前端使用

- 安装Node js

```
npm install yarn
yarn install
yarn start
```

- `localhost:8000`

后端使用

- Java
- 安装maven
- `localhost:3000`

设计报告

基于之前讨论的需求来设计。

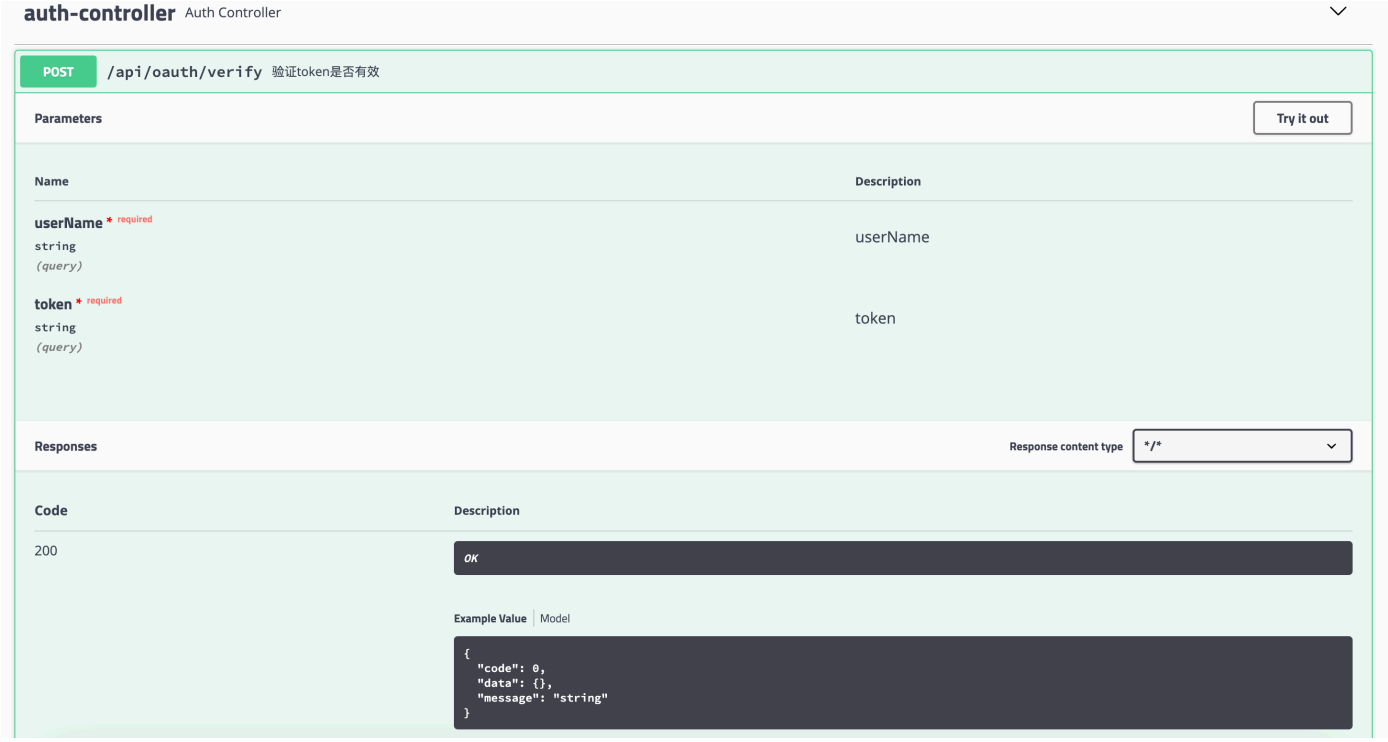
接口文档

需要包括：

1. 请求类型, URL
2. 请求参数：是否必须；参数类型
3. 响应码以及返回数据的格式

关于HTTP请求，如果没有相关经验，请自行网上学习一下。

接口文档的写法可以参考下图的格式：我们的java项目运行之后，访问`http://localhost:3000/swagger-ui.html`来查看



怎么生成上图中的API文档？在对应方法上使用`@ApiOperation`注解，具体可以网上查一下swagger。

```
...
@ApiOperation(...)
public Response example(...){
   ...
}

...
```

我们整个管理子系统的后端接口返回规定如下：

```
// ResponseData.java
{
  "code": 0,  // 0: 成功,  -1: 失败
  "data": {...}, // 实际要返回的数据放在这里，可以为null
  "message": "..."  // 发生错误时的信息
}
```

p.s. Java class -> json

```
public class Example {
  private int a;
  private string b;
}

===> json example:

{
  "a": 0,
  "b": "test"
}
```

数据表设计和E-R图

1. 根据需求，设计数据表结构（关系型）

参考格式：

属性	数据类型	not null	备注	其他
topic_id	bigint	y	主题贴ID	primary key, auto_increment
session	varchar(10)	y	所属版块	/
title	varchar(40)	y	标题	/
author_name	varchar(40)	y	作者姓名	
author_id	varchar(40)	y	作者ID	foreign key(user)
content	text	y	内容	/
create_time	Datetime	y	创建时间	/
last_edit_time	Datetime	y	最后编辑时间	/
reply_cnt	int	y	回复数	/
view_cnt	int	y	浏览数	/
like_cnt	int	y	点赞数	/
dislike_cnt	int	y	点踩数	/

