Сначала нужно разобраться в типах валидации на странице:

На стороне клиента – до момента отправки запроса на сервер, при разработке форнт энда, на стороне клиента можем изменять свою верстку, регулировать количество символов, менять валидационные сообщения

На стороне сервера – есть своя валидация, которая проверяет может ли быть такая информация обработана сервером, если возникает ошибка, то сервер отсылает некую свою ошибку, например статус код.

Мы всегда должны проводить валидацию на стороне клиента и сервера.

Крайне важно проводить валилацию на стороне клиента, например, при вводе пароля – сразу знать подходит ли по сложности

Пару слов о двух техниках тест-дизайна:

Эквивалентное разбиение:

Входные данные, которые обрабатываются нашим приложением одинаково.

Анализ граничных значений – техника тест-дизайна, которая направлена на проверку поведения системы на граничных значениях входных данных (границах классов эквивалентности).

> Например, для диапазона вводимых значений от 1 до 1000 все значения от 1 до 1000 приведут к одному и тому же результату, то границы две: нижняя и верхняя.
>
> Первое граничное значение – 1. Второе граничное значение – 1000. Добавляем к ним, стоящие рядом значения:
>
> 1. 0, 1, 2
> 2. 999, 1000, 1001

Алгоритм использования техники граничных значений следующий:

1. Определить классы эквивалентности.
2. Определить граничные значения для каждого класса (важно понимать к какому классу относится значение).
3. Провести тесты по проверке значения до границы, на границе и сразу после границы.

Открыть форму и проверить есть ли валидация на стороне клиента:

F12 в браузере и выбрать нужный элемент. Просмотреть какие свойства указаны у элемента.

http://itcareer.pythonanywhere.com/

https://www.w3schools.com/html/html_forms.asp - что такое html формы

Текстовое поле / Text field

First name, last name, password. В любом поле, где есть возможность что-то напечатать.

First name:
John
Last name:
Doe

Submit

Functional:

- Обязательность ввода
  (проверить, что произойдет если не ввести данные в поле)
- Обработка только пробелов
- Пустая строка после редактирования
- Использование пробелов в тексте
  (перед тексом, после него, и внутри; пробелы в начале и в конце строки должны отсекаться при сохранении)
- Минимально/максимально допустимое количество символов
  (использование техники граничных значений, можно посмотреть сколько символов максимально возможно ввести на клиенте и сколько принимает сервер, сравнить их, параметр maxlenght)
- Формат данных
  (исходя из его логического назначения и требований приложения)
- Ввод тегов и скриптов
  (Тэги: <> используют в том числе в различных языках, например, JS, злоумышленник может попытаться ввести скрипт, который будет получить вашу информацию при вводе в поле. Проблемы с безопасностью.
  Пример тег bold: не станет ли система выводить текст жирным шрифтом – указывает на, что система не воспринимает их как просто текст
  проверка должна осуществляться только в пользовательской части. Введенные теги должны отобразиться в том же виде, в котором они были введены
  SQL, XSS, HTML – инъекции, HTML инъекции, корректный SQL запрос)

- Использование специальных символов
  (введенные символы должны отобразиться в том же виде, в котором они были введены, если только ввод спец. символов не запрещен требованиями приложения
  “[|]’~<!--@/$%^&#*/()?>,.*/*
  "Плохие символы" (♣☺♂))

- Возможность редактирования введенных значений
- Уникальные данные
  (например, уникальность логина)
- Автоматическая постановка курсора в первое поле для ввода при открытии формы

- Возможность ввода данных из буфера обмена
  (ctrl+c/ctrl+v, нельзя вводить подтверждение пароля через ctrl+v, иногда через ctrl+v можно ввести больше символов чем разрешено.
  Дополнение. Изредка встречается копирование пользователем из буфера обмена в поле ввода нескольких строк. Для этого случая применяем правило:
  Информация после первого встреченного символа конца строки должна обрезаться)

- Введенные и отображаемые впоследствии данные должны быть одинаковыми.
- Placeholder – хорошая практика. Placeholder должен исчезать
  (Placeholder - подсказка, расположенная внутри поля)
- Каким запросом отправляется текст. Доходит ли текст, не приходит ли пустое поле
  (важно для секьюрности)

GUI – graphic user interface:

