**广 州 商 学 院**

**实验报告（第 3 次）**

实验名称 基于Redis实现分布式锁

实验时间 2025.04.01

同组同学 唐璇 小组分工 全职

**一、实验目的**

1. 掌握分布式锁的基本概念与作用：理解分布式锁在多实例、多线程环境中的重要性，特别是在高并发场景下确保数据一致性和操作的原子性。学习分布式锁如何防止多个客户端同时访问共享资源，避免竞争条件和数据不一致问题。

2. 熟悉Redis作为分布式锁的实现机制：学习如何使用Redis提供的命令（如`SET`、`EXPIRE`、`GET`等）来实现一个简单的分布式锁。理解Redis作为分布式锁的优势，如高性能、低延迟以及其广泛的应用场景。

3. 实践分布式锁的加锁和解锁操作：通过实际编程练习，掌握如何在Golang中使用Redis客户端库（如`go-redis`或`redigo`）进行加锁和解锁操作。学习如何设置合理的超时时间，防止死锁，并处理锁的续期和释放逻辑。

4. 优化分布式锁的性能与可靠性：探讨如何优化分布式锁的性能和可靠性，包括但不限于减少锁冲突、提高锁获取的成功率、合理配置锁的过期时间和重试策略等。学习如何监控和调试分布式锁的运行状态，确保系统的稳定性和高效性。

5.增加实战经验：选择常见的抢红包功能来运用实验分布式锁，增加实战经验。

**二、实验仪器设备或材料**

1.计算机：运行Windows、Linux或macOS操作系统的计算机。

2.Go语言环境：已安装并配置好Go语言环境，包括GOPATH、GOROOT等环境变量的设置。

3.文本编辑器或IDE：如VS Code、GoLand等，用于编写Go代码。

4.网络连接：用于下载依赖库。

5.Redis：用于设置分布式锁。

6.Mysql数据库：用于实现分布式锁表数据。

**三、实验原理**

1、互斥访问：使用Redis的`SET`命令为共享资源设置一个唯一的锁标识（如随机字符串），确保同一时刻只有一个客户端能够持有锁。通过设置键的过期时间，防止死锁。

2、锁的获取与释放：

加锁：使用`SET resource\_name unique\_value NX PX timeout`命令尝试获取锁。如果键不存在且成功设置，则表示获取锁成功。

解锁：使用`DEL`命令删除键以释放锁。为了避免误删其他客户端的锁，通常需要先检查锁的值是否匹配，可以通过Lua脚本原子性地完成检查和删除操作。

3、过期时间与续期：

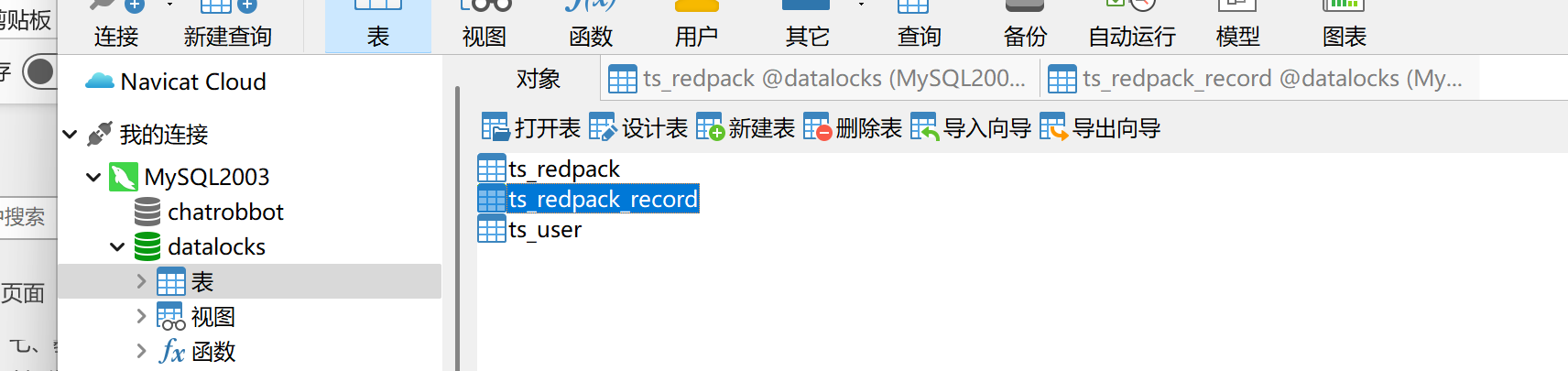
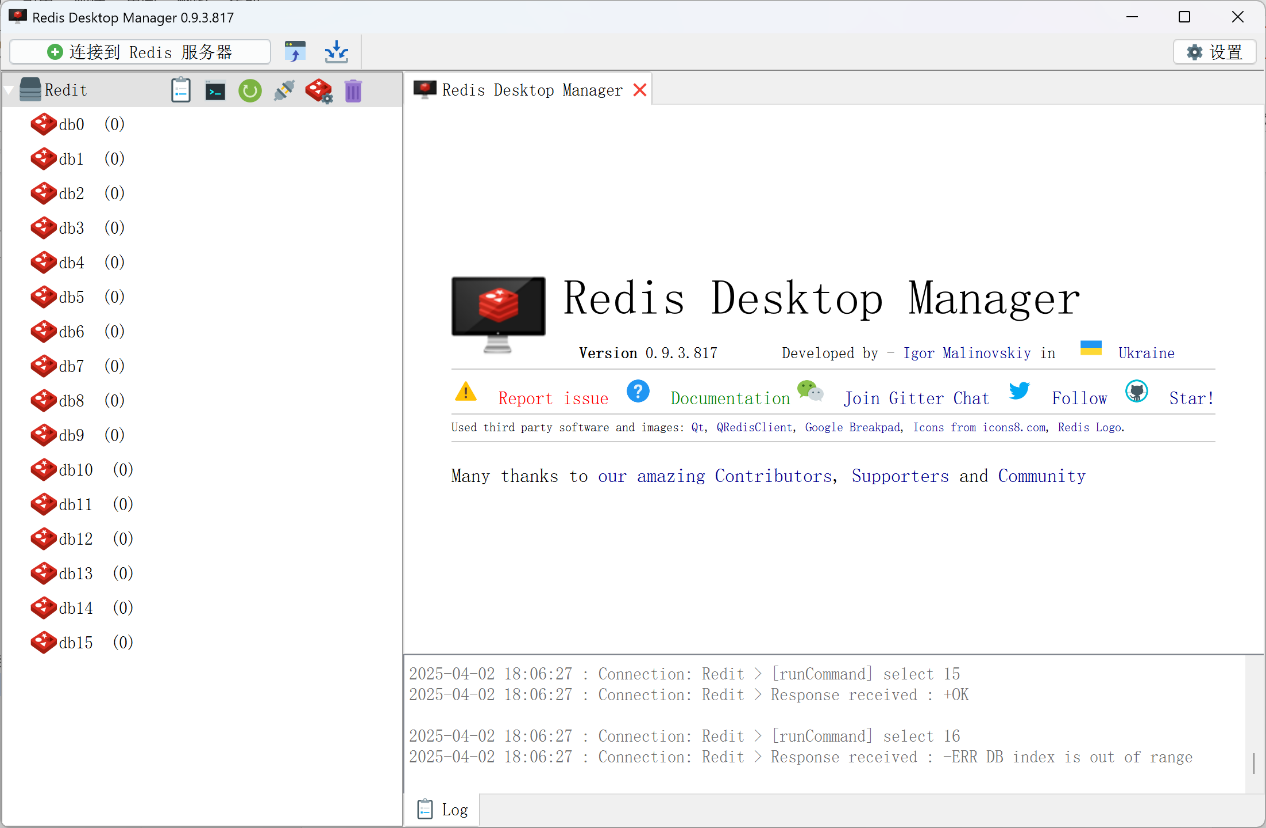
设置合理的过期时间（TTL），以确保即使持有锁的客户端崩溃或网络故障，锁也会自动释放。对于长时间运行的任务，可以定期续期锁的过期时间。

