# Домашнее задание №2

## Задание

Создать проект для автоматизации тестирования модели сервиса, создающей отчеты о работе QA-отдела.

Реализовать 2 тестовых сценария:

**Тест 1**

1. Залогиниться под администратором
2. Создать нового пользователя
3. Выйти из сессии
4. Зайти под новым пользователем
5. Результатом авторизации должен быть код 200

**Тест 2**

**Входное условие:** В базе должен быть создан пользователь - не администратор (при этом не нужно создавать зависимость от предыдущего теста)

1. Через REST API создать тестовые данные:
   * номера требований с уровнями риска
   * набор тест-кейсов с привязкой к требованиям
   * данные плана регрессии
   * результаты «выполнения» тестов

(Пример входных данных приложен к исходному коду, файл: sample\_data/test\_2\_testdata.py)

1. Загрузить отчет по созданному плану регрессии в формате PDF
2. Проверить, что отчет совпадает с образцом  
   (К коду в качестве образца приложен файл sample\_data/test\_2\_sample.pdf. Если вы будете сравнивать файлы как изображения, как описано ниже в советах, конвертировать документ в изображение нужно самостоятельно, т.к. в разных системах шрифт будет отрисовываться по-разному.)

**Дополнительные условия:**

* Тесты должны быть подготовлены для запуска с помощью одного из общеизвестных фреймворков (pytest или другого по желанию)
* Тестовые функции должны содержать только простые шаги, которые теоретически мог бы понять не-программист. Все подробности взаимодействия с сетью, консолью и т.д. должны быть вынесены в отдельные модули.

## Описание приложения

### Настройка и запуск тестируемого приложения

1. Скачать материалы к курсу из репозитория:  
   <https://bitbucket.org/Auriga_OlegLeontyev/pythonautomationcourse2017.git>  
   или:  
   <https://bitbucket.org/Auriga_OlegLeontyev/pythonautomationcourse2017/downloads/>  
   Приложение находится в папке: PythonAutomationCourse2017/src/DZ2\_AppForTest.  
   Данные и образцы для 2-го теста находятся в подпапке sample\_data.
2. Для работы понадобится установить пакеты, указанные в requirements.txt
3. В testteamreports\settings.py поменять необходимые параметры (например, порт, если он используется приложением). Там же находится пароль администратора.
4. Инициализировать базу данных, запустив create\_db.py. Этот же скрипт можно использовать для пересоздания чистой базы.
5. Запустить сервис: run\_service.py.  
   Остановить можно с помощью Ctrl+C (если процесс запущен в консоли) или просто кнопкой "стоп", если запускаете в IDE.

### Доступные REST-запросы

Аутентификация в сессии:

POST https://<host>:<port>/login

Тело запроса: JSON {'login':str, 'password': str}

Закрытие сессии:

POST https://<host>:<port>/logout

Список пользователей (только администратор):

GET https://<host>:<port>/admin/users

Создание пользователя (только администратор):

POST https://<host>:<port>/admin/user

Тело запроса: JSON {'new\_login':str, 'new\_password': str}

Тело ответа: JSON {'id': int} - id нового пользователя

Обнуление базы (только администратор):

POST https://<host>:<port>/admin/recreatedb

Список требований:

GET https://<host>:<port>/reqs  
Тело ответа: JSON-массив с полями всех требований

Добавление требования:

POST https://<host>:<port>/req

Тело запроса: JSON {'name':str, 'risk': int}

Тело ответа: JSON {'id': int} - id новой записи

Список тест-кейсов:

GET https://<host>:<port>/testcases  
Тело ответа: JSON-массив с полями всех кейсов и привязанными требованиями

Добавление тест-кейса:

POST https://<host>:<port>/testcase

Тело запроса: JSON {'name':str, 'requirements': [... список имён требований ...]}  
Тело ответа: JSON {'id': int} - id новой записи

Список планов прогона тестов:

GET https://<host>:<port>/testruns  
Тело ответа: JSON-массив с полями всех наборов и привязанными тест-кейсами

Добавление плана прогона:

POST https://<host>:<port>/testrun

Тело запроса: JSON {'start:date':str,  
'end:date':str,  
'description':str,  
'test\_cases': [... список имён тест-кейсов ...]}

start\_date и end\_date - дата в формате ISO, например "2017-12-31"

Тело ответа: JSON {'id': int} - id новой записи

Добавление результата:

POST https://<host>:<port>/testrun/<testrun\_id>/result

Тело запроса: JSON {'name':str, 'is\_passed': bool, 'date\_time': str}

testrun\_id - id плана, полученное в ответе при его создании

name - имя тест-кейса

date\_time - дата и время в формате ISO (с нулевой тайм-зоной),  
например "2017-12-31T23:59:59Z"

Получение отчета в формате PDF:

GET https://<host>:<port>/testrun/<testrun\_id>/pdf

testrun\_id - id плана, полученное в ответе при его создании

Тело ответа: бинарные данные файла отчета

## Советы по выполнению задания

Эти пункты — просто рекомендации (то, чего не было в курсе, и что может быть непонятно, как делать). Если вы захотите использовать другой или более привычный способ, всё будет приниматься.

1. Для сохранения логина в рамках сессии можно использовать объект Session:  
     
   user\_session = requests.Session()  
   user\_session.post(...)  
   user\_session.get(...)  
     
   Отменить проверки сертификата также можно для всей сессии:  
   user\_session.verify = False
2. Для скачивания небольшого файла достаточно выполнить GET-запрос по ссылке, а затем сохранить в файл, открытый для записи потока байт:  
     
   response = user\_session.get(file\_url)  
   with open(file\_path, 'wb') as fh:  
    fh.write(response.content)
3. Для сравнения идентичности файлов PDF можно использовать сравнение изображений с помощью пакета программ ImageMagick.  
     
   Страница для скачивания: <https://imagemagick.org/script/download.php>  
   Дополнительно нужно установить Ghostscript:  
   <https://www.ghostscript.com/download/gsdnld.html>  
   <http://macappstore.org/ghostscript/>  
   Для Linux обычно возможна установка этих пакетов из репозитория.  
   Исполнимые файлы из пакета Ghostscript должны быть доступны через PATH, чтобы ImageMagick мог их найти.  
     
   Следующая команда позволит создать образец с разрешением 300 dpi из имеющегося документа:  
   magick -density 300 test\_2\_sample.pdf test\_2\_sample.png  
   (для Windows - magick.exe)  
     
   Следующая команда выведет число различающихся пикселей и сохранит рисунок с попиксельными различиями:  
   magick compare -metric ae <изображение 1> <изображение 2> <файл различий>  
     
   Для автоматической конвертации PDF и сравнения с образцом можно выполнить команду:  
   magick compare -metric ae -density 300 actual\_report.pdf test\_2\_sample.png diff.png