皮大钟

年龄：27 岁　　工作年限：5年，当前未在职　　求职意向：前端工程师

电话：183 2655 6616　　邮箱：pidazhong@whu.edu.cn　　博客：https://blog.quantanalysis.cn

➢ 教育背景

2014 年 9 月 — 2018 年 6 月　　武汉大学　　本科　　计算机科学与技术

➢ 技能描述

具备扎实的 Html、Css、JS 基础，能够快速高效地开发项目。

熟练使用 SCSS、LESS 等预编译架构，了解过 Tailwind CSS 的相关使用。

熟练使用 React 架构，研究过 React 的底层原理，具备丰富的 React + TS 开发经验，拥有3年大型项目团队协作开发经验。

熟练使用 Webpack、Vite 构建工具，了解其底层运行机制，编写过在多个前端团队中广泛使用的 Vite 插件。

熟悉浏览器渲染机制，学习和研究过浏览器框架，在性能优化方面积累了丰富的经验。

熟悉 Express 架构，有 Node.js 服务端开发经验，拥有自主搭建的服务器，并稳定运行多个项目。

➢ 工作经历

2020 年 10 月 — 2024 年 1 月　　阿里–瓴羊智能有限公司、阿里–乌鹊科技有限公司　　前端开发

2018 年　7 月 — 2020 年 6 月　　合肥天源迪科科技有限公司　　前端开发

➢ 项目经验

Dataphin 智能数据建设与治理

项目描述：用中台方法论治理企业级好数据。针对各行业大数据建设、治理及应用诉求，结合数据中台方法论，一站式提供全域数据集成、可视建模及规范定义、数据资产治理等能力，助力企业打造标准统一、准确可信、便捷可消费的数据体系。

负责内容：

- 负责 Dataphin 项目的数据规划、资产治理、数据建模等核心业务模块，成功推动资产–质量、资产–标准等核心模块落地；并在团队内多次组织客户业务实例分享，促进了团队对业务需求的深刻理解。
- 主导项目角色权限升级为页面功能权限点，通过提取页面功能点并将其组织成 Auth–Tree，实现了权限管理的标准化，同时解决了不同售卖版本和多租户版本下代码不统一的问题，大幅提升了系统的维护性和扩展性。
- 主导项目中动态表单 Schema 的设计与实现，支持用户自定义配置表单内容，整体流程有效提高了开发效率，针对几十+表单类似的场景，节约了70%以上的开发和联调时间。
- 主导了项目中多个组件库的整合，如将 viser–react 和 ECharts 替换为 G2v4；并且基于 G2v4 的声明式 API 封装了基础图表，实现了配置驱动，降低了图表的开发和维护成本；同时实现了 G2 的按需加载，优化了60%的打包体积。
- 主导开发接口文档在线联调项目，在开发环境中与项目一同启动，根据后端接口文档 api–docs 地址，生成接口详情页面，实现开发环境下接口文档查看功能，并能直接获取当前登录信息进行联调，减少了前后端联调的沟通成本。
- 开发了 Vie 插件，实现了生产环境下可以从浏览器dom准确定位到vscode对应代码位置的功能，大幅提高开发效率。

使用技术：react + ts + antd + schema + Auth–Tree + g2 + vite plugin

性能监控平台、性能优化

项目描述：针对首屏加载过程中的资源下载、网络请求、渲染耗时等关键链路进行埋点监控和实时分析，有效捕捉首屏渲染的性能瓶颈。同时，通过交互埋点技术监控页面帧率（FPS），分析用户交互时的流畅度，全面提升前端应用的性能和用户体验。

负责内容：

1、性能的评估体系设计与实现

- 设计并实现了针对首屏加载和用户交互两个核心场景的性能监控评估体系，使用 Performance API 定义和计算 LCP、TTI、FPS 等关键性能指标。
- 通过 HOC 实现了 LCP 的核心时间点和 TTI 的精确计算，确保了性能数据的准确性与实时性。
- 利用 Performance API 收集并分析核心页面的资源与接口数据，包括资源总耗时、总大小以及最大资源返回时间，为性能优化提供数据支持。
- 通过 Hooks 监听核心组件的渲染详情，并收集 JavaScript 错误信息，为进一步优化提供了可靠的监控基础。
- 将监控数据接入 QuickBI，构建了性能监控数据的可视化界面，提供了直观的图表展示，帮助开发团队快速定位并解决问题。

