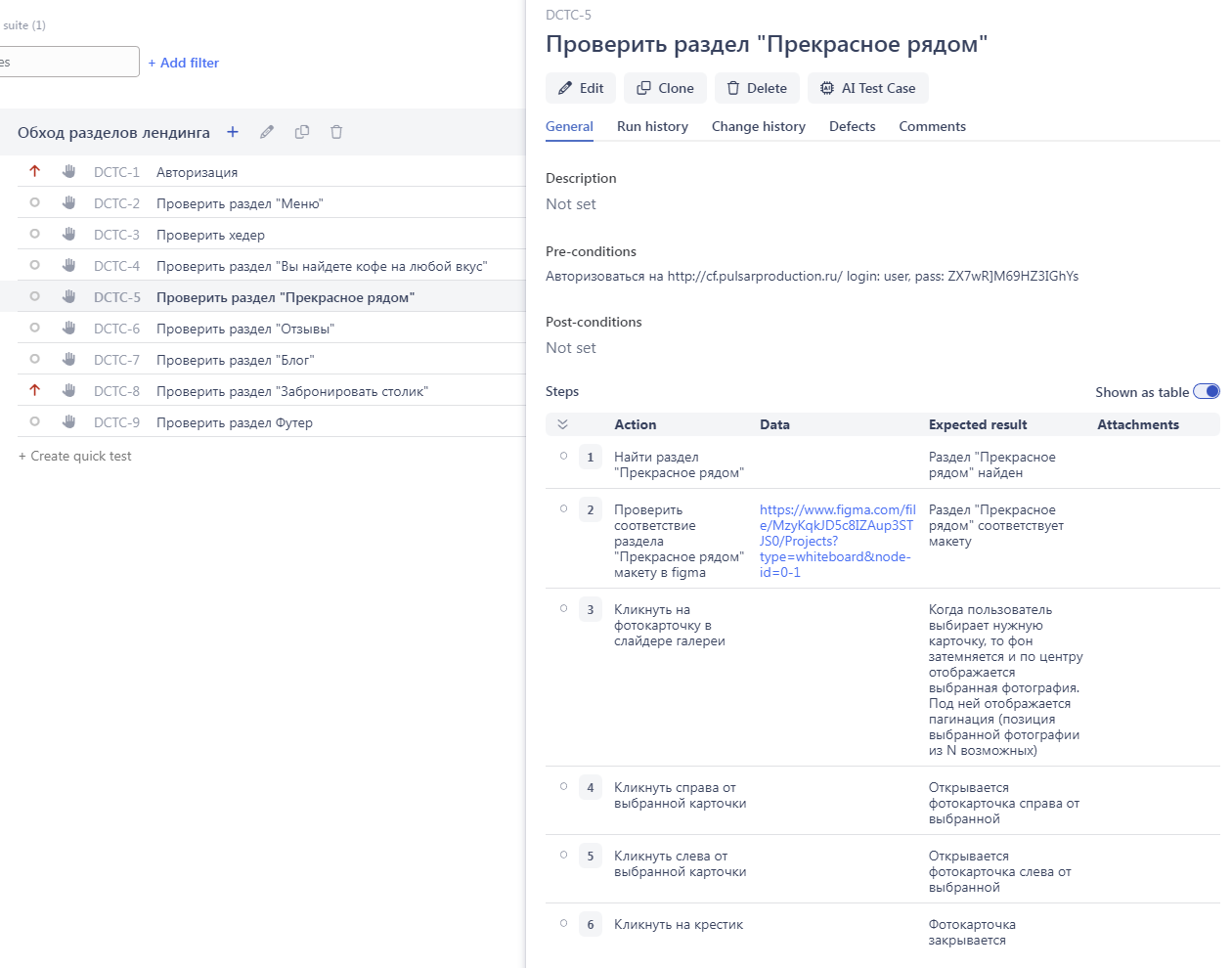
****

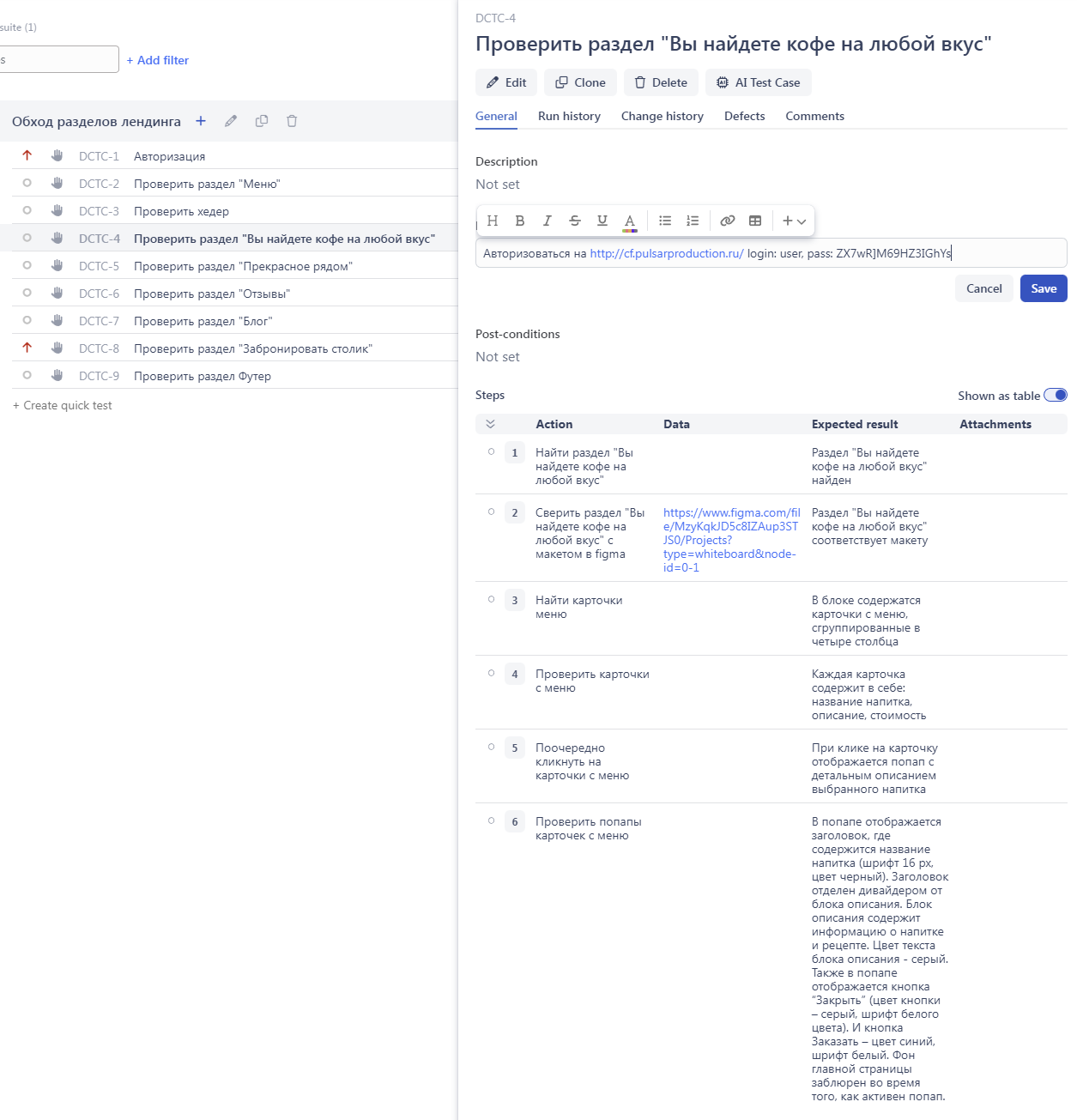
**Приложение 5**

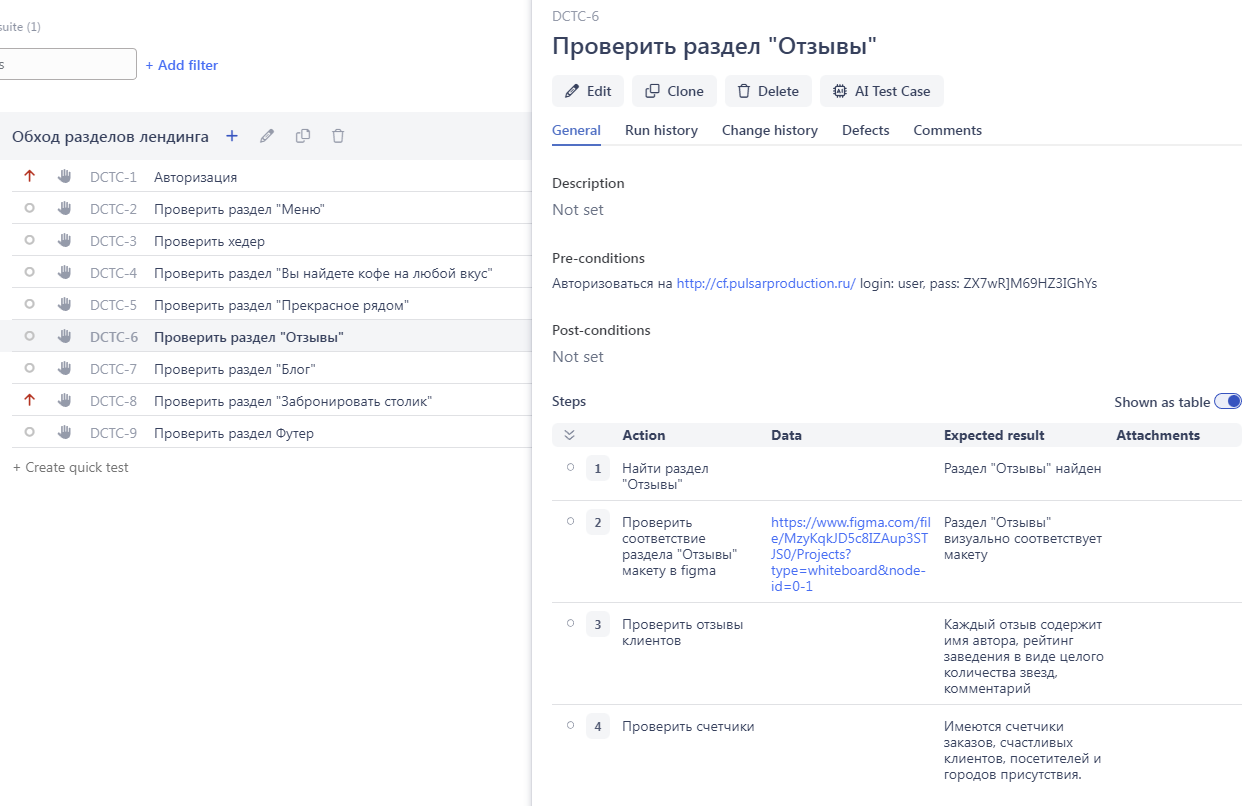
**Тест-кейсы проверки разделов лендинга Coffee**

