web表单使用及登录页面实现

1. config配置文件创建及配置

为了实现web表单功能，就要使用到flask的扩展库`FLASK-WTF`。

```
pip install flask-wtf
```

各种flask扩展库都需要配置一些参数，所以创建一个配置文件`app/config.py`用于存放这些配置信息。

对于扩展库`FLASK-WTF`主要需要以下两个配置信息：

- CSRF_ENABLED = True：激活`跨站点请求伪造`保护；
- SECRET_KEY = 'you-will-never-guess'：CSRF被激活时，需要该参数用于令牌加密，并实现表单数据的验证。

```
# ----------FLASK-WTF扩展库配置--------- #
# 激活跨站点请求伪造保护
CSRF_ENABLED = True
# CSRF被激活时，用于令牌加密，表单验证
SECRET_KEY = 'you-will-never-guess'
```

注意：SECRET_KEY参数尽量使用较为复杂的密钥，可以使用`os.urandom(16)`的方式获取随机密钥。

2. config.py配置文件加载

将`app/config.py`配置文件中的内容加载到flask应用程序中，修改`app/__init__.py`文件。

```
import os
from flask import Flask
```

```
def create_app():
    """应用工厂函数"""
    app = Flask(__name__)
    # 加载config配置文件
    app.config.from_pyfile('config.py', silent=True)

    # 注册hello视图URL
    from app.hello import HelloWorld
    app.add_url_rule('/hello', view_func=HelloWorld.as_view('hello'))

    try:
        # 确保 app.instance_path 存在
        os.makedirs(app.instance_path)
    except OSError:
        pass

    return app
```

3. 创建用户登录表单

创建并编写用户登录表单文件`app/forms.py`：

```
from flask_wtf import FlaskForm
from wtforms import StringField, PasswordField, BooleanField, SubmitField
from wtforms.validators import DataRequired


class LoginForm(FlaskForm):
    # DataRequired: 数据不可为空的验证器
    username = StringField('username', validators=[DataRequired])
    password = PasswordField('password', validators=[DataRequired])
    remember_me = BooleanField('remember_me', default=False)
    submit = SubmitField('登录')
```

4. 编写表单模板

新建并编写表单模板文件`app/templates/login/login.html`。

- `<form>`元素，用作Web表单的容器；
- `action`属性，用于告知浏览器用户在提交表单中输入的信息时应使用的URL。当操作设置为空时，表单将提交到当前位于地址栏中的URL；
- `method`属性，指定在将表单提交到服务器时应使用的HTTP请求方法。
    - 默认情况下是通过`GET`请求发送它，但几乎在所有情况下，使用`POST`请求都可以获得更好的用户体验，因为此类请求可以在当前页面URL中提交表单数据；
    - 而`GET`请求会将表单字段添加到URL，使浏览器地址栏变得混乱。
- `novalidate`属性，用于告知Web浏览器不对此表单中的字段应用验证，验证将留给服务器处理。使用`novalidate`完全是可选的，但是对于这个表单，设置它是很重要的，因为这会允许服务器端的验证；

- `form.hidden_tag()`模板参数，生成一个隐藏字段，其中包括用来防止CSRF攻击形式的令牌。要保护表单，需要在表单中包含此隐藏字段并在Flask配置中定义变量`SECRET_KEY`，Flask-WTF会自动完成剩下的工作。

```
{% extends 'base.html' %}

{% block content %}
    <h1>登录</h1>
    <form action="" method="post" novalidate>
        {{ form.hidden_tag() }}
        <p>
            {{ form.username.label }}<br>
            {{ form.username(size=32) }}}
        </p>
        <p>
            {{ form.password.label }}
            {{ form.password(size=32) }}
        </p>
        <p>{{ form.remember_me() }} {{ form.remember_me.label }}</p>
        <p>{{ form.submit() }}</p>
    </form>
{% endblock %}
```

5. 编写登录视图

5.1. 修改登录视图函数

编写登录视图函数`app/login.py`，`form.validate_on_submit()`用于处理POST请求中所有的数据，并进行字段验证，对于GET请求直接返回False。

```
from flask.views import View
from flask import flash, redirect, render_template

from app.forms import LoginForm


class LoginView(View):
    methods = ['GET', 'POST']

    def dispatch_request(self):
        # 加载登录Form表单
        form = LoginForm()
        # 登录Form表单提交验证通过，跳转至首页
        if form.validate_on_submit():
            flash('登录请求，用户{0}，记住我{1}'.format((form.username.data,
form.remember_me.data)))
            return redirect(url_for('index'))

        return render_template('login/login.html', title='登录', form=form)
```

