在线抽奖系统

项目目标

- 复习Spring相关知识：IOC/DI（Bean的注册、配置与使用）、SpringMVC、AOP
- 复习SpringBoot的配置使用
- 学习系统设计，结合并深入复习学过的知识，如异常，HTTP数据传输流程等等
- 复习并巩固Mybatis的基本使用，理解数据转换时对应的映射关系
- 学习项目开发流程

开发环境与技术栈

- Windows/Mac/Linux
- Maven
- Lombok
- Spring、SpringMVC、SpringBoot
- MySQL、Mybatis、Druid

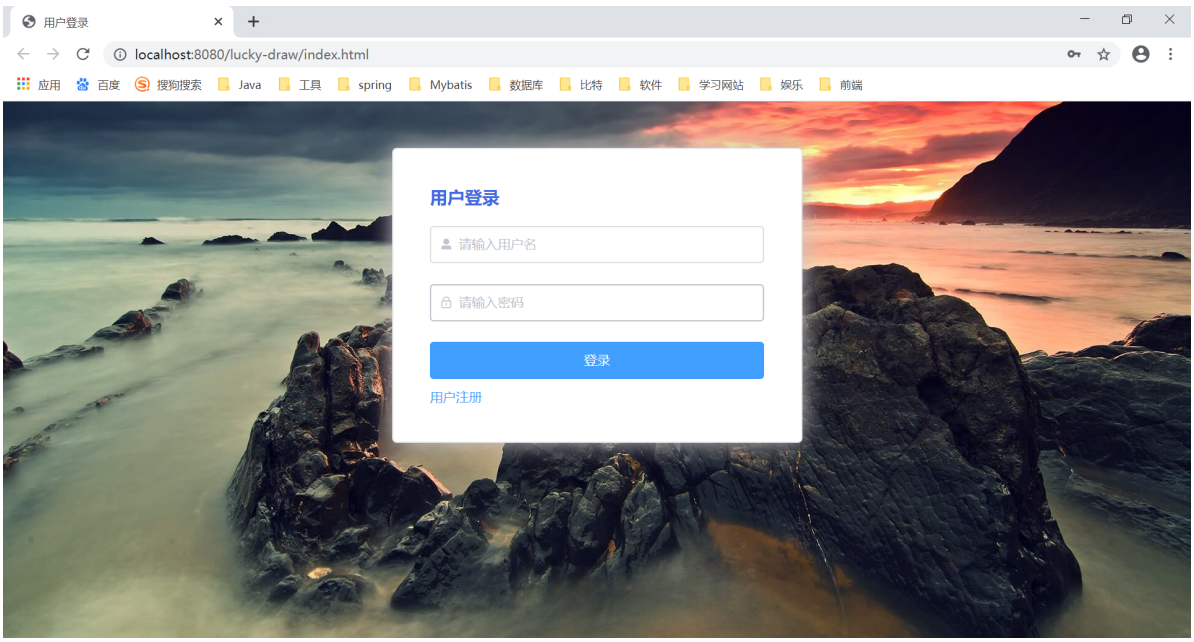
项目功能

主要业务：为公司活动（如年会等）提供在线抽奖功能，满足奖品、抽奖人员的管理，及抽奖活动的需要。

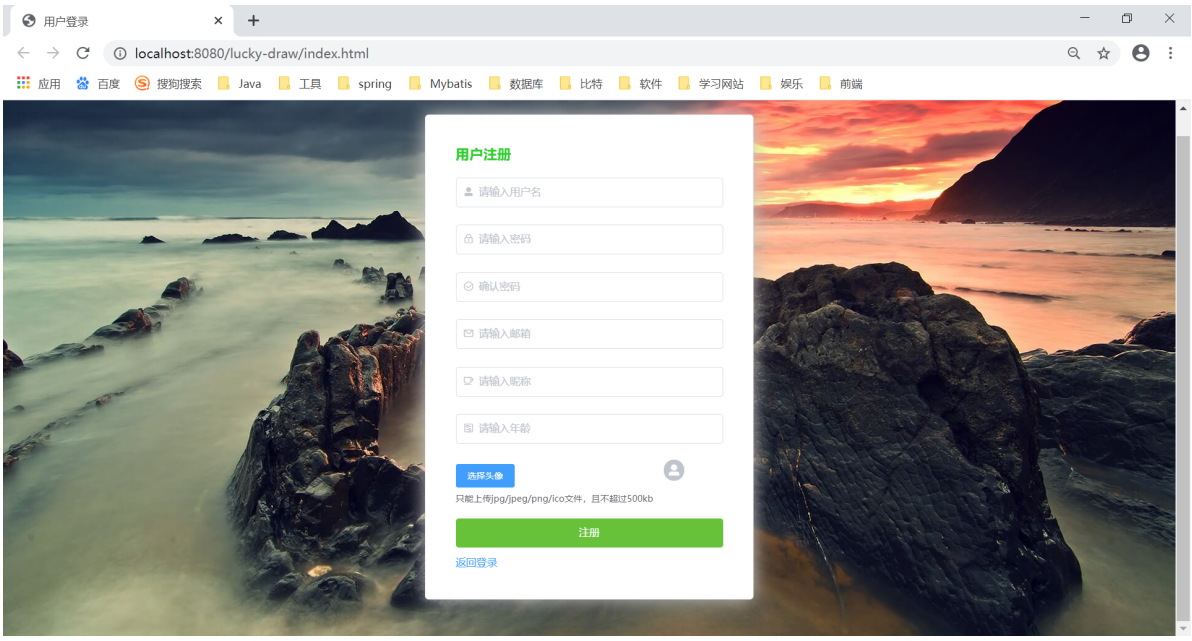
- 用户注册
- 用户登录、会话管理
- 抽奖设置：奖品管理，抽奖人员管理
- 人员抽奖

