测试及应用部署作业说明

2022-2023秋 清华大学软件学院《软件工程》课程内部资料，请勿外传。

1. 作业背景

本系列作业的背景是曾经提供给《Web前端实训课程》综合实验使用的清软论坛后端应用（基于 flask 框架开发）；另外，我们前期给大家发放了云服务器代金券，本次作业的第三部分要求同学们将应用部署到相应的服务器上。

在项目开发的过程中，测试必不可少。测试包括四个部分，分别是代码风格测试，单元测试，集成测试和端到端测试。代码风格测试保证代码仓库的风格一致性，单元测试保证基础函数功能的正确性，集成测试保证系统接口的正确性，端到端测试确保应用功能的完备性，稳定性。在本次作业的第二部分，我们要求大家为SimpleBBS补充这四部分测试。

Docker 是一个开放源代码软件项目，让应用程序布署在软件容器下的工作可以自动化进行，借此在 Linux 操作系统上，提供一个额外的软件抽象层，以及操作系统层虚拟化的自动管理机制。使用 Docker 进行应用部署有诸多好处：统一服务管理、快速部署、持续交付、应用隔离等，可以使应用做到「一次构建，到处部署」。在本次作业的第三部分，我们要求大家使用 Docker 将补全并测试后的项目部署到服务器上。

2. 作业要求

本次作业要求大家使用编程语言 Python 3 、单元测试框架 unittest 、代码覆盖率统计工具 coverage。

在本次作业的第一部分，同学们将会为项目补充四个部分的测试；在第二部分中，我们将要求同学们结合课上所学使用 Docker 将整个项目镜像化并在服务器端完成部署。

提供给大家的清软论坛后端代码使用 flask 框架，已经具备注册、登录、登出、发帖、获取帖子列表、修改帖子，回帖功能，但有一些函数没有实际功能。按照附1中的提示运行起来后，可以根据论坛的前端页面对这个简单的项目有一个更好的认识。

请注意，由于助教开发时间有限以及简化结构的考虑，该后台并不完备，存在没有日志收集以及统一的错误处理等一系列问题。同学们不应将其作为软工大作业的参考，仅作本次作业使用，同时也只需要考虑本次作业范畴内的问题。需要同学们实现的部分均有 `TODO` 的注释标记，可以使用全局搜索快速定位。

本文档附1章节中介绍了清软论坛后端的相关使用，附2章节中介绍了清软论坛的 API。

3. 测试部分（15分）

3-1 代码风格测试（2分）

在该部分中，你需要为SimpleBBS增加代码风格检查，要求如下：

- 完善flake8配置文件
  - 要求忽略且仅忽略.git，__pycache__，venv文件夹
  - 要求对app/services/post.py忽略E501错误，对app/services/user.py忽略E501错误
    对tests/test_e2e.py忽略E501错误，对tests/test_api.py忽略E501错误
- 完善格式化脚本lint.sh，脚本执行命令如下
  - 使用autopep8对代码自动格式化
  - 使用autoflake对代码自动格式化
  - 使用isort对代码自动格式化

    - 使用flake8检查代码风格
- 使用git hooks对代码风格在commit阶段进行格式化检查【可选，不占分数】

3-2 单元测试（6分）

补充基础函数和单元测试分别占50%

在该部分中，同学们需要以**测试驱动开发的方式**补完下列函数，并使用unittest补充相应的单元测试

1. `register_params_check` 函数，实现注册账号 API 参数的校验。接收参数如下：
    - `username`: 必填，用户账号
    - `password`: 必填，用户密码
    - `nickname`: 必填，用户昵称
    - `url`: 必填，用户个人地址链接
    - `mobile`: 必填，手机号
    - `magic_number`: 选填，用户喜欢的幸运数字

    参数要求：

    - 用户账号为长度5-12的字母串加数字，且必须包含这两种类型，所有字母串必须在数字前面，字母包括大写字母和小写字母
    - 用户密码为长度8-15的字符串，由大写、小写字母、数字和标点符号组成且必须包含这四种类型，有效的标签符号为-_*^
    - 用户的手机号的格式为+[区号].[手机号]，其中区号必须为两位数字，手机号必须为12位数字
    - 用户的个人地址链接包含协议和域名两部分
        - 协议部分必须为http://或者https://
        - 域名部分包含1到多个点`.`，表示以点`.`分隔的标签序列，且总长度不超过48个字符。标签序列只能由下列字符组成：
            - 大小写字母`A`到`Z`和`a`到`z`
            - 数字`0`到`9`，但最后一段顶级域名不能是纯数字（如`163.com`可以但`163.126`不可以）
            - 连字符`-`，但不能作为首尾字符
    - magic_number为非负数 int 数值，可选参数（在设计测试用例时无需考虑最大值上界）

