工程实践开题报告

1、课题目的及意义

课题目的：

本课题的目的是设计和实现一个网络在线的WEB开发IDE，基于WEB端实现，免除了传统WEB开发需要配置繁琐的环境以及耗时下载庞大的编译工具，而且拥有分享功能，为快速开发和展示提供了便利。

课题意义：

当下市场上的主流WEB开发都是使用本地的开发工具，做好各种配置后进行开发。WebIDE则是一种针对编程开发人员的在线集成开发环境，编程人员无需本地安装开发环境，只需打开浏览器就能立即开发，同时支持断点调试、版本管理、团队开发等。极大的提高了开发的效率，并且有效地解决了本地电脑因意外而导致地数据丢失，项目开发进度停滞等问题。

对于WebIDE的价值，可以从个人、团队、研发效率、公司层面来展开：

1、对于个人简单场景而言，WebIDE可以快速制作出开箱即用、随时修改的Demo，天然适用于技术知识点展示场景，例如制作一个CSS动效、一个自定义hooks、一个小游戏等等，用在技术分享、博客撰写甚至是课堂教学、布置作业等都是极佳的实践。

2、激发团队活力，提升新人留存率。本身作为一项不太普及的技术，其实或多或少能够激发团队的思考，增加团队活力。而作为一项未来可能成为前端基建的技术，对于新人的留存是有帮助的。基建不完善，那么新人招来了，内心必定会吐槽，流失也就理所当然。

3、切换项目更方便。对于multirepo的前端团队，相信不少同学经常会在3-4个项目间切换，项目启动关闭一次较麻烦不说，对电脑的电量，内存都是不小的负荷。而webide将这一部分负荷转移到远程容器，省去了本地的负荷和每次启动打开的不便(finder -> terminal -> run dev -> open vscode)，说实话，个人感觉开发起来更方便。毕竟启动项目就是打开一个网页，而关闭项目也无需担心进度未保存。

4、团队配置化的统一。统一化插件、ide配置，代码移交更顺畅远程办公。疫情之下，不多说，不必合适的设备即可写代码，便利性极大提升，代码更加安全。稍加限制，即可防止代码被拷贝，同时从公司层面，衡量程序员的工作时间会更加方便。

5、demo编写方便。开源的公开的工具很多，但是往往需要私有包，这一点外部开源项目基本无能为力

6、研发平台的整合，闭合研发链路。团队大了，各个平台层出不穷，开发过程中往往会伴随多个平台的切换，将IDE嵌入网页，同时整合各个研发平台，避免反复切换平台带来的繁琐与不便

私有插件。当然避免团队插件发布共有域的方案也有，但是直接搭建私有插件市场，更加方便直观。

对比传统的本地ide例如VS Code，webide具有以下特点：

1、分享和快捷：这是显著区别于VS Code的特性，开箱即用的demo分享可以让用户们在社区大量传播自己所制作的项目，而别人只要点进去就可以看到完整的代码和预览效果，并且可以随时调试，其背后支撑的开发环境搭建也是用户无感知的，以往的开发环境搭建每更换一次电脑，就要重新安装开发环境，但是webide无需重新安装，在任意一台电脑上打开浏览器就能立即编码开发

3、高效，webide支持团队实时在线进行代码沟通协调，提高了软件的开发效率；

4、安全，代码通过实时保存在云端，并进行多重备份，即使本地电脑硬盘坏了，也不会造成代码丢失；

5、环境恢复。有些时候你错误的设置或者删掉了某些东西，其后果是灾难性的，有时候你甚至需要重装系统。而在 WebIDE 里，你只需要重启就可以恢复环境，如果 WebIDE 支持环境快照，你甚至可以恢复到发生错误前的状态；

6、协作编辑。这个可以说是 WebIDE 的卖点之一，虽然近来传统 IDE 和代码编辑器也在引入这一特性，但与 WebIDE 比支持的力度不一样，Cloud Studio 甚至在 IDE 里嵌入了一个聊天室；

2、课题主要内容

本课题的任务是：设计和实现一个在线即开发，随时随地可开发的网络开发IDE：

1、 系统提供良好用户体验、易操作的用户交互界面。

2、 系统提供几个常见的技术栈模板，供用户选择使用，可使用模板一键创建项目。

3、 系统提供实时预览功能，开发人员在平台内即可预览自己的项目运行后的效果。

4、 通过该平台，用户能对自己的项目进行分享，其他人也能看到自己的项目代码以及效果。

3、国内外研究现状

1. 国外研究现状

Eclipse Che号称为老牌开源IDE Eclipse的下一代版本，该项目于2014年10月启动，2016年发布初始版本，现版本为6.7。其主要开发团队募集到900万美元并成立一家独立公司Codenvy，该公司现在基于Eclipse Che提供SaaS服务。由于该项目是开源的，因此其贡献者还包括IBM、红帽、三星等公司的工程师。

