计算思维可视化学习平台

一、项目选题概述

计算思维是计算机学科的核心思维。华裔科学家周以真给出了如下定义：“计算思维是运用计算机科学的基础概念进行问题求解、系统设计、以及人类行为理解等涵盖计算机科学之广度的一系列思维活动。”典型的计算思维包括分而治之、模块化、递归、迭代、抽象和分解等。下图给出了该项目可以选择的计算思维学习子主题，可以从中选择一个或者多个子主题作为可视化学习的内容。在完成 Web 技术开发实践的同时，也加深了对计算思维的进一步理解，是本选题的初衷。

本项目以计算思维为主题构建在线可视化学习平台，为计算机学科初学者提供一个直观、生动的计算思维学习平台。围绕计算思维，可以选择其中的各种子主题，如复杂数据结构和算法的可视化，图灵机可视化，生命游戏，深度学习可视化等等，也可以基于 google blockly 来实现类似于 scratch 的学习平台。从而体现了项目式学习给学生选择权的黄金原则之一，同学们可以选择自己感兴趣并且对自己有帮助的主题。下面以基于 google blockly 来实现采用可视化积木块式编程，从而学习算法的主题来进行示例。需要学习者对 Google Blockly 以及其他 Web 领域前沿技术的自主探索与学习来完成项目，体现了一定的工程素养的培养，实现具有新工科理念的学习。

二、系统需求和分析

2.1 功能要求

2.1.1 基本功能

- 学习者前端页面
    - 用户登录、注册；记录和浏览个人信息、操作记录、场景历史等。
    - 选择相应的学习场景学习计算思维。
- 学习者后端管理页面
    - 记录学习者信息和学习场景的完成情况，可以结合学习者个人情形设计丰富的内容。
- 学习过程可视化
    - 利用 Google blockly 二次开发构建通用的学习模块，将编程从代码的打字输入编程转化为可视化的模块拖拽，不仅免去学习语法的前置门槛，还能增加编程趣味性，提高用户的学习积极性。如前文所述，这里仅以采用 Google blockly 的可视化积木式编程为例。也可以采用其他 Web 前端的可视化技术来学习计算思维的某个主题。
- 计算思维学习场景
    - 分析并设计比较适合可视化学习的计算思维，归纳总结其特点，并为之设计一个或多个具体的应用场景供用户学习。每个场景可以分为多个步骤，用户通过完成每一步的任务，逐渐加深对该计算思维的理解，最后达到掌握计算思维的目的；
    - 任务完成后的结果可以通过动画演示。
- 学习历史记录
    - 考虑到时间原因或者是用户被某个场景的任务卡住，平台应提供历史记录功能，让用户在下一次进入该场景时，可以从前一次未完成的地方继续学习，而不是再一次从头开始，重复已完成的工作。

- 云平台上的部署
    - 系统部署在云服务器上，提供可以访问的公网地址。

2.1.2 进阶功能

- 协同学习。当学习者学习过程中遇到问题时，可以求助老师或者其他学习者并进行在线协同学习。具体操作上，可以新开一个共同学习房间，并把自己的房间 id 告诉朋友或老师，让其加入房间。在同一个房间的学习者之间支持视频通讯，辅导者演示等功能。辅导者演示的时候，其演示步骤可以通过 Web 页面与被辅导者共享。
- 任务完成后的结果动画演示采用 Web 3D 技术如 Three.js 实现三维展示，其中可以充分利用 Three.js 的各种功能来增强用户体验，提高学习兴趣和效果，比如物理引擎、音效等。