4、容错与监控：

通过合理配置锁的超时时间和重试策略，确保系统的容错能力。同时，可以通过日志记录和监控工具跟踪锁的状态，及时发现和解决问题。

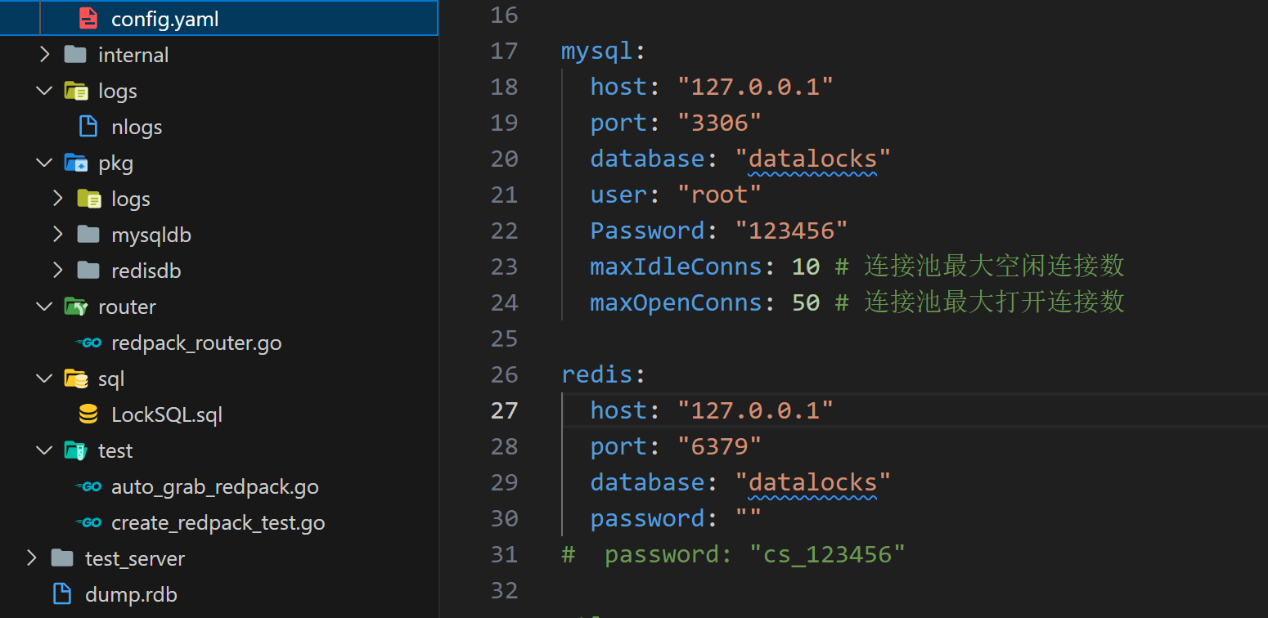
**四、实验内容与步骤**

1.确定Redis、MySQL等服务的正常启动

