

## 4 作业管理系统 HWMS

根据题目要求，本人独立完成了 bash shell 脚本程序`HWMS` (HomeWork Management System)的设计。程序功能及要求、运行结果详见**功能描述文档**；设计思想、功能模块、数据结构与算法详见**设计文档**。最后一同给出**源程序**。

### 功能描述文档：

在运行 HWMS 前，请仔细阅读该功能描述文档。

### 前置条件

HWMS 运行于支持 bash shell 的 linux 平台。

HWMS 采用 whiptail 实现了图形化 shell 界面，提供令人舒适的作业管理交互方式。因此，你的 linux 系统必须已经安装 whiptail。

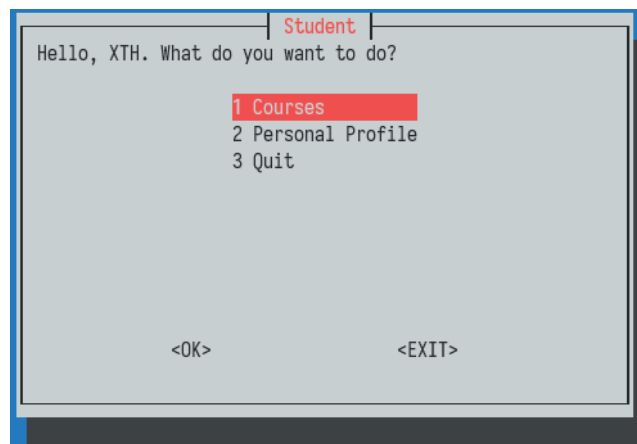
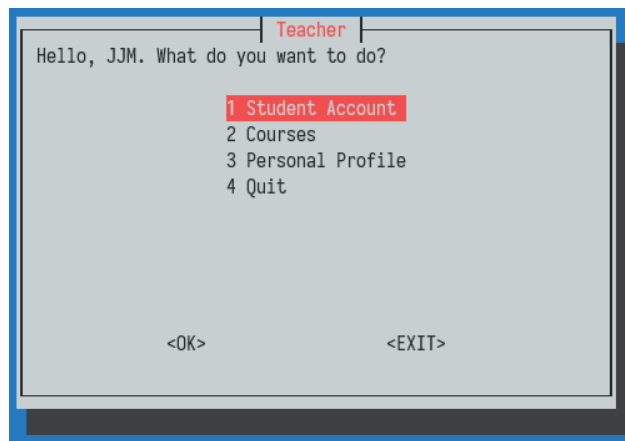
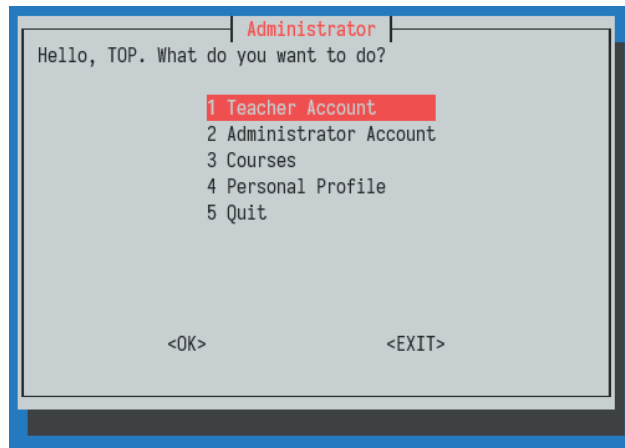
此外，HWMS 的所有数据全部存放于数据库，要求 MySQL 8.0.0 或更高版本。在完成 MySQL 8.0+的安装及基本配置后，你需要为 HWMS 单独配置一个 MySQL 账户，并在 HWMS 程序代码的第 7、8 行配置用户名及密码（该账户需要支持在 localhost 登录，且拥有所有表的所有权限，即拥有 all privileges on \*.\*）。  
请注意：你不需要手动在 MySQL 中创建或修改任何表。

当你第一运行 HWMS 时，请执行<pathname>/HWMS - intialize，并根据提示做出相应操作。HWMS 的--initialize 选项将为你初始化好需要用到的 HWMS 数据库和相关表；后续使用中，HWMS 将按照你的要求对 MySQL 的 HWMS 数据库内的表进行读写修改，不需要你进入 MySQL 手动进行任何修改。

当你已经初始化好 HWMS 需要用到的数据库后，你可以直接执行<pathname>/HWMS 以运行 HWMS。不出意外，你将可以愉快地开始使用了！

### 逻辑简述

系统中根据不同的权限分为三类用户：管理员、教师、学生。在登录完成后的一级菜单的顶部，你将可以查看你自己的身份（Administrator、Teacher、Student）。



HWMS 的交互逻辑非常简单，登陆完成后，你可以在一级菜单中选择你要做的操作，此时选择 Quit 或者 Exit 将退出 HWMS；有的操作会引导你进入二级菜单，之后的操作逻辑是相似的。当你执行一个末端的操作时，你可能会需要在 whiptail 文本框输入相关信息，而输出文本可能在命令行中打印出来，也可能仍然在 whiptail 图形界面直接显示。

## 属性说明

本节将介绍一些重要属性/信息。

每个用户都有 3 个重要信息：用户名 AID/TID/SID、名字 ANAME/TNAME/SNAME、密码 PASSWORD。其中 A/T/S 分别对应管理员/教师/学生。所有的 AID、TID、SID

均不能相同，而 NAME 可以相同。在 Personal Profile 中，可自行修改个人密码 PASSWORD、个人名字 NAME。

每个**课程**由管理员创建，由不同的 CID 进行区分，但可以有相同的课程名 CNAME。每个课程需要与教师的 TID 进行绑定。

每个**课程信息**由教师发布，同一个课程的所有信息由 IID 进行区分，IID 将根据课程信息创建顺序的先后自动从 1 开始连续排序（删除后仍然保证连续）。教师可以根据 CID、IID 修改某一条课程信息的内容，但不能修改 IID。

教师通过学生的 SID 可以将学生绑定至其开设的一门课程；即将 SID、CID 进行绑定。

类似课程信息，同一个课程的所有**作业**由 HWID 进行区分，HWID 将根据作业创建顺序的先后自动从 1 开始连续排序（删除后仍然保证连续）。教师可以根据 CID、HWID 查看唯一的一项作业完成情况、设置作业的信息；教师还可以根据 CID、HWID、SID 为某位学生的作业进行评阅、打分。对于学生，也可以通过 CID、HWID 选择唯一的一项作业进行完成（输入或修改文本内容）提交。

