**架构设计文档**

**文档介绍**

本文档是实时协作白板平台架构设计文档。其中，具体说明了系统架构设计的目标需求，规定了系统的开发环境和运行环境，阐述了逻辑架构设计的整体思想和具体设计的实现，介绍了系统的主要业务流程和系统的功能模块的划分以及模块之间的联系，表明了系统采用存储的数据库类型以及数据库又采用了什么设计编辑工具和数据库中存储的各种表的具体信息，最后是设计了系统核心功能的界面原型。

**目标需求**

为了方便广大需要协作画图的用户进行操作，系统以界面简洁明了、易于掌握和应用为前提，实现了画图所需的几种主要工具：绘制矩形、椭圆、直线和箭头、使用自由画笔和橡皮擦、插入文本，同时也支持用户对元素进行样式修改和美化，并且还实现了缩放、旋转、回滚等功能，以满足用户的需求。

**模块分工**

团队采用敏捷开发项目管理模式，重视迭代过程和用户体验持续的改良；

架构上，采用主流前后端分离独立开发的方式， 前后端采用http协议处理请求，采用ws协议进行协同

**前端：typescript + React 框架 江咏麟（主要负责人）、郑思晴**

**后端：golang + gin 框架 王中华（主要负责人）、江咏麟**

**后端架构**

**缓存架构**

主要使用Redis进行缓存，尽量减少数据库IO操作

进行Redis存储的时候，大量使用pipeline管道操作，大幅提高效率

**持久化架构**

使用MySQL进行持久化存储

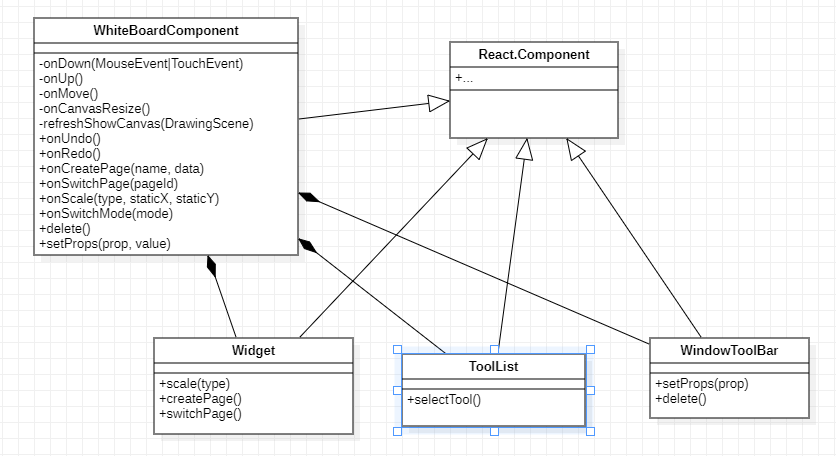
**前端架构**

**白板渲染框架概述**

**交互层**

交互层负责：

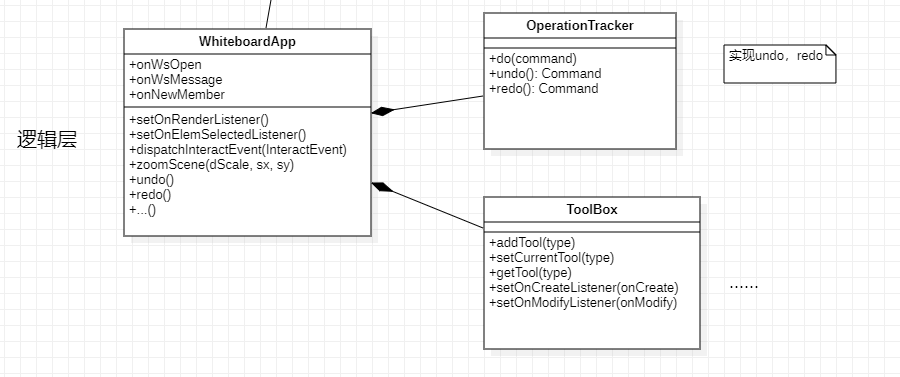
1. 接受用户交互事件
2. 对已经渲染好的场景进行绘制
3. 以及提供其他组件的比如：撤销、重做，切换页面，创建页面等。（下图只展示白板组件）



