**用户注册实习项目概要设计**

**云平台**

*TP-LINK TECHNOLOGIES CO., LTD*

[*http://www.tp-link.com.cn*](http://www.tp-link.com.cn)

*South Building, No.5 Keyuan Road, Central Zone*

*Science & Technology Park, NanShan*

*ShenZhen, P. R. China*

**版本历史**

|  |  |  |  |
| --- | --- | --- | --- |
| 版本/状态 | 作者 | 起止日期 | 描述 |
| V0.1/初稿 | 蒋琴 | 2017-07-25 | 完成初稿 |
| V0.2/修改稿 | 蒋琴 | 2017-07-26 | 按照组内修改意见修改 |

# 摘要

本文档是用户注册实习项目的设计文档，主要对用户注册项目中的主要流程和接口做了详细描述。主要内容包括接口设计、功能模块设计、缓存、数据库设计等，为后续编码开发以及项目管理提供指导和帮助

**关键字： 用户注册实习项目，流程，接口，概要设计**

**目录**

[摘要 I](#_Toc488758799)

[1 引言 1](#_Toc488758800)

[1.1 目的 1](#_Toc488758801)

[1.2 背景 1](#_Toc488758802)

[1.3 参考资料 1](#_Toc488758803)

[1.4 术语、缩略语及缩写词 1](#_Toc488758804)

[1.5 需求概述 1](#_Toc488758805)

[2 整体设计 2](#_Toc488758806)

[2.1 运行环境 2](#_Toc488758807)

[2.2 开发环境 2](#_Toc488758808)

[2.3 约束限制 2](#_Toc488758809)

[2.4 系统依赖关系 3](#_Toc488758810)

[2.5 系统设计 3](#_Toc488758811)

[3 接口设计 3](#_Toc488758812)

[3.1 新建用户 3](#_Toc488758813)

[3.1.1 接口流程 5](#_Toc488758814)

[3.2 获取用户信息 5](#_Toc488758815)

[3.2.1 接口流程 7](#_Toc488758816)

[3.3 更新用户信息 8](#_Toc488758817)

[3.3.1 接口流程 8](#_Toc488758818)

[3.4 请求认证获取token 10](#_Toc488758819)

[3.4.1 接口流程 11](#_Toc488758820)

[3.5 删除用户token 11](#_Toc488758821)

[3.5.1 接口流程 12](#_Toc488758822)

[4 数据库设计 12](#_Toc488758823)

[4.1 设计依据 12](#_Toc488758824)

[4.2 数据库选型 12](#_Toc488758825)

[4.3 数据库结构 12](#_Toc488758826)

[4.3.1 实体 12](#_Toc488758827)

[4.3.2 表 13](#_Toc488758828)

[4.3.2.1 用户表 13](#_Toc488758829)

# 引言

## 目的

本文档是用户注册项目的设计文档，文档撰写旨在为后续编码开发以及项目管理提供指导和帮助。本项目为2017年秋季刚入职同事的练习项目。读者范围为云平台全体同事。

## 背景

本文档是用户注册项目的设计文档，用于软件设计阶段的概要设计，是对模块的解决方案的详细说明。主要内容包括接口设计、功能模块设计等。

用户注册项目是可以为前端提供用户注册，认证，用户信息更改等服务，通过rest api供其它组件调用。

## 参考资料

* [AppServer概要设计\_V1.0.0](file:///\\file.tp-link.net\云平台部\new_file_server_old\项目文档评审归档\AppServer模块\AppServer概要设计_V1.0.0.doc)
* [内销安防设备数据分析系统概要设计](file:///\\file.tp-link.net\云平台部\1_模块开发\对外项目\安防设备数据分析内销\内销安防设备数据分析系统概要设计.docx)

## 术语、缩略语及缩写词

Rest api：REST API 可以让你用任何支持发送 HTTP 请求的设备来与之进行交互。REST指一组架构约束条件和原则，满足约束条件和原则的应用程序设计。Rest api有以下几个特点：

