个人信息

姓名: 吴东磊

籍贯: 江苏

性别: 男

电话: 18862859383

邮箱: 18862859383@163.com

求职意向: 测试开发

教育背景

2022.09-2026.06　南京工业大学　计算机科学与技术

四六级已过，多次获得校综合奖学金，GPA=3.42/4.0

专业技能

- 具备的扎实的编程基础，熟悉掌握 Java，熟悉 JVM，熟悉若依框架，Gmall 商城框架，有 web 开发经历
- 熟悉 SpringBoot、Spring MVC、Mybatis 后端框架
- 熟悉 MySQL，对 Mysql 的锁，事务，索引有一定的理解
- 熟练掌握 redis 缓存,对 redis 基本数据结构,RDB/AOF 存储原理,redis 主从集群有一定理解,对缓存穿透、缓存击穿、缓存雪崩等场景有深入理解；
- 熟悉 GC 常用算法，熟悉常用垃圾收集器，对 JVM 调优有一定了解
- 熟悉常用的分布式解决方案,例如:分布式锁,分布式事务技术
- 在实习过程中熟悉并使用 MySQL、Redis、RocketMQ、ES 还有流程引擎
- 实习过程中了解了微服务架构，nacos
- 熟悉使用 Git 进行项目管理，能够有效地协助团队进行代码开发和协作。
- 有丰富的联调经验

实习经历

南京六朝云智科技有限公司(阿里外包)

Java 后端开发实习生

2024 年 11 月 21 日-2025 年 4 月 3 日

技术栈：后端采用 Springboot 框架；微服务架构，服务间采用 dubbo 接口调用。基于阿里云 Gmall 商城标品开发；I18n 国际化；nacos；流程引擎；中间件（MySQL,Mybatis-plus，Redis，RocketMQ，Kafka，ES，调度任务 - XXL-JOB - 分布式任务调度平台，ELK 追踪日志，MinIO，k8s 部署，MongoDB）；项目中还应用了多种设计模式。

项目背景：

项目为哈萨克斯坦中央银行的电商项目 - 平台型 B2B2C，有 C 端线上商城，B 端运营管理平台及商家后台，提供线上交易服务，支持商家入驻；项目采用微服务架构，服务间采用 dubbo 接口调用。

项目是基于阿里云 Gmall 商城标品开发，包含了 28 个子项目，比如说有交易模块、营销模块、支付模块、聊天模块（gim）、会员模块、商品模块、中间件模块、业务引擎（流程引擎等）模块等等。项目中的核心模块都是由两部分组成：center 和 platform，以交易模块为例，分为 trade-center 和 trade-platform 两个子项目。platform 提供了领域模型扩展，流程扩展，业务定制；center 引用了 platform，主要是完成一些定制业务逻辑；一般核心业务都是写在 platform 项目中，在部署时会把 platform 项目打成 jar 包，并依赖到 center 项目中。项目通过 RocketMQ 实现服务解耦与异步通信，基于 ES 进行商品搜索。
项目中还应用了多种设计模式，其中流程引擎应用了工厂模式；业务代码中也应用了门户模式（将订单、支付、售后等等模块的接口放在一起便于查找）；项目代码中也用到了构建器模式用于分布构建复杂对象。

主要职责：

在实习过程中我熟悉并使用了 MySQL、RocketMQ、ES、nacos、流程引擎等等技术。我负责的主要是交易那块的业务，刚开始的话，以改 bug 为主，在改 bug 过程中熟悉项目代码。比如说创建订单接口，在项目中涉及到了多个服务，当调用创建订单接口时，首先 gmall-frontmanager 服务会对请求参数进行一系列的校验封装，然后通过 dubbo 远程调用 trade-platform 服务，在 trade-platform 服务中首先会对传过来的对象进行转化，然后调用流程引擎来处理该业务。以创建订单该业务为例，在流程引擎中会经过：基本入参检查、业务流程这两个状态，在业务流程状态里面有很多 node 即业务处理逻辑，举几个例子：基

础校验、拆包、收件人查询检查、商品查询检查、价格计算、库存、保存等等业务逻辑。通过流程引擎可以实现代码的解耦,如果业务逻辑有调整,可以很方便增加或减少相应的 node。

