



基本信息

姓名：梁金鹏
性别：男
政治面貌：共青团员
邮箱：1429542851@qq.com
意向城市：杭州
期待薪资：8k-12k

年龄：25
民族：汉
电话：18623750372
现居：杭州
求职岗位：全栈工程师/后端开发
到岗时间：随时到岗



教育背景

2019.09-2023.06 河南科技大学 (本科)
2023.09-2026.06 安徽师范大学 计算机技术 (硕士)
研究方向：深度学习(图像识别)
主修课程：算法设计与分析(89)、机器学习(89)、高级计算机网络(82)、高级软件工程(85)、英语阅读写作(92)

专业技能

- 熟练使用 Cursor、Codex 等工具进行代码生成、重构、单测编写与 Bug 定位，提升日常开发效率。
- 熟练使用 Golang 编程，了解 channel、goroutine 及使用方法
- 熟悉 Redis，了解 Redis 的存储数据类型、基本操作、基本原理
- 熟悉 MySQL，能使用 gorm，sqlx 对数据库进行基础操作如增删改查等
- 熟悉 Linux 常用指令，常见端口占用排查，系统服务的使用情况等
- 熟悉 Docker 常用命令和部署方法
- 通过 CET-4，计算机二级

项目经历

2026.09-至今 自托管云盘与文件同步多端平台(<https://jpcoding.xyz>) 全栈
介绍：面向个人和小团队构建多端云盘，支持文件管理、**大文件传输**、历史版本和**离线同步**。实现**桌面**、**网页**、**iOS** 和**微信小程序**多端落地。
技术栈：Go、PostgreSQL、Redis、对象存储、React、Docker
项目亮点：
1.设计 Go 后端的文件与版本数据结构，将目录信息和文件内容分别保存，支持移动、重命名及历史版本查询。
2.实现大文件分片上传、失败后继续上传、断点下载和文件校验，减少网络中断导致的重复传输。
3.设计上传进度记录、配额预留和重复请求处理，使超时或进程中断后的上传结果可以查询和恢复。
4.实现用户内重复文件内容复用、回收站与历史版本清理，支持误删恢复并回收过期数据。
5.设计文件变更记录与冲突副本保留机制，结合本地任务记录，实现 Windows 选定目录双向同步及断线恢复。
6.完善登录、设备管理、文件权限与分享限制，并提供操作记录，保护不同用户的文件访问。
7.构建桌面和网页文件管理入口，复用后端接口，提供浏览、上传下载、版本查看和分享操作。

2025.07-2025.08 **个人博客网站**(https://gitee.com/jpcoding/personal_blog) 全栈

技术栈: Go、Gin、sqlx、MySQL、Redis、JWT、Zap、Swagger、Nginx、DockerCompose

项目亮点:

- 1.基于 Gin 分层实现用户、社区、文章和点赞等 RESTful API, 使用 Swagger 自动生成并维护接口文档。
- 2.采用 JWT Access Token 与 Refresh Token 双令牌认证机制, 通过 Redis 保存会话状态, 支持令牌轮换、主动注销和会话撤销, 保证多实例环境下登录状态一致。
- 3.基于 Redis 实现分布式令牌桶限流, 在 Redis 异常时降级为进程内限流, 兼顾突发流量处理能力和服务可用性。
- 4.使用 Snowflake 算法生成趋势递增的 64 位分布式 ID, 避免数据库自增 ID 在多实例环境中的冲突; 接口层将 ID 序列化为字符串, 解决 JavaScript 大整数精度丢失问题。
- 5.使用 sqlx 封装 MySQL 数据访问层, 为文章时间线、社区筛选、作者文章及点赞关系设计索引和事务操作, 保证点赞计数与点赞记录的一致性。

2026.09-至今 **在线简历平台**(<https://gitee.com/jpcoding/online-resume>) 全栈

技术栈: Vue 3、TypeScript、Vite、Pinia、FastAPI、SQLAlchemy Async、PostgreSQL、Redis、MinIO、Docker Compose、Nginx

- 1.设计并实现多用户在线简历管理平台, 支持邮箱注册验证、登录、会话管理、密码重置及认证接口限流。
- 2.实现草稿与不可变发布版本机制, 支持版本命名、分类、历史版本编辑, 避免已分享简历被后续编辑直接篡改。
- 3.开发结构化表单与 Word 风格双编辑模式, 支持简历模板、DOCX 导入及头像上传。
- 4.实现固定版本分享链接与二维码访问, 提供适配 PC 和移动 H5 的公开简历, 记录访问时间和下载次数。
- 5.实现访客自主选择二维码和文字水印的 PDF、DOCX/WPS 兼容、Markdown 导出。
- 6.使用 FastAPI + Pydantic 构建版本化 REST API, 结合 SQLAlchemy 异步访问 PostgreSQL; 使用 Redis 实现注册、登录、邮件令牌等接口限流。
- 7.使用 Docker Compose 部署 PostgreSQL、Redis、MinIO、Mailpit 等基础服务; 通过 systemd 管理 FastAPI 进程, Nginx 提供静态前端与 API 反向代理。

自我评价

- 拥有计算机专业知识
- 拥有良好的沟通和协调能力, 善于应变, 能够快速适应新环境
- 自制力强, 善于主动思考和行动, 具有较强的抗压能力
- 积极学习新技术, 乐于解决具有挑战性的问题