软件测试报告

基于Flask的深度学习自动化部署系统

|  |  |  |  |
| --- | --- | --- | --- |
| 编写 | 许京爽，崔昕宇，张利鹏，聂磊 | 日期 | 2020年5月19日 |
| 校对 | 聂磊 | 日期 | 2020年5月19日 |

北京航空航天大学 计算机学院

软件工程综合实验课 C组

二〇二〇年五月十九日

**修订记录**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 版本号 | 日期 | 修改章节 | 修改说明 | 修改人 |
| 1.0.0 | 2020.5.19 |  | 完成第一版 | 崔昕宇，张利鹏，许京爽，聂磊 |
|  |  |  |  |  |
|  |  |  |  |  |

**目录**

[1 简介 1](#_Toc40826104)

[1.1 目的 1](#_Toc40826105)

[1.2 测试范围 1](#_Toc40826106)

[2 单元测试结果 1](#_Toc40826107)

[3 功能需求测试结果 12](#_Toc40826108)

[4 性能需求测试结果 19](#_Toc40826109)

[5 接口需求测试结果 21](#_Toc40826110)

[6 软件质量特征测试结果 22](#_Toc40826111)

[7 总结 24](#_Toc40826112)

[参考文献 24](#_Toc40826113)

基于Flask的深度学习自动化部署系统

软件测试报告

**关键词：**深度学习，数据即服务（DaaS）

**摘要：**本文描述了2020春季学期《软件工程综合实验》课程中，C组成员开发的深度学习自动化部署系统的测试报告。

**专业术语清单：**

|  |  |
| --- | --- |
| **专业术语** | **中文解释** |
| SRS | SRS指软件需求规格说明书（Software Requirements Specification），是软件需求设计阶段的产物，描述了软件所需满足的需求。 |
| Python | Python是一种跨平台的计算机程序设计语言，具有面向对象和动态类型的特性。Python最初被设计用于编写自动化脚本，随着版本的不断更新和语言新功能的添加，越多被用于独立的、大型项目的开发。 |
| URL | 统一资源定位系统（Uniform Resource Locator）是因特网的万维网服务程序上用于指定信息位置的表示方法。 |
| 深度学习 | 深度学习是机器学习领域中一个新的研究方向，通过学习样本数据的内在规律和表示层次，让机器能够像人一样具有分析学习能力，能够识别文字、图像和声音等数据。 |
| 模型部署 | 将训练完毕的深度学习模型放置在特定硬件平台上，辅以合适的软件环境，使该模型能够被用户使用。 |
| HTTP | 超文本传输协议（英文全称：HyperText Transfer Protocol）是一种用于分布式、协作式和超媒体信息系统的应用层协议。HTTP是万维网的数据通信的基础。 |
| REST API | 表现层状态转换（英文全称：Representational State Transfer）是一种万维网软件架构风格，目的是便于不同软件/程序在网络中互相传递信息。表现层状态转换是根基于超文本传输协议之上而确定的一组约束和属性，是一种设计提供万维网络服务的软件构建风格。 |

# 1 简介

## 1.1 目的

本文档是C组开发项目 ***基于Flask的深度学习自动化部署系统*** 的软件测试报告，通过对项目代码的评估，在明确测试需求的基础上，进一步完成对项目的测试和报告的撰写。

本轮测试需求分析过程以测试需求规格说明书及软件源代码为输入，分析软件存在的问题，结合软件工程综合实验具体要求，输出测试报告。本文档是为软件开发人员、测试人员、项目验收方编写的，可作为测试过程的评价指标和项目验收达标的依据。

## 1.2 测试范围

本项目的测试需求类型如下表所示，该表格描述了各个类型的测试需求在软件需求规格说明书的对应位置和测试用例编号格式。

**表1 测试需求类型**

|  |  |  |
| --- | --- | --- |
| 测试需求类型 | 测试用例编号格式 | 软件需求规格说明书对应章节 |
| 单元测试 | 0xx | 无 |
| 功能需求测试 | 1xx | 3.1 |
| 性能需求测试 | 2xx | 3.2 |
| 接口需求测试 | 3xx | 3.3 |
| 软件质量特征测试 | 4xx | 5 |

# 2 单元测试结果

**表2 用例001 login/regist接口单元测试结果表**

|  |  |  |
| --- | --- | --- |
| **名称** | login/regist接口单元测试 | |
| **简要描述** | 本测试用例用于测试regist功能是否能正确执行，并对输入数据作出检查 | |
| **前提和约束** | 安装flask框架，flask-login、flask-sqlalchemy插件，引入unittest做好测试准备  使用app = project.create\_app()初始化本项目架构 | |
| **测试方法** | 白盒测试 | |
| **测试步骤** | **1** | 使用self.client.post(‘/login /regist/’,data)向被测接口发送请求，data为本接口指定字段 |
| **2** | 使用 resp = json.loads(request.data)将返回数据转为json格式 |
| **3** | 使用self.assertEqual(resp["code"], code)判断返回值是否与预期相等 |
| **4** | 针对注册成功、数据格式错误、数据缺失、账户名已存在、确认密码错误等情况，每一种情况编写一个方法 |
| **预期结果** | ①数据缺失：当发送数据有缺失时，断言写为 self.assertEqual(resp["code"], 1005)时，执行结果为pass  ②数据格式错误：注册邮箱不符合邮箱格式时，断言写为 self.assertEqual(resp["code"], 1002)，执行结果为pass  ③账户名已存在：断言写为 self.assertEqual(resp["code"], 2002)，执行结果为pass  ④确认密码错误：注册输入的两个密码不一致时，断言写为 self.assertEqual(resp["code"], 2003)，执行结果为pass  ⑤注册成功：断言写为 self.assertEqual(resp["code"], 1000)，执行结果为pass | |
| **评价准则** | 断言self.assertEqual(resp["code"], code)是pass，还是failed | |
| **异常情况** | 在测试应当正确注册的方法中，输入已经注册过的或格式错误的邮箱，执行测试会出现failed，因为此时接口返回值为2002或1002，而预期返回值为1000  解决方法：注意该方法每次测试应当使用不同的正确格式的邮箱 | |
| **测试结果** | 各步骤的测试结果与预期结果相同 | |
| **测试结论** | 该功能通过测试 | |
| **测试人员** | 崔昕宇 | |
| **测试时间** | 2020/5/18 | |

**表3 用例002 login/login接口单元测试结果表**

|  |  |  |
| --- | --- | --- |
| **名称** | login/login接口单元测试 | |
| **简要描述** | 本测试用例用于测试login接口功能是否能正确执行，并对输入数据作出检查 | |
| **前提和约束** | 安装flask框架，flask-login、flask-sqlalchemy插件，引入unittest做好测试准备  使用app = project.create\_app()初始化本项目架构 | |
| **测试方法** | 白盒测试 | |
| **测试步骤** | **1** | 使用self.client.post(‘/login /login/’,data)向被测接口发送请求，data为本接口指定字段 |
| **2** | 使用 resp = json.loads(request.data)将返回数据转为json格式 |
| **3** | 使用self.assertEqual(resp["code"], code)判断返回值是否与预期相等 |
| **4** | 针对登录成功、数据格式错误、数据缺失、用户不存在、密码错误等情况，每一种情况编写一个方法 |
| **预期结果** | ①数据缺失：当发送数据有缺失时，断言写为 self.assertEqual(resp["code"], 1005)时，执行结果为pass  ②数据格式错误：输入邮箱不符合邮箱格式时，断言写为 self.assertEqual(resp["code"], 1002)，执行结果为pass  ③用户不存在：断言写为 self.assertEqual(resp["code"], 1007)，执行结果为pass  ④密码错误：输入密码与数据库对应的账户密码不一致时，断言写为 self.assertEqual(resp["code"], 2017)，执行结果为pass  ⑤登录成功：断言写为 self.assertEqual(resp["code"], 1000)，执行结果为pass | |
| **评价准则** | 断言self.assertEqual(resp["code"], code)是pass，还是failed | |
| **异常情况** | 在测试应当正确登录的方法中，输入未注册过的或格式错误的邮箱，执行测试会出现failed，因为此时接口返回值为1007或1002，而预期返回值为1000  解决方法：注意每种情况对应输入数据 | |
| **测试结果** | 各步骤的测试结果与预期结果相同 | |
| **测试结论** | 该功能通过测试 | |
| **测试人员** | 崔昕宇 | |
| **测试时间** | 2020/5/18 | |