除了使用Codenvy的SaaS服务，还可以在任意的Kubernetes和Docker中运行Eclipse Che的本地版本，Eclipse Che在6.0版本之后也支持OpenShift平台。2017年5月，红帽宣布了openshift.io在线开发环境，其中IDE部分由Eclipse Che负责。

3年前在AWS re:Invent 大会上AWS 宣布推出 Cloud9, 用于在云端编写、运行和调试代码，它可以直接运行在浏览器中，也就是传说中的 Web IDE。Cloud9原本是一家创业公司的产品，于2016年7月被AWS所收购，在经历一年半的雪藏后终于重新发布，而这次它是以和AWS各项产品深度整合的面目出现。 AWS Cloud9提供了一个长达470余页的文档，里面列出了Cloud9的各种使用场景，包括：

创建、运行、调试AWS Lambda函数、API Gateway、Serverless应用；在线编辑AWS Lightsail instances（相当于应用市场）里预置的应用，如WordPress、LAMP、 Drupal等；与持续交付工具链AWS CodeStar、CodePipeline集成；与AWS CLI、aws-shell、各语言环境的AWS SDK集成。

除了AWS之外，众多公有云厂商也都在打造自己的CI/CD服务，比如红帽的openshift.io，微软的Azure DevOps Project，腾讯云的CCI（暂未上线）、阿里云云效等。在这些持续交付或者DevOps服务中，IDE也是它们的支持部分之一，WebIDE可以很好的融合到CI/CD流程当中，甚至由于开发习惯和体验的原因，可以将开发者“软绑定”在自己的平台上，因此WebIDE受到部分云厂商的重视。

2. 国内研究现状

2018年4月，腾讯云一亿元战略投资Coding，推出Cloud Studio云端IDE。在开发工具中，IDE一向只是开发工具提供商的自留地，但它现在俨然已成为云计算厂商的目光焦点。

2020年3月27日，华为开发者大会2020（Cloud）开幕，吸引全球众多开发者关注和参与。 华为云DevCloud研发总监王亚伟在主题演讲中重磅介绍了华为云CloudIDE，引起开发者关注。作为一个面向云原生的轻量级WebIDE，华为云CloudIDE是华为云DevCloud在开发阶段的核心服务：

- 可以为开发者提供轻量极速的在线编程体验，帮助开发者快速可靠交付代码，并打通整个开发、测试和运行时。
- 同时，CloudIDE支持鲲鹏原生的开发环境，10分钟快速开发部署鲲鹏云原生应用，解决当前鲲鹏开发者的最大痛点。
- 3月27日，华为开发者大会2020（Cloud）开幕，吸引全球众多开发者关注和参与。 华为云DevCloud研发总监王亚伟在主题演讲中重磅介绍了华为云CloudIDE，引起开发者关注。作为一个面向云原生的轻量级WebIDE，华为云CloudIDE是华为云DevCloud在开发阶段的核心服务：
- 可以为开发者提供轻量极速的在线编程体验，帮助开发者快速可靠交付代码，并打通整个开发、测试和运行时。

4、同类产品调研

目前市面上已经存在一些较为成熟的WebIDE产品，如code sandbox，codepen等等。

CodeSandbox目前定位为快速进行Web开发的IDE和原型工具，目标定位与加快Web开发。使用CodeSandbox，可以快速原型化，轻松实验，并通过单击共享创作。可以在任何带有web浏览器的设备上创建静态站点、全堆栈web应用程序或组件。支持Angular、Vue、React以及各种类型，其提供了免安装配置，实时预览等功能，并且更容易多人协作，通过提供可分享的URL，可以和其他同事和朋友随时随处分享代码片段，可以提供可重现的bug报告，在Stack Overflow或者Twitter上提问或者回答的时候使用代码和环境。

CodePen是一个面向社交的开发环境。它的核心是允许您在浏览器中编写代码，并在构建时查看结果。对于开发人员尤其是处于学习阶段的人来说，这是一个有用且自由的在线代码编辑器。CodePen主要关注前端语言，如HTML、CSS、JavaScript，以及转化为这些东西的预处理语法。之所以称CodePen面向社交，是因为CodePen同时也是一个社区。CodePen上的大多数创作都是公开的和开源的。可以与其他人或社区有机互动，从简单的鼓励，到发表评论，再到根据自己的需要进行创建分支和修改代码。