2.2 非功能性要求

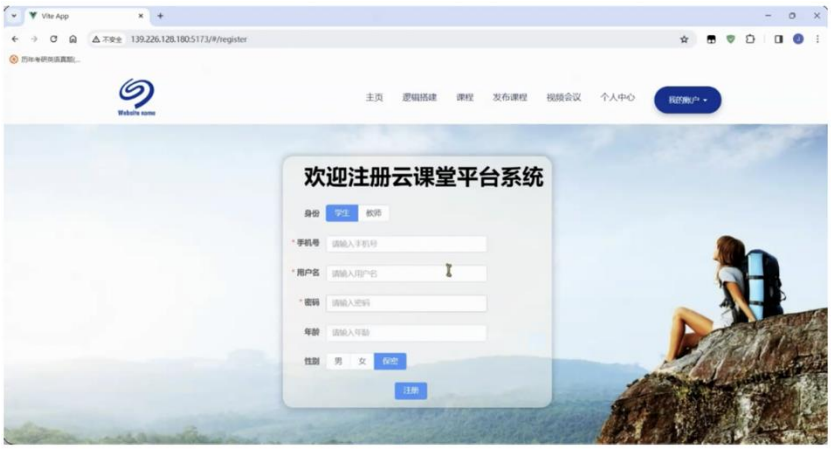
除了功能需求外，平台还有以下几个非功能性要求：

- 易用性：由于针对的是对计算机学科的初学者，平台需要通过细致的提示和简单的操作让用户简单、快速地上手，减少学习成本和受挫感。
- 稳定性：平台应在某些服务出现问题后只是部分功能缺失，比如视频聊天功能在某些网络环境下无法正常运行，但是其他功能仍能稳定运行。
- 可用性：具有较好的用户体验性和可用性。页面返回在合理的范围内，比如通常需要在 3 秒内返回。视频聊天具有合适的帧率，音视频都比较流畅。

2.3 系统功能分析和场景

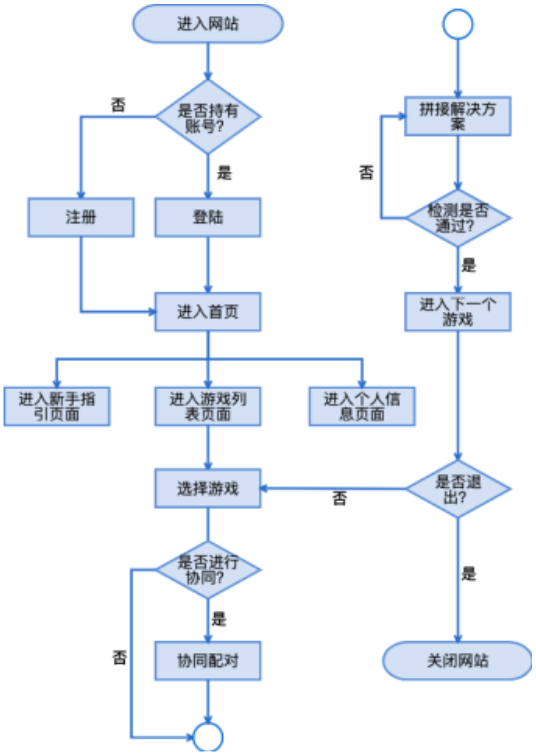
2.3.1 平台工作流程描述

- 注册与登录界面



- 计算思维学习界面

以基于 Google Blockly 的可视化积木式编程为例，下图描述了该学习平台的工作流程：



首先，用户通过在浏览器中输入网址进入主页。如果是新用户，则需要进入注册页面进行账号的注册，完成注册后账号会自动登录并跳转到主页。已有账号的用户则进入登录页面输入账号和密码，完成登录。从主页中可以选择不同的菜单项，可以进入新手引导页面，也可以进入个人信息页面。最主要的功能是从主页可进入游戏列表界面进行游戏的选择。选定游戏后就会进入到核心功能页面，显示如下 2.3.2 节所示学习场景。在该页面中，用户可以看到游戏场景的文字描述和游戏场景的 2D 或 3D 画面，通过从页面左侧工作区的工具箱中拖拽指令块，进行游戏化场景的具体问题求解。完成指令块的拼接后点击“运行”按钮即可查看应用结果，结果会在游戏场景中通过动画的方式展现出来。平台进而显示下一关的游戏场景。如果所有游戏场景的问题都已经解决，则给出提示信息，用户确认后退出该学习平台。

如果用户解题遇到困难可以通过页面右侧的协助模块进行求助，平台进行协同配对后从而实现协同学习。用户点击开设房间，等待对方加入后即可点击“开始视频”按钮开始视频聊天。双方建立连接后由一方获得控制权，进行游戏场景中的问题求解操作，另一方只可观看不可操作该界面，在操作方转移操作权后另一方才可进行操作。

