1. 课程列表

> *当用户登录成功之后，会进入到课程页面，可以购买课程，进入学习，也可以先试用*

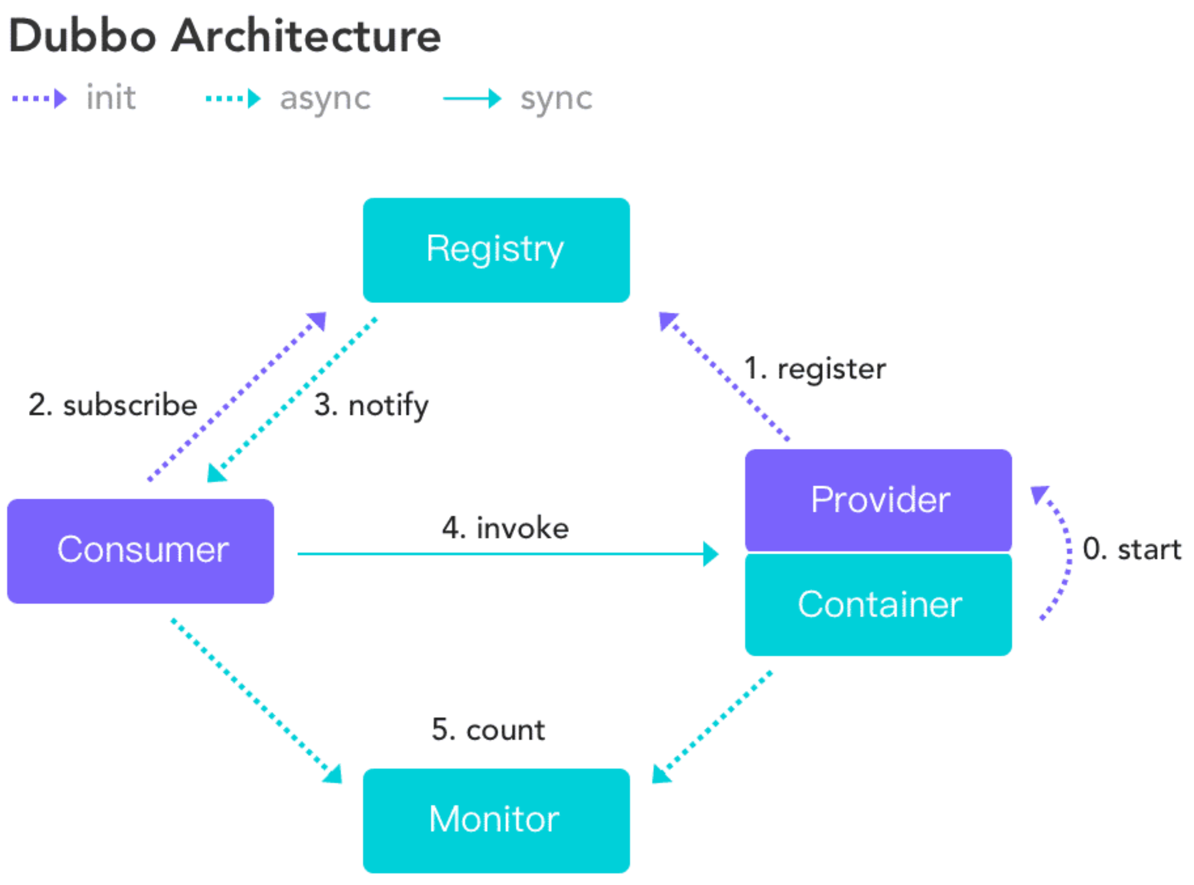
1.1 需求

> *课程列表不需要登录，但是如果用户未登录，那么是无法进行购买和练习的，以及已经购买的课程显示，所以需要先判断登录状态，登录的行为是发生在sso，所以需要去sso进行认证登录，这时候就涉及到了网络访问，可选择的有http或者rpc（远程调用），在本项目中，我们选择dubbo*

dubbo网址：https://dubbo.apache.org/zh/

nacos网址：https://nacos.io/zh-cn/

我们用到的是dubbo+nacos的组合



1.2 sso和web集成dubbo

添加dubbo依赖：

```
<properties>
    <dubbo.version>2.7.8</dubbo.version>
</properties>

<!-- Dubbo Spring Boot Starter -->
            <dependency>
                <groupId>org.apache.dubbo</groupId>
                <artifactId>dubbo-spring-boot-starter</artifactId>
                <version>${dubbo.version}</version>
            </dependency>
            <dependency>
                <groupId>org.apache.dubbo</groupId>
                <artifactId>dubbo</artifactId>
                <version>${dubbo.version}</version>
                <exclusions>
```

```
                    <exclusion>
                        <groupId>org.springframework</groupId>
                        <artifactId>spring</artifactId>
                    </exclusion>
                    <exclusion>
                        <groupId>javax.servlet</groupId>
                        <artifactId>servlet-api</artifactId>
                    </exclusion>
                    <exclusion>
                        <groupId>log4j</groupId>
                        <artifactId>log4j</artifactId>
                    </exclusion>
                </exclusions>
            </dependency>
            <!-- Dubbo Registry Nacos -->
            <dependency>
                <groupId>com.alibaba.nacos</groupId>
                <artifactId>nacos-client</artifactId>
                <version>2.0.3</version>
            </dependency>
```

nacos和dubbo-spring-boot-starter放入sso和web依赖中

1.2.1 sso服务提供方

新建mszlu-xt-sso-dubbo-service：

```
<?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">
    <parent>
        <artifactId>mszlu-xt-sso</artifactId>
        <groupId>com.mszlu</groupId>
        <version>1.0-SNAPSHOT</version>
    </parent>
    <modelVersion>4.0.0</modelVersion>

    <artifactId>mszlu-xt-sso-dubbo-service</artifactId>

    <dependencies>
        <dependency>
```

```xml
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter</artifactId>
        </dependency>
        <!-- Dubbo Spring Boot Starter -->
        <dependency>
            <groupId>org.apache.dubbo</groupId>
            <artifactId>dubbo-spring-boot-starter</artifactId>
        </dependency>

        <dependency>
            <groupId>com.alibaba.nacos</groupId>
            <artifactId>nacos-client</artifactId>
        </dependency>
        <dependency>
            <groupId>com.mszlu.xt</groupId>
            <artifactId>mszlu-xt-sso-service</artifactId>
        </dependency>
        <dependency>
            <groupId>com.mszlu.xt</groupId>
            <artifactId>mszlu-xt-sso-domain</artifactId>
        </dependency>
    </dependencies>
</project>
```

新建mszlu-xt-sso-provider：

```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">
    <parent>
        <artifactId>mszlu-xt-sso</artifactId>
        <groupId>com.mszlu</groupId>
        <version>1.0-SNAPSHOT</version>
    </parent>
    <modelVersion>4.0.0</modelVersion>

    <artifactId>mszlu-xt-sso-provider</artifactId>

    <dependencies>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter</artifactId>
        </dependency>
         <dependency>
            <groupId>com.mszlu.xt</groupId>
            <artifactId>mszlu-xt-sso-dubbo-service</artifactId>
            <version>1.0-SNAPSHOT</version>
        </dependency>
    </dependencies>
</project>
```

启动类:

```java
package com.mszlu.xt.sso.dubbo;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication
public class SSODubboApp {

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

1.2.2 提供token认证服务

```java
package com.mszlu.xt.sso.service;


public interface TokenService {

    Long checkToken(String token);
}
```

```java
package com.mszlu.xt.sso.service.dubbo;

import com.mszlu.xt.sso.domain.TokenDomain;
import com.mszlu.xt.sso.domain.repository.TokenDomainRepository;
import com.mszlu.xt.sso.service.TokenService;
import org.apache.dubbo.config.annotation.DubboService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
//将此服务 注册到 nacos 注册中心上去
//暴露的服务 TokenService.checkToken() 服务的版本号1.0.0
@DubboService(version = "1.0.0",interfaceClass = TokenService.class)
public class TokenServiceImpl extends AbstractService implements
TokenService {