## 功能说明

我们将逐一介绍各类用户的诸多功能。

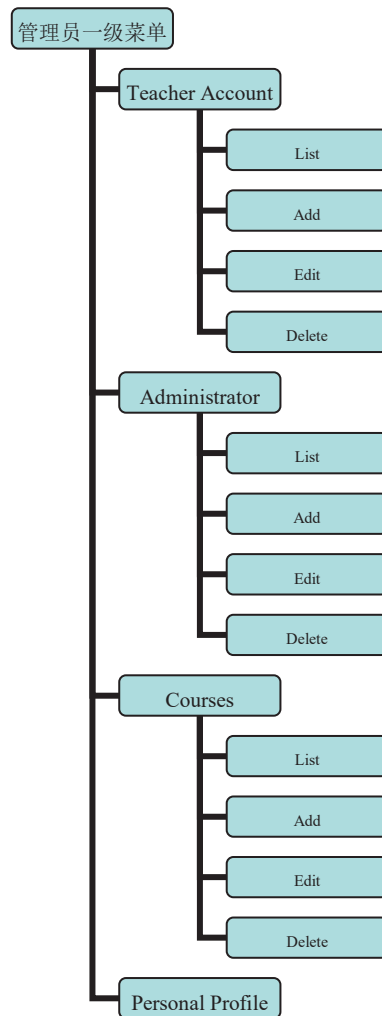
### 管理员 Administrator:

管理员用户拥有着相当高的权限。而 root 用户则是一个不可被移除的管理员用户。

管理员的所有操作：

- 查看 List、增加 Add、编辑 Edit、删除 Delete 教师账户
- 查看 List、增加 Add、编辑 Edit、删除 Delete 其他管理员账户
- 查看 List、增加 Add、编辑 Edit、删除 Delete 所有课程
- 编辑个人信息（名字 ANAME、密码 PASSWORD）

管理员的操作树如下，每个叶子都代表着一个操作。

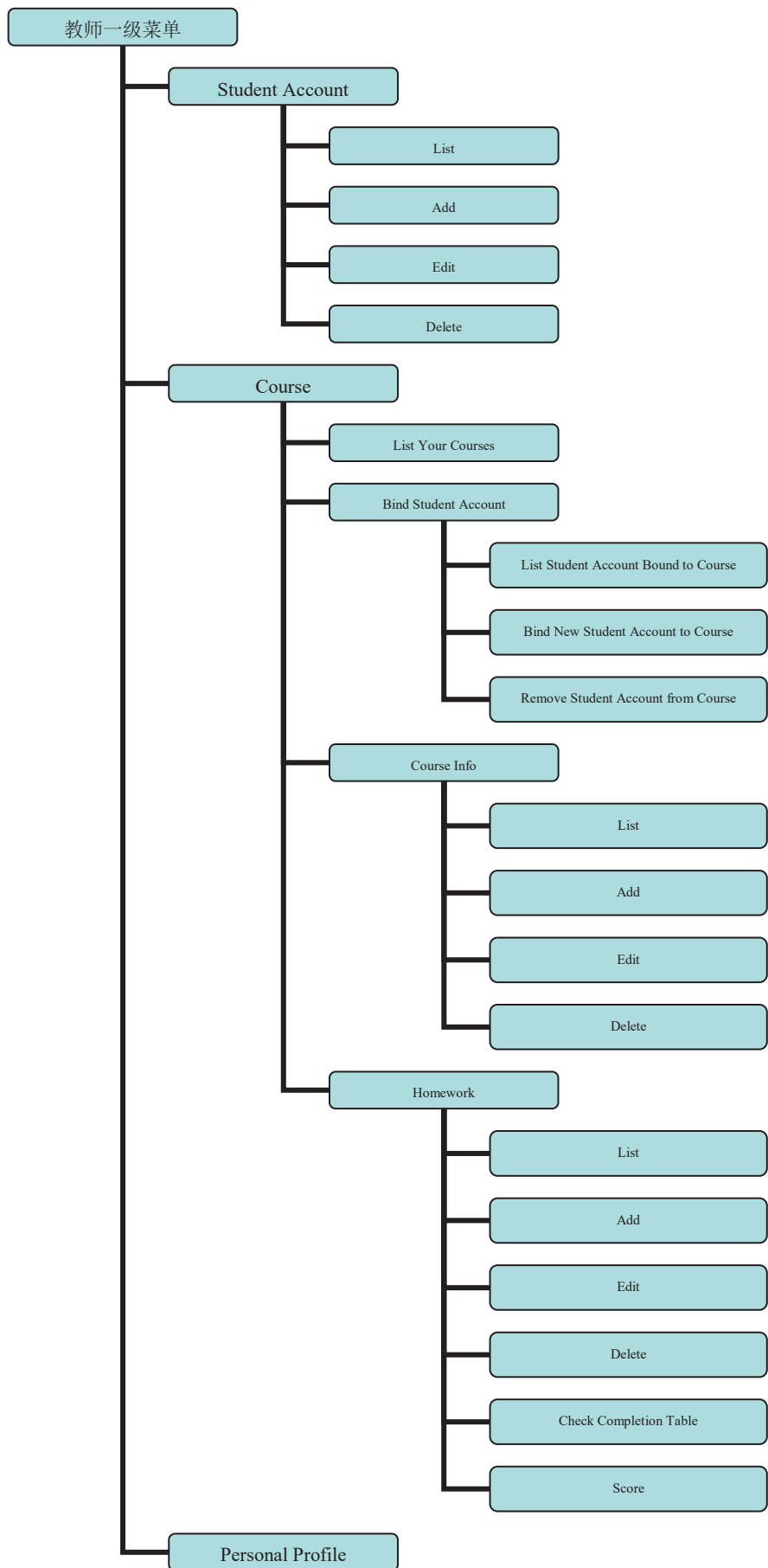


## 教师 Teacher:

教师的所有操作:

- 查看 List、增加 Add、编辑 Edit、删除 Delete 学生账户
- 查看自己的课程 List Your Courses
- 绑定学生账户
  - 查看自己某课程下学生账户绑定情况
  - 绑定新的学生到自己的某课程
  - 移除自己课程下某学生的绑定
- 查看 List、增加 Add、编辑 Edit、删除 Delete 课程信息
- 查看 List、增加 Add、编辑 Edit、删除 Delete 作业，查看作业完成情况 Check Completion Table，为某个作业打分 Score
- 编辑个人信息（名字 TNAME、密码 PASSWORD）

