蛋博后端设计文档

版本 20201024.2

一、简介

1. 框架与环境

蛋博后端使用 Python 写成，调用了 django 框架，django 是一个开放源代码的 Web 应用框架，由 Python 写成。django 采用了 MVT 的软件设计模式，即模型（Model），视图（View）和模板（Template）。

在这里我们仅仅使用了模型和视图来构建我们的 Web API，模型用来存储数据，视图里面实现了各种功能接口，使得后端成为前端能够调用的各种文件和数据库的接口的实现。

数据库接口使用了默认的 Sqlite3 数据库，实现了自给自足的、无服务器的、零配置的、事务性的 SQL 数据库引擎，满足了用户信息的高效存储。

2. 功能与交互

后端负责用户，博文，评论等相关功能的模型（数据库）实现和增删改查的接口包装，完成登陆注册，发布博文，发表评论，搜索博文等等功能。

前后端通过本机网络中 HTTP 通信，前端一般发送 POST 信息，后端根据这个信息进行响应，返回前端所需要的信息。前端只需要专注于用户引导和显示，后端负责业务处理和数据持久化的工作。这样前后端有效的进行解耦，能够区分功能，加快设计编码效率。

二、用户模块

1.模型

用户模块包括三个模型，第一个是用户模型，用户包括**用户名（主键）**，密码，**邮箱（主键）**，昵称，签名，生日，性别，地址字段。用户模型用来描述用户所持有的信息，并在数据库中进行存储。用户名，密码，邮箱是用于登陆和注册相关功能的实现，而其他字段都显示于用户的个人页面，作为用户的个人展示。

第二个是验证码模型，验证码包括**邮箱（外键）**，验证码，时间戳，发出时间字段。验证码用于注册和更改密码的时候进行邮箱验证的工具，每个验证码跟某一个邮箱和时间关联，可以通过检查时间间隔来验证有效性。

第三个是头像模型，头像包括关联的用户和图片字段。用户的头像只存在一个，多余的头像会被删除，当没有头像时返回默认头像。

2.视图

我们根据用户的需求编写了相关的功能接口设计，总的来说包括登陆，注册，个人信息展示和找回密码功能。这些功能分别在不同的一个页面内完成。

（1）注册功能：用户首先提供邮箱，然后接受验证邮件，收到验证码，然后将用户名，密码，邮箱和验证码一起提交即可。注意只有当最后注册的时候才会提交所有信息，后端会在这时进行查验是否符合规范。我们在后端实现了发送验证邮件和注册的接口，发送验证邮件对应了注册页面的“验证邮箱”按钮，注册对应了最后的“注册”按钮。

（2）登陆功能只需要提供用户名和密码，然后提交。后端进行查验之后即登陆成功。
（3）在登陆功能旁边就是修改密码/找回密码的功能，这个和注册一样首先提供邮箱，然后接受验证邮件，收到验证码，然后将用户名，密码，邮箱和验证码一起提交即可。该页面同样应有“验证邮箱”和“修改”按钮。
（4）其他信息需要在个人信息展示进行修改，我们针对每一种信息，编写了修改接口和获取接口。前端只需要提供当前的用户名和响应修改的字段就可以了。注意修改头像必须传递文件，获取头像返回值是相对路径，加上前缀才能获取真实路径。

三、博客模块

1.模型

博客暂时只包含了两个模型，第一个是博客模型，它包括**发布博客的用户（外键）**，发布时间，文本内容字段。我们查询博客的时候，根据发布博客的用户进行筛选，返回用户发送的所有博客。

第二个是图片模型，它包括**图片所属的博客（外键）**，图片编号，图片路径字段。由于图片是依赖博客的，我们查询图片的时候是根据所属博客进行查询，将博客关联的图片与博客内容一起返回，我们创建的时候也是必须先有博客然后才放入相应的图片。

2.视图

我们暂时只实现了发送和获取博客的功能。

（1）博客的发送需要提供用户名，文字内容和图片。我们后端根据用户名将当前时间和文字内容存入博客模型，生成博客实例之后，我们再根据创建的博客存入图片。图片可以存入多张，将会用不同的序号进行存储，保持上传时候的顺序。

（2）获取博客只需要相应的用户名，后端就能从博客中选取相应用户的博客，然后再根据博客选取出相应博客的图片，打包成{时间戳：博客内容}的词典类型返回给前端。博客内容则是{‘content’:文字内容, ‘pictures’:[图片路径列表]}格式的词典。同样，获取的图片路径也是相对路径，需要加上前缀获得根据网址的绝对路径。