项目演示

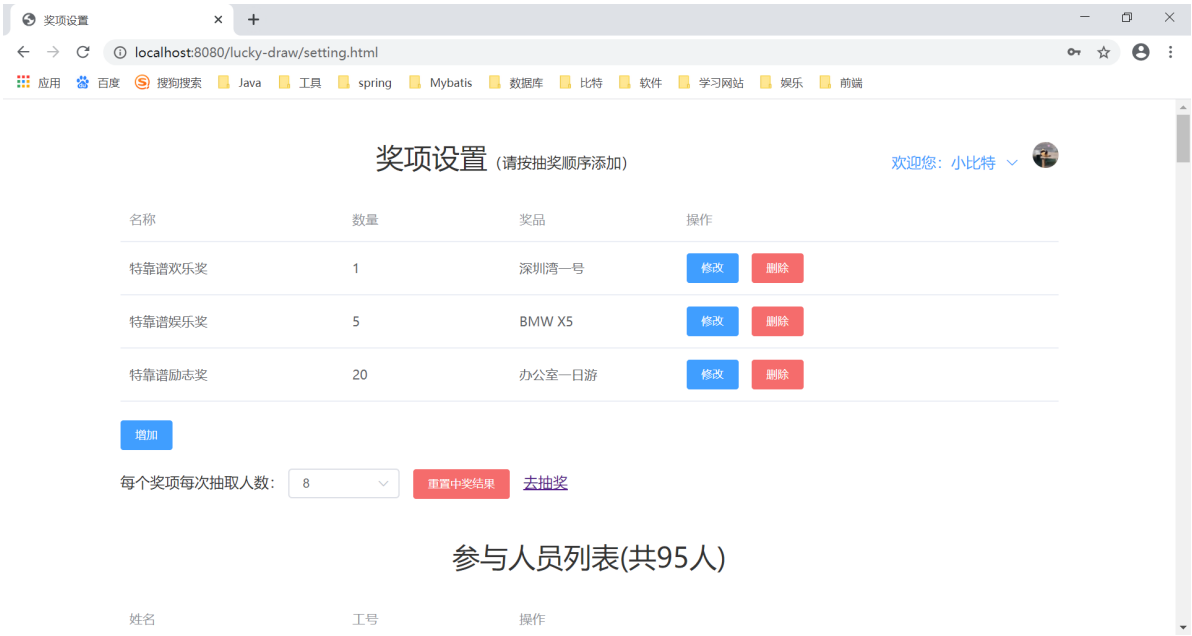
用户登录



用户注册

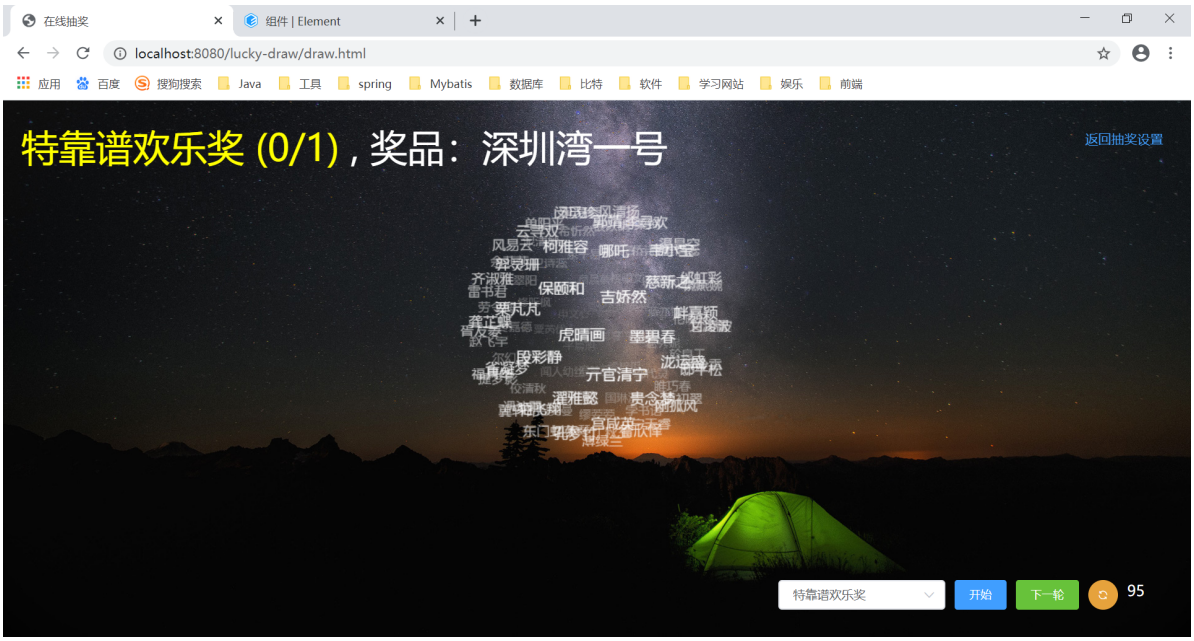


奖项设置



抽奖人员设置

抽奖



数据库设计

数据库表关系图

创建数据库及表

```
drop database if exists lucky_draw;
create database lucky_draw character set utf8mb4;

use lucky_draw;

drop table if exists user;
create table user(
    id int primary key auto_increment,
```

```
    username varchar(20) not null unique comment '用户账号',
    password varchar(20) not null comment '密码',
    nickname varchar(20) comment '用户昵称',
    email varchar(50) comment '邮箱',
    age int comment '年龄',
    head varchar(255) comment '头像url',
    create_time timestamp default NOW() comment '创建时间'
) comment '用户表';

drop table if exists setting;
create table setting(
    id int primary key auto_increment,
    user_id int not null comment '用户id',
    batch_number int not null comment '每次抽奖人数',
    create_time timestamp default NOW() comment '创建时间',
    foreign key (user_id) references user(id)
) comment '抽奖设置';

drop table if exists award;
create table award(
    id int primary key auto_increment,
    name varchar(20) not null comment '奖项名称',
    count int not null comment '奖项人数',
    award varchar(20) not null comment '奖品',
    setting_id int not null comment '抽奖设置id',
    create_time timestamp default NOW() comment '创建时间',
    foreign key (setting_id) references setting(id)
) comment '奖项';

drop table if exists member;
create table member(
    id int primary key auto_increment,
    name varchar(20) not null comment '姓名',
    no varchar(20) not null comment '工号',
    user_id int not null comment '用户id',
    create_time timestamp default NOW() comment '创建时间',
    foreign key (user_id) references user(id)
) comment '抽奖人员';

drop table if exists record;
create table record(
    id int primary key auto_increment,
    member_id int not null,
    award_id int not null,
    create_time timestamp default NOW() comment '创建时间',
    foreign key (member_id) references member(id),
    foreign key (award_id) references award(id)
) comment '学生表';



insert into user(id, username, password, nickname, email, age, head) values (1,
'bit', '123', '小比特', '1111@163.com', 18, 'img/test-head.jpg');

## 数据字典：学生毕业年份
insert into setting(id, user_id, batch_number) values (1, 1, 8);
```

```
insert into award(name, count, award, setting_id) values ('特靠谱欢乐奖', 1, '深圳湾一号', 1);
insert into award(name, count, award, setting_id) values ('特靠谱娱乐奖', 5, 'BMW X5', 1);
insert into award(name, count, award, setting_id) values ('特靠谱励志奖', 20, '办公室一日游', 1);

## 数据字典：学生专业
insert into member(name, no, user_id) values ('李寻欢', '水果刀', 1);
insert into member(name, no, user_id) values ('郭靖', '降猪十八掌', 1);
insert into member(name, no, user_id) values ('韦小宝', '抓?龙爪手', 1);
insert into member(name, no, user_id) values ('风清扬', '孤独九贱', 1);
insert into member(name, no, user_id) values ('哪吒', '喷气式电单车', 1);
insert into member(name, no, user_id) values ('渠昊空', 'no2', 1);
insert into member(name, no, user_id) values ('闵觅珍', 'no2', 1);
insert into member(name, no, user_id) values ('慈新之', 'no3', 1);
insert into member(name, no, user_id) values ('户柔绚', 'no4', 1);
insert into member(name, no, user_id) values ('柯雅容', 'no5', 1);
insert into member(name, no, user_id) values ('郃虹彩', 'no6', 1);
insert into member(name, no, user_id) values ('延易蓉', 'no7', 1);
insert into member(name, no, user_id) values ('吉娇然', 'no8', 1);
insert into member(name, no, user_id) values ('百里惜蕊', 'no9', 1);
insert into member(name, no, user_id) values ('云寻双', 'no10', 1);
insert into member(name, no, user_id) values ('衅嘉颖', 'no11', 1);
insert into member(name, no, user_id) values ('银以晴', 'no12', 1);
insert into member(name, no, user_id) values ('保颐和', 'no13', 1);
insert into member(name, no, user_id) values ('饶燕婉', 'no14', 1);
insert into member(name, no, user_id) values ('单阳平', 'no15', 1);
insert into member(name, no, user_id) values ('墨碧春', 'no16', 1);
insert into member(name, no, user_id) values ('侨诗柳', 'no17', 1);
insert into member(name, no, user_id) values ('羿灵珊', 'no18', 1);
insert into member(name, no, user_id) values ('甘凌波', 'no19', 1);
insert into member(name, no, user_id) values ('希忻然', 'no20', 1);
insert into member(name, no, user_id) values ('虎晴画', 'no21', 1);
insert into member(name, no, user_id) values ('闪雅洁', 'no22', 1);
insert into member(name, no, user_id) values ('风易云', 'no23', 1);