**表4 用例003 project/addPro接口单元测试结果表**

|  |  |  |
| --- | --- | --- |
| **名称** | project/addPro接口单元测试 | |
| **简要描述** | 本测试用例用于测试addPro接口功能是否能正确执行，并对输入数据作出检查 | |
| **前提和约束** | 安装flask框架、flask\_login、flask-sqlalchemy插件，引入unittest做好测试准备  使用app = project.create\_app()初始化本项目架构  已有账户登录系统，并将用户名存在flask\_login.current\_user.account中 | |
| **测试方法** | 白盒测试 | |
| **测试步骤** | **1** | 使用self.client.post(‘/project/addPro/’,data)向被测接口发送请求，data为本接口指定字段 |
| **2** | 使用 resp = json.loads(request.data)将返回数据转为json格式 |
| **3** | 使用self.assertEqual(resp["code"], code)判断返回值是否与预期相等 |
| **4** | 针对新建项目成功、已存在同名项目、输入数据缺失等情况，每一种情况编写一个方法 |
| **预期结果** | ①数据缺失：当发送数据有缺失时，断言写为 self.assertEqual(resp["code"], 1005)时，执行结果为pass  ②已存在同名项目：该用户已经有同名项目时，断言写为 self.assertEqual(resp["code"], 2004)，执行结果为pass  ③新建项目成功：断言写为 self.assertEqual(resp["code"], 1000)，执行结果为pass  ④用户未登录：断言写为 self.assertEqual(resp["code"], 1001)，执行结果为pass | |
| **评价准则** | 断言self.assertEqual(resp["code"], code)是pass，还是failed | |
| **异常情况** | 在测试应当正确新建的方法中，输入已经创建过的项目名，执行测试会出现failed，因为此时接口返回值为2004，而预期返回值为1000  解决方法：注意每次执行测试新建项目的方法，项目名不能重复 | |
| **测试结果** | 各步骤的测试结果与预期结果相同 | |
| **测试结论** | 该功能通过测试 | |
| **测试人员** | 崔昕宇 | |
| **测试时间** | 2020/5/18 | |

**表5 用例004 project/editPro接口单元测试结果表**

|  |  |  |
| --- | --- | --- |
| **名称** | project/editPro接口单元测试 | |
| **简要描述** | 本测试用例用于测试editPro接口功能是否能正确执行，并对输入数据作出检查 | |
| **前提和约束** | 安装flask框架、flask\_login、flask-sqlalchemy插件，引入unittest做好测试准备  使用app = project.create\_app()初始化本项目架构  已有账户登录系统，并将用户名存在flask\_login.current\_user.account中  该账户已经存在一个项目 | |
| **测试方法** | 白盒测试 | |
| **测试步骤** | **1** | 使用self.client.post(‘/project/editPro/’,data)向被测接口发送请求，data为本接口指定字段 |
| **2** | 使用 resp = json.loads(request.data)将返回数据转为json格式 |
| **3** | 使用self.assertEqual(resp["code"], code)判断返回值是否与预期相等 |
| **4** | 针对编辑项目成功、已存在同名项目、输入数据缺失等情况，每一种情况编写一个方法 |
| **预期结果** | ①数据缺失：当发送数据有缺失时，断言写为 self.assertEqual(resp["code"], 1005)时，执行结果为pass  ②已存在同名项目：该用户已经有同名项目时，断言写为 self.assertEqual(resp["code"], 2004)，执行结果为pass  ③编辑项目成功：断言写为 self.assertEqual(resp["code"], 1000)，执行结果为pass  ④用户未登录：断言写为 self.assertEqual(resp["code"], 1001)，执行结果为pass  ⑤项目不存在：断言写为 self.assertEqual(resp["code"], 2006)，执行结果为pass | |
| **评价准则** | 断言self.assertEqual(resp["code"], code)是pass，还是failed | |
| **异常情况** | 在测试应当正确编辑的方法中，输入已经创建过的项目名，执行测试会出现failed，因为此时接口返回值为2004，而预期返回值为1000  解决方法：注意每次执行测试编辑项目的方法，项目名name不能重复 | |
| **测试结果** | 各步骤的测试结果与预期结果相同 | |
| **测试结论** | 该功能通过测试 | |
| **测试人员** | 崔昕宇 | |
| **测试时间** | 2020/5/18 | |

**表6 用例005 project/deletePro接口单元测试结果表**

|  |  |  |
| --- | --- | --- |
| **名称** | project/deletePro接口单元测试 | |
| **简要描述** | 本测试用例用于测试deletePro接口功能是否能正确执行，并对输入数据作出检查 | |
| **前提和约束** | 安装flask框架、flask\_login、flask-sqlalchemy插件，引入unittest做好测试准备  使用app = project.create\_app()初始化本项目架构  已有账户登录系统，并将用户名存在flask\_login.current\_user.account中  该账户已经存在一个项目 | |
| **测试方法** | 白盒测试 | |
| **测试步骤** | **1** | 使用self.client.post(‘/project/deletePro/’,data)向被测接口发送请求，data为本接口指定字段 |
| **2** | 使用 resp = json.loads(request.data)将返回数据转为json格式 |
| **3** | 使用self.assertEqual(resp["code"], code)判断返回值是否与预期相等 |
| **4** | 针对删除项目成功、输入数据缺失等情况，每一种情况编写一个方法 |
| **预期结果** | ①数据缺失：当发送数据有缺失时，断言写为 self.assertEqual(resp["code"], 1005)时，执行结果为pass  ②删除项目成功：断言写为 self.assertEqual(resp["code"], 1000)，执行结果为pass  ③项目不存在：断言写为 self.assertEqual(resp["code"], 2006)，执行结果为pass  ④用户未登录：断言写为 self.assertEqual(resp["code"], 1001)，执行结果为pass | |
| **评价准则** | 断言self.assertEqual(resp["code"], code)是pass，还是failed | |
| **异常情况** | 无 | |
| **测试结果** | 各步骤的测试结果与预期结果相同 | |
| **测试结论** | 该功能通过测试 | |
| **测试人员** | 崔昕宇 | |
| **测试时间** | 2020/5/18 | |

**表7 用例006 model/checkVersion接口单元测试结果表**

|  |  |  |
| --- | --- | --- |
| **名称** | model/checkVersion接口单元测试 | |
| **简要描述** | 本测试用例用于测试checkVersion接口功能是否能正确执行，并对输入数据作出检查 | |
| **前提和约束** | 安装flask框架、flask\_login、flask-sqlalchemy插件，引入unittest做好测试准备  使用app = project.create\_app()初始化本项目架构  已有账户登录系统，并将用户名存在flask\_login.current\_user.account中 | |
| **测试方法** | 白盒测试 | |
| **测试步骤** | **1** | 使用self.client.post(‘/model/checkVersion/’,data)向被测接口发送请求，data为本接口指定字段 |
| **2** | 使用 resp = json.loads(request.data)将返回数据转为json格式 |
| **3** | 使用self.assertEqual(resp["code"], code)判断返回值是否与预期相等 |
| **4** | 针对项目不存在、输入数据缺失等情况，每一种情况编写一个方法 |
| **预期结果** | ①数据缺失：当发送数据有缺失时，断言写为 self.assertEqual(resp["code"], 1005)时，执行结果为pass  ②项目不存在：断言写为 self.assertEqual(resp["code"], 2006)，执行结果为pass  ③用户未登录：断言写为 self.assertEqual(resp["code"], 1001)，执行结果为pass  ④查询成功：断言写为 self.assertEqual(resp["code"], 1000)，执行结果为pass | |
| **评价准则** | 断言self.assertEqual(resp["code"], code)是pass，还是failed | |
| **异常情况** | 无 | |
| **测试结果** | 各步骤的测试结果与预期结果相同 | |
| **测试结论** | 该功能通过测试 | |
| **测试人员** | 崔昕宇 | |
| **测试时间** | 2020/5/18 | |

**表8 用例007 model/addModel接口单元测试结果表**

|  |  |  |
| --- | --- | --- |
| **名称** | model/addModel接口单元测试 | |
| **简要描述** | 本测试用例用于测试addModel接口功能是否能正确执行，并对输入数据作出检查 | |
| **前提和约束** | 安装flask框架、flask\_login、flask-sqlalchemy插件，引入unittest做好测试准备  安装yaml和ruamel.yaml插件，能够实现对yaml文件的操作  使用app = project.create\_app()初始化本项目架构  已有账户登录系统，并将用户名存在flask\_login.current\_user.account中  被测账户已有项目，可以在项目下导入模型 | |
| **测试方法** | 白盒测试 | |
| **测试步骤** | **1** | 使用self.client.post(‘/model/addModel/<pid>’,data)向被测接口发送请求，data为本接口指定字段，pid为选定项目的id |
| **2** | 使用 resp = json.loads(request.data)将返回数据转为json格式 |
| **3** | 使用self.assertEqual(resp["code"], code)判断返回值是否与预期相等 |
| **4** | 针对项目不存在、输入数据缺失，文件格式错误等情况，每一种情况编写一个方法 |
| **预期结果** | ①数据缺失：当发送数据有缺失时，断言写为 self.assertEqual(resp["code"], 1005)时，执行结果为pass  ②项目不存在：断言写为 self.assertEqual(resp["code"], 2006)，执行结果为pass  ③用户未登录：断言写为 self.assertEqual(resp["code"], 1001)，执行结果为pass  ④文件格式错误：断言写为 self.assertEqual(resp["code"], 2018)，执行结果为pass  ⑤文件丢失：断言写为 self.assertEqual(resp["code"], 2017)，执行结果为pass  ⑥导入成功：断言写为 self.assertEqual(resp["code"], 1000)，执行结果为pass | |
| **评价准则** | 断言self.assertEqual(resp["code"], code)是pass，还是failed | |
| **异常情况** | 上传的zip文件里出现格式错误文件，返回错误码2000（服务器错误） | |
| **测试结果** | 各步骤的测试结果与预期结果相同 | |
| **测试结论** | 该功能通过测试 | |
| **测试人员** | 崔昕宇 | |
| **测试时间** | 2020/5/18 | |

