

EDU-TH-2 总结和反思

汇报人：贺梓源

小组成员：贺梓源 宋鸿堃 封静涵 裴子祎 谭立德 朱文涛

设计总结

- 设计核心：美观大方、简单易用
- 需求分析
 - 重点需求：交互学习
 - 动画流畅，用户友好
 - 一般需求：论坛、题库、管理
 - 满足需求前提下剔除不必要的功能
 - 其它需求
 - 移动端也能使用功能

设计总结

- 重点需求：交互学习
- 用户友好：使用高亮控制用户注意力

变量名	left	mid	right	array[mid]	target	circle times
变量值	0	-1	12	-1	13	0

```
def binarySearch(array, target):
    left = 0
    right = len(array) - 1

    while left <= right:
        mid = (left + right) // 2
        if array[mid] == target:
            return mid
        elif array[mid] < target:
            left = mid + 1
        else:
            right = mid - 1
    return -1

array = [1, 2, 3, 4, 5, 7, 8, 9, 10, 11, 12, 13, 14]
target = 13
result = binarySearch(array, target)
print(result)
```

设计总结

- 重点需求：交互学习
- 用户友好：所见即所得，可视化调试体验

变量名	left	mid	right	array[mid]	target	circle times
变量值	7	9	12	11	13	2

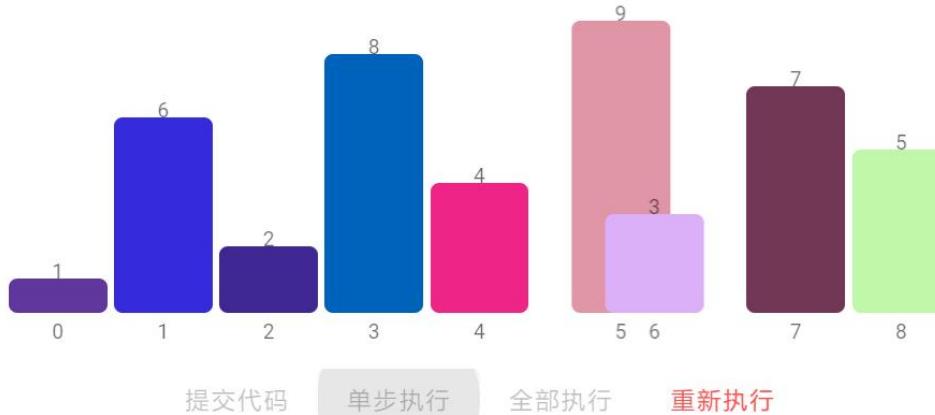
```
def binarySearch(array, target):
    left = 0
    right = len(array) - 1

    while [left <= right]:
        mid = (left + right) // 2
        if array[mid] == target:
            return mid
        elif array[mid] < target:
            left = mid + 1
        else:
            right = mid - 1
    return -1

array = [1, 2, 3, 4, 5, 7, 8, 9, 10, 11, 12, 13, 14]
target = 13
result = binarySearch(array, target)
print(result)
```

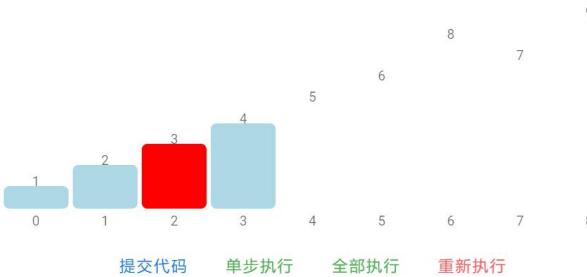
设计总结

- 重点需求：交互学习
 - 美观易用：动态交换动画；防误触



设计总结

- 重点需求：交互学习
- 用户友好：递归函数理解较难，屏蔽其他区域



选择排序方式
快速排序

```
def quickSort(array, left, right):
    if left < right:
        pivot = partition(array, left, right)
        quickSort(array, left, pivot - 1)
        quickSort(array, pivot + 1, right)
    return

def partition(array, left, right):
    pivot = array[right]
    i = left - 1
    for j in range(left, right):
        if array[j] <= pivot :
            i += 1
            array[i], array[j] = array[j], array[i]
    array[i + 1], array[right] = array[right], array[i + 1]
    return i + 1

array = [8, 1, 6, 2, 9, 4, 3, 7, 5]
quickSort(array,0,len(array) - 1)
print(array)
```

