Одно из заданий, которое давал джуниорам:  
  
Нужно сделать простой сервис проведения тестирования по каким-либо темам. Т.е. есть тесты с вариантами ответов, один или несколько вариантов должны быть правильными. Тесты группируются в наборы тестов, которые затем пользователь может проходить и видеть свой результат.  
Функциональные части сервиса:

* Регистрация пользователей
* Аутентификация пользователей
* Зарегистрированные пользователи могут
  + Проходить любой из тестовых наборов
    - Последовательный ответ на все вопросы, каждый вопрос должен выводится на новой странице с отправкой формы (перескакивать через тесты или оставлять неотмеченными нельзя)
    - После завершения тестирования смотреть результат:
      * количество правильных/неправильных ответов
      * процент правильных ответов
* Админка. Стандартная админка Django. Разделы:
  + Стандартный раздел пользователей
  + Раздел с наборами тестов
    - Возможность на странице набора тестов добавлять вопросы/ответы к вопросам/отмечать правильные ответы
    - Валидация на то, что должен быть хотябы 1 правильный вариант
    - Валидация на то, что все варианты не могут быть правильными
    - Удаление вопросов/вариантов ответов/изменение правильных решений при редактировании тестового набора

Требования

* Код в репозитории на GitHub.
* Список всех зависимостей должен храниться в requirements.txt, соответственно можно установить их командой pip install -r requirements.txt.
* Разработка должны вестись в virtualenv, но сама директория с virtualenv должна быть добавлена в .gitignore.
* Настройки должны храниться в settings.py, но также, при наличии settings\_local.py в той же директории, настройки из settings\_local.py должны переопределять настройки в settings.py. Т.е. если есть файл settings\_local.py, то определенные в нем параметры имеют больший приоритет. Сам файл settings\_local.py добавляется в .gitignore. Таким образом у каждого девелопера и на бета сервере можно использовать кастомные настройки, например для соединения с БД.
* ~~Должен работать один из способо создания структуры БД. Встроенный manage.py syncdb или миграции через~~[**~~South~~**](http://south.aeracode.org/)~~(будет плюсом).~~
* По фронт-енду требований никаких не предъявляется. Интерфейс на твое усмотрение и он не буде оцениваться. Можно использовать любимый фреймворк или, например, воспользоваться [**Twitter Bootstrap**](http://getbootstrap.com/).