目录 MULU

一、小组分工

（一）小组成员

1. 叶俊杰 19307130140 19 级计算机科学与技术
2. 崔晨昊 19307130084 19 级计算机科学与技术

（二）分工内容

1. 叶俊杰：系统后端实现，包括数据库建立、视图创建、函数、过程、触发器等的实现，以及具体后端函数的实现等；E-R 图绘制；实验报告撰写；整体功能规划与测验。
具体文件包括：create.sql, insert.py, select.py, update.py, delete.py, user.py, analyse.py，初期项目报告, Introduction, E-R Graph。
2. 崔晨昊：系统前端实现，包括前端页面建立，系统前后端接口，前端用户注册登录的实现等；架构前端功能规划与测验。
具体文件包括：views.py，access.html，admin.html，analysis.html，canteen.html，leave.html，login,html, register.html, user.html, urls.py, README.md。

二、需求分析

（一）项目题目

1. 校园一卡通管理系统的设计与实现

（二）项目背景

1. 为校园设计一卡通管理系统，实现对相关人员借助校园一卡通进行食堂消费、进出校、住宿等的统一管理。
2. 利用 django 架构完成前后端分离的数据库应用系统，同时供给管理员及普通用户使用。

（三）主要功能

1. 用户管理
   1）数据库用户分为超级管理员用户和普通管理员用户。超级管理员除可对所有信息进行查询、处理外，还能创建新的用户和查看所有用户的资料；普通管理员用户的权限由超级管理员用户创建时指定。
   2）超级管理员用户在系统完成时便已经存在（即其用户名和密码已经存在于数据库中）。而普通管理员用户的用户名和密码需要由超级管理员用户来创建。

3）用户的密码不能以明文形式保存于数据库中，而必须先加密，加密方式为 MD5。
4）系统所有功能只有用户登录了才能进行操作，本实验最终作为数据库应用系统，后端始终以超级管理员用户进行登录，若有需要进行数据库系统后台管理，可使用 user.py 创建新用户进行管理。
5）前端页面需要在对应数据库用户已登录的情况下再注册前端用户并登录，并根据实际需要区分应用用户中的管理员及普通用户（包括学生、教师、其他人员），其中管理员用户在数据库创建时已初始化设定（用户名：admin，密码：000000）。
6）前端管理员用户可以对数据库相应查询进行查询、管理，普通用户仅允许对少数数据的更新与查询。具体功能见后续描述。

2. 登录与退出
1）登录：前端界面需要用户输入用户名与对应密码进行登录，其中，管理员用户（admin）相关信息在数据库建立时已存入数据库，如需修改应通过数据库管理员进行，其余用户通过管理员用户注册得到。所有前端功能都需要用户登录后方可使用。
2）退出：前端用户可通过点击退出登录按钮退出应用，返回登录界面。
3）前端登录退出操作均采用 cookie 技术，即登录时自动设置 cookie，退出时自动删除 cookie。

3. 系统信息管理与查询，此功能提供给管理员用户
1）信息查询
i. 所有学生：管理员用户可查询数据库内所有学生的相关信息，包括学工号，姓名，入学日期，班级，并以表格方式在前端展示。
ii. 所有教师：管理员用户可查询数据库内所有教师的相关信息，包括学工号，姓名，出生日期，职称，并以表格方式在前端展示。
iii. 所有其他人员：管理员用户可查询数据库内所有其他人员的相关信息，包括学工号，姓名，工作，并以表格方式在前端展示。
iv. 出入校记录：管理员用户可查询数据库内指定时间段（由管理员用户指定）内所有校门出入校的相关信息，包括学工号，姓名，校门，时间，进/出，以时间升序排序，并以表格方式在前端展示。
v. 寝室门禁记录：管理员用户可查询数据库内指定时间段（由管理员用户指定）内所有宿舍门禁记录的相关信息，包括学工号，姓名，宿舍，时间，以时间升序排序，并以表格方式在前端展示。
vi. 餐厅消费记录：管理员用户可查询数据库内指定时间段（由管理员用户指定）内所有餐厅消费记录的相关信息，包括学工号，姓名，餐厅，菜肴，时间，以时间升序排序，并以表格方式在前端展示。
2）信息更新
i. 密码重置：管理员用户可对数据库内任一人员前端登陆密码进行重置，重置后登陆密码为 000000。
ii. 移除权限：管理员用户可移除数据库内任一人员的消费、进出校、宿舍门禁权限。此权限可通过“授予权限“功能恢复。
iii. 授予权限：管理员用户可授予或恢复数据库内任一人员的消费、进出校、宿舍门禁权限。此权限可通过“移除权限“功能移除。
iv. 寝室分配：管理员用户可为数据库内任一人员分配或更新所在寝室信息。
v. 学生班级更新：管理员用户可为数据库内任一学生更新所在班级信息。

vi. 教师职称更新：管理员用户可为数据库内任一教师更新职称信息。
vii. 外来人员工作更新：管理员用户可为数据库内任一外来人员更新工作。
viii. 移除人员：管理员用户可删除数据库内任一人员所有相关信息。