**表9 用例008 model/editParam接口单元测试结果表**

|  |  |  |
| --- | --- | --- |
| **名称** | model/editParam接口单元测试 | |
| **简要描述** | 本测试用例用于测试editParam接口功能是否能正确执行，并对输入数据作出检查 | |
| **前提和约束** | 安装flask框架、flask\_login、flask-sqlalchemy插件，引入unittest做好测试准备  安装yaml和ruamel.yaml插件，能够实现对yaml文件的操作  使用app = project.create\_app()初始化本项目架构  已有账户登录系统，并将用户名存在flask\_login.current\_user.account中  被测账户的项目下已有模型，可以修改模型参数 | |
| **测试方法** | 白盒测试 | |
| **测试步骤** | **1** | 使用self.client.post(‘/model/editParam/<model\_id>’,data)向被测接口发送请求，data为本接口指定字段，model\_id为选定模型的id |
| **2** | 使用 resp = json.loads(request.data)将返回数据转为json格式 |
| **3** | 使用self.assertEqual(resp["code"], code)判断返回值是否与预期相等 |
| **4** | 针对模型不存在、输入数据缺失等情况，每一种情况编写一个方法 |
| **预期结果** | ①数据缺失：当发送数据有缺失时，断言写为 self.assertEqual(resp["code"], 1005)时，执行结果为pass  ②模型不存在：断言写为 self.assertEqual(resp["code"], 2007)，执行结果为pass  ③用户未登录：断言写为 self.assertEqual(resp["code"], 1001)，执行结果为pass  ④模型已经部署，不能修改参数：断言写为 self.assertEqual(resp["code"], 2019)，执行结果为pass  ⑤修改成功：断言写为 self.assertEqual(resp["code"], 1000)，执行结果为pass | |
| **评价准则** | 断言self.assertEqual(resp["code"], code)是pass，还是failed | |
| **异常情况** | 找不到配置文件地址，返回错误码2000（服务器错误），测试结果为failed | |
| **测试结果** | 各步骤的测试结果与预期结果相同 | |
| **测试结论** | 该功能通过测试 | |
| **测试人员** | 崔昕宇 | |
| **测试时间** | 2020/5/18 | |

**表10 用例009 model/deleteModel接口单元测试结果表**

|  |  |  |
| --- | --- | --- |
| **名称** | model/deleteModel接口单元测试 | |
| **简要描述** | 本测试用例用于测试deleteModel接口功能是否能正确执行，并对输入数据作出检查 | |
| **前提和约束** | 安装flask框架、flask\_login、flask-sqlalchemy插件，引入unittest做好测试准备  使用app = project.create\_app()初始化本项目架构  已有账户登录系统，并将用户名存在flask\_login.current\_user.account中  被测账户的项目下已有模型，可以删除模型 | |
| **测试方法** | 白盒测试 | |
| **测试步骤** | **1** | 使用self.client.post(‘/model/deleteModel/’,data)向被测接口发送请求，data为本接口指定字段 |
| **2** | 使用 resp = json.loads(request.data)将返回数据转为json格式 |
| **3** | 使用self.assertEqual(resp["code"], code)判断返回值是否与预期相等 |
| **4** | 针对模型不存在、输入数据缺失等情况，每一种情况编写一个方法 |
| **预期结果** | ①数据缺失：当发送数据有缺失时，断言写为 self.assertEqual(resp["code"], 1005)时，执行结果为pass  ②模型不存在：断言写为 self.assertEqual(resp["code"], 2007)，执行结果为pass  ③用户未登录：断言写为 self.assertEqual(resp["code"], 1001)，执行结果为pass  ④模型正在运行，不能删除模型：断言写为 self.assertEqual(resp["code"], 2013)，执行结果为pass  ⑤删除成功：断言写为 self.assertEqual(resp["code"], 1000)，执行结果为pass | |
| **评价准则** | 断言self.assertEqual(resp["code"], code)是pass，还是failed | |
| **异常情况** | 无 | |
| **测试结果** | 各步骤的测试结果与预期结果相同 | |
| **测试结论** | 该功能通过测试 | |
| **测试人员** | 崔昕宇 | |
| **测试时间** | 2020/5/18 | |

**表11 用例010 record/startModel接口单元测试结果表**

|  |  |  |
| --- | --- | --- |
| **名称** | record/startModel接口单元测试 | |
| **简要描述** | 本测试用例用于测试startModel接口功能是否能正确执行，并对输入数据作出检查 | |
| **前提和约束** | 安装flask框架、flask\_login、flask-sqlalchemy插件，引入unittest做好测试准备  配置TensorFlow v2.1.0&v1.14.0  配置pyTorch v1.5.0  使用app = project.create\_app()初始化本项目架构  已有账户登录系统，并将用户名存在flask\_login.current\_user.account中  被测账户的项目下已有模型，可以启动该模型 | |
| **测试方法** | 白盒测试 | |
| **测试步骤** | **1** | 使用self.client.post(‘/record/startModel/’,data)向被测接口发送请求，data为本接口指定字段 |
| **2** | 使用 resp = json.loads(request.data)将返回数据转为json格式 |
| **3** | 使用self.assertEqual(resp["code"], code)判断返回值是否与预期相等 |
| **4** | 针对模型不存在、输入数据缺失、模型已经部署等情况，每一种情况编写一个方法 |
| **预期结果** | ①数据缺失：当发送数据有缺失时，断言写为 self.assertEqual(resp["code"], 1005)时，执行结果为pass  ②模型不存在：断言写为 self.assertEqual(resp["code"], 2007)，执行结果为pass  ③用户未登录：断言写为 self.assertEqual(resp["code"], 1001)，执行结果为pass  ④模型已经在部署状态：断言写为 self.assertEqual(resp["code"], 2015)，执行结果为pass  ⑤用户部署模型数达到上限：断言写为 self.assertEqual(resp["code"], 2016)，执行结果为pass  ⑥部署运行成功：断言写为 self.assertEqual(resp["code"], 1000)，执行结果为pass | |
| **评价准则** | 断言self.assertEqual(resp["code"], code)是pass，还是failed | |
| **异常情况** | 调用的部署接口deploy()发生错误，返回错误码2000（服务器错误） | |
| **测试结果** | 各步骤的测试结果与预期结果相同 | |
| **测试结论** | 该功能通过测试 | |
| **测试人员** | 崔昕宇 | |
| **测试时间** | 2020/5/18 | |

**表12 用例011 record/ deleteRecord接口单元测试结果表**

|  |  |  |
| --- | --- | --- |
| **名称** | record/ deleteRecord接口单元测试 | |
| **简要描述** | 本测试用例用于测试deleteRecord接口功能是否能正确执行，并对输入数据作出检查 | |
| **前提和约束** | 安装flask框架、flask\_login、flask-sqlalchemy插件，引入unittest做好测试准备  安装yaml和ruamel.yaml插件，能够实现对yaml文件的操作  配置TensorFlow v2.1.0&v1.14.0  配置pyTorch v1.5.0  使用app = project.create\_app()初始化本项目架构  已有账户登录系统，并将用户名存在flask\_login.current\_user.account中  被测账户的项目下已有模型启动了实例，可以删除该实例 | |
| **测试方法** | 白盒测试 | |
| **测试步骤** | **1** | 使用self.client.post(‘/record/deleteRecord/’,data)向被测接口发送请求，data为本接口指定字段 |
| **2** | 使用 resp = json.loads(request.data)将返回数据转为json格式 |
| **3** | 使用self.assertEqual(resp["code"], code)判断返回值是否与预期相等 |
| **4** | 针对实例不存在、输入数据缺失等情况，每一种情况编写一个方法 |
| **预期结果** | ①数据缺失：当发送数据有缺失时，断言写为 self.assertEqual(resp["code"], 1005)时，执行结果为pass  ②实例不存在：断言写为 self.assertEqual(resp["code"], 2020)，执行结果为pass  ③用户未登录：断言写为 self.assertEqual(resp["code"], 1001)，执行结果为pass  ④实例在运行状态，不能删除：断言写为 self.assertEqual(resp["code"], 2020)，执行结果为pass  ⑤实例运行成功：断言写为 self.assertEqual(resp["code"], 1000)，执行结果为pass | |
| **评价准则** | 断言self.assertEqual(resp["code"], code)是pass，还是failed | |
| **异常情况** | 调用的部署接口delete()发生错误，返回错误码2000（服务器错误） | |
| **测试结果** | 各步骤的测试结果与预期结果相同 | |
| **测试结论** | 该功能通过测试 | |
| **测试人员** | 崔昕宇 | |
| **测试时间** | 2020/5/18 | |

