USTB

Documentation to the Verilog Online Judger

Digital Logic & Computer Architecture

Guidance

Full Name	Student ID
Yuchen Wang	U202141XXX

Xueyuan Rd., September 21, 2023

Contents

1 | Introduction

To fill up the blank of verilog online judger, which could boost the verifying of open-problem in teaching and in exam, we designed this verilog judger based on `iverilog`, `python` and `bash`. It can be simply run by using a docker container in CG platform, or somewhere else, and generate a `json` output via standard output for further usage.

It could automatically detect the number of test points, and offers the reference and wrong answer of the first test point where error occurs.

2 | Usage

2.1 | OJ Structure

The structure of the core which belongs to the Verilog OJ is shown in Figure 2.1.

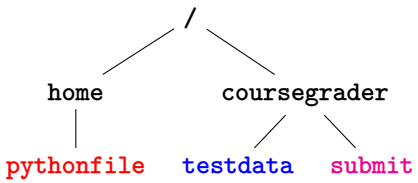

Figure 2.1: File structure

Where `pythonfile` contains python and shell scripts, it is the core of the judger; the represented structure of `testdata` is shown in Figure 2.2, `point`i, which represents the sub-testbench, is optional[1], and how many sub-testbenches depends on the number of test points n in the `testdata`; student's answer stores in `submit`, where should contain `*.v` and NOT contain `*_tb.v`.

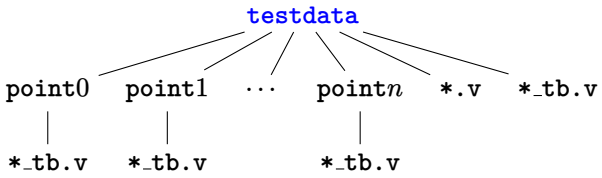

Figure 2.2: File structure of `testdata`

2.2 | Testbench

Testbench should offer some output through the console, otherwise it can not judge whether the answer is right or wrong. Sentences such as `$display`, `$monitor` or `$write` can be used.

[1] If it doesn't pass the main testbench, sub-testbench will work. Otherwise, those sub-testbench will be bypassed.

3 | References

A | Upload docker container to CG platform

CG 平台 Docker 使用流程

Vanity Fair[a]

[a] *University of Science and Technology Beijing, 30 XUEYUAN ROAD, HAIDIAN DISTRICT, Beijing, 100083, Beijing, China*