    @Autowired
    private TokenDomainRepository tokenDomainRepository;

    @Override
    public Long checkToken(String token) {
        TokenDomain tokenDomain = tokenDomainRepository.createDomain();
        return tokenDomain.checkToken(token);
    }
}
```

启动，需要在provider模块中，做dubbo相关配置：

```properties
server.port=8338
spring.application.name=xt-sso

nacos.server-address=127.0.0.1
nacos.port=8848
nacos.username=nacos
nacos.password=nacos

dubbo.scan.base-packages=com.mszlu.xt.sso.service
dubbo.registry.group=xt_dubbo
dubbo.registry.address=nacos://${nacos.server-address}:${nacos.port}/?
username=${nacos.username}&password=${nacos.password}
dubbo.registry.protocol=20881

spring.profiles.active=dev
```

application-dev.properties:

```properties
#数据库配置
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
spring.datasource.url=jdbc:mysql://localhost:3306/xt?
useUnicode=true&characterEncoding=utf-8&serverTimezone=UTC
spring.datasource.username=root
spring.datasource.password=root

#redis 配置 本机开发使用的为单机模式
spring.redis.host=localhost
spring.redis.port=6379

# redis集群配置  生产环境使用集群配置
#spring.redis.jedis.pool.max-wait=5000ms
#spring.redis.jedis.pool.max-Idle=100
#spring.redis.jedis.pool.min-Idle=10
#spring.redis.timeout=10s
#spring.redis.cluster.nodes=192.168.111.131:6379,192.168.111.131:6380,19
2.168.111.131:6381
#spring.redis.cluster.max-redirects=5

#读写分离
##shardingsphere配置
spring.shardingsphere.datasource.common.type=com.zaxxer.hikari.HikariDat
aSource
spring.shardingsphere.datasource.common.driver-class-
name=com.mysql.cj.jdbc.Driver
spring.shardingsphere.datasource.common.username=root
spring.shardingsphere.datasource.common.password= root

## 一主2从
spring.shardingsphere.datasource.names=master,slave0,slave1

# 配置第 1 个数据源
spring.shardingsphere.datasource.master.type=com.zaxxer.hikari.HikariDat
aSource
spring.shardingsphere.datasource.master.driver-class-
name=com.mysql.cj.jdbc.Driver
spring.shardingsphere.datasource.master.jdbc-
url=jdbc:mysql://localhost:3306/xt?
useUnicode=true&characterEncoding=UTF-8&serverTimezone=UTC
spring.shardingsphere.datasource.master.username=root
spring.shardingsphere.datasource.master.password=root

# 配置第 2 个数据源
spring.shardingsphere.datasource.slave0.type=com.zaxxer.hikari.HikariDat
aSource
spring.shardingsphere.datasource.slave0.driver-class-
name=com.mysql.cj.jdbc.Driver
```

```
spring.shardingsphere.datasource.slave0.jdbc-
url=jdbc:mysql://localhost:3306/xt?
useUnicode=true&characterEncoding=UTF-8&serverTimezone=UTC
spring.shardingsphere.datasource.slave0.username=root
spring.shardingsphere.datasource.slave0.password=root
# 配置第 3 个数据源
spring.shardingsphere.datasource.slave1.type=com.zaxxer.hikari.HikariDat
aSource
spring.shardingsphere.datasource.slave1.driver-class-
name=com.mysql.cj.jdbc.Driver
spring.shardingsphere.datasource.slave1.jdbc-
url=jdbc:mysql://localhost:3306/xt?
useUnicode=true&characterEncoding=UTF-8&serverTimezone=UTC
spring.shardingsphere.datasource.slave1.username=root
spring.shardingsphere.datasource.slave1.password=root

# 写数据源名称
spring.shardingsphere.rules.readwrite-splitting.data-sources.ms.write-
data-source-name=master
# 读数据源名称，多个从数据源用逗号分隔
spring.shardingsphere.rules.readwrite-splitting.data-sources.ms.read-
data-source-names=slave0,slave1
# 负载均衡算法名称
spring.shardingsphere.rules.readwrite-splitting.data-sources.ms.load-
balancer-name=round-robin

## 负载均衡算法配置
spring.shardingsphere.rules.readwrite-splitting.load-balancers.round-
robin.type=ROUND_ROBIN
## 负载均衡算法属性配置
spring.shardingsphere.rules.readwrite-splitting.load-balancers.round-
robin.props.workId=1
#打印sql
spring.shardingsphere.props.sql-show=true
```

1.2.3 web服务消费方

在web-api模块中导入dubbo相关依赖：

```
 <!-- Dubbo Spring Boot Starter -->
        <dependency>
            <groupId>org.apache.dubbo</groupId>
            <artifactId>dubbo-spring-boot-starter</artifactId>
        </dependency>

        <dependency>
            <groupId>com.alibaba.nacos</groupId>
            <artifactId>nacos-client</artifactId>
        </dependency>
 <dependency>
            <groupId>com.mszlu</groupId>
            <artifactId>mszlu-xt-sso-service</artifactId>
            <version>1.0-SNAPSHOT</version>
        </dependency>
```

将sso中实现的登录拦截器稍作修改，认证通过dubbo调用sso进行

```
package com.mszlu.xt.web.handler;

import com.alibaba.fastjson.JSON;
import com.mszlu.common.model.BusinessCodeEnum;
import com.mszlu.common.model.CallResult;
import com.mszlu.common.utils.UserThreadLocal;
import com.mszlu.xt.sso.service.TokenService;
import lombok.extern.slf4j.Slf4j;
import org.apache.commons.lang3.StringUtils;
import org.apache.dubbo.config.annotation.DubboReference;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.data.redis.core.RedisTemplate;
import org.springframework.stereotype.Component;
import org.springframework.web.servlet.HandlerInterceptor;
import org.springframework.web.servlet.ModelAndView;

import javax.servlet.http.Cookie;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import java.io.IOException;
import java.io.PrintWriter;
import java.time.Duration;

@Component
@Slf4j
public class LoginInterceptor implements HandlerInterceptor {

    @DubboReference(version = "1.0.0")
    private TokenService tokenService;
    @Override
    public boolean preHandle(HttpServletRequest request,
HttpServletResponse response, Object handler)
            throws Exception {

        log.info("==========================start=======================
=========");
        log.info("request uri:{}",request.getRequestURI());
        log.info("request method:{}",request.getMethod());
        log.info("==========================end=========================
=======");
        Cookie[] cookies = request.getCookies();
        if (cookies == null){
            returnJson(response);
            return false;
        }
        String token = null;
        for (Cookie cookie : cookies) {
            String name = cookie.getName();
            if ("t_token".equals(name)){
                token = cookie.getValue();
            }
        }
        if (token == null){
            returnJson(response);
            return false;
        }
        Long userId = tokenService.checkToken(token);
        if (userId != null){
            UserThreadLocal.put(userId);
            return true;
        }
        returnJson(response);
        return false;
    }
    @Override
    public void postHandle(HttpServletRequest request,
HttpServletResponse response, Object handler,
                           ModelAndView modelAndView) throws Exception {
    }

    @Override
    public void afterCompletion(HttpServletRequest request,
HttpServletResponse response, Object handler, Exception ex)
            throws Exception {
```

```java
            UserThreadLocal.remove();
        }
    private void returnJson(HttpServletResponse response){
        PrintWriter writer = null;
        response.setCharacterEncoding("UTF-8");
        response.setContentType("application/json; charset=utf-8");
        try {
            writer = response.getWriter();
            CallResult callResult =
CallResult.fail(BusinessCodeEnum.NO_LOGIN.getCode(),"您的登录已失效，请重新
登录");
            writer.print(JSON.toJSONString(callResult));
        } catch (IOException e){
            e.printStackTrace();
        } finally {
            if(writer != null){
                writer.close();
            }
        }
    }
}
```

```java
package com.mszlu.xt.web.config;

import com.mszlu.xt.web.handler.LoginInterceptor;
import lombok.extern.slf4j.Slf4j;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.context.annotation.Configuration;
import org.springframework.web.servlet.config.annotation.CorsRegistry;
import
org.springframework.web.servlet.config.annotation.InterceptorRegistry;
import
org.springframework.web.servlet.config.annotation.WebMvcConfigurer;

