Тестова стратегія ВЕБ застосунку Trello

Підготував

Курп’як Олег Михайлович

Студент ІПЗ МП-1

Зміст

1. Огляд застосунку
2. Тестовий підхід. Набір тестів
3. Тестове середовище
4. Технології, які будуть використовуватись
5. Go/not go критерії
6. Пігдотовка релізу до деплою на UAT
7. Олгляд застосунку

Trello(<https://trello.com>) – це популярний веб застосунок, який дозволяє користувачам моніторити задачі на проекті, їх статус та останні апдейти. Застосунок реалізує Kanban дошку, яка показує статус задачі(картки). Кожна картка має виконавця, який відповідає за її реалізацію на певному етапі. В межах картки можна вести комунікацію(залишати коментарі) про поточний статус задачі. Над однією Канбан дошкою може працювати команда з декількох людей

1. Тестовий підхід. Набір тестів
   1. Тестовий підхід

Даний пункт визначає набір функціоналу, який підлягає тестуванню, його важливість (від 1 до 4, де 4 – це критичний функціонал, а 1 – не критичний), тип тестування (UI/API), спосіб тестування (ручне/атоматичне), чи є частиною смоук тестів(так/ні).

Кожен реліз має бути посністю протестований та містити 0 або як можна менше багів. Якщо певний реліз містить баги, то в залежності від важливості функціоналу, де ці баги виявлено, приймається рішення про “go/not go” для конкретного релізу. Більше інформації буде в пункт 6.

Тип тестування – кожен фунціонал може бути протестований одним або декількома способами:

* UI – тестування веб інтерфесу. Може бути ручним або автоматизованим
* API – тестування HTTP REST API. Лише автоматичне тестування

Способи тестування:

* Ручне – певний набір функцій може бути складно протестувати автоматично або це вимагає надто багато зусиль, тому такий функціонал може бути протестований вручну
* Автоматизоване тестування – передбачає набір автотестів, які можуть бути виконані в будь-який момент. Такий набір тестів генерує звіт про результати тестування

Смоук тести – це мінімальний набір тестів, які показують чи правильно функціонує мінімальний набір функціоналу в систері. Такі тести не прив’язані до певного реліку та мають виконуватись постійно(з певною періодичністю)

* 1. Набір тестів

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Номер вимоги | Опис фунціоналу | Важливість | Тип тестування | Спосіб тестування | Смоук тест |
| UI\_AUTH\_1 | Користувач має можливість зареєструватись на сайті використовують унікальний імейл та пароль | 4 | UI | автоматичне | так |
| UI\_AUTH\_1.1 | Після вдалої реєстрації користувач має отримати імейл про підтвердження реєстрації | 2 | UI | ручне | ні |
| UI\_AUTH\_2 | Зареєстрований користувач може залогінитись в систему використовують свій імейл та пароль | 4 | UI/API | автоматичне | так |
| UI\_BRD\_1 | Користувач може створити нову дошку | 4 | UI | автоматичне | так |
| UI\_BRD\_2 | Користувач може додавати нові картки до дошки | 4 | UI | автоматичне | так |
| UI\_BRD\_3 | Користувач може запрошувати нових користувачів приєднатись до дошки | 3 | UI/API | автоматичне | ні |
| UI\_BRD\_3.1 | Новозапрошений користувач має отримати сповіщення на пошту, про те що його додали до нової дошки | 3 | UI | ручне | ні |
| UI\_BRD\_4 | Користувач може залишити коментар в карточці | 3 | UI/API | Автоматичне | так |

1. Тестове середовище

Для QA команди буде створено окреме середовище, де вони зможу протестувати кожен реліз кандидат. Це середовище має мати такі ж налаштування як і продакшин середовище, прицьому вимоги до hardware є нижчими(приклад – сервер БД може бути слабшим). Недопускається щоб QA середовище мало відмінну версію БД від тієї, яка є в продакшині.

Деплой нового реліз кандидата в QA середовища допускається лише зі згоди QA Lead та за умови що реліз кандидати повністю пройшов всі юніт тести

1. Технології, які будуть використовуватись
   1. Для ручного тестування

Для ручного тестування дозволено використовувати наступні програми:

1. Бразер Chrome/Firefox/Edge/Safari такої версії, яка відрізняється від останьої доступної не більше, ніж на 5 релізів. Приклад: поточна доступна версія Chrome v91, тому для тестування можна використовувати версію не старішу, ніж v86
2. Postman – програма, яка дозволяє виконувати HTTP запити з використанням графічного інтерфейсу
   1. Для автоматизованого тестування

Для автоматичного тестування дозволено використовувати наступні програми та технології

1. Java 17 – основна мова на якій буде реалізовано тест фреймворк
2. Cucumber – DSL для написання сценаріїв
3. Selenium – бібліотека, яка дозволяє автоматизувати тестування в браузері. Можна використовувати лише браузер, які переліченні в пунтті 4.1
4. Maven – система для збірки проекту

Після завершення тестування звіт має бути представлений у вигляді Office Excel файлу

1. Go/not go критерії

Після того як реліз кандидат пройшов тестування, менеджери мають прийняти рішення про те чи допускається даний реліз кандидат до деплою на UAT. У випадку, якщо реліз кандидати пройшов тестування без багів, то такий кандидат можна деплоїти на UAT. У випадку, якщо реліз кандидат пройшов тестування і при цьому виявленно помилки, то прийняття рішення про подальший деплой має відматись на основі важливості функціоналу, який даний реліз додає, та на основі помилок які присутні.

Якщо реліз кандадати містить **НЕ більше** 2 багів у фунціоналі з рівнем важливості 1 або 2 **і при цьому** реліз кандидат містить новий функціонал з важливістю 4, то такий реліз кандидат допускається до деплою на UAT. У всіх інших випадках, коли реліз кандидат містить якісь баги, то такий реліз кандидат недопускається до деплою на UAT

1. Пігдотовка релізу до деплою на UAT

Реліз кандидат, який допущений до деплою на UAT скалажається з набору сервісів та конфігурацій. Список сервісів, які будуть задеплоїні, на їхні версії потрібно брати з QA середовища, куди був задеплоїний цей резіл кандидат. Налаштування, які були застосовані на QA середовищі для цього реліз кандидату, такою маєють бути застосовані і на UAT середовищі, але з врахуванням відмінностей між цими середовищами (приклад – логіни та паролі для приєднання до третіх систем)