教师的操作树如下，每个叶子都代表着一个操作。

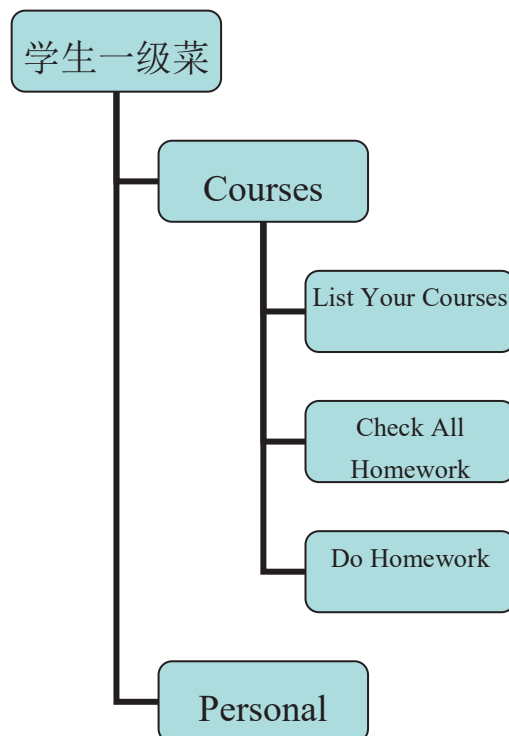


## 学生 Student:

学生的所有操作:

- 查看自己绑定的所有课程 List Your Courses
- 查看自己所有作业的完成情况
- 做某项作业
- 编辑个人信息 (名字 SNAME、密码 PASSWORD)

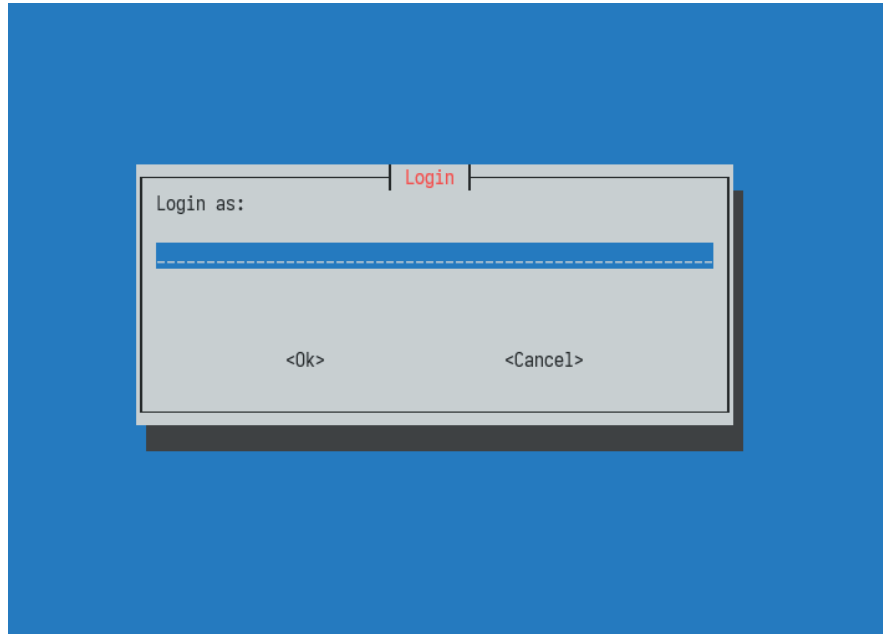
学生的操作树如下, 每个叶子都代表着一个操作。



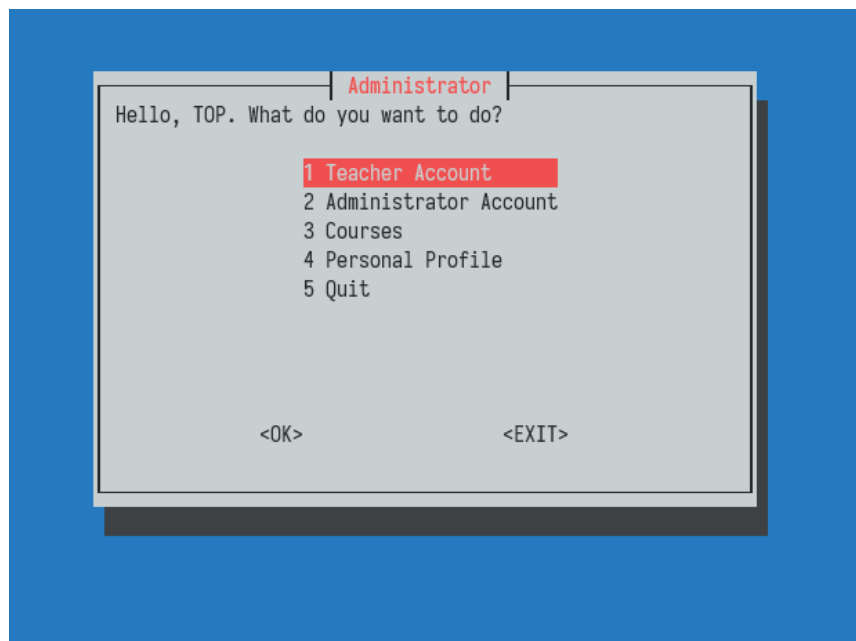
## 使用指南

本节以个别操作为例, 展示操作过程。

如果这是你第一次看到下面的[登录](#)界面, 那说明你刚刚完成了 HWMS 的初始化。此时一个 root 管理员账户已经生成了。用户名 AID 为 'root', 而密码默认为 '123456'。



