# 课程内容

**1、每特教育互联网微服务电商项目课程安排**

**2、互联网微服务电商项目技术的选型**

**3、互联网微服务电商项目技术架构**

**4、构建微服务电商项目模型 会员服务、微信服务**

**5、项目中集成httpfeign客户端调用工具**

**6、使用feign客户端继承模式实现接口通讯**

**7、布置课后作业题**

# 相关说明

**该项目“基于SpringCloud2.x构建微服务电商项目”该项目作者:97后互联网**

**该想项目是采用目前比较流行的SpringBoot/SpringCloud构建微服务电商项目，实现一套串联的微服务电商项目，**

**完全符合一线城市微服务电商的需求，对中国程序猿学习微服务电商架构，有非常大的帮助，该项目涵盖从微服务电商需求讨论、数据库设计、技术选型、互联网安全架构、整合SpringCloud各自组件、分布式基础设施、使Docker+k8s+jenkins实现微服务自动化部署、项目上线域名配置、SEO优化、大数据统计分析等。**

## 学习时间

**学习周期大概四个月时间，每周 2 4 6 晚上20:30-22:30，一个半小时编码课、半个小时面试题辅导。**

**2019年01月03日本周四，开始干我们微服务电商结业项目啦！！希望同学们，坚持哦！**

# 需求讨论与技术架构选型

## 技术选型

**A. 项目采用SpringBoot2.x+SpringCloud2.x构建微服务电商项目**

**1.使用SpringCloudEureka作为注册中心，实现服务治理**

**2.使用Zuul网关框架管理服务请求入口**

**3.使用Ribbon实现本地负载均衡器和FeginHttp客户端调用工具**

**4.使用Hystrix服务保护框架(服务降级、隔离、熔断、限流)**

**5.使用消息总线Stream RabbitMQ和Kafka**

**6.微服务API接口安全控制与单点登陆系统CAS+JWT+Oauth2.0**

**B. 分布式基础设施环境构建**

**1.分布式任务调度平台XXL-Job**

**2.分布式日志采集系统ELK**

**3.分布式事务解决方案LCN**

**4.分布式锁解决方案Zookeeper、Redis**

**5.分布式配置中心携程阿波罗**

**6.高并发分布式全局ID生成雪花算法**

**7.分布式Session框架Spring-Session**

**8.分布式服务追踪与调用链ZipKin**

**C.项目运营与部署环境**

**1.分布式设施环境，统一采用docker安装**

**2.使用jenkins+docker+k8s实现自动部署**

**3.微服务API管理ApiSwagger**

**4.使用GitLab代码管理**

**5.统一采用第三方云数据库**

**6.使用七牛云服务器对静态资源实现加速**

## 课程安排

**第一节 需求讨论与技术架构选型**

**第二节 微服务架构环境搭建Maven私服仓库、GitLiab代码仓库**

**第三节 构建微服务电商项目与数据库实战设计**

**第四节 分布式基础设施之分布式配置中心环境搭建**

**第五节 微服务电商项目-会员服务-联合登陆**

**第六节 微服务电商项目-会员服务-唯一登陆**

**第七节 微服务电商项目-会员服务-SSO联合登陆**

**第八节 微服务电商项目-微服务公众号-构建微信公众号**

**第九节 微服务电商项目-消息服务平台-消息服务平台**

**第十节 微服务电商项目-前端技术-VUE**

**第十一节 微服务电商项目-商品服务-数据设计**

**第十二节 微服务电商项目-商品服务-搜索服务**

**第十三节 微服务电商项目-商品服务-购物车设计**

**第十四节 微服务电商项目-订单服务-订单服务**

**第十五节 微服务电商项目-聚合支付-数据设计**

**第十六节 微服务电商项目-聚合支付-银联支付**

**第十七节 微服务电商项目-聚合支付-支付宝支付**

**第十八节 微服务电商项目-秒杀服务-秒杀抢购**

**第十九节 微服务电商项目-后台运营平台设计**

**第二十节 微服务电商项目ELK日志分析**

**第二十一节 微服务电商项目安全架构**

**第二十二节 微服务电商项目静态资源CDN加速**

**第二十三节 使用jenkins+docker+k8s实现自动部署**

**第二十四节 微服务电商项目上线部署环境**

**第二十五节 微服务电商项目大数据用户行为分析**

**最终以实际课程表格为准。**

## 环境要求

**为了能够更好的学习互联网微服务架构，该项目对环境要求非常高，建议电脑配置CPU在I5、32GB内存或者电脑采用集群化部署。**

**1.JDK统一要求:JDK1.8K**

**2.Maven 统一管理Jar**

**3.统一采用Docker安装软件**

**4.编码统一采用为UTF-8**

**5.开发工具IDE或者Eclipse**

## 系统架构

**详细参考架构文档:**

****

## 构建项目

meite-shop-parent-----公共Pranet接口