@Slf4j
@Configuration
public class InterceptorConfig implements WebMvcConfigurer {
    @Autowired
    private LoginInterceptor loginInterceptor;

    @Override
    public void addCorsMappings(CorsRegistry registry) {
```

```
 registry.addMapping("/**").allowedOrigins("http://www.lzxtedu.com");
    }

    @Override
    public void addInterceptors(InterceptorRegistry registry) {
        registry.addInterceptor(loginInterceptor)
                .addPathPatterns("/topic/*")
                .addPathPatterns("/subject/*")
                .addPathPatterns("/course/*")
                .addPathPatterns("/order/*")
                .addPathPatterns("/user/*")
                .addPathPatterns("/i/*")
                .excludePathPatterns("/course/courseList")
                .excludePathPatterns("/subject/listSubjectNew")
                .excludePathPatterns("/course/subjectInfo")
                .excludePathPatterns("/order/notify")
                .excludePathPatterns("/case/*")
                .excludePathPatterns("/wechat/*")
                .excludePathPatterns("/login/wxLoginCallBack")
                .excludePathPatterns("/i/u/*");
//                  .excludePathPatterns("/course/courseList");;
        System.out.println("拦截器");
    }

//    /**
//     * 跨域过滤器
//     *
//     * @return
//     */
//    @Bean
//    public CorsFilter corsFilter() {
//        UrlBasedCorsConfigurationSource source = new
UrlBasedCorsConfigurationSource();
//        CorsConfiguration corsConfiguration = new CorsConfiguration();
//        this.addAllowedOrigins(corsConfiguration);
//        corsConfiguration.addAllowedHeader("*");
//        corsConfiguration.addAllowedMethod("*");
//        corsConfiguration.addAllowedOrigin("*");
//        source.registerCorsConfiguration("/**", corsConfiguration);
//        return new CorsFilter(source);
//    }
//
//    private void addAllowedOrigins(CorsConfiguration
corsConfiguration) {
```

```
////        for (String origin : originsVal) {
//            corsConfiguration.addAllowedOrigin("*");
////        }
//    }
}
```

dubbo消费方配置:

```
nacos.server-address=127.0.0.1
nacos.port=8848
nacos.username=nacos
nacos.password=nacos

dubbo.scan.base-packages=com.mszlu.xt.web.
dubbo.registry.group=xt_dubbo
dubbo.registry.address=nacos://${nacos.server-address}:${nacos.port}/?
username=${nacos.username}&password=${nacos.password}
dubbo.consumer.check=false
```

写个接口 测试一下:

```
package com.mszlu.xt.web.api;

import com.mszlu.common.model.CallResult;
import com.mszlu.common.utils.UserThreadLocal;
import org.springframework.web.bind.annotation.*;


@RestController
@RequestMapping("course")
public class CourseApi {

    @GetMapping(value = "login")
    public CallResult courseList(){

        return CallResult.success(UserThreadLocal.get());
    }
}
```

浏览器输入http://www.lzxtedu.com/api/course/login测试:

1.3 开发课程列表

> *课程列表不需要登录，但是如果是登录状态，需要展示购买课程的状态*

1.3.1 业务逻辑分析

1. 课程列表可以根据条件进行筛选
2. 有条件，按照条件进行查询，无，则查询全部课程
3. 课程购买数量需要展示
4. 需要显示课程包含科目的详细信息
5. 登录状态，登录查询用户购买信息，展示课程购买后UI，未登录 正常显示

1.3.2 获取科目信息

> *在页面上，点击进入学习，会调用一个接口，请求所有的科目信息*

接口：/api/subject/listSubjectNew

参数：无

请求方式：POST

返回值：

```
{
    "code":2000000000,
    "message":"default success",
    "result":{
        "subjectGradeList":[
            "七年级",
            "八年级",
            "九年级"
        ],
        "subjectNameList":[
            "1234",
            "213",
            "222",
            "dd",
            "ed",
            "fff",
            "sd",
            "test",
            "初一政治",
            "初一语文",
            "初中英语",
            "历史",
            "测试多种题型",
            "道法"
        ],
        "subjectTermList":[
            "上",
            "下"
        ]
    },
    "success":true
}
```

1.3.2.1 代码

Controller

```
package com.mszlu.xt.web.api;

import com.mszlu.common.model.CallResult;
import com.mszlu.xt.model.service.SubjectService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;

@RestController
@RequestMapping("subject")
public class SubjectApi {

    @Autowired
    private SubjectService subjectService;

    @PostMapping(value = "listSubjectNew")
    public CallResult listSubjectNew(){
        return subjectService.listSubject();
    }
}
```

Service:

```java
package com.mszlu.xt.model.service;

import com.mszlu.common.model.CallResult;

public interface SubjectService {

    CallResult listSubject();
}
```

```java
package com.mszlu.xt.web.service.impl;

import com.mszlu.common.model.CallResult;
import com.mszlu.common.service.AbstractTemplateAction;
import com.mszlu.xt.model.service.SubjectService;
import com.mszlu.xt.web.domain.SubjectDomain;
import com.mszlu.xt.web.domain.repository.SubjectDomainRepository;
import com.mszlu.xt.web.model.params.SubjectParam;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

@Service
public class SubjectServiceImpl extends AbstractService implements
SubjectService {

    @Autowired
    private SubjectDomainRepository subjectDomainRepository;

    @Override
    public CallResult listSubject() {
        SubjectDomain subjectDomain =
subjectDomainRepository.createDomain(new SubjectParam());
        return this.serviceTemplate.executeQuery(new
AbstractTemplateAction<Object>() {
            @Override
            public CallResult<Object> doAction() {
                return subjectDomain.listSubject();
            }
        });
    }
}
```

```java
package com.mszlu.xt.web.model.params;

import lombok.Data;

@Data
public class SubjectParam {
}
```

Domain:

```java
package com.mszlu.xt.web.domain.repository;

import
com.baomidou.mybatisplus.core.conditions.query.LambdaQueryWrapper;
import com.mszlu.xt.pojo.Subject;
import com.mszlu.xt.web.dao.SubjectMapper;
import com.mszlu.xt.web.domain.SubjectDomain;
import com.mszlu.xt.web.model.enums.Status;
import com.mszlu.xt.web.model.params.SubjectParam;
import org.springframework.stereotype.Component;

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

@Component
public class SubjectDomainRepository {

    @Resource
    private SubjectMapper subjectMapper;

    public SubjectDomain createDomain(SubjectParam subjectParam) {
        return new SubjectDomain(subjectParam,this);
    }

    public List<Subject> findSubjectList() {
        LambdaQueryWrapper<Subject> queryWrapper = new
LambdaQueryWrapper<>();
        queryWrapper.eq(Subject::getStatus, Status.NORMAL.getCode());
        List<Subject> subjects = subjectMapper.selectList(queryWrapper);
        return subjects;
    }
}
```

```
package com.mszlu.xt.web.domain;

import com.mszlu.common.model.CallResult;
import com.mszlu.common.utils.CommonUtils;
import com.mszlu.xt.pojo.Subject;
import com.mszlu.xt.web.domain.repository.SubjectDomainRepository;
import com.mszlu.xt.web.model.SubjectModel;
import com.mszlu.xt.web.model.params.SubjectParam;
import org.springframework.beans.BeanUtils;
import org.springframework.cglib.beans.BeanCopier;

import java.util.*;