Abstract

本文介绍了如何在 CG 平台上使用、部署 Docker，并给出了目前 Verilog、Java 测评机的使用文档。

Keywords: CG, Docker

目录

1. CG 平台挂载 Docker

由于尝试使用 Dockerfile 在线构建镜像失败（无响应），因此本文采用自行开发 Docker 镜像的方式部署 Docker 测评机。

首先需要以**系统管理员**身份进入系统管理 → 实验环境管理 → 桌面镜像管理或 Jupyter 镜像。

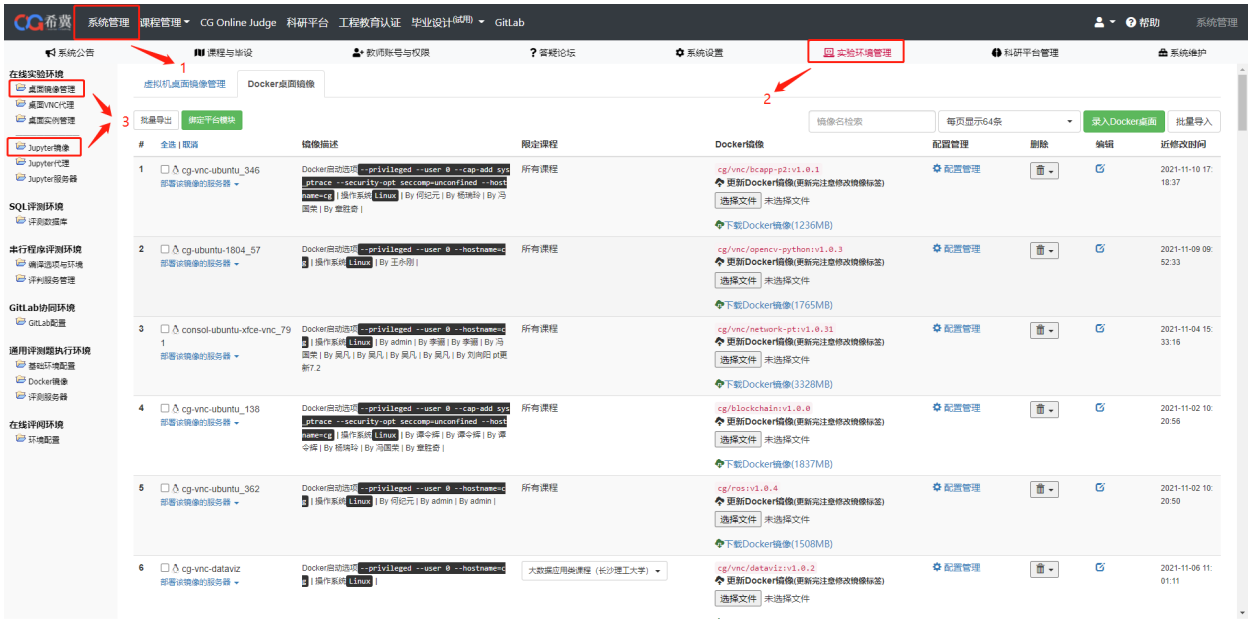

图 1: 进入镜像管理界面

之后点击批量导入，导入镜像文件

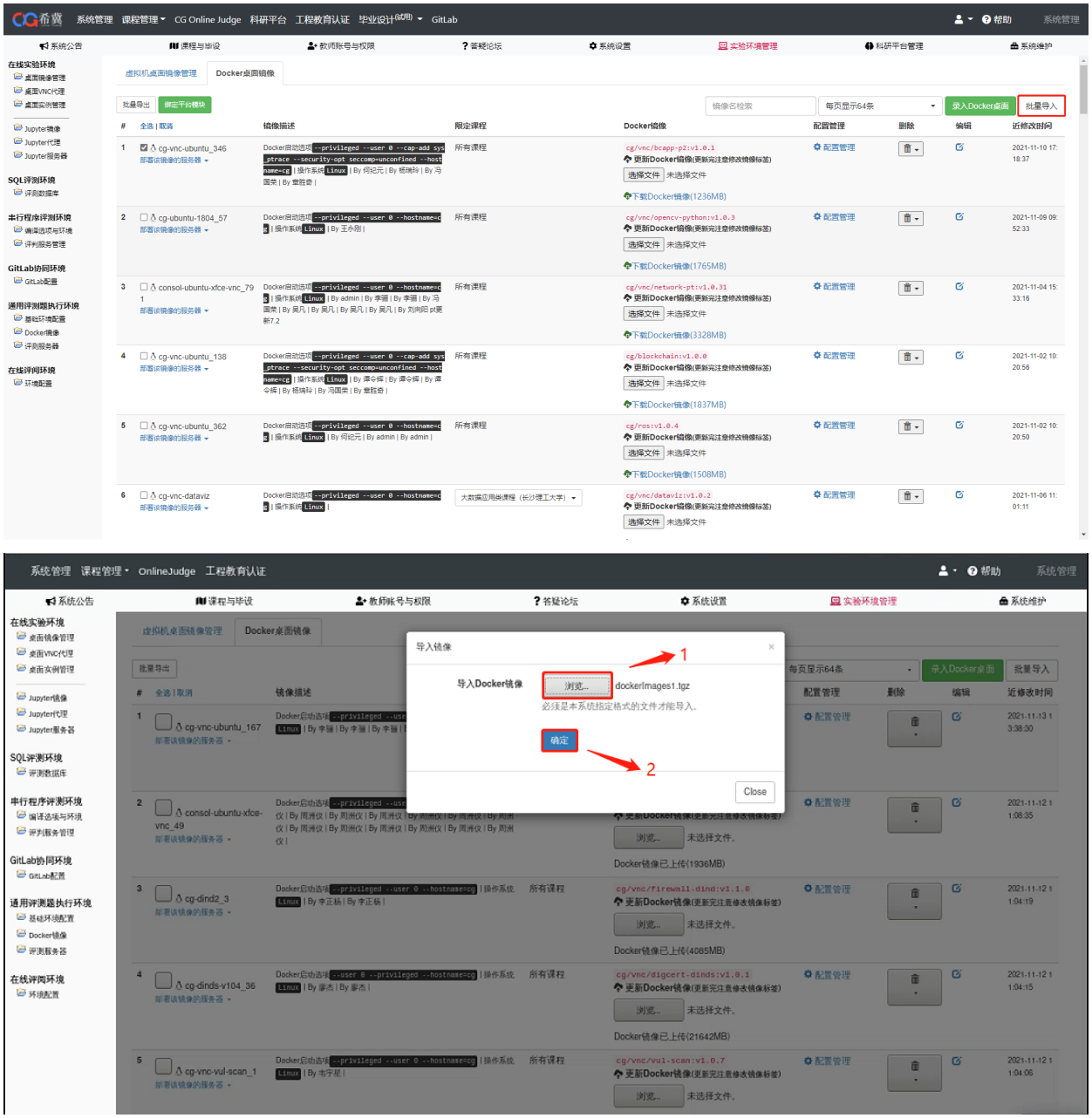

图 2: 导入镜像文件

点击浏览，导入 Docker 镜像

