**Классификация тестирования по уровням**

**Unit тестирование (компонентное, модульное).** Им обычно занимается непосредственно разработчик программного кода. Ведь именно такой вид тестирования позволяет протестировать отдельные компоненты исходного кода программы. Если взять в пример интернет-магазин, то в рамках unit тестирования будут тестироваться такие функциональности, как перемещение товара в корзину, его поиск на сайте, оплата заказа.

**Интеграционное тестирование.** В рамках него происходит тестирование части системы, которая состоит из нескольких модулей. Если взять в пример всё тот же интернет-магазин, то при интеграционном тестировании могла бы, к примеру, происходить проверка того, как со страницы корзины происходит оплата посредством той или иной платёжной системы.

При этом существует *несколько видов интеграционного тестирования*:

* Тестирование интеграции компонентов – проверяется, как отдельные модули одного приложения взаимодействуют между собой;
* Системное интеграционное тестирование – проверяется, как взаимодействуют все компоненты системы/как между собой взаимодействуют разные системы/как работают интерфейсы при помощи которых функционируют системы

**Системное тестирование.** На этом этапе проверяется законченный и полностью интегрированный программный продукт. Если мы говорим об интернет-магазине, то в рамках данного тестирования будет проверяться, как между собой работают вообще все модули, разработанные для него – можем ли мы зарегистрироваться, найти товар, положить его в корзину, оплатить заказ, отслеживать товар в пути. То есть мы тестируем приложение целиком.

**Приемочное тестирование.** Проводится на этапе сдачи готового продукта, либо какой-то готовой части заказчику. Цель приемочного тестирования – определение готовности продукта. И достигается она путем прохода тестовых сценариев, а также «прогона» различных сценариев на основе спека и требований к продукту. Существует *несколько видов приемочного тестирования:*

* Пользовательское приемочное тестирование – проводится пользователями конечного продукта. Если мы понимаем, что наш продукт готов к релизу, мы можем собрать группу конечных пользователей, которая его протестирует и расскажет, соответствует ли продукт их ожиданиям. На основе данного заключения разработчики смогут исправить баги;
* Эксплуатационное тестирование – выполняется пользователем или администратором в среде, которая имитирует реальные условия. На данном этапе обычно проводится тестирование резервного копирования, аварийного восстановления системы и ее безопасность;
* Тестирование на соответствие контракту – представляет собой проверку на соответствие продукта различным нормативным актам;
* Альфа тестирование – эксплуатационное тестирование, которое проводится на стороне разработчика, однако не в рамках нашей компании;
* Бета-тестирование – проводится на внешней стороне без участия разработчиков небольшой группой пользователей

**Классификация тестирования по позитивности**

**Позитивное тестирование.** В рамках него применяются сценарии, которые соответствуют нормальному поведению системы. При помощи него тестировщик понимает, делает ли система то, для чего она создавалась. То есть, если мы тестируем калькулятор, то в рамках позитивного тестирования, мы будем проверять, что при сложении 2 и 2, получается 4.

**Негативное тестирование.** В рамках него тестировщик применяет сценарии, которые соответствуют внештатному поведению системы. Он, к примеру, может вводить неверные данные. Если вновь взять в пример калькулятор, то в рамках негативного тестирования тестировщик мог бы попробовать поделить на ноль – так как он заведомо знает, что сделать это нельзя и мы должны получить сообщение об ошибке.

Стоит отметить, что сначала рекомендуется проводить позитивное тестирование.

**Классификация тестирования по степени важности тестируемых функций**

**Smoke тестирование.** Проводится на начальном этапе. К примеру, после нового билда. Этот вид тестирования предназначен для того, чтобы проверить, готов ли продукт к выполнению дальнейшего тестирования. Обычно это короткий цикл тестов либо же вообще один тест, который проверяет, что приложение работает или выполняет свои основные функции. В случае, если ПО не проходит smoke тестирование, процесс тестирования полностью останавливается до момента исправления багов.

**Тест критического пути.** В рамках него тестировщик проверяет значимые функции приложения при стандартном использовании. На практике на данном уровне тестирования проверяется основная масса требований к продукту. Например, возможность набора текста, шрифт, возможность вставки изображения. Этот вид тестирования может быть как позитивным, так и негативным.

**Расширенный тест.** В рамках данного вида тестирования проводится нестандартное использование программного продукта. Например, тестировщик вводит спецсимволы в поля ввода, нелогично кликает по кнопкам, одновременно работает на большом количестве вкладок (максимально нагружает систему).

Классификация тестирования по цели тестирования

**Тестирование новой функциональности.** В рамках него проводится проверка качества какой-либо новой функциональности, отправленной на тестирование. Обычно тестирование проходит все этапы – smoke тест, тест критического пути, расширенный тест.