insert into member(name, no, user_id) values ('泷运盛', 'no24', 1);
insert into member(name, no, user_id) values ('沐长菁', 'no25', 1);
insert into member(name, no, user_id) values ('栗芃芃', 'no26', 1);
insert into member(name, no, user_id) values ('义涵蕾', 'no27', 1);
insert into member(name, no, user_id) values ('泥清妙', 'no28', 1);
insert into member(name, no, user_id) values ('亓官清宁', 'no29', 1);
insert into member(name, no, user_id) values ('侯曜曦', 'no30', 1);
insert into member(name, no, user_id) values ('齐淑雅', 'no31', 1);
insert into member(name, no, user_id) values ('邸平松', 'no32', 1);
insert into member(name, no, user_id) values ('泉千易', 'no33', 1);
insert into member(name, no, user_id) values ('段彩静', 'no34', 1);
insert into member(name, no, user_id) values ('伦晓凡', 'no35', 1);
insert into member(name, no, user_id) values ('余莎莎', 'no36', 1);
insert into member(name, no, user_id) values ('贵念梦', 'no37', 1);
insert into member(name, no, user_id) values ('接骊文', 'no38', 1);
insert into member(name, no, user_id) values ('龚芷蝶', 'no39', 1);
insert into member(name, no, user_id) values ('丙冷霜', 'no40', 1);
insert into member(name, no, user_id) values ('卫诗蕊', 'no41', 1);
insert into member(name, no, user_id) values ('濯雅懿', 'no42', 1);
insert into member(name, no, user_id) values ('蓝亦竹', 'no43', 1);
insert into member(name, no, user_id) values ('雷书君', 'no44', 1);
insert into member(name, no, user_id) values ('刚孤风', 'no45', 1);
```

```
insert into member(name, no, user_id) values ('帛晨蓓', 'no46', 1);
insert into member(name, no, user_id) values ('雀凝梦', 'no47', 1);
insert into member(name, no, user_id) values ('於良工', 'no48', 1);
insert into member(name, no, user_id) values ('从翠阳', 'no49', 1);
insert into member(name, no, user_id) values ('宫咸英', 'no50', 1);
insert into member(name, no, user_id) values ('项英光', 'no51', 1);
insert into member(name, no, user_id) values ('胥友菱', 'no52', 1);
insert into member(name, no, user_id) values ('慎初翠', 'no53', 1);
insert into member(name, no, user_id) values ('锺映寒', 'no54', 1);
insert into member(name, no, user_id) values ('貊飞翔', 'no55', 1);
insert into member(name, no, user_id) values ('葛秀妮', 'no56', 1);
insert into member(name, no, user_id) values ('劳令梅', 'no57', 1);
insert into member(name, no, user_id) values ('咎欣怿', 'no58', 1);
insert into member(name, no, user_id) values ('党忆柏', 'no59', 1);
insert into member(name, no, user_id) values ('福月华', 'no60', 1);
insert into member(name, no, user_id) values ('睢巧春', 'no61', 1);
insert into member(name, no, user_id) values ('修听枫', 'no62', 1);
insert into member(name, no, user_id) values ('孔梦竹', 'no63', 1);
insert into member(name, no, user_id) values ('子车悦欣', 'no64', 1);
insert into member(name, no, user_id) values ('赵飞宇', 'no65', 1);
insert into member(name, no, user_id) values ('宁天睿', 'no66', 1);
insert into member(name, no, user_id) values ('申文心', 'no67', 1);
insert into member(name, no, user_id) values ('冀轩昂', 'no68', 1);
insert into member(name, no, user_id) values ('邬代灵', 'no69', 1);
insert into member(name, no, user_id) values ('佟嘉德', 'no70', 1);
insert into member(name, no, user_id) values ('溥绿兰', 'no71', 1);
insert into member(name, no, user_id) values ('改昊昊', 'no72', 1);
insert into member(name, no, user_id) values ('捷梦影', 'no73', 1);
insert into member(name, no, user_id) values ('孛书语', 'no74', 1);
insert into member(name, no, user_id) values ('粟芮优', 'no75', 1);
insert into member(name, no, user_id) values ('东门虹英', 'no76', 1);
insert into member(name, no, user_id) values ('漆梓玥', 'no77', 1);
insert into member(name, no, user_id) values ('尔幻玉', 'no78', 1);
insert into member(name, no, user_id) values ('丁秋玉', 'no79', 1);
insert into member(name, no, user_id) values ('平晨旭', 'no80', 1);
insert into member(name, no, user_id) values ('遇沙羽', 'no81', 1);
insert into member(name, no, user_id) values ('国琳溪', 'no82', 1);
insert into member(name, no, user_id) values ('仪谷枫', 'no83', 1);
insert into member(name, no, user_id) values ('斜尔琴', 'no84', 1);
insert into member(name, no, user_id) values ('澄慧丽', 'no85', 1);
insert into member(name, no, user_id) values ('佼清秋', 'no86', 1);
insert into member(name, no, user_id) values ('缪荽荽', 'no87', 1);
insert into member(name, no, user_id) values ('闻人幼丝', 'no88', 1);
insert into member(name, no, user_id) values ('绍美曼', 'no89', 1);
insert into member(name, no, user_id) values ('回访波', 'no90', 1);
```