1、每个资源都应该有唯一的一个标识

2、使用标准的方法更改资源的状态

3、request和response的自描述

4、资源多重表述

5、无状态服务

## 需求概述

需求分析文档如下：



# 整体设计

## 运行环境

关键软件平台的版本见表3-1：

表3- 1 关键软件版本

|  |  |  |  |
| --- | --- | --- | --- |
| **类别** | **软件** | **版本** | **说明** |
| 操作系统 | Ubuntu | 14.04 |  |
| Java虚拟机 | Oracle Hotspot VM | JDK-1.7.0\_60 |  |
| Web服务器 | Tomcat | 8.5.4 | Web容器 |

## 开发环境

本系统的核心平台是JavaEE（Java Platform Enterprise Edition）。

表3- 1 关键软件版本

|  |  |  |  |
| --- | --- | --- | --- |
| **类别** | **软件** | **版本** | **说明** |
| 操作系统 | Windows | 8 |  |
| 数据库 | MySQL | 5.6.25 |  |
| Java虚拟机 | Oracle Hotspot VM | JDK-1.7.0\_60 |  |
| Java框架 | Spring | 4.3.0 | 轻量级控制反转（IoC）和面向切面（AOP）的容器框架，能最大程度使业务组件类解耦，并对组件类实施依赖注入 |
| SpringMVC | 4.2.6 | 基于MVC模式的Web开发框架 |
| Hibernate | 4.3.5 | ORM（Object-Relation-Mapping，对象关系映射）框架，完成Java实体类到关系型数据库表的映射，使开发者能够以面向对象的方式处理关系型数据。 |
| Web服务器 | Tomcat | 8.5.4 | Web容器 |
| IDE | IntelliJ IDEA | 2017.1.2 |  |

## 约束限制

无

## 系统依赖关系



图2.1系统图

## 系统设计



图4-2 软件架构分层模型

# 接口设计

## 新建用户

### 接口定义

表3-1 新建用户

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 内容 | | |
| 请求方法 | POST | | |
| 请求参数 | body  json | email  string | 用户注册邮箱  （必须） |
| displayName  string | 用户名（必须） |
| age  int | 年龄（可选） |
| gender  string | 性别（可选） |
| password  string | 密码（必须） |
| 响应内容 | body  json | id  int | 用户id |
| email  string | 用户注册邮箱 |
| displayName  string | 用户显示名 |
| age  int | 年龄 |
| gender  string | 性别 |
| Error Code | 400 | 请求非法  （或缺少参数） |
| 409 | 该用户已存在 |
| 500 | 服务器异常 |
| 使用限制 | 无 | | |
| 功能描述 | 新建用户 | | |

### 请求示例

Request：

|  |
| --- |
| POST /user HTTP/1.1  Content-Type: application/json  Accept: application/json  {  "email": "xxxxx"，  "password": "xxxxx",  "displayName": "xxxxx"  } |

Response:

|  |
| --- |
| HTTP/1.1 200 OK Content-Type: application/json  {  "id": 0,  "email": "xxxxx",  "displayName": "xxxxx",  "age": 0,  "gender": "xxxxx"  } |

### 接口流程



图3-1 新建用户流程图

## 获取用户信息

### 接口定义

表3-2 获取用户信息

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 内容 | | |
| 请求方法 | GET | | |
| 请求参数 | path | id  string | 用户id |
| header | X-Token  string | token信息 |
| 响应内容 | body  json | id  int | 用户id |
| email  string | 用户注册邮箱 |
| displayName  string | 用户显示名 |
| age  int | 年龄 |
| gender  string | 性别 |
| Error Code | 401 | token非法  （过期或不存在） |
| 403 | 无权限请求该id下的信息 |
| 500 | 服务器异常 |
| 使用限制 | 需要登录 | | |
| 功能描述 | 获取用户信息 | | |

### 请求示例

Request：

|  |
| --- |
| GET /user/{id} HTTP/1.1  Content-Type: application/json  Accept: application/json  X-Token: XXX |

Response：

|  |
| --- |
| HTTP/1.1 200 OK Content-Type: application/json  {  "id": 0,  "email": "xxxxx",  "displayName": "xxxxx",  "age": 0,  "gender": "xxxxx"  } |

