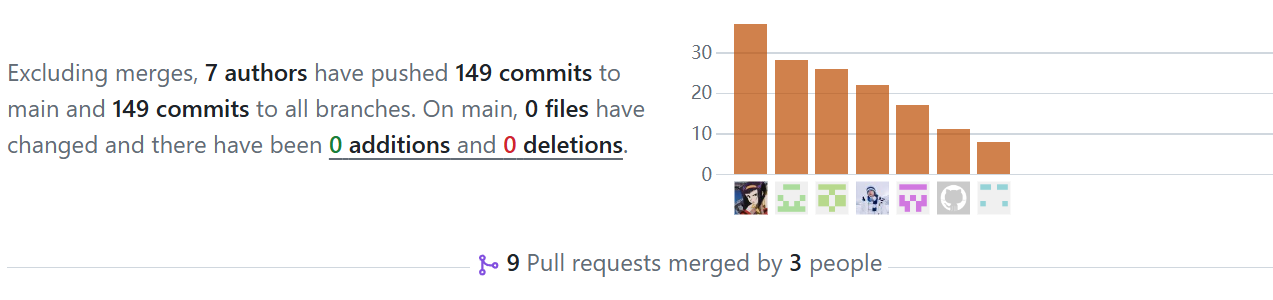
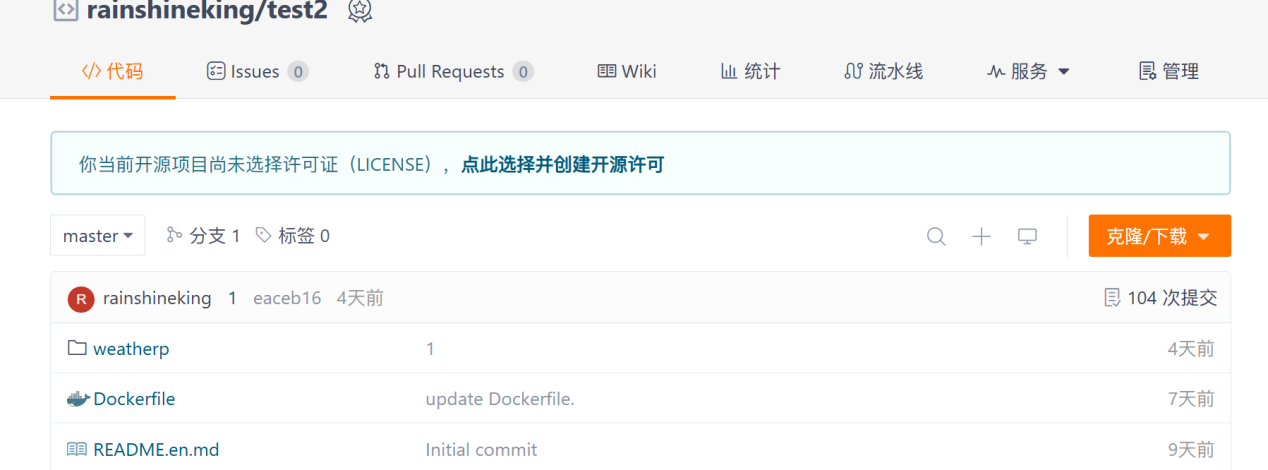
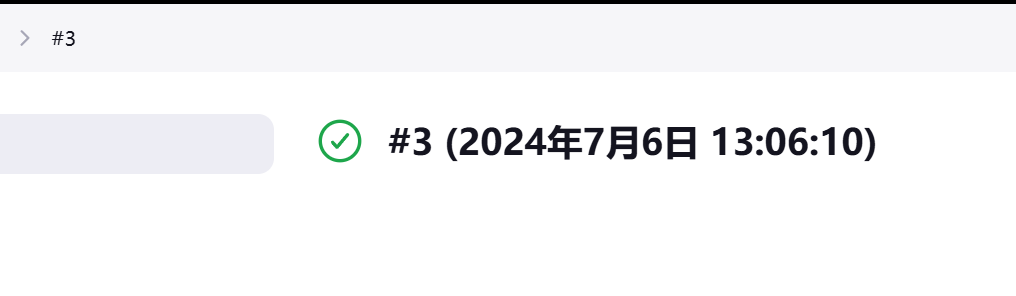
**Devops结项报告**