4

设计总结

- 重点需求：交互学习
- 用户友好：变长输入框

```
def selectionSort(array):  
    length = len(array)  
    for i in range(length):  
        swapIndex = i  
        for j in range(i + 1, length):  
            if array[j] < array[swapIndex] :  
                swapIndex = j  
        array[i], array[swapIndex] = array[swapIndex], array[i]  
    return array  
array = [8, 1, 6, 2, 9, 4, 3, 7, 5, 666, 66666666666666666666]  
selectionSort(array)  
print("Sorted array:", array)
```

设计总结

- 重点需求：交互学习
- 美观易用：动态过渡动画，节点、边颜色渐变

编辑图边

提交代码

单步运行

全部执行

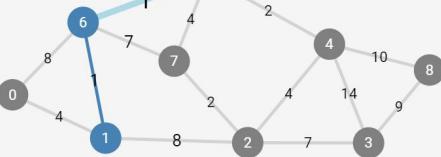
重新执行

选择图算法

```
# color表示顶点状态, gray表示未发现, lightblue表示正在处理,  
# pred表示顶点u由pred[u]发现  
# d表示各顶点的发现时刻  
# f表示各顶点的处理完成时刻  
def dfs(start):  
    time = 0  
    dfsVisit(start) # 先对起始节点进行DFS  
    for u in V:  
        if color[u] == 'gray': # 保证遍历完全  
            dfsVisit(u)  
    return  
def dfsVisit(u):  
    color[u] = 'lightblue'  
    time += 1  
    d[u] = time # 发现时刻  
    for v in G.Adj[u]: # u所临接的边  
        if color[v] == 'gray':  
            pred[v] = u  
            dfsVisit(v)  
    color[u] = 'steelblue'  
    time += 1  
    f[u] = time # 完成时刻  
    return  
# Graph[V,E]  
dfs(0)
```

设计总结

- 重点需求：交互学习
- 用户友好：动态图展示，鼠标悬停放大 (PC)



编辑图边 提交代码 单步运行 全部执行 重新执行

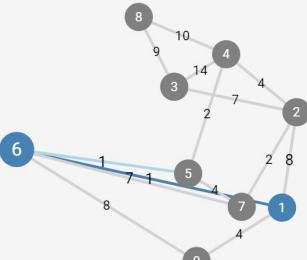
选择图算法

```
# 采用并查集判断和维护所选边的顶点是否在一棵子树
# find(x)寻找顶点x的树根结点
# union(x,y)将2顶点所在树结构合并，合并成功返回True
def kruskal():
    parent = [v for v in V] # 所有顶点的树根节点
    rank = [0] * len(V) # 并查集辅助数组
    # e的结构{'source':'', 'target':'', 'weight':''}
    edges = [e for e in E]
    edges.sort(key=lambda x: x['weight']) # 边排序
    mst = [] # 存放着选取的所有边
    weights = 0 # 最小生成树总权值
    for e in edges: # 每次加入不形成环路的最小权值边
        if union(e['source'], e['target']):
            mst.append(e)
            weights += e['weight']
    return weights, mst
# Graph[V,E]
kruskal()
```