5.2. 注册登录视图函数

编辑`app/__init__.py`，注册登录视图

```
import os
from flask import Flask


def create_app():
    """应用工厂函数"""
    app = Flask(__name__)
    # 加载config配置文件
    app.config.from_pyfile('config.py', silent=True)

    # 注册hello视图URL
    from app.hello import HelloWorld
    app.add_url_rule('/hello', view_func=HelloWorld.as_view('hello'))

    # 注册Login登录视图URL
    from app.login import LoginView
    app.add_url_rule('/login', view_func=LoginView.as_view('login'))

    try:
        # 确保 app.instance_path 存在
        os.makedirs(app.instance_path)
    except OSError:
        pass
```

6. BASE页增加闪烁消息展示

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    {% if title %}
        <title>{{ title }} - 博客</title>
    {% else %}
        <title>博客</title>
    {% endif %}
</head>
<body>
    <div>
        博客:
        <a href="{{ url_for('index') }}">首页</a>
        <a href="{{ url_for('login') }}">登录</a>
    </div>
    <hr>
    {% with messages = get_flashed_messages() %}
        {% if messages %}
            <ul>
                {% for message in messages %}
                    <li>{{ message }}</li>
```

```
                {% endfor %}
            </ul>
        {% endif %}
    {% endwith %}
    {% block content %}{% endblock %}
</body>
</html>
```

7. 登录模板增加字段验证信息展示

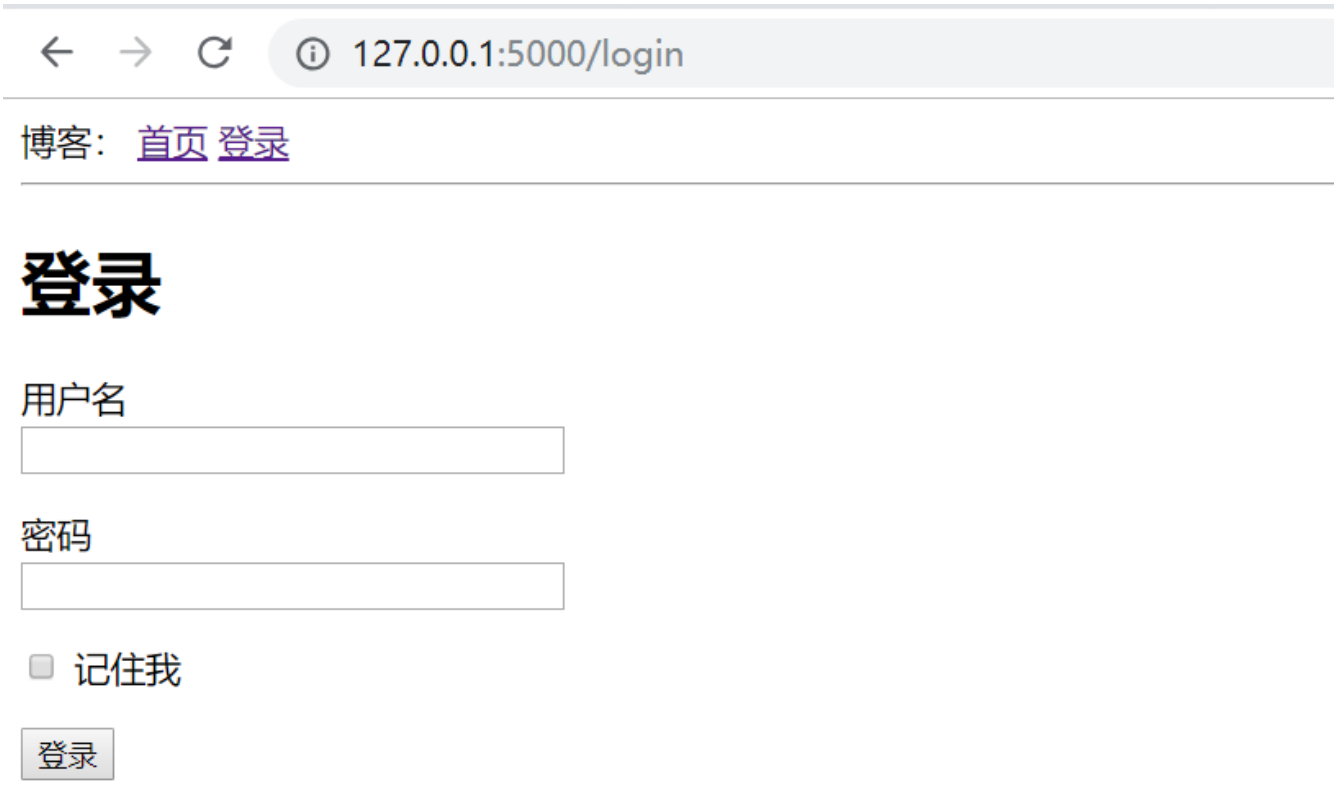
修改`app/templates/login/login.html`文件，增加字段验证的提示信息。

