

**世界技能大赛网站设计与开发**

**指导用书**

Zhonghui Yunqi Technologies Group Co.,Ltd.

Copyright © All rights reserved by Zhonghui Yunqi Technologies Group

目录

[赛项概况 4](#_Toc1703400062)

[世界技能大赛简介 4](#_Toc890997714)

[网站设计与开发赛项简介 4](#_Toc1235121184)

[网站设计与开发赛项对选手的技能要求 4](#_Toc649049071)

[网页设计与实现 10](#_Toc200291836)

[任务一 页面版式设计 11](#_Toc907725577)

[任务二 页面版式设计 15](#_Toc1270798160)

[任务三 实现页面布局 18](#_Toc1545039432)

[任务四 制作导航栏目 27](#_Toc272137940)

[任务五 制作主图栏目 32](#_Toc1005363711)

[任务六 实现轮播图 38](#_Toc1604523737)

[移动网页设计与实现 47](#_Toc730392666)

[任务一 移动端网页设计一 47](#_Toc1114434386)

[任务二 移动端网页设计二 49](#_Toc97017910)

[任务三 移动端网页实现一 53](#_Toc227452074)

[任务四 移动端网页实现二 54](#_Toc451651874)

[JavaScript脚本制作轮播图（24年赛题） 55](#_Toc53026372)

[任务一 初始化Photos Slideshow项目 56](#_Toc418485461)

[任务二 实现 Photos Slideshow游戏内容界面 58](#_Toc1671896403)

[任务三 实现Photos Slideshow 切换操作模式功能 67](#_Toc1447488311)

[任务四 实现 Photos Slideshow 切换主题功能 70](#_Toc1581757951)

[任务五 实现Photos Slideshow图片上传和照片排序功能 73](#_Toc2002598967)

[任务六 实现Photos Slideshow主要功能 85](#_Toc615036531)

[后端应用程序开发 96](#_Toc1579447538)

[任务一 设计数据库表结构 106](#_Toc521714693)

[任务二 实现用户注册登录鉴权功能 122](#_Toc109548527)

[任务四 用户端API接口开发 123](#_Toc870635674)

[任务五 管理端接口开发 124](#_Toc1610316936)

[前端应用程序开发 126](#_Toc1795340402)

[任务一 搭建前端工程 126](#_Toc487988923)

[任务二 游戏首页面实现 127](#_Toc12096140)

[任务三 用户注册登录页面实现 128](#_Toc1351256816)

[任务四 游戏排行榜页面实现 129](#_Toc1378986638)

[任务五 个人中心页面实现 130](#_Toc1247217325)

[任务六 管理端页面实现 131](#_Toc2000305887)

[竞速赛介绍 133](#_Toc1579278474)

[竞速赛主要知识点介绍 133](#_Toc7459254)

[WSC2024-竞速赛A13：滴⽔（中等） 133](#_Toc1123620400)

[WSC2024-竞速赛A22：CSS 条形图（简单） 138](#_Toc976834258)

[WSC2024-竞速赛B35：分形三角形（简单） 143](#_Toc279597194)

[WSC2024-竞速赛C11：图像拼接（简单） 147](#_Toc2128984111)

[WSC2022SE-竞速赛D3 井字游戏（困难） 153](#_Toc433874347)

[竞速赛赛题和规模介绍 166](#_Toc1009989860)

[附录 168](#_Toc1095934849)

[安装网站设计工具 169](#_Toc530892258)

[安装PHP开发工具 170](#_Toc1237441934)

[安装nodejs开发工具 174](#_Toc1547586723)

[安装前端开发工具 176](#_Toc583918630)

[安装MySQL数据库 178](#_Toc1655927395)

**项目一**

**世界技能大赛网站设计与开发**

**赛项概况**

**世界技能大赛简介**

**网站设计与开发赛项简介**

在网络日益成为现代平台核心的世界中，使用网络技术是一项关键技能。它也是所有技能中最复杂和最多样化的技能之一。

网页设计师和开发人员首先会与客户建立专业关系，深入了解客户对网站的需求。强大的设计和沟通能力，以及对目标受众、市场和趋势的了解都是必不可少的。在开发过程中，网页设计师和开发人员会创建数据库、构建程序以及测试和调试网站。

**网站设计与开发赛项对选手的技能要求**

**工作组织与管理**

个人需要知道并理解：

* 实现高效团队合作的原则和实践
* 计算机系统的原理和行为
* 有助于可持续产品、战略和实践的系统方面
* 如何展现主动性并运用创造性解决问题
* 如何识别、分析和评估来自各种来源的信息
* 如何确定问题的多种解决方案，并根据时间和其他限制提供这些解决方案
* 如何优化代码以进行部署

个人应能够：

* 解决常见的网页设计和开发问题
* 在规定的时间限制和期限内完成工作
* 使用装有各种软件包的计算机
* 运用研究技术和技能来跟上行业最佳实践
* 采用行业最佳实践进行部署优化，例如页面加载
* 确保工作按照既定时间表完成
* 归档时包含链接图像、字体、原生文件和生产文件格式
* 使用 git 等软件版本控制系统

**沟通和人际交往能力**

个人需要知道并理解：

* 如何识别问题、研究、分析并提出解决方案
* 如何制作原型并编写用户和功能测试
* 如何通过创建线框和流程图来表达设计概念
* 软件设计概念和技术，包括流程图和 ER 图

个人应能够：

* 阅读并理解规范文档
* 阅读并使用提供的前端和后端技术源代码
* 提供符合客户要求和规格的产品
* 收集、分析和评估信息
* 解释标准和要求
* 符合客户要求
* 提出概念以满足业务需求

**设计实施**

个人需要知道并理解：

* 如何遵循设计原则和模式来制作美观、有创意且易于访问的界面
* 与设计的认知、社会、文化、可访问性、技术和经济背景相关的问题
* 各种图形格式以及何时使用适当的格式来实现设计
* 不同的目标市场以及满足每个市场的设计元素
* 维护企业形象、品牌和风格指南的协议
* 互联网设备和屏幕分辨率的限制
* HTML 和 CSS 的万维网联盟 (W3C) 标准
* 可用性和交互设计
* 跨浏览器兼容性
* 多设备兼容性
* 搜索引擎优化 (SEO) 和性能优化
* 如何在需要的地方嵌入和集成动画、音频和视频
* 为有特殊需求的用户提供无障碍访问和通信服务，包括万维网联盟 (W3C) 的 WCAG 标准

个人应能够：

* 创建、分析和开发对沟通问题的视觉反应，包括理解层次结构、排版、美学和构图
* 创建、处理和优化互联网图像
* 确定目标市场并创建设计概念
* 实现在多种屏幕分辨率和/或设备上正常运行的响应式设计
* 批评草稿概念、颜色和排版选择
* 创建线框、交互式原型以及考虑用户体验的用户界面设计
* 创建符合并验证 W3C 标准（包括可访问性指南）的代码
* 为各种设备和屏幕分辨率创建可访问且可用的 Web 界面
* 使用 CSS 或其他外部文件修改 Web 界面的外观
* 使用 CSS 预处理器/后处理器
* 创建和更新 Web 界面以获得适当的用户体验并协助提高搜索引擎性能
* 使用 CSS 开发动画和交互式用户界面

**前端开发**

个人需要知道并理解：

* ECMAScript（JavaScript）
* 如何使用 JavaScript 集成库、框架和其他系统或功能
* 如何使用 JavaScript 预/后处理器和任务运行工作流
* 编码最佳实践，包括故障排除、单元测试和测试驱动开发
* 应用程序部署的最佳实践
* Linux 命令行界面

个人应能够：

* 创建网站动画和功能以协助解释上下文并增加视觉吸引力
* 创建和更新 JavaScript 代码以增强网站的功能、可用性和美观性
* 使用 JavaScript 处理数据和自定义媒体
* 创建模块化且可重用的 JavaScript 代码
* 编写文档，包括代码中的注释
* 使用开源 JavaScript 库
* 使用 JavaScript 操作图形元素和内容元素
* 编写测试用例并执行 JavaScript 实现的自动化测试
* 处理代码中的错误，调试代码并修复错误
* 使用 Linux CLI 与远程服务器交互，在远程服务器上部署应用程序
* 按要求记录应用程序的部署步骤