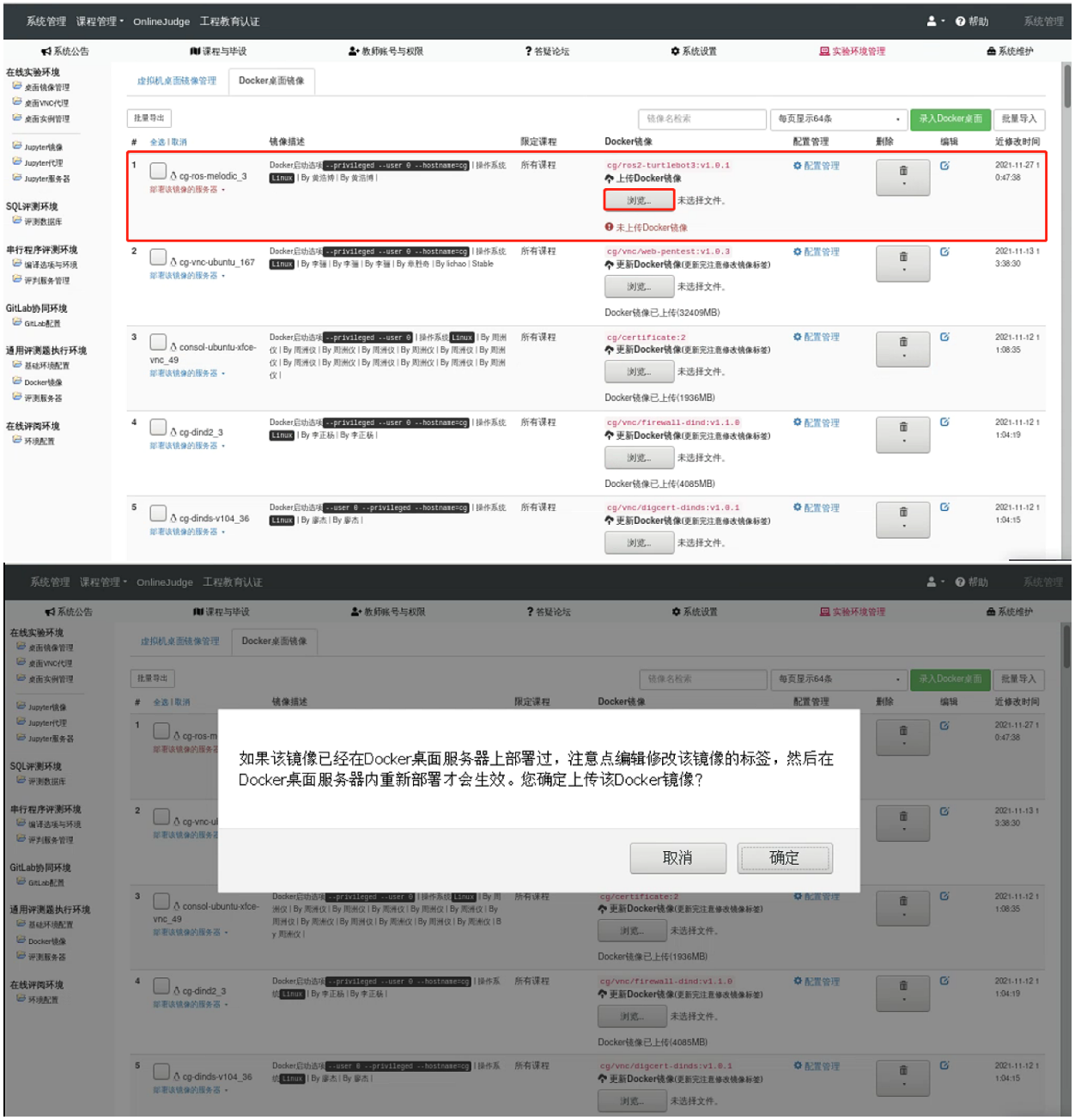

图 3: 导入 Docker 镜像

等待镜像上传完毕（**镜像上传过程中此页面不要进行其他操作，不可同时上传多个镜像**）

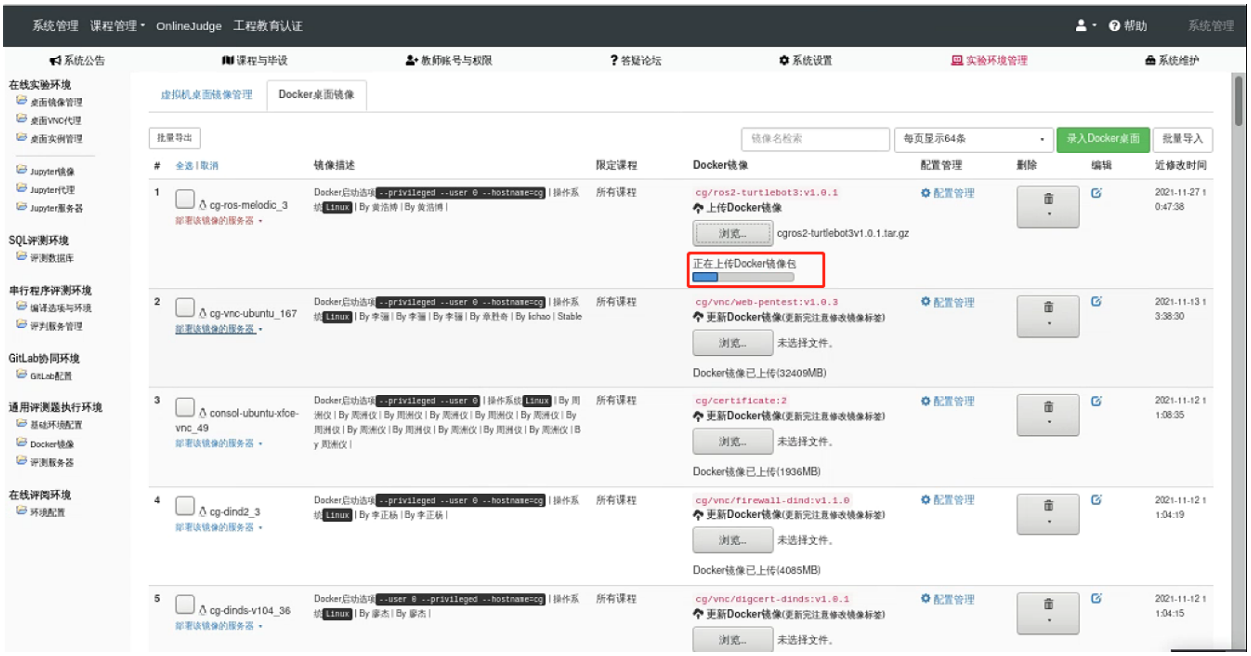

图 4: 等待镜像上传完毕

点击桌面实例管理或 Jupyter 服务器，找到部署镜像的服务器点击编辑

