Чеклист для тестирования веб-приложения.

Тестирование веб-приложения в простых терминах - это проверка вашего веб-приложения на потенциальные ошибки до его запуска или до переноса кода в рабочую среду.

Во время этой стадии проверяются такие проблемы, как безопасность веб-приложения, работоспособность сайта, его доступ для обычных пользователей и способность обрабатывать трафик.

Виды тестирования веб-приложений

Функциональное тестирование;

Что такое функциональное тестирование?

1. Тестирование особенностей и операционного поведения продукта, чтобы гарантировать, что они соответствуют его спецификациям.
2. Тестирование, которое игнорирует внутренний механизм системы или компонента и фокусируется исключительно на выходных данных, генерируемых в ответ на выбранные входы и условия выполнения.

Цель функционального тестирования:

Цель функционального тестирования - проверить, соответствует ли ваш продукт намеченным функциональным спецификациям, указанным в документации разработки.

Включенные в тестирование действия:

Проверьте, работают ли все ссылки на ваших веб-страницах правильно, и убедитесь, что нет сломанных ссылок.
Ссылки, которые нужно проверить, включают:

1. Исходящие ссылки
2. Внутренние ссылки
3. Ссылки-якоря
4. Ссылки на почту (MailTo Links).

Тестирование работы форм. Это будет включать:

1. Проверка того, что проверки скриптов на форме работают, как ожидалось. Например, если пользователь не заполняет обязательное поле в форме, то показывается сообщение об ошибке.
2. Проверка того, что значения по умолчанию заполняются.
3. После отправки данные из формы отправляются в рабочую базу данных или связываются с рабочим адресом электронной почты.
4. Формы оптимально отформатированы для лучшей читабельности.

Тестирование работы cookies (куки). Cookies - это небольшие файлы, используемые веб-сайтами, главным образом, для запоминания активных пользовательских сеансов, чтобы вам не нужно было каждый раз входить на сайт. Тестирование cookies будет включать:

1. Тестирование того, что cookies (сеансы) удаляются либо при очистке кэша, либо когда они истекают.
2. Удаление cookies (сеансов) и проверка того, что при следующем посещении сайта требуются учетные данные для входа.

Тестирование HTML и CSS, чтобы убедиться, что поисковые системы могут легко индексировать ваш сайт. Это будет включать:

1. Проверка на наличие синтаксических ошибок.
2. Читабельные цветовые схемы.
3. Соответствие стандартам. Убедитесь, что соблюдаются такие стандарты, как W3C, OASIS, IETF, ISO, ECMA или WS-I.

Тестирование бизнес-потока. Это будет включать:

1. Тестирование вашего рабочего процесса/бизнес-сценариев от начала до конца, которые приводят пользователя через серию веб-страниц для завершения.
2. Тестирование отрицательных сценариев, когда пользователь выполняет неожиданный шаг, должно появляться соответствующее сообщение об ошибке или помощь в вашем веб-приложении.

Сценарии функционального тестирования:

1. Проверка, что все обязательные поля должны проходить валидацию.
2. Проверка, что знак звездочки должен отображаться для всех обязательных полей.
3. Проверка, что система не должна отображать сообщение об ошибке для необязательных полей.
4. Проверка правильности валидации високосных годов и отсутствия ошибок/неправильных расчетов.

5. Проверка, что числовые поля не должны принимать буквы, а должно отображаться соответствующее сообщение об ошибке.
6. Проверка отрицательных чисел, если они разрешены для числовых полей.
7. Проверка того, что деление на ноль должно быть обработано правильно при расчетах.
8. Проверка максимальной длины каждого поля, чтобы убедиться, что данные не обрезаются.
9. Проверка того, что появляется всплывающее сообщение ("Это поле ограничено 500 символами"), если данные достигают максимального размера поля.
10. Проверка того, что при операциях обновления и удаления отображается сообщение с подтверждением.
11. Проверка того, что значения сумм должны отображаться в формате валюты.
12. Проверка всех входных полей на наличие специальных символов.
13. Проверка функциональности тайм-аута.
14. Проверка функциональности сортировки.
15. Проверка функциональности доступных кнопок.
16. Проверка того, что политика конфиденциальности и ЧАВО четко определены и доступны для пользователей.
17. Проверка того, что если какая-либо функциональность не работает, пользователь перенаправляется на пользовательскую страницу ошибки.

тестирование удобства использования :

Что такое тестирование удобства использования ?

1. Тестирование удобства использования - это проверка удобства для пользователя.
2. В тестировании удобства использования проверяется, как легко новый пользователь может понять приложение, тестируя поток его использования.
3. В основном в тестировании удобства использования проверяется навигация в системе.

Цель тестирования удобства использования :

Тестирование удобства использования позволяет оценить удобство и эффективность продукта с использованием стандартных практик тестирования удобства использования.

Включенные в тестирование активности :

Тестирование навигации сайта:

1. Меню, кнопки или ссылки на разные страницы на вашем сайте должны быть легко заметны и последовательны на всех веб-страницах.
2. Тестирование контента.
3. Контент должен быть читаемым без ошибок в орфографии и грамматике.