    返回值要求：

    - 返回错误或缺失字段名（如有多个只需要按前述顺序返回第一个）以及一个 bool 值表示是否出错
    - 如果正确，返回`"ok"`以及`True`
    - 如果magic_number确实，请为content添加默认值为0的magic_number字段
2. 对register_params_check补充单元测试
   请在tests/test_basic.py的`TODO`处补充相应的单元测试，并使得行覆盖率不低于 80%，在文档中说明的所有测试用例应在测试代码中有完整体现。

3-3 集成测试（4分）

在该部分中，同学们需要为SimpleBBS添加集成测试，请补充tests/test_api.py中的`TODO`部分为注册路由、登录路由和登出路由添加测试，提供了部分注册路由测试代码供同学们参考。

3-4 端到端测试（3分）

在该部分中，同学们需要在tests/test_e2e.py中使用unittes架和selenium为SimpleBBS补充端到端测试，selenium提供了自动化控制浏览器的能力，同学们需要使用selenium控制浏览器实现用户的登录、发帖、更新帖子、删除帖子操作，在tests/test_e2e.py中提供了实现自动登录的部分供同学们参考。

由于selenium需要用到WebDriver控制浏览器，可在如下链接下载对应浏览器类型及版本的webdriver，并放置于drivers目录，将tests/test_e2e.py中的第21行和第23行修改为同学们自己的浏览器地址和webdriver地址。

- Chrome：https://chromedriver.chromium.org/downloads
- FireFox：https://github.com/mozilla/geckodriver/releases/
- Edge：https://developer.microsoft.com/en-us/microsoft-edge/tools/webdriver/

4. 服务器端部署（5分）

Docker 的相关知识可以参考Docker-从入门到实践、演示项目以及网络学堂上的课程文件。

本部分同学们需要修改的文件包括`Dockerfile`，`docker-compose.yml`和`nginx/app.conf`。

同学们需要在本部分的作业中完成如下要求：

- ssh 连接至你个人的远端服务器，安装 Docker 并保持后台运行
- 编写合适的 Dockerfile 以及 docker-compose 配置（必要时你需要调整清软论坛的代码）来实现：
    - 清软论坛以 MySQL 为数据库、通过 nginx 以 8000 端口向外提供服务；
    - 通过你的服务器 `ip:8000` 可以访问到论坛前端并正常进行各项操作；
    - 通过你的服务器 `ip:8000/api/v1` 可以直接访问后端的各项 API；
- 其他要求
    - 你的 Python 版本需要恰好为 3.8.x，你的 nginx 版本为 latest，MySQL 版本恰好为 5.7；
    - 你的 service 之间的 depends_on 依赖关系应当合理；
    - 清软论坛镜像 container 名称为 app，nginx 镜像 container 名称为 nginx，MySQL 镜像的 container 名称为 mysql；（你可以在 docker-compose 配置中指定 container name）；
    - 你的数据库使用账号 root（默认值），其密码为你的学号，数据库名称为 thss，使用端口 3306（默认值）。请注意 MySQL 默认镜像的时区为 UTC，字符集是 latin1，你也需要进行调整。可以通过在 docker-compose 中指定下述值实现：

      ```
      environment:
        - MYSQL_ROOT_PASSWORD=<你的学号>
        - MYSQL_DATABASE=thss
        - TZ=Asia/Shanghai
      command: ['mysqld', '--character-set-server=utf8mb4', '--collation-server=utf8mb4_unicode_ci']
      ```