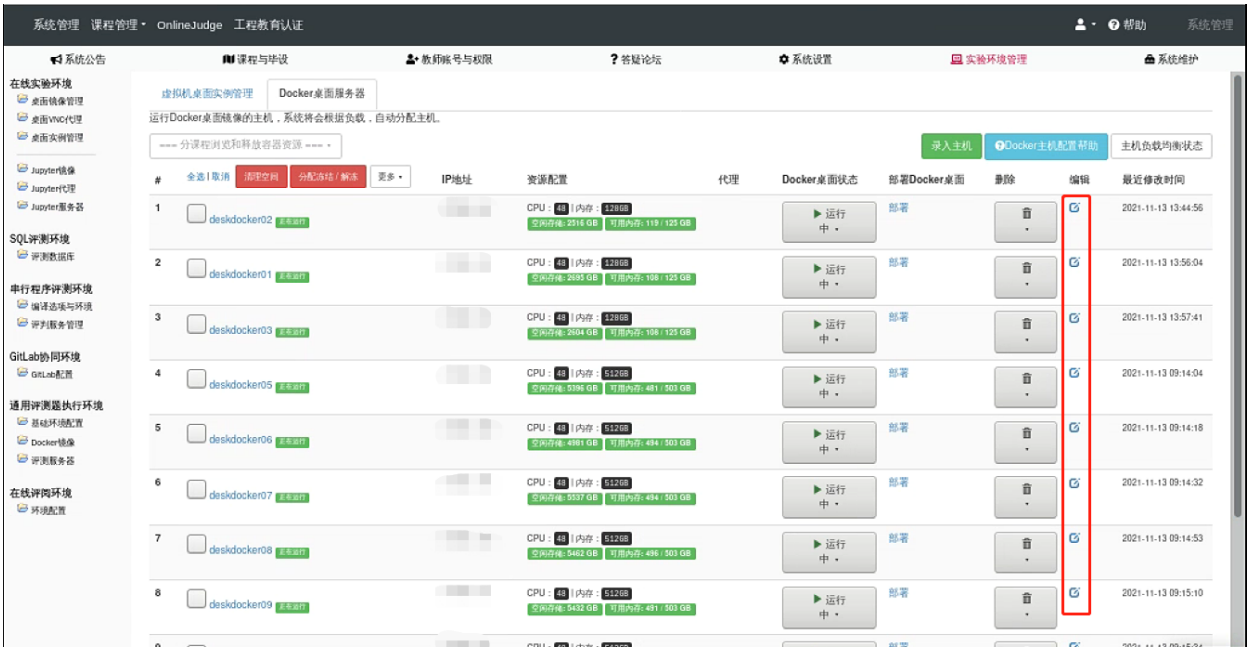

图 5: 编辑服务器

将需要部署的镜像加入到“需部署的镜像”中（单击左键勾选）

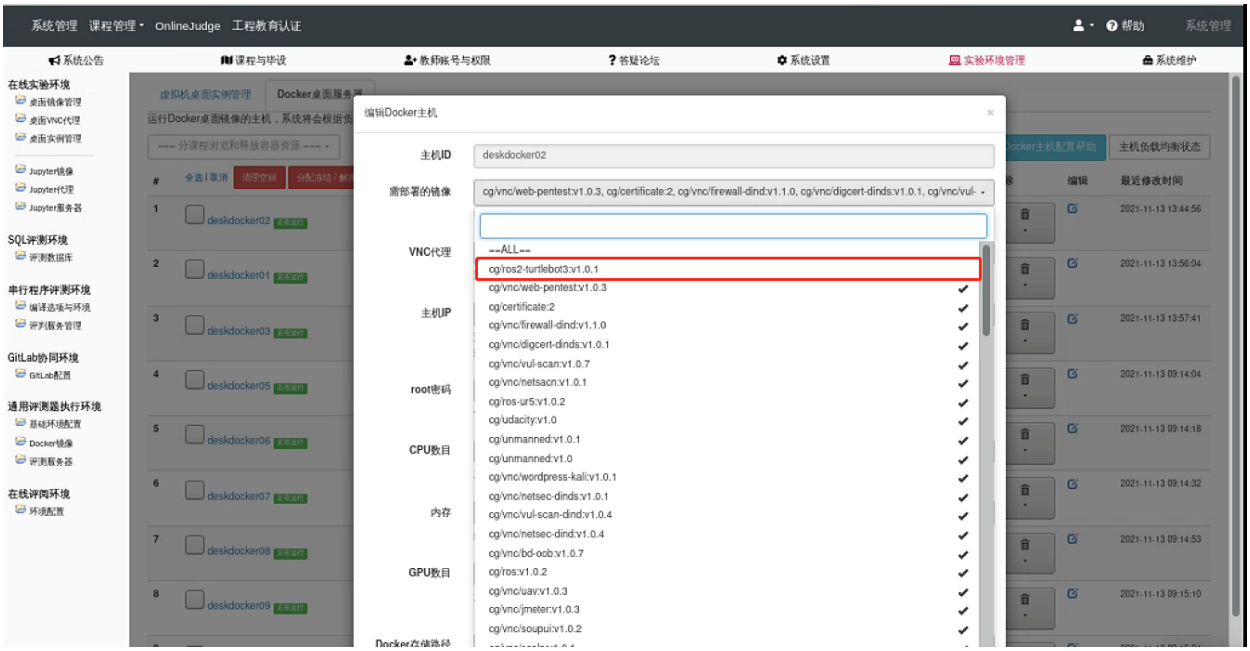

图 6: 加入镜像

点击保存

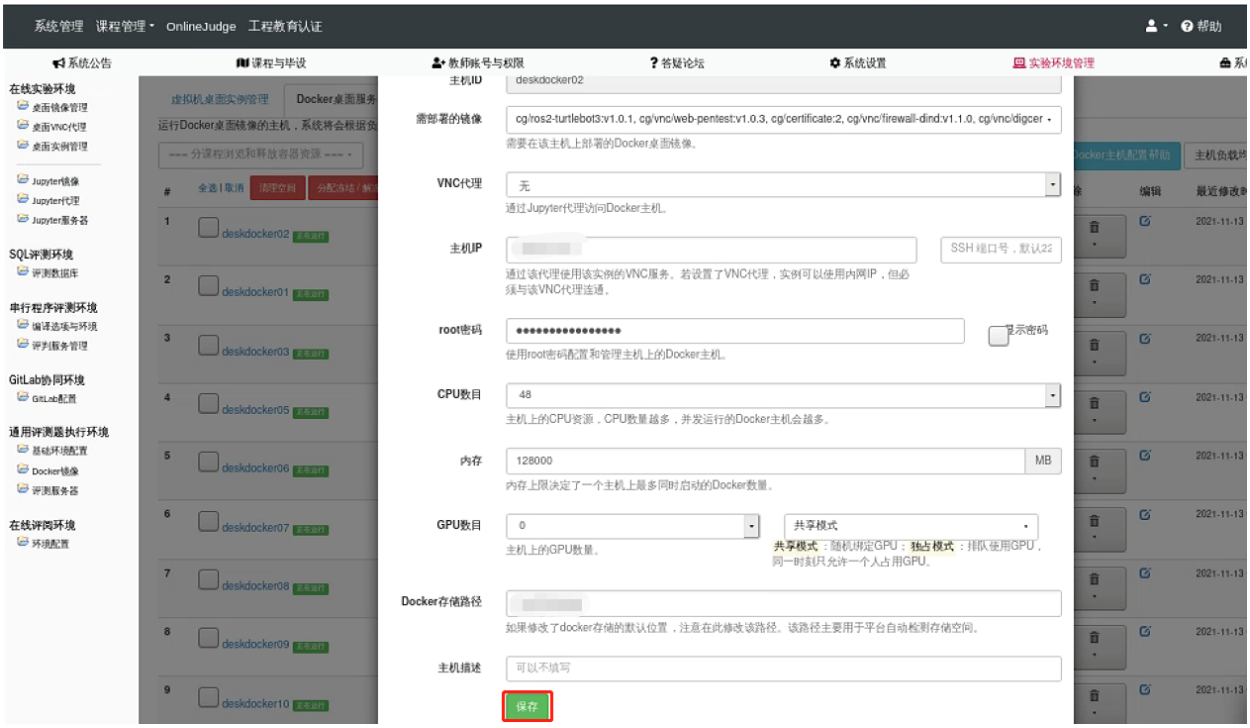

图 7: 保存镜像