4. 餐厅信息管理，此功能提供给管理员用户
   1） 新建餐厅：管理员用户可通过指定新餐厅的餐厅号、餐厅名、负责人、联系方式等信息新建餐厅。
   2） 删除餐厅：管理员用户可删除现存的任一餐厅及其相关的所有信息。
   3） 更新餐厅信息：管理员用户可更新任一现有餐厅的餐厅名、负责人、联系方式等及其相关信息。

5. 校门信息管理，此功能提供给管理员用户
   1） 新建校门：管理员用户可通过指定新校门的校门号、校门名、负责人、联系方式等信息新建校门。
   2） 删除校门：管理员用户可删除现存的任一校门及其相关的所有信息。
   3） 更新餐厅信息：管理员用户可更新任一现有校门的校门名、负责人、联系方式等及其相关信息。

6. 宿舍信息管理，此功能提供给管理员用户
   1） 新建宿舍：管理员用户可通过指定新校门的宿舍号、负责人、联系方式、层数等信息新建宿舍。
   2） 删除宿舍：管理员用户可删除现存的任一宿舍及其相关的所有信息，所有原先被分配在该寝室的用户寝室信息将置空，如有需要，应重新分配。
   3） 更新宿舍信息：管理员用户可更新任一现有宿舍的负责人、联系方式、层数等及其相关信息。

7. 数据分析，此功能提供给管理员用户
   1） 各校门出入校记录：管理员用户可查询某时间段内（由管理员用户指定）各校门出入校记录总体情况，并以图表方式显示，可在柱状图与折线图之间切换。
   2） 各寝室门禁记录：管理员用户可查询某时间段内（由管理员用户指定）各寝室门禁记录总体情况，并以图表方式显示，可在柱状图与折线图之间切换。
   3） 各餐厅营业额：管理员用户可查询某时间段内（由管理员用户指定）各餐厅营业额情况，并以图表方式显示，可在柱状图与折线图之间切换。
   4） 各菜肴销售量：管理员用户可查询某时间段内（由管理员用户指定）各菜肴销售量情况，并以图表方式显示，可在柱状图与折线图之间切换。
   5） 各职称教师人数：管理员用户可查询出生日期为某时间段内（由管理员用户指定）的各职称教师人数，并以图表方式显示，可在柱状图与折线图之间切换。
   6） 各班级学生人数：管理员用户可查询入学日期为某时间段内（由管理员用户指定）的各班级学生人数，并以图表方式显示，可在柱状图与折线图之间切换。
   7） 各楼寝居住人数：管理员用户可查询最新开卡日期为某时间段内（由管理员用户指定）的各寝室居住人数，并以图表方式显示，可在柱状图与折线图之间切换。

8. 人员注册，此功能提供给管理员用户
   1） 学生信息注册：管理员用户可通过输入对应学生学工号、姓名、入学日期、班级、

寝室等相关信息向数据库内新增人员及学生信息，同时分配一卡通，登陆密码默认为 000000。

2）教师信息注册：管理员用户可通过输入对应教师学工号、姓名、出生日期、职称等相关信息向数据库内新增人员及教师信息，同时分配一卡通，登陆密码默认为 000000。

3）其他人员信息注册：管理员用户可通过输入对应其他人员学工号、姓名、工作等相关信息向数据库内新增人员及其他人员信息，同时分配一卡通，登陆密码默认为 000000。

9. 一卡通基本信息显示，此功能提供给普通用户

1）余额：普通用户首页页面实时显示个人一卡通内余额。

2）今日已消费：普通用户首页页面实时显示个人一卡通今日已消费金额。

3）寝室：普通用户首页页面实时显示本人所被分配寝室。

4）权限：普通用户首页页面实时显示本人一卡通是否具有出入、消费、门禁权限。

10. 余额充值，此功能提供给普通用户

1）普通用户可通过输入充值金额对一卡通余额进行充值。

11. 密码修改，此功能提供给普通用户

1）普通用户可通过输入新密码修改原有登陆密码。

12. 一卡通挂失，此功能提供给普通用户

1）普通用户可通过挂失补办功能办理新一卡通，新一卡通余额为 0，登录密码需要用户重新设定，其余信息与原卡相同，同时，原卡所有功能将失效，无法登录应用，应用页面自动更换为新卡登录页面。

13. 信息查询，此功能提供给普通用户

1）消费记录：普通用户可查询指定时间段内（由普通用户指定）个人餐厅消费记录，包括学工号、姓名、餐厅、菜肴、时间、消费金额，以时间升序排序，并以表格方式在前端显示。

