

# WaveYo

|         |                                                                       |       |                                                           |
|---------|-----------------------------------------------------------------------|-------|-----------------------------------------------------------|
| 求职意向：   | Golang 后端开发工程师（2027 届校招）                                              | 期望城市： | 上海、深圳、广州、北京、南京、杭州、合肥                                      |
| 期望薪资：   | 面议                                                                    | 邮箱：   | waveyo@waveyo.cn                                          |
| GitHub： | <a href="https://github.com/WavesMan">https://github.com/WavesMan</a> | 个人主页： | <a href="https://www.waveyo.cn">https://www.waveyo.cn</a> |

## 个人简介

2027 届计算机网络技术专业在读，求职 Golang 后端开发工程师。具备 Go 后端实习及个人项目开发经历，能够围绕业务需求完成接口设计、数据建模、并发状态管理和 Linux 部署联调。项目覆盖对象存储、软件镜像、网关与查询服务，实践多级缓存、异步任务和服务注册；关注超时回收、异常分类、性能验证与可追踪交付。

## 教育经历

|          |                              |             |
|----------|------------------------------|-------------|
| 长沙南方职业学院 | 计算机网络技术 · 大专（在读，预计 2027 年毕业） | 2024 - 2027 |
|----------|------------------------------|-------------|

## 实习与工作经历

|                       |              |
|-----------------------|--------------|
| 上海旭诚网络科技有限公司   IT 实习生 | 2026.09 - 至今 |
|-----------------------|--------------|

参与服务器与算力环境部署支持，独立编写测试脚本、执行推理服务压测并汇总填写结果。

- 测试编排与异常收敛：**在 Kubernetes Pod 容器内使用 Bash 调用 vllm bench serve，将 3 种输入/输出长度与 1/12/24/50 并发上限组合为 12 组测试；组间串行、组内并发，设置冷却间隔，记录命令、日志和状态，区分请求失败、缺失指标等异常并停止后续任务。
- 数据复核与部署协作：**独立以 Python 按测试参数匹配 12 组 × 5 列共 60 个指标单元格，校验状态、数值及目标表格，保留备份；记录 12 组均成功、失败请求为 0。协助超聚变/NVIDIA GPU 物理主机及 Qwen3.6-27B 部署，压测文件上传和原始数据导出由团队协作完成。

|                      |                   |
|----------------------|-------------------|
| 安徽羚通科技有限公司   后端开发实习生 | 2026/07 - 2026/08 |
|----------------------|-------------------|

参与 EasyNVR/go2rtc 摄像头接入相关 Go 服务开发，围绕设备控制、长连接通信、媒体流生命周期和部署联调推进接口实现及异常修复。

- 长连接与请求配对：**实现主动 WebSocket Client 和流控制 SDK，以 req\_id 与等待池关联异步响应，配置 3 秒请求超时及连接异常释放；通过 Ping/Pong、读 deadline 维护 90 秒心跳窗口，支持播放、停止与保活接口。
- 并发状态与资源回收：**实现流超时回收，通过 expiryChanged 通知更新过期状态，使用任务身份比较与 CompareAndDelete，避免旧回收任务误删续期后的新任务；统一 DID + 通道组合流标识，降低多设备、多通道状态混淆。
- 故障定位与风险处理：**结合 RAW 时间轴、FLV 诊断及生命周期日志排查重连后媒体代际、旧流状态和关键帧队列问题；对永久鉴权错误终止无效重试，增加 URL 指纹与鉴权字段检查，并处理异常凭证覆盖和密钥并发初始化问题。
- 接口验证与服务化：**使用 Python 分段验证认证、Token 缓存、签名与响应解密，14 项离线测试通过；维护 OpenAPI、Consul 注册/幂等注销和健康检查，在 WSL2 Debian、Docker、FnOS、MinIO 环境排查代理路径及 S3 region 配置问题。

## 项目经历

|                                        |                   |
|----------------------------------------|-------------------|
| YoOSF-API / YoMirrorSite   对象存储与软件镜像服务 | 2025/04 - 2026/05 |
|----------------------------------------|-------------------|