/**
 * @author Jarno
 */
public class SubjectDomain {
    private SubjectParam subjectParam;
    private SubjectDomainRepository subjectDomainRepository;


    public SubjectDomain(SubjectParam subjectParam,
SubjectDomainRepository subjectDomainRepository){
        this.subjectParam = subjectParam;
        this.subjectDomainRepository = subjectDomainRepository;
    }

    public CallResult<Object> listSubject() {
        List<Subject> subjectList =
this.subjectDomainRepository.findSubjectList();
        List<SubjectModel> subjectModelList = copyList(subjectList);
        Set<String> subjectNameList = new TreeSet<>();
        Set<String> subjectGradeList = new TreeSet<>();
        Set<String> subjectTermList = new TreeSet<>();
        for (SubjectModel subjectModel : subjectModelList) {
            subjectNameList.add(subjectModel.getSubjectName());
            subjectGradeList.add(subjectModel.getSubjectGrade());
            subjectTermList.add(subjectModel.getSubjectTerm());
        }
        Set<String> sortSet = new TreeSet<String>((o1, o2) -> {
            String numberStr1 = CommonUtils.getNumberStr(o1);
```

```
                String numberStr2 = CommonUtils.getNumberStr(o2);
                long num1 = CommonUtils.chineseNumber2Int(numberStr1);
                long num2 = CommonUtils.chineseNumber2Int(numberStr2);
                return (int) (num1 - num2);
            });
            sortSet.addAll(subjectGradeList);
            Map<String,Set<String>> map = new HashMap<>();
            map.put("subjectNameList",subjectNameList);
            map.put("subjectGradeList",sortSet);
            map.put("subjectTermList",subjectTermList);
            return CallResult.success(map);
        }

        private List<SubjectModel> copyList(List<Subject> subjectList) {
            List<SubjectModel> subjectModels = new ArrayList<>();
            for (Subject subject : subjectList) {
                SubjectModel target = new SubjectModel();
                BeanUtils.copyProperties(subject, target);
                subjectModels.add(target);
            }

            return subjectModels;
        }
    }
```

Mapper:

```
package com.mszlu.xt.web.dao;

import com.baomidou.mybatisplus.core.mapper.BaseMapper;
import com.mszlu.xt.pojo.Subject;

public interface SubjectMapper extends BaseMapper<Subject> {

}
```

1.3.3 课程列表

接口：/api/course/courseList

参数：

```java
package com.xiaopizhu.tiku.model.param;

import lombok.Data;

import java.math.BigDecimal;
import java.util.List;

/**
 * @author Jarno
 */
@Data
public class CourseParam {
    private Long userId;
    private Long id;
    private String courseName;
    private String courseDesc;
    private String subjects;
    private BigDecimal coursePrice;
    private BigDecimal courseZhePrice;
    private Integer courseStatus;
    private List<Integer> subjectIdList;
    private Integer orderTime;//订购时长
    private int page = 1;
    private int pageSize = 20;
    private String subjectName;
    private String subjectGrade;
    private String subjectTerm;

    private Long courseId;
    private String imageUrl;
}
```

请求方式：POST

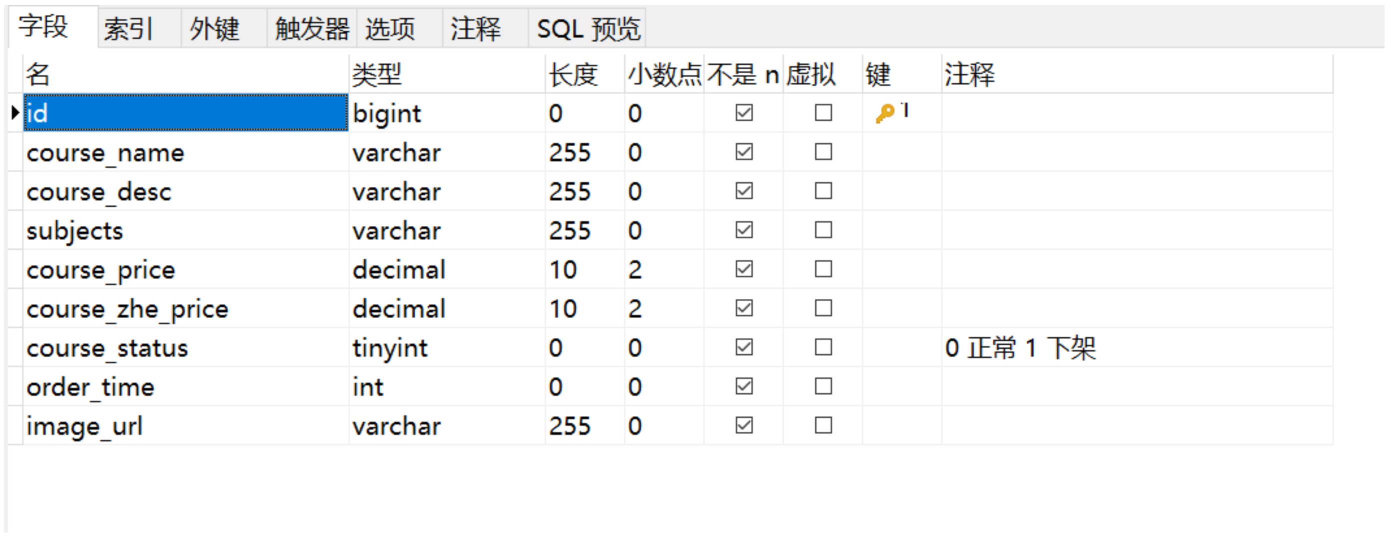
返回值：

```json
{
    "code":2000000000,
    "message":"default success",
    "result":{
        "pageCount":1,
        "page":1,
        "pageSize":0,
        "size":6,
        "list":[
            {
                "id":3,
                "courseName":"全课程",
                "courseDesc":"全课程",
                "coursePrice":199,
                "courseZhePrice":108,
                "orderTime":365,
                "studyCount":0,
                "subjectIdList":null,
                "subjectList":null,
                "subjectInfo":{
                    "id":null,
                    "subjectName":"",
                    "subjectGrade":"",
                    "subjectTerm":"",
                    "subjectUnitList":null
                },
                "buy":0,
                "expireTime":null,

 "imageUrl":"https://gimg2.baidu.com/image_search/src=http%3A%2F%2Fpic19
.nipic.com%2F20120225%2F9165552_152408494000_2.jpg&refer=http%3A%2F%2Fpi
c19.nipic.com&app=2002&size=f9999,10000&q=a80&n=0&g=0n&fmt=jpeg?
sec=1634921537&t=6219c2c069d33cc35ac8f31ef411dc24",
                "userName":null
            },
            {
                "id":4,
                "courseName":"七年级历史上",
                "courseDesc":"七年级历史上",
                "coursePrice":199,
                "courseZhePrice":99,
                "orderTime":365,
                "studyCount":0,
                "subjectIdList":null,
                "subjectList":null,
                "subjectInfo":{
                    "id":null,
                    "subjectName":"",
                    "subjectGrade":"",
                    "subjectTerm":"",
                    "subjectUnitList":null
                },
                "buy":0,
            "expireTime":null,
            "imageUrl":"",
            "userName":null
        },
        {
            "id":5,
            "courseName":"七年级历史下",
            "courseDesc":"七年级历史下",
            "coursePrice":199,
            "courseZhePrice":99,
            "orderTime":365,
            "studyCount":1,
            "subjectIdList":null,
            "subjectList":null,
            "subjectInfo":{
                "id":null,
                "subjectName":"",
                "subjectGrade":"",
                "subjectTerm":"",
                "subjectUnitList":null
            },
            "buy":0,
            "expireTime":null,
            "imageUrl":"",
            "userName":null
        },
        {
            "id":6,
            "courseName":"七年级道法上",
            "courseDesc":"七年级道法上",
            "coursePrice":199,
            "courseZhePrice":99,
            "orderTime":365,
            "studyCount":0,
            "subjectIdList":null,
            "subjectList":null,
            "subjectInfo":{
                "id":null,
                "subjectName":"",
                "subjectGrade":"",
                "subjectTerm":"",
                "subjectUnitList":null
            },
            "buy":0,
            "expireTime":null,
            "imageUrl":"",
            "userName":null
        },
        {
            "id":7,
            "courseName":"七年级道法下",
            "courseDesc":"七年级道法下",
            "coursePrice":1999,
            "courseZhePrice":99,
            "orderTime":365,
            "studyCount":0,
            "subjectIdList":null,
            "subjectList":null,
            "subjectInfo":{
                "id":null,
                "subjectName":"",
                "subjectGrade":"",
                "subjectTerm":"",
                "subjectUnitList":null
            },
            "buy":0,
            "expireTime":null,
            "imageUrl":"",
            "userName":null
        },
        {
            "id":9,
            "courseName":"12",
            "courseDesc":"3123",
            "coursePrice":123,
            "courseZhePrice":123,
            "orderTime":123,
            "studyCount":1,
            "subjectIdList":null,
            "subjectList":null,
            "subjectInfo":{
                "id":null,
                "subjectName":"",
                "subjectGrade":"",
                "subjectTerm":"",
                "subjectUnitList":null
            },
            "buy":0,
            "expireTime":null,
 "imageUrl":"https://gimg2.baidu.com/image_search/src=http%3A%2F%2Fimg.z
cool.cn%2Fcommunity%2F011299558a68d70000001fa34f5128.jpg%401280w_1l_2o_1
00sh.jpg&refer=http%3A%2F%2Fimg.zcool.cn&app=2002&size=f9999,10000&q=a80
&n=0&g=0n&fmt=jpeg?sec=1634921537&t=ca8b0f645e9ff34ff6c4db1f3a2cb18c",
                "userName":null
            }
        ]
    },
    "success":true
}
```