- Название поля
  (спеллинг, соответствие с открытым модулем или страницей)
- Отмечены ли обязательные поля
  (пример: Поля Фамилия и Имя указаны как обязательные к заполнению на уровне БД, но этой обработки нет на клиенте. При попытке передачи данных без имени приложение падает)
- Выравнивание названий полей
  (выравнивание по левому краю или правому краю (в зависимости от требований приложения, отступы, идентичность расстояний между названием и полем)
- Корректное расположение текста внутри текста, длинный текст не выходит за границы поля при вводе
- Унификация дизайна
  (цвет, шрифт, размер (высота/ширина), выравнивание полей)
- Расположение вводимого текста внутри поля
  (унификация, выравнивание по нижнему краю, если иное не определено специфичными требованиями приложения)

Текстовая область / Text Area

Дополнительно к текстовому полю, обычно по сравнению с текстовым полем большее количество символов, часто с возможностью переноса строки

About me, comments

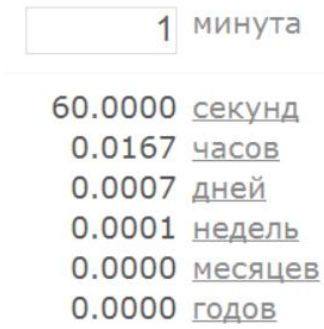
Functional:

- Корректное распределение текста по строкам
  (переход на новую строку автоматически)
- Предусмотрено ли использование тегов и скриптов
  (пример с жирным шрифтом)

GUI:

- Может ли область растягиваться. Как при этом ведет себя текст
  (а потом посмотреть, что производет при ресайзе окна, как среагирует форма на изменение размера окна)

Числовое поле



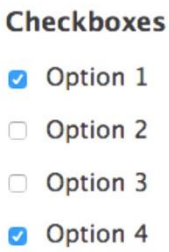
Functional:

- Есть ли граничные значения (например, допустимый возраст)
- Формат данных, формат числовых данных
  (если допускаются: негативные, дробные с точкой и запятой, с точкой и запятой 123.123.123,00, пробелы)
- Как реагирует на +, -
- Степени двойки 32768 (2^15), степени двойки плюс 1
  (*32769 (2^15 + 1)*
  65536 (2^16)
  65537 (2^16 + 1)
  2147483648 (2^31)
  2147483649 (2^31 + 1)
  4294967296 (2^32)
  4294967297 (2^32 + 1))
- Научная запись чисел (1E-16)
- Вычисляемые выражения (2+3)
- В шестнадцатеричной системе счисления (F0) принимает ли клиент, производит ли вычисления сервер
- Ввести Infinity, Nan, Nule

Чек боксы / Check boxes

Квадраты, которые мы можем отмечать для выбора необходимых опций

Могут быть со списками. Главный элемент отключает подэлементы



Functional:

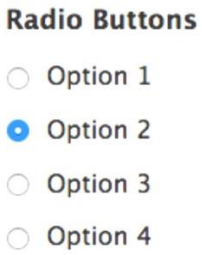
- Функционируют ли все боксы
  (включение/выключение)
- Обязательность выбора хотя бы одного чек бокса
  (проверить как ведет себя система, если мы ничего не выбираем)
- Проверить, как ведет себя система при выборе 1, нескольких и всех чек боксов
- Наличие дополнительного чек бокса, выставляющего/снимающего все чек боксы при наличии больше 10 чек боксов
  (Check all – можно занести в систему как импрувмент)
- При переходе на следующую страницу и возвращении назад выбранная радио кнопка не должна сбрасываться
- Является ли область рядом с квадратиком кликабельной и должна ли
  (текст, пространство рядом с квадратиком)

GUI:

- Унификация дизайна для всего приложения
- Выравнивание расположения чек бокса с соответствующим названием
- Корректность отображения чек бокса, который не возможно выбрать
  (к примеру, опция задизэйблена в данном случае)

Радио баттоны / Radio buttons

Можем выбрать только один элемент. НЕ может быть варианта, что не выбран ни один элемент. Всегда выбран 1 элемент