**逻辑层**

逻辑层主要负责

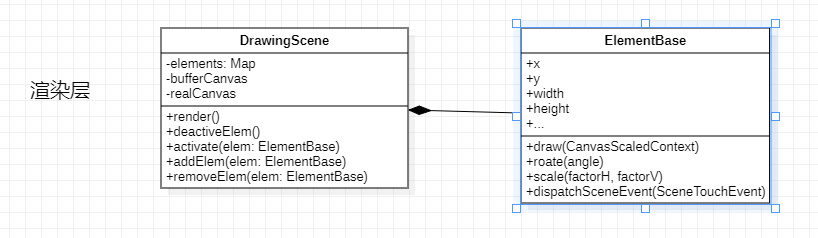
1. 交互层操作的具体业务逻辑实现
2. 接收websocket消息并进行处理
3. 与渲染层进行消息进行交互（比如缩放渲染层，分派事件到渲染层元素），同时在需要时更新渲染层，主要使用观察者模式进行消息通信



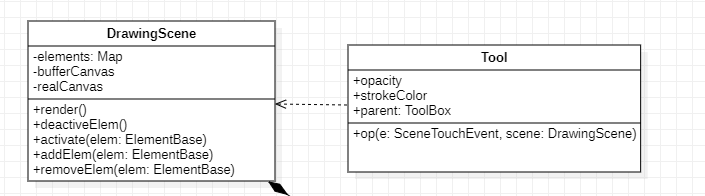
**渲染层**

渲染层主要负责

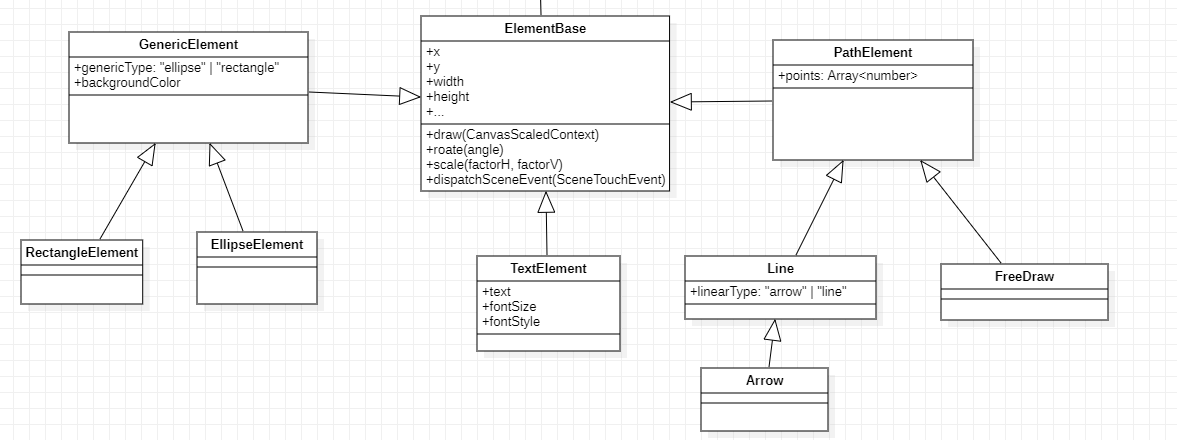
1. 实现对绘制元素的渲染，添加以及删除
2. 实现从交互事件 -> 场景事件的转化
3. 实现场景自身的缩放，平移逻辑
4. 实现元素自身的具体绘制逻辑， 以及缩放，平移转换。