设计总结

- 重点需求：交互学习
- 用户友好：动态图展示，可拖动

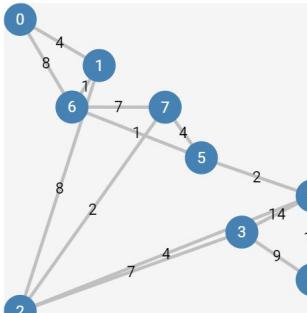
选择图算法



```
# 采用并查集判断和维护所选边的顶点是否在一棵树上
# find(x) 寻找顶点x的树根结点
# union(x,y)将2顶点所在树结构合并, 合并成功返回True
def kruskal():
    parent = [v for v in V] # 所有顶点的树根节点
    rank = [0] * len(V) # 并查集辅助数组
    # e的结构{'source':'','target':'','weight':''}
    edges = [e for e in E]
    edges.sort(key=lambda x: x['weight']) # 边排序
    mst = [] # 存放着选取的所有边
    weights = 0 # 最小生成树总权值
    for e in edges: # 每次加入不形成环路的最小权值边
        if union(e['source'], e['target']):
            mst.append(e)
            weights += e['weight']
    return weights, mst
# Graph[V,E]
kruskal()
```

编辑图边 提交代码 单步运行 全部执行 重新执行

EduCodingSpace - 图... 下午1:0... ②



编辑图边 提交代码
单步运行 全部执行
重新执行

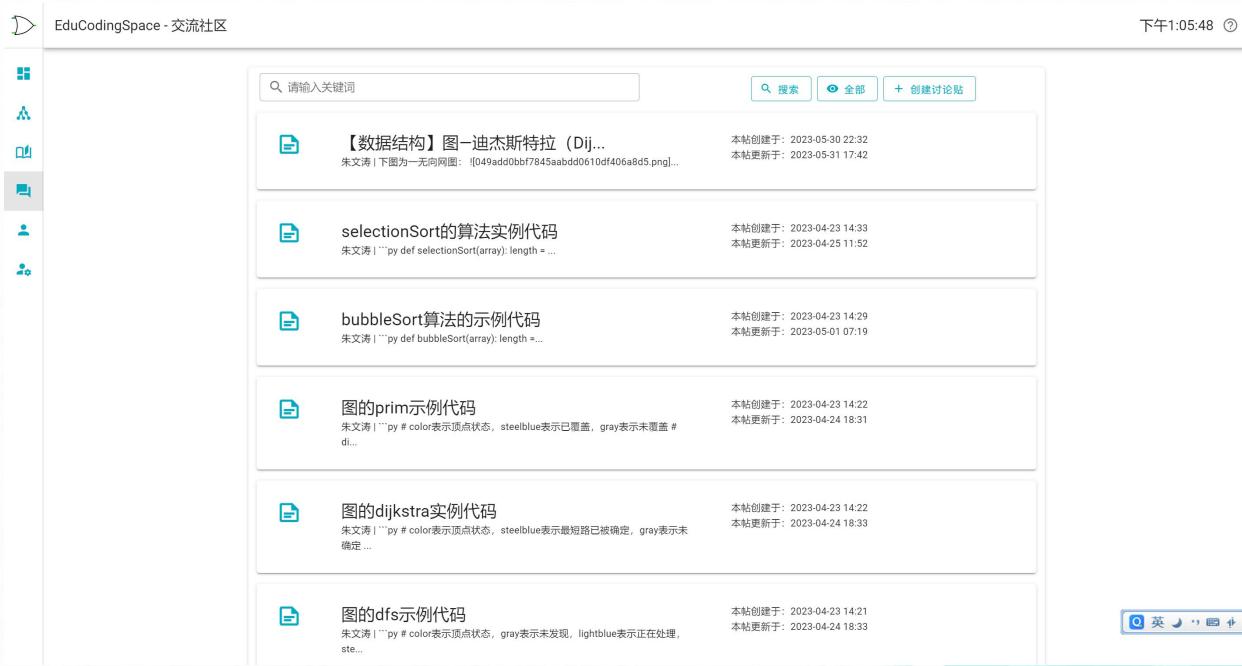
选择图算法

```
# 采用并查集判断和维护所选边的顶点是否在一棵树上
# find(x) 寻找顶点x的树根结点
# union(x,y)将2顶点所在树结构合并, 合并成功返回True
def kruskal():
    parent = [v for v in V] # 所有顶点的树根节点
    rank = [0] * len(V) # 并查集辅助数组
    # e的结构{'source':'','target':'','weight':''}
    edges = [e for e in E]
    edges.sort(key=lambda x: x['weight']) # 边排序
    mst = [] # 存放着选取的所有边
    weights = 0 # 最小生成树总权值
    for e in edges: # 每次加入不形成环路的最小权值边
        if union(e['source'], e['target']):
            mst.append(e)
            weights += e['weight']
    return weights, mst
# Graph[V,E]
kruskal()
```

