**Проект 1: регрессионное тестирование**

1. **Тест-кейсы:**

<https://docs.google.com/spreadsheets/d/1FHiOCEu3uKZN8BcMSxaaf7vY37TFzAZSuOOIU3mjL3c/edit?usp=sharing>.

1. **Баг-репорты:**

<https://docs.google.com/spreadsheets/d/1UEHMFqQi5IF2FBcTVU5AknX9nNtOA_nVtxoAzKBIoOw/edit?usp=sharing>.

1. **Выводы о работе:**

1) На тестирование ушло два вечера. Приходил после работы и занимался где-то с 21:00 до 23:00.

2) Работал на ОС Windows 8.1, браузер Google Chrome версия 109.0.5414.168.

3) Удалось выловить 8 багов, о чем составлены соответствующие БР.

4)Не все баги серьезно будут мешать работе пользователей.

Например, неверные сообщения или их отсутствие о недопустимом числе символов в полях профиля (Б4, Б5, Б6, Б7 <https://docs.google.com/spreadsheets/d/1UEHMFqQi5IF2FBcTVU5AknX9nNtOA_nVtxoAzKBIoOw/edit?usp=sharing>).

А невозможность производить редактирование профиля, наоборот, полностью лишит пользователя возможности работать с приложением (Б1, Б2, Б3 <https://docs.google.com/spreadsheets/d/1UEHMFqQi5IF2FBcTVU5AknX9nNtOA_nVtxoAzKBIoOw/edit?usp=sharing>).

5)По поводу **тест-кейсов**. Отсутствует ссылка на тестируемое приложение, пришлось брать из задания. Отсутствовал раздел окружения. Непонятно устройство на котором выполнять тестирование, не указала ОС и версия ПО.

Также, на мой взгляд, тест кейсы не удачно расставлены, они хаотично разбросаны. Я бы их сгруппировал подряд по функционалу:

ТК по проверке открытия, закрытия и положения окна профиля;

ТК, связанные с выводом текстовых сообщений об ошибке под полями “Имя” и “Занятие”;

ТК, связанные с проверкой активности кнопки “Сохранить” при разных вариантах заполненности полей “Имя” и “Занятие”.

Еще бы внес изменения в названия ТК9 и ТК13 на “При очистке поля "Имя" кнопка "Сохранить" не активна” и “При очистке поля "Занятие" кнопка "Сохранить" не активна” соответственно. Это, на мой взгляд, повысит их понятийность.

6)Выпускать приложение Mesto в релиз явно не стоит. Так-как в нем помимо отловленных багов (зачастую незначительных) еще присутствует множество крупных недоработок. Не работают или работают неправильно следующие функции:

редактирование профиля (Б1, <https://docs.google.com/spreadsheets/d/1UEHMFqQi5IF2FBcTVU5AknX9nNtOA_nVtxoAzKBIoOw/edit?usp=sharing>);

добавление карточек (Б1, <https://docs.google.com/spreadsheets/d/1ijba1dwbCFwXikZUSnREOSJ3I6eDRf7Z8nZebkEg3M8/edit?usp=sharing>);

удаление карточек (Б2, <https://docs.google.com/spreadsheets/d/1ijba1dwbCFwXikZUSnREOSJ3I6eDRf7Z8nZebkEg3M8/edit?usp=sharing>);

лайки и их счетчик(Б5, <https://docs.google.com/spreadsheets/d/1QVUFV0DUmOTRObtCvW-3lTBUR6z4BpG9f-65tHTgkdU/edit?usp=sharing>).

Приложение еще не соответствует требованиям к нему заложенным (<https://praktikum.notion.site/Mesto-9f2cfaa209734d1f8cfa0c0db3d3049f>).

Mesto нуждается в доработке с последующим глубоким тестированием. Выпуск его в релиз сейчас - это неоправданные репутационные и финансовые потери. А главное - жирный намек пользователям, что нужно уходить к конкурентам.

1. **Вопросы на рассуждение:**

Пусть линейная будет **Waterfall**. Почему именно она, по ряду следующих причин:

понятная и простая структура процесса разработки (по спланированной документации);

удобная отчетность (по ресурсам, затраченному времени и финансам) благодаря строгой этапности процесса разработки;

однозначность и неизменность поставленных задач;

четко спланированный бюджет и сроки реализации.

Сначала будут определены все требования (как именно будет происходить поиск станции зарядки или поиск самоката по сигнализации) и этап проектирования. После их согласования наступит время художников-дизайнеров (как и где нарисовать иконку самоката или зарядки к нему).

Следующий этап - непосредственное написание кода приложения (работа штата программистов).

Далее приходит время тестировщиков. Так-как до этого момента приложение не проверялось, то придется задействовать как можно больше вариантов тестирования и времени.