```
{% extends 'base.html' %}

{% block content %}
    <h1>登录</h1>
    <form action="" method="post" novalidate>
        {{ form.hidden_tag() }}
        <p>
            {{ form.username.label }}<br>
            {{ form.username(size=32) }}<br>
            {% for error in form.username.errors %}
                <span style="color: red;">[{{ error }}]</span>
            {% endfor %}
        </p>
        <p>
            {{ form.password.label }}<br>
            {{ form.password(size=32) }}<br>
            {% for error in form.password.errors %}
                <span style="color: red;">[{{ error }}]</span>
            {% endfor %}
        </p>
        <p>{{ form.remember_me() }} {{ form.remember_me.label }}</p>
        <p>{{ form.submit() }}</p>
    </form>
{% endblock %}
```

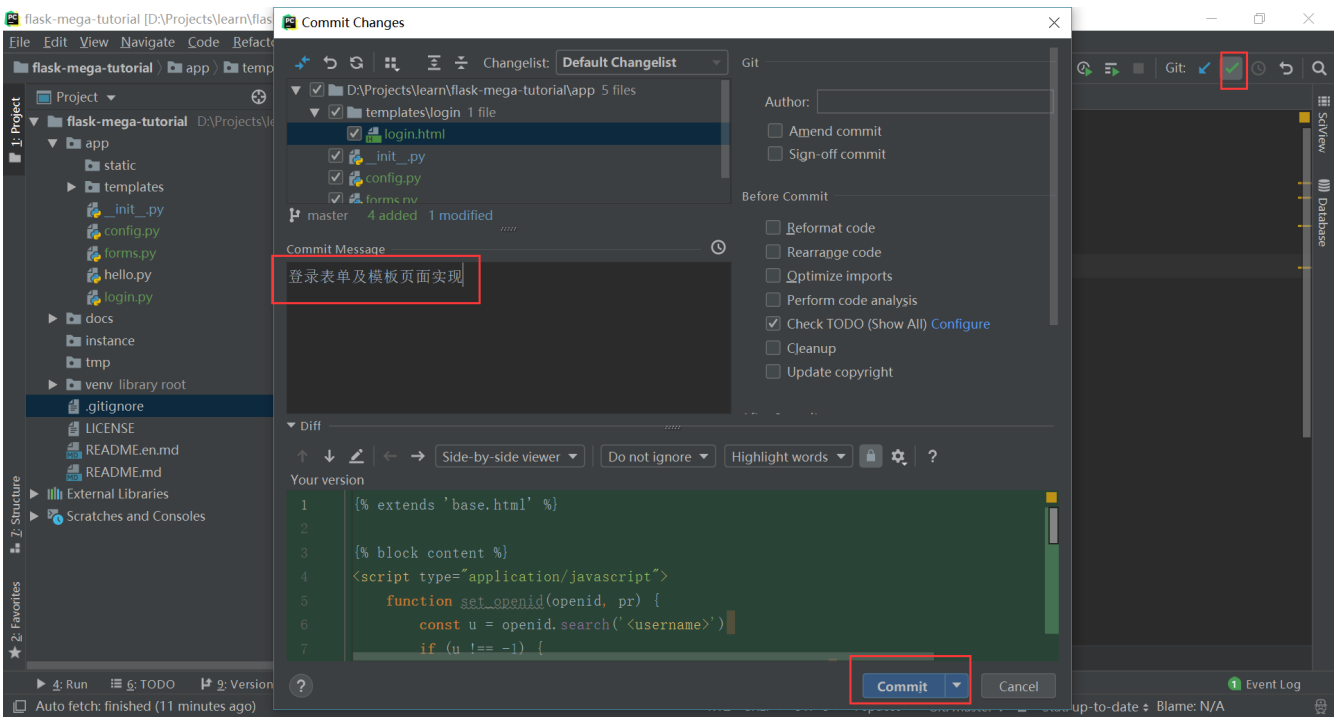
8. 启动服务查看页面

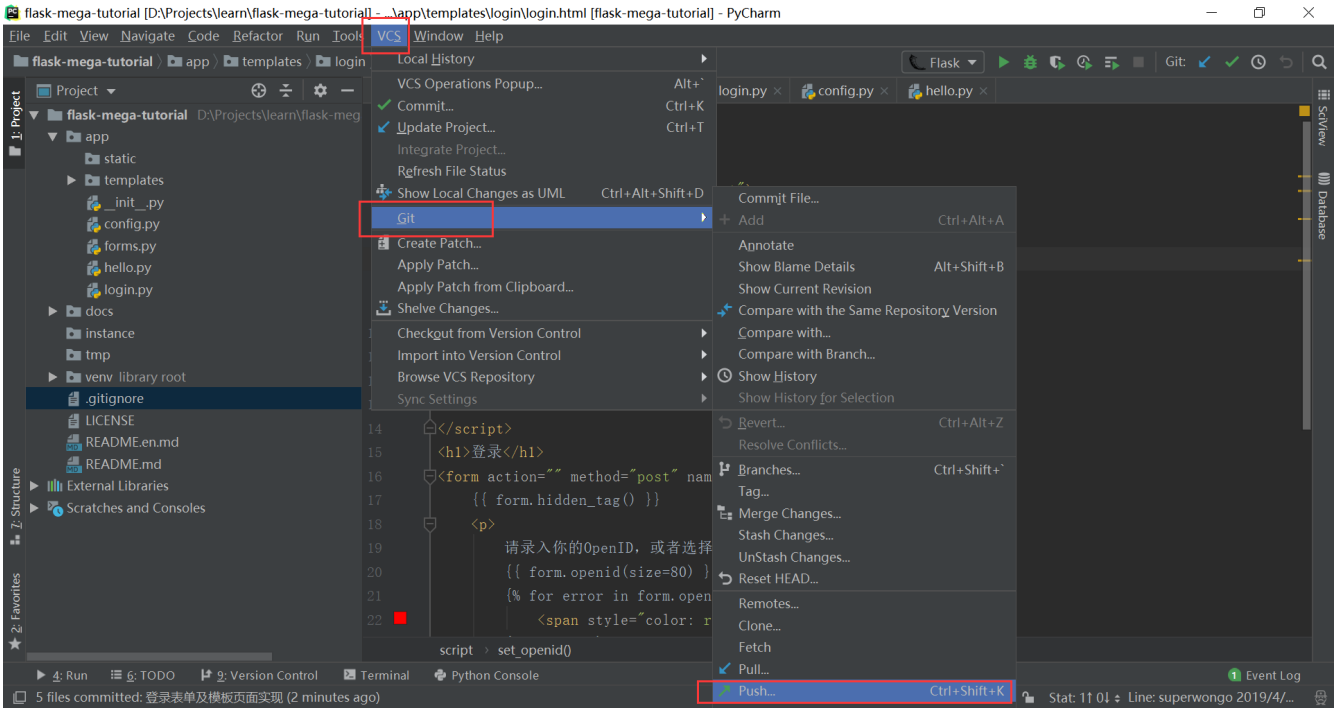
启动服务，查看`/login`页面，以下即为完成的页面部分，但是还没用编写具体的登录逻辑，后续会进行添加。



9. 代码提交版本库

通过pycharm进行版本提交




flask-mega-tutorial [D:\Projects\learn\flask-mega-tutorial] - ...\app\templates\login\login.html [flask-mega-tutorial] - PyCharm
File Edit View Navigate Code Refactor Run Tools VCS Window Help
Local History