一．源代码仓库管理  
 我们使用GitHub进行源代码管理，在本次实训中共进行了149次提交。GitHub地址为：https://github.com/creeper-RedWHU/2024SummerCamp.git  


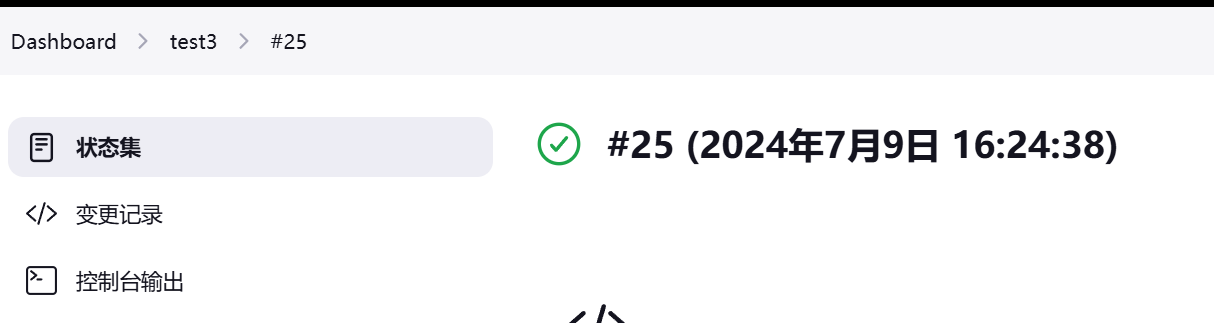
但是，这只是作为源代码管理的部分，事实上，我们组为了进行各种测试，在Jenkins上的部分测试中发现GitHub令牌问题难以解决，此外作为国外的服务器存在延迟，我们同样具有gitee源代码仓库，但只是用于进行各种代码和CI、CD的测试，如下图：（https://gitee.com/rainshineking/test2.git）  


当然这并不利于源代码库的管理，但是作为第一次进行团队的源代码管理，在其他进程上出现了各种各样的问题时不得不采用这种方法以防止拖慢进度。

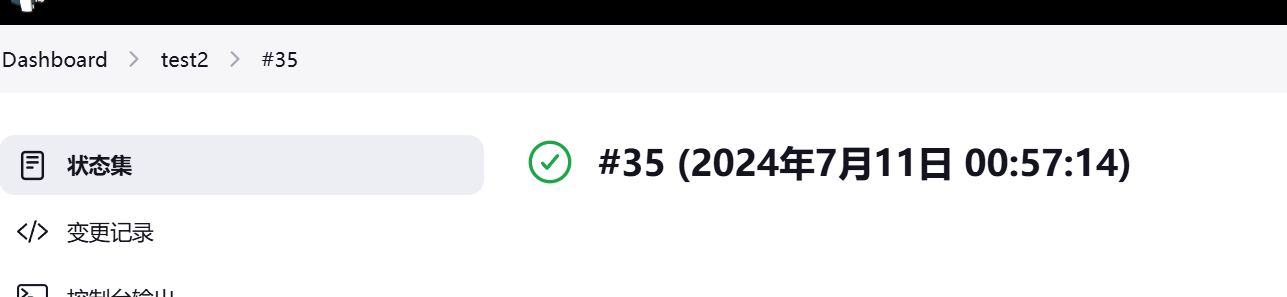
二．CI、CD流程  
 我们组是最早进行CI、CD管控的小组，在Jenkins上走的较早，并实现了较多奇思妙想。