2）楼寝记录：普通用户可查询指定时间段内（由普通用户指定）个人寝室门禁记录，包括学工号、姓名、宿舍、时间，以时间升序排序，并以表格方式在前端显示。

3）出入校记录：普通用户可查询指定时间段内（由普通用户指定）个人出入校记录，包括学工号、姓名、宿舍、时间，以时间升序排序，并以表格方式在前端显示。

14. 餐厅消费信息处理，此功能提供给普通用户

1）普通用户可在现存餐厅中任一餐厅进行实时消费，若一卡通余额不足将被拒绝，否则将新增个人消费记录，包括学工号、餐厅号、消费时间、菜肴、消费金额等信息。

15. 进出校信息处理，此功能提供给普通用户

1）普通用户可在现存校门中任一校门进行实时进出，将新增个人进出校记录，包括学工号、校门号、进出时间、进出校状态等信息。

16. 宿舍门禁信息处理，此功能提供给普通用户

1) 普通用户可在自己被分配的寝室中进行实时门禁刷卡，若刷卡非本人被分配宿舍将被拒绝，否则将新增个人宿舍门禁记录，包括学工号、宿舍号、门禁时间等信息。

17. 数据备份
    1) 考虑现实中数据丢失或损坏的可能性，该系统设置时即默认每日备份一次数据库内所有信息，以便需要时数据库管理员进行数据恢复。此功能前端不可见。

18. 页面自适应及对应提示
    1) 页面自适应：前端应用页面支持页面放大缩小自适应，增强用户体验感。
    2) 功能提示：前端对用户操作结束后进行成功或失败提示，增强用户体验感。

19. sql 防注入
    1) 由于本系统设计时后端语言采用格式化语句输入，前端输入框提前对输入内容进行判断，实现了双重保护，可以防止 sql 语言注入问题。

20. 防止重复登录同一账号
    1) 当同一个用户在不同 IP 地址下登录时，系统将在前一个 IP 地址账号试图操作时自动将其踢出登录状态，并发出提醒，以提高账号的安全性。
    2) 本功能需要同一用户在不同 IP 地址登录方可作用，故而需在远程服务器网站测试，无法在本机测试，本实验小组已提前使用远程服务器网站进行测试。

21. 备注
    1) 具体关系的信息可见“数据库设计”。

三、数据库设计

（一）概念 E-R 图

1. 见图 3.1.1（另附源文件 E-R Graph.pdf 及 E-R Graph.docx）。

（二）关系模型

1. 表

person(ID, name)
teacher(ID, birthday, rank)
student(ID, enrolmentdt, class)
others(ID, work)
dormitory(dno, dadmin, dtel, dfloor)
card(ID, carddate, remainingsum, passwd, cdno, valid)
canteen(wno, wname, wadmin, wtel)

gate(gno, gname, gadmin, gtel)
consume(wno, ID, consumetm, cuisineid, amount)
record(ID, gno, recordtm, inout)
access(ID, dno, accesstm)

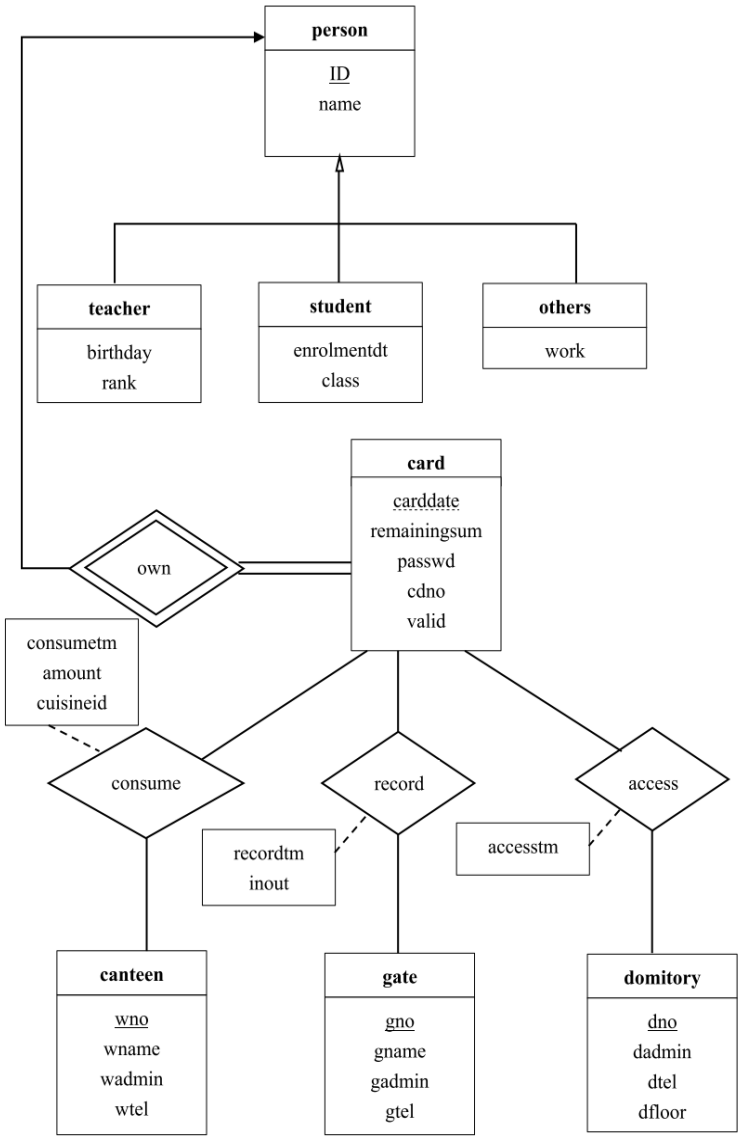


图 3.1.1

3. 视图

v_consume(ID, name, wname, cuisineid, consumetm, amount)
v_record(ID, name, gname, recordtm, inout)
v_access(ID, name, dno, accesstm)