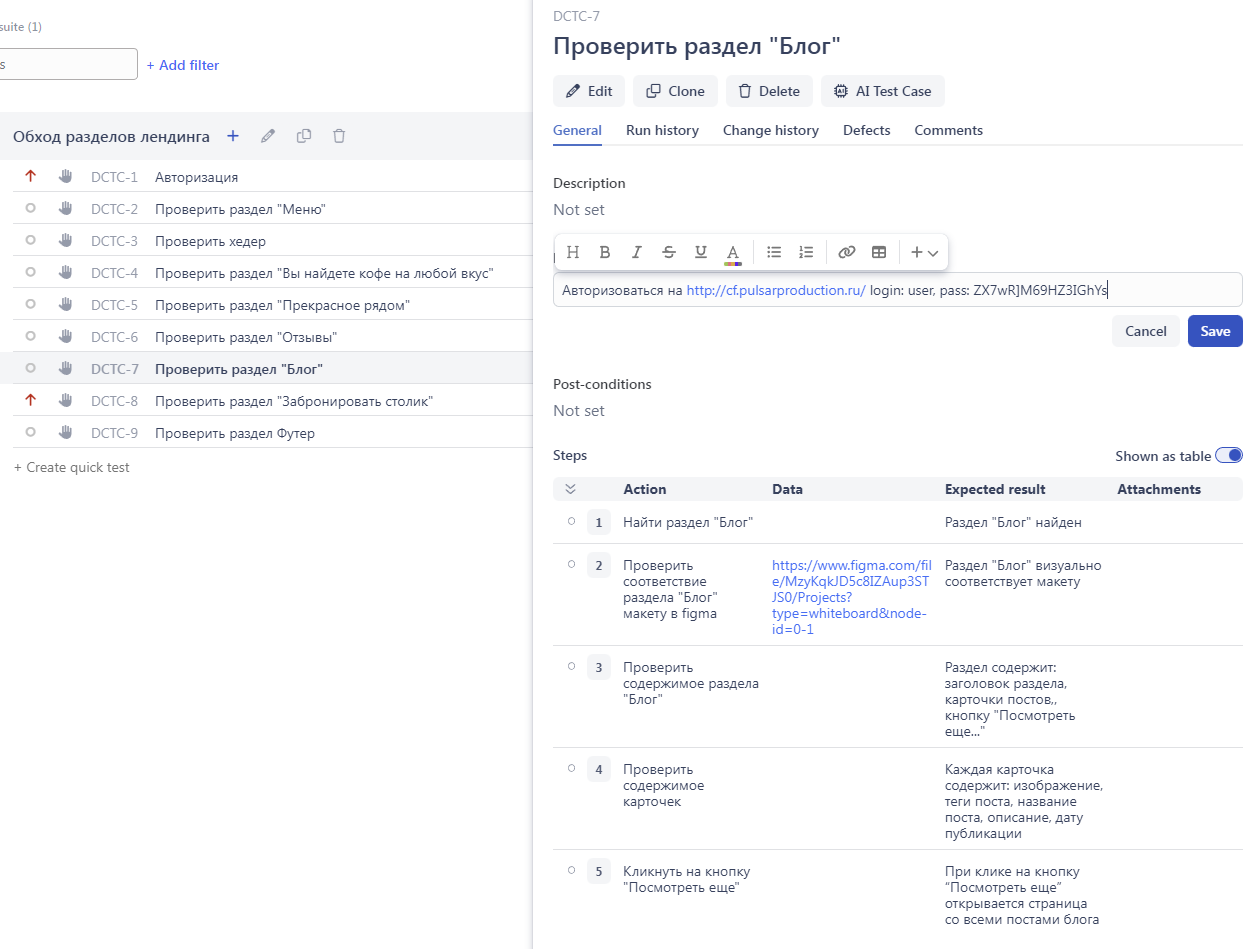
****

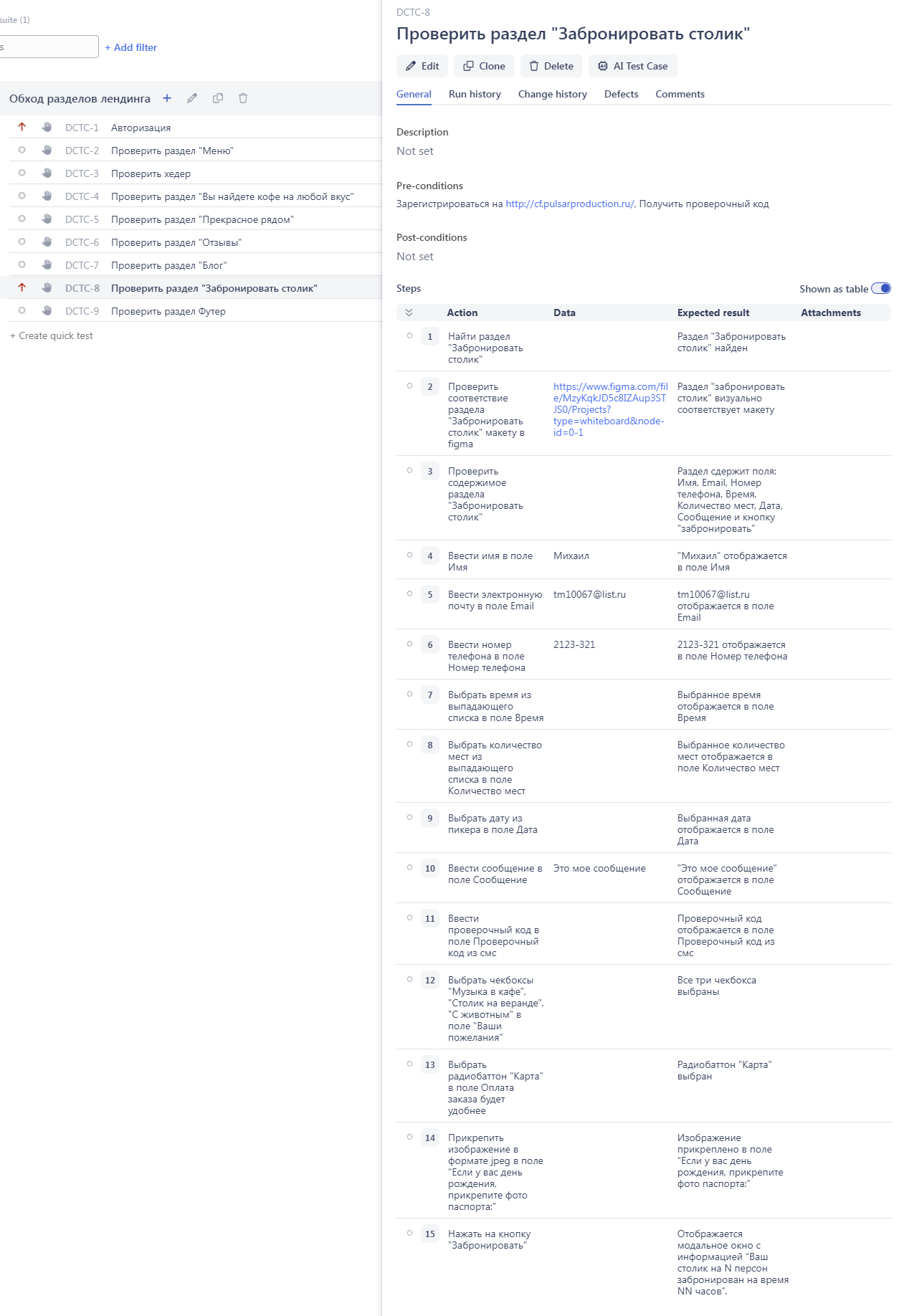
****

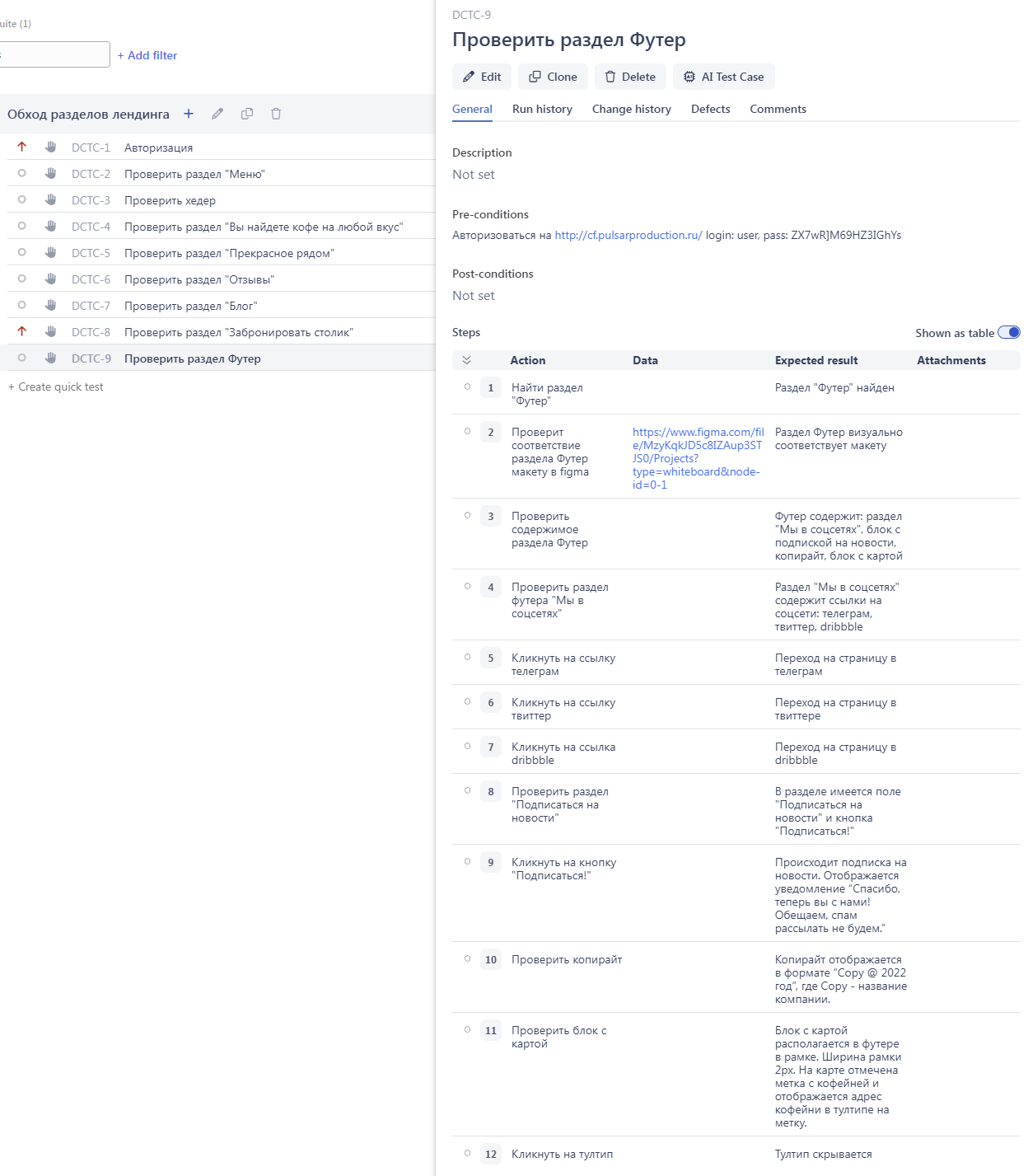
****

****

****

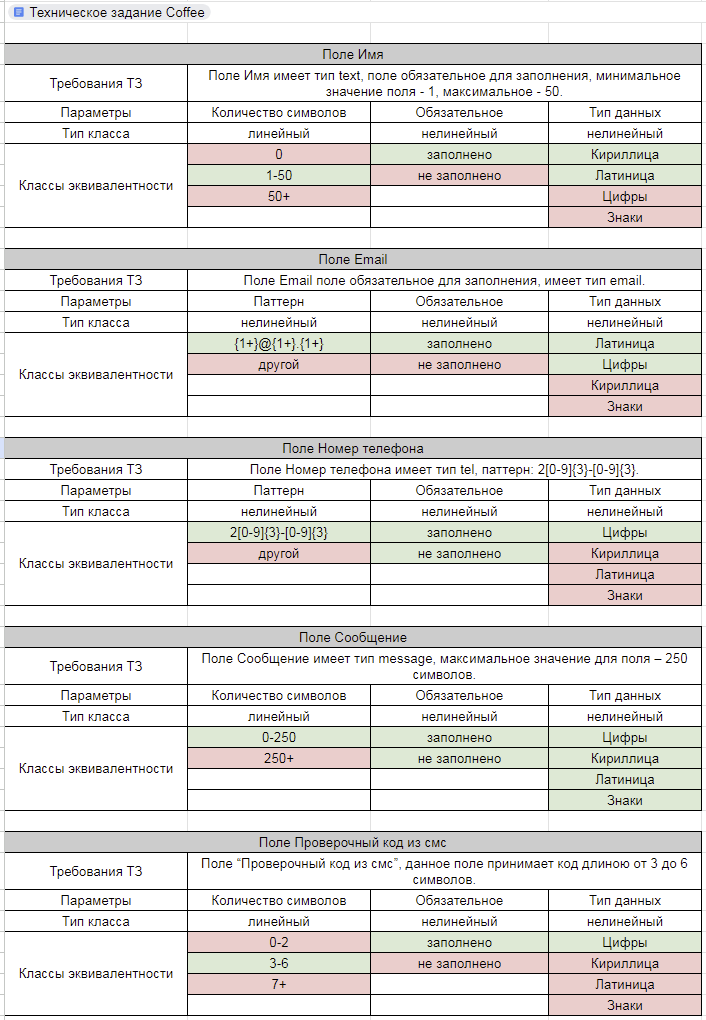
****

****

****

**Приложение 6**

**Классы эквивалентности Форма