2、项目优化

- 使用 懒加载、按需加载 等代码分割技术、预连接、DNS 预解析、组件内部使用延迟加载等方法，优化首屏渲染时间；
- 使用 打包分析进行 Tree Shaking、精简三方库、js和css文件压缩 等手段优化打包体积大小。
- 使用 React ErrorBoundary 和 手动处理 js 错误相关代码，处理白屏和报错问题，确保页面的稳定性和用户体验。

3、最终成果

- 优化了 DAG 图、ER 图及其他可视化关系图的帧率，稳定在 50 以上，确保了复杂交互场景下的流畅度。
- 首屏场景的文件都在 150KB 以下，内部访问场景中优化至 800 毫秒以内，外部访问场景中优化至 2 秒以内。
- 公共云环境下，将项目的 JavaScript 报错数量从每日 3000+ 减少到十位数，并彻底消除白屏现象，显著提升了系统的稳定性和用户满意度。

使用技术：react + ts + HOC + Hooks + Performance API

公共组件平台

项目描述：通过开发专用的脚手架工具，系统化地整理了项目中的七八十个公共组件，并构建了一个独立的公共组件 Demo 展示平台项目，供团队成员随时访问。该平台大幅提升了开发效率，减少了组件的重复开发，帮助开发、产品团队在需求评估阶段进行更精准的规划和调整，成为团队中重要的资源共享工具。

负责内容：

1、脚手架工具开发与优化

- 基于 Node.js 设计并开发了用于自动整理和生成组件库相关文件的脚手架工具。
- 实现了基于 CLI 的自动化处理流程，确保组件库的增量更新和版本控制，完全消除了人工管理的需求。

2、规范制定

- 制定并推行了组件库的命名规范、代码风格和文件结构标准，确保组件的高可维护性和一致性。
- 基于组件的 define.ts 文件，CLI 自动生成 Markdown 风格的组件 API 文档，减少了手动编写文档的工作量。

3、**项目构建与本地调试**

- 利用 Vite 进行项目的构建和性能优化，同时 CLI 支持本地开发环境的调试功能。

4、平台实施与推广

- 负责平台的整体实施与推广，主导了平台从开发到上线的全流程，并推动平台在团队内的广泛使用。

使用技术：react + js + node.js + vite + cli

代码管理与发布平台

项目描述：通过构建 Podman + GitLab 构建 CI/CD 平台，自动化了团队的代码集成、构建和部署流程。该平台支持多分支协作、秒级回滚和代码审查，并通过定制化的脚手架工具简化了前端项目的代码发布流程，项目的自动化流程为开发团队提供了更快的迭代能力。

负责内容：

1、CI/CD 管道的设计与实现

- 设计并搭建了 GitLab CI/CD 管道，结合 Podman 实现了镜像的自动化构建与推送。通过基于 .gitlab-ci.yml 文件的定制化配置，实现了自动化测试、构建、部署流程，确保了代码的高效交付。

2、秒级回滚与分支管理

- 设计实现了秒级回滚机制，支持根据不同分支回滚到任意版本，确保项目的高可靠性和故障快速恢复能力。

3、代码审查与质量保证

- 集成了 ESLint 代码检查和自动化分析打包结果大小功能。确保代码正确编译，能够生成正确的打包产物，并配置了自动推送相关信息到钉群，确保代码交付的质量和有效性。

4、可视化操作平台的构建与集成

- 构建了一个可视化操作平台，集成了公共云环境的 DEF 部署以及基于 GitLab 的独立部署功能。通过直观的界面实现对部署流程的全面管理与监控，极大地提升了操作效率和部署过程的可控性，简化了团队的日常运维操作。

使用技术：node.js + podman+ gitlab + CI/CD Pipeline

➢ 个人评价

1、热爱编程，有激情、有活力，能承受较大工作压力

2、工作认真负责，团队意识强

3、有一定后端开发基础，能与后端开发人员密切配合

4、热衷探索新技术，热衷于为项目应用新技术