Финальный этап - на радость самокатчиков и на горе пешеходов выпускать полностью готовое приложение в релиз.

Пусть гибкая модель разработки будет **Scrum**. Ее выбор обусловлен следующими факторами:

работа производится короткими этапами (спринтами), на каждом из которых определена определенная цель;

большие задачи разделяют на мелкие, поэтому внести корректировки прямо в процессе работы намного проще;

минимизация финансовых рисков благодаря оперативной реакции на изменения и устранение ошибок;

быстрота при выпуске готового продукта (релиз после каждого спринта).

Разработка будет разбита на два блока. В первый будем реализовывать функционал поиска ближайшей зарядки.

На втором решим вопрос с поиском по сигнализации. Заодно проведем мониторинг того, как пользователи встретили предыдущий выпуск приложения.

При этом каждый раз в релиз будет выпускаться обновленная версия продукта. Тестировщики будут работать одновременно с разработчиками (не как в Waterfall), что минимизирует появление багов и затрат времени на их исправление после очередного релиза приложения.

Виды тестирования, которые следует применить.

Прежде всего следует начать с дымового тестирования, на предмет выявления работоспособности основных функций и, соответственно, приложения. Если все в порядке, то переходим дальше.

Санитарного тестирования будет выполнено с целью проверить, что новые функции или исправления не сломали основную функциональность системы.

Было бы не лишним провести интеграционное тестирование (как приложение работает со сторонними объектами, например банками при оплате за пользование самокатом).

Если позволяет время и финансы можно провести негативное тестирование (искать ближайшую станцию зарядки в Ростове-на-Дону, хотя пользователь сейчас находится в Сызрани).

В конце перейдем к регрессионному тестированию, чтобы окончательно убедиться в полной работоспособности продукта. И если этот процесс не автоматизирован, то все падает на хрупкие плечи и трясущиеся руки начинающего тестировщика (джуна).

**Проект 2: ретест багов**

1. **Баги:**

https://docs.google.com/spreadsheets/d/1QVUFV0DUmOTRObtCvW-3lTBUR6z4BpG9f-65tHTgkdU/edit?usp=sharing

1. **Выводы о работе:**

1)По поводу баг-репортов такие замечания. Отсутствует раздел окружение (поэтому не ясно какой тип компьютера, какая ОС, в версии какого браузера произошла ошибка).

Также перемешаны поля (приоритет и статус, согласно требований, должны быть после ОР и ФР).

Некоторые баги можно было проиллюстрировать скриншотами (например, Б2 или Б7) или короткими скринкастами (например, Б13 или Б6).

Для многих багов были неверно указаны приоритеты (например, Б2 никак не может быть “критическим”, а Б7 вполне себе тривиален).

Для персонажей “Людей в черном” J и K Б3 вовсе не баг, а единственная возможность завести профиль на сайте.

2)Не все, осталось 5 багов, которые не исправлены или их нельзя проверить из-за других ошибок.

3)В процессе проверки (Б8 и Б9) найдены новые баги - отсутствует кнопка добавления фотографий и кнопка удаления фотографий. Ссылка на созданный БР: https://docs.google.com/spreadsheets/d/1ijba1dwbCFwXikZUSnREOSJ3I6eDRf7Z8nZebkEg3M8/edit?usp=sharing.

1. **Вопросы на рассуждение:**

1)Да, такое может произойти если ранее выявленный баг блокировал новую неисправность или закрывал часть функционала. Как только его исправили, новый баг вышел “на свободу” и ждет своей очереди на выявление. Для этого и проводится повторное тестирование.

А так, как говорил Карлсон (тот, который на крыше) “баги - дело житейское”. Они неотъемлемая часть IT-индустрии. Ошибки были, есть и будут. Разнообразные браузеры (Opera, Mozilla) с еще большим количеством версий, установлены на миллионы компьютеров (ноутбуки, desktop) от разных производителей, работающие под различными ОС (Windows, Linux). Аналогично и в мире мобильных устройств и приложений. Идеальный код написать не возможно, как и учесть всех факторов. Остается баги вовремя выявлять и исправлять по мере их критичности и влияния на работу системы.

2)Новый обнаруженный баг будет описан в БР и передан на исправление разработчикам. Далее уже дело за ними (отклонить, починить и т.д.).

3)В этом случае отправить баги в статусе переоткрыт или узнать у разработчиков, что эти ошибки не требуют сиюминутного вмешательства (будут исправлены позже, а может и не будут вовсе).

4)Никто не отменяет человеческий фактор. Разработчик мог исправить баг, но просто не внес его в обновление. Возможно, баг остался на стороне тестировщика и ему следуют проверить окружение (“очистить следы” в браузере и системе) или задействовать другую рабочую станцию.

P.S.: превращаюсь в ждуна (джуна), в ожидании повторного вердикта проекта. Спасибо!

