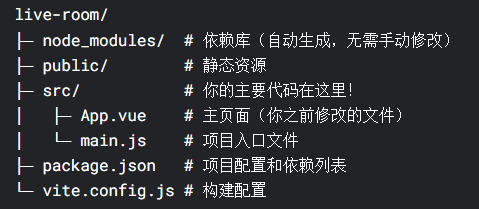
项目于分为前端Vue项目和后端PHP文件

前端Vue项目，通过Powershell，输入cd live-room进入

项目结构：



后端PHP文件，处于Laragon的根目录中





第1天：环境搭建

安装Node.js 20+（前端需要）

安装Laragon（包含PHP/MySQL/Nginx）

创建Vue3项目：npm create vite@latest live-room --template vue

测试环境是否正常

第2天：基础播放功能

前端：

安装video.js：npm install video.js

创建播放器组件（示例代码）：

vue

<template>

<video id="my-video" controls class="video-js">

<source :src="videoUrl" type="video/mp4">

</video>

</template>

<script setup>

import videojs from 'video.js'

import { ref, onMounted } from 'vue'

const videoUrl = ref('') // 从后端获取的地址

onMounted(() => {

videojs('my-video')

})

</script>

后端（PHP）：

创建简单API返回视频地址（示例）：

php

<?php

header('Content-Type: application/json');

echo json\_encode([

'code' => 0,

'msg' => 'success',

'data' => [

'url' => 'https://example.com/sample.mp4' // 替换为实际视频地址

]

]);

第3天：对接前后端

前端调用API：

vue

<script setup>

import { onMounted } from 'vue'

onMounted(async () => {

const res = await fetch('http://localhost/api.php')

const data = await res.json()

if (data.code === 0) {

videoUrl.value = data.data.url

}

})

</script>

测试完整流程

第4天：异常处理与UI优化

添加加载状态和错误提示

简单模仿参考链接的UI布局

第5天：部署测试

将前端构建（npm run build）

部署到Laragon的www目录

完整测试