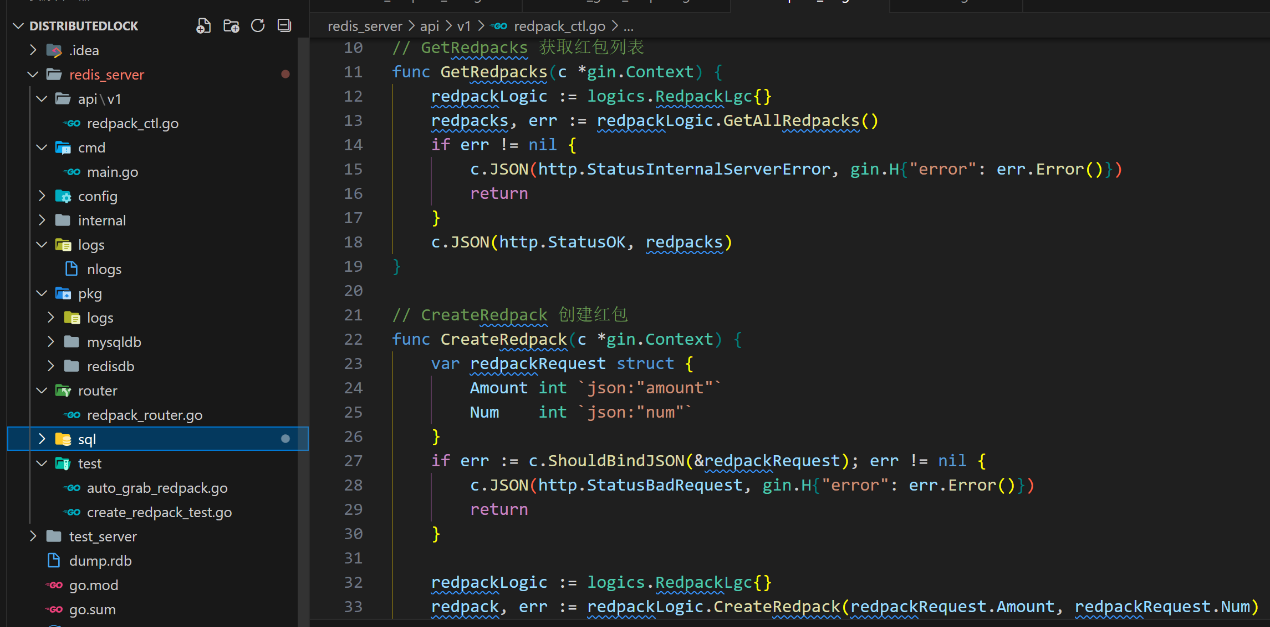


2. 移植之前实验的建立相关包，并创建相关文件

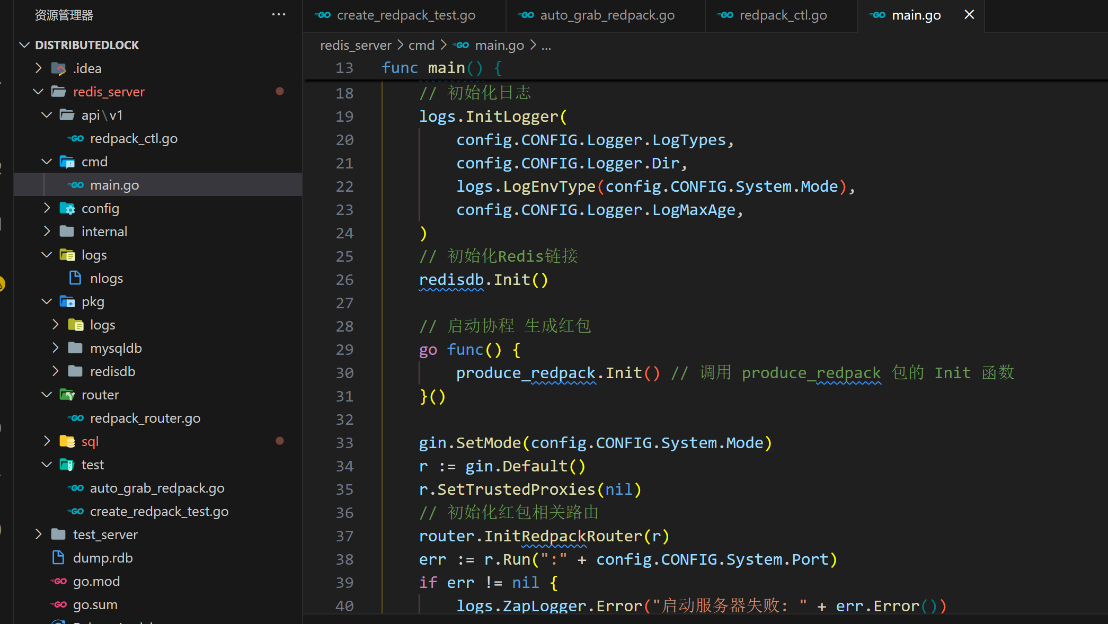
Config.yaml



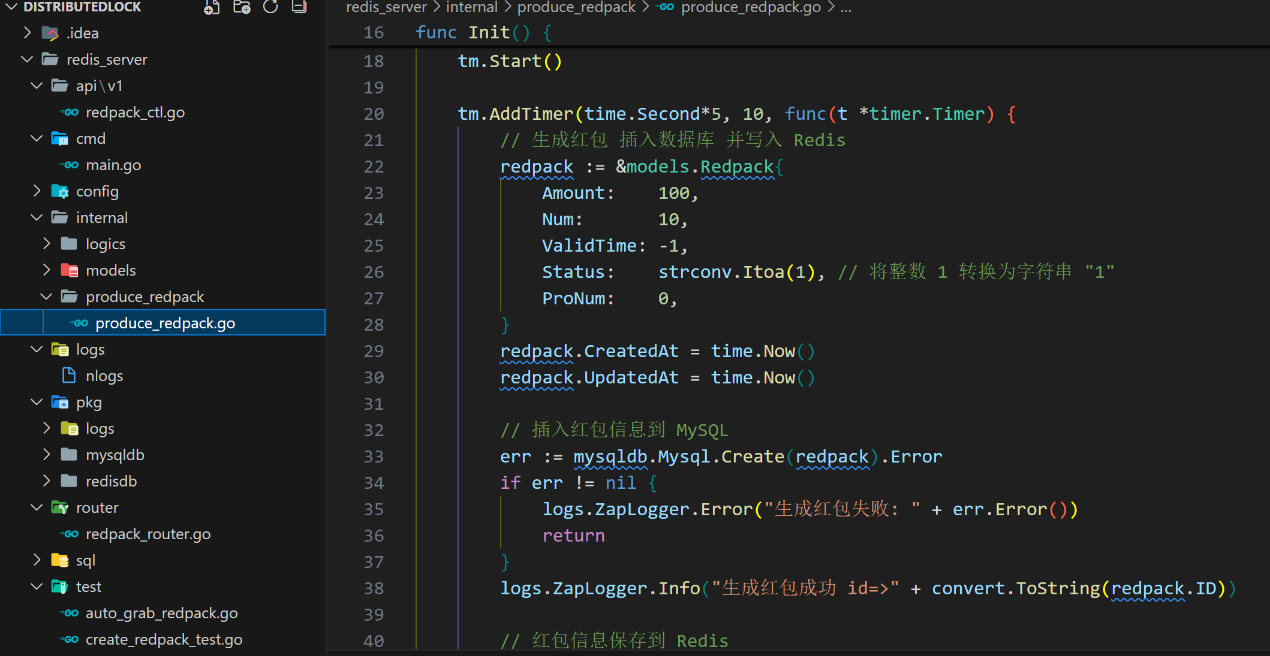