初始化数据

```
use stu_dorm;
-- 初始化数据
-- mysql中没有==，是用=号代替==。为了区分=和==，赋值时使用:=
-- 初始化数据
-- mysql中没有==，是用=号代替==。为了区分=和==，赋值时使用:=
set @username:='abc';
set @password:='123';
set @nickname:='风一样的男子🤡';
```

```
set @email:='123@qq.com';

set @building_name:='生楼-';
set @building_desc:='修炼道场(๑_๑)';

set @dorm_desc:='闭关之地( ꒪⌓꒪ )/';

set @dictionary_student_graduate_year='000001';
set @dictionary_student_major='000002';

set @student_name:='小小的梦想🐷';

insert into user(username, nickname, password, email) values (@username,
@nickname, @password, @email);
insert into user(username, nickname, password, email) values (concat(@username,
'1'), concat(@nickname, '1'), @password, @email);
insert into user(username, nickname, password, email) values (concat(@username,
'2'), concat(@nickname, '2'), @password, @email);
insert into user(username, nickname, password, email) values (concat(@username,
'3'), concat(@nickname, '3'), @password, @email);
insert into user(username, nickname, password, email) values (concat(@username,
'4'), concat(@nickname, '4'), @password, @email);
insert into user(username, nickname, password, email) values (concat(@username,
'5'), concat(@nickname, '5'), @password, @email);

insert into building(building_name, building_desc) values (concat('男',
@building_name, '1'), @building_desc);
insert into building(building_name, building_desc) values (concat('男',
@building_name, '2'), @building_desc);
insert into building(building_name, building_desc) values (concat('男',
@building_name, '3'), @building_desc);
insert into building(building_name, building_desc) values (concat('女',
@building_name, '1'), @building_desc);
insert into building(building_name, building_desc) values (concat('女',
@building_name, '2'), @building_desc);
insert into building(building_name, building_desc) values (concat('女',
@building_name, '3'), @building_desc);

## 宿舍
# 宿舍楼1的宿舍
insert into dorm(dorm_no, dorm_desc, building_id) values ('1-0-1', @dorm_desc,
1);
insert into dorm(dorm_no, dorm_desc, building_id) values ('1-0-2', @dorm_desc,
1);
insert into dorm(dorm_no, dorm_desc, building_id) values ('1-0-3', @dorm_desc,
1);
# 宿舍楼2的宿舍
insert into dorm(dorm_no, dorm_desc, building_id) values ('2-0-1', @dorm_desc,
2);
insert into dorm(dorm_no, dorm_desc, building_id) values ('2-0-2', @dorm_desc,
2);
insert into dorm(dorm_no, dorm_desc, building_id) values ('2-0-3', @dorm_desc,
2);
# 宿舍楼3的宿舍
insert into dorm(dorm_no, dorm_desc, building_id) values ('3-0-1', @dorm_desc,
3);
insert into dorm(dorm_no, dorm_desc, building_id) values ('3-0-2', @dorm_desc,
3);
```

```
insert into dorm(dorm_no, dorm_desc, building_id) values ('3-0-3', @dorm_desc,
3);
# 宿舍楼4的宿舍
insert into dorm(dorm_no, dorm_desc, building_id) values ('4-0-1', @dorm_desc,
4);
insert into dorm(dorm_no, dorm_desc, building_id) values ('4-0-2', @dorm_desc,
4);
insert into dorm(dorm_no, dorm_desc, building_id) values ('4-0-3', @dorm_desc,
4);

## 数据字典：学生毕业年份
insert into dictionary(dictionary_key, dictionary_value, dictionary_desc)values
(@dictionary_student_graduate_year, '毕业年份', '学生毕业的年份');

insert into dictionary_tag(dictionary_tag_key, dictionary_tag_value,
dictionary_id)values ('001', '2020届', 1);
insert into dictionary_tag(dictionary_tag_key, dictionary_tag_value,
dictionary_id)values ('002', '2021届', 1);
insert into dictionary_tag(dictionary_tag_key, dictionary_tag_value,
dictionary_id)values ('003', '2022届', 1);
insert into dictionary_tag(dictionary_tag_key, dictionary_tag_value,
dictionary_id)values ('004', '2023届', 1);

## 数据字典：学生专业
insert into dictionary(dictionary_key, dictionary_value, dictionary_desc)values
(@dictionary_student_major, '专业', '学生的专业');
insert into dictionary_tag(dictionary_tag_key, dictionary_tag_value,
dictionary_id)values ('001', '中文系', 2);
insert into dictionary_tag(dictionary_tag_key, dictionary_tag_value,
dictionary_id)values ('002', '英语系', 2);
insert into dictionary_tag(dictionary_tag_key, dictionary_tag_value,
dictionary_id)values ('003', '计算机科学与技术', 2);

insert into student(student_name, student_graduate_year, student_major,
student_email, dorm_id) values (concat(@student_name, 'A1'), '000001001',
'000002001', @email, 1);
insert into student(student_name, student_graduate_year, student_major,
student_email, dorm_id) values (concat(@student_name, 'A2'), '000001001',
'000002001', @email, 1);
insert into student(student_name, student_graduate_year, student_major,
student_email, dorm_id) values (concat(@student_name, 'A3'), '000001001',
'000002001', @email, 1);
insert into student(student_name, student_graduate_year, student_major,
student_email, dorm_id) values (concat(@student_name, 'A4'), '000001001',
'000002001', @email, 1);
insert into student(student_name, student_graduate_year, student_major,
student_email, dorm_id) values (concat(@student_name, 'A5'), '000001001',
'000002001', @email, 1);
insert into student(student_name, student_graduate_year, student_major,
student_email, dorm_id) values (concat(@student_name, 'B1'), '000001001',
'000002002', @email, 2);
insert into student(student_name, student_graduate_year, student_major,
student_email, dorm_id) values (concat(@student_name, 'B2'), '000001001',
'000002002', @email, 2);
insert into student(student_name, student_graduate_year, student_major,
student_email, dorm_id) values (concat(@student_name, 'B3'), '000001001',
'000002002', @email, 2);
```

```
insert into student(student_name, student_graduate_year, student_major,
student_email, dorm_id) values (concat(@student_name, 'B4'), '000001001',
'000002002', @email, 2);
insert into student(student_name, student_graduate_year, student_major,
student_email, dorm_id) values (concat(@student_name, 'B5'), '000001001',
'000002002', @email, 2);
insert into student(student_name, student_graduate_year, student_major,
student_email, dorm_id) values (concat(@student_name, 'C1'), '000001001',
'000002003', @email, 3);
insert into student(student_name, student_graduate_year, student_major,
student_email, dorm_id) values (concat(@student_name, 'C2'), '000001001',
'000002003', @email, 3);
insert into student(student_name, student_graduate_year, student_major,
student_email, dorm_id) values (concat(@student_name, 'C3'), '000001001',
'000002003', @email, 3);
insert into student(student_name, student_graduate_year, student_major,
student_email, dorm_id) values (concat(@student_name, 'C4'), '000001001',
'000002003', @email, 3);
insert into student(student_name, student_graduate_year, student_major,
student_email, dorm_id) values (concat(@student_name, 'C5'), '000001001',
'000002003', @email, 3);
insert into student(student_name, student_graduate_year, student_major,
student_email, dorm_id) values (concat(@student_name, 'D1'), '000001002',
'000002001', @email, 4);
insert into student(student_name, student_graduate_year, student_major,
student_email, dorm_id) values (concat(@student_name, 'D2'), '000001002',
'000002001', @email, 4);
insert into student(student_name, student_graduate_year, student_major,
student_email, dorm_id) values (concat(@student_name, 'D3'), '000001002',
'000002001', @email, 4);
insert into student(student_name, student_graduate_year, student_major,
