设计文档

背景

简单的用户管理系统，提供功能包括

- 用户登录
- 查看用户信息（用户名、昵称和头像）
- 编辑用户信息（昵称和头像）

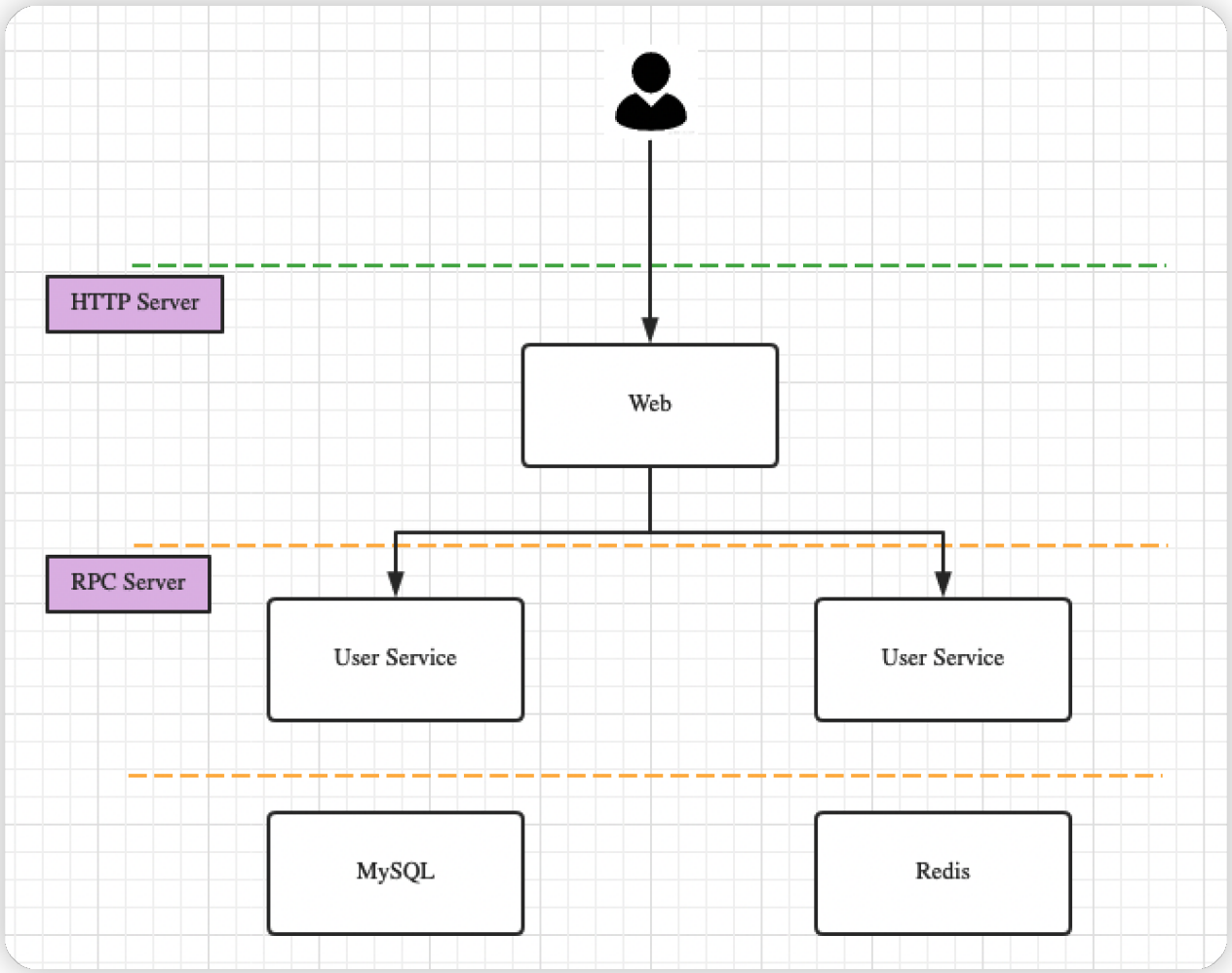
功能划分&排期

任务		人/日
文档设计		1
搭建http server		0.5
搭建tcp server		0.5
实现简单rpc		1.5
用户登录/注册		1
用户获取信息/更新信息		0.5
单元测试		0.5
性能测试		0.5
搭建页面		0.5
Total		7

整体架构

- HTTP Server：接收HTTP请求；参数校验；作为RPC client请求User服务

- RPC Server：接收HTTP Server发来的RPC请求，解析TCP报文；暴露RPC接口；处理实际业务逻辑（用户登录/注册/修改信息等）；Redis/MySQL请求读写



接口设计

这里指对外暴露的HTTP接口

~~用户注册~~

- ~~POST /api/v1/user/register~~
- ~~application/json~~

~~请求~~

字段	是否必须	类型	说明
userName	是	string	用户名，6-32
password	是	string	密码，6-32
nickName	是	string	昵称，2-20