redis\_server\api\v1\redpack\_ctl.go



redis\_server\cmd\main.go

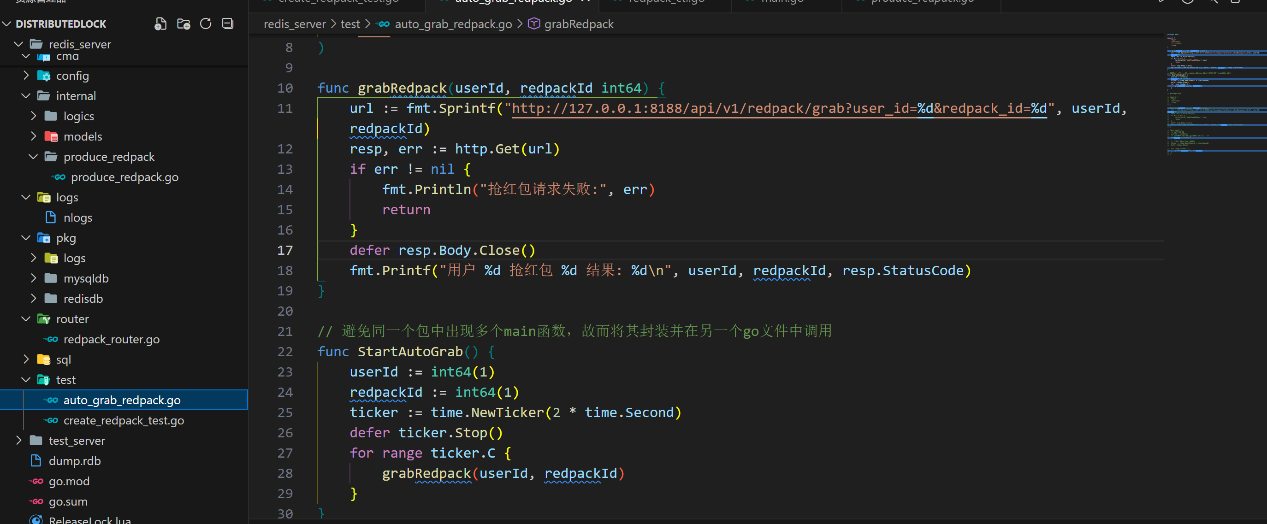


redis\_server\internal\produce\_redpack\produce\_redpack.go