输入用户名和密码后，你应该可以正常地进入管理员菜单界面：

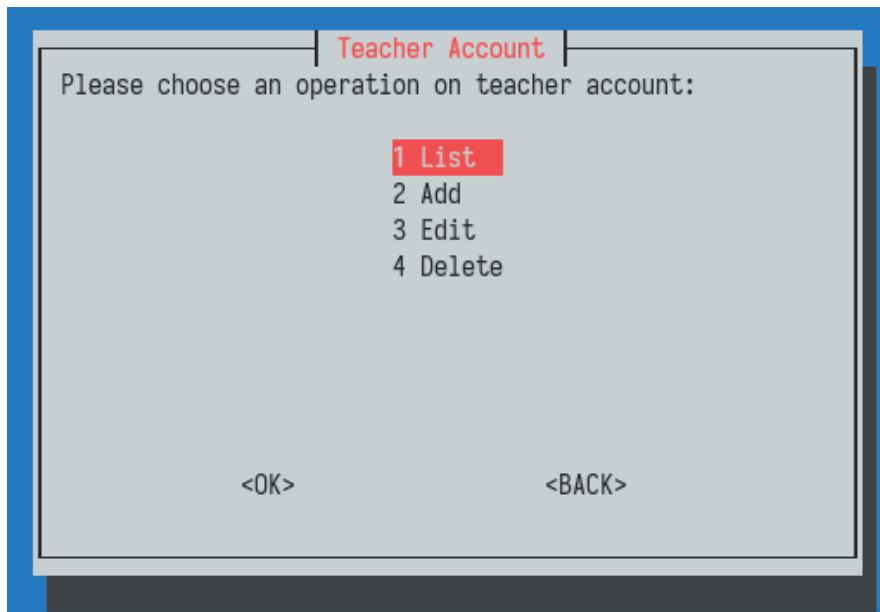


其中，'TOP'是 root 的默认初始名字；在 Personal Profile 中可以修改名字及密码，但不能修改登录用户名 AID。

账户管理操作使用频率很高。

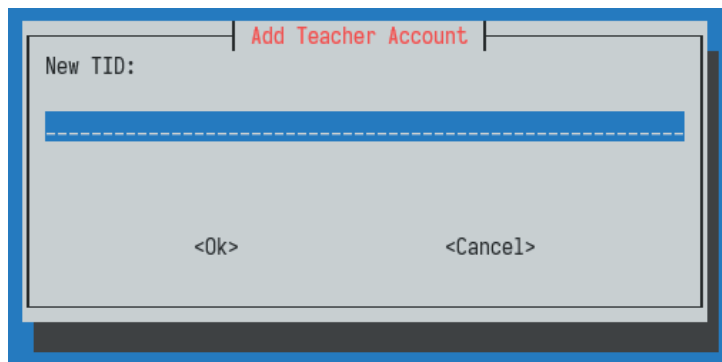
添加一个账户非常简单，以添加一名教师为例：

进入 Teacher Account：



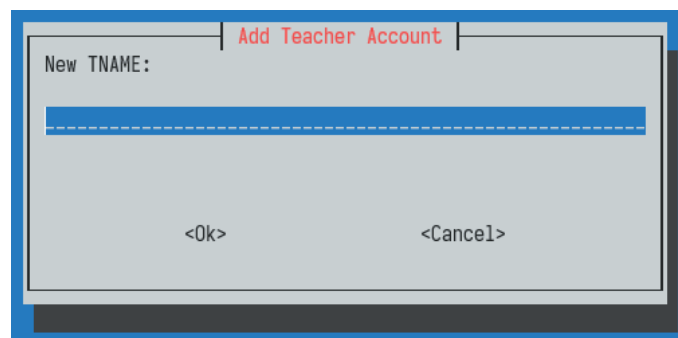
A menu screen titled "Teacher Account" in red. It prompts the user to "Please choose an operation on teacher account:". Below this, there is a list of four options: "1 List", "2 Add", "3 Edit", and "4 Delete". The "1 List" option is highlighted with a red background. At the bottom of the screen, there are two buttons: "<OK>" and "<BACK>".

选择 Add:



A screen titled "Add Teacher Account" in red. It prompts the user to enter a "New TID:". Below the prompt is a blue input field with a dashed line. At the bottom of the screen, there are two buttons: "<Ok>" and "<Cancel>".

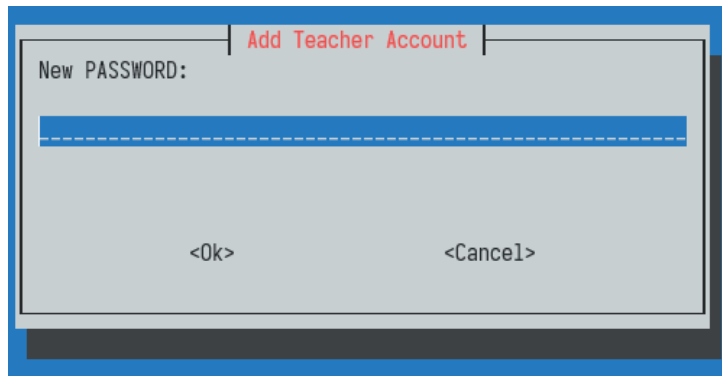
输入一个未被占用的 TID（如 T3），选择 OK:



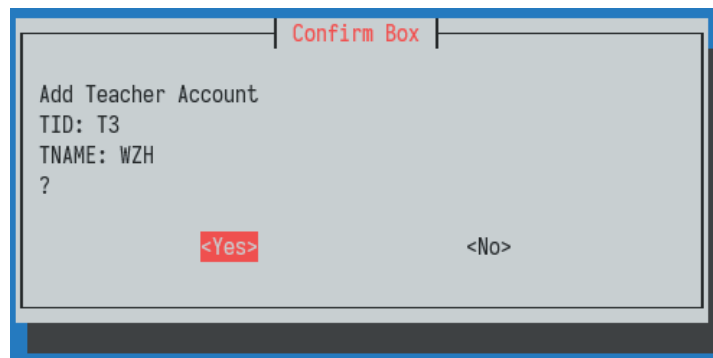
A screen titled "Add Teacher Account" in red. It prompts the user to enter a "New TNAME:". Below the prompt is a blue input field with a dashed line. At the bottom of the screen, there are two buttons: "<Ok>" and "<Cancel>".

输入 TNAME（如 WZH），选择 OK:

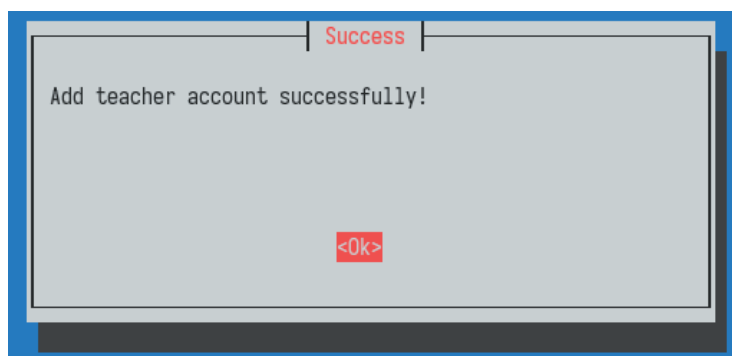




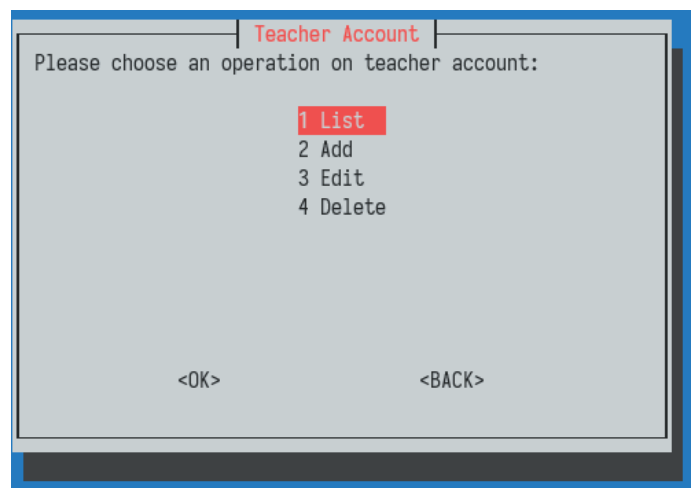
输入密码，选择 OK：



出现提示框，选择 Yes 即可添加：



如果要查询账户，选择 List->OK：



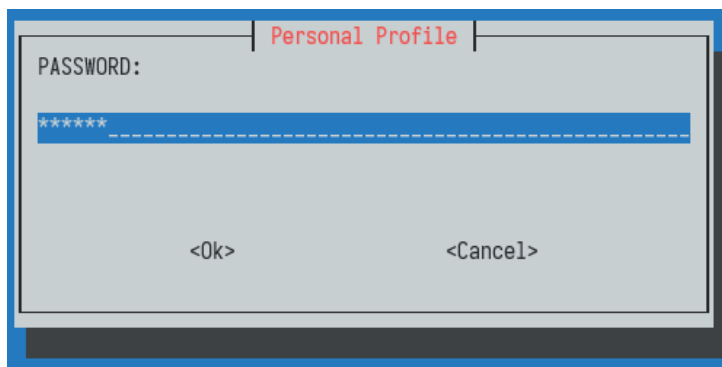
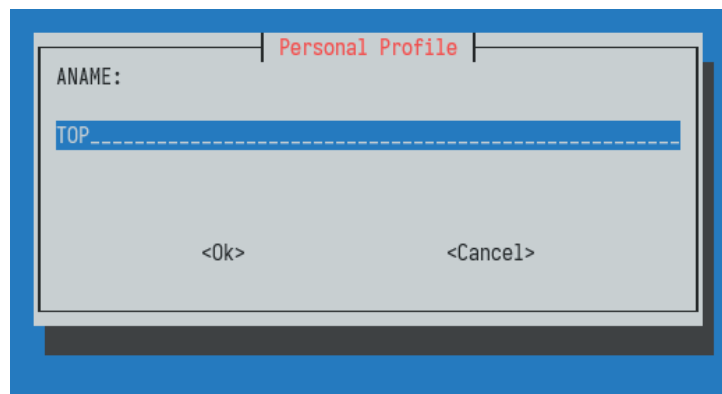
```
-----All Accounts-----
+-----+-----+
| TID | TNAME |
+-----+-----+
| T1  | JJM   |
| T2  | WK    |
| T3  | WZH   |
+-----+-----+
-----Press any key to continue-----
```

即可在命令行看到目前数据库中的所有教师！按下任意键返回图形界面。

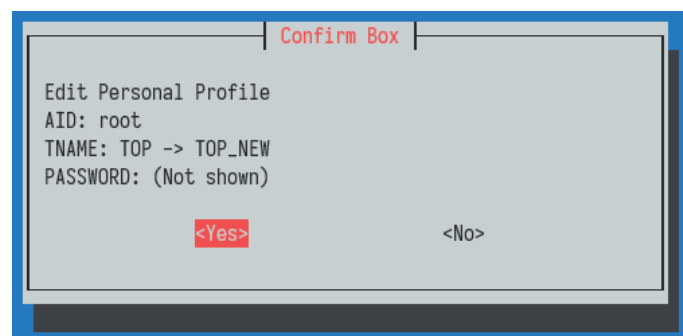
其他账户操作（如 Edit 编辑、Delete 删除）不在此一一展示。教师账户、管理员账户、学生账户的交互逻辑几乎完全一致。

管理员的课程 Courses 选项可以查看、创建、删除、编辑课程，逻辑类似账户操作，只不过需要绑定 CID 与 TID。

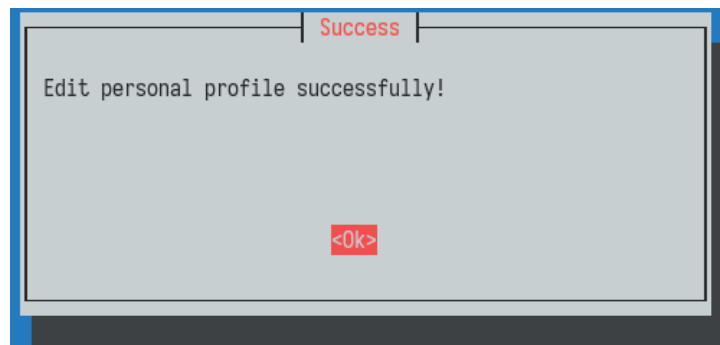
对于任意用户，如需修改个人信息，选择 Personal Profile->OK:



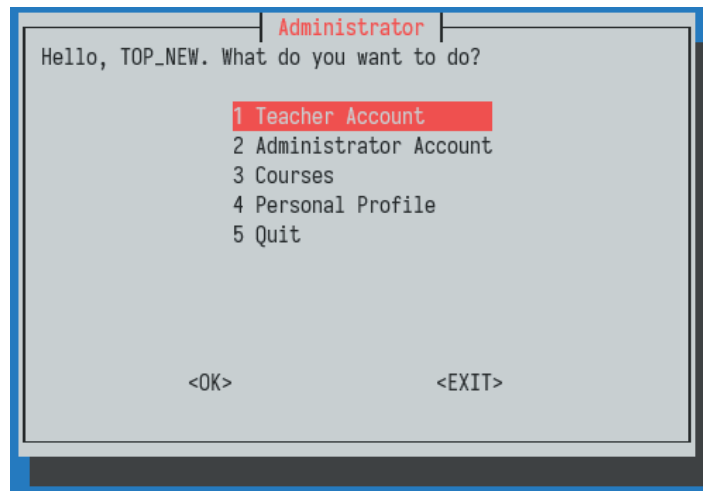
可以修改名字、密码。依次修改后选择 OK 会弹出确认框:



选择 Yes 即可修改：

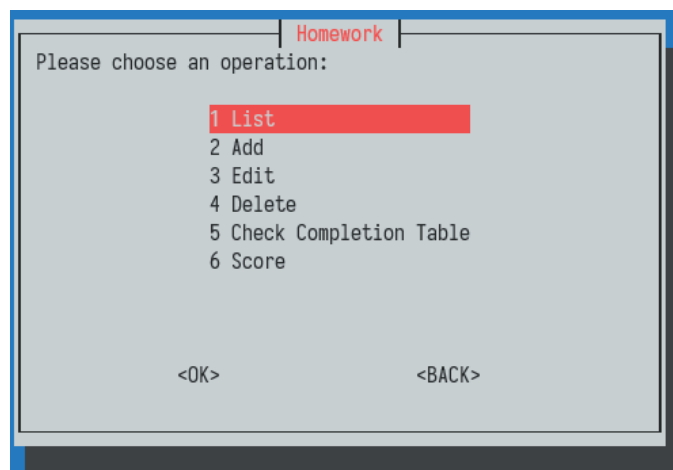


下一次登陆时，可以用新的密码登录，也可以看到新的名字：



作业管理功能是 HWMS 的核心。

a) 对于教师：



查看自己某项课程的所有作业 List：

输入 CID（如 C1）后可以看到

```

-----All Homework of Course with CID 'C1'-----
+-----+-----+
| HWID | INFO |
+-----+-----+
| 1 | Lab1 |
| 2 | Lab2 |
+-----+-----+
-----Press any key to continue-----

```

添加作业的逻辑是相似的：

**Add Homework**

Enter course CID:

C1

<Ok> <Cancel>

**Add Homework**

Enter homework info:

Final Project

<Ok> <Cancel>

**Confirm Box**

Add homework for course with CID 'C1'  
Final Project  
?

<Yes> <No>

**Success**

Add homework successfully!

<Ok>

```

+-----+-----+
| HWID | INFO |
+-----+-----+
| 1 | Lab1 |
| 2 | Lab2 |
| 3 | Final Project |
+-----+-----+

```

如果需要查看作业完成情况，选择 Check Completion Table:

**Check Completion Table**

Enter course CID:

C2

<Ok> <Cancel>

**Check Completion Table**

Enter homework HWID:

1

<Ok> <Cancel>

```

-----Completion Table of Homework with HWID '2' of Course with CID 'C1'-----
+-----+-----+
| SID | DONE | SCORE |
+-----+-----+
| S1 | 1 | NULL |
+-----+-----+
-----Press any key to continue-----

```

DONE 为 1 表示已经该学生做了该作业，SCORE 表示分数(NULL 表示尚未打分)。

如果要为一个学生评阅并打分，选择 Score:

Score

Enter course CID:

C1

<Ok> <Cancel>

Score

Enter homework HWID:

1

<Ok> <Cancel>

Score

Enter student SID:

S1

<Ok> <Cancel>

SCORE

Enter score for student  
SID: 'S1'  
DONE: '1'  
Content: 'Hello teacher!'

99

<Ok> <Cancel>

Confirm Box

Score for student  
SID: 'S1'  
DONE: '1'  
SCORE: 'NULL' -> '99'

<Yes> <No>

Success

Score successfully!

<Ok>

再次查看 Completion Table:

SID	DONE	SCORE
S1	1	99

可以看到成绩为 99。

b) 对于学生:

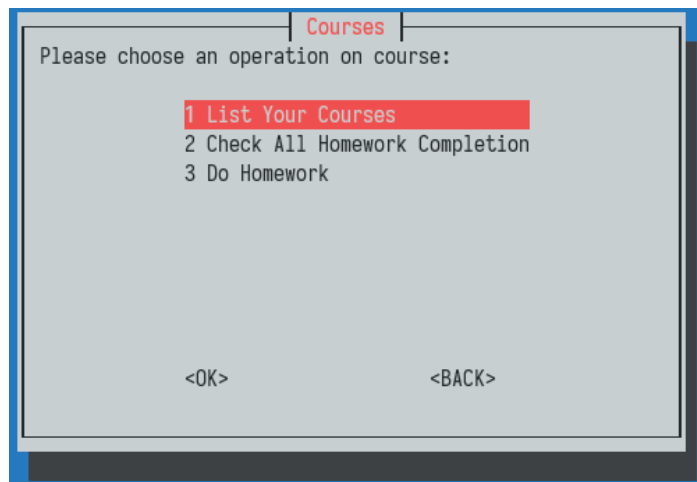
Student

Hello, XTH. What do you want to do?

1 Courses  
2 Personal Profile  
3 Quit

<OK> <EXIT>

在一级菜单下选择 Course:



选择 Check All Homework Completion 可以查看自己所有的作业情况（包括内容、得分等）：

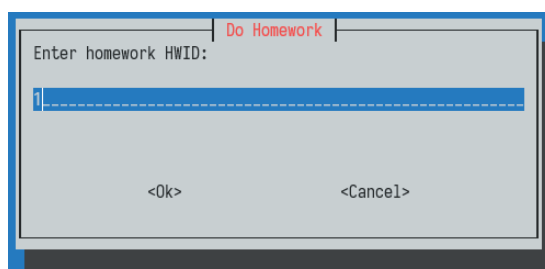
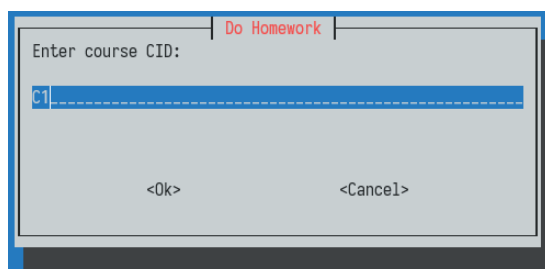
```
-----CID: C1  CNAME: Linux-----
HWID: 1
HW Info: Lab1
+-----+
| DONE | TXT          | SCORE |
+-----+
|  1  | Hello teacher! |   98  |
+-----+

HWID: 2
HW Info: Lab2
+-----+
| DONE | TXT      | SCORE |
+-----+
|  1  | OKKK!!! | NULL  |
+-----+

HWID: 3
HW Info: Final Project
+-----+
| DONE | TXT  | SCORE |
+-----+
|  0  | NULL | NULL  |
+-----+

-----Press any key to continue-----
```

选择 Do Homework 即可编辑作业的内容：





再次查看作业完成情况，可以看到内容修改了：

```
HWID: 1
HW Info: Lab1
+-----+-----+-----+
| DONE | TXT                | SCORE |
+-----+-----+-----+
| 1 | Hello teacher again! | 98 |
+-----+-----+-----+
```

最后还需提及的是：无论是删除课程、作业、学生账户还是教师账户，都会导致相关的表中数据级联删除；以删除教师账户为例：也就是说，不需要手动将教师的课程删除，更不需要将课程信息逐一删除后再删除教师账户，这些操作 HWMS 都会替你完成，不需要担心；当然，如果能在删除课程、作业、学生账户、教师账户时保证相关的关联项目已经完全移除，那是更好的。