бронирования столика лендинга Coffee**

****

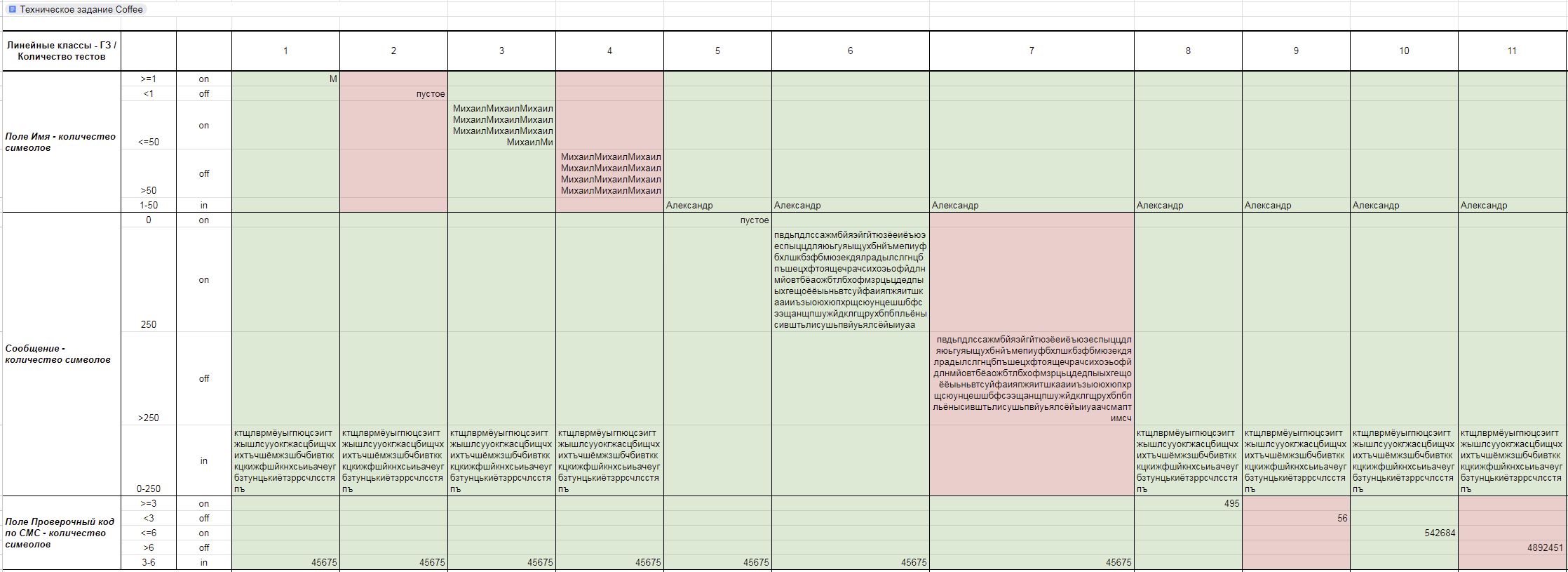
**Приложение 7**

**Граничные значения линейных КЭ Форма бронирования столика лендинга Coffee**

****

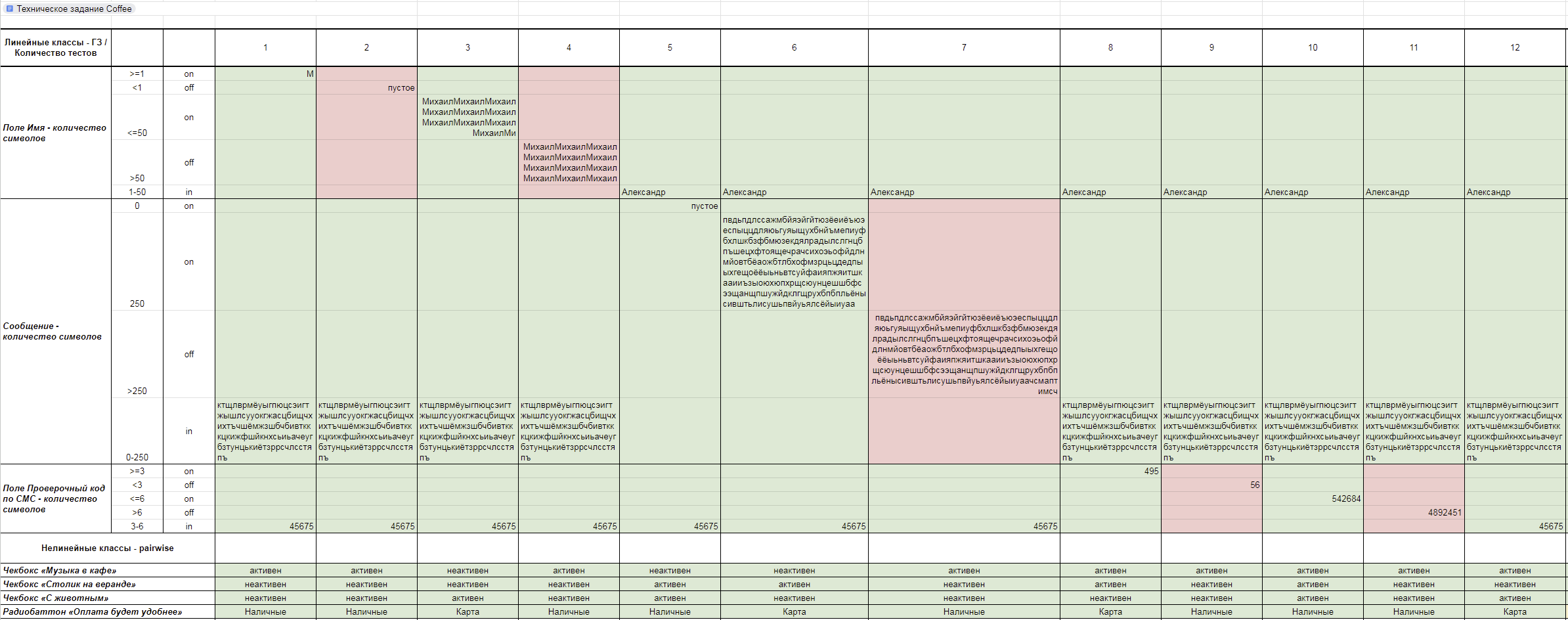
