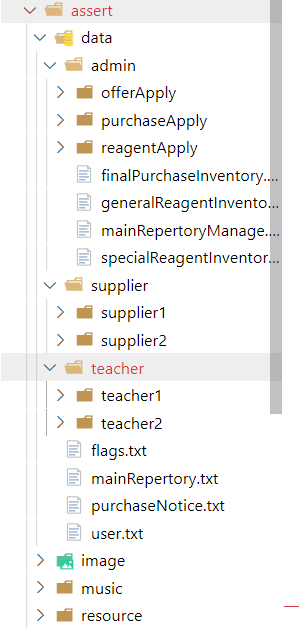
# YUFFIE

## .vscode

vscode配置文件，可以不用vs，其他编译器运行后复制即可，其他后续调整。

## assert



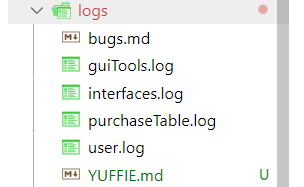
### data

数据存放处，其中flags、user、supplier部分不用管。admin-offerApply为多个供应商报价表。purchaseApply为采购申请表。reagentApply为试剂使用申请表。

purchaseNotice为采购公告（可以只为一串文字），mainRepertory为总的仓库。generalReagentInvetory为采购申请审核后的一般试剂。mainRepertoyManage为总仓库更新日记。

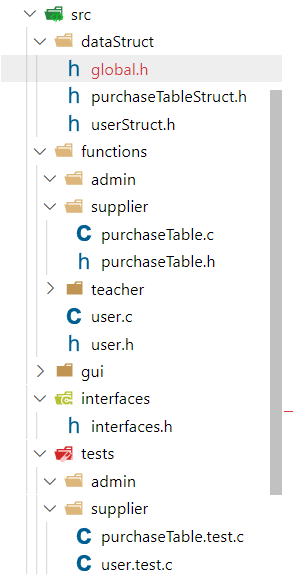
labRepertory为实验室剩余药剂，recyclingManage为回收废液记录。

### logs



bugs.md记录bug（没规定必须），function内.c文件需要对应同名log文件。

## src



### dataStruct

数据结构储存，此处.h不需要log文件。基本上一个functions中有一个同名+struct后缀文件，定义文件表的抽象，包括单个项，表：

/\*
  
 @value:user表每列最大长度。
  
\*/
  
#define INFO\_MAXSIZE 50
  
  
/\*
  
 @value:"user表最大元组数。"
  
\*/
  
#define USER\_LIST\_MAX\_SIZE 199
  
/\*
  
 @function:"用户实体。"
  
 @value:{
  
 userName:"用户名称",
  
 password:"用户密码",
  
 identity:"用户身份",
  
 id:"当前身份id"
  
 }
  
\*/
  
typedef struct User {
  
 String userName;
  
 String password;
  
 IDENTITY identity;
  
 int id;
  
}User;
  
/\*
  
 @function:"全局储存的所有用户数据，避免多次读取文件。"
  
 @range:[0,USER\_LIST\_MAX\_SIZE]
  
\*/
  
User userList[USER\_LIST\_MAX\_SIZE];
  
  
/\*
  
 @function:"用户数据长度。"
  
 @range:[0,USER\_LIST\_MAX\_SIZE]
  
\*/
  
int userListLen;
  
  
/\*
  
 @function:"当前用户。"
  
 @range:[NULL|user]
  
\*/
  
User presentUser;

这些基本上是必须的，其他的自定义。

#### global

所有datastruct中.h文件需要包含，定义一些全局的数据结构，包括Status等。

### functions

基本上在对应文件夹创建对应txt同名文件,编写操作函数即可。最少包括初始化把数据全部载入到数组。

### GUI

gui文件。

### interface

接口规范定义。

### test

function中每个.c需要同名测试函数。基本格式为：

// 载入用户测试。
  
 initUserList();
  
 assert(userListLen == 5);
  
 assert(userList[0].userName == "haruluya");

这是最终的规范格式，其他测试自行完成。

### globalConst

gui全局变量，不用管。

# 基本流程

在functions中创建txt文件同名函数.c.h，在dataStruct创建同名+struct后缀.h,定义txt数据抽象，编写对应操作函数，在log中添加同名log,在test中测试，确认输入输出准确即可，添加注释。能想到的基本函数完成即可，后续配合gui添加其他函数。