服务器 Docker 桌面状态将改为黄色，并提示镜像部署状态异常，点击部署

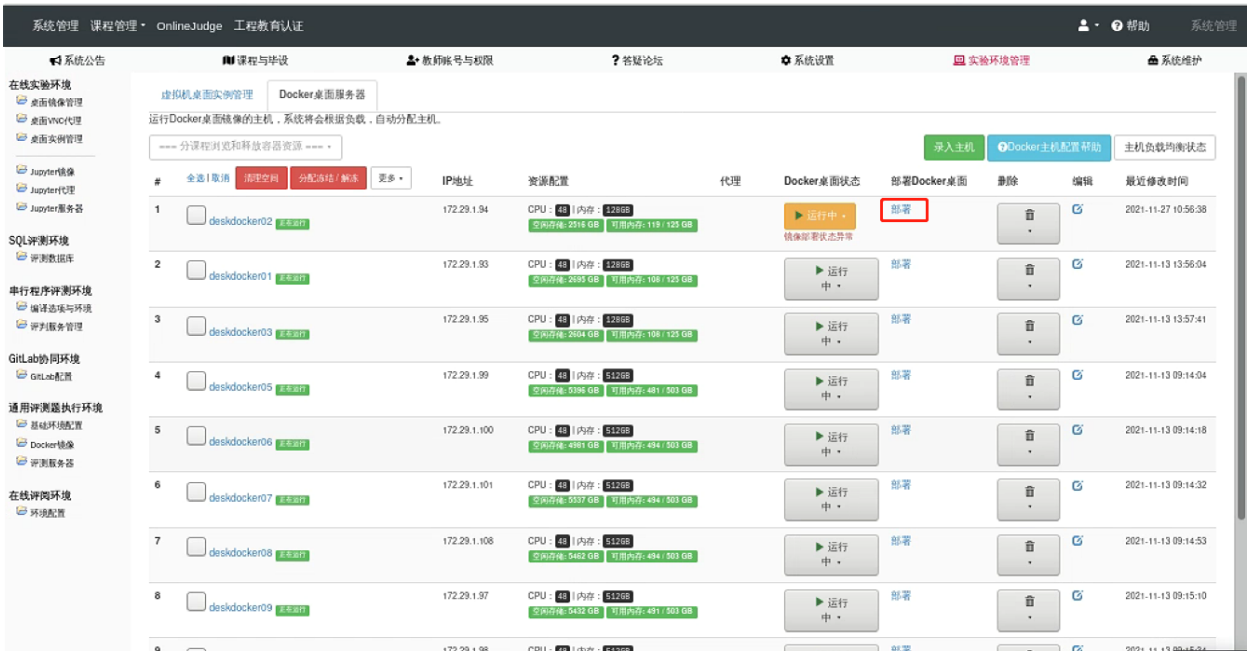

图 8: 部署状态异常

点击重新部署，将会拉取镜像（拉取镜像过程中页面可进行其他操作）

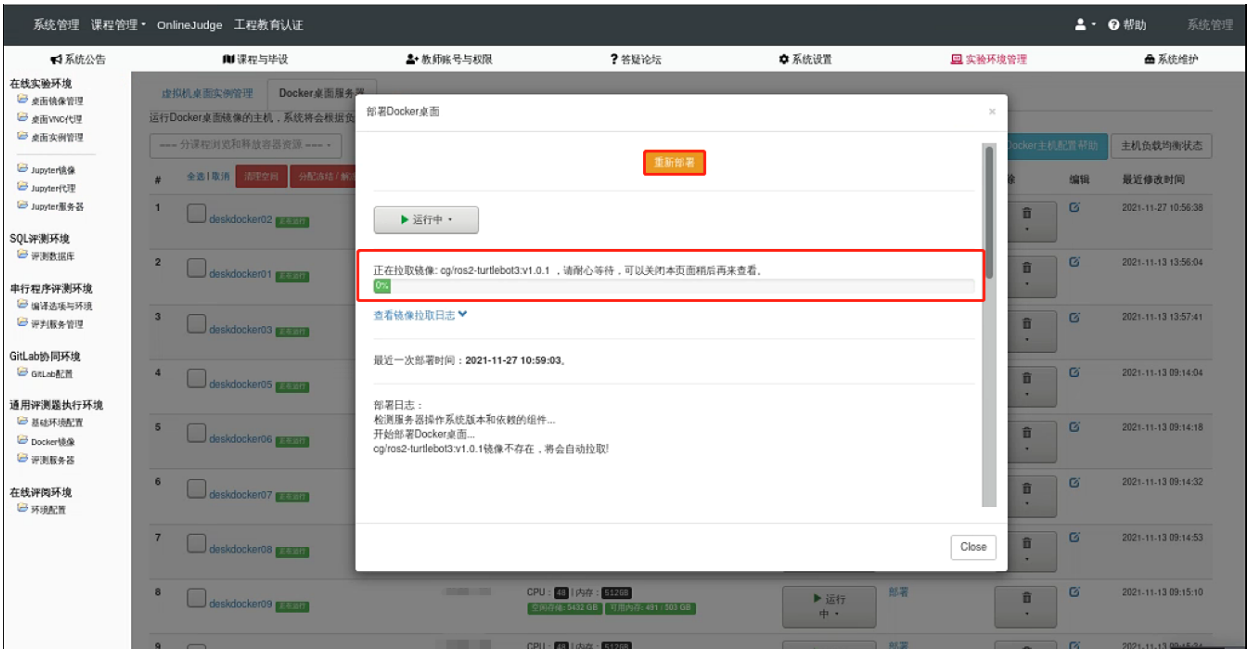

图 9: 重新部署

再次点击重新部署

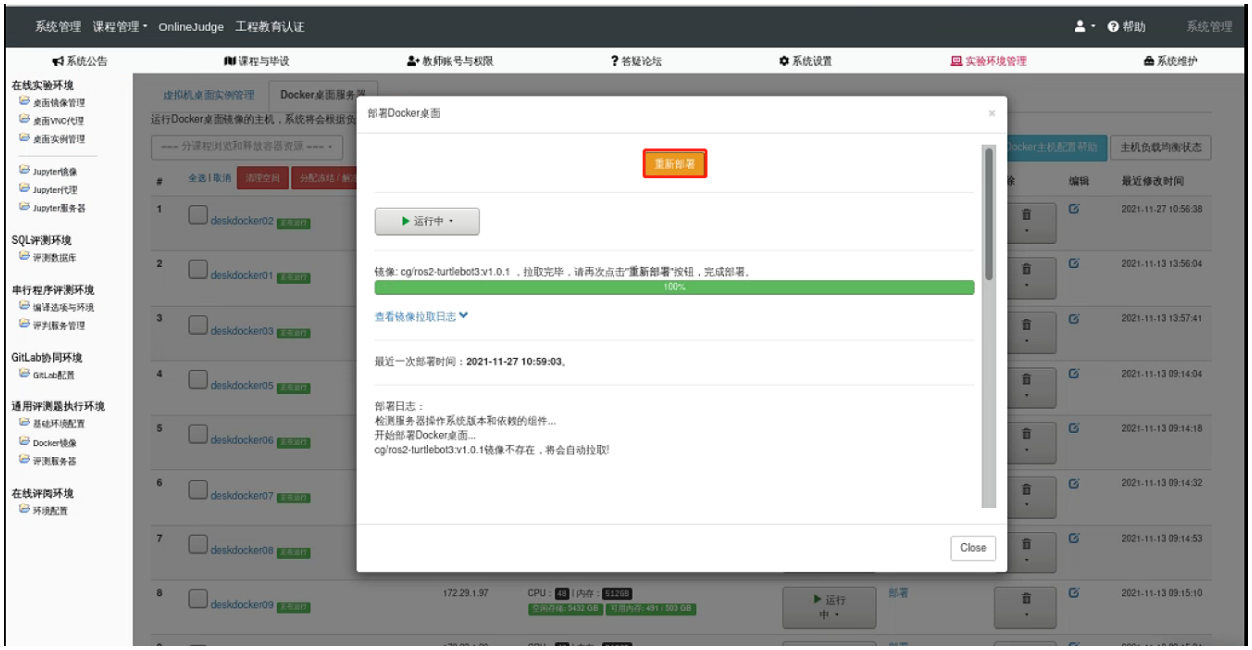

