**Идентификация** - подтверждение того, что запись о таком пользователе существует в системе. **Проверка что такой пользователь с данным логином в принципе существует в нашей системе (проверка через БД)**

**Тестирование по требованием** = Позитивное тестирование.

**Аутентификация** - подтверждение подлинности пользователя, который сейчас логинится (аутентичности, уникальности). **Проверка по паролю (тот ли это пользователь или это мошенник например пытается зайти в нашу систему)**

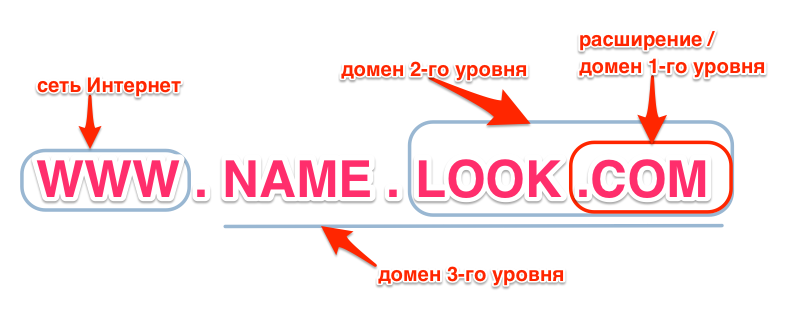
**Авторизация** - предоставление определённому лицу или группе лиц прав на выполнение определенных действий. **Какая роль у нашего пользователя в системе (какую информацию мы можем ему предоставлять а какую нет)**

**Порядок логина пользователя:**

1. **Идентификация**
2. **Аутентификация**
3. **Авторизация**

**Части E-mail:**

1. До @ = имя почтового ящика
2. После @ = доменное имя
3. **.com** = доменная зона



**Валидация на клиенте происходит через JS, HTML5, CSS**

**Social login** - Форма единого входа с использованием существующей информации из социальной сети, такой как Facebook , Twitter или Google

***Техники тест-дизайна.***

Тест-дизайн - процесс создания списков тест-кейсов в соответствии с требованиями проекта.

## **Техника анализа граничных значений**

Это техника проверки поведения продукта на крайних (граничных) значениях входных данных.

-1 0 +1

На каждой границе диапазона следует проверить по три значения:

1. граничное значение;
2. значение перед границей;
3. значение после границы.

**нужно протестировать поле Возраст,поле Номер телефона или длину Пароля, какой техникой тест дизайна ты воспользуешься?**

**2. Техника Попарного тестирования**

<https://habr.com/ru/company/otus/blog/592575/>

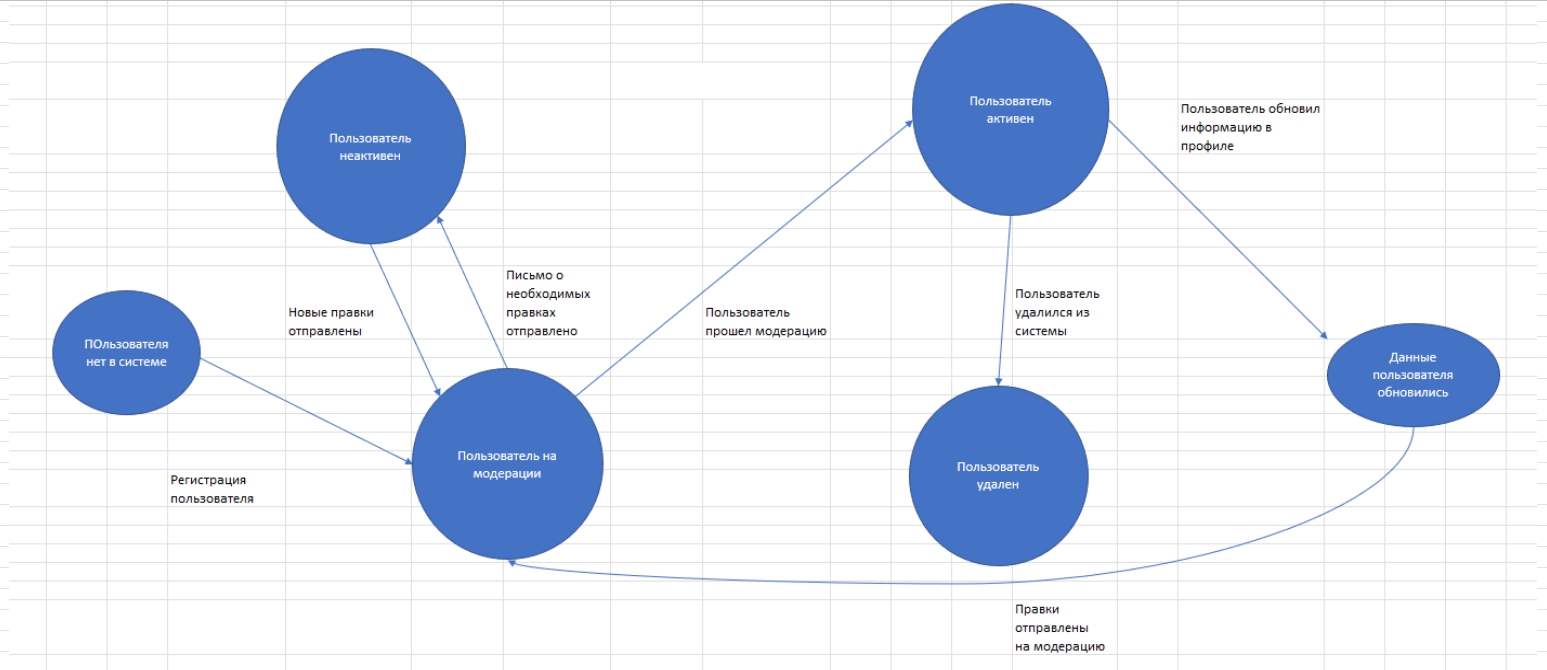
<https://pairwise.teremokgames.com/>

**3. Тестирование состояний и переходов**

**Состояние у нашего пользователя может быть несколько (например мы зарегистрировали пользователя, потом удалили = состояния изменились).**