主页 交互学习 习题社区 交流社区 个人中心

设计总结

- 一般需求：交流社区
- 简单易用，基本不用看用户手册；界面简约大方



The screenshot shows a web interface for a community platform. At the top, there is a header with a search bar containing '请输入关键词' (Please enter keywords), a search button, a '全部' (All) button, and a '创建讨论贴' (Create discussion post) button. The header also displays the time '下午1:05:48' and a refresh icon. On the left, there is a vertical sidebar with various icons: a magnifying glass, a person, a group, a gear, a chart, a person with a plus sign, a person with a minus sign, and a person with a question mark. The main content area lists six posts in a grid format:

帖子标题	创建时间	更新时间
【数据结构】图一迪杰斯特拉 (Dij... 朱文涛 下面为一无向网图: !049add0bbf7845abdd0610df406a8d5.png...	2023-05-30 22:32	2023-05-31 17:42
selectionSort的算法实例代码 朱文涛 ``py def selectionSort(array): length = ...	2023-04-23 14:33	2023-04-25 11:52
bubbleSort算法的示例代码 朱文涛 ``py def bubbleSort(array): length = ...	2023-04-23 14:29	2023-05-01 07:19
图的prim示例代码 朱文涛 ``py # color表示顶点状态, steelblue表示已覆盖, gray表示未覆盖 # di...	2023-04-23 14:22	2023-04-24 18:31
图的dijkstra实例代码 朱文涛 ``py # color表示顶点状态, steelblue表示最短路已被确定, gray表示未确定 ...	2023-04-23 14:22	2023-04-24 18:33
图的dfs示例代码 朱文涛 ``py # color表示顶点状态, gray表示未发现, lightblue表示正在处理, ste...	2023-04-23 14:21	2023-04-24 18:33

At the bottom right of the main content area, there is a small toolbar with icons for English, night mode, and other settings.

设计总结

- 一般需求：交流社区

- 简单易用，基本不用看用户手册；界面简约大方



EduCodingSpace - 交流社区

下午1:06:52 ②

selectionSort的算法实例代码

朱文涛

创建于: 2023-04-23 14:33

```
def selectionSort(array):
    length = len(array)
    for i in range(length):
        swapIndex = i
        for j in range(i + 1, length):
            if array[j] < array[swapIndex] :
                swapIndex = j
        array[i], array[swapIndex] = array[swapIndex], array[i]

array = [8, 1, 6, 2, 9, 4, 3, 7, 5]
selectionSort(array)
print("Sorted array:", array)
```

最近一次修改于: 2023-04-25 11:52

回复 编辑 删除

英

设计总结

- 一般需求：题库
- 简单易用

EduCodingSpace - 习题详情

下午1:09:33 ②

冒泡排序的复杂度分析

题面
对于长度为n的无序整数列表，使用**冒泡排序算法**进行排序的最坏时间复杂度是多少？

提示

- 冒泡排序算法的核心思想是通过不断比较相邻的元素，将较大的元素向后移动，较小的元素向前移动，从而实现排序的目的。

伪代码
这里提供一份**冒泡排序**的伪代码以供参考。

```
def bubbleSort(array):
    length = len(array)
    for i in range(length - 1):
        for j in range(length - i - 1):
            if array[j] > array[j + 1]:
                array[j], array[j + 1] = array[j + 1], array[j]

array = [8, 1, 6, 2, 9, 4, 3, 7, 5]
bubbleSort(array)
print("Sorted array:", array)
```