student_email, dorm_id) values (concat(@student_name, 'D4'), '000001002',
'000002001', @email, 4);
insert into student(student_name, student_graduate_year, student_major,
student_email, dorm_id) values (concat(@student_name, 'D5'), '000001002',
'000002001', @email, 4);
insert into student(student_name, student_graduate_year, student_major,
student_email, dorm_id) values (concat(@student_name, 'E1'), '000001002',
'000002002', @email, 5);
insert into student(student_name, student_graduate_year, student_major,
student_email, dorm_id) values (concat(@student_name, 'E2'), '000001002',
'000002002', @email, 5);
insert into student(student_name, student_graduate_year, student_major,
student_email, dorm_id) values (concat(@student_name, 'E3'), '000001002',
'000002002', @email, 5);
insert into student(student_name, student_graduate_year, student_major,
student_email, dorm_id) values (concat(@student_name, 'E4'), '000001002',
'000002002', @email, 5);
insert into student(student_name, student_graduate_year, student_major,
student_email, dorm_id) values (concat(@student_name, 'E5'), '000001002',
'000002002', @email, 5);
insert into student(student_name, student_graduate_year, student_major,
student_email, dorm_id) values (concat(@student_name, 'F1'), '000001002',
'000002003', @email, 6);
insert into student(student_name, student_graduate_year, student_major,
student_email, dorm_id) values (concat(@student_name, 'F2'), '000001002',
'000002003', @email, 6);
```

```
insert into student(student_name, student_graduate_year, student_major,
student_email, dorm_id) values (concat(@student_name, 'F3'), '000001002',
'000002003', @email, 6);
insert into student(student_name, student_graduate_year, student_major,
student_email, dorm_id) values (concat(@student_name, 'F4'), '000001002',
'000002003', @email, 6);
insert into student(student_name, student_graduate_year, student_major,
student_email, dorm_id) values (concat(@student_name, 'F5'), '000001002',
'000002003', @email, 6);
insert into student(student_name, student_graduate_year, student_major,
student_email, dorm_id) values (concat(@student_name, 'G1'), '000001003',
'000002001', @email, 7);
insert into student(student_name, student_graduate_year, student_major,
student_email, dorm_id) values (concat(@student_name, 'G2'), '000001003',
'000002001', @email, 7);
insert into student(student_name, student_graduate_year, student_major,
student_email, dorm_id) values (concat(@student_name, 'G3'), '000001003',
'000002001', @email, 7);
insert into student(student_name, student_graduate_year, student_major,
student_email, dorm_id) values (concat(@student_name, 'G4'), '000001003',
'000002001', @email, 7);
insert into student(student_name, student_graduate_year, student_major,
student_email, dorm_id) values (concat(@student_name, 'G5'), '000001003',
'000002001', @email, 7);
insert into student(student_name, student_graduate_year, student_major,
student_email, dorm_id) values (concat(@student_name, 'H1'), '000001003',
'000002002', @email, 8);
insert into student(student_name, student_graduate_year, student_major,
student_email, dorm_id) values (concat(@student_name, 'H2'), '000001003',
'000002002', @email, 8);
insert into student(student_name, student_graduate_year, student_major,
student_email, dorm_id) values (concat(@student_name, 'H3'), '000001003',
'000002002', @email, 8);
insert into student(student_name, student_graduate_year, student_major,
student_email, dorm_id) values (concat(@student_name, 'H4'), '000001003',
'000002002', @email, 8);
insert into student(student_name, student_graduate_year, student_major,
student_email, dorm_id) values (concat(@student_name, 'H5'), '000001003',
'000002002', @email, 8);
insert into student(student_name, student_graduate_year, student_major,
student_email, dorm_id) values (concat(@student_name, 'I1'), '000001003',
'000002003', @email, 9);
insert into student(student_name, student_graduate_year, student_major,
student_email, dorm_id) values (concat(@student_name, 'I2'), '000001003',
'000002003', @email, 9);
insert into student(student_name, student_graduate_year, student_major,
student_email, dorm_id) values (concat(@student_name, 'I3'), '000001003',
'000002003', @email, 9);
insert into student(student_name, student_graduate_year, student_major,
student_email, dorm_id) values (concat(@student_name, 'I4'), '000001003',
'000002003', @email, 9);
insert into student(student_name, student_graduate_year, student_major,
student_email, dorm_id) values (concat(@student_name, 'I5'), '000001003',
'000002003', @email, 9);
insert into student(student_name, student_graduate_year, student_major,
student_email, dorm_id) values (concat(@student_name, 'J1'), '000001004',
'000002001', @email, 10);
```

```
insert into student(student_name, student_graduate_year, student_major,
student_email, dorm_id) values (concat(@student_name, 'J2'), '000001004',
'000002001', @email, 10);
insert into student(student_name, student_graduate_year, student_major,
student_email, dorm_id) values (concat(@student_name, 'J3'), '000001004',
'000002001', @email, 10);
insert into student(student_name, student_graduate_year, student_major,
student_email, dorm_id) values (concat(@student_name, 'J4'), '000001004',
'000002001', @email, 10);
insert into student(student_name, student_graduate_year, student_major,
student_email, dorm_id) values (concat(@student_name, 'J5'), '000001004',
'000002001', @email, 10);
insert into student(student_name, student_graduate_year, student_major,
student_email, dorm_id) values (concat(@student_name, 'K1'), '000001004',
'000002002', @email, 11);
insert into student(student_name, student_graduate_year, student_major,
student_email, dorm_id) values (concat(@student_name, 'K2'), '000001004',
'000002002', @email, 11);
insert into student(student_name, student_graduate_year, student_major,
student_email, dorm_id) values (concat(@student_name, 'K3'), '000001004',
'000002002', @email, 11);
insert into student(student_name, student_graduate_year, student_major,
student_email, dorm_id) values (concat(@student_name, 'K4'), '000001004',
'000002002', @email, 11);
insert into student(student_name, student_graduate_year, student_major,
student_email, dorm_id) values (concat(@student_name, 'K5'), '000001004',
'000002002', @email, 11);
insert into student(student_name, student_graduate_year, student_major,
student_email, dorm_id) values (concat(@student_name, 'L1'), '000001004',
'000002003', @email, 12);
insert into student(student_name, student_graduate_year, student_major,
student_email, dorm_id) values (concat(@student_name, 'L2'), '000001004',
'000002003', @email, 12);
insert into student(student_name, student_graduate_year, student_major,
student_email, dorm_id) values (concat(@student_name, 'L3'), '000001004',
'000002003', @email, 12);
insert into student(student_name, student_graduate_year, student_major,
student_email, dorm_id) values (concat(@student_name, 'L4'), '000001004',
'000002003', @email, 12);
insert into student(student_name, student_graduate_year, student_major,
student_email, dorm_id) values (concat(@student_name, 'L5'), '000001004',
'000002003', @email, 12);
```

