后端接口

第一种是 json 类型的参数，就是把函数需要的参数以 json 字符串的形式传递给函数

第二种是直接参数，就是可以将对象直接赋给参数

函数会先判断直接参数是否为默认值，如果为默认值，会从 json 参数中把函数需要的参数给 load 出来

新建工程 *create_project*

功能：新建一个工程

参数：

```
project_name: str # 工程名字
project_tags: list # 预设tags
```

返回值：

```
{
    "status": true,
    "project_id": 123456,
    "message": "创建成功"
}
```

修改项目名字 *modify_project_name*

功能：修改项目名字

参数：

```
project_id: int 项目id号
new_name: str 项目新名字
```

返回值：

```
{
    "status": true,
    "code": 200,
```

```
    "message": "项目名字修改成功"
}
```

添加标签 *add_tag_to_project*

功能：向项目添加一个标签

参数：

```
project_id: int 项目id号
tag: str 标签名字
```

添加标签组 *add_tags_to_project*

功能：向项目添加多个标签

参数：

```
project_id: int 项目id号
tags: list 标签列表
```

删除标签 *delete_tag_from_project*

功能：删除项目中的某个标签

参数：

```
project_id: int 项目id号
tag: str 需要删除的标签
```

获得所有基础标签 *get_base_tags*

功能：获得所有的基础标签

参数：无

删除项目 *delete_project*

功能：删除项目

参数：

```
project_id: int 项目id号
```

返回值：

```
{
    "status": true,
    "code": 200,
    "message": "项目删除成功"
}
```

上传文件 *upload_file*

功能：将文件上传到数据库

参数：

```
file_name: str # 文件名称
file_contents: list # 文件内容
project_id: int # 项目id号
pid: int # 工程 id 号
```

返回值：

```
{
    "status": true,
    "file_id": 1,
    "code": 200,
    "message": "上传成功"
}
```

获取没有被用户标注的数据 *fetch_unlabeled_data*

如果剩余数据条目大于 num，返回 num 条数据，否则返回所有数据。

参数：

```
project_id: int # 工程 id 号
```

```
num: int # 获取的数量
```

返回值：

```
{
    "status": true,
    "data":
        [
            {
                "id": 1,
                "text": "aaa",
                "predicted_relation": "friend",
                "predicted_e1": "马云",
                "predicted_e2": "马化腾",
            },
            {
                "id": 2,
                "text": "aaa",
                "predicted_relation": "friend",
                "predicted_e1": "奥巴马",
                "predicted_e2": "特朗普",
            },
        ]
    "code": 200,
    "message": "成功取出 num 条数据",
}
```

提交标注数据 *commit_labeled_data*

参数：

```
{
    "data":
        [
            {
                "unlabeled_id": "364",
                "text": "<e1>特朗普</e1>是<e2>奥巴马</e2>的朋友",
                "project_id": 1,
                "predicted_relation": "friend",
                "predicted_e1": "奥巴马",
                "predicted_e2": "特朗普",
```

```
                "labeled_relation": "friend",
                "labeled_e1": "奥巴马",
                "labeled_e2": "特朗普",
                "additional_info": ["朋友", "是"]
            },
            ...
        ]
}
```

返回值：

```
{
    "status": True,
    "code": 200,
    "message": "已标注数据提交成功"
}
```

导出工程 *export_project*

功能：将数据库中的标注工程中已标注好的数据导出

参数：

```
project_id: 123456 # 工程 id 号
```

返回值：

```
{
    "status": True,
    "data":
        [
            "1  \"The system as described above has its greatest application in an arrayed <e1>configuration</e1> of antenna <e2>elements</e2>.\"
            Component-Whole(e2,e1)
            AdditionalInfo: Not a collection: there is structure here, organisation.

            ",
            "2  \"The <e1>child</e1> was carefully wrapped and bound into the <e2>cradle</e2> by means of a cord.\"
```

```
            Other
            AdditionalInfo:

            ",
            ...
        ],
    "code": 200,
    "message": "工程导出成功"
}
```

代码示例

```
def test_create_project(project_name="test_project"):
    interface = DB_interface()
    ret = interface.create_project(project_name='123')
    ret = interface.create_project(project_name='12345')
    ret = interface.create_project(project_name='test_project')


def test_upload_file(project_id=-1, file_name=''):
    # from annotator.models import *
    # from annotator.DB_interface import *
    # interface = DB_interface()
    # 测试通过
    print('\n上传文件', file_name)
    interface = DB_interface()
    ret = interface.upload_file(file_name='123', project_id=1,
file_contents=['奥巴马和特朗普是基友', 'Today is a good day'])
    print(ret)
    file_id = json.loads(ret)["file_id"]
    return file_id


def test_fetch_unlabeled_data(file_id=-1, project_id=-1, num=-1):
    print('\n获取未标注数据')
    interface = DB_interface()
    ret = interface.fetch_unlabeled_data(project_id=1, num=1)
    print("unlabeled_data", ret)
    # print(type(ret))
```

```
    data = json.loads(ret)["data"]
    return data


def test_commit_labaled_data(unlabeled_data, file_id=-1):
    print('\n提交已标注数据')
    interface = DB_interface()
    labeled_data = {
        "unlabeled_id": 3,
        "text": "<e1>Today</e1> is a good <e2>day1</e2>",
        "predicted_relation": "is",
        "predicted_e1": "Today",
        "predicted_e2": "day",
        "labeled_relation": "is",
        "labeled_e1": "Today",
        "labeled_e2": "day",
        "additional_info": ["a", "good"]
    }
    ret =
interface.commit_labeled_data(labeled_data=[labeled_data, ],
file_id=3)
    print("unlabeled_data", ret)


def test_export_project(project_id=-1):
    print('\n导出工程')
    interface = DB_interface()
    ret = interface.export_project(project_id=project_id)
    print("导出工程", ret)
    # print(type(ret))
    data = json.loads(ret)["data"]
    return data
```