冒泡排序的复杂度分析

O(n^2)
 O($n\log n$)
 O(n)
 O($\log n$)

提交

英 月 四 书

设计总结

- 一般需求：管理端
- 简单易用

EduCodingSpace - 管理员用户管理

用户编码	学号	用户名	用户权限	操作
1	20373864	Tan	管理员	编辑 重置密码 删除
4	20373782	封静涵	管理员	编辑 重置密码 删除
10	20373934	贺梓源	管理员	编辑 重置密码 删除
11	20373918	宋鸿翌	管理员	编辑 重置密码 删除
15	BY6666666	博士	助教	编辑 重置密码 删除
16	2023SE	黎昊轩	教师	编辑 重置密码 删除
21	20373781	封静涵	管理员	编辑 重置密码 删除
27	20373915	朱文涛	管理员	编辑 重置密码 删除
28	jsA	教师A	教师	编辑 重置密码 删除
29	guest	体验账号	学生	编辑 重置密码 删除

Rows per page: 10 1-10 of 21

设计总结

EduCodingSpace - 仪... 下午1:11:... ②

● 其它需求：移动端适配

- 根据设备宽度渲染，仍有提升空间

欢迎，在使用平台过程中，遇到问题可以及时向教师反馈，祝你学习顺利！

为你推荐的习题

Sort by		
题目ID	28	
题目名	binarySearch的算法实例	
标签	分治 模拟	
题目ID	24	
题目名	dijkstra的算法实例	
标签	图 模拟	
题目ID	53	
题目名	Python语言的种类	
标签	基础知识	
题目ID	27	
学习	交互学习	习题社区
发布	发布社区	交流社区

设计总结

- 其它需求：移动端适配
- 根据设备宽度渲染，仍有提升空间

EduCodingSpace - 排序

下午1:12:42 ②

选择排序方式

提交代码 单步执行 全部执行 重新执行

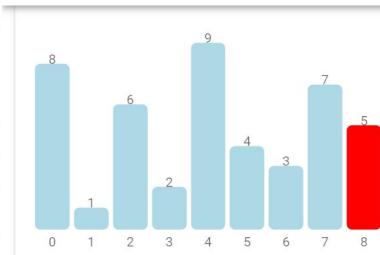
```
def quickSort(array, left, right):
    if left < right:
        pivot = partition(array, left, right)
        quickSort(array, left, pivot - 1)
        quickSort(array, pivot + 1, right)
    return

def partition(array, left, right):
    pivot = array[right]
    i = left - 1
    for j in range(left, right):
        if array[j] <= pivot:
            i += 1
            array[i], array[j] = array[j], array[i]
    array[i + 1], array[right] = array[right], array[i + 1]
    return i + 1

array = [8, 1, 6, 2, 9, 4, 3, 7, 5]
quickSort(array, 0, len(array) - 1)
print(array)
```

English 中文

EduCodingSpace - 排... 下午1:12:... ②



提交代码 单步执行 全部执行
重新执行

选择排序方式

```
def quickSort(array, left, right):
    if left < right:
        pivot = partition(array, left, right)
        quickSort(array, left, pivot - 1)
        quickSort(array, pivot + 1, right)
    return
```

```
def partition(array, left, right):
    pivot = array[right]
    i = left - 1
    for j in range(left, right):
        if array[j] <= pivot:
            i += 1
            array[i], array[j] = array[j], array[i]
```