**表13 用例012 record/ pauseModel接口单元测试结果表**

|  |  |  |
| --- | --- | --- |
| **名称** | record/pauseModel接口单元测试 | |
| **简要描述** | 本测试用例用于测试pauseModel接口功能是否能正确执行，并对输入数据作出检查 | |
| **前提和约束** | 安装flask框架、flask\_login、flask-sqlalchemy插件，引入unittest做好测试准备  配置TensorFlow v2.1.0&v1.14.0  配置pyTorch v1.5.0  使用app = project.create\_app()初始化本项目架构  已有账户登录系统，并将用户名存在flask\_login.current\_user.account中  被测账户的项目下已有模型启动了实例，可以删除该实例 | |
| **测试方法** | 白盒测试 | |
| **测试步骤** | **1** | 使用self.client.post(‘/record/pauseModel/’,data)向被测接口发送请求，data为本接口指定字段 |
| **2** | 使用 resp = json.loads(request.data)将返回数据转为json格式 |
| **3** | 使用self.assertEqual(resp["code"], code)判断返回值是否与预期相等 |
| **4** | 针对实例不存在、输入数据缺失等情况，每一种情况编写一个方法 |
| **预期结果** | ①数据缺失：当发送数据有缺失时，断言写为 self.assertEqual(resp["code"], 1005)时，执行结果为pass  ②实例不存在：断言写为 self.assertEqual(resp["code"], 2020)，执行结果为pass  ③用户未登录：断言写为 self.assertEqual(resp["code"], 1001)，执行结果为pass  ④实例已经在暂停状态：断言写为 self.assertEqual(resp["code"], 2014)，执行结果为pass  ⑤实例暂停成功：断言写为 self.assertEqual(resp["code"], 1000)，执行结果为pass | |
| **评价准则** | 断言self.assertEqual(resp["code"], code)是pass，还是failed | |
| **异常情况** | 调用的部署接口pause()发生错误，返回错误码2000（服务器错误） | |
| **测试结果** | 各步骤的测试结果与预期结果相同 | |
| **测试结论** | 该功能通过测试 | |
| **测试人员** | 崔昕宇 | |
| **测试时间** | 2020/5/18 | |

**表14 用例013 record/restartModel接口单元测试结果表**

|  |  |  |
| --- | --- | --- |
| **名称** | record/restartModel接口单元测试 | |
| **简要描述** | 本测试用例用于测试restartModel接口功能是否能正确执行，并对输入数据作出检查 | |
| **前提和约束** | 安装flask框架、flask\_login、flask-sqlalchemy插件，引入unittest做好测试准备  配置TensorFlow v2.1.0&v1.14.0  配置pyTorch v1.5.0  使用app = project.create\_app()初始化本项目架构  已有账户登录系统，并将用户名存在flask\_login.current\_user.account中  被测账户的项目下已有模型的实例被暂停，可以重新启动该实例 | |
| **测试方法** | 白盒测试 | |
| **测试步骤** | **1** | 使用self.client.post(‘/record/restartModel/’,data)向被测接口发送请求，data为本接口指定字段 |
| **2** | 使用 resp = json.loads(request.data)将返回数据转为json格式 |
| **3** | 使用self.assertEqual(resp["code"], code)判断返回值是否与预期相等 |
| **4** | 针对模型不存在、输入数据缺失、实例已运行等情况，每一种情况编写一个方法 |
| **预期结果** | ①数据缺失：当发送数据有缺失时，断言写为 self.assertEqual(resp["code"], 1005)时，执行结果为pass  ②模型不存在：断言写为 self.assertEqual(resp["code"], 2007)，执行结果为pass  ③用户未登录：断言写为 self.assertEqual(resp["code"], 1001)，执行结果为pass  ④实例已经运行：断言写为 self.assertEqual(resp["code"], 2015)，执行结果为pass  ⑤用户部署模型数达到上限：断言写为 self.assertEqual(resp["code"], 2016)，执行结果为pass  ⑥实例重启成功：断言写为 self.assertEqual(resp["code"], 1000)，执行结果为pass | |
| **评价准则** | 断言self.assertEqual(resp["code"], code)是pass，还是failed | |
| **异常情况** | 调用的部署接口restart()发生错误，返回错误码2000（服务器错误） | |
| **测试结果** | 各步骤的测试结果与预期结果相同 | |
| **测试结论** | 该功能通过测试 | |
| **测试人员** | 崔昕宇 | |
| **测试时间** | 2020/5/18 | |

**表15 用例014 部署模块-deploy单元测试**

|  |  |  |  |
| --- | --- | --- | --- |
| **名称** | deploy单元测试 | | |
| **简要描述** | 本测试主要用于在部署模块内部检测模型的实例能否正常启动运行。 | | |
| **前提和约束** | 给定模型文件、类型、配置项文件以及模型模板；且需满足实例运行的内存和数量限制。 | | |
| **测试方法** | 白盒测试 | | |
| **测试步骤** | **1** | 准备模型文件（以MNIST数据集及其模型为例）、配置项文件、模型模板文件；为后续测试服务是否可用，还需准备该模型对应的输入样例。 | |
| **2** | 设置内存限制（包含以用户为单位、以项目为单位、以实例为单位）、数量限制（以用户为单位）。   |  |  |  | | --- | --- | --- | | 内存限制 | 以用户为单位 | 2000MB | | 以项目为单位 | 1024MB | | 以实例为单位 | 512MB | | 数量限制 | 以用户为单位 | 10个 | | 以项目为单位 | 5个 | | |
| **3** | 调用deploy()方法，将上述参数传入。并将结果打印至控制台。 | |
| **4** | 使用post方法（form\_data形式）请求步骤3中传出的url，携带待预测数据和key。 | |
| **预期结果** | 步骤3 : 控制台打印该实例运行的URL、端口、实例运行进程ID、key值以及状态码。  步骤4 : 服务器返回预测结果。 | | |
| **评价准则** | 1. 超出设定的内存和数量限制时选择不运行实例或杀掉实例运行进程；未超出限制时则正常运行。 2. 返回的部署结果URL是可用的，只有在携带key的情况下才能请求服务。 3. 请求服务返回的结果与该模型在其他条件下（如本地）得到的结果是一致的。 | | |
| **异常情况** | 1. 模型文件不存在或加载错误 | | 程序终止运行并返回相应错误码（4031、4034） |
| 1. 配置文件不存在或配置项异常 | | 程序终止运行并返回相应错误码（4033） |
| 1. 无可用端口 | | 程序终止运行并返回相应错误码（4036） |
| 1. 运行实例的flask程序启动失败 | | 程序终止运行并返回相应错误码（4035） |
| **测试结果** | 各步骤的测试结果与预期结果相同 | | |
| **测试结论** | 该接口通过测试 | | |
| **测试人员** | 张利鹏 | | |
| **测试时间** | 2020/5/17 | | |

**表16 用例015 部署模块-delete单元测试**

|  |  |  |  |
| --- | --- | --- | --- |
| **名称** | delete单元测试 | | |
| **简要描述** | 本测试主要用于在部署模块内部检测模型的实例能否正常删除。 | | |
| **前提和约束** | 给定模型实例的进程ID，该实例不处于运行状态。 | | |
| **测试方法** | 白盒测试 | | |
| **测试步骤** | **1** | 将pid传入delete()，调用该方法。 | |
| **2** | 打印delete()方法的运行结果。 | |
| **预期结果** | 步骤2 : 控制台输出删除结果的状态码。 | | |
| **评价准则** | 1. 返回的状态码为4044。 2. 请求该实例服务时URL不可访问。 | | |
| **异常情况** | 1. 实例正在运行状态 | | 程序终止运行并返回相应错误码（4042） |
| 1. 不存在该实例 | | 程序终止运行并返回相应错误码（4041） |
| **测试结果** | 各步骤的测试结果与预期结果相同 | | |
| **测试结论** | 该接口通过测试 | | |
| **测试人员** | 张利鹏 | | |
| **测试时间** | 2020/5/18 | | |

**表17 用例016 部署模块-pause单元测试**

|  |  |  |  |
| --- | --- | --- | --- |
| **名称** | pause单元测试 | | |
| **简要描述** | 本测试主要用于在部署模块内部检测模型的实例能否正常暂停服务。 | | |
| **前提和约束** | 给定模型实例的进程ID以及运行进程所在的端口；该实例处于运行状态。 | | |
| **测试方法** | 白盒测试 | | |
| **测试步骤** | **1** | 将pid和port参数传入pause()，调用该方法。 | |
| **2** | 打印pause()的运行结果至控制台。 | |
| **预期结果** | 步骤2 : 控制台输出暂停结果（True | False） | | |
| **评价准则** | 1. 返回的暂停结果为True。 2. 请求该实例服务时URL不可访问。 | | |
| **异常情况** | 1. 传入端口参数port无进程运行 | | 程序终止运行并返回False |
| 1. 传入端口参数port对应进程ID与传入参数pid不一致 | | 程序终止运行并返回False |
| **测试结果** | 各步骤的测试结果与预期结果相同 | | |
| **测试结论** | 该接口通过测试 | | |
| **测试人员** | 张利鹏 | | |
| **测试时间** | 2020/5/18 | | |