在实习过程中我也熟悉并使用了 RocketMQ,在使用 RocketMQ 的过程中我也遇到了一些问题，比如说 RocketMQ 消费状态显示为：NOT_CONSUME_YET（可视化界面上显示生产者成功发送了消息，但是消费者没有消费）。后来排查，猜测原因可能是有其它的应用错用了消费组名，RocketMQ 将消息意外投递到该应用中去了，后来将消费组名更改后，消费者就成功消费了。然后的话就是这次实习期间我对使用消息队列目的的理解：一个是为了业务解耦还有一个就是为了应对高并发场景。以创建订单为例，如果我没有使用消息队列，业务代码需要同步调用积分发放、分账处理等多个下游服务，任何一个下游服务异常都会导致主流程中断，并且在高并发场景下同步调用可能会引发系统雪崩。而如果我使用了消息队列，那么当订单完成后，我会将与该订单相关的数据（比如说 primaryOrderId，orderId，primaryOrderStatus 这些数据）封装成消息发送到指定的 topic 中去，RocketMQ 采用长轮询机制主动推送消息到消费者，而非被动的"扫描"机制。消费者订阅了该 topic，并持续监听其中的消息，一旦有新消息到达，消费者便会自动接收，并调用相关的方法（会将消息载体以参数的形式传入进去）。使用消息队列会让订单服务不再感知下游业务的具体实现，只需关注自身业务逻辑；同时消息队列也可以进行异步削峰，突发流量被消息队列缓冲，将大量的消息直接放到 MQ 中，然后系统去按自己最大的消费能力去消费这些消息，避免下游服务被冲垮。

在实习过程中，我也熟悉并使用了 ES，在使用 ES 过程中我也遇到了一些问题，第一个问题是：在原索引中添加了一个新字段 seller_name，新添加的字段映射不会应用到旧文档中，只会对映射更新后新写入的文档生效。当使用 exists 查询时，ES 会检查字段是否在索引中定义且非空值。由于旧文档的 seller_name 字段未被新映射重新索引，它们会被视为“不存在”。这时候需要通过 _update_by_query 强制重新索引所有旧文档，使新映射生效。第二个问题是：ES 与数据库不同步的问题，在实习期间，经常遇到测试或开发人员直接在数据库中添加数据导致线上环境出现 bug 的问题。

在实习过程中，我还参与了一个小场景的算法优化 - 如何分配最少的城市来运送所需的商？简单来说就是买家在商城下单了很多商品，这些商品分属在不同的城市，现在有一个业务需求：需要设计一个算法使得分配最少的城市来运送买家购买的商品。

实习项目的技术架构 - 应用分层架构：首先对于读请求和写请求，通过 DTO 将请求的参数进行转换校验；然后进入业务层，这边负责 center 核心业务处理 还有 数据服务组装、模型转换、日志、异常流处理等；业务层也可以调用 repository，repository 提供对 DAO 数据库访问的封装，Adaptor（Manager）提供对其他中心 RPC 接口的封装和适配、或 rest 接口的调用，也可以封装调用链路中独立的业务模块。

当然在实习过程中，我也学到了一些非常实用的小技巧，比如说看日志：在将本地代码

发布到线上环境后，时常会出现线上服务挂掉的情况，这时最好的办法就是使用终端模拟器连接线上服务器查看日志，找到报错原因；比如说对集合元素进行流操作处理，可以很方便的将流中的元素进行转换比如 .map() 操作可以将原本存储 ItemSkuId 的列表，转换为存储 ItemSkuId 中的 cartId（ItemSkuId 的一个属性）的列表；比如说在 ELK 的 Dev Tools 控制台页面编写一些 ES 查询语句查看某索引库下的文档或者该索引的映射。