5、项目目标描述

功能目标：

1. 文件树，支持拖拽移动、拖拽上传、打包下载、重命名、搜索、同名覆盖确认提醒和多选批量操作；

2. 代码浏览与编辑：实现基本的代码编辑、保存功能，支持主流语言的语法高亮和代码提示，与VsCode一致的快捷键 ；

3. 窗口管理，支持自由分窗和拖拽，并能保存布局；

4. 预制多款主题：允许用户修改代码字体和大小，并设置不同背景颜色

进阶目标：

1. 预置环境：前端技术栈支持 预置Vue、React、Svelte等项目模板，可使用模板一键创建项目；

2. Web Terminal、NPM Dependencies Support 后端运行在 docker container 内部，随意安装软件，不会影响宿主机环境；

3. 与Github、Vercel等平台结合，平台提供完整的WorkFlow 代码仓库建立、Github Action 使用、当合并请求中代码存在合并冲突时，可通过 WebIDE 快速解决；

4. WebIDE实现专业代码的在线开发，同时能对代码进行在线调试，提高排错效率;

6、课题需求分析

（1）用例分析

WebIDE旨在解决传统的NativeIDE的一些缺陷，因此需要实现传统IDE的一些功能，这些功能包括：

账号体系：实现注册登录，鉴权等功能。

代码编辑：实现基本的代码编辑、保存等功能，并能同步到服务端。

文件管理：实现文件的增删查改，并同步到服务端。

实时预览：工程项目可以在平台内进行预览。

项目分享：项目可以分享，其他人而已看到用户的代码以及效果。

基于以上功能需求，系统用例图如下所示：

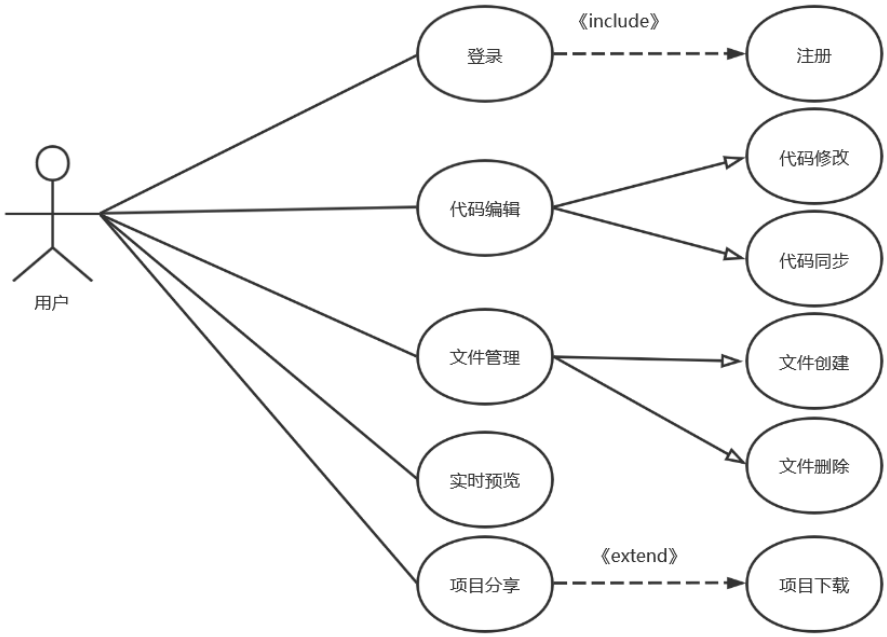

图6-1 系统用例图

（2）抽象设计

在功能性需求中，最基本的就是用户登录鉴权功能。因此最基本的类为User，系统中其它类应和User保持依赖。

由于每个用户都可以创建单独的项目，因此系统中最重要的类为项目类(Project)。项目的来源是项目模板，因此整个项目和模板具有相同的生命周期，是组合关系。考虑到用户编写的代码文件包含多个文件以及文件夹，Project类和File为聚合关系，因为每个项目中，包含多个文件和文件夹。项目中创建时除了自带的模板文件，还有用户自己创建的静态文件，因此抽取了公共基类File，文件夹类FileFolder和编辑文件类EditFile,二者都继承了File，不同的是，编辑文件多了属性file_suffix,这个属性标识了文件的后缀，那么前端可以根据文件的类型来选择对编辑文本内容高亮的规则(比如对.js文件和.html文件，高亮的规则则是不同的)。