**表18 用例017 部署模块-restart单元测试**

|  |  |  |  |
| --- | --- | --- | --- |
| **名称** | restart单元测试 | | |
| **简要描述** | 本测试主要用于在部署模块内部检测处于暂停状态的模型实例能否正常地恢复服务。 | | |
| **前提和约束** | 给定模型实例的进程ID以及运行程序所在的端口；实例处于暂停状态。 | | |
| **测试方法** | 白盒测试 | | |
| **测试步骤** | **1** | 将pid和port参数传入restart()，调用该方法。 | |
| **2** | 打印restart()的运行结果至控制台。 | |
| **预期结果** | 步骤2 : 控制台输出暂停结果（True | False） | | |
| **评价准则** | 1. 返回的重启结果为True。 2. 请求该实例服务时URL可访问，且携带key和待遇测数据请求服务所返回的结果与其他环境下模型的表现一致。 | | |
| **异常情况** | 1. 传入端口参数port无进程运行 | | 程序终止运行并返回False |
| 1. 传入端口参数port对应进程ID与传入参数pid不一致 | | 程序终止运行并返回False |
| **测试结果** | 各步骤的测试结果与预期结果相同 | | |
| **测试结论** | 该接口通过测试 | | |
| **测试人员** | 张利鹏 | | |
| **测试时间** | 2020/5/18 | | |

# 3 功能需求测试结果

**表19 用例101用户注册功能测试**

|  |  |  |  |
| --- | --- | --- | --- |
| **名称** | 用户注册功能测试 | | |
| **简要描述** | 本测试主要用于测试用户是否能在网站注册账号以获取网站功能的使用权限 | | |
| **前提和约束** | 安装好Flask及系统运行环境 | | |
| **测试方法** | 黑盒测试 | | |
| **测试步骤** | **1** | 登陆网站页面，点击注册按钮 | |
| **2** | 根据页面提示信息，填写注册的用户名、设置密码、确认密码和邮箱 | |
| **3** | 点击注册按钮 | |
| **预期结果** | 页面返回用户注册成功结果信息，并提示用户可以通过注册的账号登陆，将正确有效的注册信息存入账户数据库中 | | |
| **异常情况** | 用户在页面注册中填写的信息不符合提示要求规范 | | 点击注册按钮后，不符合规范要求的输入框会标红并再次弹出正确的规范提示，让用户进行修改 |
| 输入的用户名在账户数据库中已存在 | | 注册失败，页面显示提示信息“该用户名已存在” |
| 设置密码和确认密码不匹配 | | 注册失败，在页面显示提示信息“两次输入的密码不匹配” |
| 输入的邮箱格式不正确 | | 注册失败，在页面显示提示信息“邮箱格式不正确” |
| 输入的邮箱在账户数据库中已存在 | | 注册失败，在页面显示提示信息“该邮箱已被注册” |
| **测试结果** | 各步骤的测试结果与预期结果相同 | | |
| **测试结论** | 该功能通过测试 | | |
| **测试人员** | 许京爽 | | |
| **测试时间** | 2020/5/16 | | |

**表20 用例102用户登录功能测试**

|  |  |  |  |
| --- | --- | --- | --- |
| **名称** | 用户登录功能测试 | | |
| **简要描述** | 本测试主要用于测试用户是否能通过输入正确的用户名和密码登录其账号，获得管理其私有的项目和模型信息的权限 | | |
| **前提和约束** | 安装好Flask及系统运行环境 | | |
| **测试方法** | 黑盒测试 | | |
| **测试步骤** | **1** | 登陆网站页面，点击登录按钮 | |
| **2** | 根据页面提示信息，输入用户名/邮箱和对应的密码 | |
| **3** | 点击登录按钮 | |
| **预期结果** | 后端数据库进行搜索，当信息匹配时，前端页面进行跳转到用户个人主页 | | |
| **异常情况** | 输入的用户名/邮箱和密码在账户数据库中不匹配 | | 登录失败，在页面显示提示信息“密码或用户名有误” |
| **测试结果** | 各步骤的测试结果与预期结果相同 | | |
| **测试结论** | 该功能通过测试 | | |
| **测试人员** | 许京爽 | | |
| **测试时间** | 2020/5/16 | | |

**表21 用例103用户登录功能测试**

|  |  |  |  |
| --- | --- | --- | --- |
| **名称** | 用户登出功能测试 | | |
| **简要描述** | 本测试主要用于测试用户在使用完毕管理功能后，可登出账户，避免他人误操作用户数据，滥用权限 | | |
| **前提和约束** | 安装好Flask及系统运行环境 | | |
| **测试方法** | 黑盒测试 | | |
| **测试步骤** | **1** | 在个人页面中，点击右上角退出按钮 | |
| **预期结果** | 页面跳转到登录页面 | | |
| **异常情况** | 用户登录后，打开多个个人页面，登出其中一个后，在另一个停留的个人页面中进行操作 | | 操作失败，直接跳转到首页登录页面 |
| **测试结果** | 各步骤的测试结果与预期结果相同 | | |
| **测试结论** | 该功能通过测试 | | |
| **测试人员** | 许京爽 | | |
| **测试时间** | 2020/5/16 | | |

**表22 用例104用户新建项目功能测试**

|  |  |  |  |
| --- | --- | --- | --- |
| **名称** | 用户新建项目功能测试 | | |
| **简要描述** | 本测试主要用于测试用户在个人页面中能否创建新的项目 | | |
| **前提和约束** | 安装好Flask及系统运行环境 | | |
| **测试方法** | 黑盒测试 | | |
| **测试步骤** | **1** | 在个人页面中，点击新建项目 | |
| **2** | 跳转到创建项目页面中，输入项目名称、项目路由和项目描述，其中项目描述不是必填项 | |
| **3** | 点击创建按钮 | |
| **预期结果** | 跳转到项目页面，新建的项目出现在项目页面中，项目信息传递给后端数据库 | | |
| **异常情况** | 当前用户存在同名的项目 | | 添加项目失败，输出存在同名项目提示信息 |
| **测试结果** | 该功能正常情况下的测试结果与预期结果相同，但是对于同名项目的创建未给出对应提示 | | |
| **测试结论** | 该功能的异常处理未通过测试 | | |
| **测试人员** | 许京爽 | | |
| **测试时间** | 2020/5/16 | | |

**表23 用例105用户查看项目功能测试**

|  |  |  |  |
| --- | --- | --- | --- |
| **名称** | 用户查看项目功能测试 | | |
| **简要描述** | 本测试主要用于测试用户是否能够查看个人页面中的所有项目 | | |
| **前提和约束** | 安装好Flask及系统运行环境 | | |
| **测试方法** | 黑盒测试 | | |
| **测试步骤** | **1** | 在个人页面中，选择所需要查看的项目列表 | |
| **2** | 点击当前选定项目的查看按钮 | |
| **预期结果** | 跳转到相应的项目页面，能够显示当前项目的基本信息及其模型 | | |
| **异常情况** | 当用户名下没有任何项目时 | | 此时项目列表为空，没有查看功能 |
| **测试结果** | 各步骤的测试结果与预期结果相同 | | |
| **测试结论** | 该功能通过测试 | | |
| **测试人员** | 许京爽 | | |
| **测试时间** | 2020/5/16 | | |

**表24 用例106用户删除项目功能测试**

|  |  |  |
| --- | --- | --- |
| **名称** | 用户删除项目功能测试 | |
| **简要描述** | 本测试主要用于测试能否实项目管理中删除项目的需求 | |
| **前提和约束** | 安装好Flask及系统运行环境 | |
| **测试方法** | 黑盒测试 | |
| **测试步骤** | **1** | 在个人页面中，选择需要删除的项目 |
| **2** | 点击当前选定项目的删除按钮 |
| **3** | 核实项目名称后，点击确定按钮 |
| **预期结果** | 当前项目列表中去掉了删除的项目，后端数据库中去除用户-项目 | |
| **测试结果** | 各步骤的测试结果与预期结果相同 | |
| **测试结论** | 该功能通过测试 | |
| **测试人员** | 许京爽 | |
| **测试时间** | 2020/5/16 | |