用户可进入个人信息界面查看相关信息，如在个人信息选项卡中查看账号信息，在“游

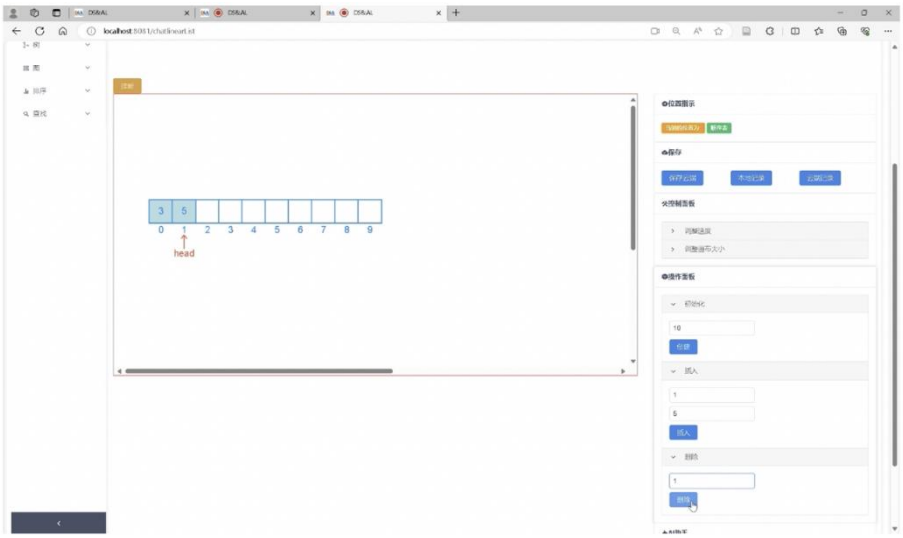
玩情况”菜单项中查看游戏的通过情况和近七天的游玩时长统计图，在历史操作中查看游玩历史记录。当用户不会使用该学习平台时，可以进入新手引导页面查看新手指南。

2.3.2 示例场景

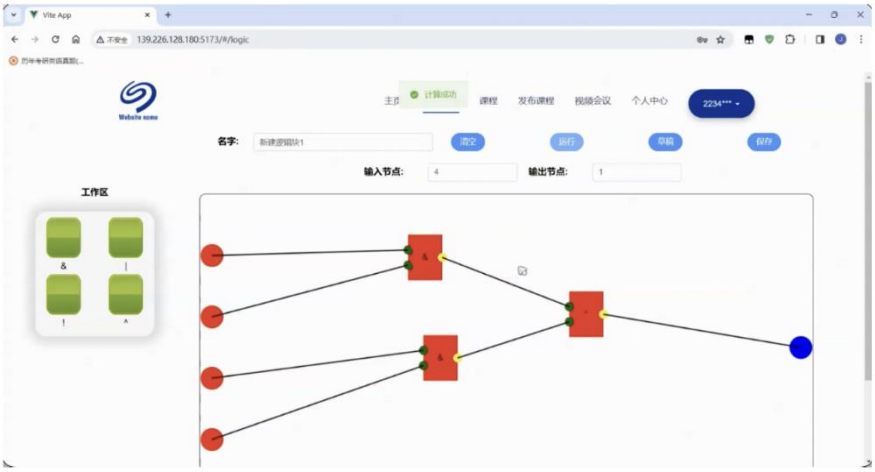
以下为一个计算思维可视化学习平台的场景截图。在这个页面中的左侧，用户可以拖拽自定义的积木块，可视化地完成给定的场景的任务。用户点击运行后，场景中的角色可以模拟用户指定的动作并动画演示，无论失败还是成功都要给出对应提示。最右侧的协同学习是进阶功能，可以请求老师或其他学习者辅导自己，实现协同学习。协同包括视频通讯和操作演示，一般可以开设共同学习房间实现协同。

还可以是其他计算思维的选题，而不是一定要使用基于 Google blockly 的可视化积木式编程。下面给出一些示例界面。

比如数据结构和算法的可视化学习：



又如，逻辑电路可视化学习：



三、技术要求和参考

3.1 技术方案

- Web 开发技术基础如 HTML，CSS，JavaScript，AJAX 技术等。
- 前端采用 Angular 框架开发，也可以自主学习并采用 Vue 或者 React 等框架。
- 如果需要，自主学习和探索采用 Google Blockly 二次开发实现可视化积木式编程。
- 后端采用 Spring Boot 框架，MyBatis 作数据库持久化层，数据库不限制，MySQL、MongoDB 或者图数据库如 Neo4j。
- 采用 WebRTC 支持学习者基于音视频交流的协同学习。可以选择搭建 TurnServer 或者 coturn 等服务器实现 WebRTC，推荐 coturn。
- 前端与后端服务程序均部署在云服务器上。

