# 项目需求

通过登录与注册的页面，实现可以登录和注册

# 项目需求分析

定义一个脚本：通过引用，flask框架，定义两个函数，分别为登录与注册

定义登录的html，向后端传输username和password两个参数

定义注册的html，想后端传输username,new\_password,confirm\_password 三个参数

## 1、登录

定义一个登录的html，html中定义一个表单，用户在表单中输入：用户名和密码

Python脚本中定义一个login的函数，先从user.txt文件读取每行，将用户名放入一个列表，将用户名和密码对应存放在一个字典

Longin从前端html获取到用户名和密码，先判断用户名是否在列表中，判断用户名是否正确，如果正确，再通过字典，去判断密码是否正确，正确则提示登录成功

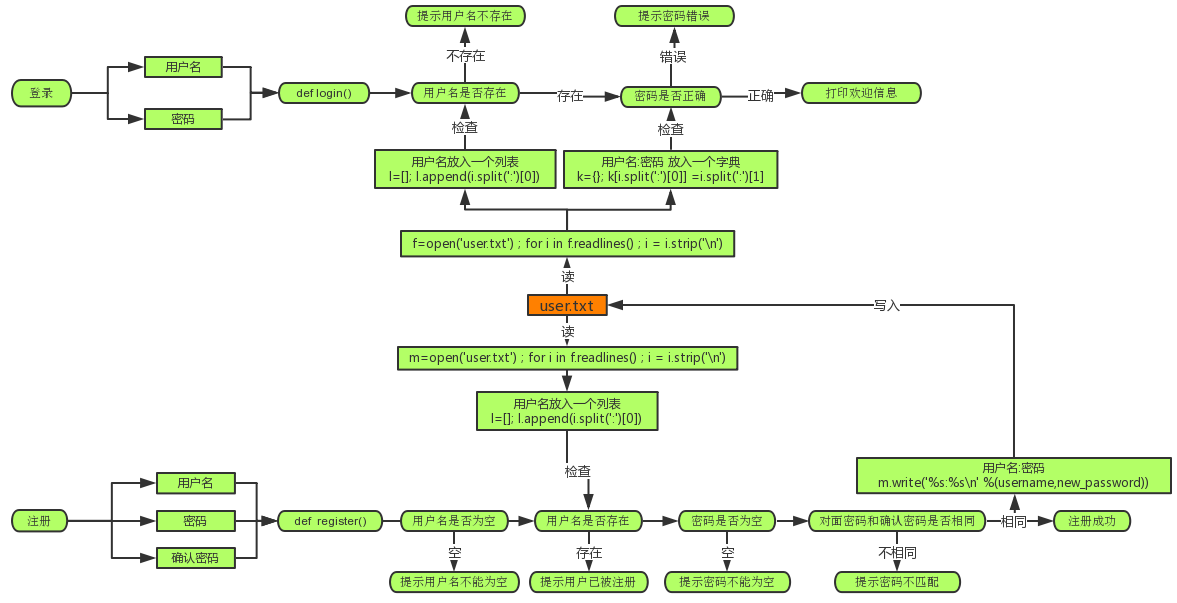
## 2、注册

定义一个注册的html，html中定义一个表单，用户在表单中输入：用户名、密码和确认密码

Python脚本中定义一个register的函数，先从user.txt文件读取每行，将用户名放在一个列表

Register从前端html获取到用户名、密码和确认密码，先判断用户名是否为空，然后判断用户名是否在列表里，如果在则提示已被注册，然后输入的密码是否为空，然后判断两次输入密码是否相同，如果两次密码也想通，则提示注册成功，并将用户名:密码，写入user.txt

# 项目总体设计（思维导图）



# 项目各模块划分

## 整理目录结构

├── login\_register.py 登录与注册脚本，定义登录与 注册的函数

├── templates templates目录，存放html

│   ├── login.html 登录的html

│   └── register.html 注册html

└── user.txt 用户信息表，存放用户名与密码

## 函数引用flask架构

#/usr/bin/python

#coding:utf-8

from flask import Flask,request,render\_template

app = Flask(\_\_name\_\_)

if \_\_name\_\_ == '\_\_main\_\_':

app.run(host='0.0.0.0',port=80,debug=True)

## 登录

### 1 ) Html

定义表单，方法为post, 获取两个参数，username和password

<form role="form" data-parsley-validate novalidate class="mb-lg login-form" method="post">

**<div class="form-group has-feedback">**

**<input class="form-control" type="text"**

**name="username"**

**placeholder="用户名"**

**data-parsley-error-message="请输入用户名"**

**required>**

**</div>**

**<div class="form-group has-feedback form-group-top">**

**<input class="form-control login-input" type="password"**

**name="password"**

**placeholder="密码"**

**data-parsley-error-message="请输入密码"**

**required>**

**</div>**

### 2）函数

获取用户名和密码，并通过与user.txt获取到的用户名和密码进行对比，先检查用户名，在检查密码

@app.route('/login/',methods=["GET","POST"])

def login():

if request.method=='POST':

username = request.form.get('username')

password = request.form.get('password')

print username,password

f = open('user.txt')

l = []

k = {}

for i in f.readlines():

i =i.strip('\n')

a = i.split(':')[0]

b = i.split(':')[1]

k[a] = b

l.append(a)

if username not in l:

return "用户名不存在"

exit()

else:

if password == k.get(username):

return "欢迎%s登录" %username

else:

return "登录密码错误"

f.close()

return render\_template("login.html")

## 注册

### 1）html

<form role="form" data-parsley-validate novalidate class="mb-lg" method="post">

**<div class="form-group has-feedback ">**

**<input class="form-control" type=""**

**name="username"**

**placeholder="请输入用户名"**

**data-parsley-error-message="请输入用户名"**

**required>**

**</div>**

**<div class="form-group has-feedback register-group-top">**

**<input class="form-control" type="password"**

**name="new\_password"**

**placeholder="密码"**

**</div>**

**<div class="pwd\_v"></div>**

**<div class="form-group has-feedback register-group-top">**

**<input class="form-control" type="password"**

**name="confirm\_password"**

**placeholder="确认密码"**

**required>**

**</div>**

### 2）函数

从html中获取username,new\_password,confirm\_password三个参数，先判断用户名是否为空，然后先通过user.txt获取的user信息判断用户是否被注册，然后判断密码是否为空，最后判断两次密码是否匹配，如果匹配则提示注册成功

@app.route('/register/',methods=["GET","POST"])

def register():

if request.method=='POST':

username = request.form.get('username')

new\_password = request.form.get('new\_password')

confirm\_password = request.form.get('confirm\_password')

print username,new\_password,confirm\_password

m = open('user.txt','a+')

if len(username) == 0:

return "用户名不能为空"

exit()

l = []

for i in m.readlines():

a = i.split(':')[0]

l.append(a)

if username in l:

return "该账户已被注册"

exit()

if len(new\_password) == 0:

return "密码不能为空"

if confirm\_password == new\_password:

m.write('%s:%s\n' %(username,new\_password))

return "注册成功，您的账号为:%s,密码为:%s,请妥善保管" %(username,new\_password)

exit()

else:

return "密码不匹配"

m.close()

return render\_template("register.html")