    - 你的前端文件（static 文件夹中的内容以及 templates 目录下的 index.html）应该通过 nginx 的静态文件服务实现（在目前的后端中是通过 flask 实现的）。你可以使用 volume 实现也可以构建一个基于 nginx 的镜像实现；
    - nginx 与论坛后端处于一个 network，论坛后端与数据库处于一个 network。也即通过 nginx 所在容器无法访问数据库容器；
    - 仅 nginx 容器将端口映射给宿主机，端口号为 8000；
    - MySQL 镜像需要指定 `/home/ubuntu/mysql/` 文件夹为持久化存储 Volume；
        - 也即将镜像内 `/var/lib/mysql` 目录挂载到宿主机的 `/home/ubuntu/mysql/` 目录

    - 你的服务需要至少持续工作至作业截止日期后两周。如果无法访问，助教会与你取得联系，请保持联系方式的畅通。

5. 提交要求

在完成上述任务后，同学们需要在`实验报告.pdf`中补充自己的实验报告，报告中需要至少包括如下内容：

- 简要描述3-2单元测试部分用例的设计思路、实现思路、测试用例列表及覆盖率并进行分析；
- 简要描述3-3集成测试和3-4端到端测试部分的实现思路；
- 简要描述 Docker 部署部分的实现思路以及体会；

在完成实验报告后，同学们执行`python zip.py name student_id`即可得到压缩包文件，不允许添加新文件或者修改未包含TODO的文件，其中name为同学们自己的名字，student_id为同学们的学号，例如`python zip.py 清小软 2020000000`即可得到`清小软_2020000000.zip`的压缩包文件，请提交该压缩包文件，**不合规的提交将会被扣除部分分数**。

6. 评测说明

- “3-1 代码风格测试”部分的评测采用脚本对代码风格进行检查，正确通过即可全分；
- “3-2 单元测试”部分采用黑盒脚本对基础函数进行测试，正确通过测试用例即可全分，单元测试人工评测，覆盖率达到要求，测试用例设计恰当合理即可全分；
- “3-3 集成测试”部分采用Monkey Patch对集成测试进行测试，同学们按照顺序正确测试路由函数即可全分
- “3-4 端到端测试”部分采用Monkey Patch对端到端测试进行测试，同学们按照顺序正确控制浏览器即可全分
- “4 服务器端部署”部分采用脚本测试+人工测试的方式进行，完成要求即可得分，请注意一定按照要求来进行实现，脚本无法进行评测会扣除大量分数；
    - 注意：如未使用 Docker，最多只能获得 40% 的分数。
- DDL 日期之后，按照 $0.9^{\text{迟交天数}}$ 的衰减系数计算分数，迟交时间未满一天记作一天。

附1. 清软论坛开发说明

本项目依赖 flask 1.1.2 以及 Python 3.8.x

参考资料：[flask文档]

开发启动

```
# 安装项目依赖
pip install -r requirements.txt

python manage.py runserver
# 项目模板提供的配置文件 config.yaml 将应用跑在了 localhost:5000 上

# 会有如下提示:
# ...
# * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
# ...
# 使用浏览器访问上述地址可以看到这个作业项目自带的前端页面
```

migration

```
# 初始化数据库，会在根目录下创建一个 migrations 文件夹，并且在数据库中生成一个
alembic_version 表
python manage.py db init

# 创建迁移历史
python manage.py db migrate

# 更新数据库
python manage.py db upgrade

# 填充Fake数据
python manage.py init_db
```

test

```
python manage.py test
```

使用 gunicorn 启动

```
gunicorn -w4 -b 127.0.0.1:5000 --log-level=debug manage:app
```

附2. 清软论坛 API 文档

成功运行后，可访问http://127.0.0.1:5000/apidocs查看swagger文档

测试API-无身份验证

```
Method: GET
URL: /api/v1/hello
Response:
{
    'message': 'Hello world'
}
```

测试API-有身份验证

```
Method: GET
URL: /api/v1/hello-user
Response:
{
    'message': 'Hello world, USERNAME' # USERNAME是登录者的学号
}
```

注册

```
Method: POST
URL: /api/v1/register
Request:
{
    'username': '',
    'password': '',
    'nickname': '',
```

```
    'url': '',
    'magic_number': ''
    'mobile': '',
}
Response:
{
    'message': 'ok'
}
```

登录

```
Method: PATCH
URL: /api/v1/login
Request:
{
    'username': '',
    'password': ''
}
Response:
{
    'username': '',
    'nickname': '',
    "userId": 1,
    'jwt': ''
}
```