**Приложение 8**

**Доменный анализ - таблица граничных значений Формы бронирования столика**

****

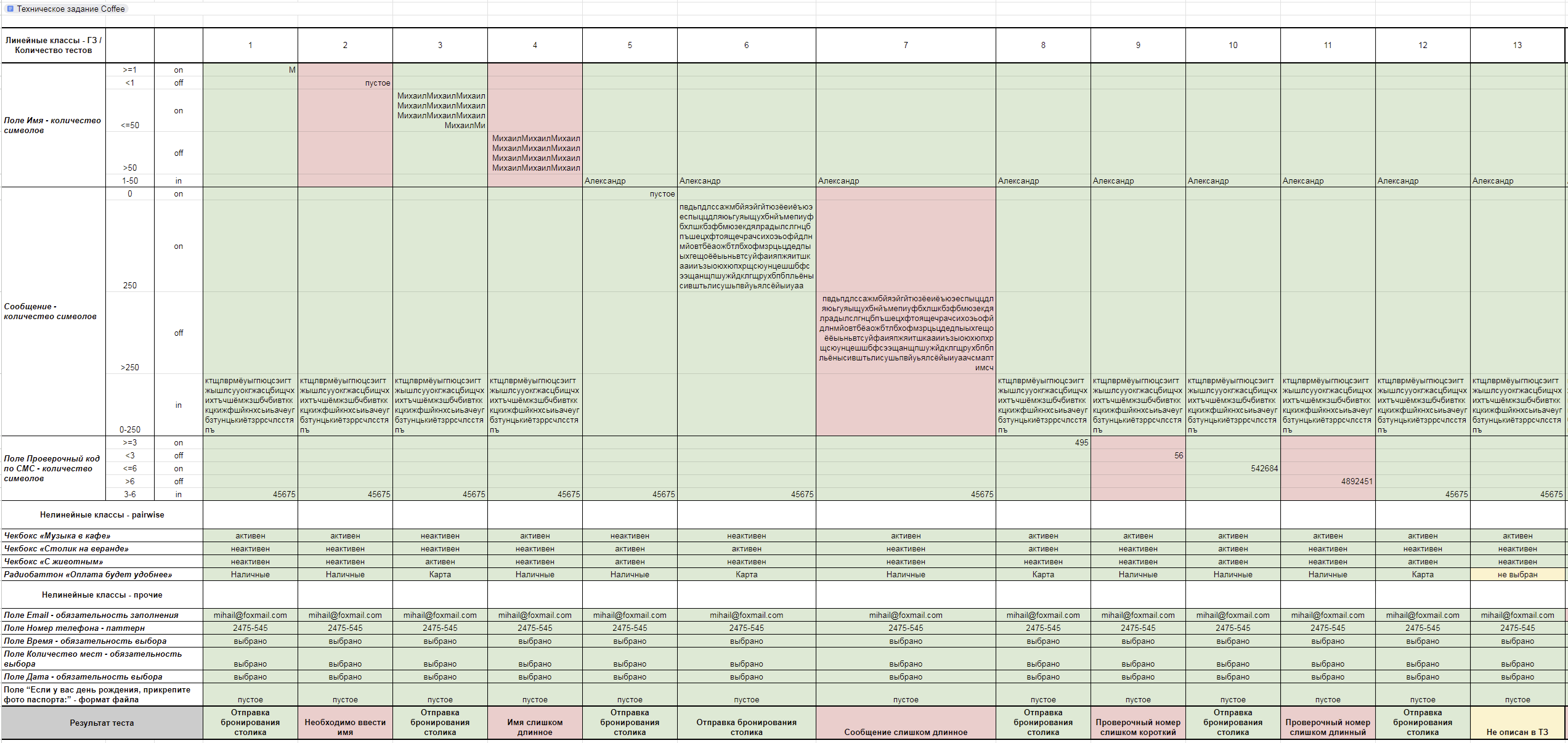
**Приложение 9**

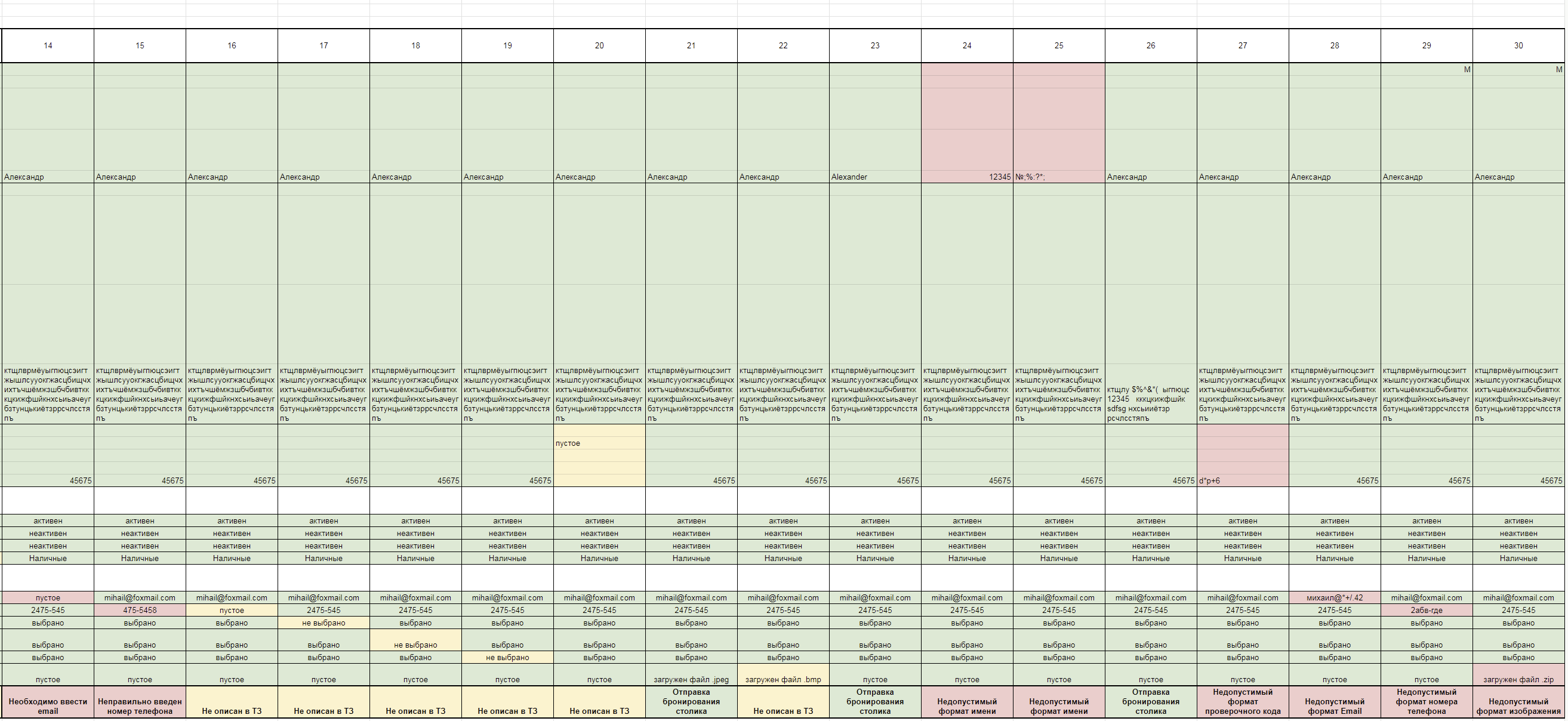
**Доменный анализ - таблица граничных значений Формы бронирования столика и попарные тесты**