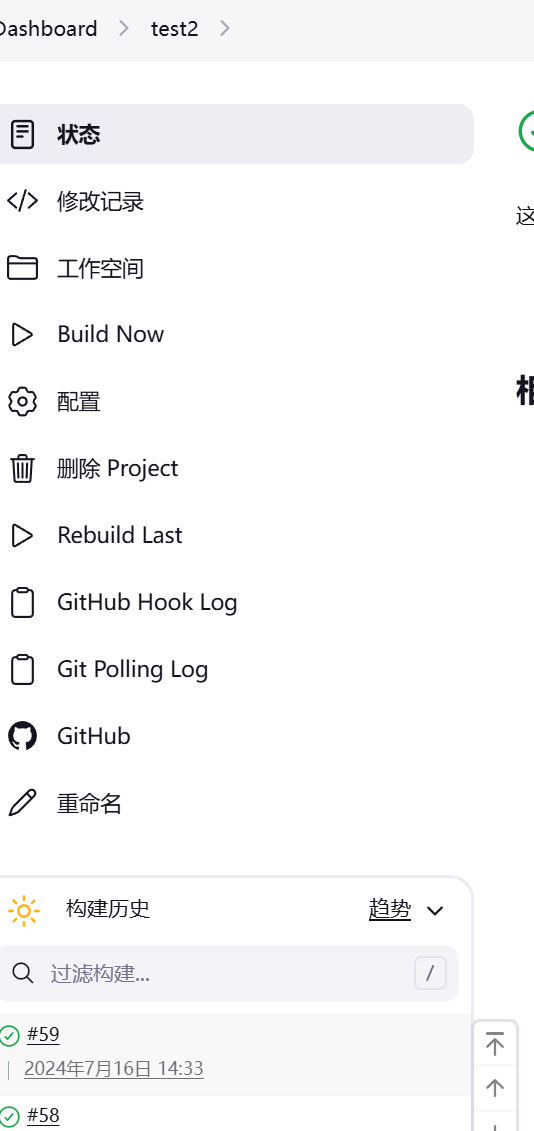
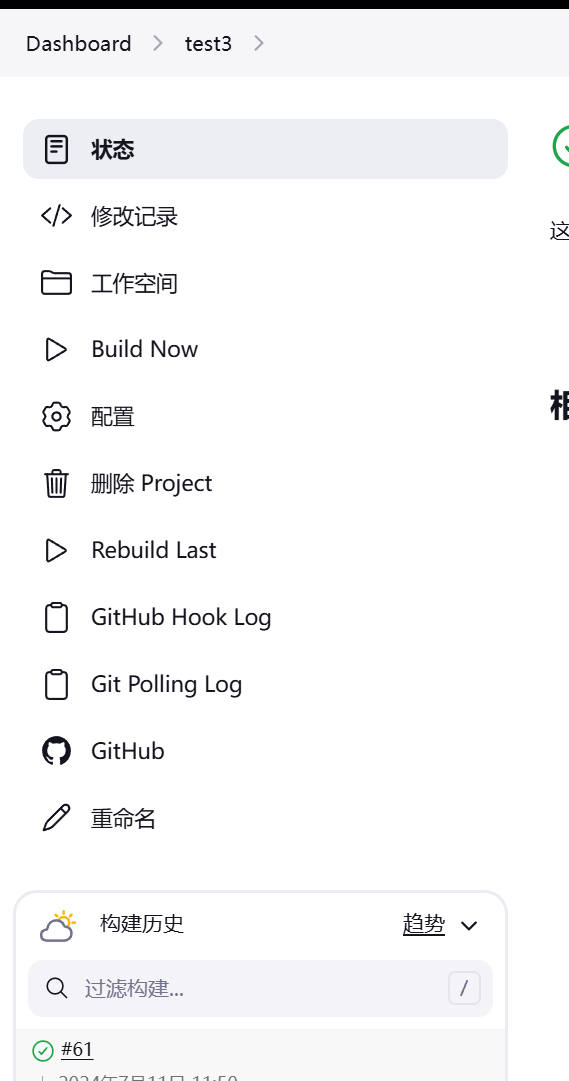
第一次实现Jenkins：（7月6日）  


