13. Тестування веб-додатків (Web Testing Specific)

Веб-тестування є важливою частиною розроблення будь-якого веб-додатку або сайту.

На сьогодні кількість програмного забезпечення, що використовується на кожному комп'ютері є величезною, тому, неправильне тестування може призвести до катастрофічних наслідків, збитки від яких можуть коштувати мільйони, а то й мільярди доларів.

Тестування веб-додатків є комплексним завданням, яке вимагає глибоких знань і досвіду, а також володіння різноманітними методами тестування.

Особливості тестування простих веб-додатків

Прості веб-додатки включають різні сайти, електронні магазини, а також SPA-додатки, розміщені на одній сторінці, і прості веб-сервіси.

Для тестування використовується стандартний набір тестів.

Особливості тестування комплексних веб-додатків

До цієї групи входять різні інтернет-портали, соціальні мережі, інтернет-аукціони, торгові майданчики та інше.

Оскільки подібні веб-додатки мають розширену функціональність і працюють під високим навантаженням, тестувальники часто спільно з бізнес-аналітиками розробляють індивідуальні стратегії тестування, пишуть тест-плани та сценарії користувача.

Особливості тестування веб-додатків підвищеної складності

Це в першу чергу SaaS рішення, а також різні інноваційні продукти, пошукові системи, брокерські торгові системи, платіжні системи тощо.

При тестуванні таких веб-додатків залучаються бізнес-аналітики, спільно з якими проводиться глибокий аналіз і планується робота над проектом. Розробляються індивідуальні тест-кейси та сценарії, а також тестується додаток на відповідність певним стандартам.

Належне використання інструментів для веб-тестування або автоматизації веб-тестування може бути реалізовано наступними способами:

1. Тестування функціональності

Функціональне тестування є найбільш базовим, але надзвичайно важливим для будь-якої програми, у тому числі веб-додатку. Функціональне тестування гарантує, що додаток працює правильно і без помилок.

Функціональне тестування охоплює:

- **Модульне тестування:** на цьому етапі функціонального тестування перевіряються невеликі окремі області застосування на ранніх стадіях розробки, щоб зменшити ймовірність появи більш серйозних помилок в подальшому.

- **Димове тестування (Smoke testing)** - мінімальний набір тестів на явні помилки. Програму, що не проходила цей тест, не має сенсу віддавати на більш глибоке тестування. Наприклад, Помилки інсталяції: якщо програмний продукт не встановлюється, його тестування швидше за все виявиться неможливим; Помилки при з'єднанні з базою даних; Помилки завантаження конфігурації та отримання налаштувань для ініціалізації під час запуску.
- **Санітарне тестування (Sanity Testing)**: специфічна перевірка працездатності окремих функціональних елементів, систем, веб-архітектур та розрахунків. Вона виконується з єдиною метою – дати 100% поняття того, що створена система працює саме так, як було потрібно. Подібний тип тестування виконується після закінчення димового тестування. Будь-які дефекти, знайдені на ранній стадії, зможуть заощадити масу часу на процесі перевірки недоробленого складання.
- **Інтеграційне тестування**: важливий аспект тестування програмного забезпечення, який призначений для оцінки того, наскільки ефективно різні програми інтегруються один з одним. Знаходить помилки у роботі взаємопов'язаних модулів (наприклад, скерування на сторінку поштової скриньки після успішної реєстрації).

2. Тестування веб-API

Веб-API є інтерфейсами прикладного програмування для Інтернету. Тестування API вимагає виконання запитів до кількох кінцевих точок API для перевірки відповіді, включаючи функціональність, безпеку та продуктивність. Воно має ключове значення, оскільки воно перевіряє вузькі місця логіки, відповідей, безпеки та продуктивності.

Наприклад, припустимо, що ви хочете протестувати форму входу. Зокрема, ви хочете, щоб кожне введене ім'я користувача та пароль правильно зберігалося у базі даних.