个人项目 / Go 后端开发

基于 Go Fiber、Redis、GORM/PostgreSQL 和 S3 构建文件服务，并扩展 GitHub Release 自动同步，将文件访问、同步调度与元数据管理拆分为独立职责。

- 缓存架构与性能：**组合本地 LRU/LFU 与 Redis，针对文件列表采用分片缓存、异步刷新和锁保护，减少重复回源；YoOSF-API README 压测记录核心查询由约 28k 提升至 41.7k QPS，约提升 49%，为项目测试环境下的对比结果。

- **分布式互斥与同步**: 提供增量、仅保留最新、全量历史 3 种同步策略, 按仓库配置任务; 使用 Redis 锁控制同步互斥, 组织 GitHub 检测→流式下载直传 S3→元数据持久化→缓存更新, 避免中间文件落盘并支持同步记录追踪。
- **数据与依赖治理**: 以 Software/Version/SyncLog 3 类 Repository 隔离持久化操作, 实现分页、级联删除与 AutoMigrate; /health 检查 Redis、S3、PostgreSQL 3 项依赖, 结合 Docker/systemd 与 GitHub Actions 完成构建发布。
- **审查与迭代**: 围绕安全修复、前端缺陷、同步规则增强开展 PR #4/#5/#6 共 3 轮 AI 辅助代码审查并合并, 保留人工复核; 配套管理页面承接版本查询与同步操作, 将问题拆分为接口、规则和验证任务。

## YoBFF | Go 网关与管理控制面

2026

个人项目 / 后端与管理端集成

基于 Go net/http、SQLite 和内嵌 React UI 开发网关, 按代理数据面与管理控制面组织职责, 以单进程交付路由、配置和审计能力。

- **路由与转发设计**: 实现 Host 路由、节点权重、默认池及 fallback 池, 处理 HTTPS/SNI 证书匹配、HSTS 和可信代理 CIDR; 通过 OpenAPI 明确配置、站点和证书管理接口, 降低代理逻辑与管理功能的耦合。
- **配置与变更风险**: 实现环境变量覆盖、配置校验、watcher 热重载、版本快照与回滚, 使用 SQLite 持久化配置和站点状态; 围绕原子快照与配置切换整理技术复盘, 关注变更过程中请求读取的一致性。
- **权限与问题追踪**: 实现 Bearer Token 登录、失败次数触发验证码及审计查询, 支持按 action/operator/target/time 筛选和分页; 联调前后端筛选状态, 保留操作主体、目标与时间, 为配置故障复盘提供依据。

## ip-api | IP 查询与可观测服务

个人项目

Go 服务开发

使用 Go net/http、Redis、PostgreSQL 及本地 IP 数据库组织查询服务, 覆盖缓存、回退、插件融合和监控指标。

- **查询与扩展边界**: 组织 KV 覆盖、Redis 热点缓存、本地缓存、数据库范围回退与插件融合链路; 使用布隆去重减少统计重复写入, 以插件健康状态、聚合结果和置信度组织不同来源数据。
- **监控与流量保护**: 接入 Prometheus, 记录请求量、耗时、Redis 命中/未命中及插件调用成功/失败/耗时; 通过令牌桶限流和源站防护中间件控制访问, 为分析慢请求、缓存效果与外部依赖异常提供指标。

## 专业技能

- **Go 开发**: Go、net/http、Fiber、HTTP/REST、WebSocket、OpenAPI; 接口分层、请求响应配对、超时控制与并发任务生命周期管理
- **数据与缓存**: Redis、本地 LRU/LFU、缓存分片与异步刷新; GORM/PostgreSQL、SQLite、Repository 分层、分页及元数据持久化; S3 兼容对象存储
- **分布式基础**: 了解服务化、异步处理、高可用与可扩展设计, 以及超时、重试、幂等和一致性问题; 实践 Redis 互斥锁、Consul 注册/注销与依赖健康检查
- **工程交付**: Linux、Docker/Compose、systemd、Git/GitHub Actions、Prometheus、Wireshark/tcpdump; Python/Bash 测试与数据处理; AI 辅助开发及代码审查后的人工复核