**后端开发**

个人需要知道并理解：

* 面向对象编程
* PHP 和/或 NodeJS
* 开源服务器端库和框架
* 如何通过 SSH（包括 sFTP）连接到服务器
* 文件系统权限和所有权
* 如何设计和实现数据库
* 如何管理服务器和客户端系统之间的数据交换
* 软件设计模式包括 MVC（模型视图控制器）
* Web 应用程序安全
* 编码最佳实践，包括故障排除、单元测试和测试驱动开发
* 应用程序部署的最佳实践
* Linux 命令行界面

个人应能够：

* 利用编程技能处理数据
* 防范安全漏洞
* 使用 API（应用程序编程接口）、库和框架与现有代码集成
* 创建或维护数据库表以支持系统要求，考虑数据规范化、关键约束和正确的数据类型
* 创建模块化且可重用的代码
* 在代码中编写文档和注释
* 编写测试用例并执行后端实现的自动化测试
* 处理代码中的错误，调试代码并修复错误
* 使用 Linux CLI 与远程服务器交互，在远程服务器上部署应用程序
* 按要求记录应用程序的部署步骤
* 为给定的应用程序配置 Web 服务器

**项目二**

**世界技能大赛典型网站设计与开发**

**网页设计与实现**

**项目需求**

在这个模块中，我们设计并创建了一个营销页面来推广游戏平台。该页面的标题是“下一代网络游戏平台”，副标题是“对开发者、广告商和玩家的双赢”。

交付项目：

1.桌面级单页面实现营销网页

2.移动版本的单页面实现营销网页

3.一种展示用来在社交网络服务上发布的网页的设计。

**任务一** **页面版式设计**

1. 学习目标

➢ 掌握网站配色。

➢ 熟悉色彩心理。

1. 开发任务

1）根据网站主题选择合适颜色

1. 开发思路

1） 在选择颜色之前，首先要明确网站的核心功能和品牌定位。例如，游戏平台通常希望传递科技感、创新感和娱乐氛围，品牌定位将直接影响颜色的选择。

**2） 分析目标受众：**游戏平台的目标用户包括开发者、广告商和玩家，开发者更倾向于专业、科技感的颜色，而玩家更偏向活泼、吸引眼球的颜色组合。

**3） 主色调的选择：**主色调是页面的核心颜色，贯穿整个设计。根据平台的定位和用户心理，选择适合的主色调。例如，蓝色代表信任和科技感，红色代表激情和活力。

**4） 辅助色调的选择：**辅助色调通过与主色调形成对比，突出页面的重点元素。例如，蓝色可以搭配白色、灰色以营造简洁感，而加入少量的橙色或红色则可以吸引用户注意。

**5） 色彩心理学应用：**色彩与用户情感的关联：根据色彩心理学，选择能够引发特定情感反应的颜色。例如：

**蓝色：信任、专业、科技感**

蓝色常与平静和信任感联系紧密，能给人带来安全和可靠的感觉。它在科技、金融等领域中非常常见，因为它能够传达出冷静、理性和专业性。此外，蓝色也能帮助网站给用户留下严谨、现代的印象，特别适合技术性和专业性较强的平台。

**红色：激情、紧迫感、活力**

红色是一种具有强烈视觉冲击力的颜色，能够迅速吸引用户的注意力，常用于引发情绪的场合。它传递激情和活力，也可以营造紧迫感，适用于促销活动或需要即时行动的网站内容。在设计中，红色可以有效激发用户的积极响应和参与感。

**绿色：健康、自然、平衡**

绿色是自然界的象征，与环保、健康紧密相连。它给人以宁静、放松的感觉，并能传达出和谐与平衡的意象。绿色的使用通常适合健康、环保和生活类的网站，因为它能让用户感受到一种积极、友好的氛围。

**黄色：乐观、温暖、警示**

黄色是一种明亮、充满活力的颜色，容易让人联想到阳光和温暖，因此能够提升用户的积极情绪。它可以用于传达乐观和快乐的感觉，适合活泼、创新类的网站。不过，黄色也常用于警示类信息，应适当使用以避免视觉疲劳。

**紫色：高贵、创造力、神秘感**

紫色历史上常与皇室和贵族联系在一起，因此它能够传递出奢华、神秘的氛围。紫色也常常被用来激发创造力，适合艺术性和创造性内容的网站。它能帮助网站塑造出与众不同、富有艺术感的形象。

**黑色：权威、力量、优雅**

黑色往往带有权威和力量的象征，在奢侈品和时尚领域中尤为常见。它能够传达出一种高贵、庄重的气质，同时也带有一定的神秘感和现代感。黑色在网站设计中通常作为背景色使用，以凸显其他元素的视觉效果。

**白色：纯净、简洁、透明**

白色常用于传递纯洁、简洁和透明的感觉，是简约设计的首选颜色。它能帮助界面保持清新感，并提升内容的可读性。白色在科技类、医疗类和设计类网站中非常常见，因为它能够让用户感受到高效、清晰和整洁。

**橙色：活力、热情、创造力**

橙色是一种充满活力的颜色，能激发人们的行动欲望和互动意愿。它介于红色的热情与黄色的明快之间，具有吸引人的温暖与亲和力，适合用于娱乐、社交平台等需要活跃气氛的场合。

**6） 对比度与可读性：**提高文字与背景的对比度，确保文字与背景颜色之间有足够的对比度，方便用户阅读。

1. 实现步骤
2. **确定主色调与辅助色调**：
   1. 目标用户包括开发者、广告商和玩家，我们选择专业但不失活泼的颜色，比如蓝色+橙色，以突出科技感和活力。
   2. 主色调：选择蓝色作为核心色调，能够传达科技和信任感。
   3. 辅助色调：可选用白色、灰色进行背景处理，并使用橙色或红色作为按钮和重要提示的颜色，以吸引注意力。
3. **提高文字与背景的对比度：**
   1. 在深色背景上的使用浅色文字使文字更容易阅读
4. **色彩的一致性与层次感：**
   1. 保持整体色彩一致性：整个网站的色彩风格应保持统一，不同页面之间的色彩选择应避免过于分散，确保整体视觉的连贯性。
   2. 层次感的创建：通过不同的颜色层次区分页面内容的主次。使用深色背景和浅色文字强调重要内容，而次要信息可以使用中性色调（如灰色、米色）淡化处理，帮助用户更快地抓住页面的重点内容。
5. **实际应用与视觉验证：**
   1. 将色彩应用到页面设计中：将选定的主色调和辅助色调应用于网页的不同元素，背景、按钮、标题、文字、导航栏等。确保颜色在静态页面中保持整体协调，同时在动态交互中有效提升用户体验。
6. 任务总结

在设计时应考虑色彩的对比度、易读性以及用户的视觉习惯。

1. 任务思考

色彩的选择是否能够传递网站的主题和品牌信息？

是否考虑到了用户的色彩偏好及色彩心理学的影响？

**任务二 页面版式设计**

1. 学习目标

* 掌握页面的版式设计原则。
* 学习如何通过布局优化用户体验，提升网页的可读性和视觉吸引力。

1. 开发任务

• 设计合理的页面版式，使用户在浏览网站时能轻松获取信息。

• 根据内容层次和用户需求调整页面结构，确保页面布局简洁、逻辑清晰。

1. 开发思路

**1） 网格系统应用：**使用网格系统（grid system）来保证页面元素的对齐和布局的一致性。网格系统有助于页面内容的合理分布，避免杂乱无章。

**2） 核心内容优先：**在设计版式时，内容优先原则尤为关键。应确保最重要的内容出现在用户的首屏区域内，即用户无需滚动即可看到的部分。这个设计决定应基于对目标用户行为和需求的深入理解。例如，在电商网站首页，通常会把最新促销、热门商品或用户感兴趣的内容放置在显眼位置，以提高用户的参与度。

**3） 响应式设计：**设计时考虑不同设备的屏幕尺寸，确保在手机、平板和电脑上都能保持良好的版式和可读性。

4） 用户交互优化：用户交互设计不仅需要让按钮、链接等交互元素显眼且易用，还应确保交互后的反馈清晰。例如，按钮的点击、悬停等状态应该通过视觉反馈（如颜色变化或轻微的动画效果）来体现。这样可以增强用户的操作感，提升整体交互体验的流畅性和反馈及时性。

1. 实现步骤

1. **需求分析：**确定页面的核心内容模块和用户行为路径。首先明确首页需要展示的主要内容模块，如导航栏、产品信息、用户反馈等，确保最重要的信息出现在用户视线的中心位置。同时，分析用户浏览习惯，以优化布局和用户体验。