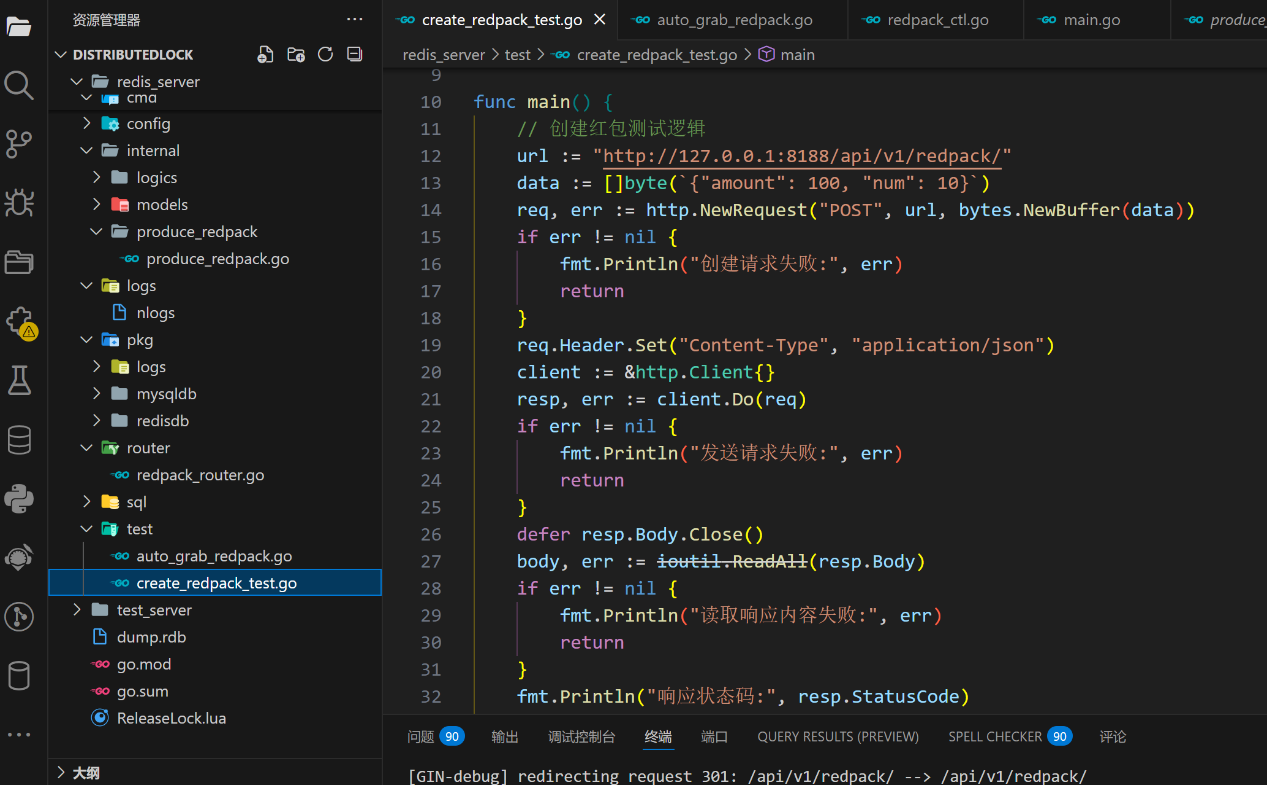


3.添加测试包test，用于创建红包和自动抢红包

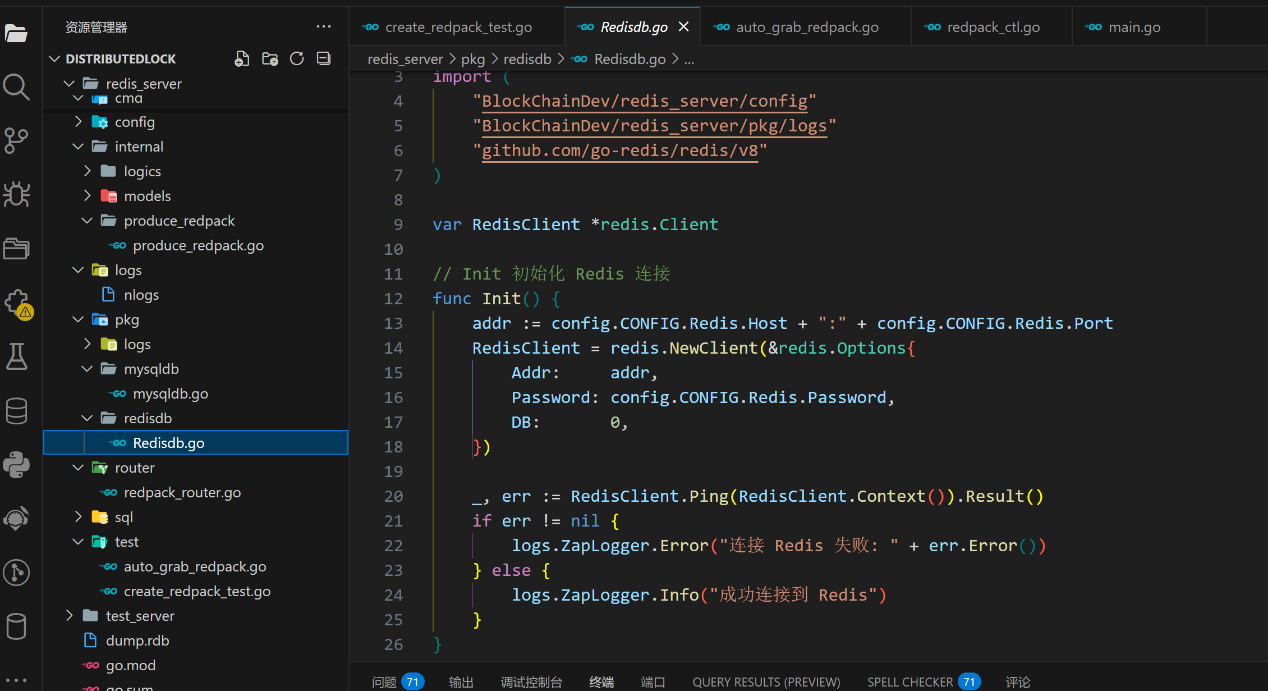
redis\_server\test\auto\_grab\_redpack.go