人员分工

- 工作内容尽量解耦
- 取长补短
 - 文笔优美
 - 工程能力强

姓名	分工
贺梓源	主要负责前端开发及审核、Docker构建与CI/CD，兼任PM、运维、文档撰写与审核
宋鸿堃	主要负责交互学习开发及审核、前端开发及审核，兼任脚本审核
谭立德	主要负责后端开发及审核、数据库管理，兼任运维
朱文涛	主要负责脚本开发，兼任文档审核、数据录入、视频录制、机动任务
封静涵	主要负责文档撰写及审核，兼任前端开发
裴子祎	主要负责文档撰写及审核，兼任前端开发

经验和反思

- 代码/文档管理

- GitLab

- 任务分配：议题看板
 - 代码提交及审核：合并请求

- 版本控制问题

- 使用版本控制工具不熟练
 - 交互学习脚本
 - 导致很多不必要的变基任务

C CodingEduSpace 

群组 ID: 64978707 

[子组和项目](#) [共享项目](#) [归档项目](#)

 C	client-frontend 
 D	docs 
 E	eduspace-backend 

经验和反思

- 代码/文档管理

- GitLab

- 任务分配：议题看板
 - 代码提交及审核：合并请求

- 版本控制问题

- 使用版本控制工具不熟练
 - 交互学习脚本
 - 导致很多不必要的变基任务

The screenshot shows the interface of a GitLab group named 'CodingEduSpace'. The group ID is 64978707. The interface includes tabs for '子组和项目' (Subgroups and Projects), '共享项目' (Shared Projects), and '归档项目' (Archived Projects). Under '子组和项目', there are three projects listed: 'client-frontend', 'docs', and 'eduspace-backend'. Each project has a lock icon, indicating they are private.

项目	状态
client-frontend	私有
docs	私有
eduspace-backend	私有

经验和反思

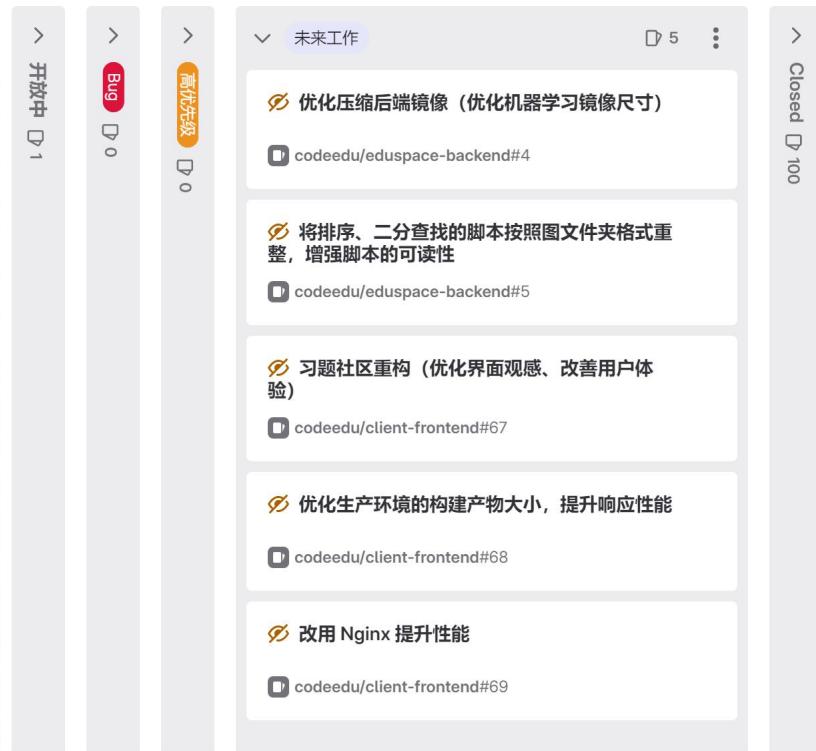
● 代码/文档管理

● GitLab

- 任务分配：议题看板
- 代码提交及审核：合并请求

● 版本控制问题

- 使用版本控制工具不熟练
 - 交互学习脚本
 - 导致很多不必要的变基任务



经验和反思

- 代码/文档管理

- GitLab

- 任务分配：议题看板
 - 代码提交及审核：合并请求

- 版本控制问题

- 使用版本控制工具不熟练
 - 交互学习脚本
 - 导致很多不必要的变基任务

GitLab screenshot showing the 'Merge Requests' page for the 'CodingEduSpace' repository. The page displays a list of merge requests, with the first few being merged recently. The sidebar on the left includes 'Group Information', 'Issues' (6), 'Merge Requests' (3), 'Security', 'CI/CD', 'Software Packages & Libraries', and 'Settings'.