南京达牧网络科技有限公司

Java 后端开发实习生

2024 年 7 月 1 日 - 2024 年 8 月 30 日

技术栈：SpringBoot、MySQL、若依框架

主要职责：

1. 参与完成南京交通建管集团数智建养管理平台中智能决策报告模块的工作，项目中需要用到道路的车辙、平整度等数据，我在若依框架的基础上开发和维护了多个接口，确保从数据库中提取出所需要的数据。
2. 由于甲方需要，将公司很早之前用 .NET 开发的项目转化为用 Java 语言编写；我在充分理解原项目代码的基础上，基于若依框架进行项目的开发，对原项目的每一个功能，在理解了其代码逻辑后，将每一个功能通过三层架构封装为接口；项目完成后，对比发现 Java 要比用 .NET 开发的项目快的多。
3. 参与完成南京交通局“最美交通人”投票系统项目的开发，该项目的功能主要由三部分组成：所有候选人信息的展示、投票功能的实现、具体候选人信息的展示；基于若依框架将三个功能封装为接口；为了预防爬虫对投票系统的干扰，我在设计接口时，通过获取并接收客户端的微信号，并将其存储在数据库中，规定每一天同一个微信号只能投十票，否则投票无效；同时，我还深入理解分析了若依框架验证码功能的源码，在充分理解了若依框架验证码的逻辑后，我在起始页添加了一个验证码功能，防止刷票。

项目经历

项目一：茶语道

2024.4-2024.6

软件架构：SpringBoot+SpringCloud+Redis+RabbitMQ+MySQL 等技术。

项目描述：茶物语是一个综合性茶业平台，旨在连接茶业各个环节，提供茶叶生产、茶文化传承、茶艺学习和茶叶品鉴等多种服务。该项目的目标是促进茶文化的传播和推广，打造全方位的茶文化生态系统。项目分为两大系统，茶源网站和茶源后台管理系统，其中茶源网站包括用户登录、主页、商品详情、视频、社区、点赞评论、购物车、订单等模块，后台管理系统包括管理员登录、管理用户信息、管理品类信息、销量信息、管理订单信息等模块。

模块负责：

1.首页:使用 ElasticSearch 根据关键词进行搜索，像咨询类型，茶山茶厂、茶庄、茶器、茶艺师、茶业等的机构类型进行一个条件搜索。

2.茶评:使用 ElasticSearch 根据茶类，评分，产区，参考价格进行搜索，主要是展示对于各种茶叶的总评，对于每一种茶叶都会有相应的评分，可以根据评分，价格或是人气对于茶叶进行一个展示，使用 freemarker 进行一个静态详情的展示，实现高并发。相应的展示出一个茶语的评分排行。

3.视频:茶艺师会发布一些自己对某种茶品理解的视频，付费的课程是需要用户购买的，我们增加了视频续播功能来提高用户体验，这里利用合并写请求和异步 MQ 来减轻数据库的并发压力来完成视频续播。

4.订单:使用策略模式，通过异步 MQ 来完成订单的流量消峰并生成订单，并完成分库分表。解决订单重复提交的问题。

5.优惠券:实现优惠券的发放功能，部署集群来保证高可用，基于 Redisson 的分布式锁解决优惠券的并发控制问题。设计了兑换码的生成方案。

6.支付:对接微信支付 API、支付宝支付 SDK、银联支付、操作类型包括退款、对账。解决超时未支付订单取消问题。

https://github.com/wdlwdlwdlwdlwdl/competition

项目二：ImmunoFlash--基于 Transformer 的免疫大语言模型

2023.12-2024.4

该项目开发了一个智能预测系统，旨在通过大语言模型预测 TCR 与抗原的亲和度，特别针对癌症免疫治疗中的个性化治疗方案。该系统通过深度学习技术（包括 Transformer 模型）进行 TCR 和抗原结合亲和度的分析，并且支持 TCR 与抗原的 3D 分子对接可视化。

项目职责

模型侧：

1. 负责大规模 TCR-抗原序列的收集与预处理（统一长度、去除首尾密码子）；基于自研 ProtFlash 模型通过掩码语言建模进行预训练（随机生成长度 3–5 的序列掩码进行训练）；预训练后加载模型权重，在经验证的 227,798 个配对数据上微调模型（调整学习率、冻结层等），以提升生物序列预测精度。
2. 引入 FGM 对抗训练增强模型泛化能力，通过向输入嵌入添加微扰生成对抗样本，有效提高模型对输入扰动的鲁棒性。
3. 使用 DeepSpeed 分布式训练框架（v0.13.5）加速大规模模型训练，通过混合精度训练（fp16）和 ZeRO-2 技术优化显存与计算效率，结合分层学习率和轻度正则化策略提升训练效率。
4. 对模型性能进行全面评估：在独立测试集、Triple 测试集和 COVID-19 测试集上使用 ROC-AUC、Accuracy、MCC、F1 Score 等指标进行对比分析，验证模型在标准和极端场景下的准确性和泛化能力。