2. **设计草图：**使用线框图工具绘制初步的版式草图，将各个内容模块合理布局在页面中。通过线框图可以明确页面的主要信息区域和次要信息区域，确保用户能够快速抓住页面核心内容。草图中的每个模块应根据其重要性进行优先排列，尤其是需要用户重点关注的区域。

3. **网格系统应用：**根据页面内容，选择合适的网格布局系统，如12列网格，以确保页面元素的对齐和均匀分布。在设计时，注意将页面划分为不同的区域，包括主视觉区域、次视觉区域、导航栏等，确保用户可以轻松浏览各个模块。

4. **布局优化：**优化主视觉区域，突出显示最重要的信息和交互按钮，如CTA按钮、关键图片或标题。对于次要内容区域，放置用户可能会感兴趣但不需要立即关注的信息，如详细的文本内容、次要链接等。注意调整元素之间的间距和留白，避免页面过于紧凑，造成视觉疲劳。

5. **视觉层次感创建：**通过文字大小、图片比例、颜色对比等方式，强化页面的层次感。例如，使用大字号标题、小字号正文内容，引导用户从标题进入具体内容的阅读。

6. 用户交互优化：为按钮、链接等交互元素提供视觉反馈，如颜色变化或阴影效果，以增强用户操作的反馈感。在用户悬停或点击时，元素的样式会发生变化，提示用户当前的交互状态。确保所有交互元素易于点击，尤其在移动端设备上，增加交互区域的大小以提升触摸体验。

1. 任务总结

• 页面版式设计应保证简洁、清晰，避免用户视觉疲劳。通过网格系统和视觉引导元素，优化页面布局，提升用户体验。

1. 任务思考

• 页面布局是否能够清晰传递核心信息？内容优先原则是否得到了充分体现？

• 页面设计是否在不同设备上保持了良好的响应性和一致性？

**任务三 实现页面布局**

1. 学习目标

掌握页面布局的基本原理和技术，并能够运用这些技能在多种设备上实现响应式布局。

1. 开发任务

为网站创建合理且用户友好的页面布局，确保页面的主次内容得到良好展示，并提高用户的浏览体验。

1. 开发思路

1. **需求分析**：首先，分析并确定页面的主要内容模块和用户浏览路径。例如，在首页应优先展示的内容模块包括产品信息、导航栏和用户反馈。

2. **网格系统的应用**：选择12列的网格布局系统（或其他合适的网格布局），确保页面元素间的对齐和一致分布。通过网格系统，可以清晰地划分主要和次要视觉区域，确保内容合理分布，提升视觉体验。

3. **布局优化**：在主视觉区域突出重要信息和互动按钮；次要信息放置在次视觉区域。保证元素之间有适当的间距和空白，避免内容过于紧凑。

4. **视觉层次感创建**：通过字体大小、图片比例和颜色对比的变化来增强页面的层次感。例如，使用大字号标题和小字号正文，引导用户从主要标题到具体内容的阅读。

5. **内容优先**：确保最重要的内容放置在用户最容易访问的区域，通常是页面的顶部或中心位置。这将确保用户在不滚动页面的情况下可以获取关键信息。

6. **视觉引导**：利用空白、大小对比和颜色对比引导用户的视觉流动，使用户能够轻松识别页面中的重要信息和按钮。

1. 实现步骤

1. 通过HTML5中的语义化标签（如<header>、<nav>、<main>、<section>、<footer>等）构建页面的基础结构。这些标签不仅提升了代码的可读性，还能为SEO和屏幕阅读器用户提供友好的体验。

布局代码：

|  |
| --- |
| <!DOCTYPE html>  <html lang="zh-CN">  <head>      <meta charset="UTF-8">      <meta name="viewport" content="width=device-width, initial-scale=1.0">      <title>网页布局</title>      <link rel="stylesheet" href="style.css">  </head>  <body>  <header>      <nav>          <!-- 导航栏 -->          <ul></ul>      </nav>  </header>  <main>      <!-- 主视觉区域 -->      <section id="hero">          <div class="hero-content"></div>      </section>      <!-- 功能展示区域 -->      <section id="features">          <div class="feature-container"></div>      </section>      <!-- 路线图区域 -->      <section id="roadmap">          <div></div>      </section>  </main>  <footer>      <!-- 页脚 -->      <p></p>  </footer>  </body>  </html> |

2. 使用CSS实现页面的布局，通过Flexbox或CSS Grid创建灵活的布局，确保内容在不同设备上的响应式适配。同时，通过设置字体、颜色和间距，确保页面的视觉美观和层次感。

CSS代码：

|  |
| --- |
| /\* 通用样式 \*/  \* {      margin: 0;      padding: 0;      box-sizing: border-box;  }  body {      font-family: 'Arial', sans-serif;      color: #333;      line-height: 1.6;  }  /\* 导航栏样式 \*/  nav ul {      list-style: none;      display: flex;      justify-content: space-around;      background-color: #222;      padding: 10px;  }  nav ul li a {      color: #fff;      text-decoration: none;      font-size: 18px;  }  /\* 主视觉区域 \*/  #hero {      background-color: #444;      height: 100vh;      display: flex;      align-items: center;      justify-content: center;      text-align: center;      color: white;  }  /\* 功能展示区域 \*/  #features {      padding: 50px 0;  }  .feature-container {      display: flex;      justify-content: space-around;  }  /\* 路线图区域 \*/  #roadmap {      padding: 50px 0;      text-align: center;  }  /\* 页脚样式 \*/  footer {      background-color: #222;      color: white;      text-align: center;      padding: 20px 0;  }  /\* 响应式设计 \*/  @media (max-width: 768px) {      .feature-container {          flex-direction: column;          align-items: center;      }      #hero {          height: 50vh;      }  } |

3. 使用media queries处理不同设备的屏幕尺寸，确保在手机和平板上布局的调整，保持良好的用户体验。在小屏设备上，功能展示区可以从横向排列变为纵向排列。

4. 为导航栏、按钮和重要区域增加交互效果，例如悬停时的颜色变化和点击反馈，提升用户体验。确保布局的视觉层次感，通过颜色和交互效果区分重要信息和次要信息。

交互效果CSS代码：

|  |
| --- |
| /\* 导航栏悬停效果 \*/  nav ul li a {      color: #fff;      text-decoration: none;      font-size: 18px;      transition: color 0.3s ease;  }  nav ul li a:hover {      color: #FFD700; /\* 鼠标悬停时文字颜色变化 \*/  }  /\* 按钮样式及交互效果 \*/  button {      padding: 10px 20px;      background-color: #FFD700;      border: none;      cursor: pointer;      font-size: 18px;      transition: background-color 0.3s ease, transform 0.2s ease;  }  button:hover {      background-color: #FFA500; /\* 鼠标悬停时按钮颜色变化 \*/      transform: scale(1.05); /\* 悬停时按钮微微放大 \*/  }  button:active {      background-color: #FF8C00; /\* 点击时按钮颜色变化 \*/      transform: scale(0.95); /\* 点击时按钮微缩 \*/  }  /\* 主视觉区域悬停效果 \*/  #hero {      background-color: #444;      height: 100vh;      display: flex;      align-items: center;      justify-content: center;      text-align: center;      color: white;      transition: background-color 0.5s ease;  }  #hero:hover {      background-color: #555; /\* 主视觉区域悬停时背景颜色变化 \*/  } |

(1) 导航栏：当用户将鼠标悬停在导航链接上时，链接文字的颜色将从白色变为金色（#FFD700），增加视觉反馈。

(2) 按钮的背景颜色在不同交互状态下会发生变化，悬停时背景变成橙色（#FFA500），点击时则变成深橙色（#FF8C00）。此外，悬停时按钮会稍微放大（scale(1.05)），点击时稍微缩小（scale(0.95)），模拟真实的点击效果。

(3) 主视觉区域：当用户将鼠标悬停在主视觉区域时，背景颜色会稍微变暗，增加页面动态感。

1. 任务总结

页面布局设计应简洁清晰，避免视觉疲劳。通过视觉引导元素优化页面布局，以提升用户体验。

1. 任务思考

• 页面布局是否清晰传达了核心信息？

• 内容优先原则是否得到了充分体现?

• 页面设计是否在不同设备上保持了良好的响应性?