4. 函数

charge

5. 过程

eat

in_and_out

back

6. 触发器

consume_update

update_valid

四、数据库实现

（一）建表

```
--人员信息表
create TABLE person(
    ID VARCHAR(11) PRIMARY KEY,
    name VARCHAR(50) NOT NULL,
    check(length(ID) in (5, 6, 11))
);
--教师信息表
create TABLE teacher(
    ID VARCHAR(11) PRIMARY KEY,
    birthday DATE NOT NULL,
    rank VARCHAR(20),
    foreign key(ID) REFERENCES person(ID) ON DELETE CASCADE ON UPDATE C
ASCADE,
    check(length(ID) = 6)
);
--学生信息表
create TABLE student(
    ID VARCHAR(11) PRIMARY KEY,
    enrolmentdt DATE NOT NULL,
    class VARCHAR(20) NOT NULL,
    foreign key(ID) REFERENCES person(ID) ON DELETE CASCADE ON UPDATE C
ASCADE,
    check(length(ID) = 11)
);
--其他人员信息表
create TABLE others(
    ID VARCHAR(11) PRIMARY KEY,
    work VARCHAR(20) NOT NULL,
```

```
    foreign key(ID) REFERENCES person(ID) ON DELETE CASCADE ON UPDATE C
ASCADE,
    check(length(ID) = 5)
);
--宿舍信息表
create TABLE domitory(
    dno INT PRIMARY KEY,
    dadmin VARCHAR(50) NOT NULL,
    dtel VARCHAR(11) NOT NULL,
    dfloor INT NOT NULL
);
--一卡通信息表
create TABLE card(
    ID VARCHAR(11),
    remainingsum NUMERIC(10, 2) DEFAULT 0,
    carddate TIMESTAMP DEFAULT now(),
    passwd VARCHAR(50) DEFAULT '000000',
    cdno INT DEFAULT null,
    valid INT DEFAULT 1,
    PRIMARY KEY(ID, carddate),
    foreign key(ID) REFERENCES person(ID) ON DELETE CASCADE ON UPDATE C
ASCADE,
    foreign key(cdno) REFERENCES domitory(dno) ON DELETE
    SET DEFAULT ON UPDATE CASCADE
);
--餐厅信息表
create TABLE canteen(
    wno INT PRIMARY KEY,
    wname VARCHAR(50) NOT NULL,
    wadmin VARCHAR(50) NOT NULL,
    wtel VARCHAR(11) NOT NULL
);
--校门信息表
create TABLE gate(
    gno INT PRIMARY KEY,
    gname VARCHAR(50) NOT NULL,
    gadmin VARCHAR(50) NOT NULL,
    gtel VARCHAR(11) NOT NULL
);
--消费信息表
create TABLE consume(
    wno INT NOT NULL,
    ID VARCHAR(11) NOT NULL,
    consumetm TIMESTAMP DEFAULT now(),
```

```
    cuisineid INT NOT NULL,
    amount numeric(10, 2) NOT NULL,
    PRIMARY KEY(wno, ID, consumetm),
    foreign key(ID) REFERENCES person(ID) ON DELETE CASCADE ON UPDATE C
ASCADE,
    foreign key(wno) REFERENCES canteen(wno) ON DELETE CASCADE ON UPDAT
E CASCADE
);
--进出校记录表
create TABLE record(
    ID VARCHAR(11) NOT NULL,
    gno INT NOT NULL,
    recordtm TIMESTAMP DEFAULT now(),
    inout VARCHAR(3) NOT NULL,
    PRIMARY KEY(ID, gno, recordtm),
    foreign key(ID) REFERENCES person(ID) ON DELETE CASCADE ON UPDATE C
ASCADE,
    foreign key(gno) REFERENCES gate(gno) ON DELETE CASCADE ON UPDATE C
ASCADE,
    check(inout in('in', 'out'))
);
--门禁信息表
create TABLE access(
    ID VARCHAR(11) NOT NULL,
    dno int NOT NULL,
    accesstm TIMESTAMP DEFAULT now(),
    PRIMARY KEY(ID, dno, accesstm),
    foreign key(ID) REFERENCES person(ID) ON DELETE CASCADE ON UPDATE C
ASCADE,
    foreign key(dno) REFERENCES domitory(dno) ON DELETE CASCADE ON UPDA
TE CASCADE
);
```

（二）视图

```
--视图：消费信息
create view v_consume as(
    select ID,
        name,
        wname,
        cuisineid,
        consumetm,
        amount
```

```
    from person
        natural join card
        natural join canteen
        natural join consume
    where valid <> 0
);
--视图：进出校记录
create view v_record as(
    select ID,
        name,
        gname,
        recordtm,
        inout
    from person
        natural join card
        natural join gate
        natural join record
    where valid <> 0
);
--视图：门禁记录
create view v_access as(
    select ID,
        name,
        dno,
        accesstm
    from person
        natural join card
        natural join access
    where valid <> 0
);
```