3.2 参考技术架构

为了实现高内聚、低耦合，本平台建议采用前后端分离的总体架构设计，前端采用 Angular 等 MVVM 框架和 node.js 编写单页面应用，后端采用 Spring Boot 进行数据服务，前后端之间通过基于 WebSocket 的 Socket.IO 框架，或者基于 RestfulWeb 服务和 JSON 数据格式进行交互。

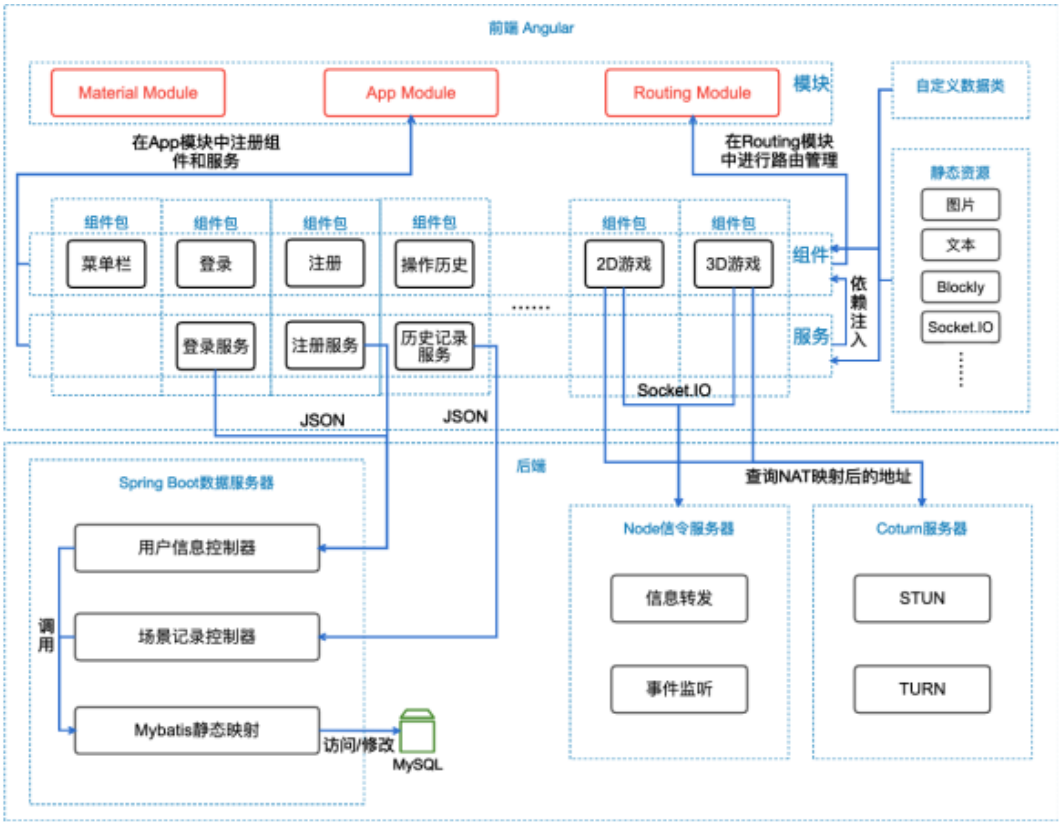
后端服务器主要有三个：基于 Spring Boot 的数据处理服务器，基于 Socket.IO 的服务器客户端双向通信服务器（信令服务器）和基于 Coturn 的 NAT 穿越服务器。

数据处理服务器采用 Mybatis 技术与 MySQL 数据库进行交互，利用 XML 文件进行 Mybatis 静态映射，包括游玩记录映射文件和用户信息映射文件。数据处理服务器提供了：登录服务和注册服务，由用户信息操作控制器控制；游玩记录服务，由场景记录控制器控制。由于本平台采用 token 机制用于用户登陆状态的保持，故数据处理服务器还拥有 JWT