此外，系统中较为重要的类还有View类，这个类允许用户完成代码编写后进行在线预览，和Project是依赖关系。

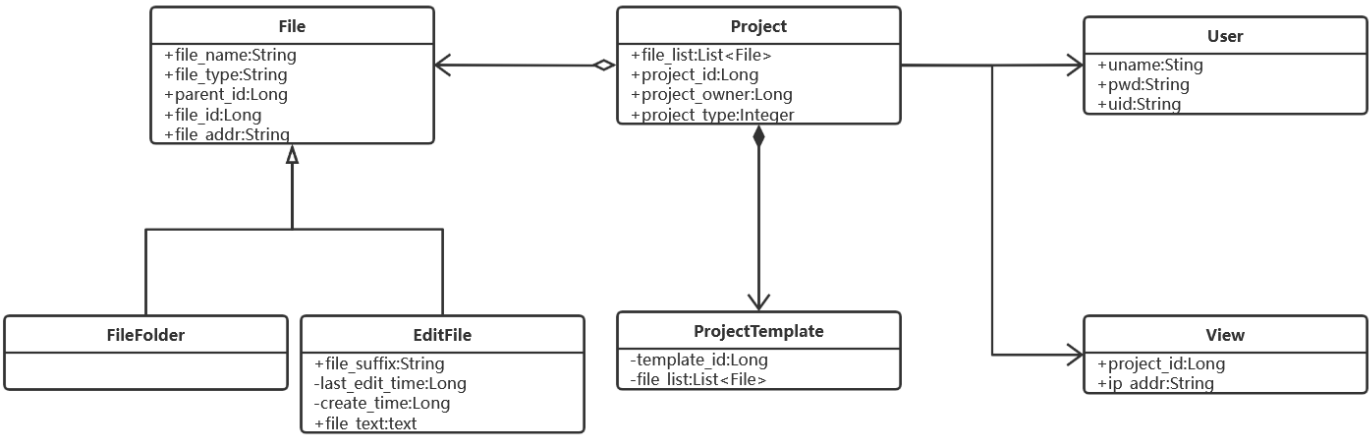

图6-2 系统类图

7、技术方案

从技术上讲，系统作为典型的Web应用依旧是分为前端、后端两个部分，其中前端框架采用React + TypeScript开发，后端使用Node.js，后端框架可以在Nest.js + TypeScript 或 express 之间选择。

前端框架React我们会采用函数式组件React Hooks，默认的useContext作为状态管理，UI的选择还需要后续分析。

抛开框架之外，IDE中的文本编辑器是用户体验的重中之重，其内部复杂程度也是远超我们的想象，还好市面上有成熟的解决方案——monaco和code mirror，monaco就是我们熟知的VS Code中使用的文本编辑器，而code mirror的诞生时间则更早。

为什么选择TypeScript

TypeScript为前端增加了静态类型检查，我们只需要提前写好接口类型，编辑器就会自动给我们语法提示、类型纠错，实现真正的“代码一把梭”，再也不用在使用变量的时候对其是否存在或是类型不确定而畏手畏脚。

TypeScript有一些学习成本，这对团队编码不太有利，所以我们仅使用TypeScript中的一些基础概念和类型，例如尽量不使用高阶类型和各种体操，对于冗长陌生的静态类型名也可以考虑用any直接一笔带过。虽然这确实是图方便，但也算很好地利用了TypeScript来为项目增加可维护性。

另外一点就是，TypeScript对现代框架例如React、Vue 3、Nest.js的支持度已经非常高了，可以做到真正的开箱即用。

为什么选择React

首先，前端中型大型项目普遍使用React，大厂基本的前端框架技术栈也是React，这算是共识了。不过在这里我想多对比一下React和Vue，而不是因为“别人说要用，所有人都在用，那我们就用吧”。

相比较于React，Vue的上手难度确实是要低一些，我本人也非常喜欢Vue，而且官方还贴心地准备了各种指令语法糖和配套库，真的是开发者怎么爽怎么来，例如表单、CSS模块化、全局状态管理、

全局函数都有很统一且完善的解决方案。

而React比Vue在“框架”的道路上要更纯粹，我们只提供一个框架，至于内部怎么实现要靠开发者自己去填——上段Vue中提到的特性，在React都有各种五花八门的解决方案，对于大厂而言，我想他们一定有自己独特且一致的方案去解决，“写Vue就像写Vue，写React就像写JS”。React的另一个点是Hooks，Hooks虽然带来了很多人都一头雾水的渲染逻辑，但其简洁的函数式组件以及逻辑片段复用确实提高了效率，使用Hooks函数式组件已经是React的趋势。

为什么选择Monaco

首先是因为VS Code中使用的编辑器就是它，有VS Code背书，这已经足够有吸引力了，而且其美观程度和完善度均要强于其他编辑器。

再者，据调研时发现，像LeetCode等以编辑器为核心元素的网站，其使用的框架也是Monaco，这足以证明Monaco的吸引力。

8、时间安排及工作进度

2021.11.-2021.12资料收集，分析和整理。

2022.01 -2022.02系统需求分析。

2022.02-2022.02系统数据库设计及数据准备

2022.03系统概要设计

2022.03-2022.04系统详细设计

2022.04系统测试

2022.05系统工作总结，准备答辩。