**一、 数据库模型设计**

作为一个用户登录和注册项目，需要保存的都是各种用户的相关信息。很显然，我们至少需要一张用户表User，在用户表里需要保存下面的信息：

* 用户名
* 密码
* 邮箱地址
* 性别
* 创建时间

我们现在就暂定保存这些信息吧，更多的内容，请大家在实际项目中自行添加。

进入login/models.py文件，这里将是我们整个login应用中所有模型的存放地点，代码如下：

**from** django.db **import** models

*# Create your models here.*

**class** **User**(models**.**Model):

gender **=** (

('male', "男"),

('female', "女"),

)

name **=** models**.**CharField(max\_length**=**128, unique**=**True)

password **=** models**.**CharField(max\_length**=**256)

email **=** models**.**EmailField(unique**=**True)

sex **=** models**.**CharField(max\_length**=**32, choices**=**gender, default**=**"男")

c\_time **=** models**.**DateTimeField(auto\_now\_add**=**True)

**def** \_\_str\_\_(self):

**return** self**.**name

**class** **Meta**:

ordering **=** ["-c\_time"]

verbose\_name **=** "用户"

verbose\_name\_plural **=** "用户"

各字段含义：

* name必填，最长不超过128个字符，并且唯一，也就是不能有相同姓名；
* password必填，最长不超过256个字符（实际可能不需要这么长）；
* email使用Django内置的邮箱类型，并且唯一；
* 性别使用了一个choice，只能选择男或者女，默认为男；
* 使用\_\_str\_\_帮助人性化显示对象信息；
* 元数据里定义用户按创建时间的反序排列，也就是最近的最先显示；

## 二、 设置数据库后端

定义好了模型后，就必须选择我们用来保存数据的数据库系统。Django支持Mysql，SQLite，Oracle等等。

Django中对数据库的设置在settings文件中，如下部分：

# Database

# https://docs.djangoproject.com/en/1.11/ref/settings/#databases

DATABASES = {

'default': {

'ENGINE': 'django.db.backends.sqlite3',

'NAME': os.path.join(BASE\_DIR, 'db.sqlite3'),

}

}

## 三、注册app

每次创建了新的app后，都需要在全局settings中注册，这样Django才知道你有新的应用上线了。在settings的下面部分添加‘login’，建议在最后添加个逗号。

# Application definition

INSTALLED\_APPS = [

'django.contrib.admin',

'django.contrib.auth',

'django.contrib.contenttypes',

'django.contrib.sessions',

'django.contrib.messages',

'django.contrib.staticfiles',

'login',

]

## 四、创建记录和数据表

app中的models建立好了后，并不会自动地在数据库中生成相应的数据表，需要你手动创建。

进入Pycharm的terminal终端，激活虚拟环境，执行下面的命令：

python manage.py makemigrations

返回结果：

(mysite\_env) F:\Django\_course\mysite>python manage.py makemigrations

Migrations for 'login':

login\migrations\0001\_initial.py

- Create model User

Django自动为我们创建了login\migrations\0001\_initial.py文件，保存了我们的第一次数据迁移工作，也就是创建了User模型。

接着执行下面的命令：

python manage.py migrate

Django将在数据库内创建真实的数据表。如果是第一次执行该命令，那么一些内置的框架，比如auth、session等的数据表也将被一同创建，如下所示：

(mysite\_env) F:\Django\_course\mysite>python manage.py migrate

Operations to perform:

Apply all migrations: admin, auth, contenttypes, login, sessions

Running migrations:

Applying contenttypes.0001\_initial... OK

Applying auth.0001\_initial... OK

Applying admin.0001\_initial... OK

Applying admin.0002\_logentry\_remove\_auto\_add... OK

Applying contenttypes.0002\_remove\_content\_type\_name... OK

Applying auth.0002\_alter\_permission\_name\_max\_length... OK

Applying auth.0003\_alter\_user\_email\_max\_length... OK

Applying auth.0004\_alter\_user\_username\_opts... OK

Applying auth.0005\_alter\_user\_last\_login\_null... OK

Applying auth.0006\_require\_contenttypes\_0002... OK

Applying auth.0007\_alter\_validators\_add\_error\_messages... OK

Applying auth.0008\_alter\_user\_username\_max\_length... OK

Applying login.0001\_initial... OK

Applying sessions.0001\_initial... OK