（三）触发器

```
--触发器：消费后及时更新余额
create or replace function consume_trigger() returns trigger as $$ begi
n
update card
set remainingsum = remainingsum - new.amount
where card.ID = new.ID
    and card.valid = 1;
return new;
end;
$$ language plpgsql;
```

```
create trigger consume_update
after
insert on consume for each row execute procedure consume_trigger();
--触发器：卡挂失后及时将原卡置为无效
create or replace function valid_trigger() returns trigger as $$ begin
update card
set valid = 0
where card.ID = new.ID;
return new;
end;
$$ language plpgsql;
create trigger update_valid before
insert on card for each row execute procedure valid_trigger();
```

（四）存储过程

```
--过程：食堂就餐
create or replace procedure eat(
        in pID varchar(11),
        in pwno int,
        in pcuisineid int,
        in pamount numeric(10, 2)
    ) as $$ begin if pID not in(
        select ID
        from card
        where valid = 1
    ) then raise notice 'Failed. The ID is error.';
elsif pwno not in(
    select wno
    from canteen
) then raise notice 'Failed. The wno is error.';
elsif pamount > (
    select remainingsum
    from card
    where card.ID = pID
        and card.valid = 1
) then raise notice 'Failed. The balance is not enough.';
else
insert into consume(wno, ID, cuisineid, amount)
values(pwno, pID, pcuisineid, pamount);
end if;
end;
$$ language plpgsql;
```

```
--过程：进出校门
create or replace procedure in_and_out(
        in pID varchar(11),
        in pgno int,
        in pinout varchar(3)
    ) as $$ begin if pID not in(
        select ID
        from card
        where valid = 1
    ) then raise notice 'Failed. The ID is error.';
elsif pgno not in(
    select gno
    from gate
) then raise notice 'Failed. The gno is error.';
elsif pinout not in('in', 'out') then raise notice 'Failed. The inout i
s wrong.';
else
insert into record(ID, gno, inout)
values(pID, pgno, pinout);
end if;
end;
$$ language plpgsql;
--过程：回寝
create or replace procedure back(
        in pID varchar(11),
        in pdno int
    ) as $$ begin if pID not in(
        select ID
        from card
        where valid = 1
    ) then raise notice 'Failed. The ID is error.';
elsif pdno not in(
    select dno
    from domitory
) then raise notice 'Failed. The dno is error.';
elsif pdno not in(
    select cdno
    from card
    where card.ID = pID
        and card.valid = 1
)
or (
    select cdno
    from card
```

```
    where card.ID = pID
        and card.valid = 1
) is null then raise notice 'Failed. The ID can not access to the gno.'
;
else
insert into access(ID, dno)
values(pID, pdno);
end if;
end;
$$ language plpgsql;
```

（五）函数

```
--函数：一卡通余额充值
create or replace function charge(fID varchar(11), famount numeric(10,
2)) returns numeric as $$ begin
update card
set remainingsum = remainingsum + famount
where fID = card.ID
    and card.valid <> 0;
return (
    select remainingsum
    from card
    where fID = card.ID
        and card.valid <> 0
);
end;
$$ language plpgsql;
```

五、系统实现

（一）关键代码

1. 总说
   1) 由于本实验小组采用 django 架构，前后端代码分离，故下述所列文件均为关键代码，不在此一一粘贴。

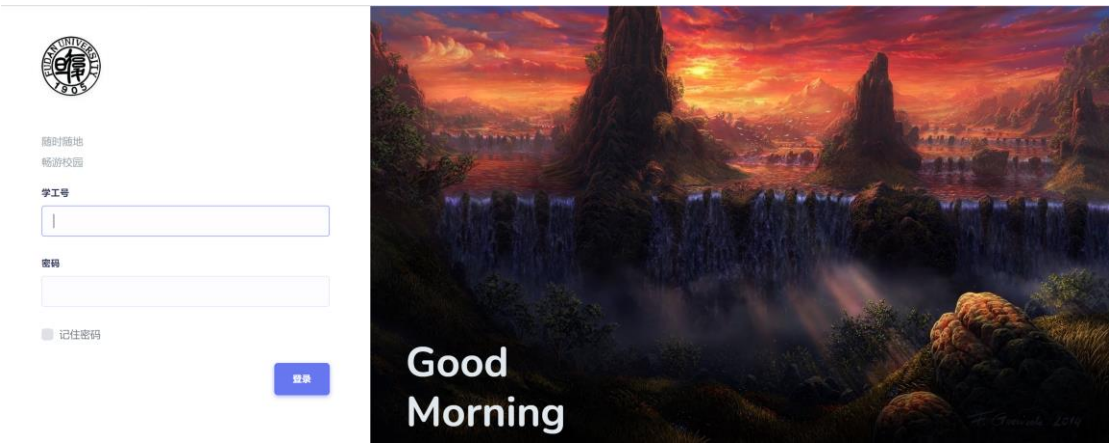
2. 后端关键代码文件及简要说明
   1) create.sql：数据库建立和初始化的 sql 脚本，包括表格、视图、过程、函数、触发器的建立及数据库的记录初始化，其中每一项的作用在文件中已有注释说明。
   2) insert.py：关于对数据库中关系进行记录插入等的所有函数。