注：Java项目中已有一个数据表实现:

数据表： `entity/User.java`

数据持久化： `repository/UserRepository.java`

增删改查封装：`service/UserService.java`

HTTP请求处理：`controller/UserController.java`

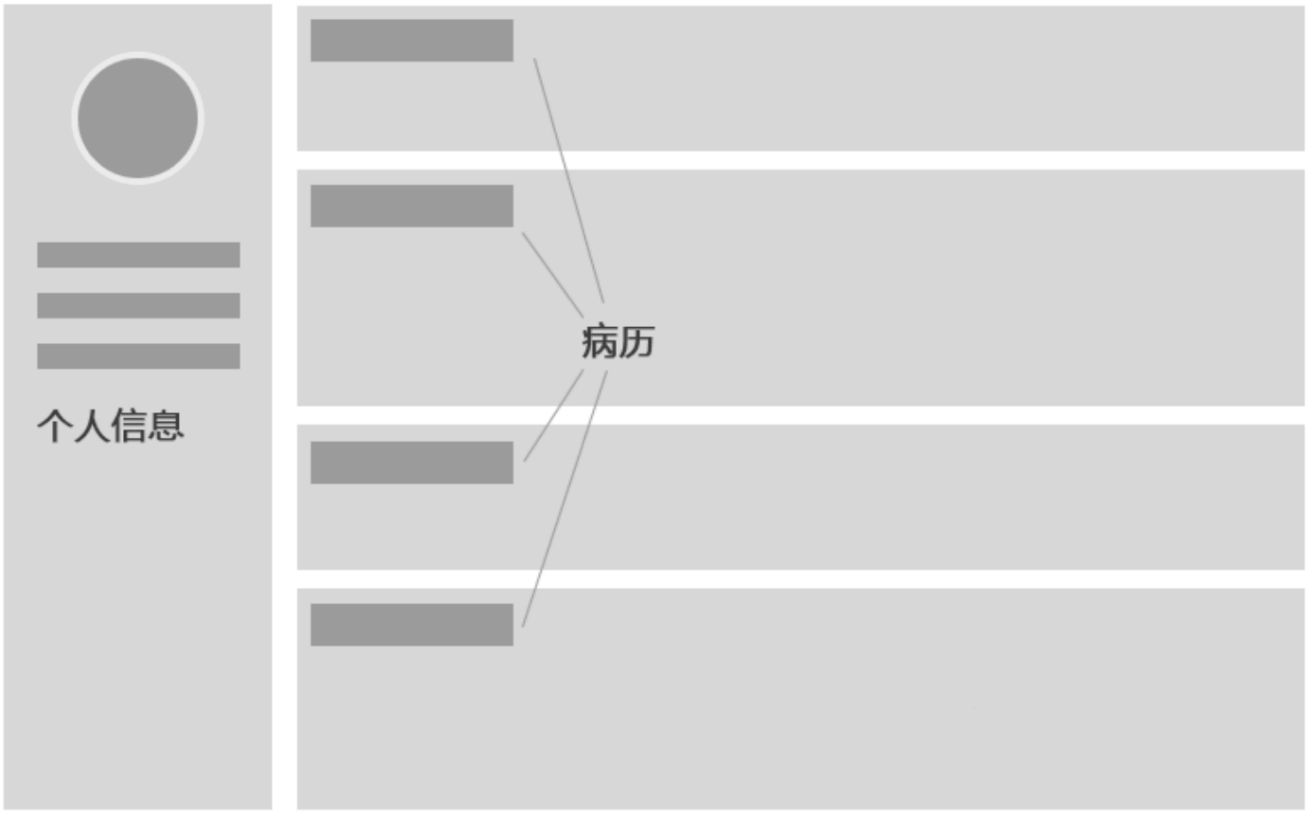
2. 根据数据表画E-R图
3. 建表语句

```
drop table if exists ...;
create table (...);

# 如果需要初始数据
insert into ...values(...);
```

前端基本界面、模块展示

如果来得及做出界面，可以对网页截图。如果来不及，可以像下图一样，描述一下网页结构。目前能用的图不多，等到代码写完之后可以再详细补。



前端模块：**介绍一下有哪些模块，分别是什么功能**；描述模块之间的运行流程，可以从需求报告那里复制一下流程图。

杂项

参考大群里的pdf吧。

代码分工

`https://github.com/pan2013e/se-management`

目前组长已经做了一个简易的登录界面和后端接口（后续可能需要改进）

根据之前的讨论情况，前端：pzy, wh

后端代码的编写 2 人 （Java，一人写数据库操作和其他逻辑 zyh ，一人写HTTP接口处理 lyq)

测试代码的编写 1 人（后端：JUnit单元测试；前端可以用浏览器自动测试工具 wjh）

报告由大家共同完成。