李繁宸

1991.06.13 | 13641875733 | lifanchenjulius@hotmail.com

教育经历

纽卡斯尔大学	计算机科学（硕士）	2013:09 - 2015.01
中瑞典大学	信息科学（学士）	2012:09 - 2013.06
东华大学	软件工程（学士）	2009:09 - 2013.06

专业技能

- 熟悉 VueJS
- 熟悉 JavaScript ES6语法
- 了解 CSS3
- 了解 ReactJS
- 熟悉前端工程化
- 熟悉敏捷开发

工作经历

Bybit | 前端开发工程师 | 2021.03 - 至今

- 支持公司业务推进。完成法币入金业务 0-1 落地。
- 支持公司技术栈更新，参与公司旧有项目从vue迁移到react开发
- 参与公司前端工程化建设，参与公司前端工程脚手架开发

竹间智能 | 前端开发工程师 & 前端组长 | 2018.06 - 2021.03

- 支持公司业务推进。完成包括机器人平台，电话机器人，NLU 平台，语音平台等多个旧有产品的有序迭代。协助知识工程，智能知识库，知识图谱等新产品 0-1 落地。
- 主导公司前端开发流程统一与优化，推出前端开发统一规范并以公共组件库与公共脚手架的形式落地。主导公司前端模块业务整合与公司前端产品性能优化。
- 建设专业前端团队，强调团队成员思辨能力，鼓励对潜在业务需求和技术优化点进行自我规划和递进式解决。参与跨团队沟通协调，开展团队技术分享等工作。

思爱普中国 | 开发工程师 | 2015.10 - 2018.06

- 基于业务场景，完成资金管理工具前端操作界面开发工作。保证产品按时迭代

携程 | 开发工程师 | 2014.10 - 2015.10

- 基于酒店后台业务场景，开发并完成酒店预定后端流程操作
- 完成酒店历史数据归集，预定信息发送，异常数据补偿等业务操作。

项目经历

法币入金2.0 | 开发者&管理者 | 2021.03 - 至今

现有痛点

- 旧有项目交互逻辑设计不合理，用户反馈不佳
- 旧有项目代码结构复杂，难以维护

解决过程

- 拆分旧有代码，提取法币入金相关逻辑基于react重构
- 基于Antd封装组件，快速迭代页面，与交互讨论封装完成的组件标准化为组件库
- 抽离公共方法（防抖，节流，轮询等）为公共方法，降低后续相似功能的开发成本

项目成果

开发两周后完成上线，PV，UV提升30%，用户反馈问题下降30%

Common Modules | 开发者&管理者 | 2019.12 - 2021.03

现有痛点

- 公司旧有的前端技术栈包括 VueJS，JQuery 和 PHP，VueJS 项目存在不同种类组件库混用情 况。
- 前端开发水平参差不齐。产品设计，交互设计，前端开发缺乏统一设计基准和开发标准，需 求传递时间长。
- 公司产品迭代周期短，要求快速产出交互统一产品，快速达到可 demo 可投入市场

解决过程

- 统一前端技术栈至 VueJS，基础组件库选择 ElementUI，统一基础功能交互，基于 ElementUI 和自定义样式实现基础功能样式的统一化
- 基于 ElementUI，结合公司业务场景，与交互，产品交流后封装与业务流程相关的业务公共 组件。如页面头部，侧边菜单栏，卡片页，侧边弹出框等。部分业务公共组件由多个业务公 共组件组成且支持单独使用，开发可选择直接使用完整的业务组件直接搭建页面，也支持自 由搭配业务组件形成新页面
- 封装完成的公共组件支持 npm 依赖形式和基于 nginx webserver 的访问两种形式，nginx webserver 支持多产品部署和开发时，不用关注公共部分是否有版本差异或版本升级。Nginx 依赖形式使得某些单独运行产品可以快速集成公共组件能力
- 引入基于 storybook 的前端自动化测试，保证公共组件开发质量

项目成果

Common modules 推出第一个版本后在两周之内帮助公司 12 个主产品，20+功能模块完成交 互和展示初步统一。近一年持续迭代近 20 个业务相关组件。提升新产品开发效率和旧有功 能维护效率 30%左右

前端整合改造 | 开发者&管理者 | 2019.12 - 2021.03

现有痛点

- 公司产品独立开发，独立部署，需要展示融合版本或客户使用多个产品时，产品之间关系零 碎，界面体感分散
- 公司前端技术栈尚未统一，要求短时间内完成整合，业务上需要产品支持独立部署和按业务 需求整合

解决过程

• 第一阶段（2019.12 – 2020.1）为了快速串接业务，达到整合效果。选用基于 nginx rewrite 的多页应用模式。即每个功能模块都统一复用公共业务组件（头部，侧边菜单 栏等）菜单跳转时通过恰当缓存共享信息，从体感上给予用户接近无感切换的效果。

• 第二阶段（2020.5 – 至今）经过一段时间的技术调研，完成前置的技术储备和业务调 整准备后，使用 qiankun 微前端框架来实现前端业务模块有机整合。

项目成果

整合方案第一阶段完成后近一年完成支撑产品客户部署需求和演示需求

qiankun 方案在部分产品中实践得到使用者和开发者的认可，正在全公司铺开推广

前端性能优化 | 开发者&管理者 | 2019.12 – 2021.03

现有痛点

• 无缓存情况下 FCP 时间， Speed Index 时间均在 10 秒以上，白屏现象明显。用户体验不 好，达不到演示标准

解决过程

• 制定性能优化指标，FCP 时间及 Speed Index 时间压缩至 chrome 推荐的 moderate 水平（FCP 时间 2~4s，Speed Index 4s 左右）

• 结合浏览器访问数据指标和 webpack 打包结果，定位单文件过大问题，一方面启用 Vue-router 资源异步加载和 Vue 组件异步加载，支持打包后形成多文件。另一方面清 理，剔除不需要的静态资源，开启 gzip 压缩，压缩单个文件大小

• 基于浏览器访请求访问情况，与后端一同 review 相关逻辑，合并重复功能逻辑，去除 非关联逻辑之间的依赖，页面开启时的请求数由原先的 20+缩减为 5 个

• 与测试部门分享性能检测流程，推荐其将性能测试加入迭代测试范围

项目成果

• 页面 FCP 时间及 Speed Index 时间降低至 3s 以下，平均 2s 左右

• 性能检测纳入测试检测项，大版本迭代必须通过

• 定期开展打包结果 review，单个大文件发现后会统一分析确认是否合适