其余功能交互方式较为相似，不逐一介绍。

## 设计文档：

### 设计思想

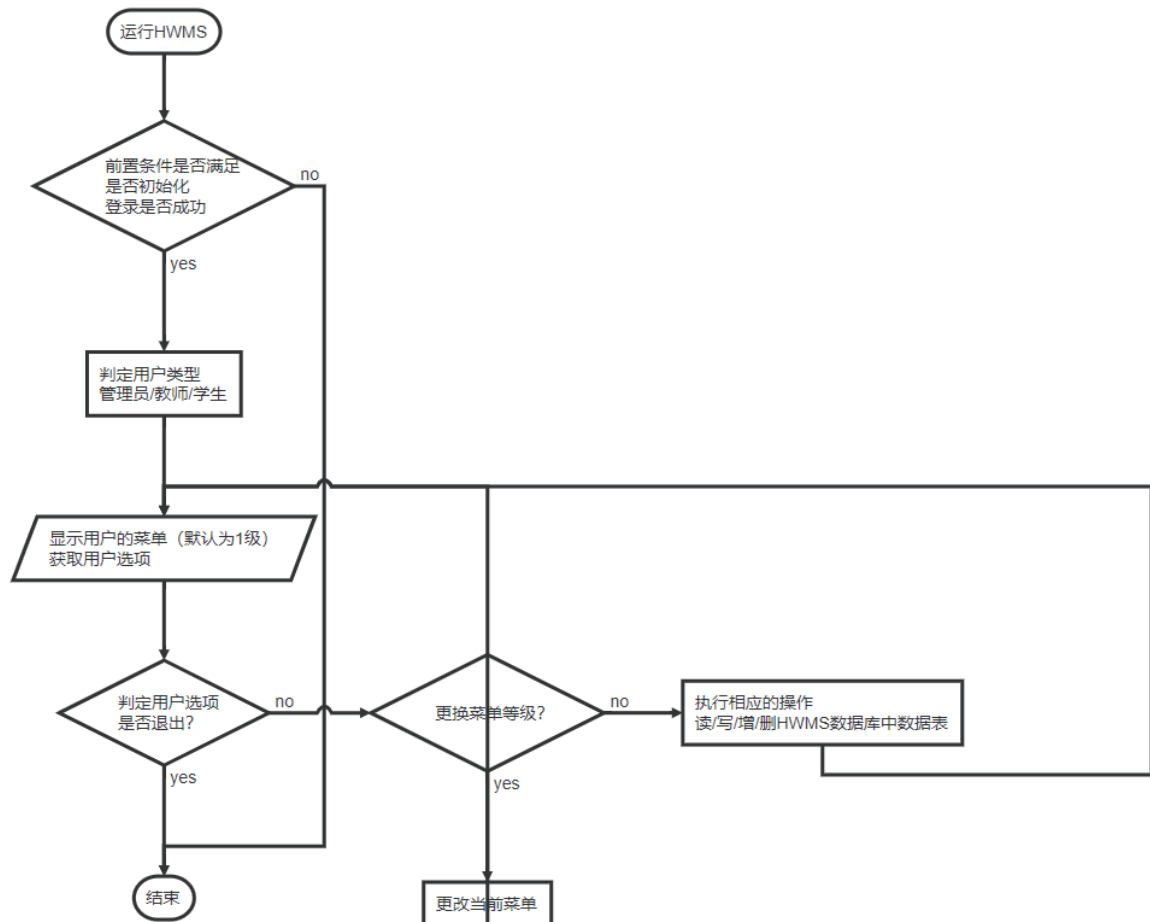
HWMS 的设计利用 Bash Shell 脚本语言完成；同时利用了

- 相关的 whiptail 图形包进行可视化交互。

- MySQL Ver 8.0.21 for Linux on x86\_64 (MySQL Community Server - GPL)  
数据库管理系统保存账户、作业、课程等的信息。

## 整体结构 / 数据结构与算法

首先简单介绍 HWMS 程序的**整体结构/算法**。



运行 HWMS，程序经过相关的判定后进入主循环，利用 whiptail 显示菜单界面，等待用户选择选项。随后程序判定用户的选项，共有 3 类：

- 退出
- 更换菜单等级
- 执行操作

如果退出，程序结束；如果更换菜单等级，则更换后重启主循环，显示新的菜单；如果执行操作，则程序可能会从 whiptail 文本框读取用户的输入，随后执行相应的操作，如有必要会利用 mysql 命令执行 mysql 语句，读写增删 HWMS 数据库中的表，完成交互后回到当前菜单继续循环。

HWMS 用到的**数据结构**极其简单，因此只介绍用到的**数据表结构**；各表结构如下，创建语句一同给出。

- 管理员账号表 ADMIN(AID, PASSWORD, ANAME)



```
create table ADMIN(  
    AID char(30) primary key,  
    PASSWORD char(30),  
    ANAME char(30));
```

- 教师账号表 TEACHER(TID, PASSWORD, TNAME)

```
create table TEACHER(  
    TID char(30) primary key,  
    PASSWORD char(30),  
    TNAME char(30));
```

- 学生账号表 STUDENT(SID, PASSWORD, SNAME)

```
create table STUDENT(  
    SID char(30) primary key,  
    PASSWORD char(30),  
    SNAME char(30));
```

- 课程表 COURSE(CID, CNAME, TID)

```
create table COURSE(  
    CID char(30) primary key,  
    CNAME char(30),  
    TID char(30));  
  
alter table COURSE add constraint COURSE_TID_REF foreign  
key(TID) references TEACHER(TID) on delete cascade on  
update cascade;
```

- 课程信息表 COURSE\_INFO(CID, IID, TXT)

```
create table COURSE_INFO(  
    CID char(30),  
    IID int,  
    TXT char(200));  
  
alter table COURSE_INFO add constraint  
COURSE_INFO_CID_REF foreign key(CID) references  
COURSE(CID) on delete cascade on update cascade;
```

- 学生选课表 SC(SID, CID)

```
create table SC(
    SID char(30),
    CID char(30));

alter table SC add constraint SC_SID_REF foreign key(SID)
references STUDENT(SID) on delete cascade on update
cascade;

alter table SC add constraint SC_CID_REF foreign key(CID)
references COURSE(CID) on delete cascade on update
cascade;
```

- 作业信息表（每个课程的作业均从 1 开始递增编号，即不同课程的作业 HWID 可能相同）HW(CID, HWID, TYPE, TXT)

```
create table HW(
    CID char(30),
    HWID int,
    TXT char(200));

alter table HW add constraint HW_CID_REF foreign key(CID)
references COURSE(CID) on delete cascade on update
cascade;
```