图 10: 再次重新部署

出现此界面即镜像部署完成

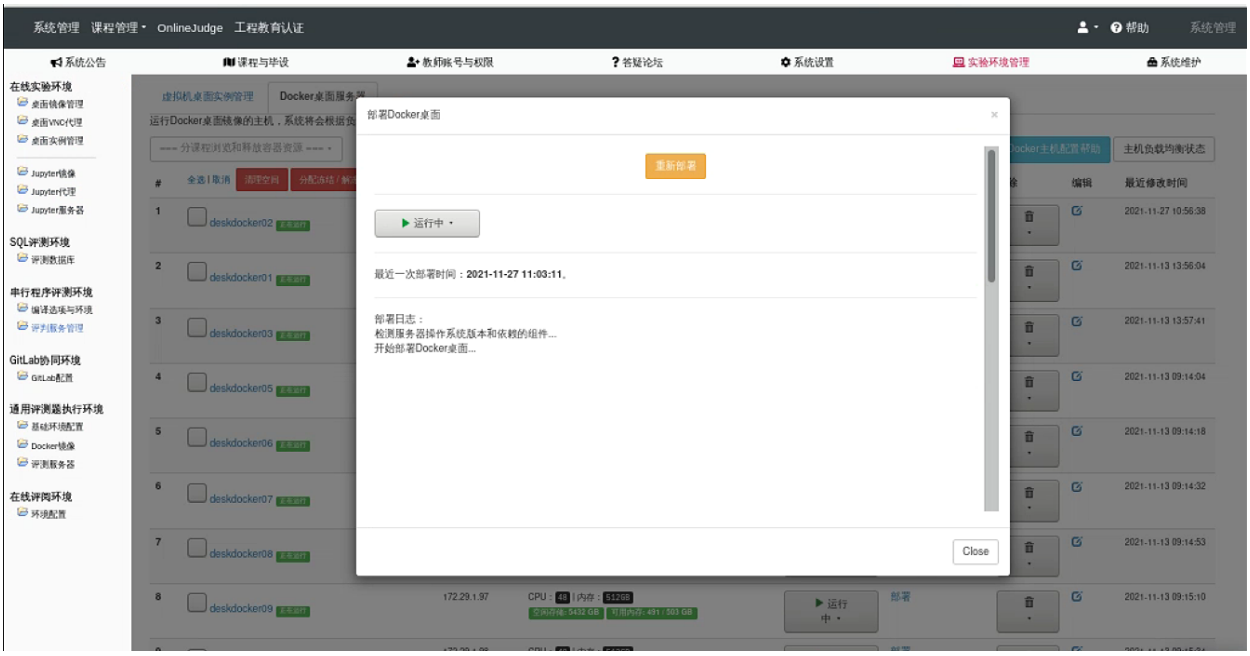

图 11: 部署完成

2. 添加通用评测题

进入作业栏目，单击录入题目中的添加通用评测题，并在运行环境中选择相应评测机。

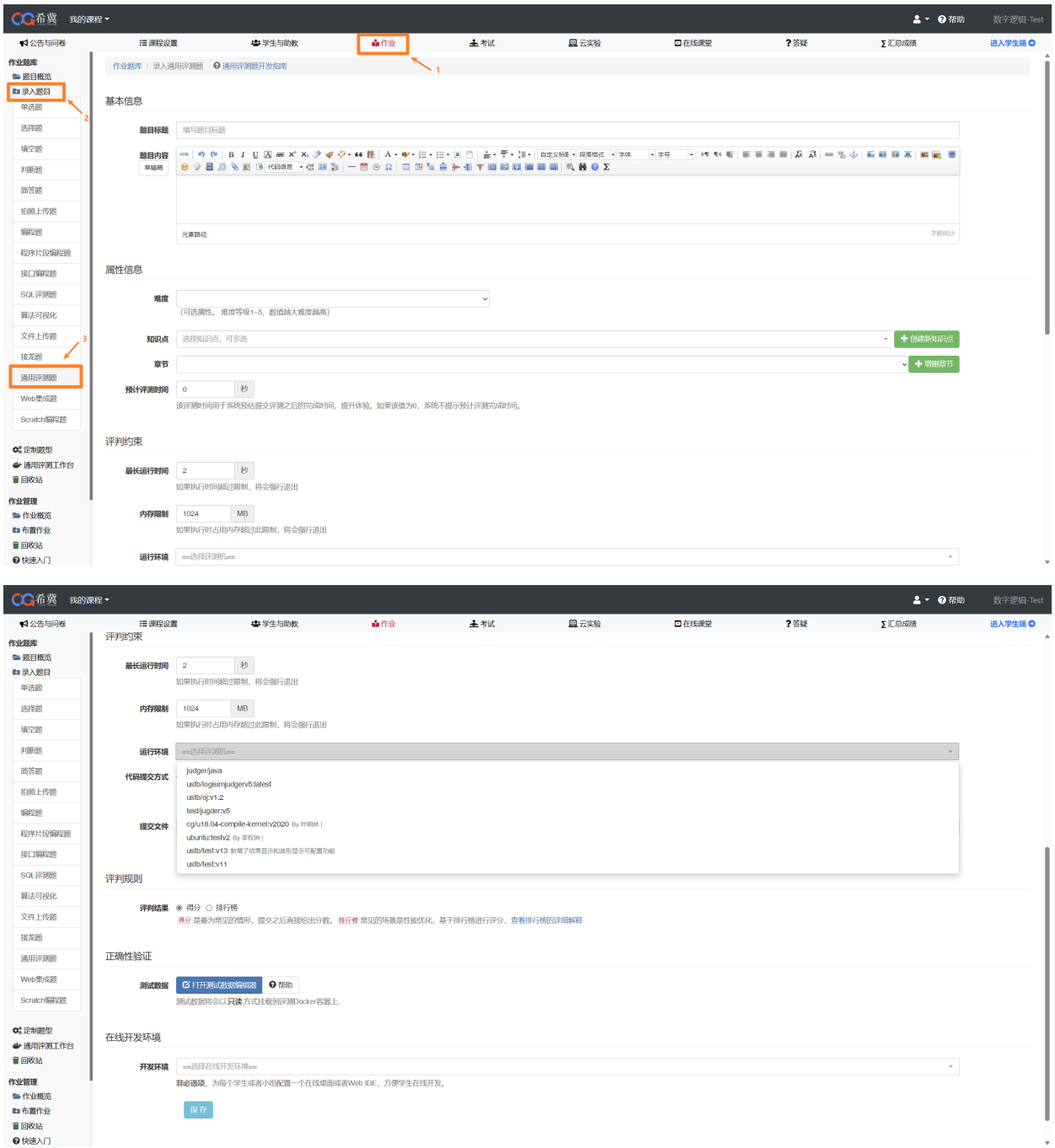

图 12: 添加通用评测题