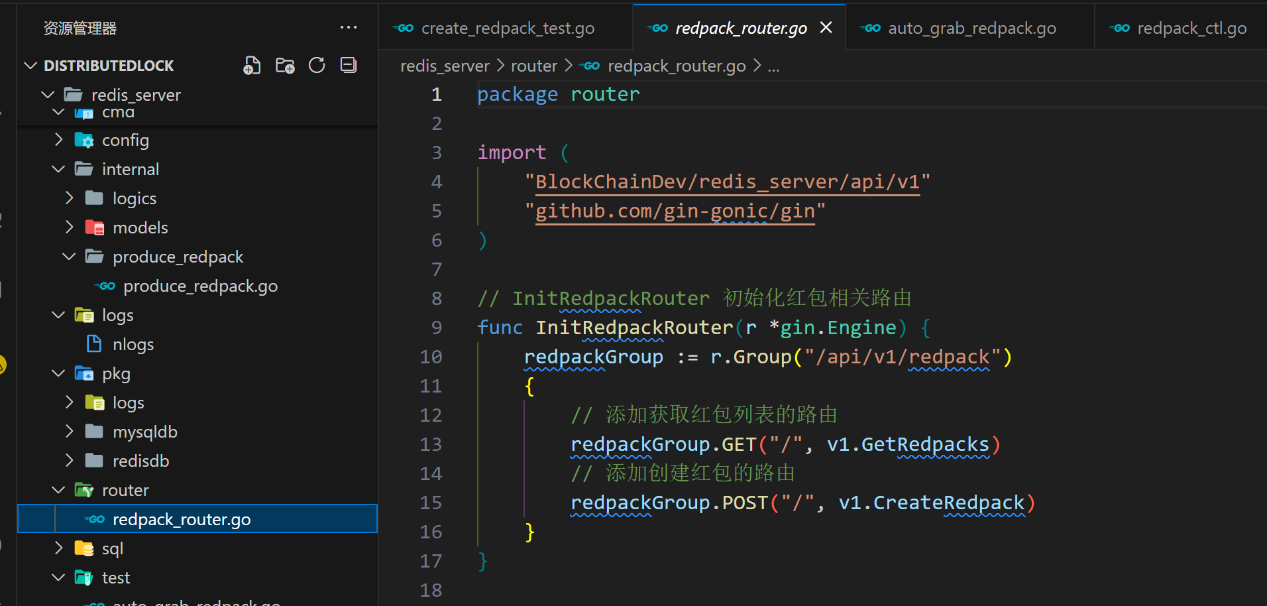
redis\_server\test\create\_redpack\_test.go



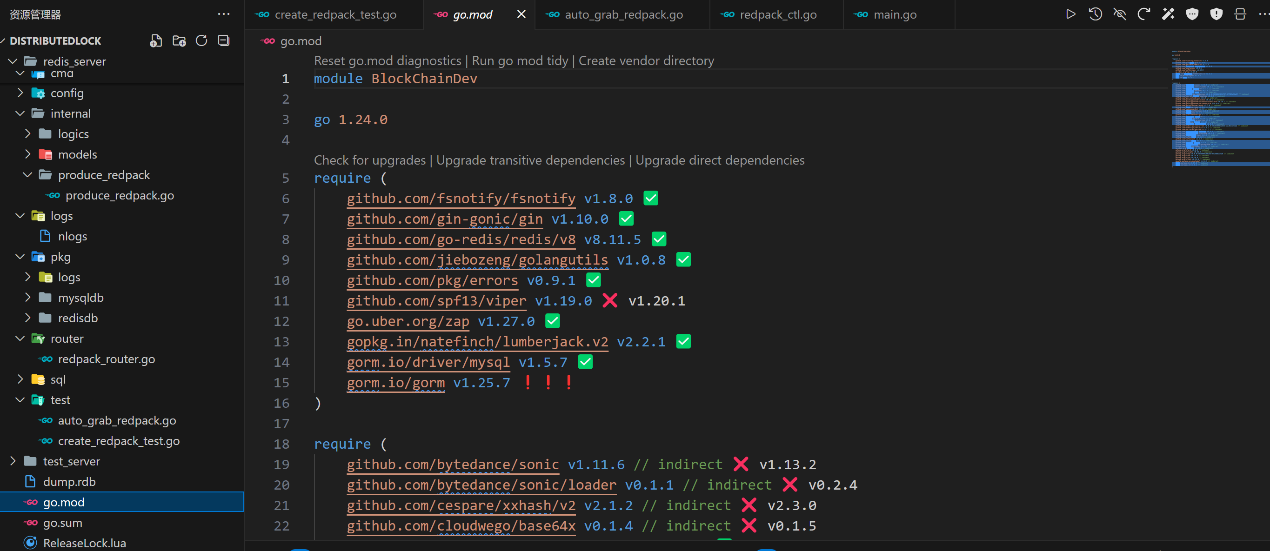
初始化Redis连接 redis\_server\pkg\redisdb\Redisdb.go



