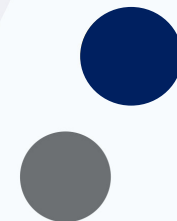


# Интеллектуальные информационные системы

## Практическое задание 2

Кафедра информатики  
Институт кибербезопасности и цифровых технологий  
РТУ МИРЭА



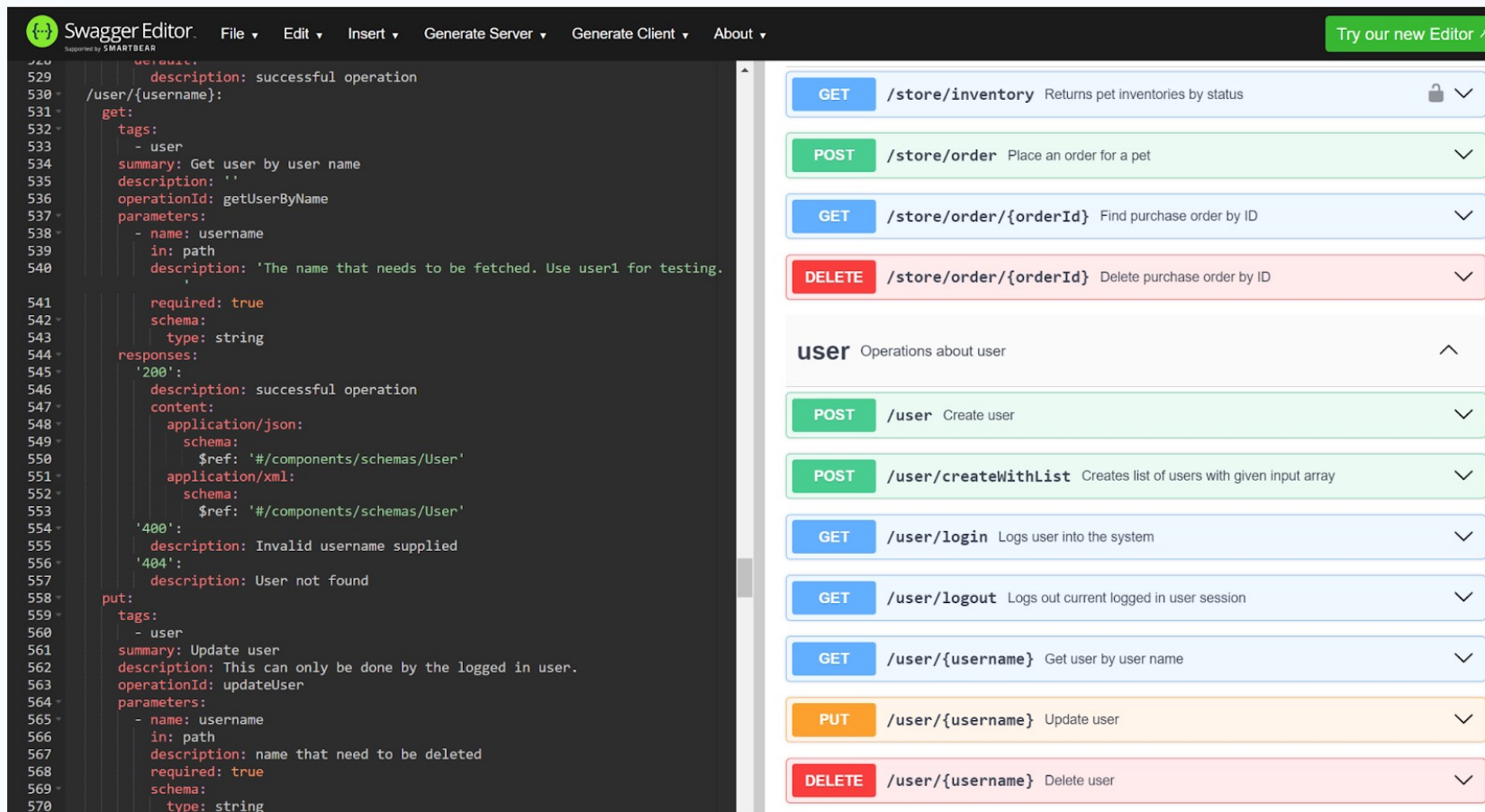
- • Описание сервера на уатл, генерация клиента и сервера
- •

## 1. Swagger Редактор

1. Открыть редактор по адресу

<https://editor.swagger.io/>

2. Изучить интерфейс



Если интересно исходники редактора можно скачать и собрать свою версию:  
<https://github.com/swagger-api/swagger-editor>

# • • Описание сервера на yaml, генерация клиента и сервера

Общая информация о методах (глаголах) http:

**GET:** Используется для запроса данных с сервера. Он безопасен и идемпотентен (многократные запросы не изменяют состояние сервера). Например, используется для получения веб-страниц.