在正确性验证中打开测试数据编辑器

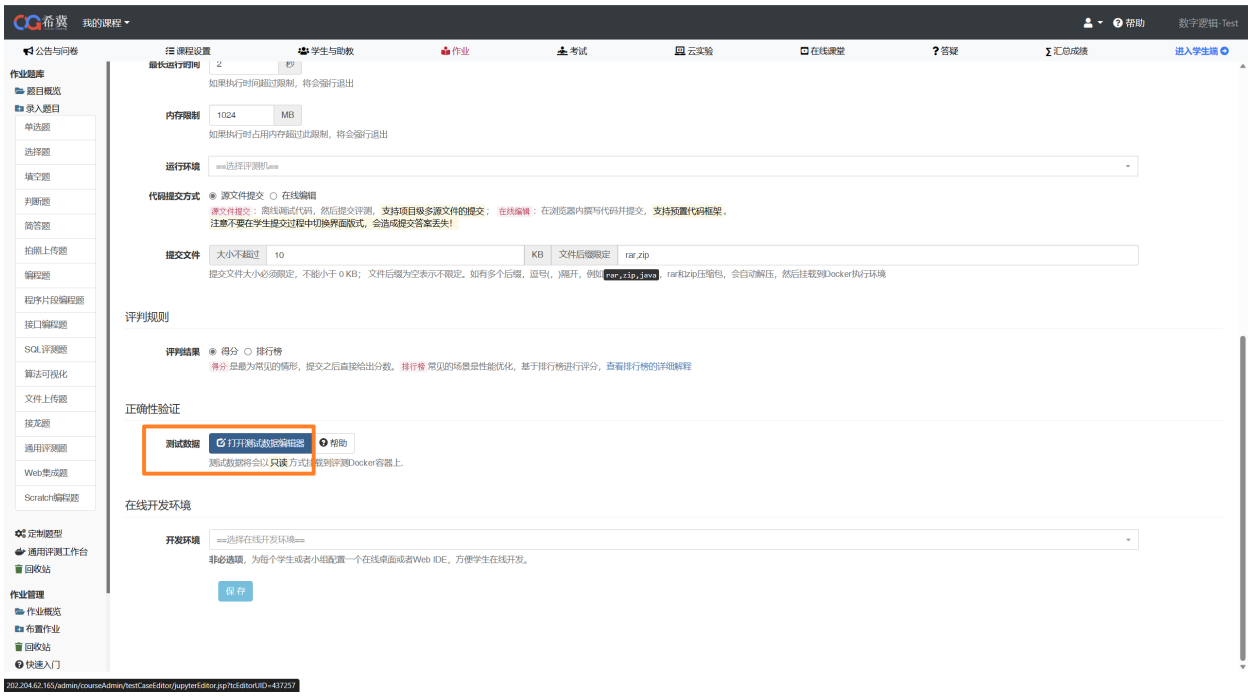

图 13: 打开测试数据编辑器

在测试数据中选择 File 中的 Open

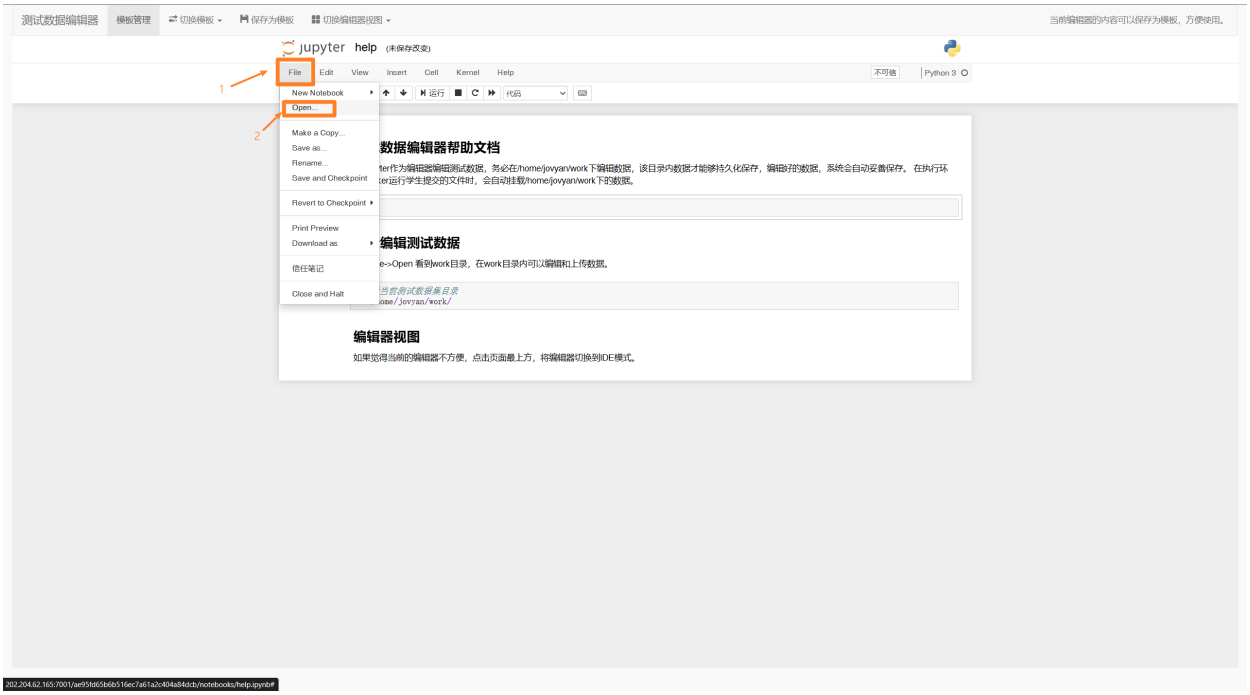

图 14: 打开测试数据

进入 work 文件夹，选择 Upload 即可上传数据

图 15: 上传数据