-----meite-shop-basics----分布式基础设施

---------meite-shop-basics-springcloud-eureka—注册中心 8010

---------meite-shop-basics-apollo-config-server—阿波罗分布式配置中心

---------meite-shop-basics-springcloud-zuul—统一请求入口

---------meite-shop-basics-xuxueli-xxljob—分布式任务调度平台

---------meite-shop-basics-codingapi-lcn—分布式事务解决框架

---------meite-shop-basics-codingapi- ZipKin —分布式调用链系统

-----meite-shop-service-api提供公共接口

------------ meite-shop-service-api-weixin 微信服务接口

------------ meite-shop-service-api-member会员服务接口

------------ meite-shop-service-api-sso sso服务接口

------------ meite-shop-service-api-item商品服务接口

------------ meite-shop-service-api-search 搜索服务接口

------------ meite-shop-service-api-pay聚合支付平台

------------ meite-shop-service-api-order订单服务接口

------------ meite-shop-service-api-spike 秒杀服务接口

服务接口中包含内存内容: 实体类层、接口层

-----meite-shop-service-impl公共接口的实现

------------ meite-shop-service-weixin 微信服务接口实现

------------ meite-shop-service-member会员服务接口实现

------------ meite-shop-service-api-sso sso服务接口实现

------------ meite-shop-service-tem商品服务接口实现

------------ meite-shop-service-search 搜索服务接口实现

------------ meite-shop-service-pay聚合支付平台接口实现

------------ meite-shop-service-order订单服务接口实现

------------ meite-shop-service-api-spike 秒杀服务接口

-----meite-shop-common 工具类

---------meite-shop-common-core—核心工具类

-----meite-shop-portal 门户平台

--------meite-shop-portal-web 门户网站

--------meite-shop-portal-sso 单点登陆系统

--------meite-shop-portal-search 搜索系统

--------meite-shop-portal-spike 秒杀系统

--------meite-shop-portal-cms 系统

**后期有新的功能在加。**

**预计40个项目**

**如果项目需要被其他项目继承的话，那么该项目选择maven类型为pom类型**

## 版本说明

**SpringCloud2.x以上版本**

## 注意事项

1. [统一采用@GetMapping](mailto:1.统一采用@GetMapping)**或者@PostMapping实现接口映射**

# 环境构建

步骤①创建以下项目，maven类型选择为Pom类型

1. 创建meite-shop-parent
2. 创建meite-shop-basics
3. 创建meite-shop-portal
4. 创建meite-shop-service
5. 创建meite-shop-service-api

