**Проект 1: регрессионное тестирование**

1. Проект в Qase:

[Тимофеев Сергей 31А\_qa-plus\_project\_venus - Спринт 3/21](https://docs.google.com/spreadsheets/d/1P6-DUmcmbRft2gDnTI4k01LrvzHqRwcTxIdX3vIsCUs/edit?usp=sharing)

1. Баг-репорты в YouTrack:

https://sergmtyprojecttester.youtrack.cloud/issues

1. Выводы о работе:

Было проведено регрессионное тестирование которое выявило 9 багов - 1 блокирующий (неотложная), 3 серьезных и 5 средних багов,  
на тестовом окружении: браузер Google chrome, Версия 129.0.6668.60 (Официальная сборка), (64 бит)Windows 11, version 23H2

на тестирование ушло достаточно много времени, несколько дней в часах приблизительно 10  
практически все баги не позволят использовать приложение в полном объеме хотя не все критичны. я бы не рекомендовал нести в прод данное приложение так так имеется блокирующий и несколько критических багов как только они будут устанены можно выпустить и фиксить баги с низким приоритетом в следующих релизах. Тест- кейсы сформулированы неплохо но я бы заполнил столбец предусловия где бы указал что нужно перейти на сайт и указал бы URL сайта, также указал бы роль пользователя неавтор, в комментариях указал бы устройство на на котором тестируют, версию ОС и ПО.

1. Вопросы на рассуждение:

*Релизный цикл состоит из следующих этапов:*

* планирование,
* создание требований и макетов,
* написание кода,
* тестирование,
* выпуск.

***ПЛАНИРОВАНИЕ***

При гибкой модели например **канбан** сначала обсудили бы на Daily митинге какие изменения нужно внести в выпускаемую версию продукта, какие задачи из бэклога имеют наиболее важный приоритет для пользователя  
Далее совместно с аналитиками и с разработкой обсудили бы все тонкости реализуемых фичей,требования к реализуемым функциональностям а также проблемы которые могут возникнуть в результате тестирования и запожили время на разработку и на тестирование   
После этого обсудили с QA Lead и командой тестирования план работ и распределили задачи между собой, кто что тестирует и соответственно если важнее реализация “поиск ближайшей станции зарядки”, то начать с нее и далее если остается время выполнить менее приоритетную задачу например Поиск самоката по сигнализации.  
**Создание требований и макетов.** На этом этапе менеджер общается с заказчиками и описывает как будет работать поиск станций, как они будут отображаться, в каком радиусе будет проводиться поиск и передает данные аналитикам которые составляют или корректируют требования и дизайнерам которые рисуют макет по полученной информации, если на это есть время   
**Разработка** как только требования готовы разработчики приступают к написанию кода а тестировщики, чтобы не терять время приступают к тестированию требований и уточнению моментов в работе приложения которые непонятны.Далее приступают к проектированию тестов(написание тест- кейсов или чек- листов)  
**Тестирование.** как только код готов приступают к smoke,sanity, ретестам багов, тестированию новых фич и в финале проводят регрессионное тестирование   
По результатам тестирования готовят отчёт. В нём старший тестировщик указывает, сколько ошибок удалось найти, и оценивает готовность к релизу. Если приложение не готово, даёт рекомендации: например, исправить блокирующие баги и провести повторный регресс.

**РЕЛИЗ И ПОДДЕРЖКА.** Далее происходит релиз и поддержка получает обратную связь от клиента и опять цикл разработки повторяется, выявляется наиболее приоритетная задача,корректируются требования, исправляются дефекты с прода, приходит обратная связь от клиентом и заказчиков и начинается новый цикл разработки.

При **водопадной модели** процесс от идеи до реализации занимает большой объем времени:сначала обсудить с фокус группой чего хотят клиенты потом проанализироать как будет работать приложение, после этого приступают к отрисовке макета, далее разработчик приступает к реализации идеи, после того как все готово передают в тестирование

Вначале нужно провести **smoke** тестирование. Ошибки, найденные в процессе смоук-тестирования, всегда блокирующие. Поэтому нужно убедиться что приложение выполняет основные функции, если находятся баги на данном этапе тестировать дальше не имеет смысла.  
Далее провести **testing** чтобы убедиться что самые важные компоненты такие как оплата аренды самоката и т.д. работают

Далее провести **модульное** или **функциональное тестирование** чтобы проверить что новые фичи работают корректно.  
и в конце провести регрессионное тестирование чтобы убедиться что новые фичи не сломали то что было реализовано до а также что вся система работает корректно.

**Проект 2: ретест багов**

1. Баги в YouTrack:

[Сергей Тимофеев, 31А\_qa-plus\_project\_venus-Спринт 3/21- ретест багов Место\Самокат - Hojas de cálculo de Google](https://docs.google.com/spreadsheets/d/1cC7hNrJGiFw3WJ1qBd9HlvSMeSld7wm9LPBik1zhu4c/edit?gid=379530441#gid=379530441)

1. Выводы о работе:

Было легко произвести проверку так как были описаны шаги для воспроизведения бага и также ожидаемый и фактический результат но не было указано окружение в котором были найдены баги например номер версии операционной системы, название и модель браузера а также версия приложения, также отсутствуют скриншоты и скринкасты которые помогают разработчику быстрее разобраться в чем дело, также в предусловиях я бы давал ссылку на приложения так как чтобы осуществить проверку любого из перечисленных тест-кейсов нужно сначала перейти на сайт <https://code.s3.yandex.net/qa/files/mesto/index.html> , а также указал роль пользователя,что касается ошибок, то не все ошибки были исправлены некоторые пришлось переоткрыть но новых багов мной обнаружено не было.

1. Вопросы на рассуждение:

Думаю что на месте исправленной может возникнуть другая ошибка из - за взаимосвязи кода: исправление бага на одном участке может привести к изменениям в работе программы на другом поэтому для того чтобы убедиться что этого не произошло в конце спринта проводят регрессионное тестирование, где проверяется работа смежных участков на работу которых могло повлиять изменение а также изменения в коде которые были сделаны после нахождения бага могут привести к тому что баг не воспроизводится хотя и остался в системе.Если в ретесте я обнаружу новый баг то сообщу разработчику, при проверке был обнаружен новый баг, также укажу версию приложения и окружения где был обнаружен баг и после того как будет исправлен новый быг проведу еще один ретест с проверкой старого и нового бага.Если разработчики не исправили несколько багов нужно написать под задачей что баги не исправлены а ткже указать номер версии приложения на которой осуществлялась проверка и окружение и тегнуть ПМ в задаче,и если баги не критичны то выпустить продукт с данными багами, если же критичны не закрывать задачу до полного исправления багов.  
разработчик сообщил, что исправил ошибки, но после повторной проверки выяснилось — все баги по-прежнему на месте такое может случиться что баг плавающий и был проверен разработчиком после исправления в другом тестовом окружении на его машине но при тестировании на контуре QA/ift могла всплыть ошибка а также при тестирвоании в другом браузере и etc.

\*Не забудь открыть доступ к этому документу. Нажми в верхнем правом углу «Настройки Доступа» → «Разрешить доступ всем, у кого есть ссылка». В окне «Разрешение» поставь роль «Комментатор».