```java
package com.mszlu.xt.web.model;

import lombok.Data;

import java.math.BigDecimal;
import java.util.List;

@Data
public class CourseViewModel {
    private Long id;
    private String courseName;
    private String courseDesc;
    private BigDecimal coursePrice;
    private BigDecimal courseZhePrice;
    private Integer orderTime;
    private Integer studyCount;
    private List<Long> subjectIdList;
    private List<SubjectModel> subjectList;
    private SubjectModel subjectInfo;
    //0 未购买 1 已购买
    private Integer buy;
    private String expireTime;
    private String imageUrl;

    //用户名称
    private String userName;




}
```

1.3.3.1 涉及到的表

课程表：

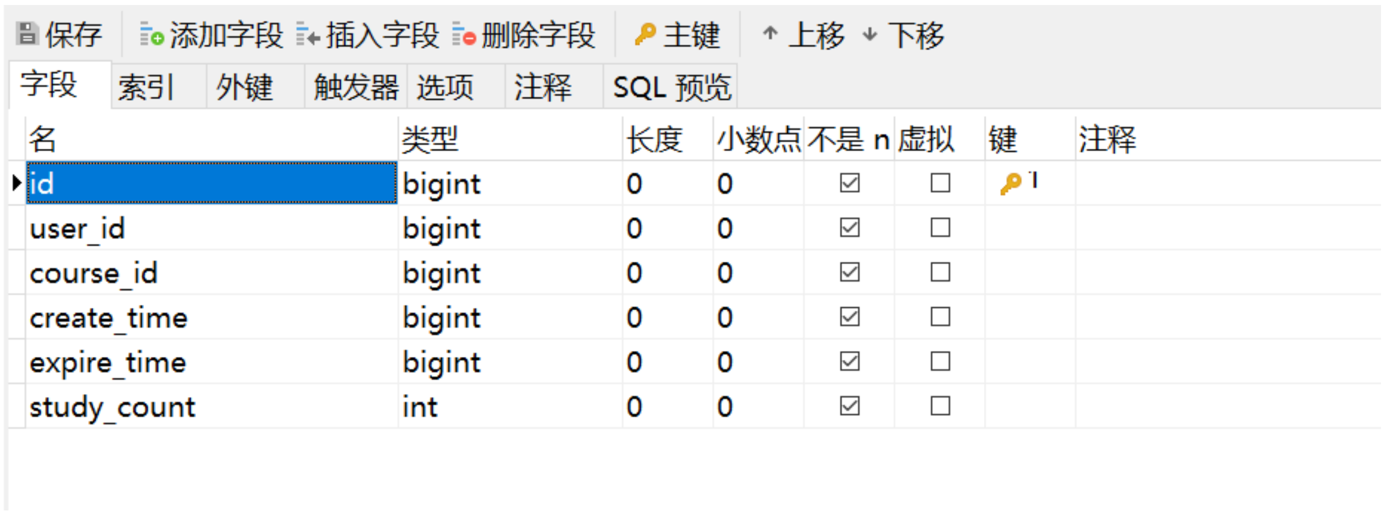


名	类型	长度	小数点	不是 null	虚拟	键	注释
id	bigint	0	0	☑	☐	🔑1	
course_name	varchar	255	0	☑	☐		
course_desc	varchar	255	0	☑	☐		
subjects	varchar	255	0	☑	☐		
course_price	decimal	10	2	☑	☐		
course_zhe_price	decimal	10	2	☑	☐		
course_status	tinyint	0	0	☑	☐		0 正常 1 下架
order_time	int	0	0	☑	☐		
image_url	varchar	255	0	☑	☐		

```java
package com.mszlu.xt.pojo;

import lombok.Data;

import java.math.BigDecimal;

@Data
public class Course {
    private Long id;
    private String courseName;
    private String courseDesc;
    private String subjects;
    private BigDecimal coursePrice;
    private BigDecimal courseZhePrice;
    private Integer orderTime;//天数
    private Integer courseStatus;//正常，下架
    private String imageUrl;

}
```

用户购买课程：

> *此表数据会随着用户量的增多而增大，但考虑到增值较为缓慢，可以暂不做分表考虑*



名	类型	长度	小数点	不是 n	虚拟	键	注释
id	bigint	0	0	☑	☐	🔑1	
user_id	bigint	0	0	☑	☐		
course_id	bigint	0	0	☑	☐		
create_time	bigint	0	0	☑	☐		
expire_time	bigint	0	0	☑	☐		
study_count	int	0	0	☑	☐		

```java
package com.mszlu.xt.pojo;

import lombok.Data;

@Data
public class UserCourse {
    private Long id;
    private Long userId;
    private Long courseId;
    private Long createTime;
    private Long expireTime;
    private Integer studyCount;
}
```

1.3.3.2 代码

Controller:

```java
package com.mszlu.xt.web.api;

import com.mszlu.common.model.CallResult;
import com.mszlu.xt.model.service.CourseService;
import com.mszlu.xt.web.model.params.CourseParam;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.*;



@RestController
```

```java
@RequestMapping("course")
public class CourseApi {

    @Autowired
    private CourseService courseService;

    @PostMapping(value = "courseList")
    public CallResult courseList(@RequestBody CourseParam courseParam){
        return courseService.courseList(courseParam);
    }
}
```

Service:

```java
package com.mszlu.xt.model.service;

import com.mszlu.common.model.CallResult;
import com.mszlu.xt.web.model.params.CourseParam;

public interface CourseService {

    CallResult courseList(CourseParam courseParam);
}
```

```java
package com.mszlu.xt.web.service.impl;

import com.mszlu.common.model.CallResult;
import com.mszlu.common.service.AbstractTemplateAction;
import com.mszlu.xt.model.service.CourseService;
import com.mszlu.xt.web.domain.CourseDomain;
import com.mszlu.xt.web.domain.repository.CourseDomainRepository;
import com.mszlu.xt.web.model.params.CourseParam;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

@Service
public class CourseServiceImpl extends AbstractService implements
CourseService {

    @Autowired
    private CourseDomainRepository courseDomainRepository;
    @Override
    public CallResult courseList(CourseParam courseParam) {
        CourseDomain courseDomain =
courseDomainRepository.createDomain(courseParam);
        return this.serviceTemplate.executeQuery(new
AbstractTemplateAction<Object>() {
            @Override
            public CallResult<Object> doAction() {
                return courseDomain.courseList();
            }
        });
    }
}
```

Domain:

```
package com.mszlu.xt.web.domain;

import com.baomidou.mybatisplus.extension.plugins.pagination.Page;
import com.mszlu.xt.common.login.UserThreadLocal;
import com.mszlu.xt.common.model.CallResult;
import com.mszlu.xt.common.model.ListPageModel;
import com.mszlu.xt.pojo.Course;
import com.mszlu.xt.pojo.UserCourse;
import com.mszlu.xt.web.domain.repository.CourseDomainRepository;
import com.mszlu.xt.web.model.CourseViewModel;
import com.mszlu.xt.web.model.SubjectModel;
import com.mszlu.xt.web.model.params.CourseParam;
import com.mszlu.xt.web.model.params.SubjectParam;
import com.mszlu.xt.web.model.params.UserCourseParam;
import org.apache.commons.lang3.StringUtils;
import org.joda.time.DateTime;
import org.springframework.beans.BeanUtils;

import java.util.ArrayList;
import java.util.List;

public class CourseDomain {

    private CourseDomainRepository courseDomainRepository;
    private CourseParam courseParam;
```

```java
    public CourseDomain(CourseDomainRepository courseDomainRepository,
CourseParam courseParam) {
        this.courseDomainRepository = courseDomainRepository;
        this.courseParam = courseParam;
    }

    public CallResult<Object> courseList() {
        /**
         * 1. 如果根据年级进行查询，需要先找到年级对应的科目列表，根据科目列表去查询
课程列表
         * 2. 如果年级为空，查询全部的课程即可
         * 3. 用户购买课程的信息，课程中科目的名称信息
         * 4. 判断用户是否登录，如果登录 去user_course 表 去查询相关信息
         * 5. 根据课程id，去查询对应的科目名称
         */
        int page = this.courseParam.getPage();
        int pageSize = this.courseParam.getPageSize();
        String subjectGrade = this.courseParam.getSubjectGrade();
        Page<Course> coursePage;
        if (StringUtils.isNotBlank(subjectGrade)){
            coursePage =
this.courseDomainRepository.findCourseByGrade(page,pageSize,subjectGrade
);
        }else{
            coursePage =
this.courseDomainRepository.findAllCourse(page,pageSize);
        }
        List<Course> courseList = coursePage.getRecords();
        List<CourseViewModel> courseViewModels = new ArrayList<>();
        for (Course course : courseList) {
            CourseViewModel courseViewModel = new CourseViewModel();
            BeanUtils.copyProperties(course,courseViewModel);
            //购买的数量
            long studyCount =
this.courseDomainRepository.createUserCourseDomain(new
UserCourseParam()).countUserCourseByCourseId(course.getId());
            courseViewModel.setStudyCount((int) studyCount);
            Long userId = UserThreadLocal.get();
            if (userId != null){
                //代表用户已登录
                UserCourse userCourse =
this.courseDomainRepository.createUserCourseDomain(new
UserCourseParam()).findUserCourse(userId,course.getId(),System.currentTi
meMillis());
                if (userCourse == null){
                    courseViewModel.setBuy(0);
                }else {
                    courseViewModel.setBuy(1);
                    courseViewModel.setExpireTime(new
DateTime(userCourse.getExpireTime()).toString("yyyy-MM-dd"));
                }
            }
            //科目信息，根据课程id查找对应的科目信息
            List<SubjectModel> subjectModelList =
this.courseDomainRepository.createSubjectDomain(new
SubjectParam()).findSubjectListByCourseId(course.getId());
            courseViewModel.setSubjectList(subjectModelList);

 courseViewModel.setSubjectInfo(createSubjectModel(subjectModelList));
            courseViewModels.add(courseViewModel);

        }
        ListPageModel<CourseViewModel> listPageModel = new
ListPageModel<>();
        listPageModel.setSize(coursePage.getTotal());
        listPageModel.setPageCount(coursePage.getPages());
        listPageModel.setPage(page);
        listPageModel.setPageSize(pageSize);
        listPageModel.setList(courseViewModels);
        return CallResult.success(listPageModel);
    }

    private SubjectModel createSubjectModel(List<SubjectModel>
subjectModelList) {
        SubjectModel subjectModel = new SubjectModel();
        StringBuilder nameBuilder = new StringBuilder();
        StringBuilder termBuilder = new StringBuilder();
        for (SubjectModel model : subjectModelList) {
            if
(!nameBuilder.toString().contains(model.getSubjectName())) {
                nameBuilder.append(model.getSubjectName()).append(",");
            }
            if
(!termBuilder.toString().contains(model.getSubjectTerm())) {
                termBuilder.append(model.getSubjectTerm()).append(",");
            }
        }
        String name = nameBuilder.toString();

 subjectModel.setSubjectName(name.substring(0,name.lastIndexOf(",")));
 subjectModel.setSubjectGrade(subjectModelList.get(0).getSubjectGrade())
;
        String term = termBuilder.toString();

 subjectModel.setSubjectTerm(term.substring(0,term.lastIndexOf(",")));
        return subjectModel;
    }

//    public static void main(String[] args) {
//        System.out.println(new
DateTime(1633511129910L).toString("yyyy-MM-dd"));
//        System.out.println(System.currentTimeMillis() + 300 * 24 * 60
* 60 * 1000);
//    }
}
```

```
package com.mszlu.xt.web.domain.repository;

import
com.baomidou.mybatisplus.core.conditions.query.LambdaQueryWrapper;
import com.baomidou.mybatisplus.core.metadata.IPage;
import com.baomidou.mybatisplus.extension.plugins.pagination.Page;
import com.mszlu.xt.pojo.Course;
import com.mszlu.xt.web.dao.CourseMapper;
import com.mszlu.xt.web.domain.CourseDomain;
import com.mszlu.xt.web.domain.SubjectDomain;
import com.mszlu.xt.web.domain.UserCourseDomain;
import com.mszlu.xt.web.model.params.CourseParam;
import com.mszlu.xt.web.model.params.SubjectParam;
import com.mszlu.xt.web.model.params.UserCourseParam;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Component;

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

@Component
public class CourseDomainRepository {

    @Resource
    private CourseMapper courseMapper;
    @Autowired
    private UserCourseDomainRepository userCourseDomainRepository;
    @Autowired
    private SubjectDomainRepository subjectDomainRepository;

    public CourseDomain createDomain(CourseParam courseParam){
        return new CourseDomain(this,courseParam);
    }

    public Page<Course> findCourseByGrade(int currentPage,
                                         int pageSize,
                                         String subjectGrade) {
        //select * from t_course where course_status = 0 and id in
(select course_id from t_course_subject where subject_id in ( select id
from t_subject where subject_grade='七年级') group by course_id)
        Page<Course> page = new Page<>(currentPage,pageSize);
        return courseMapper.findCourseByGrade(page,subjectGrade);
    }

    public Page<Course> findAllCourse(int currentPage,
                                     int pageSize) {
        LambdaQueryWrapper<Course> queryWrapper = new
LambdaQueryWrapper<>();
        queryWrapper.eq(Course::getCourseStatus,0);
        Page<Course> page = new Page<>(currentPage,pageSize);
        Page<Course> courseIPage = courseMapper.selectPage(page,
queryWrapper);
        return courseIPage;
    }

    public UserCourseDomain createUserCourseDomain(UserCourseParam
userCourseParam) {
        return
this.userCourseDomainRepository.createDomain(userCourseParam);
    }

    public SubjectDomain createSubjectDomain(SubjectParam subjectParam)
{
        return this.subjectDomainRepository.createDomain(subjectParam);
    }
}
```

```java
package com.mszlu.xt.web.domain;

import com.baomidou.mybatisplus.extension.plugins.pagination.Page;
import com.mszlu.xt.common.enums.Status;
import com.mszlu.xt.common.model.BusinessCodeEnum;
import com.mszlu.xt.common.model.CallResult;
import com.mszlu.xt.common.model.ListPageModel;
import com.mszlu.xt.common.utils.CommonUtils;
import com.mszlu.xt.pojo.Subject;
import com.mszlu.xt.web.domain.repository.SubjectDomainRepository;
import com.mszlu.xt.web.model.SubjectModel;
import com.mszlu.xt.web.model.params.SubjectParam;
import org.apache.commons.lang3.StringUtils;
import org.springframework.beans.BeanUtils;

import java.util.*;
import java.util.stream.Collectors;

public class SubjectDomain {

    private SubjectDomainRepository subjectDomainRepository;

    private SubjectParam subjectParam;

    public SubjectDomain(SubjectDomainRepository
subjectDomainRepository, SubjectParam subjectParam) {
        this.subjectDomainRepository = subjectDomainRepository;
        this.subjectParam = subjectParam;
    }

    public List<SubjectModel> copyList(List<Subject> subjectList){

        List<SubjectModel> subjectModels = new ArrayList<>();
        for (Subject subject : subjectList) {
            SubjectModel target = new SubjectModel();
            BeanUtils.copyProperties(subject, target);
            subjectModels.add(target);
        }
        return subjectModels;
    }



    public List<SubjectModel> findSubjectListByCourseId(Long courseId) {
        List<Subject> subjectList =
subjectDomainRepository.findSubjectListByCourseId(courseId);
        return copyList(subjectList);
    }
}
```