工程侧：

1. 后端实现：采用 Java Spring Boot/Spring MVC 框架设计接口，处理前端请求并调用模型计算；前端提供文件上传和手动输入功能，用户提交 TCR 和抗原序列后通过后台接口调用模型计算，结果以表格形式返回并支持下载或邮件发送。
2. 可视化模块：集成 TCR-抗原分子对接功能。用户上传 TCR 和抗原的 PDB 文件后，利用网站集成的 HDOCK 对接算法将其对接，生成 GIF 图像并返回前端，从而将复杂分子交互直观可视化。

项目三：微信核心功能测试 链接地址：https://github.com/wdlwdlwdlwdlwdl/zuo-pin-ji.git

项目背景：

微信作为中国最广泛使用的社交工具，其聊天页、通讯录、发现、我等核心模块的功能和性能稳定性对用户体验至关重要。本项目针对这些核心模块进行全面测试，验证其功能正确性和性能稳定性。

测试方案设计：

测试策略：制定了功能测试、性能测试和兼容性测试方案。其中功能测试包括手动执行核心功能和编写自动化脚本的组合方式；性能测试评估响应时间、内存/CPU 占用等指标；兼容性测试覆盖不同手机型号、系统版本等场景。

环境搭建：搭建 Android 测试环境，选用 OPPO Reno8 Pro+手机，安装微信 8.0.42 版本，工具则使用 PyCharm 和 UIAutomator2 进行自动化脚本开发。

手动测试：

覆盖**微信聊天页**的文字/图片消息收发、聊天记录管理、消息搜索等功能；

覆盖**通讯录**的好友添加、删除、分组、拉黑及联系人信息查看等功能；
覆盖**发现页（朋友圈）**的动态发布和浏览，包括发布文字、图片、视频动态及评论点赞等场景；
覆盖**“我”**模块中的设置、个人信息查看与修改、支付流程等功能。
通过高覆盖率的手动测试用例，验证了各功能场景的正确性，及时发现并定位潜在缺陷，确保了产品功能的健壮性。
自动化测试：
使用 **Python + UIAutomator2 + Appium** 对聊天、通讯录、发现等核心场景进行自动化回归测试。
编写自动化脚本模拟用户操作（如发送消息、添加朋友、发布朋友圈动态、朋友圈点赞评论等），提高测试执行效率并复用测试用例，实现了对主要功能的快速验证。
问题与解决：
在测试过程中发现自动化脚本在微信高版本中存在元素定位失败的问题（部分 resourceId 无法获取），解决方法是：使用低版本的微信（8.0.42）。
项目成果：
项目完成后编写了详尽的测试文档 (测试计划、测试用例、测试报告等) , 系统总结了 Android 端测试方法及手动/自动化测试结合的实践经验。
通过该项目，我深刻理解了移动应用质量评估的方法，提高了独立设计并实施完整测试流程的能力，对产品质量评估和潜在风险分析有了系统性的认识。

竞赛经历

校二等奖学金 - 2023.4.20
江苏省第二十届高等数学竞赛本科 A 组三等奖 - 2023.7
第十七届中国大学生计算机设计大赛三等奖 - 2024.5
第十六届蓝桥杯全国软件和信息技术专业人才大赛全国总决赛 C/C++程序设计大赛二等奖 - 2025.6.23
第六届全球校园人工智能算法精英大赛全国总决赛一等奖

证明材料

作品集链接：
https://github.com/wdlwdlwdlwdlwdl/zuo-pin-ji.git

实习证明材料链接：
https://github.com/wdlwdlwdlwdlwdl/shi-xi-zheng-ming.git

竞赛证明材料链接：
https://github.com/wdlwdlwdlwdlwdl/jing-sai-zheng-ming.git