**表25 用例107用户更新项目功能测试**

|  |  |  |  |
| --- | --- | --- | --- |
| **名称** | 用户更新项目功能测试 | | |
| **简要描述** | 本测试主要用于测试用户在新建项目一段时间后，发现当初新建项目时填写的信息并不准确，而删除项目重建会影响到项目下包含的模型和实例，因此只修改更新项目的功能 | | |
| **前提和约束** | 安装好Flask及系统运行环境 | | |
| **测试方法** | 黑盒测试 | | |
| **测试步骤** | **1** | 在个人页面中，选择所需要更新的项目列表 | |
| **2** | 点击当前选定项目的修改按钮 | |
| **3** | 选择对应要修改的内容（项目名称、路由、描述）进行填写 | |
| **4** | 点击保存更改 | |
| **预期结果** | 当前项目的修改信息进行更新，包括此项目下的所有模型和实例，其对应的项目名也进行了修改 | | |
| **异常情况** | 修改的项目名与已存在的项目名重复 | | 修改失败，提示“该用户下存在同名项目” |
| **测试结果** | 各步骤的测试结果与预期结果相同 | | |
| **测试结论** | 该功能通过测试 | | |
| **测试人员** | 许京爽 | | |
| **测试时间** | 2020/5/16 | | |

**表26 用例108项目中导入模型功能测试**

|  |  |  |  |
| --- | --- | --- | --- |
| **名称** | 项目中导入模型功能测试 | | |
| **简要描述** | 本测试主要用于测试是否能够在制定项目中导入模型文件 | | |
| **前提和约束** | 安装好Flask及系统运行环境 | | |
| **测试方法** | 黑盒测试 | | |
| **测试步骤** | **1** | 在指定的项目页面，点击导入模型按钮 | |
| **2** | 跳转到导入模型页面，填写名称、类型、版本，以及选择本地的模型文件进行上传，模型描述为选填项 | |
| **3** | 点击导入按钮 | |
| **预期结果** | 需要上传的模型上传至指定的项目页面下，可以查看此模型的详情 | | |
| **异常情况** | 后端只能找到文件而找不到模型信息 | | 再次设置模型信息到数据库中，直到成功保存至数据库，返回模型导入成功 |
| 只能找到模型信息找不到模型文件 | | 再次将模型保存，生产文件路径保存至数据库中，返回模型导入成功 |
| **测试结果** | 各步骤的测试结果与预期结果相同 | | |
| **测试结论** | 该功能通过测试 | | |
| **测试人员** | 许京爽 | | |
| **测试时间** | 2020/5/16 | | |

**表27 用例109模型部署参数设置功能测试**

|  |  |  |  |
| --- | --- | --- | --- |
| **名称** | 模型部署参数设置功能测试 | | |
| **简要描述** | 本测试主要用于测试是否能够配置指定模型的参数 | | |
| **前提和约束** | 安装好Flask及系统运行环境 | | |
| **测试方法** | 黑盒测试 | | |
| **测试步骤** | **1** | 在指定的项目页面中，选择所要设置参数的模型，点击设置参数按钮 | |
| **2** | 跳转到参数设置页面，选择网络服务运行环境 | |
| **3** | 选择是否需要预留内存，是的话，填入预留内存的大小 | |
| **4** | 点击确定按钮 | |
| **5** | 重新启动模型实例 | |
| **预期结果** | 页面返回设置成功，更新指定模型的参数配置，可成功地重新启动模型实例 | | |
| **异常情况** | 用户预留的内存大小超过最大可预留的大小 | | 配置失败，提示“修改失败” |
| 溢出、通讯失败的服务器错误 | | 配置失败，提示“服务器错误，请稍后” |
| 用户处于非法登录状态 | | 配置失败，提示“用户未登录” |
| **测试结果** | 设置内存限制为300MB时，尝试启动实例，浏览器显示“服务器错误，请检查参数”错误提示 | | |
| **测试结论** | 该功能未通过测试 | | |
| **测试人员** | 许京爽 | | |
| **测试时间** | 2020/5/16 | | |

**表28 用例110查看项目下所有模型及其配置的功能测试**

|  |  |  |  |
| --- | --- | --- | --- |
| **名称** | 查看项目下所有模型及其配置的功能测试 | | |
| **简要描述** | 本测试主要用于测试用户是否能够查看某一项目下所有的模型文件，并查看模型相关的配置信息、参数信息、版本号等 | | |
| **前提和约束** | 安装好Flask及系统运行环境 | | |
| **测试方法** | 黑盒测试 | | |
| **测试步骤** | **1** | 在指定的项目页面中，选择需要查看的模型，点击查看详情按钮 | |
| **2** | 跳转到指定模型的详细信息页面，可以查看其所有的基本信息，包括配置信息、参数信息、版本号等 | |
| **预期结果** | 跳转到当前指定模型信息页面，显示所有当前模型的信息，包括配置信息等，可以下载对应上传的文件 | | |
| **异常情况** | 溢出、通讯失败的服务器错误 | | 配置失败，提示“服务器错误，请稍后” |
| 用户处于非法登录状态 | | 配置失败，提示“用户未登录” |
| **测试结果** | 各步骤的测试结果与预期结果相同 | | |
| **测试结论** | 该功能通过测试 | | |
| **测试人员** | 许京爽 | | |
| **测试时间** | 2020/5/16 | | |

**表29 用例111删除项目下模型的功能测试**

|  |  |  |  |
| --- | --- | --- | --- |
| **名称** | 删除项目下模型的功能测试 | | |
| **简要描述** | 本测试主要用于测试用户是否能够删除项目中指定模型 | | |
| **前提和约束** | 安装好Flask及系统运行环境 | | |
| **测试方法** | 黑盒测试 | | |
| **测试步骤** | **1** | 在指定的项目页面中，选择需要删除的模型，点击删除模型按钮 | |
| **2** | 确认是否要删除的模型信息，点击确定 | |
| **预期结果** | 当前项目页面下的模型列表进行更新，删除的模型进行去处 | | |
| **异常情况** | 溢出、通讯失败的服务器错误 | | 配置失败，提示“服务器错误，请稍后” |
| 用户处于非法登录状态 | | 配置失败，提示“用户未登录” |
| **测试结果** | 各步骤的测试结果与预期结果相同 | | |
| **测试结论** | 该功能通过测试 | | |
| **测试人员** | 许京爽 | | |
| **测试时间** | 2020/5/16 | | |

**表30 用例112 启动实例功能测试**

|  |  |  |  |
| --- | --- | --- | --- |
| **名称** | 启动实例功能测试 | | |
| **简要描述** | 本测试主要用于测试在索引到文件所在服务器位置后使用对应的机器学习运行时是否能够加载该模型并使其常驻端口，新建后台运行的Flask程序用于响应用户发出的模型使用请求，并成为一个具体的实例 | | |
| **前提和约束** | 安装好Flask及系统运行环境 | | |
| **测试方法** | 黑盒测试 | | |
| **测试步骤** | **1** | 在指定的项目页面中，选择需要启动实例的模型，点击查看详情按钮 | |
| **2** | 跳转到当前模型基本信息页面，在部署信息中，点击启动实例 | |
| **预期结果** | 实例启动完毕，页面返回给用户生成的URL，可以复制链接 | | |
| **异常情况** | 文件夹地址错误 | | 启动实例失败，后端接收到错误码4031 |
| 文件地址错误 | | 启动实例失败，后端接受到错误码4032 |
| 配置项错误 | | 启动实例失败，后端接收到错误码4033 |
| 初始化Flask程序错误 | | 启动实例失败，后端接收到错误码4034 |
| 无多余端口号 | | 启动实例失败，后端接收到错误码4035 |
| **测试结果** | 各步骤的测试结果与预期结果相同 | | |
| **测试结论** | 该功能通过测试 | | |
| **测试人员** | 许京爽 | | |
| **测试时间** | 2020/5/16 | | |

**表31 用例113 删除实例功能测试**

|  |  |  |  |
| --- | --- | --- | --- |
| **名称** | 删除实例功能测试 | | |
| **简要描述** | 本测试主要用于测试是否能够删除已启动或暂停中的实例 | | |
| **前提和约束** | 安装好Flask及系统运行环境 | | |
| **测试方法** | 黑盒测试 | | |
| **测试步骤** | **1** | 在指定的项目页面中，选择需要删除实例的模型，点击查看详情按钮 | |
| **2** | 跳转到当前模型基本信息页面，在部署信息中，点击删除实例 | |
| **3** | 检查各项参数是否已经被清空 | |
| **4** | 重新启动一个实例，检查上一个实例的配置是否有遗留 | |
| **预期结果** | 在用户没有对参数修改的情况下，再次点击“启动实例”，应当按照系统的默认参数运行实例 | | |
| **异常情况** | 不存在该实例 | | 删除实例失败，后端接收到错误码4041 |
| 当前实例正在使用，无法删除 | | 删除实例失败，后端接受到错误码4042 |
| 实例进程删除失败 | | 删除实例失败，后端接受到错误码4043 |
| **测试结果** | 在用户没有对参数修改的情况下，再次点击“启动实例”，按照用户上一次设置的参数运行了实例，此时用户并不知情 | | |
| **测试结论** | 该功能未通过测试 | | |
| **测试人员** | 许京爽 | | |
| **测试时间** | 2020/5/17 | | |