作业完成情况表（每个作业 or 实验都会有单独的一张表，记录学生完成情况、教师登记的评分）HW\_INFO\_CID\_HWID(SID, STATUS, SCORE)

```
create table HW_INFO_${CID}_${HWID}(
    SID char(30) primary key,
    DONE boolean,
    TXT char(300),
    SCORE int);

alter table HW_INFO_${CID}_${HWID} add constraint
HW_INFO_${CID}_${HWID}_REF foreign key(SID) references
STUDENT(SID) on delete cascade on update cascade;
```

```
create table HW_INFO_C1_3(
    SID char(30) primary key,
    DONE boolean,
```

```

        TXT char(255),
        SCORE int);

alter table HW_INFO_C1_3 add constraint HW_INFO_C1_3_REF
foreign key(SID) references STUDENT(SID) on delete cascade
on update cascade;

```

## 功能模块

### 初始化模块

程序开头有一部分初始化模块,用于在—initialize 参数传入时初始化 HWMS 数据库:

```

#初始化数据库阶段
sql="drop database HWMS;" #不管是否存在, 都先抹除 HWMS 数据库
mysql -u"${SQLUSERNAME}" -p"${SQLPASSWORD}" -e"$sql" 1>/dev/null 2>&1
#创建新的空 HWMS 数据库、相关的表
sql="create database HWMS;\
    use HWMS; \
    create table ADMIN( \
    AID char(30) primary key, \
    PASSWORD char(30), \
    ANAME char(30)); \
    create table TEACHER( \
    TID char(30) primary key, \
    PASSWORD char(30), \
    TNAME char(30)); \
    create table STUDENT( \
    SID char(30) primary key, \
    PASSWORD char(30), \
    SNAME char(30)); \
    create table COURSE( \
    CID char(30) primary key, \
    CNAME char(30), \
    TID char(30)); \
    alter table COURSE add constraint COURSE_TID_REF foreign key(TI
D) references TEACHER(TID) on delete cascade on update cascade; \
    create table COURSE_INFO( \
    CID char(30), \
    IID int, \
    TXT char(200)); \
    alter table COURSE_INFO add constraint COURSE_INFO_CID_REF fore
ign key(CID) references COURSE(CID) on delete cascade on update cascade
; \

```

```

        create table SC( \
        SID char(30), \
        CID char(30)); \
        alter table SC add constraint SC_SID_REF foreign key(SID) refer
ences STUDENT(SID) on delete cascade on update cascade; \
        alter table SC add constraint SC_CID_REF foreign key(CID) refer
ences COURSE(CID) on delete cascade on update cascade; \
        create table HW( \
        CID char(30), \
        HWID int, \
        TXT char(200)); \
        alter table HW add constraint HW_CID_REF foreign key(CID) refer
ences COURSE(CID) on delete cascade on update cascade;"
mysql -u"${SQLUSERNAME}" -p"${SQLPASSWORD}" -e"$sql" 1>/dev/null 2>&1
exitstatus=$?
if [ $exitstatus = 1 ]
then
    echo "Something went wrong on the database!"
    exit 0
fi
#创建root 管理员账户
sql="use HWMS;\
    insert into ADMIN values ('root','123456','TOP')"
mysql -u"${SQLUSERNAME}" -p"${SQLPASSWORD}" -e"$sql" 1>/dev/null 2>&1
exitstatus=$?
if [ $exitstatus = 1 ]
then
    echo "Something went wrong on the database!"
    exit 0
else

```

该部分代码与上一节中的各表创建语句一致，负责各表的创建、部分数据的初始化。

## 登录模块

登录模块的思路很简单；在 ADMIN、STUDENT、TEACHER 表中，有各类用户的用户名、密码信息，登录时根据用户的输入在 MySQL 中查找 3 张表是否有一致的信息；如有符合的即可登录，否则无法登录！

## 菜单模块

所有的多选菜单均与下面的代码结构类似；利用 whiptail，可以提供图形界面的菜单。

```

OPTION=$(whiptail --title "Administrator" --menu "Hello, $userrealname.
What do you want to do?" 15 60 5 \

```

```

"1" "Teacher Account" \
"2" "Administrator Account" \
"3" "Courses" \
"4" "Personal Profile" \
"5" "Quit" --cancel-button EXIT --ok-button OK 3>&1 1>&2 2>
&3)

exitstatus=$?
case "$OPTION" in
    1)
        cmd="teacher_account"
        ;;
    2)
        cmd="admin_account"
        ;;
    3)
        cmd="admin_course"
        ;;
    4)
        cmd="admin_personal_profile"
        ;;
    5)
        cmd="quit"
        ;;
    *)
        ;;
esac
if [ $exitstatus = 1 ]
then
    cmd="exit"
fi

```

随后根据\$cmd 变量的值，执行相应的操作，如：

```

elif [[ $cmd == "admin_account" ]]
then
    OPTION1=$(whiptail --title "Administrator Account" --menu "Please choose an operation on admin account:" 15 60 4 \
        "1" "List" \
        "2" "Add" \
        "3" "Edit" \
        "4" "Delete" --cancel-button BACK --ok-button OK 3>&1 1>&2 2>&
3)

    exitstatus=$?
    #用户选择 cancel，返回一级目录
    if [ $exitstatus = 1 ]

```

```

then
    cmd=""
    continue
fi

accountOptionJudge $OPTION1 0

continue #继续循环

```

注意到

```
accountOptionJudge $OPTION1 0
```

为函数调用，调用了预先定义的 accountOptionJudge 函数（具体内容参见源程序代码），执行相应的操作。

## MySQL 交互模块

与 MySQL 数据库的交互相当重要。预先配置好 MySQL 账户：

```

SQLUSERNAME="guest"
SQLPASSWORD="123456"

```

后，通过下面的语句可以查询用户名、用户密码是否在 STUDENT 数据库中（即是否可以登录）。

```

sql="use HWMS; select * from STUDENT where SID='$username' and PASSWORD
='$password';"
tmp=`mysql -u"${SQLUSERNAME}" -p"${SQLPASSWORD}" -e"$sql" 2>/dev/
null | grep "$username" | wc -w`

```

其中\$tmp 如果为 0 则表示没有改用户名、密码信息。

再以查询课程的模块为例：

#查看所有课程（在命令行界面）

```

--"
sql="use HWMS; select TID, CID, CNAME from COURSE;"
echo "-----All Courses-----"

mysql -u"${SQLUSERNAME}" -p"${SQLPASSWORD}" -e"$sql" 2>/dev/
null

echo "-----Press any key to continue-----"
--"

read -n1 -s
;;

```

同样的，通过 mysql -u -p -e 的命令可以执行\$sql 变量中存放的 sql 语句，并打印到命令行。