前后端接口

要实现功能，需要先明确前后端约定好的接口。

需要说明的是，接口的定义一般是前后端约定好的，所以也和前端代码息息相关，前端需要什么数据，需要什么格式的数据，也会在接口中体现。

接口主要体现在

- 请求需要的信息：请求方法，请求路径，请求数据
- 响应数据

用户登录

请求

```
POST api/user/login
Content-Type: application/json

{username: "bit", password: "123"}
```

响应

```
{
  "success" : true
}
```

用户注册

请求

```
POST api/user/register
Content-Type: multipart/form-data; boundary=----
WebKitFormBoundarypOUwkGIMUyLOaOZT

username: haha
password: 111
nickname: 牛牛牛
email: 666@163.com
age: 66
headFile: (binary)
```

注意：以上请求数据是解析过的，http原生发送的数据还包含其他很多内容，比较多，可以动手抓包看看。其中boundary后边的是随机生成的，请求数据中会使用该信息。

响应

```
{
  "success" : true
}
```

查询抽奖设置

请求

```
GET api/setting/query
```

响应

```
{
  "success" : true,
  "data" : {
    "id" : 1,
    "userId" : 1,
    "batchNumber" : 8,
```

```
    "createTime" : "2020-08-14 08:16:31",
    "user" : {
      "id" : 1,
      "username" : "bit",
      "password" : "123",
      "nickname" : "小比特",
      "email" : "1111@163.com",
      "age" : 18,
      "head" : "img/test-head.jpg",
      "createTime" : "2020-08-14 08:16:31",
      "settingId" : 1
    },
    "awards" : [ {
      "id" : 1,
      "name" : "特靠谱欢乐奖",
      "count" : 1,
      "award" : "深圳湾一号",
      "settingId" : 1,
      "createTime" : "2020-08-14 08:16:31",
      "luckyMemberIds" : [ 5 ]
    }, {
      "id" : 2,
      "name" : "特靠谱娱乐奖",
      "count" : 5,
      "award" : "BMW X5",
      "settingId" : 1,
      "createTime" : "2020-08-14 08:16:31",
      "luckyMemberIds" : [ 56, 40, 32, 65, 81 ]
    }, {
      "id" : 3,
      "name" : "特靠谱励志奖",
      "count" : 20,
      "award" : "办公室一日游",
      "settingId" : 1,
      "createTime" : "2020-08-14 08:16:31",
      "luckyMemberIds" : [ 48, 68, 43, 73, 13, 83, 63, 25 ]
    } ],
    "members" : [ {
      "id" : 1,
      "name" : "李寻欢",
      "no" : "水果刀",
      "userId" : 1,
      "createTime" : "2020-08-14 08:16:31"
    }, {
      "id" : 2,
      "name" : "郭靖",
      "no" : "降猪十八掌",
      "userId" : 1,
      "createTime" : "2020-08-14 08:16:31"
    }, {
      "id" : 3,
      "name" : "韦小宝",
      "no" : "抓?龙爪手",
      "userId" : 1,
      "createTime" : "2020-08-14 08:16:31"
    } ]
  }
}
```

修改抽奖人数

请求

```
GET api/setting/update?batchNumber=5
```

接口对应抽奖设置页面中，点每次抽奖人数下拉菜单切换时修改

响应

```
{
  "success" : true
}
```

新增奖项

请求

```
POST api/award/add
Content-Type: application/json

{name: "牛哄哄", count: 3, award: "华为手机"}
```

响应

```
{
  "success" : true
}
```

修改奖项

请求

```
POST api/award/update
Content-Type: application/json

{name: "牛哄哄", count: 3, award: "小米手机", id: 4}
```

响应

```
{
  "success" : true
}
```

删除奖项

请求

```
GET api/award/delete/4
```

最后的数字4，对应奖项的id

响应

```
{
  "success" : true
}
```

新增抽奖人员

请求

```
POST api/member/add
Content-Type: application/json

{name: "羞羞的粉拳", no: "007"}
```

响应

```
{
  "success" : true
}
```

修改抽奖人员

请求

```
POST api/member/update
Content-Type: application/json

{name: "泰山", no: "000", id: 96}
```

响应

```
{
  "success" : true
}
```

删除抽奖人员

请求

```
GET api/member/delete/97
```

注意最后的数字为抽奖人员的id

响应

```
{
  "success" : true
}
```

抽奖

请求

```
POST api/record/add/3
Content-Type: application/json

[92, 22, 43, 76]
```

以上路径中最后的数字代表奖项id，请求数据为抽奖人员id组成的数组

响应

```
{
  "success" : true
}
```

删除当前奖项某个获奖人员

请求

```
GET api/record/delete/member?id=22
```

根据人员id删除对应的获奖记录

响应

```
{
  "success" : true
}
```

删除当前奖项已获奖人员

请求

```
GET api/record/delete/award?id=3
```

根据奖项id删除对应所有获奖人员记录

响应

```
{
  "success" : true
}
```

开发环境准备

参考《开发环境配置》，完成以下配置

1. Maven在IDEA中的配置，主要是全局配置文件和本地仓库路径的配置
2. 数据库面板中配置MySQL数据库连接
3. Lombok插件的安装

配置项目pom.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>

    <parent>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-parent</artifactId>
        <version>2.2.4.RELEASE</version>
    </parent>

    <groupId>frank</groupId>
    <artifactId>lucky-draw</artifactId>
    <version>1.0-SNAPSHOT</version>

    <dependencies>

        <!-- spring-boot-starter-web: 基于SpringBoot开发的依赖包，
                                    会再次依赖spring-framework中基本依赖包，aop相关依赖
包，web相关依赖包，
                                    还会引入其他如json，tomcat，validation等依赖 -->
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-web</artifactId>
        </dependency>

        <!-- mybatis-spring-boot-starter: Mybatis框架在SpringBoot中集成的依赖包，
                                Mybatis是一种数据库对象关系映射Object-Relationl
Mapping（ORM）框架，
                                其他还有如Hibernate等 -->
        <dependency>
            <groupId>org.mybatis.spring.boot</groupId>
            <artifactId>mybatis-spring-boot-starter</artifactId>
            <version>2.1.1</version>
        </dependency>

        <!-- Mybatis代码生成工具 -->
        <dependency>
            <groupId>org.mybatis.generator</groupId>
            <artifactId>mybatis-generator-core</artifactId>
            <version>1.3.5</version>
        </dependency>

        <!-- mysql-connector-java: mysql数据库驱动包
                                在编译时没有直接使用，但是运行时需要，所以使用
                                scope runtime -->
        <dependency>
```

```
            <groupId>mysql</groupId>
            <artifactId>mysql-connector-java</artifactId>
            <version>5.1.47</version>
            <scope>runtime</scope>
        </dependency>

        <!-- druid-spring-boot-starter: 阿里Druid数据库连接池，同样的运行时需要 -->
        <dependency>
            <groupId>com.alibaba</groupId>
            <artifactId>druid-spring-boot-starter</artifactId>
            <version>1.1.21</version>
        </dependency>

        <!-- spring-boot-devtools: SpringBoot的热部署依赖包 -->
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-devtools</artifactId>
            <scope>runtime</scope>
            <!-- 不能被其它模块继承，如果多个子模块可以去掉 -->
            <optional>true</optional>
        </dependency>

        <!-- lombok: 简化bean代码的框架 -->
        <dependency>
            <groupId>org.projectlombok</groupId>
            <artifactId>lombok</artifactId>
            <optional>true</optional>
        </dependency>

        <!-- spring-boot-starter-test: SpringBoot测试框架 -->
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-test</artifactId>
            <scope>test</scope>
            <exclusions>
                <exclusion>
                    <groupId>org.junit.vintage</groupId>
                    <artifactId>junit-vintage-engine</artifactId>
                </exclusion>
            </exclusions>
        </dependency>

    </dependencies>

    <build>
        <finalName>lucky-draw</finalName>
        <plugins>
            <!-- SpringBoot的maven打包插件 -->
            <plugin>
                <groupId>org.springframework.boot</groupId>
                <artifactId>spring-boot-maven-plugin</artifactId>
            </plugin>
        </plugins>
    </build>
</project>
```

准备Springboot配置文件

```
debug=true
# 设置打印日志的级别，及打印sql语句
logging.level.root=INFO
logging.level.druid.sql.Statement=ERROR
logging.level.frank=DEBUG

# 美化JSON数据格式
spring.jackson.serialization.indent-output=true
# 设置JSON数据的日期格式
spring.jackson.date-format=yyyy-MM-dd HH:mm:ss
spring.jackson.time-zone=GMT+8
# JSON数据属性为null时不返回
spring.jackson.default-property-inclusion=non_null