状态	标题	操作	更新日期
已合并	设计文档前半部分终审 - zwt	已合并	更新于2小时前
已合并	管理端批量新建用户bug修复	已合并	更新于1星期前
已合并	Shk optimize question	已合并	更新于1星期前
已合并	Shk optimize forum	已合并	更新于1星期前
已合并	修复登录输入错误密码后清空密码栏	已合并	更新于1星期前
已合并	修复登录输入错误密码后清空密码栏	已合并	更新于1星期前
已合并	添加容器错误提示	已合并	更新于1星期前

经验和反思

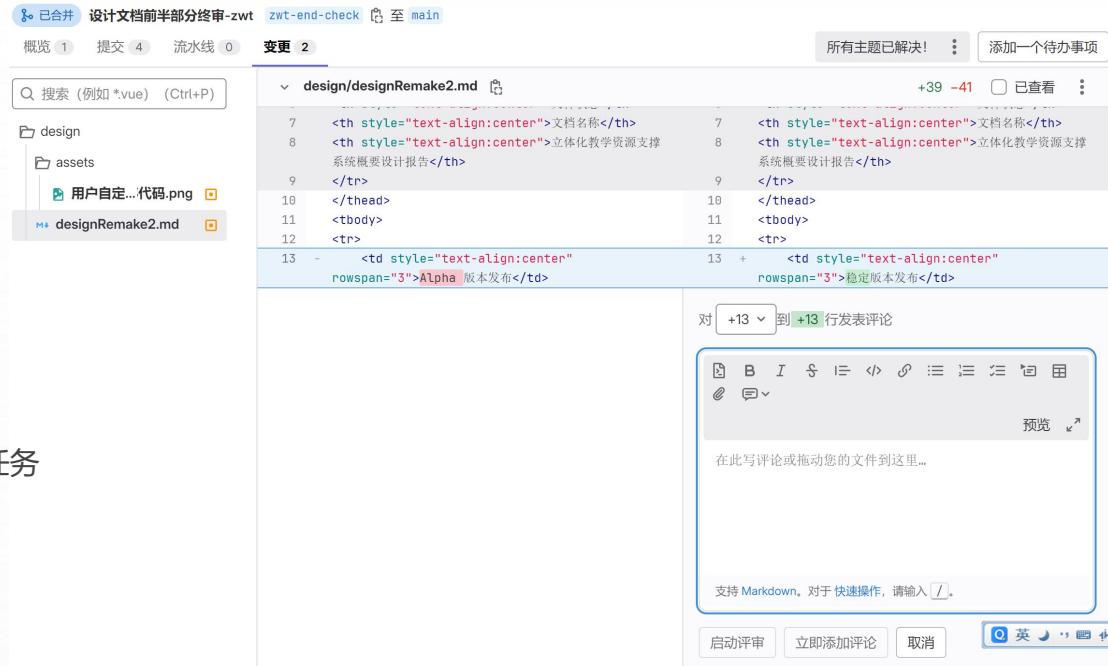
● 代码/文档管理

● GitLab

- 任务分配：议题看板
- 代码提交及审核：合并请求

● 版本控制问题

- 使用版本控制工具不熟练
 - 交互学习脚本
 - 导致很多不必要的变基任务



设计文档前半部分终审-zwt zwt-end-check 至 main

概览 1 提交 4 流水线 0 变更 2

所有主题已解决! 添加一个待办事项

design/designRemake2.md

+39 -41 已查看

7 <th style="text-align:center">文档名称</th> 7 <th style="text-align:center">文档名称</th>
8 <th style="text-align:center">立体化教学资源支撑 8 <th style="text-align:center">立体化教学资源支撑
系统概要设计报告</th> 系统概要设计报告</th>
9 </tr> 9 </tr>
10 </thead> 10 </thead>
11 <tbody> 11 <tbody>
12 <tr> 12 <tr>
13 - <td style="text-align:center" rowspan="3">Alpha 版本发布</td> 13 + <td style="text-align:center" rowspan="3">稳定版本发布</td>

对 +13 到 +13 行发表评论

在此写评论或拖动您的文件到这里...

启动评审 立即添加评论 取消

经验和反思

- 代码 CI/CD
 - 代码风格检查
 - 合并请求
- 代码自动部署
 - 主要分支

已通过

00:01:13
1星期前

管理端批量新建用户bug修复

#884604679 ➝ production -o ea4e237b 🐛

最新



已通过

00:01:28
1星期前

管理端批量新建用户bug修复