**POST:** Применяется для отправки данных на сервер, например, при отправке формы. Этот метод может изменять состояние сервера, так как создаёт новые ресурсы.

**PUT:** Используется для обновления существующего ресурса или создания нового, если он не существует. Запрос с PUT отправляет все данные ресурса, которые должны быть обновлены.

**DELETE:** Применяется для удаления ресурса с сервера. Этот метод также идемпотентен, так как повторные запросы не изменяют состояние (ресурс уже удалён).

**PATCH:** Используется для частичного обновления ресурса. В отличие от PUT, который требует полных данных, PATCH отправляет только изменения, которые необходимо внести.

# • • Описание сервера на yaml, генерация клиента и сервера

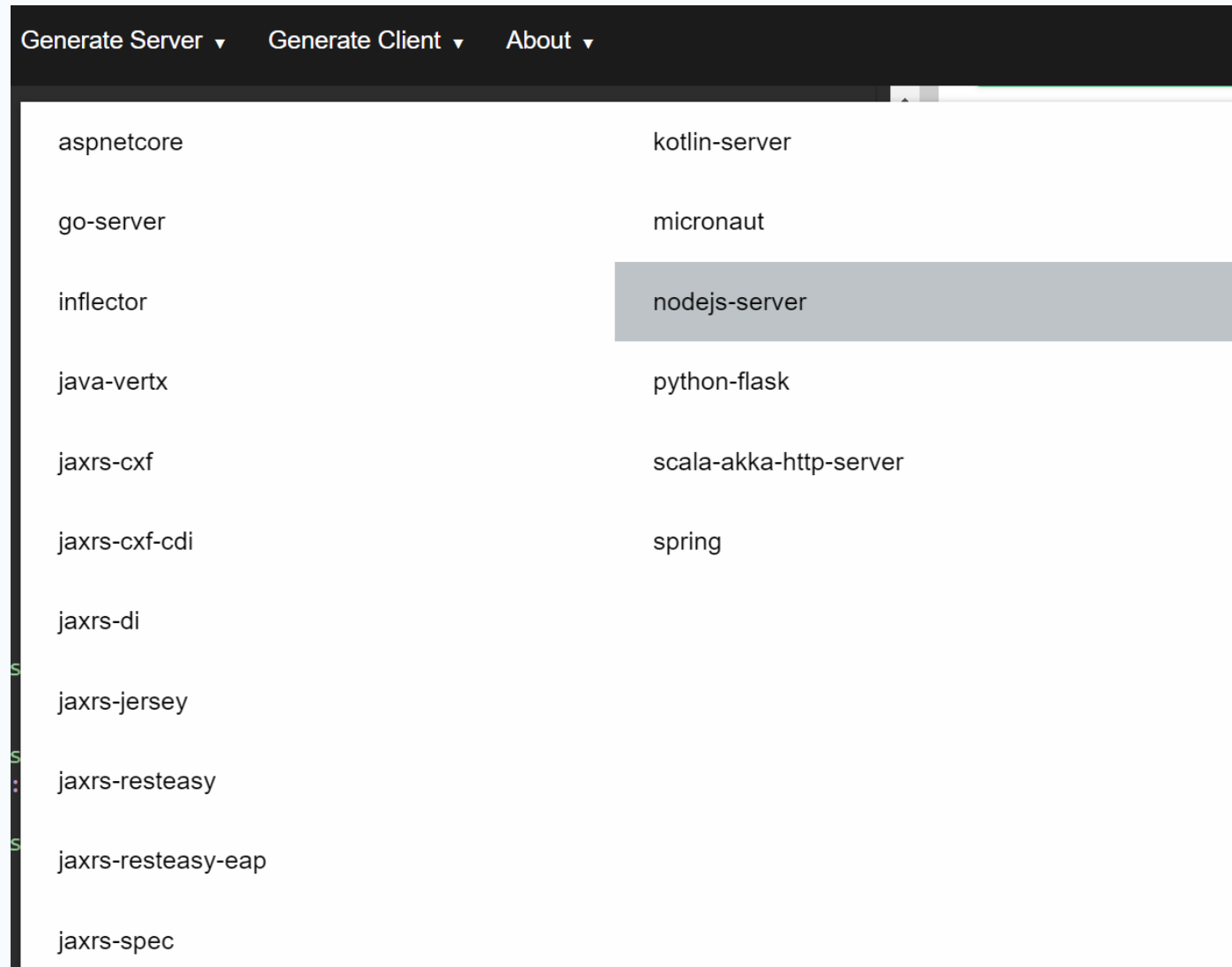
## 2. Скачать файлы из репозитория гитхаба:

1. Создайте папку на рабочей машине, откройте консоль и зайдите в эту папку.
2. Выполните в консоле команду `git clone https://github.com/labprof/IIS.git`
3. Зайдите в папку 2
4. Откройте файл `pets.yaml` в любом редакторе текста скопируйте его содержимое и вставьте в редактор Swagger
5. Изучите как изменился список методов справа
6. Скопируйте содержимое файла `pet_delete_annotation.txt` и вставьте после описание операции `/pets/{petId}: get`
7. Посмотрите как изменился список доступных операции, какая операция добавилась?

# • • Описание сервера на yam!, генерация клиента и сервера

## 3. Примеры генерации кода:

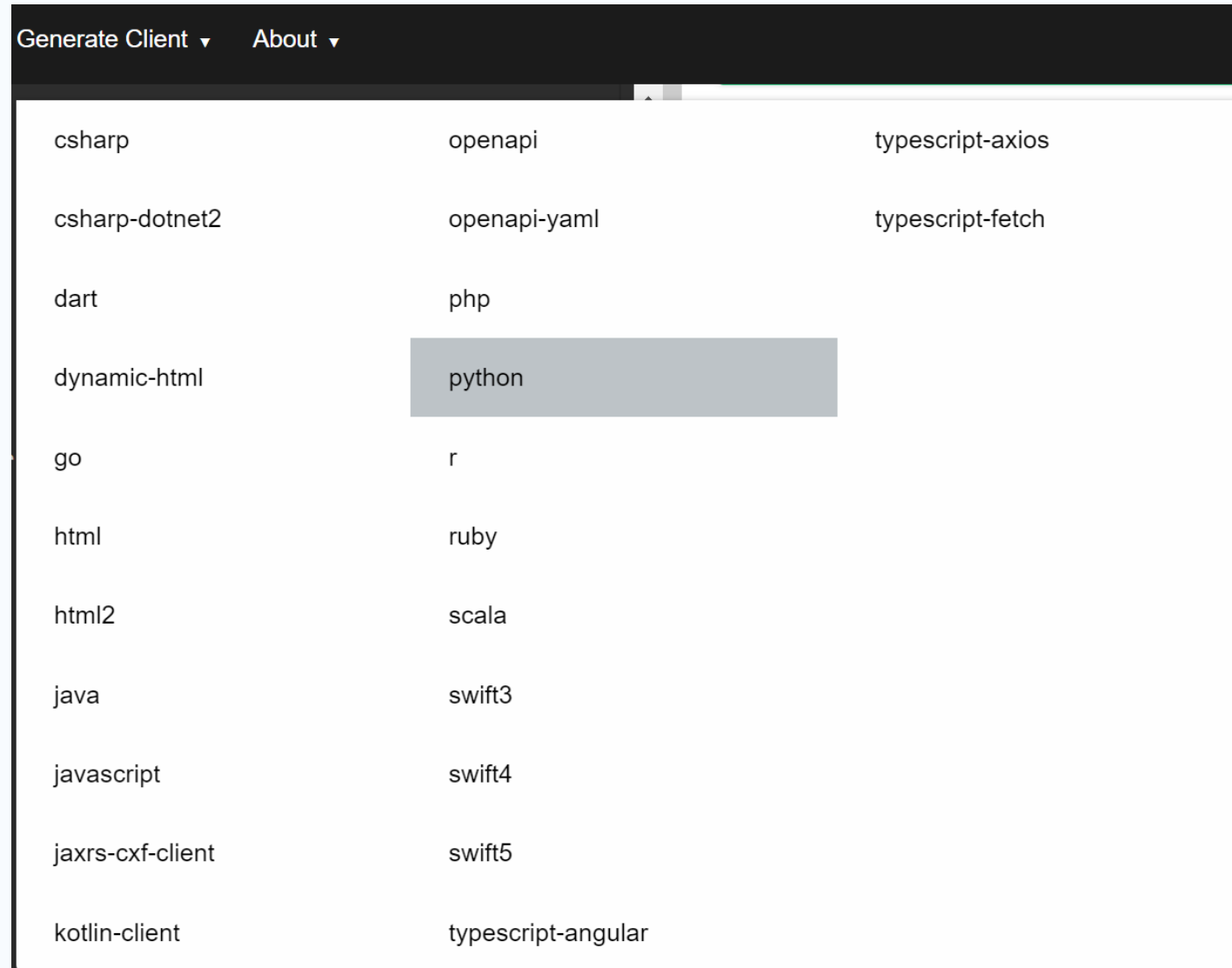
1. Сгенерировать сервер на любом представленном языке.