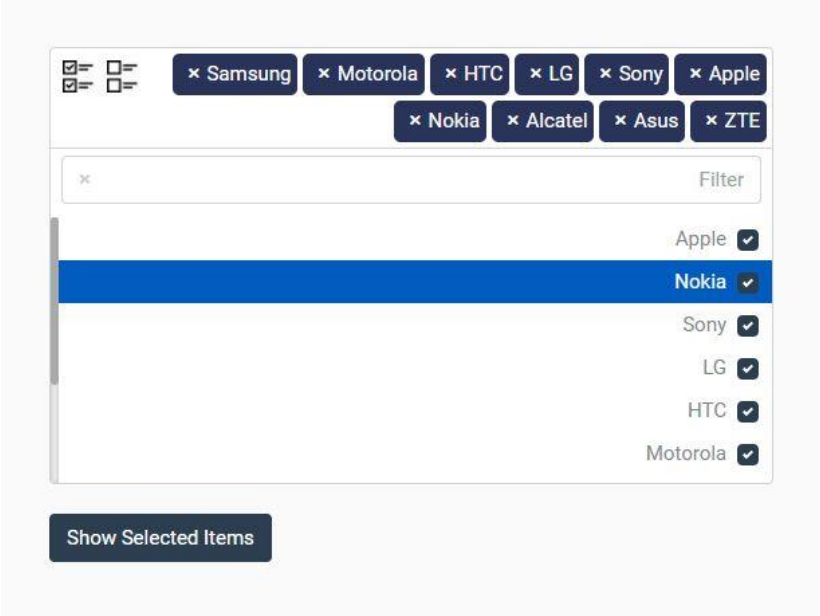
Functional:

- Функциональность
  (включение/выключение)
- Не может быть меньше 2 радиокнопок
- По умолчанию одна радиокнопка должна быть включена
- Не может быть включено более 1 радиокнопки
- При переходе на следующую страницу и возвращении назад выбранная радио кнопка не должна сбрасываться
- Является ли область рядом с квадратиком кликабельной и должна ли
  (текст, пространство рядом с квадратиком)

GUI:

- Унификация дизайна для всего приложения
- Выравнивание расположения радиобаттона с соответствующим названием
- Выравнивание расположений радио баттонов (по краю)

Поля со списком / Multipleselector



обращаем внимание на требования, requirments

Functional:

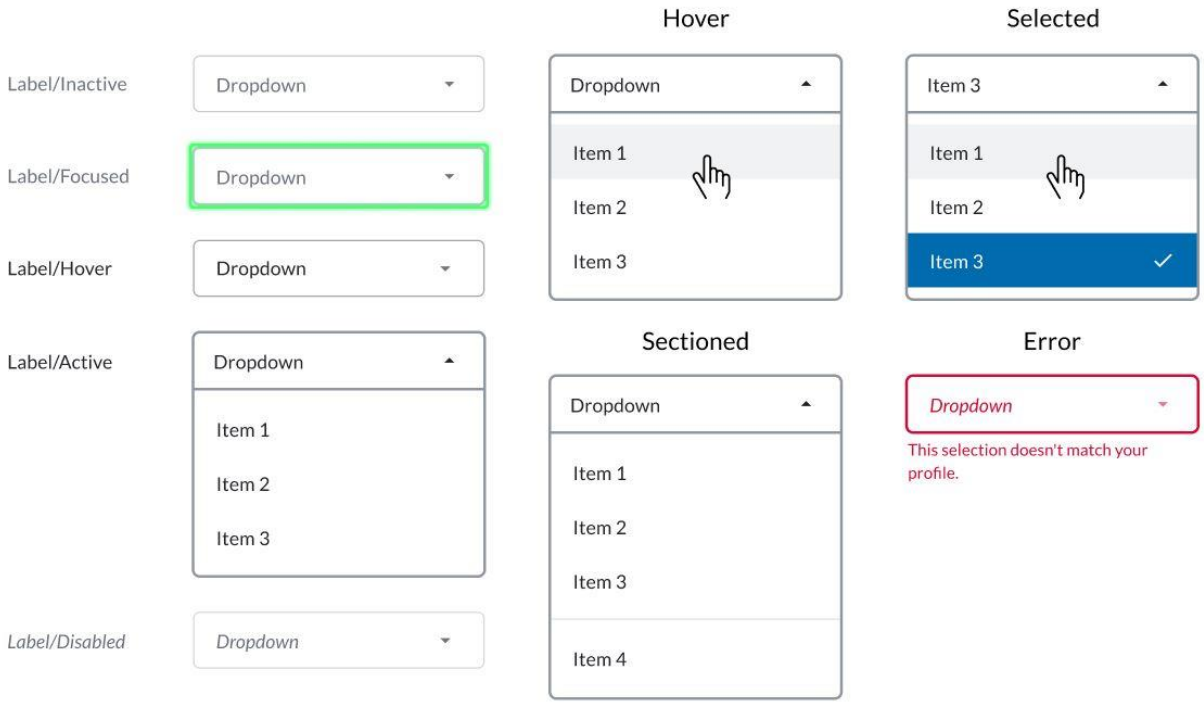
- Сортировка по алфавиту или по смыслу, фильтрация
  (если предусмотрена сортировка, проверить ее верность в списке выбранного, списке предложенных вариантов)
- В случае, если значения выходят за границы списка, и нет возможности увеличения размера списка, то необходимо отображение хинтов (всплывающих подсказок)
- Выбор пункта списка по нажатии соответствующей первой буквы на клавиатуре
- Возможность выбора нескольких значений для поля со списком
  (проверяем что можем выбрать ничего, один, несколько, все параметры)
- Возможность введения значений вручную,
  (если это позволяет приложение)
- Возможность «дрэг энд дроп»
  (перетаскивание в список выбранных элементов)
- Возможность добавления/удаления с помощью плюса, checkbox-а и т.п.
- Если предусмотрено: добавить, удалить все элементы
- Если есть поиск по элементам, проверить отрабатывает ли он
- Протестировать скролл, если есть