工具介于逻辑层和渲染层之间，主要负责实现操作页面中的元素。



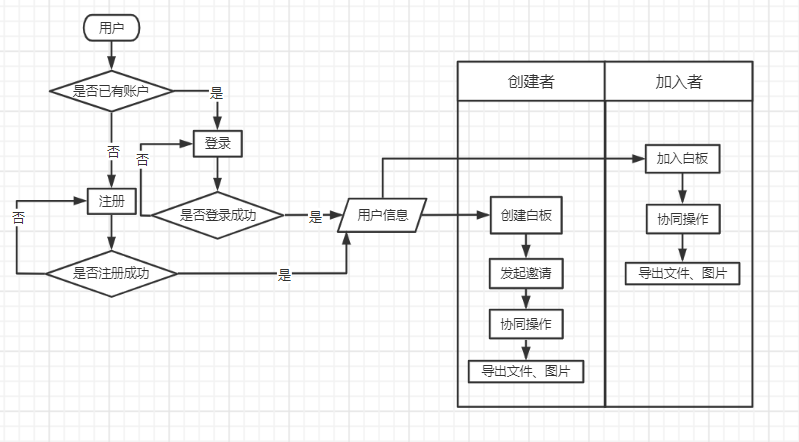
主要元素：通用元素（椭圆/矩形），路径元素（直线，箭头，自由曲线），文本元素



主要工具：对应于上面的元素，对应有不同的工具，除此之外，还有橡皮擦（Eraser），选择工具（Selection）。

**业务流程**

系统的主要业务流程如下图所示：



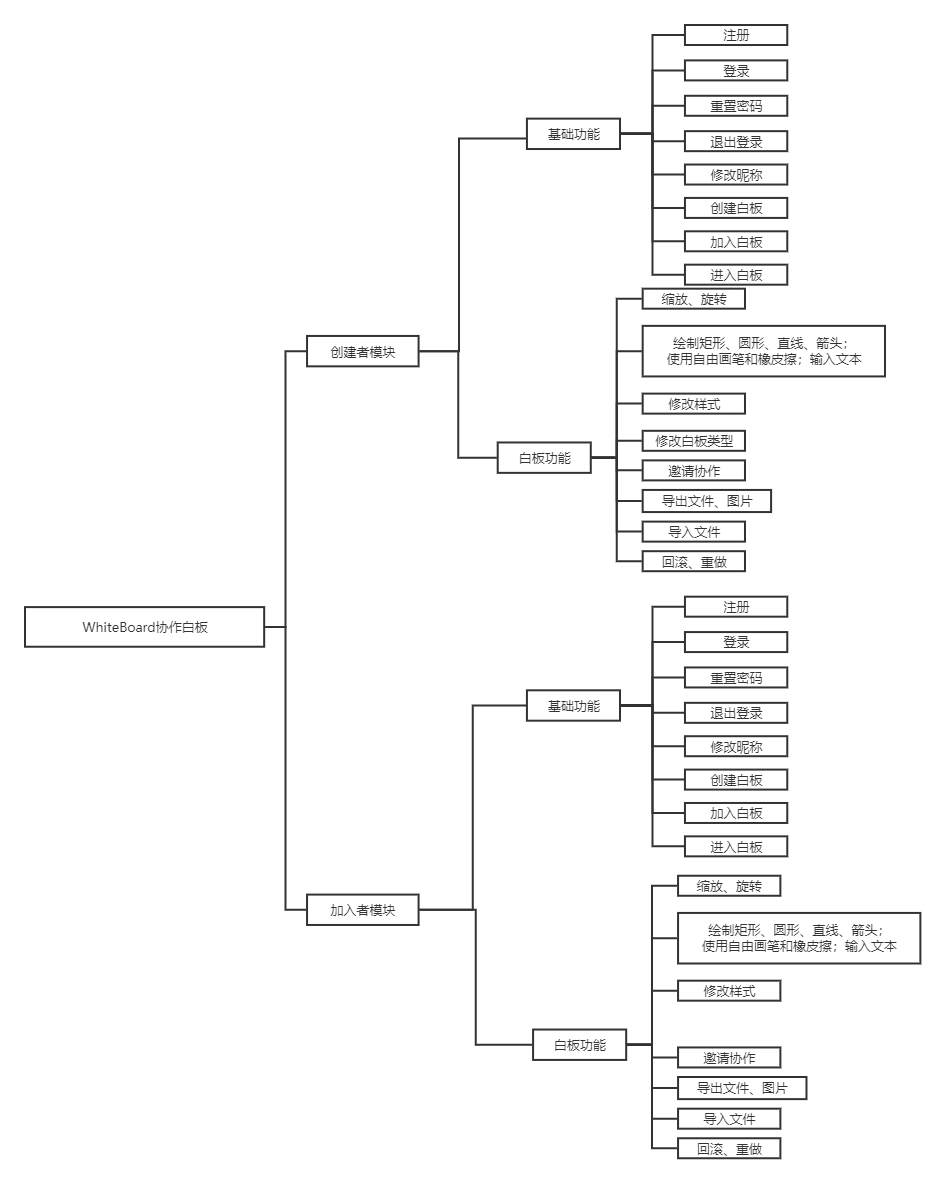
用户成功登录之后，可以选择快速创建新的空白白板，或者通过输入白板序列号匿名加入已有白板进行操作。每个白板默认只有一页，用户可以自行添加新页。每个用户都可以在白板的某一页上创建矩形和椭圆这两种规则图形；也可以使用直线和箭头工具；还可以使用自由画笔、橡皮擦和文本输入，同时以上操作会实时同步到其它进入同一白板的用户的操作界面。

用户还可以选择某一个元素（图形、文本、线条）来修改样式：填充、描边、线宽、字体大小、字体样式、文本对齐和透明度；还可以对所选元素进行移动、缩放和删除；同时能够回滚/重做自己最近编辑的内容。

白板创建者可以修改白板的模式——编辑和只读。在只读模式下，每个用户都不能编辑，且当前页自动同步。

用户还可以以文件或图片的形式导出白板当前页的内容，也可以导入之前导出的文件。

**功能模块**



**界面原型**

Figma原型文件链接：

https://www.figma.com/file/4NNHX7abZy9yhXqsjBBp4l/%E7%99%BD%E6%9D%BF?node-id=0%3A1&t=FrnaJ1j1cphmvYaN-0