### 接口流程



图3-2 获取用户信息流程图

## 更新用户信息

### 接口定义

表3-3 更新用户信息

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 内容 | | |
| 请求方法 | PUT | | |
| 请求参数 | body  json | displayName  string | 用户显示名  （可选） |
| age  int | 年龄（可选） |
| gender  string | 性别（可选） |
| path | id  string | 用户id（必须） |
| header | X-Token  string | token信息  （必须） |
| 响应内容 | body  json | id  int | 用户id |
| email  string | 用户注册邮箱 |
| displayName  string | 用户显示名 |
| age  int | 年龄 |
| gender  string | 性别 |
| Error Code | 401 | token非法  （过期或不存在） |
| 403 | 无权限请求该id下的信息 |
| 500 | 服务器异常 |
| 使用限制 | 需要登录 | | |
| 功能描述 | 更新用户信息 | | |

### 请求示例

Request：

|  |
| --- |
| PUT /user HTTP/1.1  Content-Type: application/json  Accept: application/json  X-Token：XXX  {  "displayName": "xxxxx",  "age": "xxxxx",  "gender": "xxxxx"  } |

Response：

|  |
| --- |
| HTTP/1.1 200 OK Content-Type: application/json  {  "id": 0,  "email": "xxxxx",  "displayName": "xxxxx",  "age": 0,  "gender": "xxxxx"  } |

### 接口流程



图3-3 更新用户信息流程图

## 请求认证获取token

### 接口定义

表3-4 请求认证

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 内容 | | |
| 请求方法 | POST | | |
| 请求参数 | body  json | email  string | 用户登录邮箱  （必须） |
| password  int | 密码（必须） |
| 响应内容 | body  json | id  int | 用户id |
| token  string | 用户token |
| expiresin  int | 是否过期 |
| Error Code | 401 | 用户名或密码错误（也包含该用户不存在的情况） |
| 500 | 服务器异常 |
| 使用限制 | 无 | | |
| 功能描述 | 获取token | | |

### 请求示例

Request：

|  |
| --- |
| POST /token HTTP/1.1  Content-Type: application/json  Accept: application/json  {  "email": "xxxxx",  "password": "xxxxx"  } |

Response:

|  |
| --- |
| HTTP/1.1 200 OK Content-Type: application/json  {  "id": 0,  "token": "xxxxx",  "expiresIn": 0  } |

### 接口流程



图3-4 获取token流程图

## 删除用户token

### 接口定义

表3-5 删除用户token

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 内容 | | |
| 请求方法 | DELETE | | |
| 请求参数 | path | token  string | token（必须） |
| 响应内容 | Response Code | 200 | 成功删除token |
| Error Code | 401 | 用户名或密码错误（也包含该用户不存在的情况） |
| 使用限制 | 无 | | |
| 功能描述 | 删除token | | |

### 请求示例

Request:

|  |
| --- |
| DELETE /token/{token} HTTP/1.1  Content-Type: application/json  Accept: application/json |

Response:

|  |
| --- |
| HTTP/1.1 200 OK |

### 接口流程



图3-5删除token流程图

# 数据库设计

## 设计依据

作为实习训练项目，数据存储量不会很多，所以这里选用mysql存储数据。

## 数据库选型

本系统选用mysql存储数据，mysql对数据库选型进行确认。Mysql体积小、速度快、总体拥有成本低，开源；支持多种操作系统；是开源数据库，提供的接口支持多种语言连接操作。

## 数据库结构

### 实体

数据库中只存储了一个用户信息表。

### 表

#### 用户信息表

表 4-1 user

|  |  |  |
| --- | --- | --- |
| **表名：jiangqin\_user** | | |
| **列名** | **类型** | **描述** |
| id | int | 主键 |
| email | string | 邮箱 |
| displayName | varchar(255) | 显示名 |
| age | int | 年龄 |
| gender | varchar(255) | 性别 |
| password | varchar(255) | 密码的hash编码 |