步骤②分布式基础设施eureka

1. 创建meite-shop-basics-springcloud-eureka

步骤③创建微信、会员服务接口

1. 创建meite-shop-service-api-member
2. 创建meite-shop-service-api-weixin

步骤④创建微信、会员服务接口实现maven类型选择为Jar类型

1.创建meite-shop-service-member

1.创建meite-shop-service-weixin

## 相关配置信息

#### meite-shop-parent

|  |
| --- |
| <parent>  <groupId>org.springframework.boot</groupId>  <artifactId>spring-boot-starter-parent</artifactId>  <version>2.0.1.RELEASE</version>  </parent>  <dependencies>  <!-- 集成commons工具类 -->  <dependency>  <groupId>org.apache.commons</groupId>  <artifactId>commons-lang3</artifactId>  </dependency>  <!-- 集成lombok 框架 -->  <dependency>  <groupId>org.projectlombok</groupId>  <artifactId>lombok</artifactId>  </dependency>  <dependency>  <groupId>org.springframework.boot</groupId>  <artifactId>spring-boot-starter-test</artifactId>  <scope>test</scope>  </dependency>  <!-- fastjson -->  <dependency>  <groupId>com.alibaba</groupId>  <artifactId>fastjson</artifactId>  <version>1.2.30</version>  </dependency>  <dependency>  <groupId>junit</groupId>  <artifactId>junit</artifactId>  </dependency>  <dependency>  <groupId>commons-net</groupId>  <artifactId>commons-net</artifactId>  <version>3.3</version>  </dependency>  <dependency>  <groupId>dom4j</groupId>  <artifactId>dom4j</artifactId>  </dependency>  <!-- SpringBoot-整合Web组件 -->  <dependency>  <groupId>org.springframework.boot</groupId>  <artifactId>spring-boot-starter-web</artifactId>  </dependency>  <!-- SpringBoot整合eureka客户端 -->  <dependency>  <groupId>org.springframework.cloud</groupId>  <artifactId>spring-cloud-starter-netflix-eureka-client</artifactId>  </dependency>  </dependencies>  <!-- 管理依赖 -->  <dependencyManagement>  <dependencies>  <dependency>  <groupId>org.springframework.cloud</groupId>  <artifactId>spring-cloud-dependencies</artifactId>  <version>Finchley.M7</version>  <type>pom</type>  <scope>import</scope>  </dependency>  </dependencies>  </dependencyManagement>  <!-- 注意： 这里必须要添加， 否者各种依赖有问题 -->  <repositories>  <repository>  <id>spring-milestones</id>  <name>Spring Milestones</name>  <url>https://repo.spring.io/libs-milestone</url>  <snapshots>  <enabled>false</enabled>  </snapshots>  </repository>  </repositories> |

#### meite-shop-basics-springcloud-eureka

##### maven依赖

|  |
| --- |
| <parent>  <groupId>com.mayikt</groupId>  <artifactId>meite-shop-basics</artifactId>  <version>0.0.1-SNAPSHOT</version>  </parent>  <artifactId>meite-shop-basics-springcloud-eureka</artifactId>  <dependencies>  <!--SpringCloud eureka-server -->  <dependency>  <groupId>org.springframework.cloud</groupId>  <artifactId>spring-cloud-starter-netflix-eureka-server</artifactId>  </dependency>  </dependencies> |

##### Application信息

|  |
| --- |
| ###服务端口号  server:  port: 8100  ###eureka 基本信息配置  eureka:  instance:  ###注册到eurekaip地址  hostname: 127.0.0.1  client:  serviceUrl:  defaultZone: http://${eureka.instance.hostname}:${server.port}/eureka/  ###因为自己是为注册中心，不需要自己注册自己  register-with-eureka: **false**  ###因为自己是为注册中心，不需要检索服务  fetch-registry: **false** |

#### meite-shop-service

##### maven依赖

|  |
| --- |
| <dependencies>  <!-- springcloud feign组件 -->  <dependency>  <groupId>org.springframework.cloud</groupId>  <artifactId>spring-cloud-starter-openfeign</artifactId>  </dependency>  </dependencies> |

#### meite-shop-service-weixin

##### 配置文件

|  |
| --- |
| ###服务启动端口号  server:  port: 8200  ###服务名称(服务注册到eureka名称)  spring:  application:  name: app-mayikt-weixin  ###服务注册到eureka地址  eureka:  client:  service-url:  defaultZone: http://localhost:8100/eureka |

##### AppWeiXin

|  |
| --- |
| @SpringBootApplication  @EnableEurekaClient  **public** **class** AppWeiXin {  **public** **static** **void** main(String[] args) {  SpringApplication.*run*(AppWeiXin.**class**, args);  }  } |

#### meite-shop-service-member

##### Application信息

|  |
| --- |
| ###服务启动端口号  server:  port: 8300  ###服务名称(服务注册到eureka名称)  spring:  application:  name: app-mayikt-member  ###服务注册到eureka地址  eureka:  client:  service-url:  defaultZone: http://localhost:8100/eureka |

##### AppMember

|  |
| --- |
| @SpringBootApplication  @EnableEurekaClient  @EnableFeignClients  **public** **class** AppMember {  **public** **static** **void** main(String[] args) {  SpringApplication.*run*(AppMember.**class**, args);  }  } |

# 课后作业题

## 项目相关

你们微服务电商项目？使用了那些技术？项目是如何架构的？

你们项目是如何进行拆分服务的？

你参与了你们项目那些模块？

完成该接口环境搭建。

## 技术相关

微服务架构与SOA架构的区别？

SpringCloud与Dubbo的区别

如果在架构项目中，提前思考项目如何能够支持多地方来源。

PC端、移动端（安卓、IOS）H5(微信公众号) 小程序端

你在做项目中？遇到了那些问题？

回答分布式解决方案一些问题。