**Регрессионное тестирование.** Это тестирование ранее проверенной функциональности с целью удостовериться, что изменения в коде (например, добавление новой функциональности или исправление дефекта) не повлияли на ее работу. Данный вид тестирования в обязательном порядке проводится в каждом новом билде. Он может включать в себя проверку исправленных багов, их степень влияния на весь продукт либо же на определенную функциональность. Следует понимать, что регрессионное тестирование обычно покрывает не все приложение, а только те участки, которые хоть как-то соприкасаются с изменением кода. Регрессионное тестирование рекомендуется проводить несколько раз. Обязательно нужно включать в регрессию области, которые чаще всего меняются в ходе разработки.

**Re-test.** В рамках ретеста тестировщик убеждается в том, что разработчик действительно пофиксил тот или иной дефект.

Классификация тестирования по степени автоматизации

**Ручное тестирование.** В рамках него тестировщики выполняют тесты вручную и не используют средств автоматизации.

**Автоматизированное тестирование.** Оно предполагает использование специального программного обеспечения для контроля выполнения тестов и сравнения фактического и ожидаемого результатов работы программы.

Классификация тестирования по знанию кода

**Тестирование черного ящика.** Тестировщик не знает, как устроена тестируемая система «под капотом» – не знает код. Данная техника подразумевает работу только с внешними интерфейсами тестируемой системы.

**Тестирование белого ящика.** Тестировщику известны детали реализации тестируемой системы.

**Тестирование серого ящика.** Тестировщику известны лишь некоторые особенности реализации тестируемой системы.

Классификация тестирования по исполнению сценария

**Ad-hoc тестирование.** Представляет собой тестирование без использования каких-либо спецификаций, планов и разработанных тест-кейсов.

**Исследовательское тестирование.** В отличие от ad-hoc, оно представляет собой более формальную версию тестирования. Оно не требует написания тест-кейсов, но в то же время подразумевает, что каждый последующий тест выбирается на основании результатов предыдущего теста.

**Сценарное тестирование.** Классическое тестирование по предварительно написанным и уже задокументированным тестовым сценариям.

Классификация тестирования по запуску кода

**Статическое тестирование.** Программный код во время данного тестирования не выполняется. Оно начинается на ранних этапах жизненного цикла ПО. К нему можно отнести вычитку кода, инспекцию проектной документации, функциональных спецификаций и требований.

**Динамическое тестирование.** Данный тип тестирования предполагает запуск программного кода. Происходит анализ поведения программы во время ее работы.

Функциональное тестирование

Представляет собой проверку соответствия функциональных требований ПО его реальным характеристикам. Тестировщик проверяет, что система *делает.* Основная задача функционального тестирования – подтверждение того, что разрабатываемый продукт обладает всем функционалом, который требует заказчик. Данное тестирование проводится на всех уровнях.

Нефункциональное тестирование

В рамках него тестировщик проверяет, как система *работает.* Проводится тестирование свойств, которые не относятся к функциональности системы.

**Тестирование на отказ и восстановление.** Это исследование программных систем на предмет восстановления после ошибок и сбоев.

**Тестирование производительности.** В рамках негопроверяется работоспособность приложения, стабильность потребления ресурсов, качество работы приложения в условиях различных сценариев использования. *Тестирование производительности*, между тем, *делится на* *несколько видов* тестирования:

* Нагрузочное тестирование. Проверяется производительность при нормальных условиях;
* Стресс-тестирование. Проверяется работоспособность системы при экстремальных нагрузках;
* Тестирование стабильности. Представляет собой проверку приложения при длительной работе;
* Объемное тестирование. Представляет собой тестирование системы при увеличенных объемах обрабатываемых данных

**Тестирование удобства использования**. В рамках него проводится исследование удобства работы с продуктом с точки зрения пользователя. Тестировщик может, к примеру, проверить используемые графические элементы, цветовые оформления с точки зрения восприятия пользователем, а также удобство навигации.

**Тестирование безопасности.** В рамках него проверяется защищенность пользовательских данных. Тестировщик может, к примеру, протестировать насколько просто неавторизованный пользователь получит доступ к системе, либо же насколько просто постороннему лицу получить доступ к данным.

**Тестирование установки.** В рамках него проверяется успешность установки приложения, его настройки обновления и удаления.

**Конфигурационное тестирование.** В рамках него исследуется работоспособность программной системы в условиях различных программных конфигураций. К данному виду тестирования можно отнести кроссплатформенное и кроссбраузерное тестирование.

**Тестирование локализации.** В рамках него происходит тестирование локализованной версии программного продукта. Тестировщик проводит проверку содержимого приложения или сайта на соответствие лингвистическим, культурным требованиям, а также специфике конкретной страны или региона. Тестирование локализации также включает проверку форматов времени и валюты, графических элементов, иконок, фотографий, цветовых схем.

**Тестирование интернационализации.** В рамках него тестировщик проверяет, насколько продукт может в дальнейшем адаптироваться для той или иной локации. В приложении необходимо предусмотреть возможность поддержки некоторых изменяющихся элементов. Например, это может быть вертикальный текст азиатских стран, либо же чтение справа налево в арабских странах.