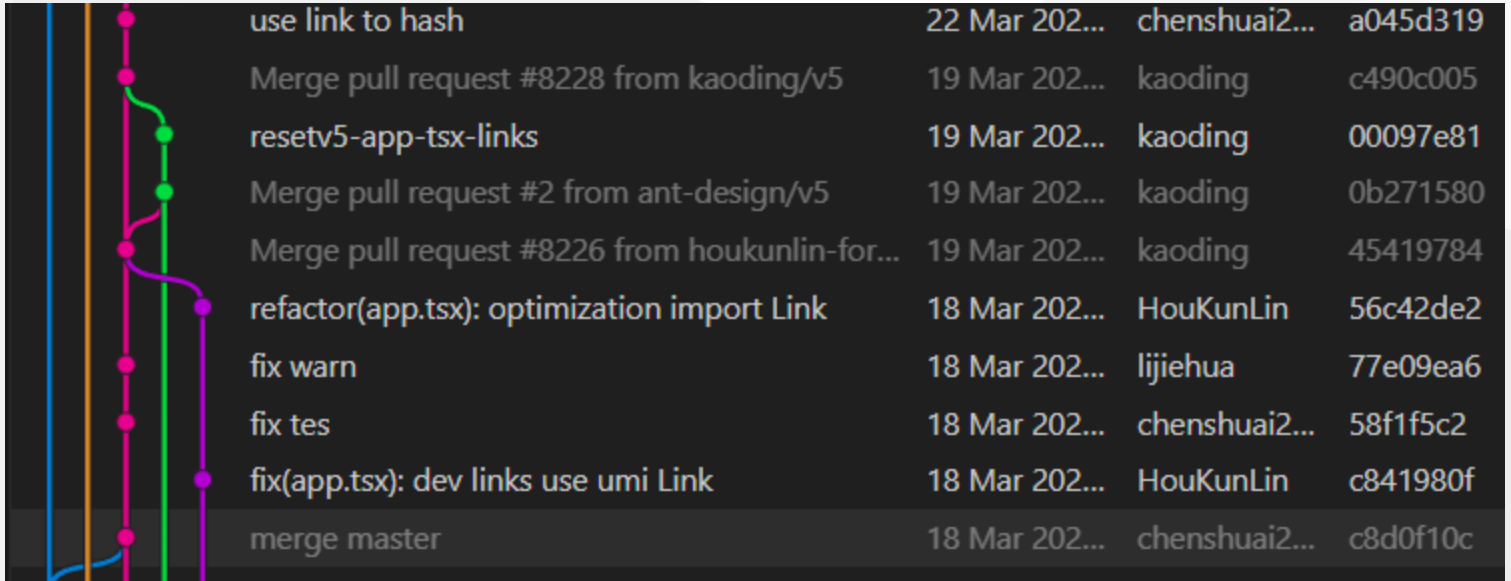
为什么需要分支管理规范

- 并行开发:
  - 团队成员在独立的分支上进行工作，避免冲突和互相影响
- 提供清晰的代码版本历史:
  - 通过合理的分支命名和操作规范，使版本追踪和问题排查更加简单



常见项目分支类型

1. 主分支:
   命名： `master` 或 `main`
   作用：用于存储稳定的生产代码，一般认为是线上的最新版本。
   合并说明：所有合并到主分支的更改都应该经过严格的测试和审核。

2. 开发分支:
   命名： `develop`
   作用：用于集成新功能和修复问题。所有开发人员都应该基于 develop 分支开始他们的工作，并将其提交到该分支上。
   分支创建来源：主分支
   合并说明：当开发周期结束时，应合并回主分支。

常见项目分支类型

3. 特性分支/个人分支:
   命名： `功能名称` / `dev_人员_版本号_功能名称`
   作用：用于开发新功能或修复问题。
   分支创建来源：最新的开发分支
   合并说明：并在完成后合并回 develop 分支。完成后删除分支

4. 测试分支
   命名： `test`
   作用：用于集成测试和回归测试。
   分支创建来源：最新的开发分支
   合并说明：并在完成后合并回 develop 分支。完成后删除分支

常见项目分支类型

5. （预）发布分支:
   命名： release 或 v1.0.0 或 迭代号
   作用： 在准备发布之前，创建一个发布分支 (release)，用于进行最终的集成测试和版本控制。此时不应该再向 release 分支提交新的功能或问题修复。
   分支创建来源：开发分支
   合并说明：当 release 分支准备好发布时，它将被合并回主分支和开发分支，并打上 tag。完成后删除分支