第一次实现自动化编译docker：

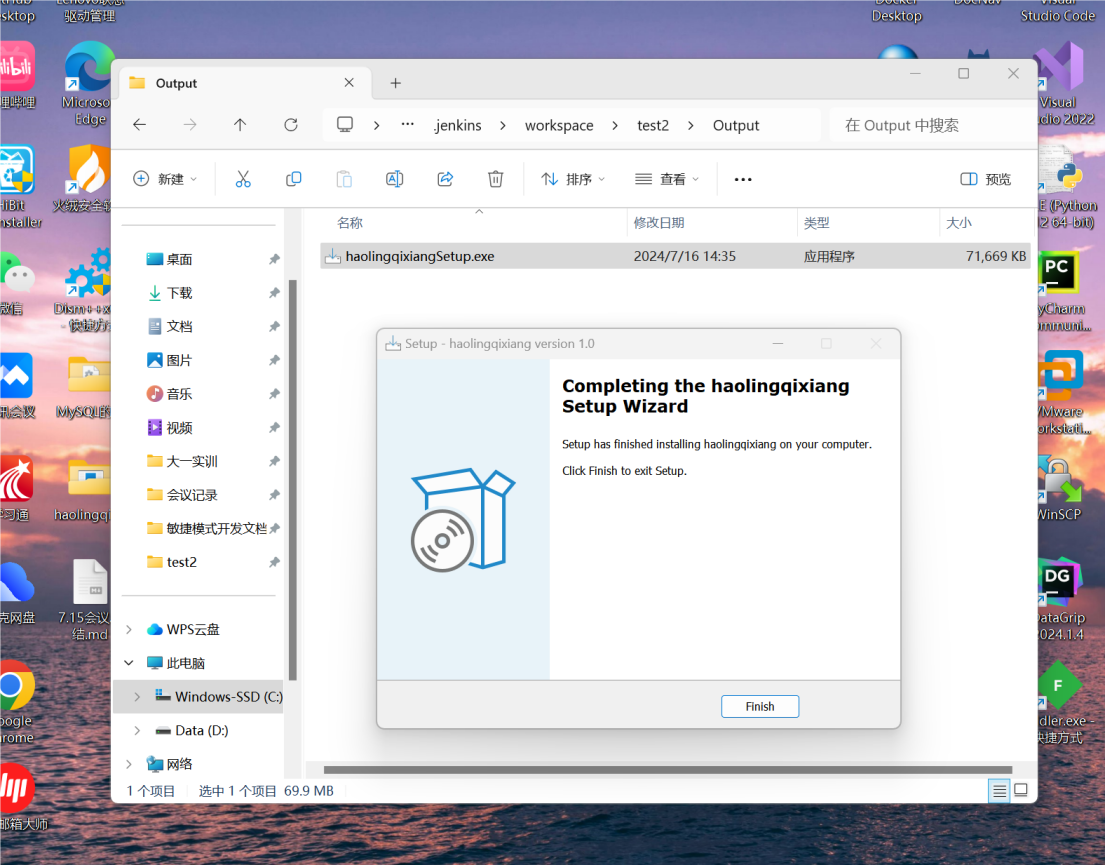


第一次实现Windows编译exe文件并打包成setup程序：



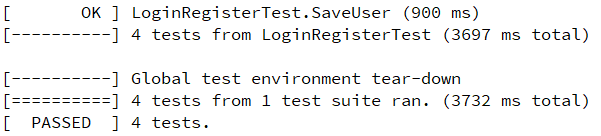
经过总计120次Jenkins流程的测试（包括测试失败与成功的总案例）。（59次） （61次）