4. Изображения, если они есть, должны содержать текст "alt".

Сценарии тестирования удобства использования:

1. Контент веб-страницы должен быть правильным без ошибок в орфографии и грамматике.
2. Все шрифты должны быть такими же, как требуется.
3. Весь текст должен быть правильно выровнен.
4. Все сообщения об ошибках должны быть правильными без ошибок в орфографии и грамматике, и сообщение об ошибке должно соответствовать метке поля.
5. Для каждого поля должен быть текст всплывающей подсказки.
6. Все поля должны быть правильно выровнены.
7. Должно быть достаточно места между метками полей, столбцами, строками и сообщениями об ошибках.
8. Все кнопки должны быть в стандартном формате и размере.
9. Домашняя ссылка должна присутствовать на каждой странице.
10. Отключенные поля должны быть серыми.
11. Проверить наличие неисправных ссылок и изображений.
12. Должно отображаться сообщение с подтверждением для любого вида операций обновления и удаления.
13. Проверить сайт на разных разрешениях (640 x 480, 600x800 и т. д.).
14. Проверить, что конечный пользователь может запустить систему без фрустрации.
15. Если при отправке появляется сообщение об ошибке, информация, заполненная пользователем, должна быть там.
16. Заголовок должен отображаться на каждой веб-странице.
17. Все поля (текстовое поле, раскрывающийся список, радиокнопка и т. д.) и кнопки должны быть доступны с помощью горячих клавиш, и пользователь должен иметь возможность выполнять все операции с помощью клавиатуры.
18. Проверить, не обрезаются ли данные в раскрывающемся списке из-за размера поля, и также проверить, управляются ли данные жестко или управляются ли они администратором.

Тестирование интерфейса :

Здесь необходимо протестировать три области - приложение, веб-сервер и базу данных.

1. Приложение: проверить, что запросы правильно отправляются в базу данных и вывод на стороне клиента отображается правильно. Ошибки, если

они есть, должны быть пойманы приложением и показываться только администратору, а не конечному пользователю.

2. Веб-сервер: проверить, что веб-сервер обрабатывает все запросы приложения без отказа в обслуживании.

База данных: убедиться, что запросы, отправленные в базу данных, дадут ожидаемые результаты.

3. Протестировать ответ системы, когда связь между тремя уровнями (приложение, веб-сервер и база данных) не может быть установлена, и конечному пользователю отображается соответствующее сообщение.

Тестирование базы данных

База данных является одним из критически важных компонентов вашего веб-приложения, поэтому следует уделить должное внимание ее тщательному тестированию. В тестировании должны быть включены следующие действия:

1. Проверка наличия ошибок при выполнении запросов.
2. Поддержание целостности данных при создании, обновлении или удалении данных в базе данных.
3. Проверка времени ответа запросов и их настройка при необходимости.

Проверка точности отображения данных, полученных из базы данных, в вашем веб-приложении.

1. Для выполнения тестирования базы данных тестировщик должен учитывать следующие моменты:
2. Тестировщик должен полностью понимать функциональные требования, бизнес-логику, поток приложения и дизайн базы данных.
3. Тестировщик должен определить таблицы, триггеры, хранимые процедуры, представления и курсоры, используемые для приложения.
4. Тестировщик должен понимать логику триггеров, хранимых процедур, представлений и курсоров, созданных для приложения.
5. Тестировщик должен определить таблицы, которые затрагиваются при выполнении операций вставки, обновления и удаления (DML) через веб- или настольные приложения.

Сценарии тестирования базы данных:

1. Проверка имени базы данных: имя базы данных должно соответствовать спецификациям.
2. Проверка таблиц, столбцов, типов столбцов и значений по умолчанию.

Все должно соответствовать спецификациям.

3. Проверка, разрешается ли столбцу иметь пустое значение или нет.
4. Проверка первичных и внешних ключей каждой таблицы.

5. Проверка хранимой процедуры.
6. Тестирование установленности хранимой процедуры.
7. Проверка имени хранимой процедуры.
8. Проверка имени параметров, их типов и количества.
9. Проверка обязательности параметров.
10. Тестирование хранимой процедуры путем удаления некоторых параметров.
11. Проверка, что при выводе нулевых значений, нулевые записи будут затронуты.
12. Тестирование хранимой процедуры путем написания простых SQL-запросов.
13. Проверка, возвращает ли хранимая процедура значения.
14. Тестирование хранимой процедуры с примерами входных данных.
15. Проверка поведения каждого флага в таблице.
16. Проверка сохранения данных в базе данных после каждой отправки страницы.
17. Проверка данных, если выполняются операции DML (обновление, удаление и вставка).
18. Проверка длины каждого поля. Длина поля в backend и frontend должна быть одинаковой.
19. Проверка имен баз данных QA, UAT и продукции. Имена должны быть уникальными.
20. Проверка зашифрованных данных в базе данных.
21. Проверка размера базы данных. Также проверка времени отклика каждого выполненного запроса.
22. Проверка отображаемых данных на frontend и убедиться, что они совпадают с данными в backend.
23. Проверка допустимости данных, введенных в базу данных.
24. Проверка триггеров.