返回

字段	是否必须	类型	说明
code	是	int	错误码
msg	否	string	说明
data	否	JSONObject	数据

用户登录

- POST /api/v1/user/login
- application/json

请求

字段	是否必须	类型	说明
userName	是	string	用户名
password	是	string	密码，6-32

返回

字段		是否必须	类型	说明
code		是	int	错误码
msg		否	string	说明
data		是	JSONObject	数据
	sessionId	是	string	sessionId

获取用户信息

- GET /api/v1/user/info

请求

sessionId通过cookie传递，无其他字段

		字段	是否必须	类型	说明

返回

字段		是否必须	类型	说明
code		是	int	错误码
msg		否	string	说明
data		是	JSONObject	数据
	userName	是	string	用户名
	nickName	是	string	昵称
	avatarUrl	是	string	头像url

编辑用户信息

- POST /api/v1/user/edit
- application/json

请求

字段	是否必须	类型	说明
nickName	否	string	昵称，2–20
avatarUrl	否	string	头像url

返回

字段	是否必须	类型	说明
code	是	int	错误码
msg	否	string	说明
data	是	JSONObject	数据

上传用户头像文件

- POST /api/v1/user/uploadAvatar
- application/json

请求

字段	是否必须	类型	说明
file	是	file	头像文件，大小<=1M

返回

字段		是否必须	类型	说明
code		是	int	错误码
msg		否	string	说明
data		是	JSONObject	数据
	avatarUrl	是	string	头像访问url

错误码

code	说明
0	Success
1	Timeout
2	Internal Error
101	用户名不合法
102	密码不合法
103	昵称不合法
104	用户登录名称或密码错误
105	用户需要登录
106	用户更新信息失败
107	用户上传头像失败
108	用户已存在
109	获取用户信息失败
110	用户session不存在

数据设计

MySQL数据表

表名 **sp_et_user**

字段	是否必须	类型	说明
id	是	bigint	主键
user_name	是	varchar(64)	用户名，UK
password	是	varchar(64)	密码，哈希值
nick_name	是	varchar(64)	昵称
avatar_url	否	varchar(2048)	头像url
modify_time	是	datetime	最近修改时间
create_time	是	datetime	创建时间

数据表初始化

```sql
-- 建数据库
CREATE DATABASE IF NOT EXISTS user;
USE user;

-- 建表
CREATE TABLE if NOT EXISTS `sp_et_user` (
    id bigint unsigned auto_increment NOT NULL COMMENT '主键',
    user_name varchar(64) NOT NULL COMMENT '用户名',
    password varchar(64) NOT NULL COMMENT '密码',
    nick_name varchar(64) NOT NULL COMMENT '昵称',
    avatar_url varchar(2048) NULL COMMENT '头像url',
    create_time DATETIME NOT NULL COMMENT '创建时间',
    modify_time datetime NOT NULL COMMENT '最近修改时间',
    PRIMARY KEY (id),
    UNIQUE KEY `uk_username` (user_name)
) ENGINE=InnoDB
DEFAULT CHARSET=utf8mb4
COLLATE=utf8mb4_unicode_ci
COMMENT='entry task用户表';
```

Redis缓存

redis用于存储session信息和用户信息。

session信息

key为**sessionId**，value为**userName**，expire时间为3天。

1. 登录时生成sessionId，写入缓存
2. 获取个人信息和更新信息时，读取缓存

用户信息

1. key为**userName**，value为json string，expire时间为3天
2. value结构 {"nickName": "ha", "avatarUrl": "xx/1.jpg"}
3. 缓存读写采取cache-aside模式
    a. 读取：先从缓存获取数据，有数据直接返回，没有数据从DB取出返回并放到缓存

b. 更新：先更新DD，再删除缓存

功能设计

用户登录

1. 用户密码通过bcrypt（https://www.cnblogs.com/flydean/p/15292400.html算法哈希，存储在数据库；登录时的密码校验通过bcrypt算法实现校验
2. 用户登录成功后，生成sessionId，返回给客户端；后续获取/编辑用户信息都通过sessionId操作

RPC

通过client-server模式进行网络通信，参考go-rpc(https://pkg.go.dev/net/rpc)

> https://segmentfault.com/a/1190000013532622

方法声明

Go | 复制代码

```go
// serviceMethod 提供rpc服务的方法名，格式“Service.Method”
// args 入参
// reply 响应
func (client *Client) Call(serviceMethod string, args, reply interface{})
error {
}
```

方法注册

注册服务内的方法，符合以下条件

- 方法所属类型是导出的
- 两个入参，均为导出或内置类；第二个入参必须是一个指针，返回结果； 返回值为 error 类型

```go
func Register(rcvr interface{}) error {}
```

编解码

协议基于TCP。字段包含两部分

- 魔数字段（0xaabb），JSON编码
- 数据部分，使用Gob编码（ https://cloud.tencent.com/developer/section/1141539）
  - header
  - body

客户端发送请求

```go
type Header struct {
    ServiceMethod string
    Seq           uint64
    Error         string
}
```

单元测试

使用gomonkey进行mock，查看以_test结尾的go文件，单测主要覆盖：

- RPC服务：登录/编辑用户信息等
- DAL层，db读写

性能测试

基于样例的benchmark.go