3) select.py：关于对数据库中各种关系、视图的查询及复合查询等的功能函数，具体每一个函数功能在文件中已有注释说明。
4) update.py：关于对数据库中各种关系的更新等的功能函数，具体每一个函数功能在文件中已有注释说明。
5) delete.py：关于对数据库中各种关系的记录删除等的功能函数，具体每一个函数功能在文件中已有注释说明。
6) user.py：关于创建新数据库用户的功能函数，此函数应有数据库管理员根据实际需要进行完善与后台调用。
7) analyse.py：关于对数据库信息进行特定查询并到处 html 文件供前端调用的功能函数，具体每一个函数功能在文件中已有注释说明。

3. 前端关键代码及简要说明
1) access.html：渲染产生前端门禁页面。
2) admin.html：渲染产生前端管理员用户首页界面。
3) analysis.html：渲染产生前端分析界面。
4) canteen.html：渲染产生前端餐厅界面。
5) leave.html：渲染产生前端出入界面。
6) login.html：渲染产生前端登录界面。
7) register.html：渲染产生前端注册界面。
8) user.html：渲染产生前端普通用户首页界面。

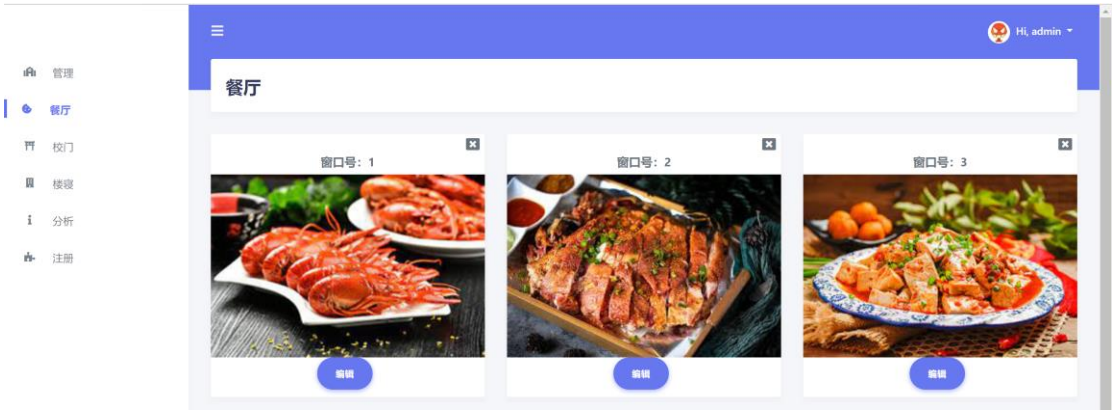
（二）典型界面

1. 登录界面

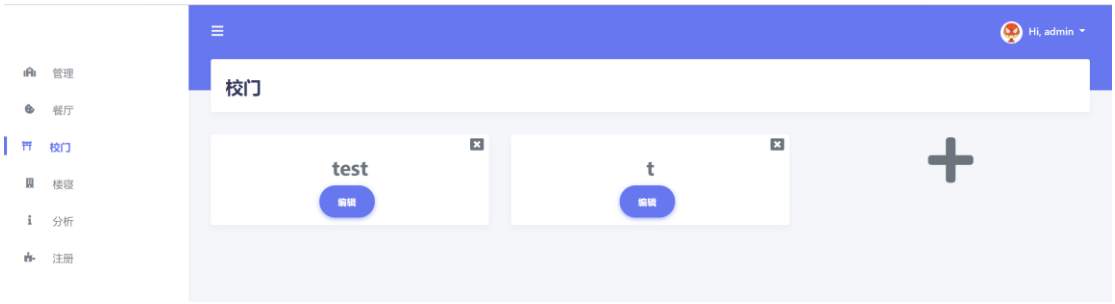


2. 管理员用户首页界面

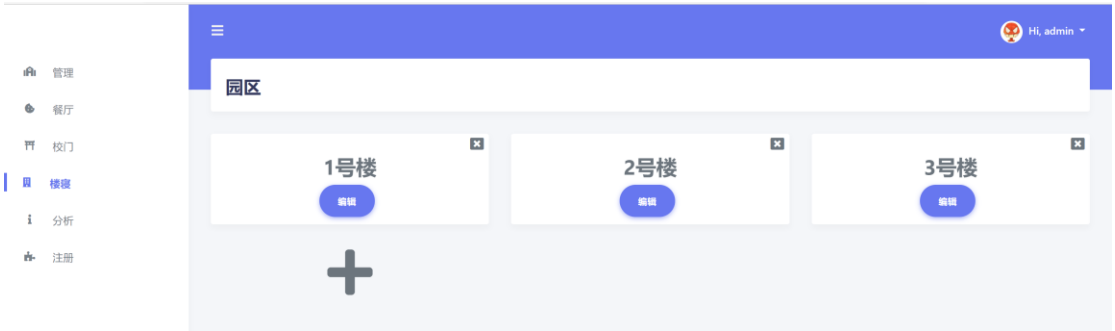
3. 餐厅管理界面



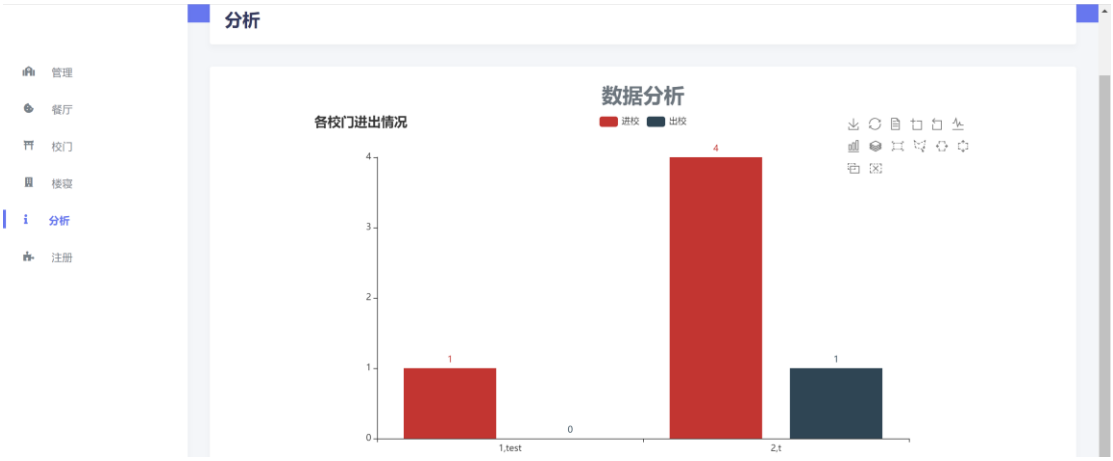
4. 校门管理界面



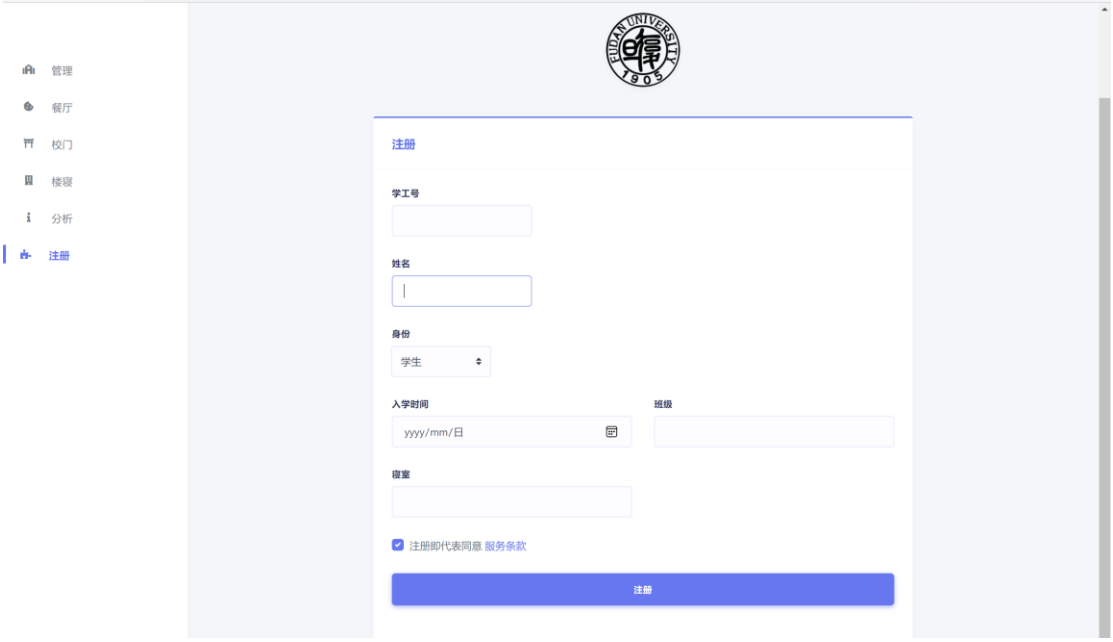
5. 宿舍管理界面