- У термінах API ви перевіряєте, що обмін даними HTTP зашифрований і захищений від потенційних атак. При цьому слід звертати увагу на такі негативні сценарії як:
- Доступ до API без дотримання правил аутентифікації.
- Перевірка неприпустимих значень у JSONS (наприклад, неможливо зареєструвати користувача з неіснуючою адресою електронної пошти).
- Тестування файлів cookie — це ще один аспект, на який слід звернути увагу при надсиланні запитів користувача та сеансів. При використанні API як посередника способи видалення або зберігання інформації про сеанси та дії користувачів повинні включатися до тестових сценаріїв API.

3. Тестування баз даних

Тестування баз даних гарантує, що значення даних у базі даних є правильними. Це допоможе запобігти втраті даних, зберегти відомості про зміни та запобігти несанкціонованому доступу до інформації.

Тестування включає перевірку схеми бази даних та таблиць. Часто містить стрес-тестування та використання складних запитів на одному або кількох файлах даних. Тестування баз даних забезпечує впевненість у тому, що це дані передаються успішно, незалежно від кількості запитів до базі.

4. Регресійне тестування

Робота з великомасштабним програмним забезпеченням означає постійне внесення змін. Насправді, якщо програмне забезпечення не призначене виключно для особистого використання, за умовчанням слід очікувати нових функцій та модифікацій коду.

Зміни коду можуть виявити несподівані залежності та збої. Регресійне тестування гарантує, що програма, як і раніше, працює після внесення змін до коду, оновлень або інших змін. Це важливий крок, оскільки він відповідає за загальну стабільність та працездатність існуючої функціональності.

Регресійне тестування найчастіше підходить для автоматизації. Тест кейси, які охоплюють основні сфери використання програмного забезпечення: сторінка авторизація та домашня сторінка – всі вони можуть стати об'єктами автоматизації, залишивши для ручного тестування складніші випадки.

5. Тестування сумісності

Тестування сумісності гарантує, що основні функції веб-додатку будуть доступні користувачам у різних браузерах, різних пристроях та операційних системах. Причому це стосується не лише різних операційних систем, а й різних їх версій.

Не всі операційні системи, браузери та пристрої влаштовані однаково. Це може призвести до різної поведінки веб-додатків у різних тестових середовищах. Саме тому тестування сумісності таке важливе.

Залежно від того, звідки підключаються користувачі, їх клієнти можуть різнитися навігацією між сторінками або навіть спосіб підключення до API. Старі версії браузерів, операційних систем та мобільних пристроїв нікуди не зникають з часом – деякі користувачі продовжують їх використовувати.

Звичайно, тестування сумісності у всіх існуючих браузерах, платформах та пристроях неможливо. Щоб знати, на чому зосередитись, варто з'ясувати пріоритети клієнта або популярність браузерів та пристроїв на перспективному ринку. Достатньої уваги приділити адаптивному тестуванню. Забезпечення узгодженості розмірів екрану, інтервалів, вертикального та горизонтального прокручування, вирівнювання та масштабування елементів не менш важливо, ніж працездатність програми в трьох найбільш поширених браузерах.

6. Тестування інтерфейсу користувача

Трапляється, що візуальні аспекти зовсім не беруться до уваги. Всі зусилля зосереджуються на функціональності, а не на тому, наскільки естетично виглядає інтерфейс користувача. Звертати увагу на нього починають лише тоді, коли команда розробників збільшується у розмірі та з'являється можливість виділити для цього відповідні ресурси.

У разі оновлення CSS або зміни макету можуть виникнути проблеми з дизайном. Причини можуть бути різними, але найбільш практичним рішенням є тестування. Не кожен браузер, ОС чи мобільний пристрій однаково обробляє дані. Нові версії браузерів використовують найсучасніші технології та найсучасніші можливості CSS і JavaScript.

Таке тестування вимагає пильної уваги, щоб переконатися, що всі тексти читаються так, як повинні, а всі зображення знаходяться у правильних місцях та мають вірні пропорції.