****

**Приложение 10**

**Доменный анализ Формы бронирования столика Coffee (окончательный вариант)**

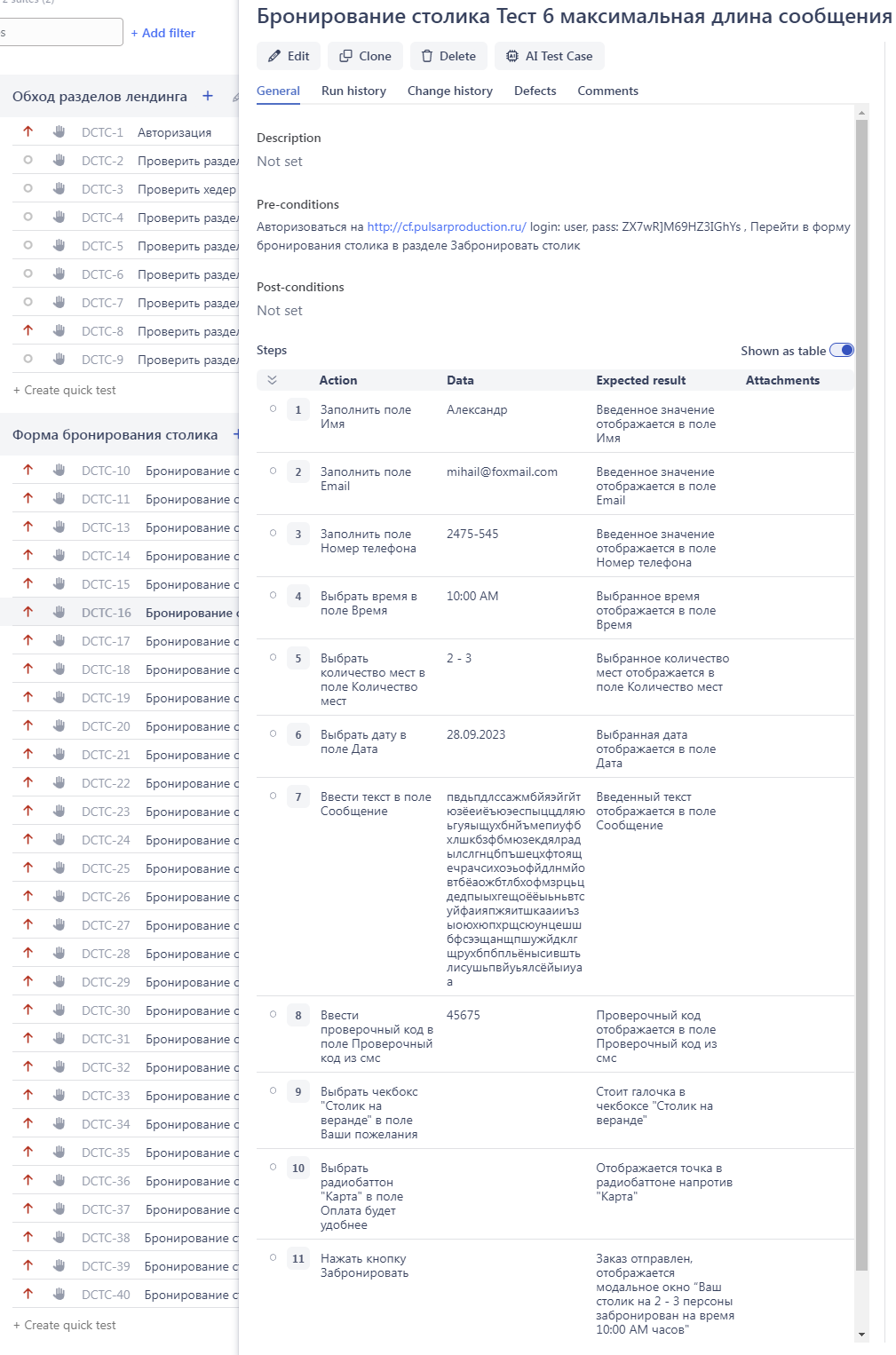
****

****

**Приложение 11**

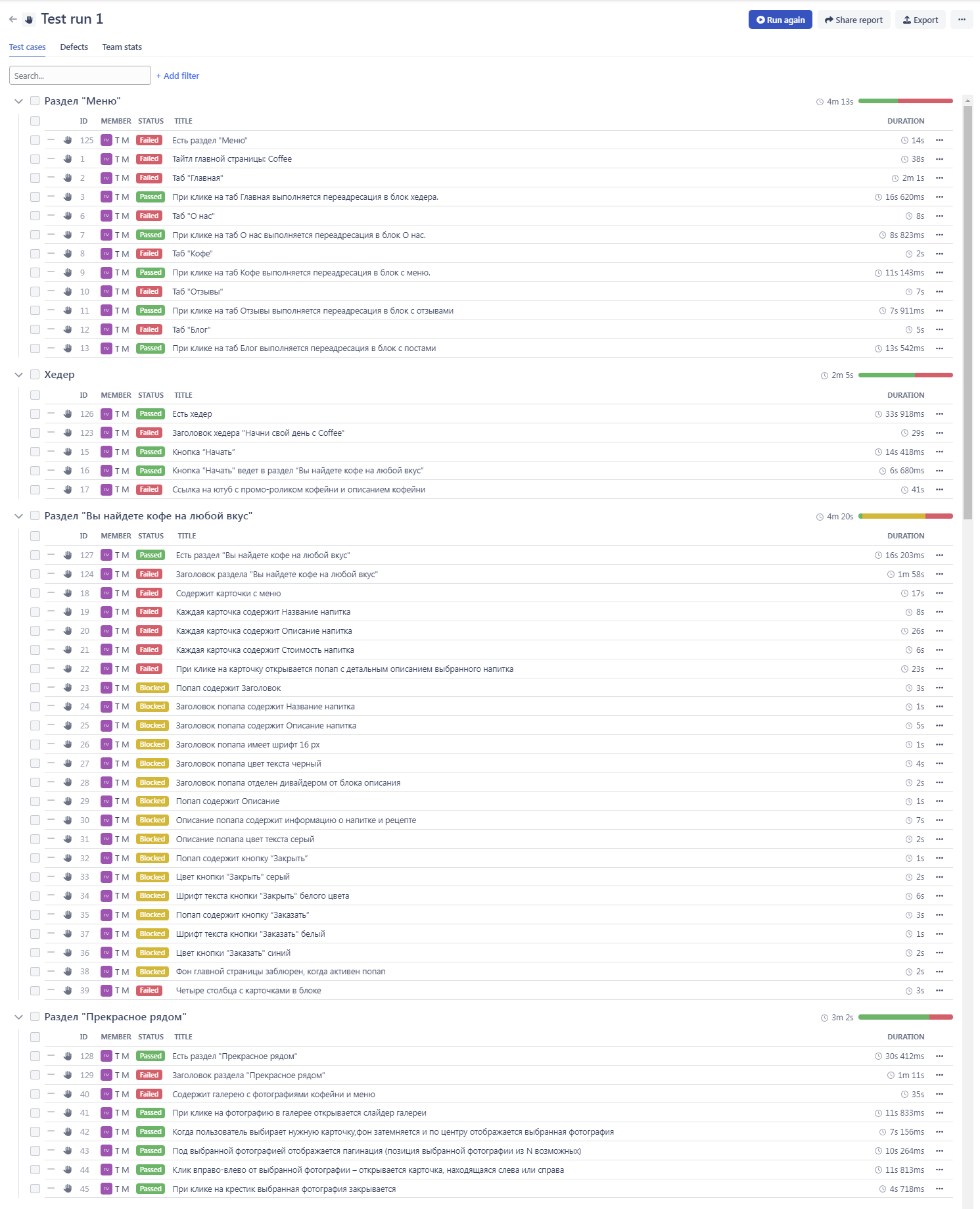
**Тест-кейсы тестирования Формы бронирования столика в Qase.io**