```
package com.mszlu.xt.web.domain.repository;

import
com.baomidou.mybatisplus.core.conditions.query.LambdaQueryWrapper;
import com.mszlu.xt.pojo.UserCourse;
import com.mszlu.xt.web.dao.UserCourseMapper;
import com.mszlu.xt.web.domain.UserCourseDomain;
import com.mszlu.xt.web.model.params.UserCourseParam;
import org.springframework.stereotype.Component;

import javax.annotation.Resource;

@Component
public class UserCourseDomainRepository {

    @Resource
    private UserCourseMapper userCourseMapper;

    public UserCourseDomain createDomain(UserCourseParam
userCourseParam) {
        return new UserCourseDomain(this,userCourseParam);
    }

    public UserCourse findUserCourse(Long userId, Long courseId, long
currentTime) {
        LambdaQueryWrapper<UserCourse> queryWrapper = new
LambdaQueryWrapper<>();
        queryWrapper.eq(UserCourse::getCourseId,courseId);
        queryWrapper.eq(UserCourse::getUserId,userId);
        queryWrapper.ge(UserCourse::getExpireTime,currentTime);
        return userCourseMapper.selectOne(queryWrapper);
    }

    public long countUserCourseByCourseId(Long courseId) {
        LambdaQueryWrapper<UserCourse> queryWrapper = new
LambdaQueryWrapper<>();
        queryWrapper.eq(UserCourse::getCourseId,courseId);
        return userCourseMapper.selectCount(queryWrapper);
    }
}
```

```
package com.mszlu.xt.web.domain;

import com.mszlu.xt.pojo.UserCourse;
import com.mszlu.xt.web.domain.repository.UserCourseDomainRepository;
import com.mszlu.xt.web.model.params.UserCourseParam;

public class UserCourseDomain {
    private UserCourseDomainRepository userCourseDomainRepository;
    private UserCourseParam userCourseParam;
    public UserCourseDomain(UserCourseDomainRepository
userCourseDomainRepository, UserCourseParam userCourseParam) {
        this.userCourseDomainRepository = userCourseDomainRepository;
        this.userCourseParam = userCourseParam;
    }

    public UserCourse findUserCourse(Long userId, Long courseId,long
currentTime) {
        return
userCourseDomainRepository.findUserCourse(userId,courseId,currentTime);
    }

    public long countUserCourseByCourseId(Long courseId) {
        return
userCourseDomainRepository.countUserCourseByCourseId(courseId);
    }
}
```

```
package com.mszlu.xt.web.domain.repository;

import
com.baomidou.mybatisplus.core.conditions.query.LambdaQueryWrapper;
import com.baomidou.mybatisplus.extension.plugins.pagination.Page;
import com.mszlu.xt.common.enums.Status;
import com.mszlu.xt.pojo.Subject;
import com.mszlu.xt.pojo.SubjectUnit;
import com.mszlu.xt.web.dao.SubjectMapper;
```

```java
import com.mszlu.xt.web.domain.SubjectDomain;
import com.mszlu.xt.web.model.params.SubjectParam;
import org.apache.commons.lang3.StringUtils;
import org.springframework.stereotype.Component;

import javax.annotation.Resource;
import java.util.List;
import java.util.stream.Collectors;

@Component
public class SubjectDomainRepository {

    @Resource
    private SubjectMapper subjectMapper;

    public SubjectDomain createDomain(SubjectParam subjectParam) {
        return new SubjectDomain(this,subjectParam);
    }

    public List<Subject> findSubjectList() {
        LambdaQueryWrapper<Subject> queryWrapper = new
LambdaQueryWrapper<>();
        queryWrapper.eq(Subject::getStatus, Status.NORMAL.getCode());
        return subjectMapper.selectList(queryWrapper);
    }

    public List<Subject> findSubjectListByCourseId(Long courseId) {
        return subjectMapper.findSubjectListByCourseId(courseId);
    }
}
```

Mapper:

```java
package com.mszlu.xt.web.dao;

import com.baomidou.mybatisplus.core.mapper.BaseMapper;
import com.mszlu.xt.pojo.Subject;

import java.util.List;

public interface SubjectMapper extends BaseMapper<Subject> {

    //select * from t_subject where id in (SELECT subject_id FROM
`t_course_subject` where course_id=9)
    List<Subject> findSubjectListByCourseId(Long courseId);
}
```

```xml
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE mapper PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"
        "http://mybatis.org/dtd/mybatis-3-mapper.dtd">

<mapper namespace="com.mszlu.xt.web.dao.SubjectMapper">

    <resultMap id="SubjectMap" type="com.mszlu.xt.pojo.Subject">
        <id column="id" property="id" jdbcType="BIGINT" />
        <result column="subject_name" property="subjectName"
jdbcType="VARCHAR" />
        <result column="subject_grade" property="subjectGrade"
jdbcType="VARCHAR" />
        <result column="subject_term" property="subjectTerm"
jdbcType="VARCHAR" />
    </resultMap>

    <select id="findSubjectListByCourseId" parameterType="long"
resultMap="SubjectMap">
        select * from t_subject where id in (SELECT subject_id FROM
`t_course_subject` where course_id=#{courseId})
    </select>

</mapper>
```

```java
package com.mszlu.xt.web.dao;
```

```java
import com.baomidou.mybatisplus.core.mapper.BaseMapper;
import com.baomidou.mybatisplus.extension.plugins.pagination.Page;
import com.mszlu.xt.pojo.Course;

public interface CourseMapper extends BaseMapper<Course> {
    /**
     * 根据年级 进行课程的列表查询
     *
     * @param page
     * @param subjectGrade
     * @return
     */
    Page<Course> findCourseByGrade(Page<Course> page, String
subjectGrade);
}
```

```xml
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE mapper PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"
        "http://mybatis.org/dtd/mybatis-3-mapper.dtd">

<mapper namespace="com.mszlu.xt.web.dao.CourseMapper">

    <resultMap id="CourseMap" type="com.mszlu.xt.pojo.Course">
        <id column="id" property="id" jdbcType="BIGINT" />
        <result column="course_name" property="courseName"
jdbcType="VARCHAR" />
        <result column="course_desc" property="courseDesc"
jdbcType="VARCHAR" />
        <result column="subjects" property="subjects" jdbcType="VARCHAR"
/>
        <result column="course_price" property="coursePrice"
jdbcType="DECIMAL" />
        <result column="course_zhe_price" property="courseZhePrice"
jdbcType="DECIMAL" />
        <result column="course_status" property="courseStatus"
jdbcType="INTEGER" />
        <result column="order_time" property="orderTime"
jdbcType="INTEGER" />
        <result column="image_url" property="imageUrl"
jdbcType="VARCHAR" />
    </resultMap>
    <select id="findCourseByGrade" parameterType="string"
resultMap="CourseMap">
        select * from t_course where course_status = 0
                                     and id in
                                         (select course_id from
t_course_subject
                                         where subject_id in
                                               ( select id from t_subject
where subject_grade=#{subjectGrade})
                                         group by course_id
                                             )
    </select>