亮点突出：时间较早，完成从源代码库到Linux版本的docker自动构建镜像并运行（test2）、从源代码库到生成Windows版本的安装程序（test3）。



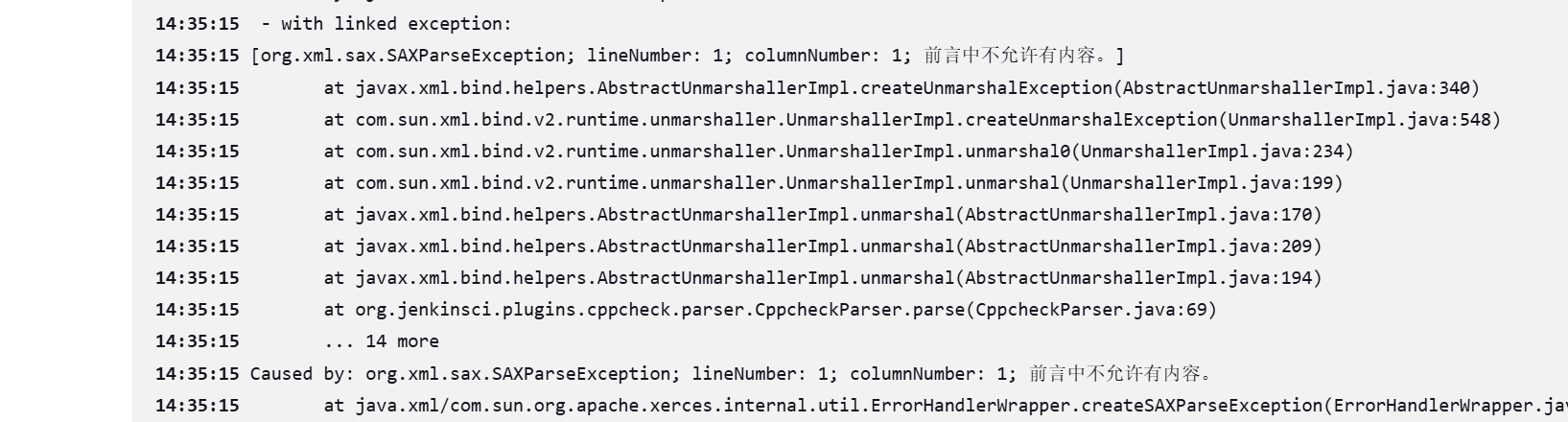
这是Windows自动化生成的安装程序效果。

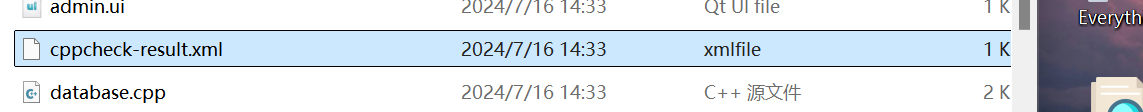
三．自动化测试

目前我们实现的单元化测试（采用google test）实现，如下为检测注册与登录功能的输出成功：  


目前覆盖率较低，仅测试注册与登录功能，即检测数据库和API的功能。（一般注册与登录成功即代表大部分功能成功）

实现源代码的检查，采用cppcheck结合Jenkins方式实现：(





暂时未实现全自动化功能测试和接口测试，皆是人工测试，输入案例进行测试输出。

四．自动化运维

由于我们产品仍未完全发布，之前虽然在GitHub上上传了beta版（测试版），但受限于云服务器的不稳定，产品的debug还有很长一段路要走。

希望我们能够实现ansible跨平台打包发布，使用普罗米修斯（或夜莺）等监控工具实现对数据库的监控，升级服务注册等功能，实现软件更加稳定运行；同时实时收取用户的反馈进行产品的维护和升级。