**表32 用例114 暂停实例功能测试**

|  |  |  |  |
| --- | --- | --- | --- |
| **名称** | 暂停实例功能测试 | | |
| **简要描述** | 本测试主要用于测试是否能够对运行中的模型部署实例进行暂停服务的功能 | | |
| **前提和约束** | 安装好Flask及系统运行环境 | | |
| **测试方法** | 黑盒测试 | | |
| **测试步骤** | **1** | 在指定的项目页面中，选择需要暂停实例的模型，点击查看详情按钮 | |
| **2** | 跳转到当前模型基本信息页面，在部署信息中，点击暂停实例 | |
| **预期结果** | 页面对于实例状态显示为，已部署，暂停中 | | |
| **异常情况** | 数据库中不存在该实例或者当前实例已处于暂停状态 | | 暂停实例失败 |
| **测试结果** | 各步骤的测试结果与预期结果相同 | | |
| **测试结论** | 该功能通过测试 | | |
| **测试人员** | 许京爽 | | |
| **测试时间** | 2020/5/17 | | |

**表33 用例115 恢复实例功能测试**

|  |  |  |  |
| --- | --- | --- | --- |
| **名称** | 恢复实例功能测试 | | |
| **简要描述** | 本测试主要用于测试是否能够对暂停中的实例进行恢复服务的功能 | | |
| **前提和约束** | 安装好Flask及系统运行环境 | | |
| **测试方法** | 黑盒测试 | | |
| **测试步骤** | **1** | 在指定的项目页面中，选择需要恢复实例的模型，点击查看详情按钮 | |
| **2** | 跳转到当前模型基本信息页面，在部署信息中，点击恢复实例 | |
| **预期结果** | 页面对于实例状态显示为，已部署，运行中 | | |
| **异常情况** | 数据库中不存在该实例或者当前实例已处于运行中状态 | | 恢复实例失败 |
| **测试结果** | 各步骤的测试结果与预期结果相同 | | |
| **测试结论** | 该功能通过测试 | | |
| **测试人员** | 许京爽 | | |
| **测试时间** | 2020/5/17 | | |

# 4 性能需求测试结果

**表34 用例201 运行模型实例的常规性能测试**

|  |  |  |
| --- | --- | --- |
| **名称** | 运行模型实例的常规性能测试 | |
| **简要描述** | 本用例主要用于测试系统在运行小型模型实例时是否能够满足SRS的性能需求 | |
| **前提和约束** | 服务器系统正常工作，客户端处于平均网络环境中，准备好若干训练好的深度学习模型文件，每个模型文件在运行时的内存占用不超过500 MB | |
| **测试方法** | 黑盒测试 | |
| **测试步骤** | **1** | 在项目中新建项目，导入前述深度学习模型，并启动实例 |
| **2** | 使用Postman向API发送图片预测请求 |
| **3** | 查看Postman中的返回数据报的Time属性，该属性为API的响应时间 |
| **预期结果** | 所有的测试模型实例的响应时间均小于300毫秒 | |
| **评价准则** | 响应时间小于300毫秒的测试用例占总测试用例的比例 | |
| **测试结果** | 所有的测试模型实例的响应时间均小于300毫秒 | |
| **测试结论** | 该用例通过测试 | |
| **测试人员** | 聂磊 | |
| **测试时间** | 2020/5/18 | |

**表35 用例202 运行模型实例的负载性能测试**

|  |  |  |
| --- | --- | --- |
| **名称** | 运行模型实例的负载性能测试 | |
| **简要描述** | 本用例主要用于测试系统在运行中型模型实例时是否能够满足SRS的性能需求 | |
| **前提和约束** | 服务器系统正常工作，客户端处于平均网络环境中，准备好若干训练好的深度学习模型文件，每个模型文件在运行时的内存占用不超过2 GB | |
| **测试方法** | 黑盒测试 | |
| **测试步骤** | **1** | 在项目中新建项目，导入前述深度学习模型，并启动实例 |
| **2** | 使用Postman向API发送图片预测请求 |
| **3** | 查看Postman中的返回数据报的Time属性，该属性为API的响应时间 |
| **预期结果** | 所有的测试模型实例的响应时间均小于1秒 | |
| **评价准则** | 响应时间小于1秒的测试用例占总测试用例的比例 | |
| **测试结果** | 所有的测试模型实例的响应时间均小于1秒 | |
| **测试结论** | 该用例通过测试 | |
| **测试人员** | 聂磊 | |
| **测试时间** | 2020/5/18 | |

**表36 用例203 运行模型实例的压力情况测试**

|  |  |  |
| --- | --- | --- |
| **名称** | 运行模型实例的压力情况测试 | |
| **简要描述** | 本用例主要用于测试系统在运行大型模型实例时是否能够满足SRS的性能需求 | |
| **前提和约束** | 服务器系统正常工作，客户端处于平均网络环境中，准备好若干训练好的深度学习模型文件，每个模型文件在运行时的内存占用均大于2 GB | |
| **测试方法** | 黑盒测试 | |
| **测试步骤与预期结果** | **1** | 在项目中新建项目，导入前述深度学习模型，并尝试启动实例 |
| **2** | 系统阻止该模型运行，浏览器返回消息“部署失败” |
| **3** | 再次导入用例202的深度学习模型，尝试启动实例 |
| **4** | 实例启动成功，使用Postman向API发送图片预测请求，可以得到与用例202相同的响应 |
| **评价准则** | 系统能在模型实例占用内存大于2 GB时阻止模型实例运行，以保障系统稳定性 | |
| **测试结果** | 各步骤的测试结果与预期结果相同 | |
| **测试结论** | 该用例通过测试 | |
| **测试人员** | 聂磊 | |
| **测试时间** | 2020/5/18 | |

**表37 用例204 网站界面并发访问的常规性能测试**

|  |  |  |
| --- | --- | --- |
| **名称** | 网站界面并发访问的常规性能测试 | |
| **简要描述** | 本用例主要用于测试系统是否能够满足SRS中关于网站界面并发访问的性能需求 | |
| **前提和约束** | 服务器系统正常工作，客户端安装http\_load | |
| **测试方法** | 黑盒测试 | |
| **测试步骤** | **1** | 新建urls.txt文件，在文件中保存本系统网页前端的访问URL |
| **2** | 运行http\_load -fetches 30 -seconds 60 urls.txt |
| **3** | 在运行压力测试期间，从浏览器访问本系统，尝试新建、修改、查看项目，尝试新建、修改、查看模型，并尝试运行与暂停模型实例 |
| **预期结果** | 所有访问请求能得到返回码200，即服务器正常响应，同时在浏览器中所做的操作能在200毫秒内得到响应 | |
| **评价准则** | 响应时间小于200毫秒的访问请求数量占总的访问请求数量的百分比，在浏览器中执行操作的结果 | |
| **测试结果** | 所有访问请求能得到返回码200，同时在浏览器中所做的操作能在200毫秒内得到响应 | |
| **测试结论** | 该用例通过测试 | |
| **测试人员** | 聂磊 | |
| **测试时间** | 2020/5/18 | |

**表38 用例205 网站界面并发访问的负载性能测试**

|  |  |  |
| --- | --- | --- |
| **名称** | 网站界面并发访问的负载性能测试 | |
| **简要描述** | 本用例主要用于测试系统是否能够满足SRS中关于网站界面并发访问的性能需求 | |
| **前提和约束** | 服务器系统正常工作，客户端安装http\_load | |
| **测试方法** | 黑盒测试 | |
| **测试步骤** | **1** | 新建urls.txt文件，在文件中保存本系统网页前端的访问URL |
| **2** | 运行http\_load -fetches 100 -seconds 60 urls.txt |
| **3** | 在运行压力测试期间，从浏览器访问本系统，尝试新建、修改、查看项目，尝试新建、修改、查看模型，并尝试运行与暂停模型实例 |
| **预期结果** | 所有访问请求能得到返回码200，即服务器正常响应，同时在浏览器中所做的操作能在800毫秒内得到响应 | |
| **评价准则** | 响应时间小于800毫秒的访问请求数量占总的访问请求数量的百分比，在浏览器中执行操作的结果 | |
| **测试结果** | 所有访问请求能得到返回码200，同时在浏览器中所做的操作能在800毫秒内得到响应 | |
| **测试结论** | 该用例通过测试 | |
| **测试人员** | 聂磊 | |
| **测试时间** | 2020/5/18 | |

**表39 用例206 网站界面并发访问的压力性能测试**

|  |  |  |
| --- | --- | --- |
| **名称** | 网站界面并发访问的压力性能测试 | |
| **简要描述** | 本用例主要用于测试系统是否能够满足SRS中关于网站界面并发访问的性能需求 | |
| **前提和约束** | 服务器系统正常工作，客户端安装http\_load | |
| **测试方法** | 黑盒测试 | |
| **测试步骤** | **1** | 新建urls.txt文件，在文件中保存本系统网页前端的访问URL |
| **2** | 运行http\_load -fetches 200 -seconds 60 urls.txt |
| **3** | 在运行压力测试期间，从浏览器访问本系统，尝试新建、修改、查看项目，尝试新建、修改、查看模型，并尝试运行与暂停模型实例 |
| **预期结果** | 90%的访问请求能得到返回码200，即服务器正常响应，同时在浏览器中所做的操作能在3秒内得到响应 | |
| **评价准则** | 被正常响应的访问请求数量占总的访问请求数量的百分比，在浏览器中执行操作的结果 | |
| **测试结果** | 100%的访问请求能得到返回码200，同时在浏览器中所做的操作能在3秒内得到响应 | |
| **测试结论** | 该用例通过测试 | |
| **测试人员** | 聂磊 | |
| **测试时间** | 2020/5/18 | |