GUI:

- Спеллинг значений
  (название поля)
- Подсветка при выборе каждого из значений, при выборе нескольких значений одновременно
- Унификация дизайна (цвет, шрифт, размер (высота/ширина), цвет подсветки, выравнивание). Проверка осуществляется как для поля, как элемемета, так и для значений, и их названий

Выпадающее меню / Dropdown menu

позволяет выбрать одно из нескольких значений параметра



Functional:

- Сортировка по алфавиту или по смыслу
  (если предусмотрена сортировка, проверить ее верность в списке выбранного, списке предложенных вариантов)
- В случае, если значения выходят за границы списка, и нет возможности увеличения размера списка, то необходимо отображение хинтов (всплывающих подсказок)
- Выбор пункта списка по нажатии соответствующей первой буквы на клавиатуре
- Если есть поиск по элементам, проверить отрабатывает ли он (максимальное/минимальное количество символов, цифры, буквы кириллица, все что в требованиях)
- Что будет если выбрать опцию [None], если такая предусмотрена
- Проверить наличие варианта по умолчанию
  (оказывается люди очень часто выбирают именно его)
- Протестировать скролл, если есть

GUI:

- Подсветка при выборе каждого из значений
- Унификация дизайна
  (цвет, шрифт, размер (высота/ширина), цвет подсветки, выравнивание).

Поле для загрузки файлов / Download

Attachment

Functional:

- Обязательность выбора файла
  (не выбрать никакой файл, проверить показано ли валидационное сообщение)
- Форматы
- Ограничения на размер
  (проверка граничных значений 0мб, +0,1мб, -0,1мб, саму границу)
- Проверяем формат файлов. Сначала позитивные проверки: загрузка исполняемых файлов (EXE, PHP, JSP etc.). Негативные проверки. *Переименованный EXE
- Если есть возможность ввести путь к файлу, вводим проверяем как обрабатывает. Применяем правила из тестирования текстового поля.
- Срабатывает ли перетаскивание
- При отсутствии изображений должен быть соответствующий thumbnail, либо картинка совсем не должна отображаться
  (thumbnail – миниатюрное изображение на месте, где должна быть картинка, если не подгружена основная картинка)
- Контроль за размером (высота/ширина), должен ли быть ресайз
- Длинное имя файла (> 255 знаков)
- Загрузить уже существующий файл
- Доступность хранилища куда грузим файл

GUI:

- Унификация дизайна
  (цвет, шрифт, размер (высота/ширина), выравнивание)
- Выравнивание названий загруженных файлов, самих thumbnails файлов

Поле для записи номера телефона / Phone

Functional:

- Обязательность ввода
- Оставить поле пустым
- Использование пробелов в номере телефона
- Допустимое количество символов
- Ввод букв
  (кириллицы, латиницы)
- Использование знаков + и -. Использование специальных символов
  (считывает ли +, -)
- Маски – первые символы – код, можно видоизменить маску: вместо 8029 написать +375
- Проверить шаблон
  (пример: вводим 8 символов номера телефона, первые 3 символа – код - выделяются скобками, далее предусмотрено ли еще разбиение с тире)

- Обработка только пробелов
- Возможность редактирования введенных значений
- Уникальные данные
  (например, уникальность логина)
- Автоматическая постановка курсора в первое поле для ввода при открытии формы
- Ввод данных из буфера обмена
- Введенные и отображаемые впоследствии данные должны быть одинаковыми.
- Placeholder – хорошая практика. Placeholder должен исчезать
  (Placeholder - подсказка, расположенная внутри поля, информация о том как должен вводиться номер телефона)
- Ввод тегов и скриптов

GUI:

- Унификация дизайна
  (цвет, шрифт, размер (высота/ширина), выравнивание)
- Корректное расположение номера телефона внутри поля

Поле для записи email / Email

Functional:

- Ввод кириллицы – email только латиницей
- Ввод символа @. Как ведет себя система, если @ пропускаем, или вводим две @@.
- Использование цифр, спецсимволов, пробелов. Проверить разные типы email-ов на валидность

Часть адреса до символа @ (логин) может включать любые из этих символов:
Английские буквы верхнего и нижнего регистра (a–z, A–Z)
Цифры от 0 до 9
Символы ! # $ % & ' * + - / = ? ^ _ ` { | } ~
Символ . (точка) при условии, что он не первый и не последний, а также, если он не повторяется больше одного раза подряд (например, John..Doe@example.com)
Общая длина логина может быть вплоть до 64 символов.

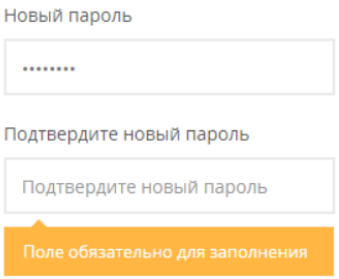
Часть адреса после символа @ (домен) может включать любые из этих символов:
Английские буквы нижнего регистра (a–z)
Цифры от 0 до 9
Символ - (тире)
Символ . (точка)
Части домена, разделенные точкой не должны начинаться с цифры или тире и не должны заканчиватся тире, а также они должны иметь длину от 1 до 63 символов.
Общая длина домена должна иметь максимум 253 символа. Доменная часть может быть IP-адресом, заключенным в квадратные скобки (например, jsmith@[192.168.2.1])

- Обязательность ввода
- Оставить поле пустым
- Допустимое количество символов
- Обработка только пробелов
- Возможность редактирования введенных значений
- Уникальные данные
  (например, уникальность логина)
- Автоматическая постановка курсора в первое поле для ввода при открытии формы
- Ввод данных из буфера обмена
- Введенные и отображаемые впоследствии данные должны быть одинаковыми.
- Placeholder – хорошая практика. Placeholder должен исчезать
  (Placeholder - подсказка, расположенная внутри поля, информация о том как должен вводиться номер телефона)
- Ввод тегов и скриптов

GUI:

- Унификация дизайна
  (цвет, шрифт, размер (высота/ширина), выравнивание)
- Корректное расположение email внутри поля

Поле пароля / Password



Functional:

- Проверка валидаций связанных со сложностью пароля. Ввести пароли, которые будут содержать все необходимые элементы; пароли, которые не соответствуют требованиям
  (пример: латинские символы, минимум 1 прописная буква, 1 цифра, 1 спецсимвол)
- Возможность ввода данных из буфера обмена
  (ctrl+c/ctrl+v, нельзя вводить подтверждение пароля через ctrl+v)

- Обработка только пробелов
- Использование пробелов в тексте
- Минимально/максимально допустимое количество символов
- Формат данных
- Ввод тегов и скриптов
- Использование специальных символов
  “[|]’~<!--@/$%^&#*/()?>,.*/*
- Возможность редактирования введенных значений
- Автоматическая постановка курсора в первое поле для ввода при открытии формы
- Введенные и отображаемые впоследствии данные должны быть одинаковыми
- Placeholder – хорошая практика. Placeholder должен исчезать