**任务四 制作导航栏目**

1. 学习目标

学习如何设计和开发导航栏，提升网站的可用性和用户体验。

1. 开发任务

创建一个清晰且功能齐全的导航栏，使用户能够轻松浏览网站的各个部分，并能快速找到所需信息。

1. 开发思路

1. **简洁明了**：导航栏目是网站中最关键的元素之一，设计应保持简洁、直观，避免过多的子菜单。导航栏的主要功能应该一目了然，用户能够快速识别并访问关键页面。

2. **响应式设计**：导航栏需要适应各种屏幕尺寸。对于桌面设备，导航栏可以采用水平排列，而在移动设备上，导航栏可以折叠成下拉菜单，节省空间。

3. **突出重要页面**：将用户最常访问的页面，如“首页”、“功能介绍”等放在导航栏的首要位置。对于子菜单，应当在鼠标悬停时清晰地展示，避免混乱。

4. **一致性和固定导航**：保持导航栏在整个网站上的一致性，并考虑使用固定导航栏，这样当用户滚动页面时，导航栏始终可见，方便用户随时进行页面切换。

5. 交互效果：通过CSS或JavaScript实现悬停效果，用户在鼠标悬停时能够清楚地看到选中的导航链接。例如，链接的颜色变化能够增强视觉反馈，提升用户体验。

1. 实现步骤

1. HTML结构构建：使用语义化的HTML标签，构建一个包括主菜单和子菜单的导航栏结构。

|  |
| --- |
| <header>      <nav>          <ul>              <li><a href="#home">首页</a></li>              <li><a href="#features">功能介绍</a></li>              <li>                  <a href="#roadmap">路线图</a>                  <ul>                      <li><a href="#sub-item1">子项1</a></li>                      <li><a href="#sub-item2">子项2</a></li>                      <li><a href="#sub-item3">子项3</a></li>                  </ul>              </li>              <li><a href="#contact">联系我们</a></li>          </ul>      </nav>  </header> |

2. CSS样式实现：实现导航栏的样式，使其在不同设备上表现良好。使用Flexbox进行布局，并结合媒体查询实现响应式。

|  |
| --- |
| /\* 导航栏样式 \*/  nav {      background-color: rgba(34, 34, 34, 0.8); /\* 背景半透明效果 \*/      position: fixed; /\* 固定在页面顶部 \*/      width: 100%; /\* 占据全宽 \*/      top: 0;      padding: 10px 0;      z-index: 10; /\* 保证导航栏位于其他元素上层 \*/      transition: background-color 0.5s ease;  }  nav ul {      list-style-type: none; /\* 去掉默认列表样式 \*/      display: flex;      justify-content: center; /\* 水平居中 \*/  }  nav ul li {      position: relative; /\* 子菜单相对于父级导航项定位 \*/      margin: 0 20px; /\* 设置菜单项之间的间隔 \*/  }  nav ul li a {      color: white;      text-decoration: none;      font-size: 18px;      transition: color 0.3s ease; /\* 悬停颜色渐变效果 \*/  }  /\* 鼠标悬停时变为金色 \*/  nav ul li a:hover {      color: #FFD700;  }  /\* 子菜单初始隐藏 \*/  nav ul li ul {      display: none;      position: absolute;      top: 100%;      left: 0;      background-color: rgba(34, 34, 34, 0.9);      padding: 10px 0;      list-style: none;      min-width: 200px;  }  /\* 鼠标悬停时显示子菜单 \*/  nav ul li:hover ul {      display: block;  }  /\* 滚动时背景颜色变深 \*/  body.scrolled nav {      background-color: #222;  } |

3. 交互优化：利用JavaScript增强用户体验，使导航栏在页面滚动时变得更加实用。例如，当页面滚动时导航栏背景变深，以增强可见性。

|  |
| --- |
| // 监听页面滚动事件，控制导航栏的背景变化  window.addEventListener('scroll', function() {      const nav = document.querySelector('nav');      if (window.scrollY > 50) {          document.body.classList.add('scrolled');      } else {          document.body.classList.remove('scrolled');      }  }); |

4. 响应式设计：通过媒体查询调整导航栏的布局，使其在小屏设备上表现良好。例如，将水平排列的菜单项改为垂直排列，或使用折叠菜单。

|  |
| --- |
| @media (max-width: 768px) {      nav ul {          flex-direction: column; /\* 在小屏设备上垂直排列导航栏 \*/      }  } |

1. 任务总结

导航栏是网站的核心部分，实现时应考虑简洁性、响应性和可用性。通过清晰的布局和一致的设计，确保用户能够轻松导航到网站的各个部分。

1. 任务思考

• 导航栏是否清晰且易于使用？

• 用户是否能够快速找到所需的信息？

• 导航栏在不同设备上是否都能保持良好的显示效果？

**任务五 制作主图栏目**

1. 学习目标

掌握如何制作网页中的主图栏目，确保其具有吸引力并且有效传递关键信息。

1. 开发任务

设计一个主图栏目，能够展示网站的核心信息或功能，吸引用户的注意力，并引导用户进一步了解内容。

1. 开发思路

1. **突出视觉效果**：主图栏目通常是用户进入网站后看到的第一个区域，因此需要通过图片、文字、颜色等视觉元素抓住用户的注意力。

2. **信息简洁明了**：主图栏目应以最简洁的方式传递网站的核心价值。标题应突出产品或服务的关键优势，副标题用来补充详细信息。

3. **响应式设计**：主图栏目需要适应不同屏幕尺寸。大屏设备上，背景图像和文字可以占据较多空间，而在小屏设备上，布局应简化，文字和按钮需要缩小以保证良好的用户体验。

4. 按钮应具有吸引力，使用鲜明的颜色和易于识别的文本，例如“了解更多”或“立即体验”。其目的在于引导用户进行操作，因此按钮的设计应突出且具备点击反馈

5. **互动性元素**：按钮在鼠标悬停时应给予用户清晰的反馈，例如颜色变化或轻微的缩放效果。这样可以吸引用户的注意，并提升页面的互动感。页面加载时，可以为主标题、副标题和按钮添加淡入或滑入的动画效果，增加页面的动态感，提升用户体验。

1. 实现步骤

**1. 主图栏目HTML结构构建**：通过语义化的HTML标签构建主图栏目的基础结构，包括标题、副标题和CTA按钮。主图应放置在一个独立的<section>标签中，便于布局管理和样式控制。

|  |
| --- |
| <section id="hero">      <div class="hero-content">          <h1>下一代网页游戏平台</h1>          <h2>开发者、广告商和玩家的三赢局面</h2>          <button>立即了解更多</button>      </div>  </section> |

**2.  CSS样式设计**：为主图部分添加背景图片和文本样式，确保文字在背景上清晰可见。使用flexbox布局使内容居中，并确保在不同设备上内容的显示效果良好。

|  |
| --- |
| /\* 主图栏目样式 \*/  #hero {      background-image: url('images/hero-background.jpg'); /\* 背景图片 \*/      background-size: cover;      background-attachment: fixed; /\* 视差滚动效果 \*/      background-position: center;      height: 100vh; /\* 全屏高度 \*/      display: flex;      align-items: center;      justify-content: center;      text-align: center;      position: relative;      color: white;  }  /\* 增加背景遮罩提高文字对比度 \*/  #hero::before {      content: '';      position: absolute;      top: 0;      left: 0;      right: 0;      bottom: 0;      background-color: rgba(0, 0, 0, 0.5); /\* 半透明黑色遮罩 \*/      z-index: 1; /\* 确保在文字后面 \*/  }  .hero-content {      position: relative;      z-index: 2; /\* 确保文字在遮罩上方 \*/  }  .hero-content h1 {      font-size: 48px;      margin-bottom: 20px;      text-shadow: 2px 2px 5px rgba(0, 0, 0, 0.5); /\* 增加文字阴影 \*/  }  .hero-content h2 {      font-size: 28px;      margin-bottom: 30px;  }  .hero-content button {      padding: 15px 30px;      background-color: #FFD700;      border: none;      color: black;      font-size: 20px;      cursor: pointer;      transition: background-color 0.3s ease;  }  .hero-content button:hover {      background-color: #FFA500; /\* 鼠标悬停效果 \*/  } |

**3. 交互效果和动画：**为主图栏目中的内容添加动画效果，使页面加载时具有动态性。可以使用CSS动画，如淡入效果，让内容逐步呈现。

|  |
| --- |
| /\* 主图内容的淡入动画 \*/  @keyframes fadeIn {      from {          opacity: 0;          transform: translateY(20px);      }      to {          opacity: 1;          transform: translateY(0);      }  }  .hero-content {      animation: fadeIn 2s ease-in;  } |