VCS Operations Popup... Alt+`
Commit... Ctrl+K
Update Project... Ctrl+T
Integrate Project...
Refresh File Status
Show Local Changes as UML Ctrl+Alt+Shift+D
Git
Create Patch...
Apply Patch...
Apply Patch from Clipboard...
Shelve Changes...
Checkout from Version Control
Import into Version Control
Browse VCS Repository
Sync Settings
Commit File...
Add Ctrl+Alt+A
Annotate
Show Blame Details Alt+Shift+B
Show Current Revision
Compare with the Same Repository Version
Compare with...
Compare with Branch...
Show History
Show History for Selection
Revert... Ctrl+Alt+Z
Resolve Conflicts...
Branches... Ctrl+Shift+`
Tag...
Merge Changes...
Stash Changes...
UnStash Changes...
Reset HEAD...
Remotes...
Clone...
Fetch
Pull...
Push... Ctrl+Shift+K
Project
flask-mega-tutorial D:\Projects\learn\flask-meg
app
static
templates
__init__.py
config.py
forms.py
hello.py
login.py
docs
instance
tmp
venv library root
.gitignore
LICENSE
README.en.md
README.md
External Libraries
Scratches and Consoles
login.py
config.py
hello.py
script set_openid()
4: Run 6: TODO 9: Version Control Terminal Python Console
Event Log
5 files committed: 登录表单及模板页面实现 (2 minutes ago)
Stat: 1↑ 0↓ Line: superwongo 2019/4/...