作为经验丰富的前端开发人员，我曾与基建小组合作优化前端项目的交接流程。我们通过模拟项目交接来确定所需资源和交接考核标准，并就整个流程进行讨论和调整，以确保交接流程既高效又可靠。此外，我还与同事们制定了一份体系化的前端新人培养方案。通过资料搜集和分析，我们决定采用阶段式的深入学习模式，并提供多种格式的参考资料和整体学习路线。我们还会通过作业形式来检验学习成果，以帮助新人能够全面了解并掌握前端开发相关的知识和技能。

自入职起，我一直在 ReportEase 项目组从事前端开发工作，这是一个简化用户写报告的APP。起初，我与 DRI 合作完成基本的开发任务。在2022年初，我开始负责设计品质项目前端工作，并协调其他三位前端开发人员完成了从 React 到 Vue 的项目重构。统一了系统风格并为该系统添加了精密的权限管控、甘特图、PDF 水印预览、任务看板等功能。随后，由于 DRI 离职，我接任了 ReportEase 的前端 DRI，期间主要是加强系统功能，提高报告模版的通用性，并引入 Insight 模式让报告生成更加自动化。在 2023 年，我独自完成了 RD HC 系统的前端开发，并首次将项目从 Vue2 转换为 Vue3 框架，该系统旨在方便EPM进行类似Excel的人员信息管理。此外，我也对 ReportEase 进行了性能优化和用户体验改善的工作，同时添加了新的模块丰富报告的格式。未来一年，ReportEase 项目将同时进行APP维护并逐渐引入 Insight 平台，我将迅速适应与苹果的开发团队合作，以确保不会因模式变动而导致效率下降或产生差异

由于ReportEase的前端代码量大，数据通过前端存储，逻辑也在前端处理，因此优化ReportEase相当具有挑战性。然而，随着性能引起的问题不断增多，比如页面卡顿时间变长、数据偶然丢失等，客户要求我们优先解决这些问题。在优化过程中，首先使用四象限法确定优化的优先级，并发现合并大量数据文件时保持数据准确性是一个关键问题。容量超出一定范围的优化难度相当大，合并后编辑卡顿可酌情优化。对于这个目标，我的优化思路是分两阶段进行，第一阶段是代码层面的逻辑优化，第二阶段是通过工具进行加载优化。代码逻辑优化方面主要采取了ECRS分析法，通过应用ECRS的四大原则：取消、合并、重组和简化，可以让业务流程、系统结构、功能逻辑都会变得更清晰。举例来说，取消冗余数据在系统优化中发挥了重要作用，通过浏览器调试工具发现PASS数据占用了大部分内存。鉴于并非所有PASS中的信息都是有用的，我们对PASS信息进行了精确提取，从而显著减少了PASS数据在系统中所占用的内存空间，合并文件最大容量也从150M扩充到200多M，进一步优化了系统性能。此外，随着需求的不断叠加，一些UI界面变得越来越复杂。通过ECRS原则也成功地将这些零散功能重新整合，为系统优化带来了实质性改进。

在第二阶段，通过Lighthouse对网页进行全面的性能和质量审查后，发现打包后的文件体积过大。为了解决这一问题，我们采取了三个措施：

1. 使用按需加载的方式引入Echart等组件：通过按需加载的方式，只引入页面需要使用的部分，以减少初始页面加载时间，并且降低打包后的文件体积。

2. 启用gzip打包并开启文本压缩，对传输过程中的文本文件（如JavaScript、CSS和HTML）进行压缩，从而减小网络传输的数据量，提高加载速度。

3. 去除未使用的CSS：移除未被使用的CSS样式，以精简页面所需加载的样式文件，从而减少文件体积。

经过以上方案的实施及验证后，与之前的版本对比在，系统反应速度、占用内存、稳定性上有着显著的提升。

接下来一年的主要目标是在Q4季度前完成将ReportEase整合到IPA平台的基础版本，为实现该目标，关键任务包括迁移1.0中各种格式的报告导出。确保在IPA平台中实现ReportEase 2.0报告共享功能。完成ERS新功能开发。次要目标是推動並落地部門項目管理系統，其中包括完成Ship板块的开发与集成以实现项目交付流程的规范化和可视化，设计并实现Project板块，包括任务分配、进度追踪等功能以提高项目管理效率和透明度，开发Testhub模块用于测试管理和质量监控以确保产品质量，最后设计并部署Goals板块，方便团队设定和跟踪OKR，并生成可视化报表。