Доброе утро уважаемая комиссия, преподаватели.

Меня зовут Еременко Алексей Николаевич.

Представляю вашему вниманию свой итоговый проект   
«Комплексное тестирование платформы QAhacking.guru»  
по программе повышения квалификации – тестировщик ПО ()  
 преподаватель Гриненко Виталий Владимирович  
 выполнил Еременко Алексей Николаевич   
 поток ТП 848  
по федеральному проекту содействия занятости.

Моя работа содержит 8 пунктов, первый это:  
 • майнд-карта STLC (жизненного цикла тестирования ПО;  
 • далее майнд-карта SDLC (методологии разработки ПО);  
 • следующий тестовая документация, которая включает в себя (ЧЛ, ТКы, БРы);  
 • я также покажу вам результаты применения техник тест дизайна;  
 • листинг автотеста;  
 • результат выполнения автотеста;  
 • Анализ результатов тестирования выбранного приложения;  
 • и выводы об оптимальности выбранной стратегии тестирования.

Итак, на первом слайде вы видите майнд-карту STLC, составленную мною в виде цикла.   
 Цикл начинается с этапа - **Общее планирование и анализ требований**далее идет **- Уточнение критериев приемки**;  
- **Уточнение стратегии тестирования;  
- Разработка тест кейсов;  
- Выполнение тест кейсов и Фиксация найденных дефектов;  
-** и завершается **Анализом результатов тестирования** и оформлением **Отчетности.**

На следующем слайде представлена майнд-карта SDLC,   
мною была выбрана каскадная методология поэтапной разработки ПО (),   
она включает в себя:  
• Анализ;  
• Проектирование;  
• Реализацию;  
• Внедрение;  
• и Эксплуатацию.

Данная методология - Используется при создании ПО, для которого можно достаточно точно и полно сформулировать все требования.  
Эту модель - Используют в государственных организациях (в оборонной отрасли), где требуется высокая отчётность с минимальными рисками возврата на предыдущий этап;   
 - а также для недорогих и несложных проектов.

**Переходим к рассмотрению тестовой документации**:

На этом слайде показан фрагмент Чек листа тестирования выбранного мною сайта.  
Он имеет основные **атрибуты**: ID, Заголовок, статус и Заметки:

- В Заголовке указывается краткое описание проверки,  
- в поле Статус может быть Pass или Fail,  
- поле Заметки предназначено для ссылок на БРы.

Далее представлены несколько ТКов из 34, написанных мною.  
Они имеют такие **атрибуты,** как:  
ID, Заголовок, Предварительные условия, Шаги, Тестовые данные, ОР, Статус , пЗссБРы:

- В Заголовке указывается, что мы будем проверять,  
 - в Предварительных условиях – подготовительные действия,  
- в Шагах – подробные и последовательные действия, которые будем выполнять,  
- далее прописываются Тестовые данные и ожидаемый результат,  
- поля Статус и заметки заполняются после прохождения тестов, (в них прописываются Пасс, Фэйл или Блокед, и ссылка на Багрепы, соответственно).

На следующем слайде представлены БРы, которые также содержат некоторые **атрибуты**:  
• ID, Тема, Подробное описание, Шаги, ФЗ, ОР, Вложения, Серьезность, Приоритет, Статус и Окружение:

- в Теме указывается краткое описание дефекта, которое отвечает на вопросы (что? где? когда?), что – невозможно полностью ввести номер телефона, где – в форме бронирования на странице «О нас», когда – при её заполнении.  
- в Подробном описании дополнительно ещё раскрывается суть проблемы, что поле ограничено вводом 10 знаков.  
- в Шагах – подробные и последовательные действия, которые мы выполнили,  
- далее показываем несовпадение ФР с ОР и прикрепляем скрин,  
- в Северити и Приорити, выставляем соответственно Критикал и Асап, т.е. Критическая важность и Наивысшая срочность, поскольку бронирование питомцев является основной функциональностью сайта Питомника.   
- статус дефекта новый,   
- в Окружении указаны устройства (компьютер и телефон), их ОС и браузеры в которых производилось тестирование.

Следующий пункт презентации - Применение техник тест-дизайна:  
На данном слайде приведен пример **результата** Применения техник тест-дизайна в ЧЛ.  
Когда однотипные проверки объединяются с помощью техники Классов Эквивалент-ти, что позволило сократить количество проверок на 4п при сохранении покрытия ПО.

На другом слайде приведен пример **результата** Применения техник тест-дизайна в ТКсах. Здесь мы видим однотипные ТКы с валидными тестовыми данными, которые также объединяются с помощью техники Классов Эквив-ти и их кол-во сокращ-ся.  
В итоге кол-во ТКсов сократилось с 34 до 18. Также исп-лись тГрЗн и ПрогнОшибок.

На следующих двух слайдах представлены фрагменты кода автотеста, написанного мною на языке программирования Пайтон и запущенного в VSC. При этом также использовались такие инструменты, как Вебдрайвер и Селениум ИДЕ.

На слайде **Результат выполнения автотеста** в окне терминала виден результат после прохождения автотеста:  
- данные введены,  
- процесс завершен.

Теперь хочу представить вам **Анализ результатов тестирования  
 выбранного мною приложения.**В ходе проверки веб-платформы QAhacking.guru были написаны 34 тест-кейса.  
В результате проверки было обнаружено 20 дефектов, блокирующие функционал дефекты отсутствуют.  
Всего, вместе с нефункциональным тестированием,   
было проведено 75 проверок.

Имеются 5 критических дефектов.

На текущий момент, веб-платформа не может быть отдана в релиз.  
Необходимы уточнения ТЗ и правки основного функционала.

Для увеличения покрытия проверками нефункциональной части,   
заказчику необходимо предоставить макеты дизайна веб-платформы.  
На данном слайде также представлена статистика по выявленным дефектам в виде диагр.

**Выводы об оптимальности выбранной стратегии тестирования**В ходе работы над проектом «Комплексное тестирование платформы QAhacking.guru» была применена оптимальная стратегия тестирования, в которой было применено достаточное количество тестов в функциональном тестировании, а также было проведено дополнительное тестовое покрытие с помощью чек листа в нефункциональном тестировании.  
Применение техник тест дизайна дало возможность добиться оптимальных результатов и максимизировать тестовое покрытие ПО.

В конце хотел бы дать Рефлексию по данному курсу:  
В процессе изучения программы я прослушал базовый курс теории тестирования ПО, ознакомился с инструментами, используемыми в тестировании, получил практические навыки работы, используя XMind, TestRail, TestIT, DoQA, Git, Jira, Postman, MySQL, Selenium IDE, PyCharm, VSC , Python, DevTools. Самостоятельно написал автотест на языке Python. Наиболее интересной для меня стала работа с базами данных в SQL, тестирование API и составление автотестов на языке Python в VSC.   
 Самым затруднительным в изучении теории тестирования стал тот факт, что «понятие типов, методов и видов в англоязычной литературе часто не разделяется и может быть перечислено вообще вперемешку, а также зачастую там выделяют только функциональное и нефункциональное тестирование».

С моими практическими работами вы можете ознакомиться по ссылке на GIT HUB: https://github.com/AlexDanish/proekttest

Благодарю за внимание!