初始化路由 redis\_server\router\redpack\_router.go



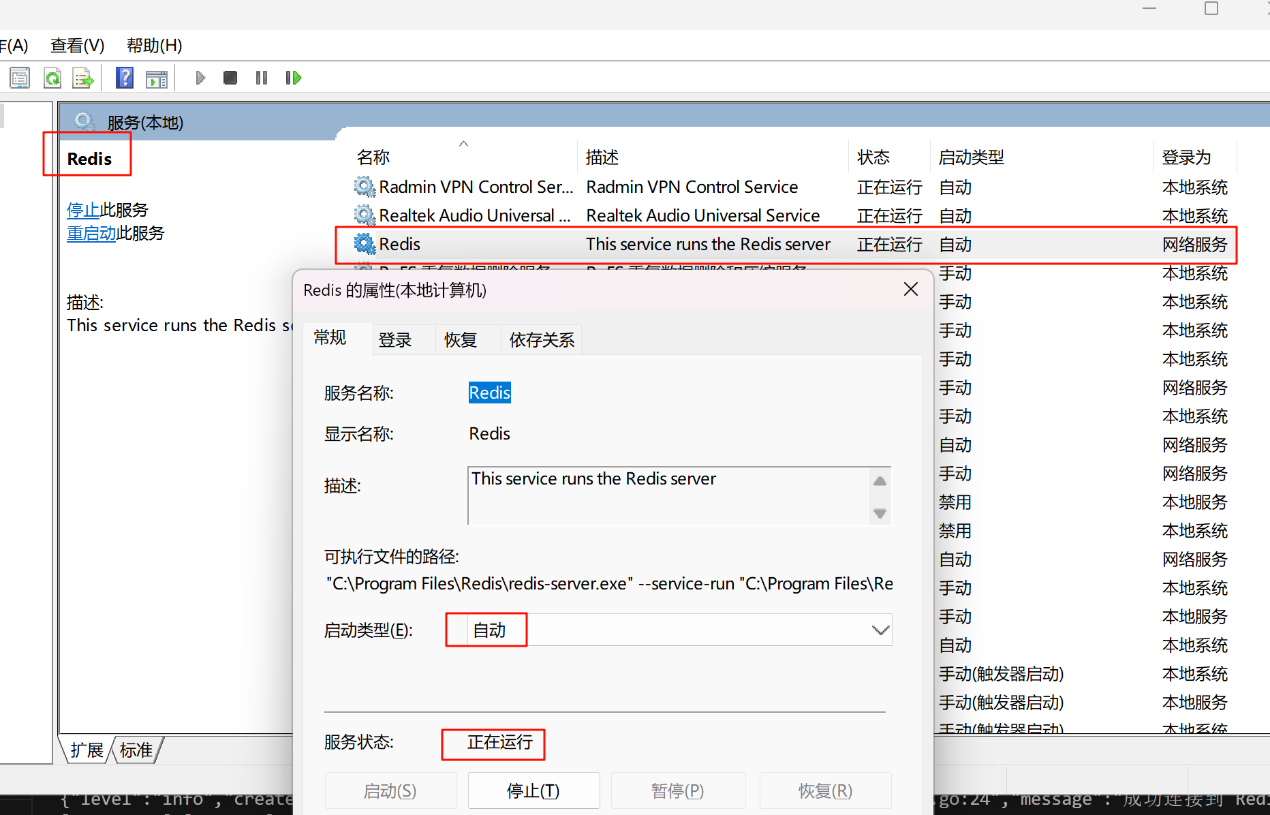
4.以防万一，go mod tidy一下获取所需要的包



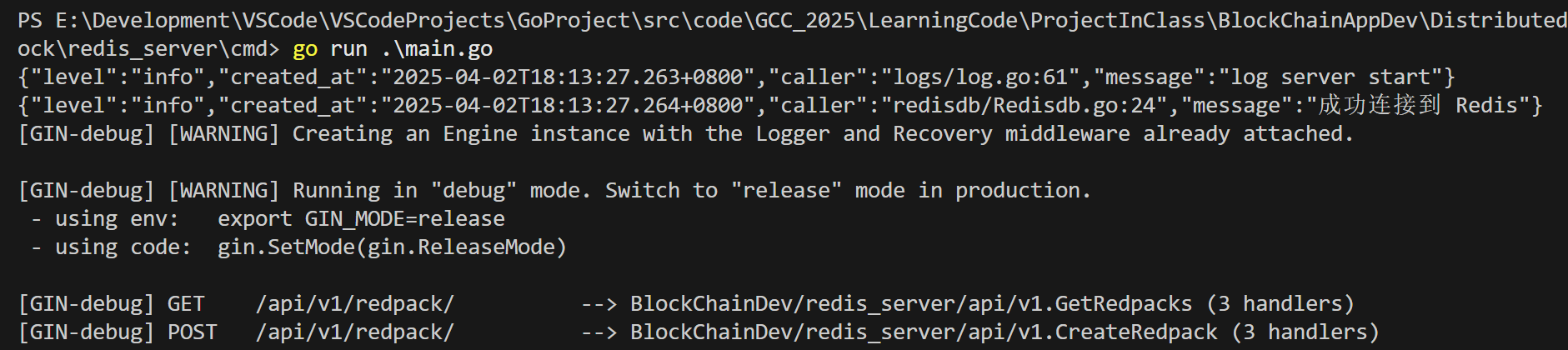
基本准备完毕

**五、实验结果与分析**

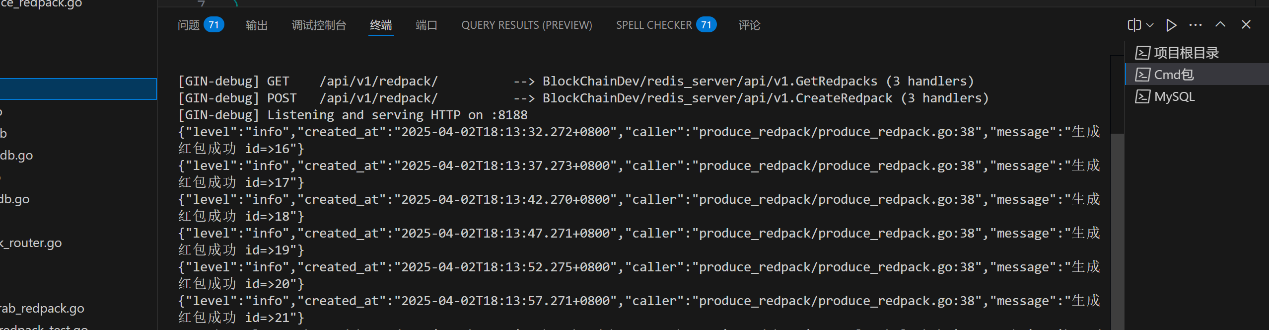
1.启动服务（Redis已设置为开机自启动，故无需redis-server）



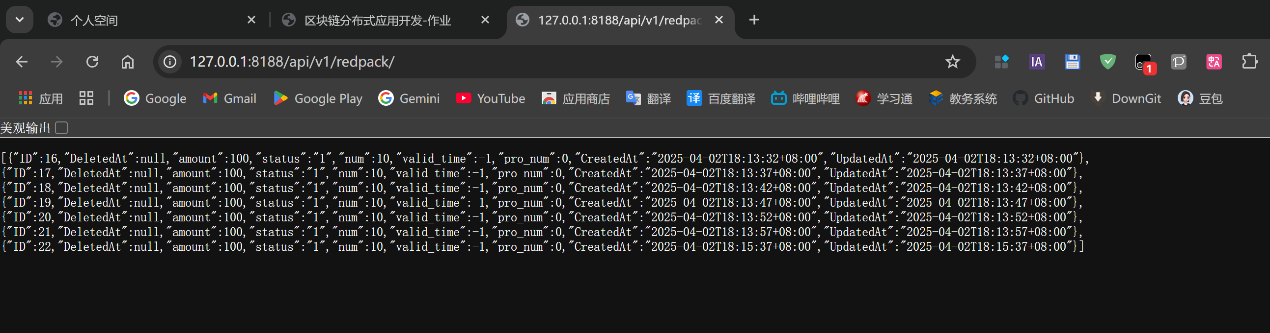
2.启动客户端（go run .\main.go）



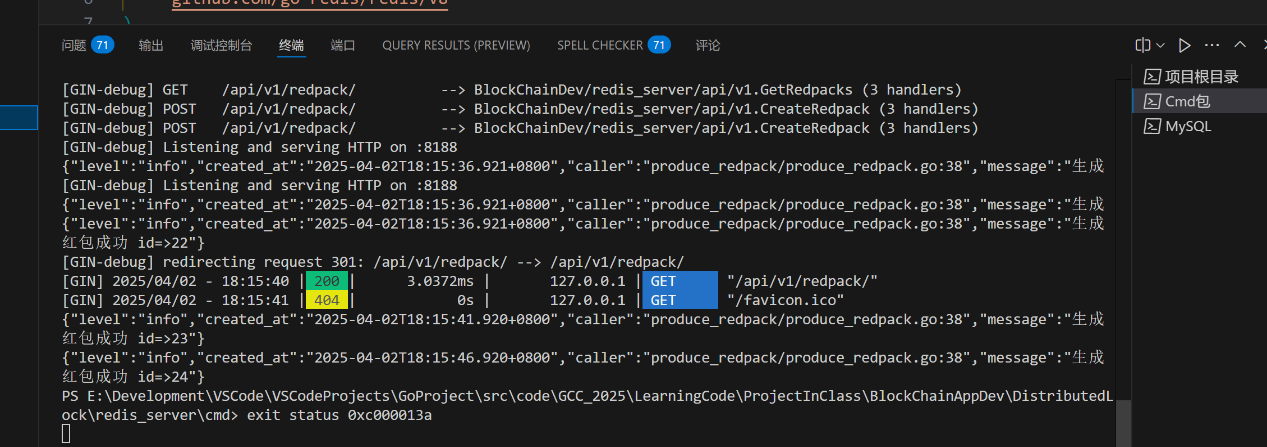
3.等待红包生成并模拟抢红包



4.返回浏览器查看POST内容（http://127.0.0.1:8188/api/v1/redpack/）



5.返回终端查看Gin调用实况（favicon.ico 404为正常现象）



实验结束

**六、结论与体会**

通过本次实验，我成功掌握了分布式锁的核心概念与Redis实现机制。在实践中，利用Redis的命令和Golang的客户端库实现了抢红包功能中的加锁与解锁操作，深刻理解了合理设置超时时间、处理锁续期和释放逻辑的重要性。

同时，我也认识到优化分布式锁性能与可靠性的关键意义，如减少锁冲突和提高锁获取成功率等。这次实验不仅提升了我的技术能力，还让我明白在分布式系统开发中，细节处理对系统稳定性和高效性的重要影响，为后续学习和项目实践积累了宝贵经验。

**七、教师评语**