**Итоговая аттестационная работа**

**Техническое задание и дефекты**

В условиях итоговой работы нет подробного технического задания, но в реальной работе я понимаю, что оно будет в любом случае. Нужно строго следовать ТЗ заказчика и проверять реализованные им требования на реальном продукте. Моей задачей как тестировщика является проверка продукта на соответствие ТЗ, туда входит:

* Выявление дефектов, описание их в доступной и понятной форме, фиксирование их в багтрекере и определение приоритета. При возможности указывать возможные причины дефекта.
* При необходимости вводить уточнения для определенных моментов в проекте так же в багтрекере. Чтобы было меньше конфликтов и не понимания между нами и заказчиком нужно все непонятные и противореящие моменты в ТЗ поднимать наружу на стадии разработки приложения.
* Так же если есть идей по различным улучшениям, которые не являются громоздкими в своей реализации, можно предлагать так же описывая их в багтрекере. Такие идеи должны быть реально обоснованы для реализации в проекте.

На все записи нужно будет указать приоритеты, Blocker, Critical, Major, Normal.

**Тестирование**

В итоговой работе я посчитал нужным использовать следующие техники тест-дизайна:

* Тестирование граничных значений
* Диаграмма состояний и переходов

***Примеры тестов с применением этих техник предоставлены в Exel таблице***

Использование техники ***decision table*** считаю неуместным так как по Флоу в задании понятно, что практически все условия будут жесткими, и положительных сценариев тестирования будет буквально 1 или 2.

Техника ***попарного тестирования*** на реальном проекте уместна, так как реальное приложение может работать на разных ОС (Андроид или IOS), разных моделях смартфонов, разных версиях ОС, так же нужно будет тестировать языки интерфейса (например русский, английский, французкий, немецкий и т.д.). Полный 100% охфат тетсами всех вожможных вариантов сборок очень трудоемко и практический невозможно, поэтому можно использовать технику попарного тестирования (можно использовать ресурсы такие как pairwise)

Зачастую не нужно ждать от разработчиков готового решения и потом только проверять его, тестировать можно начинать и на начальных этапах.

Unit тесты. Эти тесты обычно встроены сразу в код и их проведением занимаются разработчики, но всегда есть небольшой шанс что и тестировщикам нужно будет заниматься таким видом тестирования. Этот вид тестов не беру в расчет.

Интеграционное тестирование, API тесты. Основная работа тестировщика. Им можно заниматься и без UI интерфейса, тут нужно проверить работоспособность путей связи. В итоговой работе API тесты будут проверять интеграцию между мобильным приложением – локальной БД – БД УФНС. Оправку, прием SQL запросов в/из базы данных. Проверку функционала приложения, различные действия с расчетным счетом ИП, проверку цикла CRUD с расчетным счетом ИП, для все этого нужно интеграционное тестирование. Ресурс, в котором я мог бы осуществить такую задачу это Postman.

E2E тестирование. Примеры таких тестов есть в Чек-листе. Какие еще тесты можно провести:

* Проверки разного функционала приложения (регистрация, авторизация)
* Проверки кликабельности различных кнопок
* Проверку переходов в разные разделы приложения
* Проверки ***наличия масок*** на формах по заполнению ИНН, номера телефона, адреса жительства. (так же это нужно чтобы было меньше мусорных данных на сервере)
* Так же проверку CRUD расчетного счета уже через UI.
* И т.д.

.

***Тестирование мобильного приложения***

Теститование обновлений. По условиям работы у нас мобильное ПО, любое по должно иметь возможность обновляться. Это нужно будет проверить, обновляя приложение различными способами, через WiFi, SD, мобильный интернет, через провод на ПК.

Тестирование интернационализации и локализации. Необходимо узнать особенности стран предполагаемой аудитории, языки, культуры, традиции, законы, для предотвращения неприятных ситуаций с пользователями и законодательством страны.

Тестирование usability. Нужно будет выявить неудобные моменты в работе приложения, в UI, в энергоемкости и оптимизации. Работе на различны размерах экранов, на сенсорных и не сенсорных дисплеях.

Тестирование производительности (нагрузочное тестирование), сколько ресурсов телефона потребляет приложение, выявить узкие места в работе по.

Аттестационное тестирование. Проверить соответствия приложения стандартам, лицензионным соглашениям и условиям пользования (например, для работы на IOS)

Так же нужно будет провести тестирование пуш нотификаций и внешних прерываний (смс, ммс, звонки, зарядка, резкое отключение, авиарежим и т.д.)

Для тестирования мобильного приложения могу использовать не только именно сами телефоны, но и просто эмуляторы и симуляторы на ПК, в нем могут быть различные модели телефонов с разными ОС и их версиями.

***Тест кейсы.***

Указывать конкретные примеры тест кейсов считаю чрезмерным, просто подчеркиваю, что в отличии от чек-листа, тест кейс должен быть с четким описанием шагов которые нужно сделать. Например:

1. Запустить приложение.
2. Авторизоваться правильной парой логин/пароль
3. На стартовой странице нажать кнопку «открыть расчетный счет ИП»
4. В необходимых полях заполнить данные о регистрации и месте жительства, паспортные данные, номер ИНН, номер мобильного телефона.
5. Нажать кнопку «открыть счет»
6. Дождаться уведомления об успешности операции
7. Закрыть приложение