GUI:

- Унификация дизайна
  (цвет, шрифт, размер (высота/ширина), выравнивание)
- Подсказки по требованиям к паролю

Календарь / Date picker

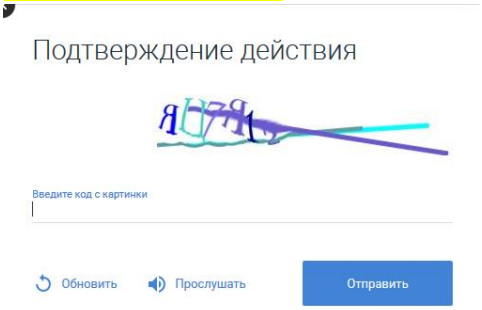
Functional:

- Проверить как ведет себя система при выборе даты из календаря
- При возможности ввода даты вручную необходимо проверить разные форматы. Ввод букв, цифр, спецсимволов.
  (Июнь 5, 2001; 60/05/2001; 06/05/01; 05-05-01; 6/5/2001 12:34)
- Проверки логичности ввода
  (даты в будущем, и т.д в требованиях должно быть прописано какую максимальную дату мы можем выбрать)
- Есть ли возможность изменять заблокированную дату
  (к примеру, услуга оказана, записать ее на другую дату нельзя)
- Ограничения по возрасту
  (самый ранний и самый поздний срок дня рождения)
- Проверка високосного года
- Некорректная дата (31 февраля, 29 февраля в не високосном году)
- Срабатывание всех кнопок в календаре

GUI:

- Унификация дизайна для всего приложения (цвет, шрифт, размер (высота/ширина), выравнивание)
- Отображение календаря рядом с полем
- Корректное выравнивание всех элементов и ссылок в календаре

Капча / Captcha



Functional:

- Действительно ли срабатывает капча при полном соответствии картинке
- При перезагрузке картинки капча изменяется и срабатывает
- Если есть звуки, они воспроизводятся и соответствуют картинке, срабатывают
- Ничего не вводить

GUI:

- Унификация стилей (в соответствие с дизайном сайта)

Поле с ссылками / Links

Share (прикрепить ссылку)

Functional:

- Тестирование префиксов при ссылках
  (http// и т.п. что проверяем: оставляем поле пустым, вводим различные виды протоколов ftp, https//, система должна не найти сайт т.к. применен другой протокол)
- Ввести www, без него
- Осуществить проверки Functional раздела для текстового поля
  (максимум минимум символов, русские символы и т.д.)
- Проверить предусмотрены ли какие-то валидационные сообщения
  (Появляется ли ошибка в случае некорректных данных, валидационные сообщение – со стороны клиента, или статус код от сервера)

GUI:

- Название поля, подсказка по вводу
- Выравнивание названий полей
- Унификация дизайна

Кнопка / button

Functional:

- Отсутствие вызова одного и того же действия повторно при нажатии на кнопку несколько раз
- Недоступные кнопки не скрыты, а заблокированы (как в документации?)
- Нажатие на пространство между близко расположенными кнопками не должно приводить к действию

GUI:

- Название кнопки
  (спеллинг, соответствие с действием)
- Эффект ‘нажатия’
  (вид кнопки должен изменяться при нажатии, если это не противоречит возможностям браузера)
- Название хинтов
  (соответствие с названием кнопки, спеллинг)
- Унификация дизайна (цвет, шрифт, размер (высота/ширина), цвет подсветки, выравнивание) . Проверка осуществляется как для кнопки, как элемента, так и для названия кнопки

Скроллинг / Scroll

Functional:

- Отсутствие скролла в случае, если текст вмещается на странице без прокрутки
- Соответствующее изменения текста при использовании скролла
- Возможность изменения положения скролла при помощи мыши, кнопок Page up/down, Home/End

GUI:

- Унификация видов и типов скроллов на всех страницах (если есть кастомный скролл, он должен быть применен на всех идентичных формах)

Проверить пишет ли в Json то, что мы отправляем на сервер

Каким запросом отправляется на сервер информация (секьюрность)

Выдается ли сообщение об ошибке, если приходит «не то» в БД

http://kirdenoff.blogspot.com/2013/02/blog-post.html