Тестирование совместимости :

Что такое тестирование совместимости?

Тестирование совместимости используется для определения совместимости вашего программного обеспечения с другими элементами системы, с которой оно должно работать, например, браузерами, операционными системами или аппаратным обеспечением.

Цель тестирования совместимости:

Цель тестирования совместимости заключается в оценке того, как хорошо программа работает в определенном браузере, операционной системе, аппаратном или программном обеспечении. Тестирование совместимости обеспечивает правильное отображение вашего веб-приложения на разных устройствах.

Включенные в тестирование деятельности:

Тестирование совместимости браузеров: один и тот же веб-сайт в разных браузерах будет отображаться по-разному. Вам нужно проверить, отображается ли ваше веб-приложение правильно в разных браузерах, работает ли JavaScript, AJAX и аутентификация. Вы также можете проверить совместимость мобильных браузеров.

Отображение элементов веб-страниц, таких как кнопки, текстовые поля и т. д., изменяется при изменении операционной системы. Убедитесь, что ваш веб-сайт работает хорошо для различных комбинаций операционных систем, таких как Windows, Linux, Mac и браузеров, таких как Firefox, Internet Explorer, Safari и т. д.

Сценарии тестирования совместимости:

1. Проверьте веб-сайт в разных браузерах (IE, Firefox, Chrome, Safari и Opera) и убедитесь, что он отображается правильно.
2. Проверьте, что используемая версия HTML совместима с соответствующими версиями браузеров.
3. Проверьте, что изображения правильно отображаются в разных браузерах.
4. Проверьте, что шрифты могут использоваться в разных браузерах.
5. Проверьте, что код JavaScript может использоваться в разных браузерах.
6. Проверьте анимированные GIF-изображения в разных браузерах."

Производительность тестирования

Что такое тестирование производительности?

Тестирование производительности проводится для оценки соответствия системы или компонента заданным требованиям производительности.

Включенные в тестирование виды деятельности:

1. Время отклика веб-приложения на разных скоростях подключения.
2. Тестирование нагрузки на ваше веб-приложение для определения его поведения при нормальных и пиковых нагрузках.
3. Стресс-тестирование вашего сайта для определения точки его выхода из строя при превышении нормальной нагрузки в пиковое время.
4. Тестирование, что произойдет, если произойдет сбой из-за пиковой нагрузки, и как сайт восстановится после такого события.

Обеспечьте включение техник оптимизации, таких как сжатие gzip, кэширование браузера и сервера, для сокращения времени загрузки.

5. Общие сценарии тестирования:

6. Определение производительности, стабильности и масштабируемости приложения в различных условиях нагрузки.
7. Определение того, может ли текущая архитектура поддерживать приложение на пиковых уровнях пользователей.
8. Определение того, какая конфигурация обеспечивает наилучший уровень производительности.
9. Определение узких мест приложения и инфраструктуры.
10. Определение того, если новая версия программного обеспечения негативно повлияла на время отклика.
11. Оценка продукта и/или оборудования для определения того, сможет ли оно обрабатывать запроектированные объемы нагрузки."

Тестирование безопасности

Тестирование безопасности включает проверку на наличие ошибок и пробелов в аспектах безопасности.

Сценарии тестирования безопасности:

1. Проверка, что веб-страницы, содержащие важные данные, такие как пароли, номера кредитных карт, ответы на секретные вопросы для безопасности и т.д., отправляются через HTTPS (SSL).
2. Проверка, что важная информация, такая как пароли, номера кредитных карт и т.д., отображается в зашифрованном формате.
3. Проверка реализации правил пароля на всех страницах аутентификации, таких как регистрация, забытый пароль, изменение пароля.
4. Проверка, что после изменения пароля пользователь не может войти с использованием старого пароля.
5. Проверка того, что сообщения об ошибках не отображают важную информацию.
6. Проверка того, что если пользователь вышел из системы или время его сеанса истекло, он не может продолжить работу на сайте.
7. Проверка доступа к защищенным и незащищенным веб-страницам напрямую без входа в систему.
8. Проверка того, что опция «Просмотреть исходный код» отключена и не видна пользователю.
9. Проверка блокировки учетной записи пользователя, если он вводит неправильный пароль несколько раз.
10. Проверка того, что куки не хранят пароли.
11. Проверка того, что если какая-либо функциональность не работает, система не отображает никакой информации о приложении, сервере или базе данных. Вместо этого должна отображаться настраиваемая страница ошибки.
12. Проверка на SQL-инъекции.

13. Проверка ролей пользователей и их прав. Например, запрос не должен иметь доступ к странице администратора.
14. Проверка того, что важные операции записываются в журнал, и что информацию можно отследить.
15. Проверка того, что значения сеанса находятся в зашифрованном формате в адресной строке.
16. Проверка того, что информация о куки хранится в зашифрованном формате.
17. Проверка приложения на атаки методом перебора."