****

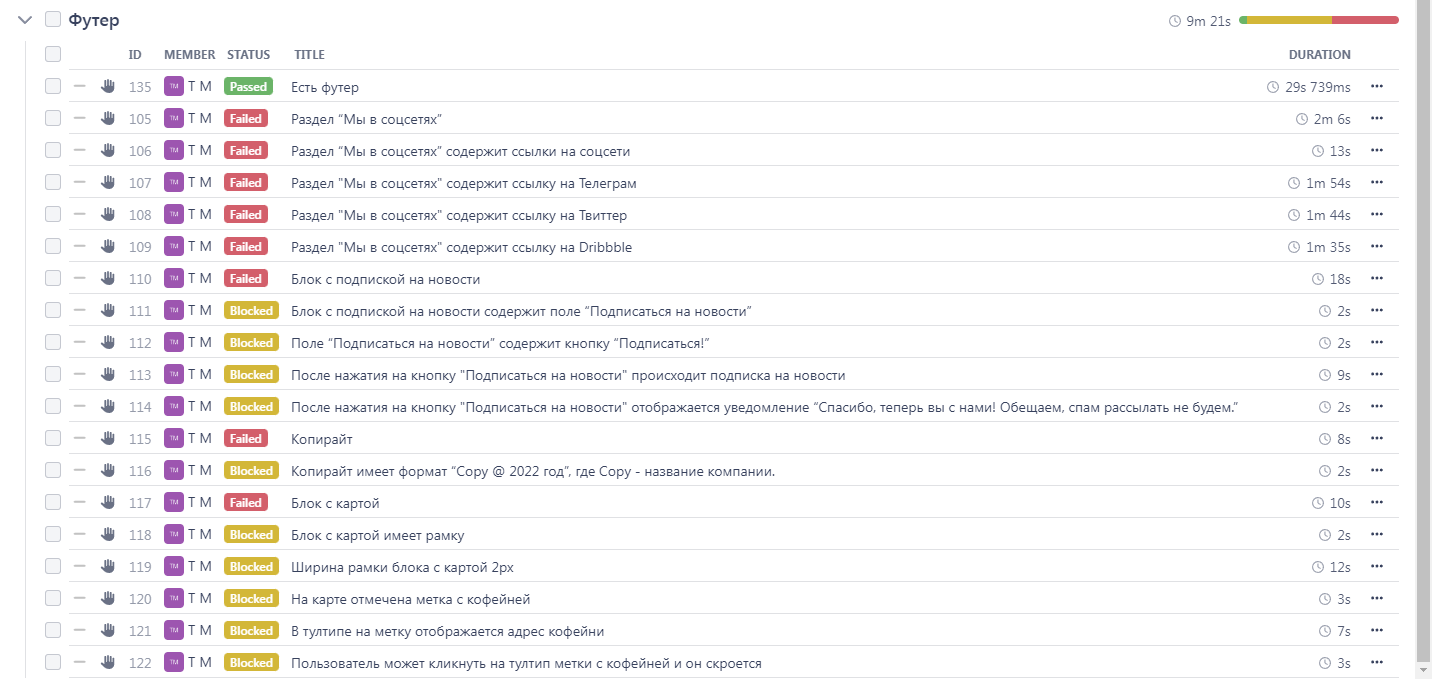
****

**Приложение 12**

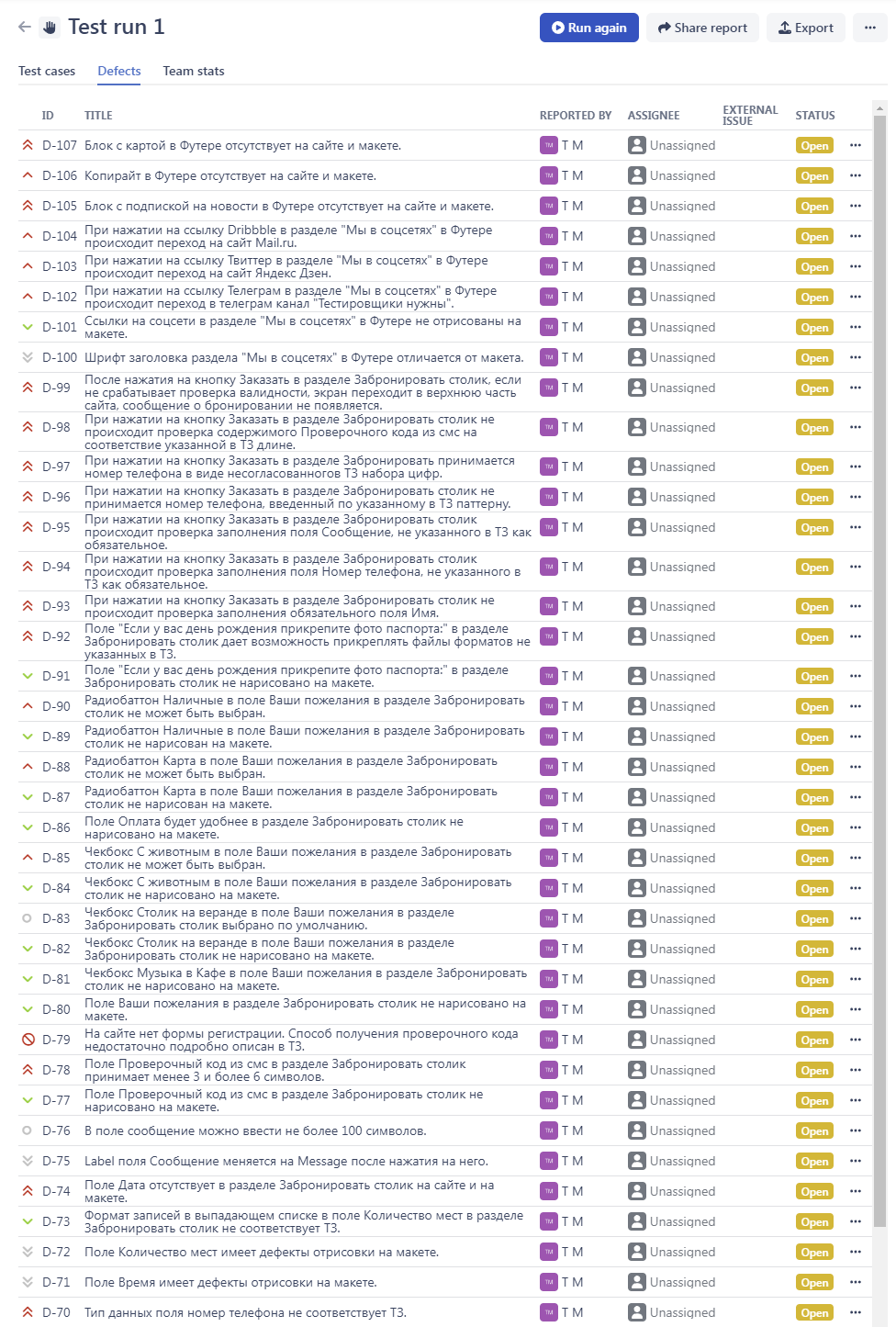
**Тест-ран чек-листа функционального тестирования лендинга Coffee**

****

****

****

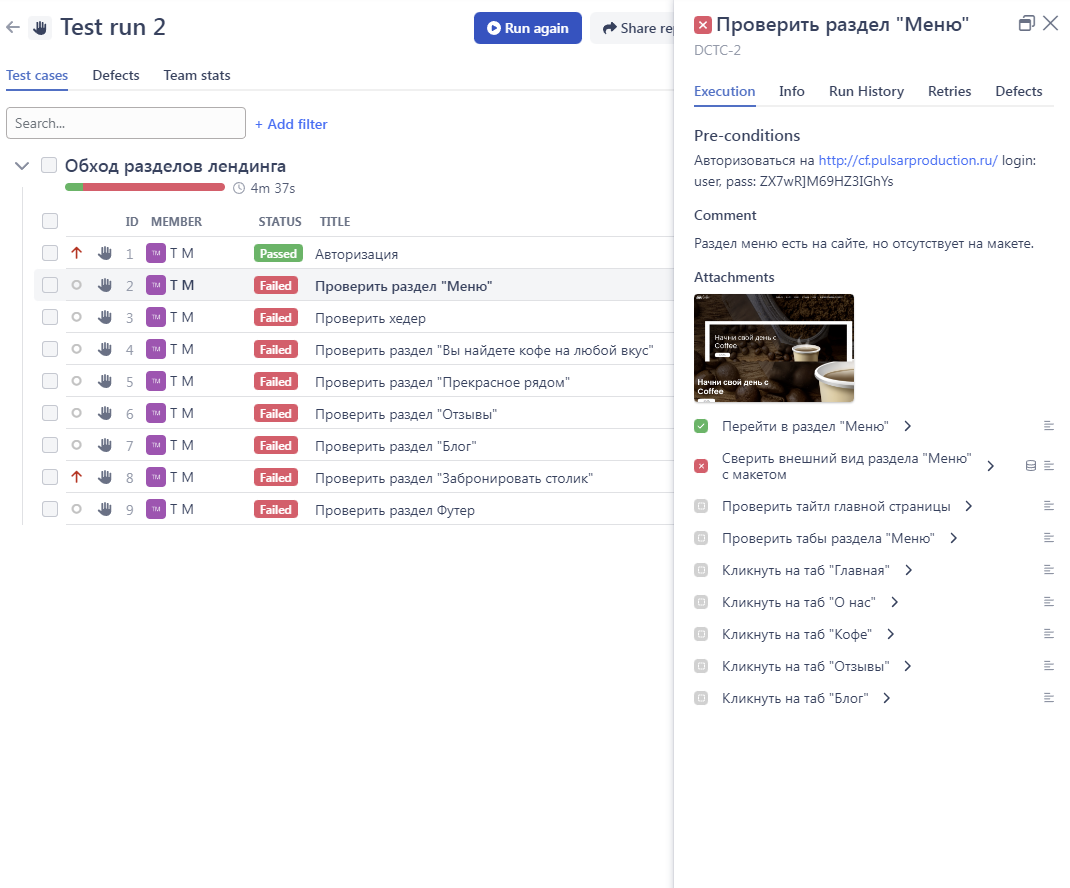
**Приложение 13**

**Выявленные дефекты в функционале лендинга Coffee в Qase.io**

****

**Приложение 14**

**Тест-ран тест-кейсов обхода разделов лендинга Coffee.**

****

**Приложение 15**

**Отчет о функциональном тестировании лендинга Coffee**

[**http://cf.pulsarproduction.ru/index.html**](http://cf.pulsarproduction.ru/index.html)

|  |  |  |
| --- | --- | --- |
| **Составил** | **Роль** | **Дата** |
| Тринченко М.А. | тестировщик | 25.08.2023 |

**Оглавление**

Краткое описание ................................................................................................................. 3

Сведения о ходе испытаний ................................................................................................ 4

Затраты времени и ресурсов ................................................................................................ 8

Конфигурация оборудования и инструменты .................................................................... 8

Выводы по результатам испытаний .................................................................................... 9

Приложение 1........................................................................................................................ 10

**Краткое описание**

В ходе тестирования был применен метод тестирования по чек-листу и двум наборам тест-кейсов, которые проверяют внешний вид и работу лендинга Coffee:

<http://cf.pulsarproduction.ru/index.html>

Тесты чек-листа и тест-кейсов были составлены и сгруппированных в 8 разделов исходя из структуры лендинга описанной в функциональных требованиях тех.задания.

<https://docs.google.com/document/d/1w_yhOU8x9miePJRXe2e-sbAFC0Vy67jsv_e49kBVoEI/edit>

Дополнительный набор-тест кейсов проверяет корректность работы формы бронирования столика в разделе «Забронировать столик».