**Переходы = то, с помощью чего происходит изменение 1 состояния на другое (например регистрация пользователя = переход, изменение данных профиля = переход).**

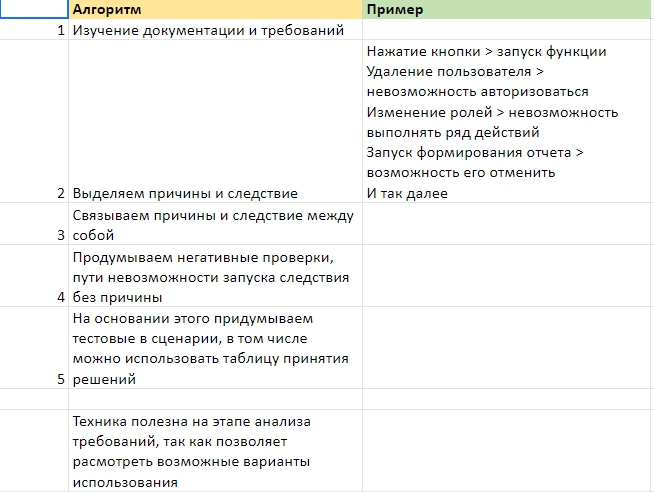
1. Переход может осуществлять как пользователь, так и система
2. Объект может находиться только в 1 состоянии одновременно
3. Объект должен быть 1
4. Данный вид тест-дизайна не про GUI (а про объект в БД)



**4. Причины и следствия**

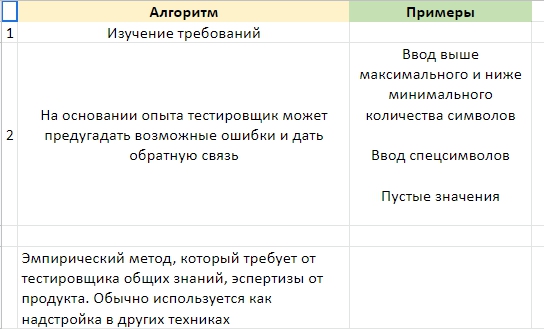
**Какое-то действие (нажатие на кнопку) определяет следствие (регистрация пользователя). Негативные проверки тоже!!!**

**Хорошо помогает на этапе анализа требований!!!**

****

**5. Предугадывание ошибок (error guesing)**

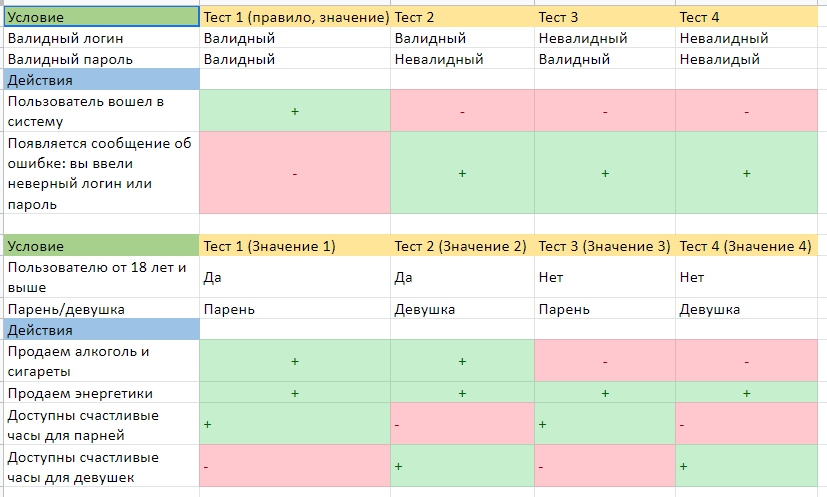
Когда у человека есть опыт с работы в каком-то проекте. Или долго работала с каким-то определенном проекте пришла на новый проект и там тот же модуль. Или когда ты знаком с разработчиком и знаешь где он бажит. Таким образом предугадываешь что примерно смотреть (места где часто баги и где разработчики часто косячат).

****

**6. Таблица принятия решений**

**(таблица решений) — способ компактного представления модели со сложной логикой; инструмент для упорядочения сложных бизнес требований, которые должны быть реализованы в продукте. Это взаимосвязь между множеством условий и действий.**

**В таблицах решений представлен набор условий, одновременное выполнение которых должно привести к определенному действию.**

****

**7. техника тест дизайна "Use Case"?**

**мы делим кейсы на "Роли" (Напр: админ/пользователь/незарег.пользов)?**

1. ***Поля ввода*** = Техники “Класс эквивалентности” и “Граничные значения”
2. ***Много веб-форм (фильтрация, настройки, сортировка***) = “Попарное тестирование” + в какой-то степени “Граничные значения” и “Классы эквивалентности” тоже.
3. ***Пару условий имеется*** = “Таблица принятия решений” + “Классы эквивалентности”