**Завдання з QA**

1. В чому різниця між Manual Software Test Engineer та Automation Software Test Engineer? (\*\*розгорнута відповідь з порівнянням обов’язків\*\*)

Automation Software Test Engineer - це фахівець із забезпечення якості продукту, який використовує програмні засоби для створення тестів і перевірки результатів виконання.

У обов’язки QA Automation інженера входить:

- Аналіз вимог;

- Розробка позитивних/негативних / граничних сценаріїв;

- Опис автоматизаційних тест-кейсів;

- Розробка фреймворку для автоматизації продукту;

- Налаштування тестових оточень;

- Підготовка тестових даних;

- Аналіз тестового покриття;

- Написання автотестів, тестових скриптів;

- Інтеграція з системами автоматичного запуску або Continuos Integration;

- Виконання автоматичних тестів;

- Підтримка працездатності вже написаних тестів;

- Написання документації;

- Аналіз результатів прогонів авто-тестів;

- Баг-репортинг, загальні звіти про тестування.

Manual Software Test Engineer - це фахівець, який вміє дивитися на програмний продукт як очима розробника цього продукту, так і очима кінцевого користувача, який прагне почути і залишити задоволеними і тих, і інших.

В обов'язки Manual Software Test Engineer входять:

* аналіз вимог-Тестувальник повинен чітко розуміти, що має робити ПЗ і як воно повинно працювати, а також сприяти тому, щоб у всіх членів команди було однакове уявлення про це;
* складання тестової документації - для самого тестувальника це впевненість в тому, що баги не будуть пропущені, для інших учасників процесу-прозорість і можливість мати уявлення про те, які перевірки проводяться;
* безпосередньо функціональне та нефункціональне тестування продукту;
* оформлення баг-репортів і звітів про тестування.

1. Навіщо потрібна автоматизація на проєкті? Що саме підлягає для автоматизації? (\*\*які тести та чому?\*\*)

Автоматизація на проект:

* Це хороший спосіб підвищити ефективність, а також збільшити охоплення та швидкість тестування програмного забезпечення, коли вам потрібно повторювати одні й ті самі тестові сценарії.
* Ручне тестування всіх робочих процесів, полів та негативних сценаріїв вимагає більше часу та грошей (за певних умов).
* Важко тестувати багатомовні сайти вручну.
* Не потребує втручання людини. Запускаєте та переходьте до інших завдань.
* Збільшує швидкість виконання тестів.
* Допомагає збільшити тестування.
* Ручне тестування може набриднути, і наслідками стануть втрата залучення та поява помилок.

Практично кожна команда розробників працює над проектом, який критично залежить від термінів, а значить, що часу на застосування всіх передових практик завжди не вистачає. Те ж саме стосується стратегії тестування, оскільки тестування як вид діяльності не завжди є пріоритетом для команд розробки. Потрібно спробувати знайти баланс і зробити правильний вибір в залежності від типу розробляється додатки, часових рамок, використовуваного ПЗ для тестування і наявних ресурсів. Ось важливі типи тестів, які можна автоматизувати.

Модульне тестування

Це відмінний спосіб приступити до автоматизації тестування, оскільки модульні тести спрямовані лише на частину коду, в ході яких він перевіряється на працездатність, і не залежать від інших частин програми. Таким чином, розробники отримують більше інформації про роботу створеної функціональності. Завдяки сучасній культурі тестування багато команд використовують методологію розробки через тестування (test-driven development, TDD), при якій вони починають складати тести до написання коду. Таким чином гарантується якість і коду, і тестів.

Пріоритетні функції

Якщо у вас в плані десятки функцій і стислі терміни на їх розробку, ви можете виділити серед них ті, що мають високу ймовірність збоїв. Тестування подібних функцій потрібно починати якомога раніше.

Регресійні та інтеграційні тести

Інтеграційні тести використовуються для визначення того, чи працюють окремі модулі в додатку як група, А регресійні тести перевіряють, що функції програми працюють належним чином. Ці два тести зазвичай виконуються після змін / поліпшень програми, тому тестувальники постійно проводять ці тести. Автоматизація таких тестів економить величезну кількість часу, вивільняючи його для виконання інших типів тестів.