Суть тестирования заключалась в проверке наличия всех заявленных в ТЗ элементов лендинга, сверке их внешнего вида с макетом <https://www.figma.com/file/MzyKqkJD5c8IZAup3STJS0/Projects?type=whiteboard&node-id=0-1> и проверке корректной работы интерактивных элементов.

**Сведения о ходе испытаний**

**1. Проверки, пройденные в ходе тестирования:**

* 1. **Проверка по чек-листу:**

Чек-лист с проверками доступен по ссылке:

<https://app.qase.io/project/CL>

**1.2. Проверка по тест-кейсам разделов:**

Набор тест-кейсов доступен по ссылке:

<https://app.qase.io/project/DCTC?case=11&suite=2>

**1.3. Проверка по тест-кейсам Формы бронирования столика:**

Набор тест-кейсов доступен по ссылке:

<https://app.qase.io/project/DCTC?case=11&suite=2>

**2. Результаты прохождения проверок:**

**2.1. Тест-ран чек-листа**

Результат тест-рана по чек-листу доступен по ссылке:

<https://app.qase.io/run/CL/dashboard/5/e355ba53f4efada4c835e317e5090c4e3049b805>

Список выявленных дефектов доступен по ссылке:

<https://app.qase.io/run/CL/dashboard/5/e355ba53f4efada4c835e317e5090c4e3049b805#defects-tab>

В ходе проверки было пройдено 129 проверок по 8 разделам лендинга, по результатам тест-рана были получены следующие данные:

Всего проверок – 129, Passed – 34 (26%), Failed – 66 (51%), Blocked – 29 (23%)

Рис.1 Результаты прохождения проверок чек-листа.

Прохождение проверок по чек-листу в разрезе разделов:

***1) Меню с табами***

Всего проверок – 12, Passed – 5, Failed – 7

***2) Хедер***

Всего проверок – 5, Passed – 3, Failed – 2

***3) Вы найдете кофе на любой вкус***

Всего проверок – 24, Passed – 1, Failed – 7, Blocked –16

***4) Прекрасное рядом***

Всего проверок – 8, Passed – 6, Failed – 2

***5) Отзывы***

Всего проверок – 11, Passed – 4, Failed – 7

***6) Блог***

Всего проверок – 11, Passed – 4, Failed – 7

***7) Забронировать столик***

Всего проверок – 39, Passed – 10, Failed – 26, Blocked – 3

***8) Футер***

Всего проверок – 19, Passed – 1, Failed – 8, Blocked –10

Рис.2 Распределение результатов проверок по разделам

**2.1.1. Выявленные дефекты по чек-листу:**

Список всех выявленных дефектов доступен по ссылке:

<https://app.qase.io/run/CL/dashboard/5#defects-tab>

Всего было выявлено 72 дефекта, по критичности дефекты распределяются следующим образом:

Blocker – 1 (1%), Critical – 15 (20%), Major – 10 (13%),

Normal – 5 (7%), Minor – 20 (28%), Trivial – 21 (29%)

Рис.3 Распределение дефектов по критичности.

Выявленные дефекты в разрезе разделов лендинга:

***1) Меню с табами***

Всего дефектов – 3, Major и Normal – 3

***2) Хедер***

Всего дефектов – 2, Major и Normal – 1, Minor и Trivial – 1

***3) Вы найдете кофе на любой вкус***

Всего дефектов – 8, Major и Normal – 1, Minor и Trivial – 7

***4) Прекрасное рядом***

Всего дефектов – 2, Major и Normal – 1, Minor и Trivial – 1

***5) Отзывы***

Всего дефектов – 5, Minor и Trivial – 5

***6) Блог***

Всего дефектов – 7, Major и Normal – 1, Minor и Trivial – 6

***7) Забронировать столик***

Всего дефектов – 37, Critical и выше – 14, Major и Normal – 6, Minor и Trivial – 17

***8) Футер***

Всего дефектов – 8, Critical и выше – 2, Major и Normal – 4, Minor и Trivial

Рис.4 Распределение дефектов по разделам лендинга

Выявленные по результатам тестирования дефекты уровня критический и выше были внесены в систему управления тестированием Jira как указано в Приложении 1.

**2.2. Проверка по тест-кейсам обхода разделов лендинга.**

Результат тест-рана по тест-кейсам обхода разделов лендинга доступен по ссылке:

<https://app.qase.io/run/DCTC/dashboard/2/a07a881995336fa42f66a0d53f7581ff652bad3a>

В целом данный дизайн тест-кейсов следует признать неудачным для начальных этапов тестирования лендинга, все тест-кейсы проваливаются на первом же найденном дефекте. Результат: успешно пройден только тест-кейс авторизации, оставшиеся 8 тест-кейсы разделов провалены на тривиальных дефектах наподобие несоответствия шрифтов заголовков макету.

* 1. **Проверка по тест-кейсам формы бронирования столика**

Проверка по тест-кейсам формы бронирования столика не проводилась ввиду большого количества критических дефектов формы бронирования столика, выявленных при прохождении проверки по чек-листу.

1. **Затраты времени и ресурсов.**

Тестирование лендинга по чек-листу проводилось 1 тестировщиком.

Оценка затрат времени на тестирование:

1. Анализ технического задания и макета – 3 часа

2. Составление тест-плана – 1 час

3. Составление чек-листа – 3 часа

4. Составление тест-кейсов – 6 часов

5. Тест-ран чек-листа – 3 часа

6. Тест-ран тест-кейсов обхода разделов – 0 часов

7 Тест-ран тест-кейсов формы бронирования столика – не проводился

7. Заведение баг-репортов на критические баги – 3 часа

8. Составление отчета о тестировании – 1 час

Итого: 17 часов

1. **Конфигурация тестового оборудования и инструменты:**

**4.1. Рабочее место:**

Тип ПК: десктоп

Операционная система: Windows 10

Браузер: Chrome

**4.2. Тестовые инструменты:**

Qase.io – составление чек-листа, тест-ран, первичный учет дефектов

Jira – заведение баг-репортов

Chrome DevTools – проверка кода элементов веб-страницы

**Выводы по результатам тестирования.**

**1. Выводы о готовности продукта.**

Тестирование лендинга Coffee выявило низкую степени готовности как веб-приложения, так и его макета. Успешно пройдено всего 26 % тестов, 51% тестов провалены и 23% тестов заблокированы неработающими элементами приложения. В разрезе разделов наиболее низкую степень готовности показали разделы «Вы найдете кофе на любой вкус», «Забронировать столик» и «Футер».

Было выявлено 72 дефекта, из них 16 (21%) уровня критичекий и выше. Разделы с наибольшим числом критических дефектов – «Забронировать столик» (14) и «Футер» (2).

По результатам тестирования лендинг следует признать не готовым к релизу и отправить на доработку в первую очередь работоспособности Формы бронирования столика, а также приведения в соответствие содержимого разделов лендинга и его отражения на макете техническому заданию.

**2. Выводы о процессе тестирования.**

Тестирование по чек-листу показало себя действенным инструментом по первоначальной оценке общей степени готовности лендинга, вместе с тем оно не позволяет подробно исследовать причины возникающих дефектов.

Тестирование по тест-кейсам с разбивкой по разделам показало себя неэффективным на первоначальной стадии тестирования, поскольку выполнение тест-кейса раздела прерывается при первом же выявленном дефекте. Такой сценарий больше подходит для приемочного тестирования.

Нельзя недооценивать роль работы с ТЗ и макетом. Необходимо провести проверку и доработку ТЗ и статическое тестирование макета по ТЗ. Это поможет избежать проблем в тестировании самого приложения.

Продолжение тестирования по тест-кейсам формы бронирования столика не представляется возможным ввиду большого количества выявленных критических дефектов, например отсутствия поля Дата, отсутствия возможности получить проверочный код, неправильных форматов отдельных полей. Этот этап следует отложить до устранения критических дефектов раздела «Забронировать столик». Составленные тест-кейсы после минимальной доработки можно будет использовать для тестирования формы.

**3. Рекомендации по улучшению приложения.**

- Доработать макет веб-страницы добавив отсутствующие элементы интерфейса.

- Устранить критические дефекты Формы бронирования столика.

- Добавить форму регистрации и задокументировать процесс получения проверочного кода по смс.

- Форма авторизации не должна препятствовать просмотру лендинга

**Приложение 1**

**Баг-репорты на дефекты с уровнем критический и выше в Jira**