# 5 接口需求测试结果

**表40 用例301 REST API测试**

|  |  |  |
| --- | --- | --- |
| **名称** | REST API测试 | |
| **简要描述** | 本用例主要用于测试系统是否能够满足SRS中关于REST API的用户接口需求 | |
| **前提和约束** | 服务器系统正常工作，已经完成模型实例的启动并获得REST API访问URL | |
| **测试方法** | 黑盒测试 | |
| **测试步骤** | **1** | 启动Postman，在目标URL中输入模型实例的访问地址 |
| **2** | 编辑数据报的字段，在data字段输入待检测图片的灰度矩阵，在key字段输入浏览器显示的模型实例的key字符串 |
| **3** | 点击发送按钮，稍等片刻，接收REST API返回的预测结果 |
| **预期结果** | 返回数据报的data字段与对应模型的预期运行结果相符，status字段为‘success’ | |
| **测试结果** | 各步骤的测试结果与预期结果相同 | |
| **测试结论** | 该用例通过测试 | |
| **测试人员** | 聂磊 | |
| **测试时间** | 2020/5/18 | |

**表41 用例302 软件接口测试**

|  |  |  |
| --- | --- | --- |
| **名称** | 软件接口测试 | |
| **简要描述** | 本用例主要用于测试系统是否能够满足SRS中关于MySQL数据库的软件接口需求 | |
| **前提和约束** | 服务器系统正常工作，MySQL已启动并与系统建立连接 | |
| **测试方法** | 黑盒测试 | |
| **测试步骤** | **1** | 在MySQL中启动查询日志的记录选项 |
| **2** | 从浏览器访问本系统，尝试注册与登录账户，尝试新建、修改、查看项目，尝试新建、修改、查看模型，并尝试运行与暂停模型实例 |
| **3** | 保存查询日志文件，在MySQL中关闭查询日志的记录选项 |
| **预期结果** | 查询日志文件中记录的SQL查询语句与用户的操作行为相符，数据库条目与用户的操作行为相符 | |
| **测试结果** | 各步骤的测试结果与预期结果相同 | |
| **测试结论** | 该用例通过测试 | |
| **测试人员** | 聂磊 | |
| **测试时间** | 2020/5/18 | |

# 6 软件质量特征测试结果

**表42 用例401 用户接口兼容性测试**

|  |  |  |
| --- | --- | --- |
| **名称** | 用户接口兼容性测试 | |
| **简要描述** | 本用例主要用于测试系统是否能够满足SRS中关于用户接口兼容性的要求 | |
| **前提和约束** | 服务器系统正常工作 | |
| **测试方法** | 黑盒测试 | |
| **测试步骤** | **1** | 通过Firefox浏览器访问本系统，尝试注册与登录账户，尝试新建、修改、查看项目，尝试新建、修改、查看模型，并尝试运行与暂停模型实例。 |
| **2** | 通过Chrome浏览器访问本系统，尝试注册与登录账户，尝试新建、修改、查看项目，尝试新建、修改、查看模型，并尝试运行与暂停模型实例。 |
| **3** | 通过Microsoft Edge浏览器访问本系统，尝试注册与登录账户，尝试新建、修改、查看项目，尝试新建、修改、查看模型，并尝试运行与暂停模型实例。 |
| **预期结果** | 使用各个浏览器访问本系统的过程中，各类操作均能正常完成 | |
| **评价准则** | 正常完成的操作数量占总的操作数量的百分比 | |
| **测试结果** | 各步骤的测试结果与预期结果相同 | |
| **测试结论** | 该用例通过测试 | |
| **测试人员** | 聂磊 | |
| **测试时间** | 2020/5/18 | |

**表43 用例402 特殊符号容错性测试**

|  |  |  |
| --- | --- | --- |
| **名称** | 特殊符号容错性测试 | |
| **简要描述** | 本用例主要用于测试系统是否能够满足SRS中关于系统容错性的要求 | |
| **前提和约束** | 服务器系统正常工作 | |
| **测试方法** | 黑盒测试 | |
| **测试步骤** | **1** | 分别在以下区域输入的特殊符号：a)注册时的用户名和密码文本框中，b)登录时的用户名和密码文本框中，c)管理项目时的项目名称和项目简介文本框中 |
| **2** | 观察提交包含特殊符号的字符串后是否会引发系统异常 |
| **预期结果** | 对于包含特殊符号的字符串，系统能不受这些字符串影响，继续正常工作 | |
| **评价准则** | 系统因为异常输入而崩溃的次数占异常输入测试次数的比例 | |
| **测试结果** | 系统能未受到包含特殊符号的字符串影响，继续正常工作 | |
| **测试结论** | 该用例通过测试 | |
| **测试人员** | 聂磊 | |
| **测试时间** | 2020/5/18 | |

**表44 用例403 超长字符串容错性测试**

|  |  |  |
| --- | --- | --- |
| **名称** | 超长字符串容错性测试 | |
| **简要描述** | 本用例主要用于测试系统是否能够满足SRS中关于系统容错性的要求 | |
| **前提和约束** | 服务器系统正常工作 | |
| **测试方法** | 黑盒测试 | |
| **测试步骤** | **1** | 分别在以下区域输入的随机构造的长度大于1000的字符串：a)注册时的用户名和密码文本框中，b)登录时的用户名和密码文本框中，c)管理项目时的项目名称和项目简介文本框中 |
| **2** | 观察提交超长字符串后是否会引发系统异常，系统是否返回错误提示 |
| **预期结果** | 对于超长字符串，系统能返回错误提示，并继续正常工作 | |
| **评价准则** | 系统因为异常输入而崩溃的次数占异常输入测试次数的比例 | |
| **测试结果** | 系统未受到提交的超长字符串的影响，继续正常工作 | |
| **测试结论** | 该用例通过测试 | |
| **测试人员** | 聂磊 | |
| **测试时间** | 2020/5/18 | |

**表45 用例404 易用性测试**

|  |  |  |
| --- | --- | --- |
| **名称** | 易用性测试 | |
| **简要描述** | 本用例主要用于测试系统是否能够满足SRS中关于网站界面易用性的要求 | |
| **前提和约束** | 服务器系统正常工作 | |
| **测试方法** | 黑盒测试 | |
| **测试步骤** | **1** | 打开网站界面，尝试以不同比例缩放页面，并在保持缩放状态的情况下点击各个功能 |
| **2** | 观察页面元素是否有重叠和乱序，各个按钮和文本框是否能正常工作 |
| **预期结果** | 网站页面内容无重叠，无乱序，各个按钮和文本框能正常工作 | |
| **评价准则** | 系统缩放时出现显示异常的页面元素的数量 | |
| **测试结果** | 在50%，100%，150%，200%的缩放比例下，网站页面内容无重叠，无乱序，各个按钮和文本框能正常工作 | |
| **测试结论** | 该用例通过测试 | |
| **测试人员** | 聂磊 | |
| **测试时间** | 2020/5/18 | |

**表46 用例405 SQL注入攻击测试**

|  |  |  |
| --- | --- | --- |
| **名称** | SQL注入攻击测试 | |
| **简要描述** | 本用例主要用于测试系统抵御SQL注入攻击的能力 | |
| **前提和约束** | 服务器系统正常工作 | |
| **测试方法** | 黑盒测试 | |
| **测试步骤** | **1** | 构造包含各种SQL注入攻击信息的访问请求 |
| **2** | 分别在以下区域输入包含攻击信息的字符串：a)注册时的用户名和密码文本框中，b)登录时的用户名和密码文本框中，c)管理项目时的项目名称和项目简介文本框中 |
| **3** | 检查构造的SQL注入攻击是否成功 |
| **预期结果** | SQL注入攻击失败，网页提示非法字符串，或是直接忽略非法SQL语句 | |
| **评价准则** | SQL注入攻击失败的次数占总尝试次数的比例 | |
| **测试结果** | 构造的SQL注入攻击方式均失败，构造的特殊字符串被服务器忽略，无法获取数据库信息或绕开登录检查 | |
| **测试结论** | 该用例通过测试 | |
| **测试人员** | 聂磊 | |
| **测试时间** | 2020/5/18 | |

# 7 总结

本文档描述了C组开发项目 ***基于Flask的深度学习自动化部署系统*** 的测试结果，在46个测试用例中，软件通过了44个测试用例，没有通过2个测试用例。本组将在软件错误修复报告中详细介绍对2个未通过的测试用例的处理过程。

# 参考文献

1. 邹欣.构建之法[M].北京.人民邮电出版社,2015:264-296
2. 软件工程测试方法[EB/OL]. <https://blog.csdn.net/fightfaith/article/details/47858101>