7. Тестування зручності використання (юзабіліті)

Це певна процедура дослідження рівня навчання користувача при його взаємодії з додатком. Саме тестування зручності використання вважається найважливішим моментом будь-якого інтернет-маркетингу.

Кінцева мета будь-якого юзабіліті-тестування полягає в тому, щоб ще на стадії розробки програмного забезпечення зрозуміти, чи зможуть клієнти, для яких безпосередньо створюється веб-продукт, самостійно розібратися в наданому інтерфейсі і як швидко вони це зроблять. Зручний до сприйняття сайт дозволяє істотно збільшити кількість людей, які стануть його користувачами як короткий так і тривалий проміжок часу.

Чим більше клієнтів, тим вища конверсія веб-сайту (відсоткове співвідношення кількості клієнтів, які здійснили певні дії на сайті, до загальної кількості відвідувачів за вибраний часовий проміжок) . Клієнти завжди отримують максимальне задоволення від взаємодії з правильно налаштованим та організованим контентом сайту.

Але, коли веб-дизайнери зайняті створенням прототипу майбутнього сайту, вони не можуть максимально об'єктивно оцінити майбутню зручність його використання через певні особистісні якості, а також через безпосередню участь у його створенні. Макет, який вони розробляють, завжди здаватиметься їм максимально зручним та зрозумілим для сприйняття.

Більшість незручностей і помилок виникає ще на стадії розробки макетів. Потім до проекту підключаються спеціалісти з відділу QA, на яких покладено завдання протестувати програмне забезпечення з боку кінцевого користувача.

8. Тестування веб-безпеки

Тестування веб-безпеки – один із найважливіших етапів веб-тестування. Зокрема, надійне веб-тестування гарантує, що ваш веб-додаток витримає будь-які спроби зламування або витоку даних.

Потенційні вразливості зберігання даних можуть призвести до витоку особистої інформації користувачів, включаючи вкрадені особисті дані, паролі та дані для входу в систему або банківську та фінансову інформацію. Витік будь-якої особистої інформації може призвести до надзвичайно дорогих юридичних проблем для розробників та власників бізнесу.

Тестування безпеки для Інтернету дозволяє:

1. Знайти потенційні вразливості та загрози.
2. Дотримуватись законів, що регулюють політику безпеки та конфіденційності даних в інтернеті.
3. Виконати сканування зверху вниз, щоб зрозуміти поточні можливості безпеки програми.
4. Планувати швидку реакцію на погрози чи спроби злому у майбутньому

9. Тестування продуктивності та швидкості завантаження

Останнім етапом є тестування продуктивності та швидкості завантаження. Після переконання, що програма працює, потрібно перевірити, чи потрібно для запуску програму не більше 2 секунд. Файли JavaScript, сторонні плагіни, різні розміри сторінок або компоненти, які завантажуються довше за інших – все це може зменшити швидкість роботи програми. Тестування швидкості завантаження надає підказки для оптимізації загального часу відгуку.

Важливо забезпечити оптимальну роботу веб-додатку, оскільки низька швидкість завантаження сторінок може негативно позначитися на досвіді користувача. Дослідження, проведене Kissmetrics, показало, що коефіцієнт конверсії знижується на 40% на сторінках, завантаження яких займає більше 3 секунд.

Висновок

Якісне тестування веб-додатків дуже важливе і при належній організації дозволяє не тільки підвищити якість програми, а й суттєво скоротити спрямовані на це витрати та зусилля.

Джерела інформації

1. Тестування веб-додатків https://xbsoftware.ru/testirovanie-po/polnij-tsykl/test-web-prilozheniya/
2. Критерії юзабіліті-тестування https://testmatick.com/ru/zachem-i-po-kakim-kriteriyam-provodit-yuzabiliti-testirovanie/
3. Типи тестування веб-додатків https://habr.com/ru/articles/715376/