# 找不到资源404时抛出异常
spring.mvc.throw-exception-if-no-handler-found=true
# 禁用静态资源的自动映射，如不禁用，不存在的url将被映射到/**，servlet不有机会抛出异常
#spring.resources.add-mappings=false
# get请求参数及表单提交数据的日期格式
spring.mvc.date-format=yyyy-MM-dd HH:mm:ss
# 应用/项目的部署路径，默认为/
server.servlet.context-path=/lucky-draw
#server.port=8081
# SpringMVC中，DispatcherServlet的映射路径，默认为/**
#spring.mvc.servlet.path=/**
# 自定义属性：用户头像本地保存根路径
user.head.local-path=D:/TMP
user.head.remote-path=http://localhost:8080${server.servlet.context-path}
user.head.filename=head.jpg
# 静态资源映射：将路径映射为/，即/static/xxx，映射为/xxx，支持多个字符串，逗号间隔
# 默认为/META-INF/resources/, /resources/, /static/, /public/
spring.resources.static-
locations=classpath:/static/,classpath:/public/,${user.head.local-path}

#druid数据库连接池配置
spring.datasource.driver-class-name=com.mysql.jdbc.Driver
spring.datasource.url=jdbc:mysql://localhost:3306/lucky_draw?
useUnicode=true&characterEncoding=utf8&autoReconnect=true&failOverReadOnly=false
spring.datasource.username=root
spring.datasource.password=root
spring.datasource.druid.initial-size=1
spring.datasource.druid.min-idle=1
spring.datasource.druid.max-active=20
spring.datasource.druid.test-on-borrow=true

#Mybatis配置
mybatis.mapper-locations=classpath:mapper/**Mapper.xml
#mybatis.type-aliases-package=frank.mapper
mybatis.configuration.map-underscore-to-camel-case=true
#mybatis.config-location=classpath:mybatis/mybatis-config.xml

#mapper
#mappers 多个接口时逗号隔开
#mapper.mappers=tk.mybatis.mapper.common.Mapper,tk.mybatis.mapper.common.MySqlMa
pper,tk.mybatis.mapper.common.IdsMapper
```

```
#mapper.notEmpty=true
#mapper.identity=MYSQL

#pagehelper
#数据库方言:
oracle,mysql,mariadb,sqlite,hsqldb,postgresql,db2,sqlserver,informix,h2,sqlserver2012,derby
pagehelper.helperDialect=mysql
#默认值为 false，该参数对使用 RowBounds 作为分页参数时有效。 当该参数设置为 true 时，会将 RowBounds 中的 offset 参数当成 pageNum 使用，可以用页码和页面大小两个参数进行分页。
#pagehelper.offset-as-page-num=falses
#默认值为false，该参数对使用 RowBounds 作为分页参数时有效。 当该参数设置为true时，使用 RowBounds 分页会进行 count 查询。
pagehelper.row-bounds-with-count=true
#默认值为 false，当该参数设置为 true 时，如果 pageSize=0 或者 RowBounds.limit = 0 就会查询出全部的结果（相当于没有执行分页查询，但是返回结果仍然是 Page 类型）。
#pagehelper.page-size-zero=false
#分页合理化参数，默认值为false。当该参数设置为 true 时，pageNum<=0 时会查询第一页，pageNum>pages（超过总数时），会查询最后一页。默认false 时，直接根据参数进行查询。
pagehelper.reasonable=true
#为了支持startPage(Object params)方法，增加了该参数来配置参数映射，用于从对象中根据属性名取值， 可以配置 pageNum,pageSize,count,pageSizeZero,reasonable，不配置映射的用默认值， 默认值为
pageNum=pageNum;pageSize=pageSize;count=countSql;reasonable=reasonable;pageSizeZero=pageSizeZero。
pagehelper.params=pageNum=pageNumber;pageSize=pageSize;count=countSql;reasonable=reasonable;
#支持通过 Mapper 接口参数来传递分页参数，默认值false，分页插件会从查询方法的参数值中，自动根据上面 params 配置的字段中取值，查找到合适的值时就会自动分页。 使用方法可以参考测试代码中的 com.github.pagehelper.test.basic 包下的 ArgumentsMapTest 和 ArgumentsObjTest。
#pagehelper.supportMethodsArguments=true
#用于控制默认不带 count 查询的方法中，是否执行 count 查询，默认 true 会执行 count 查询，这是一个全局生效的参数，多数据源时也是统一的行为。
pagehelper.default-count=false
```

准备SpringBoot启动类

```
@SpringBootApplication
@MapperScan(修改为Mybatis要扫描Mapper的包)
public class Application {

    public static void main(String[] args) {
        SpringApplication.run(Application.class, args);
    }
}
```

准备前端资源

准备Mybatis代码生成工具所需资源

代码设计

设计数据库实体类

设计http请求基类

主要针对HTTP请求数据的统一字段设计（本项目没有用到这些字段，可以先保留，为以后扩展设计预留）

```
@Getter
@Setter
public class BaseEntity {
}
```

设计统一响应类

主要为返回数据的统一字段设计

```
import lombok.Getter;
import lombok.Setter;

@Getter
@Setter
public class ResponseResult {

    private boolean success;
    private String code;
    private String message;
    private Object data;

    private ResponseResult(){}

    public static ResponseResult ok(Object data){
        ResponseResult result = new ResponseResult();
        result.setSuccess(true);
        result.setData(data);
        return result;
    }

    public static ResponseResult error(){
        return error("ERR000", "未知错误");
    }

    public static ResponseResult error(String code, String message){
        ResponseResult result = new ResponseResult();
        result.setCode(code);
        result.setMessage(message);
        return result;
    }
}
```

设计自定义异常类型

主要针对不同的场景，需要抛异常来处理时，能定位业务含义

主要分为

1. 客户端请求错误时的异常：需要给定错误码，方便前端提示用户，如用户名存在不允许注册（只简单实现，不考虑具体字段的报错）
2. 业务发生错误时的异常：需要给定错误码，方便后端定位问题，一般如程序上的业务错误都可以抛（BUG）
3. 系统发生错误时的异常：需要给定错误码，方便后端定位问题，程序出错，如数据库连接获取失败都可以抛（一般是系统发生错误，如网络断了，数据库挂了等等）

   自定义异常前端需要显示错误码和错误消息（一般是自己抛的中文异常描述），用户可以根据提示信息判断原因。

非自定义异常，异常信息一般是框架或JDK抛出的英文，是给开发人员描述错误的，无法给用户提示，所以错误信息提示为未知异常。

先定义异常的基类：抛异常时，有时候是自己抛，有时候是捕获到异常，再往外抛，所以提供两个构造方法

```
@Getter
@Setter
public class BaseException extends RuntimeException {

    protected String code;

    public BaseException(String code, String message) {
        this(code, message, null);
    }

    public BaseException(String code, String message, Throwable cause) {
        super(message, cause);
        this.code = code;
    }
}
```

再完成异常的子类：提供不同的错误码前缀，以便于前端提示时，知道是哪的问题

```
public class BusinessException extends BaseException {

    public BusinessException(String code) {
        this(code, null);
    }

    public BusinessException(String code, String message) {
        this(code, message, null);
    }

    public BusinessException(String code, String message, Throwable cause) {
        super("BUS" + code, message, cause);
    }
}
```

```
public class ClientException extends BaseException {
```

```
    public ClientException(String code) {
        this(code, null);
    }

    public ClientException(String code, String message) {
        this(code, message, null);
    }

    public ClientException(String code, String message, Throwable cause) {
        super("CLI" + code, message, cause);
    }
}
```