**4. 响应式媒体设计：**使用媒体查询调整主图栏目在小屏设备上的布局。缩小文字和按钮尺寸，确保主图栏目在移动设备上也能保持美观和易用。

|  |
| --- |
| @media (max-width: 768px) {      #hero {          height: 70vh; /\* 调整主图高度 \*/      }      .hero-content h1 {          font-size: 36px; /\* 缩小标题 \*/      }      .hero-content h2 {          font-size: 20px;      }      .hero-content button {          font-size: 18px;          padding: 10px 20px;      }  } |

**5. JavaScript优化：**如果需要进一步增强交互体验，可以通过JavaScript为主图中的按钮或图片添加更多交互功能，例如滚动到特定页面位置，或者动态加载内容。

|  |
| --- |
| // 为CTA按钮添加点击事件，滚动到"功能介绍"部分  document.querySelector('.hero-content button').addEventListener('click', function() {      document.querySelector('#features').scrollIntoView({ behavior: 'smooth' });  }); |

1. 任务总结

主图栏目是网站最显眼的部分，设计时应注重视觉效果、信息传递的简洁性，以及通过CTA按钮引导用户进行进一步操作。响应式设计和动态效果可以显著提升用户体验。

1. 任务思考

• 主图栏目是否能够清晰传达核心信息？

• 是否吸引了用户的注意力并引导他们进行后续操作？

• 在不同设备上是否能够保持一致的视觉效果和用户体验？

• CTA按钮是否明确引导用户进行下一步操作？

**任务六 实现轮播图**

1. 学习目标

掌握如何设计和实现网页中的轮播图（Carousel），确保其能够展示多个关键信息或图片，具备良好的视觉效果和过渡效果，并能够适应不同设备的屏幕尺寸。轮播图的实现不依赖JavaScript，而是使用纯HTML和CSS完成切换效果。

1. 开发任务

实现一个自动轮播的图像展示区，包含手动切换功能。轮播图区域需要具备切换的视觉效果，支持用户通过按钮选择特定的图片，并在不同设备上自适应显示。

1. 开发思路

**1. 轮播图结构设计**：

(1) 图片展示：轮播图应展示多张图片，并支持用户在这些图片之间进行切换。每张图片应能展示在指定区域内，其他图片应隐藏，确保用户在任意时间内只看到一张图片。

(2) 内容组织：每张图片可能伴随一些简短文字或标题说明，这些内容应该在切换时一起显示与隐藏。

(3) 手动切换与指示器：轮播图应允许用户通过底部的导航指示器（通常是圆点）手动切换到特定的图片，并且应高亮显示当前选中的图片

**2. CSS控制切换效果：**

(1) 使用CSS的radio输入元素实现切换：每张图片都与一个radio按钮关联，通过用户选择不同的radio按钮来控制图片的显示状态。只使用CSS来管理这些交互，避免依赖JavaScript。

(2) 平滑过渡效果：图片之间的切换应有平滑的过渡效果，提升视觉体验。可以使用CSS的transition属性来实现这种平滑切换效果。

**3. 自动轮播实现**：

(1) 通过CSS动画实现自动播放：可以利用CSS的@keyframes和animation属性为轮播图添加自动切换功能，实现不需要JavaScript代码的自动轮播。

**4. 响应式设计**：

(1) 适应不同屏幕尺寸：轮播图应支持用户手动滑动操作，提供控制按钮（如前进、后退按钮和页码指示器）。

(2) 布局调整：在小屏设备上，轮播图的指示器和图片大小应做出相应的缩放和布局调整，保证良好的用户体验。

1. 实现步骤

**1. 轮播图结构**：使用input标签（radio类型）和label标签配合实现用户点击时的图片切换。每个radio元素控制一张图片，使用CSS来实现它们的切换显示。

|  |
| --- |
| <section id="carousel">      <div class="carousel">          <input type="radio" id="slide1" name="carousel" checked>          <input type="radio" id="slide2" name="carousel">          <input type="radio" id="slide3" name="carousel">          <div class="slides">              <div class="slide" id="slide1-content">                  <img src="images/img1.jpg" alt="图片1">              </div>              <div class="slide" id="slide2-content">                  <img src="images/img2.jpg" alt="图片2">              </div>              <div class="slide" id="slide3-content">                  <img src="images/img3.jpg" alt="图片3">              </div>          </div>          <!-- 轮播控制按钮 -->          <div class="carousel-controls">              <label for="slide1"></label>              <label for="slide2"></label>              <label for="slide3"></label>          </div>      </div>  </section> |

2. CSS样式设计：使用CSS来控制轮播图的布局和切换效果。通过radio按钮控制transform属性，完成图片的滑动切换。使用transition实现平滑过渡。

|  |
| --- |
| /\* 轮播图整体样式 \*/  .carousel {      position: relative;      width: 100%;      max-width: 1200px;      height: 500px;      margin: 50px auto;      overflow: hidden;      border-radius: 10px;  }  /\* 滑动区域 \*/  .slides {      display: flex;      width: 300%; /\* 包含三张图片，每张图片占33.33% \*/      transition: transform 1s ease; /\* 平滑过渡效果 \*/  }  /\* 切换不同图片的显示 \*/  input[type="radio"]#slide1:checked ~ .slides {      transform: translateX(0); /\* 显示第一张图片 \*/  }  input[type="radio"]#slide2:checked ~ .slides {      transform: translateX(-33.33%); /\* 显示第二张图片 \*/  }  input[type="radio"]#slide3:checked ~ .slides {      transform: translateX(-66.66%); /\* 显示第三张图片 \*/  }  /\* 每张图片的样式 \*/  .slide {      width: 100%;      flex: 1 0 33.33%;  }  .carousel img {      width: 100%;      height: auto;      border-radius: 10px;  }  /\* 控制按钮的样式 \*/  .carousel-controls {      position: absolute;      bottom: 20px;      left: 50%;      transform: translateX(-50%);      display: flex;      justify-content: center;  }  .carousel-controls label {      cursor: pointer;      width: 15px;      height: 15px;      border-radius: 50%;      background-color: white;      margin: 0 5px;      transition: background-color 0.3s ease;  }  /\* 鼠标悬停时高亮显示 \*/  .carousel-controls label:hover {      background-color: #FFD700;  }  /\* 当前选中的图片指示器高亮 \*/  input[type="radio"]:checked + label {      background-color: #FFD700;  } |

3. 自动轮播实现：使用CSS的@keyframes和animation来实现图片的自动切换。通过将每张图片的显示绑定到CSS动画中，控制图片的自动切换。

|  |
| --- |
| /\* 自动播放的动画效果 \*/  @keyframes slide {      0% {          transform: translateX(0);      }      33% {          transform: translateX(-33.33%);      }      66% {          transform: translateX(-66.66%);      }      100% {          transform: translateX(0);      }  }  /\* 应用到slides上，图片自动轮播 \*/  .slides {      animation: slide 9s infinite;  } |

4. 响应式设计：为确保轮播图能够在各种设备上都显示良好，使用媒体查询调整布局和图片大小，以适应移动设备的屏幕。

|  |
| --- |
| @media (max-width: 768px) {      .carousel {          height: 300px; /\* 移动设备上调整轮播图高度 \*/      }      .slides {          width: 300%; /\* 保持三张图片的布局 \*/      }      .slide {          flex: 1 0 100%; /\* 小屏幕设备下，每张图片占满屏幕宽度 \*/      }  } |

5. 手动切换与指示器：用户可以通过轮播图底部的指示器手动切换图片。radio按钮与每张图片对应，通过选择不同的radio，控制图片的显示。

|  |
| --- |
| /\* 手动选择图片时的指示器反馈 \*/  .carousel-controls label {      cursor: pointer;      width: 15px;      height: 15px;      border-radius: 50%;      background-color: white;      margin: 0 5px;      transition: background-color 0.3s ease;  }  /\* 当前选中的图片指示器高亮 \*/  input[type="radio"]:checked + label {      background-color: #FFD700; /\* 选中后的高亮颜色 \*/  } |

