面谈（电话、视频）

自我介绍

- 基本信息，姓名、年龄、学校，目前公司；
- 项目信息，微信公众号、项目管理系统后台、bff 中台（生成 pdf、接口聚合）
- 技术栈，前台（vue 框架、element-ui、vant 移动端、ant-design for vue），后台（midway，nodejs）
- 除此以外，还有以下技能：
    - 基于 github actions 的 CI/CD 脚本的编写（workflows、build 打包、deploy 部署（ssh 连接））
    - 基于 docker 容器的部署（编写 Dockerfile，镜像优化）
    - vscode，git，webpack 工具的使用、vscode 插件开发有一定经验

关于报告打印的说法（可以重点说的）

服务端 nodejs 生成 pdf

- google 开源的 puppeteer 库，生成 pdf 是其中的一种功能，可以截屏、生成 pdf 等等
- 在服务端安装启动无头 headless 的浏览器
- 在浏览器中访问 url
- 打印，生成 pdf，保存在某一个地址。

原因

- 用户手动打印，会中断操作，不可批量，而且等待时间长。
- 使用 html2canvas，生成图片，然后在上传后端生成 pdf，（清晰度、pdf 文件体积、等待时间长，不可复制）。
- 不希望中断用户操作，可批量打印、异步执行（用户可以做其他事情，等待执行结果）、清晰度、可复制等等

难点

- 网页的分页，会碰到文字、图片阶段的问题，这时需要动态计算：一整块的（一页内容放不下，需要整体放到下一页），可部分截断的文本（根据行高动态计算偏移的距离，使得文本每行完整的显示出来），具体方法是在该分页的内容上方填充一定高度的 div。
- 页眉、页脚、页码的自定义（比如报告封面、目录没有页码，从第一章开始算起等等）。
- 页面内容可编辑（编辑页面、渲染报告页面分开）
- 报告打印时需要等待的时间（在上面的动态计算，完成分页后，设置一个全局变量为 true，服务端利用 puppeteer 库，访问网页的时候，启动定时器检查：渲染完成后执行打印）。

关于 css

大多数运用组件提供的样式，也有使用 tailwind.css（原子类，接近于内嵌的行内 style，生产环境使用 postcss 构建），其他的编写少量的 css 进行适配。

布局使用 flex 布局，部分使用 grid 布局。

关于 js

- 上面提到的分页计算
- 封装的烟花类，fireworks 类，在页面的 input 中，输入是，展示 fireworks 绽放效果。
- 对 element-ui 的组件的封装：弹框 Dialog 可以拖动（onmousemove 等、边界判断）、放大、缩小；table 的拖动。

vue 组件封装

- 属性透传（element-ui 组件二次封装）
- 弹框组件（拖动、最大化、最小化效果）
- 表格（列拖动效果，二次开发）

Docker 镜像优化

原则

- 减少层，去除非必要的文件
- 多阶段构建
- 选择合适的基础镜像

具体方法

选择合适的基础镜像：

- 使用 alpine 基础镜像（不包含额外的无用程序），减轻体积

减少层，去除非必要文件：

- 使用.dockerignore 忽略不需要的文件
- 多个操作可以合并成一行（因为 Dockerfile 大多数命令会生成 1 层，都会产生额外的体积）

多阶段构建：

因为正式运行的情况下，只需要最后的可执行文件，和必须的依赖就行，所以可以通过前面阶段构建完成后，把结果拷贝到最终的镜像上。

熟悉 liunx 文件操作，方便解决特殊问题

比如 ant-design for vue 和 element-ui 冲突，打包报错的问题解决方案，文件中代码替换（sed 文件操作）。

关于 github actions

介绍

- 仓库托管在 github 上，github 的一种 CI、CD
- 可以方便的根据各种触发条件，编写各种 workflows 工作流，编写多个 jobs、actions
- 账号、密码可以在仓库里面添加 secrets，在脚本中进行引用
- 官方、市场上提供了很多 actions 供调用，比如 docker、ssh 连接等等
- 测试环境、生产环境的部署

流程

- 指定触发条件、指定执行机器
- 拉取代码
- 使用 docker：登录 docker、制作镜像（两个：latest、带日期的）、推送镜像

- 使用 ssh action，登录远程服务器，执行自设的脚本：停止容器、清理容器、清理镜像、拉取最新版镜像、运行镜像。

关于 midway

介绍

阿里开源的一个服务端 nodejis 上层框架。

- 依赖注入，控制器 Controller、服务 Service、各种装饰器等概念
- 配置项、生命周期
- 日志
- 组件能力，各种扩展
- swagger
- mongodb 等等

监控

使用官网介绍的 Grafana + Prometheus

部署

使用 docker 容器 + pm2-runtime

关于 vscode 插件开发

emoji 自动补全、语义提示

code-generate

- 实现：选中一个 js 函数，其返回值将代替选中的字符串本身。
- 优化，有若干预设的全局函数，也可以在配置文件中编写自定义的处理函数，方便复用。
- 应用，通过 swagger 中的接口字符串，生成一个 api 的 controller 类，包括若干静态方法。
- 应用，通过一个数组，生成另一个配置数组