```
public class SystemException extends BaseException {

    public SystemException(String code) {
        this(code, null);
    }

    public SystemException(String code, String message) {
        this(code, message, null);
    }

    public SystemException(String code, String message, Throwable cause) {
        super("SYS" + code, message, cause);
    }
}
```

设计Controller中抛异常时的拦截器

这里不光是Controller抛异常，只要Controller代码执行，调用其他方法产生的异常，都会退出Controller方法，即HTTP请求方法结束，由拦截器统一处理。

```
@Slf4j
@ControllerAdvice
public class ExceptionAdvisor {

    /**
     * 请求数据错误：包括类型转换错误，校验失败
     * @param e
     */
    @ExceptionHandler({
            BindException.class//使用@Valid 验证路径中请求实体校验失败后抛出的异常
            , ConstraintViolationException.class//处理请求参数格式错误 @RequestParam上validate失败后抛出的异常
            , MethodArgumentNotValidException.class//处理请求参数格式错误@RequestBody上validate失败后抛出的异常
            , MethodArgumentTypeMismatchException.class//请求参数类型转换错误
    })
    @ResponseStatus(HttpStatus.BAD_REQUEST)
    public void handleMethodArgumentTypeMismatchException(Throwable e){
        log.debug("================================");
        log.debug("Controller方法参数类型转换错误", e);
    }
```

```
    @ExceptionHandler({
            MethodNotAllowedException.class
            , HttpRequestMethodNotSupportedException.class
    })
    @ResponseStatus(HttpStatus.METHOD_NOT_ALLOWED)
    public void handleMethodNotAllowedException(Throwable e){
        log.debug("================================");
        log.debug("Controller提供的http方法不支持", e);
    }

    @ExceptionHandler(NoHandlerFoundException.class)
    @ResponseStatus(HttpStatus.NOT_FOUND)
    public void handleNoHandlerFoundException(Throwable e){
        log.debug("================================");
        log.debug("找不到http请求处理器", e);
    }

    @ExceptionHandler(BaseException.class)
    @ResponseStatus(HttpStatus.OK)
    @ResponseBody
    public Object handleBaseException(BaseException e){
        log.debug("================================");
        log.debug("自定义异常", e);
        return ResponseResult.error(e.getCode(), e.getMessage(), e);
    }

    @ExceptionHandler(Throwable.class)
    @ResponseStatus(HttpStatus.OK)
    @ResponseBody
    public Object handleException(Throwable e){
        log.error("================================");
        log.error("未知异常：“+e.getClass().getName()+”，请联系管理员", e);
        return ResponseResult.error(e);
    }
}
```

设计会话管理的拦截器及统一数据响应配置

```
import org.springframework.web.servlet.HandlerInterceptor;

import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;

public class LoginInterceptor implements HandlerInterceptor {

    @Override
    public boolean preHandle(HttpServletRequest request, HttpServletResponse
response, Object handler) throws Exception {
        HttpSession session = request.getSession(false);
        //登录校验
        if(session != null && session.getAttribute("user") != null){
            return true;
        }
        response.setStatus(401);
        return false;
    }
```

```
}
```

设计Web统一处理的配置

1. 会话管理的拦截器要引入配置：只拦截后端服务路径，并排除登录、注册、注销接口
2. 后端服务器路径都有/api的前缀，可以加上统一的路径映射
3. 统一的响应数据格式封装：这里不使用`@ControllerAdvice`和`ResponseBodyAdvice`进行拦截，原因是，返回值为null，会出现无法统一包装，响应体为空

```
import frank.config.interceptor.LoginInterceptor;
import frank.config.web.RequestResponseBodyMethodProcessorWrapper;
import org.springframework.beans.factory.InitializingBean;
import org.springframework.context.annotation.Configuration;
import org.springframework.web.method.support.HandlerMethodReturnValueHandler;
import org.springframework.web.servlet.config.annotation.InterceptorRegistry;
import org.springframework.web.servlet.config.annotation.PathMatchConfigurer;
import org.springframework.web.servlet.config.annotation.WebMvcConfigurer;
import
org.springframework.web.servlet.mvc.method.annotation.RequestMappingHandlerAdapt
er;
import
org.springframework.web.servlet.mvc.method.annotation.RequestResponseBodyMethodP
rocessor;

import javax.annotation.Resource;
import java.util.ArrayList;
import java.util.List;

@Configuration
public class SysConfig implements WebMvcConfigurer, InitializingBean{

    @Resource
    private RequestMappingHandlerAdapter adapter;

    @Override
    public void afterPropertiesSet() {
        List<HandlerMethodReturnValueHandler> returnValueHandlers =
adapter.getReturnValueHandlers();
        List<HandlerMethodReturnValueHandler> handlers = new
ArrayList(returnValueHandlers);
        for(int i=0; i<handlers.size(); i++){
            HandlerMethodReturnValueHandler handler = handlers.get(i);
            if(handler instanceof RequestResponseBodyMethodProcessor){
                handlers.set(i, new
RequestResponseBodyMethodProcessorWrapper(handler));
            }
        }
        adapter.setReturnValueHandlers(handlers);
    }

    @Override
    public void addInterceptors(InterceptorRegistry registry) {
        registry.addInterceptor(new LoginInterceptor())
                .addPathPatterns("/api/**")
                .excludePathPatterns("/api/user/login")
                .excludePathPatterns("/api/user/register")
                .excludePathPatterns("/api/user/logout")
```

```
        ;
    }

    @Override
    public void configurePathMatch(PathMatchConfigurer configurer) {
        configurer.addPathPrefix("api", c -> true);
    }
}
```

提供统一响应体封装处理类：

```
import frank.base.ResponseResult;
import org.springframework.core.MethodParameter;
import org.springframework.web.context.request.NativeWebRequest;
import org.springframework.web.method.support.HandlerMethodReturnValueHandler;
import org.springframework.web.method.support.ModelAndViewContainer;

public class RequestResponseBodyMethodProcessorWrapper implements
HandlerMethodReturnValueHandler {

    private final HandlerMethodReturnValueHandler delegate;

    public
RequestResponseBodyMethodProcessorWrapper(HandlerMethodReturnValueHandler
delegate) {
        this.delegate = delegate;
    }

    @Override
    public boolean supportsReturnType(MethodParameter returnType) {
        return delegate.supportsReturnType(returnType);
    }

    @Override
    public void handleReturnValue(Object returnValue, MethodParameter
returnType, ModelAndViewContainer mavContainer, NativeWebRequest webRequest)
throws Exception {
        if(!(returnValue instanceof ResponseResult)){
            returnValue = ResponseResult.ok(returnValue);
        }
        delegate.handleReturnValue(returnValue, returnType, mavContainer,
webRequest);
    }
}
```

设计Mybatis中Mapper的基类

使用Mybatis的接口方法，所有接口方法都是类似，只是传入参数和返回值不同，可以考虑设计统一的基类，以泛型的方式定义出不同的参数类型、返回类型

```
public interface BaseMapper<T extends BaseEntity>{

    T selectByPrimaryKey(Integer id);

    int insert(T record);
```

```
    int insertSelective(T record);

    int updateByPrimaryKeySelective(T record);

    int updateByPrimaryKey(T record);

    int deleteByPrimaryKey(Integer id);

    T selectOne(T record);

    List<T> selectAll();

    List<T> selectByCondition(T record);

    int deleteByIds(List<Integer> ids);
}
```

生成Mybatis资源：实体类、Mapper和XML