#884602881 🐛 165 -o ea4e237b 🐛

最新 合并请求



已通过

00:00:24
1星期前

管理端批量新建用户bug修复

#884602615 ➝ develop -o ea4e237b 🐛

最新



已通过

00:01:22
1星期前

管理端批量新建用户bug修复

#884514792 🐛 164 -o ea4e237b 🐛

最新 合并请求



经验和反思

- 小组协作
 - 组会
 - 进度安全时避免线下会议
 - 对进度控制要求较高
 - 组员积极性高，按时完成任务
 - 沟通问题
 - 分工时对组件、界面的规格不明确
 - 分工粒度问题：单个任务太大
 - 每个组员的空闲时间、能联系上的时间不同

经验和反思

- 小组协作
- 进度控制问题
 - Alpha 前习题社区
 - 分工时单任务规模太大，交流不及时
 - Alpha 前设计文档
 - 忙于其它任务，未及时关注进度
 - 测试
 - “边开发边测试” 做得不好
 - 开发时测试过弱

收获和感想

- 封静涵：

- 在软件工程的工作中，我负责前端部分的工作，这是我第一次正式地参与团队合作项目。虽然这个过程中也遇到过种种问题，但最终都通过各种方法解决。我认为这是一次成功的团队合作，我从小组成员那里学习到了很多，也帮助我积累了团队合作的经验。对我来说，这是一次难忘的经历。

- 裴子祎：

- 总结：在这个软件工程的工作中，我主要负责前端开发和主要文档撰写工作。通过这个过程，我进一步锻炼了前端设计开发能力，并认识到代码维护和可扩展性的重要性。在团队合作中，我学会了如何有效地沟通和协作，并更好地理解了任务工作量和时间安排规划的重要性，以及文档撰写整理的任务量和意义。同时，我也从小组成员中学习到了很多知识和经验。这个项目让我在多个方面得到了锻炼和收获，并为我的未来职业发展打下了坚实的基础。

收获和感想

- 朱文涛：

- 在软件工程的工作中，我负责后端脚本和审核测试。通过这个过程，我学到了如何编写高质量、可维护、可扩展的代码，以及如何更好地与团队成员进行沟通和合作。我也认识到了软件工程实践对于提高代码质量和减少错误的重要性。这些知识和技能将帮助我更好地应对未来的挑战和机会。

- 宋鸿堃：

- 我在项目中的主要职责是设计交互式学习的实现方案以及前端开发。我采用了代码填空加执行真实代码的设计，提供了友好的用户体验和辅助学习效果。在前端开发方面，我对算法添加了动画效果并实现了高质量的图算法的展示。并且实现动画自动播放和并修复随之而来线程同步bug。我的设计与开发满足了用户交互式学习的需求，具有一定的实用价值。

- 谭立德：

- 久练则成



Thanks!