1. 任务总结

轮播图是网站展示多条信息的重要元素，设计时应确保其视觉美观、功能流畅。通过交互性设计，提升用户体验，并确保其在各种设备上表现出色。

1. 任务思考

• 轮播图是否能够清晰传达关键信息？

• 动画效果是否足够平滑，且不会对用户体验造成干扰？

• 轮播图在不同设备上是否具有良好的适配性和响应性？

**项目三**

**世界技能大赛典型网站设计与开发**

**移动网页设计与实现**

**项目需求**

在这个模块中，我们设计并创建了一个营销页面来推广游戏平台。该页面的标题是“下一代网络游戏平台”，副标题是“对开发者、广告商和玩家的双赢”。

交付项目：

1.桌面级单页面实现营销网页

2.移动版本的单页面实现营销网页

3.一种展示用来在社交网络服务上发布的网页的设计。

**任务一** **移动端网页设计一**

1. 学习目标

了解移动端网页的设计原则，并掌握如何通过导航优化提升用户体验。

1. 开发任务

设计一个移动设备友好的导航方案，使用图标（通常为汉堡菜单）作为入口，点击该图标后展示一个从屏幕侧边滑出的导航菜单，帮助用户在有限的屏幕空间内快速访问网站的各个部分

1. 开发思路

1. **移动优先设计**：针对移动设备的特性，设计时考虑屏幕空间有限，确保最重要的内容始终处于用户视野内。传统的水平导航栏在移动设备上占用过多的宝贵空间，因此选择导航图标（如汉堡菜单）点击后展示侧边栏的设计可以优化空间利用。

2. **导航图标与侧边栏设计**：在移动端，导航图标通常使用简单易懂的“☰”汉堡图标作为触发按钮，点击该图标时侧边栏从屏幕左侧滑出，展示导航内容。这样能够在不干扰用户体验的情况下，快速访问其他页面或功能。导航栏内容应保持简洁，突出关键链接，如“首页”、“功能介绍”、“路线图”等。

3. **交互设计优化**：用户点击图标时，侧边栏通过平滑的动画效果滑出。当点击其他区域时，侧边栏自动关闭。这种交互方式既简洁又直观，符合移动端用户的操作习惯，避免过多复杂的菜单操作。

1. 实现步骤

1.  **设计导航图标与侧边栏的布局结构：**在移动端，展示一个简洁的汉堡菜单图标，点击后侧边栏显示完整的导航项。

2. **侧边栏的交互设计**：点击汉堡图标，侧边栏滑出；再次点击或点击外部区域时，侧边栏收回，保证用户的操作体验流畅。

3. **响应式设计**：设计方案需要保证在不同设备上均能适应屏幕宽度，移动设备上采用图标导航，桌面设备上保持传统水平导航。

1. 任务总结

移动端导航设计通过使用导航图标（汉堡菜单）和侧边栏展示，使用户能够在有限的屏幕空间内快速访问网站的不同部分。该设计不仅节省了屏幕空间，还增强了用户的交互体验。通过合理的布局和交互设计，可以确保移动设备上的导航操作便捷顺畅，且与桌面设备保持一致的用户体验。

1. 任务思考

• 导航图标是否足够显眼，用户能否快速识别并使用？

• 侧边栏设计是否简洁明了，能够快速展示和关闭？

• 设计是否在移动设备和桌面设备上都能提供良好的体验？

**任务二 移动端网页设计二**

1. 学习目标

学习如何通过优化移动端网页的布局设计，提升用户的视觉效果与交互体验。

掌握在有限屏幕空间内的内容展示技巧，使用折叠式内容展示等功能，使页面更加简洁、直观。

1. 开发任务

优化移动端网页的设计，使页面结构更为简洁，用户体验更友好。使用折叠式内容展示等设计技巧，帮助用户高效地获取信息，并避免内容过多带来的视觉负担。

1. 开发思路

**1. 折叠式内容展示**：

(1) 移动设备的屏幕空间有限，用户不能同时看到过多信息。因此，为了减少用户滚动页面的次数，可以将次要内容或较长的内容部分折叠起来，用户只需点击标题即可展开或收起这些内容。

(2) 这种折叠式内容展示（Accordion）适用于展示大量信息的页面，例如常见问题解答（FAQ）、功能说明或详细产品介绍部分。用户可以通过点击展开所需的信息，避免一次性展示所有内容带来的混乱感。

**2. 简化页面结构**：

(1) 在移动端，内容应保持直观且简洁。优化时应确保重要的内容出现在页面的上半部分，让用户无需频繁滚动。

(2) 设计时可使用分块式布局，将页面内容分为若干独立且紧凑的模块，如功能展示区、用户评论区等。通过这种简化结构，用户可以快速找到所需的内容，而不必被冗余的设计分散注意力。

**3. 优化用户交互**：

(1) 在移动端设计时，应考虑到用户的触摸操作习惯。按钮、导航栏、内容展开区域等交互元素的尺寸应设计得足够大，以确保用户能够轻松点击，避免误触。

(2) 可使用清晰的视觉反馈来增强用户体验。例如，当用户点击折叠标题时，展示一个箭头图标或色彩变化，提示该区域已经展开或收起。

**4. 减少信息干扰**：

(1) 使用渐进显示内容的设计方式，帮助用户专注于当前阅读的内容。例如，页面初次加载时只展示简要信息或摘要，用户可通过点击“阅读更多”展开详细信息

(2) 避免一次性呈现过多的信息块，特别是在移动端屏幕上，用户不希望被过量的文本或图片内容所淹没。通过对信息的精简和视觉上的层次感，可以使页面显得更加有条理。

1. 实现步骤

**1. HTML结构设计**：为每个折叠内容创建基本的HTML结构，包括标题和隐藏的内容区域。确保每个折叠区域具有唯一的标识，以便于后续的JavaScript交互实现。

|  |
| --- |
| <div class="accordion">      <h2 class="accordion-title">功能介绍</h2>      <div class="accordion-content">          <p>详细介绍功能1...</p>      </div>  </div> |

2. CSS样式设计：为折叠标题和内容区域添加样式，确保标题突出，内容区有适当的内边距，保持视觉上的分隔。

|  |
| --- |
| .accordion-content {      display: none; /\* 默认隐藏 \*/      padding: 10px; /\* 内容内边距 \*/      background-color: #f9f9f9; /\* 背景色 \*/  } |

3. 交互效果实现：使用JavaScript监听用户的点击事件。当用户点击折叠标题时，切换内容区域的显示状态（展开或收起）。

|  |
| --- |
| document.querySelectorAll('.accordion-title').forEach(title => {      title.addEventListener('click', function() {          const content = this.nextElementSibling;          content.style.display = content.style.display === 'block' ? 'none' : 'block';      });  }); |

1. 任务总结

移动端网页设计应强调简洁、快速加载和操作便捷，同时确保视觉效果的统一性。通过不断测试和优化，提升用户体验。

1. 任务思考

页面设计是否能够适应不同的屏幕尺寸？

**任务三 移动端网页实现一**

1. 学习目标
2. 开发任务
3. 开发思路
4. 实现步骤
5. 任务总结
6. 任务思考

**任务四 移动端网页实现二**

1. 学习目标
2. 开发任务
3. 开发思路
4. 实现步骤
5. 任务总结
6. 任务思考

**项目四**

**世界技能大赛典型网站设计与开发**

**JavaScript脚本制作轮播图**

**项目需求**

在这个项目中，客户要求您创建一个照片幻灯片工具。

用户可以加载外部照片，或将示例照片加载到幻灯片中。然后，照片会一张一张地出现、停留几秒钟，然后消失。根据所应用的主题，照片的出现和消失可能会有过渡和动画。

照片幻灯片播放有不同的模式：

* 手动模式：使用左右键盘按键。
* 自动播放：照片不断出现，最后一张照片显示后循环播放第一张照片。
* 随机播放照片：几秒后随机显示一张新照片。在此模式下，幻灯片将永远播放，永不结束。

照片幻灯片也可以变成全屏浏览。全屏后，浏览器工具栏和 Windows 任务栏将隐藏。

用户可以通过将图像文件拖到放置区域来加载图像，这些图像将根据主题的动画进行显示和播放。如果 CSS 不可用或禁用，用户仍然可以通过文件文件上传控件选择照片文件。照片将加载并列在网页上。无需应用样式。