加密模块。并且为了保证用户的隐私，存储用户账号信息时需要对密码进行加密存储，因此还具有密码加密模块。

信令服务器采用 Socket.IO 技术编写，利用 Socket.IO 已有的名字空间的房间功能，实现协同操作和视频聊天时的用户组管理。服务器实现了对用户加入房间、用户离开房间和用户发送消息等事件的监听和处理。

NAT 穿越服务器则是直接使用了技术成熟的 STUN 服务器和 TURN 服务器一体化的 Coturn 服务器，只需要对配置文件进行自定义配置即可部署在云服务器上运行。



3.3 参考资料

- Google Blockly 资料 ： https://developers.google.cn/blockly/
- Scratch Blocks 官方文档 ： https://scratch.mit.edu/developers
- Sphere Engine 官方文档 ： https://developer.sphere-engine.com/api/compilers?version=3
- Web Storage 等 W3C 标准文档 ： https://www.w3.org/TR/webstorage/

四、 评分细则

4.1 分数组成

- 基本功能分: 完成系统基本功能，如下面量表中所给出的基本功能项，满分 100 分。
- 进阶任务分:包括但不限于协同学习功能、结合 Web3D 的场景展示，AI 功能、云计算服务应用，以及其他具有创新性的功能等。最多 30 分。
- 根据小组分工及个人完成工作量，由小组成员间协调，给出贡献比例，该比例以相对方式提供，比如某位贡献度最大的同学是 100%的话，其他组员按照相对于他的得分给出比例，比如另外一位组员 90%表示贡献度最大同学 PJ 得分是 100 分的话，这位组员是 90 分。

4.2 评分量表

	功能项	评分指标	分值
基本功能	UI 和交互	UI 设计合理，具有较好的用户体验	5
	基本页面与流程	登录和注册功能	5
		新手导航页面功能	5
		个人学习信息页面，如学习历史记录等	5
		后端用户管理功能	5
	可视化学习场景以及交互	可交互的计算思维学习界面，如 Blockly 场景	10
		多关卡学习场景的设计和交互的丰富程度	20
		动画效果展示	15
	工程能力	文档说明清晰、详细，图文并茂，图示准确	10
		系统架构设计合理规范	5
		代码清晰，风格合理，具有良好的设计模式	10
		服务部署在云平台上，具有很好的可访问性	5
进阶功能	协同学习和交互性	采用 Socket.io 支持房间功能	5
		采用 WebRTC 等实现远程桌面控制和权限管理	5
		创新交互与多模式交流支持（如 WebRTC）	5
	用户体验	结合 Web3D 的场景展示	5
	AI 功能	调用 API 访问外部大模型作为学习中的 AI 教师	5
	云计算应用	合理采用 Docker 以及多种云服务	5
	其他	以上仅做参考，支持增强功能和用户体验的其他创新设计和开发，进阶功能不超过 30 分。	每项 5 分

4.3 评分点说明：

1）不得抄袭，否则得分为 0 并且后果自负！

2）每一项的分数取决于该项功能的完成度。完成度和可用性越好，分数越高。

3）项目完整度和易用性评价标准:

- A. 功能残缺，不能完整运行，有明显 bug。
- B. 完成规定的用户功能和操作，无明显瑕疵。
- C. 界面舒适，操作合理，响应迅速，鲁棒性强。

A、B、C 分别对应 分数的 0 - 30% ，30% - 70% 分，70% - 100% 分。

4）附加功能必须在文档中明确写出，概述该功能并描述实现原理。

5）项目设计文档需要至少包含:

- 项目组织以及其中每个文件的说明。
- 关键功能实现的细节。
- 服务器部署配置的详细介绍。

6）团队分工文档需要至少包含:

- 团队成员、分工、具体完成工作，列出每个人的贡献比例。
- 其他需要补充说明的问题，比如创新之处的思考。

五、 提交物

提交物包含以下三项:

- 源代码:推荐使用 Git 进行协作，提交到 GitHub 等 Git 托管平台上。
- 文档:推荐使用 Markdown 编写项目文档，与源代码一同提交到 Git 托管平台上。
- 可供访问的公网地址，以及系统的使用说明文档。