# • • Описание сервера на yaml, генерация клиента и сервера

2. Сгенерировать клиент на выбранном языке.

3. Изучить полученные исходные тексты.



# • • Описание сервера на уaml, генерация клиента и сервера

## 5. Запустить сервер FastApi, убедиться что работает вызов api через Swagger tool, для этого:

1. Убрать . в import model
2. Установить модуль FastApi
3. Установить модуль uvicorn: <https://www.uvicorn.org/>
4. Добавить возвращаемые значение в get метод (заглушку), так как будет ругаться без этого
5. Изменить аннотацию к get методу show\_pet\_by\_id , так как будет ругаться в response\_model=Union[Pet, Error]
6. Вызвать код из созданного Swagger tool

Пример  
сгенерированного  
кода для  
обучения с  
сохранением  
ресурса (питомца)  
в словаре:

```
pets_dict = {"0":Pet(id = 0, name = "pet", tag="hardcoded")}  
  
@app.get(  
    '/pets',  
    response_model=List[Pet],  
    responses={'default': {'model': Error}},  
    tags=['pets'],  
)  
def list_pets(limit: Optional[conint(le=100)] = None) -> Union[List[Pet], Error]:  
    """  
    список всех питомцев  
    """  
    return list(pets_dict.values())  
  
@app.post(  
    '/pets', response_model=None, responses={'default': {'model': Error}}, tags=['pets']  
)  
def create_pets(body: Pet) -> Optional[Error]:  
    """  
    Создание питомца  
    """  
    id = len(pets_dict) + 1  
    new_pet = Pet(id = id, name = f"pet number {id}")  
    pets_dict[id] = new_pet  
    return {"result": id}  
  
@app.get(  
    '/pets/{petId}',  
    response_model=Union[Pet, Error],  
    responses={'default': {'model': Error}},  
    tags=['pets'],  
)  
def show_pet_by_id(pet_id: str = Path(..., alias='petId')) -> Union[Pet, Error]:  
    """  
    Информация о питомце  
    """  
    if pets_dict.get(pet_id) is not None:  
        return pets_dict[pet_id]  
    return Error(code = 0, message = f"Pet with id {pet_id} doesn't exists")
```



# • • Описание сервера на уатл, генерация клиента и сервера

## 6. Сделать тестового клиента для работы с Аpi:

1. Создайте еще одну папку рядом (во вне) папки labs
2. Откройте консоль, зайдите в созданную папку и создайте виртуальное окружения для Python
3. Установите модуль Request
4. Написать код запроса данных и вывода на экран:

```
resp2 = requests.get('http://127.0.0.1:8000/pets')  
print(r.json())
```

5. Написать код запроса данных и вывода на экран:

```
resp2 = requests.post('http://127.0.0.1:8000/pets', json={"id" : "0", "name": "test2", "tag": "test2" })  
print(resp2.json());
```