6. 数据分析界面

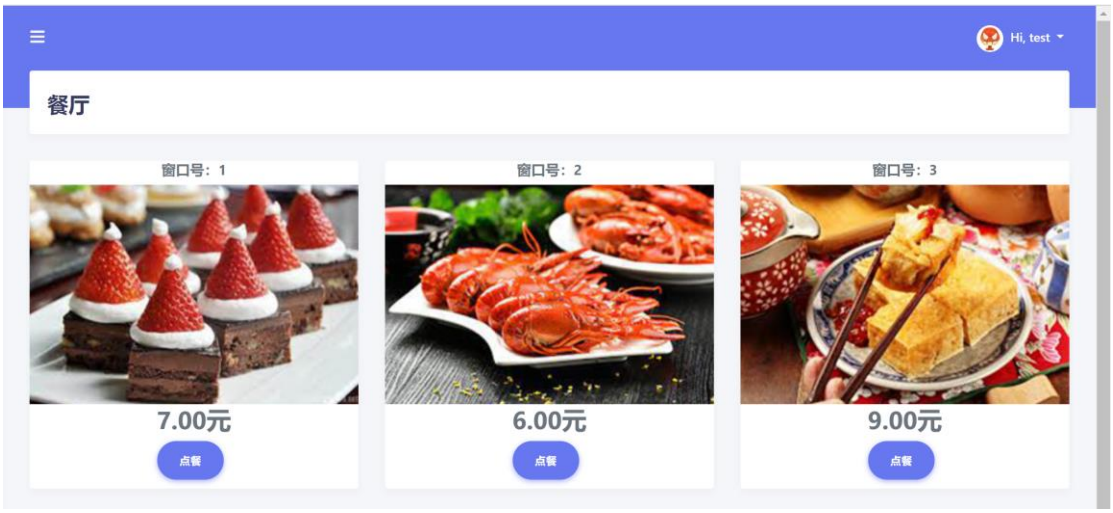


7. 注册界面

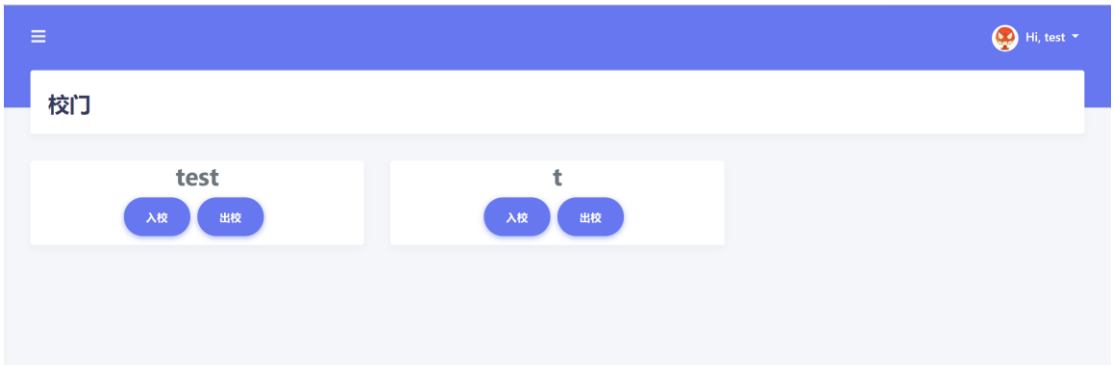


8. 普通用户首页界面

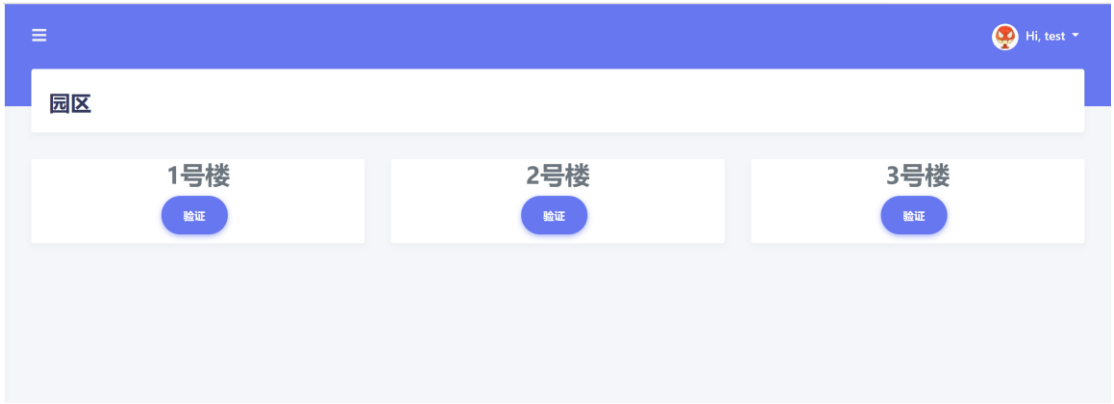
9. 餐厅消费界面



10. 出入校门界面



11. 宿舍门禁页面



六、总结

1. 通过本次实验，小组成员进一步熟悉与掌握了数据库语言的使用，完成了对所学 sql 语言知识点的全面应用，在实践中增进理解，较好地达到了预期目标。
2. 通过本次实验，小组成员进一步了解与熟悉了前端网页的编写方法，通过前端更加清晰地展示后端功能，较好地提升了用户体验感。

3. 通过本次实验，小组成员互相配合，通过前后端分离编写与组织，进一步提升了小组成员的合作能力与协调能力，为更好地完成实验奠定了基础。