Навантажувальні тести і тести продуктивності

Для тестів продуктивності і навантажувальних тестів немає альтернативи у вигляді ручного тестування, оскільки необхідно моделювати сотні або тисячі користувачів, що працюють в різних умовах: з-під різних браузерів, в різних часових поясах, що використовують різні операційні системи і т. п.

Повторювані тестові сценарії

Це дуже важливі тести, які команди розробки змушені запускати мало не постійно. Наприклад, працездатність функції входу в систему-вона забезпечує можливість користуватися додатком, впливаючи на його доступність. Тому краще автоматизувати тестування і заощадити прорву часу тестувальників і розробників.

Базова функціональність (димові тести)

На відміну від інших тестів, димові тести не такі складні і відносно легко реалізовані. При цьому проходження цих тестів має вирішальне значення. Вони інформують команди розробки про те, чи правильно працюють базові функції програми, наприклад: чи відкривається вікно входу в додаток, чи можуть користувачі увійти в систему, чи доступний API, чи доступний додаток з різних місць і т. п.

1. \* Вибрати на власній розсуд сайт - allo.ua, [rozetka.com.ua](http://rozetka.com.ua) та ін.

\* Написати 2-4 мануальних тестів (\*\* в Excel, Microsoft Office та ін. текстові редактори\*\*)

|  |  |  |  |
| --- | --- | --- | --- |
| **Название:** | Логотип BrowserStack. | | |
| **Функция:** | Переходит на домашнюю страницу BrowserStack и проверяет логотип компании. | | |
| **Действие** | | **Ожидаемый результат** | **Результат теста:**   * пройден * провален * заблокирован |
| **Предусловие:** | |  | |
| Откройте сайт | | Сайт открыт, публикации выданы в качестве нисходящего списка, страница загружена | пройден |
| **Шаги теста:** | |  | |
| Проверка наличия логотипа на главной странице | | Логотип отображен верно, и виден для пользователя | пройден |
| **Постусловие:** | |  | |
| Сайт отображен корректно | | Логотип и остальные элементы отображены верно | пройден |

|  |  |  |  |
| --- | --- | --- | --- |
| **Название:** | Проверка наличия пунктов меню | | |
| **Функция:** | Проверяет наличие всех пунктов меню на главной странице BrowserStack. | | |
| **Действие** | | **Ожидаемый результат** | **Результат теста:**   * пройден * провален * заблокирован |
| **Предусловие:** | |  | |
| Откройте сайт | | Сайт открыт, публикации выданы в качестве нисходящего списка, страница загружена | пройден |
| **Шаги теста:** | |  | |
| Проверка наличия пунктов меню на главной странице | | Пункты меню отображены верно, и видны для пользователя | пройден |
| Проверка количества пунктов меню | | Пунктов меню 4, как и планировалось разработчиком | провален |
| **Постусловие:** | |  | |
| Сайт отображен корректно | | Пункты меню отображены верно и их нужное количество | пройден |

|  |  |  |  |
| --- | --- | --- | --- |
| **Название:** | Проверка наличия страницы с ценами | | |
| **Функция:** | Переходит в раздел ценообразования BrowserStack и проверяет заголовок. | | |
| **Действие** | | **Ожидаемый результат** | **Результат теста:**   * пройден * провален * заблокирован |
| **Предусловие:** | |  | |
| Откройте сайт | | Сайт открыт, публикации выданы в качестве нисходящего списка, страница загружена | пройден |
| **Шаги теста:** | |  | |
| Перейти на страницу с вариантами подписок на сайт | | Перенаправление на выбранную страницу | пройден |
| **Постусловие:** | |  | |
| Отображение страницы | | Страница отображена корректно, необходимые элементы показаны | пройден |

\* Автоматизувати написані тести (\*\*Самостійно обрати мову програмування та фреймворк для тестів\*\*)

Проект з автоматизованими тестами розташовані у GitHub:

\* Створити профіль у <https://github.com/> та завантажити написані автотести на свій репозіторій (\*\*по бажанню можу перевірити, мій нік - masharova у <https://github.com/>\*\*)