照片的标题由文件名定义。请将文件名的 slug 大写。标题不包括文件扩展名。

例如，如果文件名是“hello.jpg”，则标题将为“Hello”。例如，“hello world.jpg”的标题将为“Hello World”。

例如，“a-sample-photo.jpg”的标题将为“A Sample Photo”。

**任务一** **初始化Photos Slideshow项目**

1. 学习目标

* 学习如何使用Vue创建初始化项目。

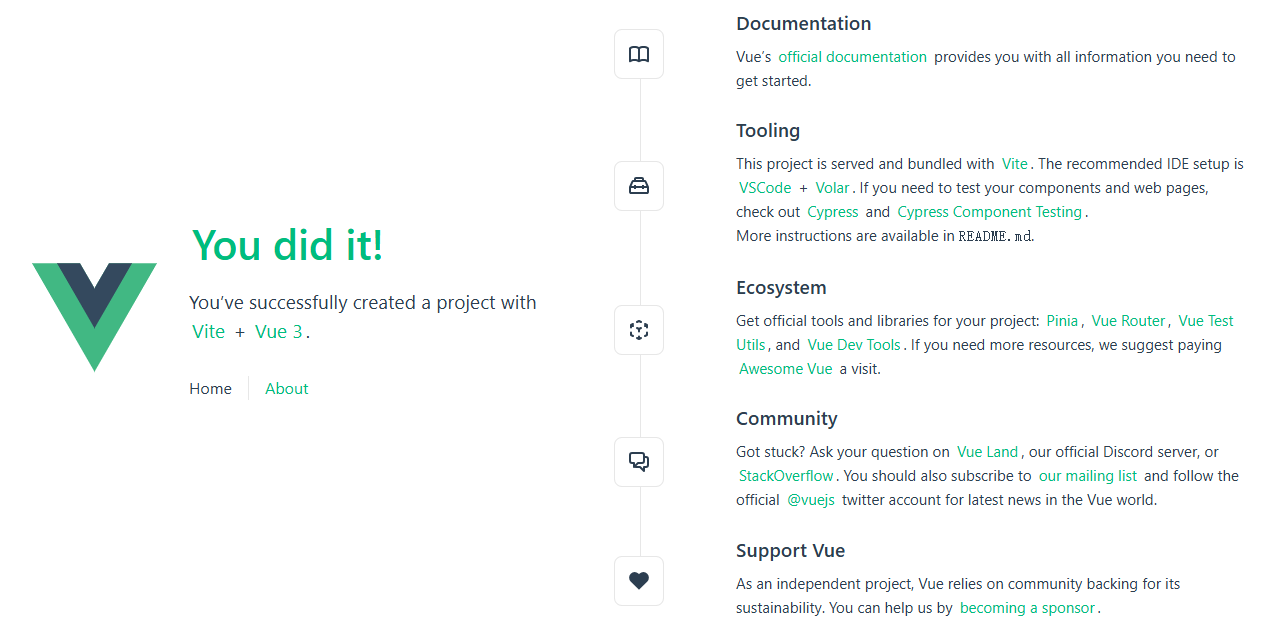
1. 开发任务

* 创建Vue项目。

1. 实现步骤

* 打开CMD窗口。
* 输入命令npx create-vue。
* 根据所需要的内容进行选择创建。
* 进入创建项目的目录cd XXX
* 输入npm install
* 等待依赖安装完毕输入 npm run dev成功运行项目

运行效果如下:



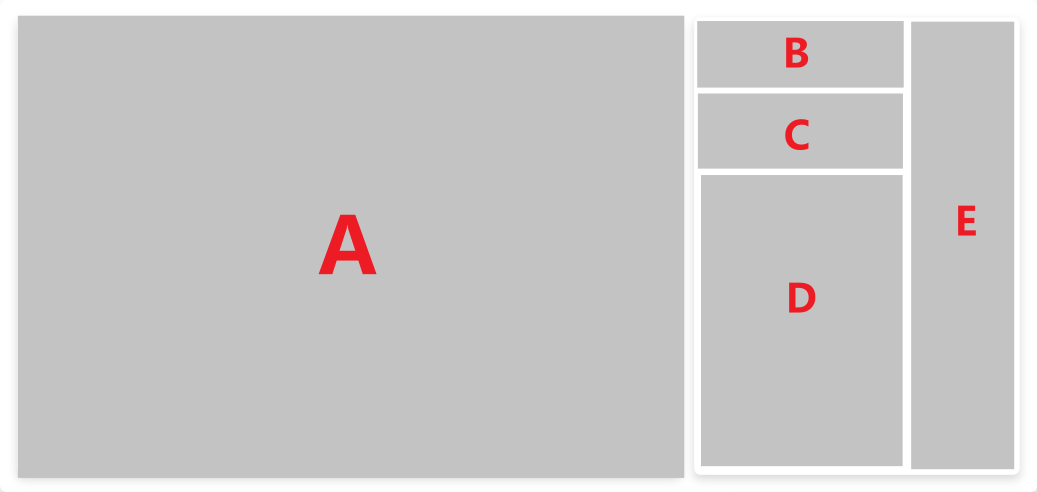
**任务二 实现 Photos Slideshow游戏内容界面**

1. 学习目标

* 学习如何创建网页的基本结构，通过CSS来改变网页元素的布局。

1. 开发任务

* 编写Photos Slideshow配置面板界面
* 页面参考图如下:
  + A:幻灯片展示区域
  + B:操作模式配置区域
    - 在面板上，用户可以切换手动控制、自动播放、随机播放3种操作模式。
  + C:活动主题切换区域
    - 在面板中，用户可以在主题 A 到 F 之间切换。
  + D:照⽚上传区域
  + E:照⽚排序区域
    - 在这个配置面板中，用户可以通过拖放来对选定的照片进行排序。



1. 开发思路

* 将五个区域划分为三个组件，A为"slide-controller"组件，BCD为"setting-area"组件，E为"ordering-area"组件。

1. 实现步骤
   1. 首先在Vue项目的组件目录components下分别创建SlideController.vue、SettingArea.vue、OrderingArea.vue。
   2. 创建主样式main.css

|  |
| --- |
| #app {  height: 100vh;  padding: 2rem;  }  .centerBox {  width: 100%;  height: 100%;  display: flex;  justify-content: center;  align-items: center;  }  .staticBox {  position: absolute;  left: 0;  right: 0;  top: 0;  bottom: 0;  } |

* 1. 创建公共样式common.css

|  |
| --- |
| /\* variables \*/  :root {  --mask: radial-gradient(ellipse at 20% -10%, rgba(255,255,255,.7), rgba(255,255,255,1));  }  /\* reset \*/  \* {  margin: 0;  padding: 0;  box-sizing: border-box;  }  body {  letter-spacing: -0.025em;  line-height: 1.5;  font-family: Arial, serif;  }  label small,  .labelSmall {  display: block;  margin-bottom: .3rem;  }  label button {  cursor: pointer;  }  img, video {  object-fit: cover;  object-position: center;  }  /\* common \*/  .section {  background: #fff;  border-radius: .75rem;  box-shadow: var(--bs-box-shadow, none);  padding: 1.5rem;  }  .singleForm {  width: 440px;  margin: 0 auto;  }  .form-control,  .form-select {  border-radius: .5rem;  }  .rounded {  border-radius: .75rem !important;  }  h1 {  font-size: 2.2rem;  font-weight: bold;  }  h2 {  font-size: 1.5rem;  }  h3{  font-size: 1.25rem;  }  h4 {  font-size: 1rem;  }  .w-40 {  width: 40%;  }  /\* list \*/  .list-wrapper {  padding: 0;  margin-top: calc(var(--bs-gutter-y) \* -1);  margin-bottom: calc(var(--bs-gutter-y) \* -1);  }  .list-wrapper > .list-item {  padding-top: calc(var(--bs-gutter-y) \* 1);  padding-bottom: calc(var(--bs-gutter-y) \* 1);  }  .list-wrapper > .list-item:not(:last-child) {  border-bottom: 1px solid rgba( 0,0,0,.125);  }  /\* btn \*/  .btn {  position: relative;  z-index: 1;  border-radius: 50rem;  padding: .5em 1em;  mask: var(--mask, none);  }  .btn-link {  padding: 0;  }  .btn-fill {  border: none;  }  .btn-fill::before {  content: "";  position: absolute;  z-index: -1;  left:0;  right: 0;  top: 0;  bottom: 0;  background: currentColor;  opacity: 0;  transition: .3s;  border-radius: inherit;  }  .btn-fill:hover::before,  .btn-fill.active::before {  opacity: .07;  }  .btn-fill.active:hover::before {  opacity: .15;  }  .btn\_\_big {  width: 100%;  text-align: center;  border-radius: .5rem;  padding: .5em 1em;  }  /\* badge \*/  .badge {  position: relative;  z-index: 1;  border-radius: 50rem;  }  .badge::before {  content: "";  position: absolute;  z-index: -1;  left:0;  right: 0;  top: 0;  bottom: 0;  background: currentColor;  opacity: 0.07;  border-radius: inherit;  } |