登出

```
Method: PATCH
URL: /api/v1/logout
Response:
{
    'message': 'ok'
}
```

获取当前登录账号个人信息

```
Method: GET
URL: /api/v1/user
Response:
{
    "id": 1,
    "username": "201901****",
    "nickname": "清小软",
    "url": "", # 个人链接地址
    "magic_number": "" # 个人幸运数字
    "mobile": “+86.13312341234”, # 手机号
    "created": "2020-08-14T23:15:49+08:00" # 创建时间
}
```

获取用户昵称

```
Method: GET
URL: /api/v1/user/:userId
Response:
{
    "id": 1,
    "nickname": "清小软",
    "created": "2020-08-14T23:15:49+08:00"
}
```

获取帖子列表

```
Method: GET
URL: /api/v1/post
QueryParam:
{
    'page': 1, # 可选，默认获取第一页
    'size': 10, # 可选，默认为10
    'userId': 1, # 可选，默认获取所有用户帖子
    'orderByReply': true # 默认表示按照主贴更新时间降序，为true表示按照最新回复时间降序
}
# Example: GET /api/v1/post?page=1&size=10&orderByReply=true
Response:
{
    'page': 1,
    'size': 10,
    'total': 1,
    'posts': [
        {
            "id": 1,
            "userId": 1,
            "nickname": "清小软",
            "title": "hello, world", # 帖子标题
            "content": "welcome to simplebbs", # 帖子内容，为富文本
            "lastRepliedUserId": 2, # 最新回复用户id，默认为发帖人，非评分要求字段
      "lastRepliedNickname": "清大软", # 最新回复用户昵称，默认为发帖人，非评分要求字段
      "lastRepliedTime": "2020-08-22T20:18:19+08:00", # 最新回复时间，非评分要求字段
            "created": "2020-08-14T00:00:00+08:00",
            "updated": "2020-08-14T00:00:00+08:00"
        }
    ]
}
```

发帖

```
Method: POST
URL: /api/v1/post
Request:
{
    "title": "欢迎使用清软论坛", # 帖子标题
    "content": "请同学们畅所欲言" # 帖子内容
}
Response:
{
    'message': 'ok',
    'postId': 1
}
```

编辑当前用户发布的帖子

```
Method: PUT
URL: /api/v1/post/:postId
# :postId 为帖子的id，需要是本人所发帖才能修改
Request:
{
    "title": "欢迎使用清软论坛", # 帖子标题
    "content": "请同学们畅所欲言！" # 帖子内容
}
Response:
{
    'message': 'ok'
}
```

获取帖子详情与回帖列表

```
Method: GET
URL: /api/v1/post/:postId
# :postId 为帖子的id
Response:
{
    "id": 1,
    "userId": 1,
    "nickname": "清小软",
    "title": "hello, world", # 帖子标题
    "content": "welcome to simplebbs", # 帖子内容
    "created": "2020-08-14T00:00:00+08:00",
    "updated": "2020-08-14T00:00:00+08:00",
    "lastRepliedTime": "2020-08-22T20:18:19+08:00", # 最新回复时间，非评分要求字段
    "reply": [ # 回帖列表，创建时间升序
        {
            "id": 1,
            "userId": 1,
            "nickname": "清小软",
            "postId": 1,
            "replyId": 0, # 回复主帖
            "content": "Hello, EveryOne!",
            "created": "2020-08-15T11:31:41+08:00",
            "updated": "2020-08-15T11:31:41+08:00"
        },
        {
            "id": 2,
```

```
            "userId": 1,
            "nickname": "清小软",
            "postId": 1,
            "replyId": 1, # 回复上一条回帖
            "content": "reply to the first reply",
            "created": "2020-08-15T11:33:23+08:00",
            "updated": "2020-08-15T11:33:23+08:00"
        }
    ]
}
```

回帖

```
Method: POST
URL: /api/v1/post/:postId/reply
# :postId 为回复帖子的id
Request:
{
    "content": "", # 回帖内容
    "replyId": 1 # 回复目标回复Id，不提供表示回复主楼
}
Response:
{
    'message': 'ok'
}
```

编辑当前用户发布的回帖

```
Method: PUT
URL: /api/v1/post/:postId/reply/:replyId
# :postId 为回复帖子的id
# :replyId 为修改回复的id
Request:
{
    content": "", # 修改回帖内容
}
Response:
{
    'message': 'ok'
}
```