</mapper>
```

```
package com.mszlu.xt.web.dao;

import com.baomidou.mybatisplus.core.mapper.BaseMapper;
import com.mszlu.xt.pojo.UserCourse;

public interface UserCourseMapper extends BaseMapper<UserCourse> {
}
```

```
package com.mszlu.xt.web.dao;

import com.baomidou.mybatisplus.core.mapper.BaseMapper;
import com.mszlu.xt.pojo.CourseSubject;

public interface CourseSubjectMapper extends BaseMapper<CourseSubject> {
}
```

2.1 问题

> *在获取课程列表的时候，未登录用户也可以访问，所以登录拦截器中，将课程列表接口设为了未拦截，也就是无法获取登录用户的信息*

注释掉课程列表的拦截：

```java
@Override
    public void addInterceptors(InterceptorRegistry registry) {
        registry.addInterceptor(loginInterceptor)
                .addPathPatterns("/topic/*")
                .addPathPatterns("/subject/*")
                .addPathPatterns("/course/*")
                .addPathPatterns("/order/*")
                .addPathPatterns("/user/*")
                .addPathPatterns("/i/*")
//                .excludePathPatterns("/course/courseList")
                .excludePathPatterns("/subject/listSubjectNew")
                .excludePathPatterns("/course/subjectInfo")
                .excludePathPatterns("/order/notify")
                .excludePathPatterns("/case/*")
                .excludePathPatterns("/wechat/*")
                .excludePathPatterns("/login/wxLoginCallBack")
                .excludePathPatterns("/i/u/*");
//                .excludePathPatterns("/course/courseList");;
        System.out.println("拦截器");
    }
```

开发一个注解，寓意为需要获取登录信息，但是如果未登录不进行拦截

```java
package com.mszlu.common.annotation;

import java.lang.annotation.*;

@Target(ElementType.METHOD)
@Retention(RetentionPolicy.RUNTIME)
@Documented
public @interface NoAuth {
}
```

改造拦截器：

```java
package com.mszlu.xt.web.handler;

import com.alibaba.fastjson.JSON;
import com.mszlu.common.annotation.NoAuth;
import com.mszlu.common.model.BusinessCodeEnum;
import com.mszlu.common.model.CallResult;
import com.mszlu.common.utils.UserThreadLocal;
import com.mszlu.xt.sso.service.TokenService;
import lombok.extern.slf4j.Slf4j;
import org.apache.commons.lang3.StringUtils;
import org.apache.dubbo.config.annotation.DubboReference;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.data.redis.core.RedisTemplate;
import org.springframework.stereotype.Component;
import org.springframework.web.method.HandlerMethod;
import org.springframework.web.servlet.HandlerInterceptor;
import org.springframework.web.servlet.ModelAndView;

import javax.servlet.http.Cookie;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import java.io.IOException;
import java.io.PrintWriter;
import java.time.Duration;

@Component
@Slf4j
public class LoginInterceptor implements HandlerInterceptor {

    @DubboReference(version = "1.0.0")
    private TokenService tokenService;

    @Override
    public boolean preHandle(HttpServletRequest request,
HttpServletResponse response, Object handler)
            throws Exception {

        log.info("--------------------------start------------------------
---------");
        log.info("request uri:{}",request.getRequestURI());
        log.info("request method:{}",request.getMethod());
        log.info("--------------------------end--------------------------
-------");
        boolean isAuth = true;
        if (!(handler instanceof HandlerMethod)){
            //不是controller的方法访问 直接放行
            return true;
        }
        HandlerMethod handlerMethod = (HandlerMethod) handler;
        boolean hasNoAuth =
handlerMethod.hasMethodAnnotation(NoAuth.class);
        if (hasNoAuth){
            isAuth = false;
```

```java
        }
        Cookie[] cookies = request.getCookies();
        if (cookies == null){
            return handlerResponse(response, isAuth);
        }
        String token = null;
        for (Cookie cookie : cookies) {
            String name = cookie.getName();
            if ("t_token".equals(name)){
                token = cookie.getValue();
            }
        }
        if (token == null){
            return handlerResponse(response, isAuth);
        }
        Long userId = tokenService.checkToken(token);
        if (userId != null){
            UserThreadLocal.put(userId);
            return true;
        }
        return handlerResponse(response, isAuth);
    }

    private boolean handlerResponse(HttpServletResponse response,
boolean isAuth) {
        if (isAuth){
            returnJson(response);
            return false;
        }else {
            return true;
        }
    }

    @Override
    public void postHandle(HttpServletRequest request,
HttpServletResponse response, Object handler,
                           ModelAndView modelAndView) throws Exception {
    }

    @Override
    public void afterCompletion(HttpServletRequest request,
HttpServletResponse response, Object handler, Exception ex)
            throws Exception {
        UserThreadLocal.remove();
    }
    private void returnJson(HttpServletResponse response){
        PrintWriter writer = null;
        response.setCharacterEncoding("UTF-8");
        response.setContentType("application/json; charset=utf-8");
        try {
            writer = response.getWriter();
            CallResult callResult =
CallResult.fail(BusinessCodeEnum.NO_LOGIN.getCode(),"您的登录已失效，请重新
登录");
            writer.print(JSON.toJSONString(callResult));
        } catch (IOException e){
            e.printStackTrace();
        } finally {
            if(writer != null){
                writer.close();
            }
        }
    }
}
```

在课程列表接口上，加注解：

```
  @PostMapping(value = "courseList")
    @NoAuth
    public CallResult courseList(@RequestBody CourseParam courseParam){
        return courseService.courseList(courseParam);
    }
```

2. 优化

2.1 缓存

> *商品列表在业务中，一定是处于一个被频繁访问的状态*

1. 课程列表需要加缓存

加缓存：

这里有个问题，由于课程列表中的内容涉及到当前的登录用户，所以需要将之前的缓存，添加一个参数，是否需要登录用户做为key

```java
package com.mszlu.common.cache;


import java.lang.annotation.*;

@Target({ElementType.METHOD})
@Retention(RetentionPolicy.RUNTIME)
@Documented
public @interface Cache {

    long expire() default 1 * 60 * 1000;

    String name() default "";

    boolean hasUser() default false;

}
```

```java
 @PostMapping(value = "courseList")
    @NoAuth
    @Cache(name = "web_courseList",time = 5*60*1000,hasUser = true)
    public CallResult courseList(@RequestBody CourseParam courseParam){
        return courseService.courseList(courseParam);
    }
```

改造之前的缓存aop：

```
//在做key的时候，将userId考虑进来
String redisKey = name + "::" + className+"::"+methodName+"::"+params;
            if (annotation.hasUser()){
                Long userId = UserThreadLocal.get();
                if (userId != null){
                    redisKey = redisKey + "::" + userId;
                }
            }
```

> *上述时间设为5分钟，也就是数据有变更，最多5分钟之后才能生效，实际业务中可以接受*

2.2 @Enable注解

> *在InitConfig中，使用了@ComponentScan注解，这时候需要想一个问题，如果包多了，是不是每个包都要扫一遍，是不是每个包都要记住呢？*

```
@ComponentScan({"com.mszlu.common.service","com.mszlu.common.cache"})
```

上述的注解使用@Enable开头的注解代替

```
package com.mszlu.common.cache;

import com.mszlu.common.cache.redis.CacheAspect;
import org.springframework.context.annotation.Import;

import java.lang.annotation.*;

@Target({ElementType.TYPE})
@Retention(RetentionPolicy.RUNTIME)
@Documented
@Import(CacheAspect.class)
public @interface EnableCache {

}

```

```
package com.mszlu.common.service;
```

```java
import com.mszlu.common.service.impl.ServiceTemplateImpl;
import org.springframework.context.annotation.Import;

import java.lang.annotation.*;

@Target({ElementType.TYPE})
@Retention(RetentionPolicy.RUNTIME)
@Documented
@Import(ServiceTemplateImpl.class)
public @interface EnableService {
}
```

```java
package com.mszlu.xt.web.config;

import com.mszlu.common.cache.EnableCache;
import com.mszlu.common.service.EnableService;
import org.springframework.context.annotation.ComponentScan;
import org.springframework.context.annotation.Configuration;
import org.springframework.stereotype.Component;

@Configuration
@EnableCache
@EnableService
public class InitConfig {
}
```