* 1. 导入Bootstrap.css,将提供的和自定义的css文件放入到Vue项目src/assets目录下，并且在main.js当中导入。

|  |
| --- |
| import './assets/bootstrap-5.3.3.min.css'  import './assets/common.css'  import './assets/main.css'  import { createApp } from 'vue'  import App from './App.vue'  createApp(App).mount('#app') |

* 1. 一起导入到App.vue根组件当中，并使用Bootstrap的栅格系统，创建一个包含两列的布局。
     1. 左侧占用8列，包含SlideController组件。
     2. 右侧占用4列，包含一个section，其中又分为两部分：
        1. 上方8列，包含SettingArea组件。
        2. 下方4列，包含OrderingArea组件

|  |
| --- |
| <script setup>  import SlideController from "@/components/SlideController.vue";  import SettingArea from "@/components/SettingArea.vue";  import OrderingArea from "@/components/OrderingArea.vue";  </script>  <template>  <div class="row h-100">  <div class="col-8">  <slide-controller></slide-controller>  </div>  <div class="col">  <section class="section h-100">  <div class="row h-100">  <div class="col-8">  <setting-area></setting-area>  </div>  <div class="col">  <ordering-area></ordering-area>  </div>  </div>  </section>  </div>  </div>  </template>  <style scoped>  </style> |

1. 任务总结

本任务主要是创建的题目当中所提的要求，要求中显示要有配置面板，本节将面板关键部分进行组件划分。

1. 任务思考

* 为什么将配置功能区域进行组件划分？
* 划分组件的好处是什么？

**任务三 实现Photos Slideshow 切换操作模式功能**

1. 学习目标

* 学习如何制作如何切换幻灯片轮播模式的功能。

1. 开发任务

操作模式切换区域的布局及功能编写。

1. 开发思路

* 创建应用程序的公共状态管理文件store.js。
* 在SettingArea组件中编写操作模式切换功能，去操作公共状态管理文件当中的值（此任务和后续任务会使用此值）。

1. 实现步骤
   1. 在src目录下创建一个 store.js 文件，文件的作用是定义和应用程序的状态。

|  |
| --- |
| import { ref} from "vue";  export const appMode = ref('RANDOM'); // MANUAL AUTO RANDOM |

* 1. 在我们的SettingArea组件中进行模式切换功能的编写。
     1. 使用**<div class="d-flex flex-column h-100">**创建一个垂直方向的弹性布局，确保内容能够在全高容器中垂直排列。
     2. **<p class="mb-2">Switching operating</p>**：显示一个标题，内容为“Switching operating”，用于说明下面的按钮功能。
     3. **<div class="border rounded-pill p-2 d-flex align-items-center mb-4">**：这是一个按钮组的容器，添加了一些边框、圆角和内边距，使得整体外观更美观。d-flex和align-items-center类使得容器内的内容居中对齐。
     4. 使用**<div class="row gx-1 w-100">**创建一个行，gx-1表示在列之间添加水平间距。内部包含三列，分别放置三个按钮。
     5. 每个按钮使用<button>元素，设置为全宽（w-100）并居中显示文本。按钮根据当前的appMode状态，通过调用btnClass函数动态添加样式类。
     6. 当按钮被点击时，通过@click事件更新appMode的值，切换不同的操作模式。
     7. btnClass(a, b)：这个函数，用于根据当前的操作模式返回按钮的样式类。如果按钮的操作模式与传入的参数b相同，则返回"btn-primary"，表示该按钮被选中；否则返回"btn-fill text-primary"，表示该按钮未被选中。

|  |
| --- |
| <script setup>  import {appMode} from "@/store.js";  /\* selected btn and unselected btn class \*/  function btnClass(a, b) {  if (a === b) {  return "btn-primary";  }  return "btn-fill text-primary";  }  </script>  <template>  <div class="d-flex flex-column h-100">  <p class="mb-2">Switching operating</p>  <div class="border rounded-pill p-2 d-flex align-items-center mb-4">  <div class="row gx-1 w-100">  <div class="col">  <button class="btn w-100 text-center" :class="btnClass(appMode, 'MANUAL')" @click="appMode = 'MANUAL'">  MANUAL  </button>  </div>  <div class="col">  <button class="btn w-100 text-center" :class="btnClass(appMode, 'AUTO')" @click="appMode = 'AUTO'">AUTO  </button>  </div>  <div class="col">  <button class="btn w-100 text-center" :class="btnClass(appMode, 'RANDOM')" @click="appMode = 'RANDOM'">  RANDOM  </button>  </div>  </div>  </div>  </div>  </template>  <style scoped>  </style> |

1. 任务总结

**操作模式切换**：允许用户在“手动”（MANUAL）、“自动”（AUTO）和“随机”（RANDOM）三种操作模式之间进行切换。

**动态样式应用**：根据当前选中的操作模式动态改变按钮的样式，提升用户体验。

**状态管理**：使用Vue的响应式特性，实时更新和展示当前的操作模式。

1. 任务思考

* 为什么创建公共状态管理文件？

**任务四 实现 Photos Slideshow 切换主题功能**

1. 学习目标

* 学习如何制作如何切换幻灯片主题的功能。

1. 开发任务

轮播图主题切换区域的布局及功能编写。

1. 开发思路

* 在应用程序的公共状态管理文件中添加关于主题的状态。
* 在SettingArea组件中编写主题切换功能，去操作公共状态管理文件当中的值（此任务和后续任务会使用此值）。

1. 实现步骤
   1. 在store.js中添加当前轮播图主题的状态

|  |
| --- |
| import { ref} from "vue";  export const appMode = ref('RANDOM'); // MANUAL AUTO RANDOM  export const appTheme = ref("F"); // A B C D E F |

* 1. 继续在SettingArea组件中进行主题切换功能的编写

|  |
| --- |
| <script setup>  import {appMode, appTheme} from "@/store.js";  /\* selected btn and unselected btn class \*/  function btnClass(a, b) {  if (a === b) {  return "btn-primary";  }  return "btn-fill text-primary";  }  </script>  <template>  <div class="d-flex flex-column h-100">  <p class="mb-2">Switching operating</p>  <div class="border rounded-pill p-2 d-flex align-items-center mb-4">  <div class="row gx-1 w-100">  <div class="col">  <button class="btn w-100 text-center" :class="btnClass(appMode, 'MANUAL')" @click="appMode = 'MANUAL'">  MANUAL  </button>  </div>  <div class="col">  <button class="btn w-100 text-center" :class="btnClass(appMode, 'AUTO')" @click="appMode = 'AUTO'">AUTO  </button>  </div>  <div class="col">  <button class="btn w-100 text-center" :class="btnClass(appMode, 'RANDOM')" @click="appMode = 'RANDOM'">  RANDOM  </button>  </div>  </div>  </div>  <p class="mb-2">Switching theme</p>  <div class="border rounded-pill p-2 d-flex align-items-center mb-4">  <div class="row gx-1 w-100">  <div class="col">  <button class="btn w-100 text-center" :class="btnClass(appTheme, 'A')" @click="appTheme = 'A'">A</button>  </div>  <div class="col">  <button class="btn w-100 text-center" :class="btnClass(appTheme, 'B')" @click="appTheme = 'B'">B</button>  </div>  <div class="col">  <button class="btn w-100 text-center" :class="btnClass(appTheme, 'C')" @click="appTheme = 'C'">C</button>  </div>  <div class="col">  <button class="btn w-100 text-center" :class="btnClass(appTheme, 'D')" @click="appTheme = 'D'">D</button>  </div>  <div class="col">  <button class="btn w-100 text-center" :class="btnClass(appTheme, 'E')" @click="appTheme = 'E'">E</button>  </div>  <div class="col">  <button class="btn w-100 text-center" :class="btnClass(appTheme, 'F')" @click="appTheme = 'F'">F</button>  </div>  </div>  </div>  </div>  </template>  <style scoped>  </style> |

1. 任务总结

成功实现了主题切换功能，用户可以通过按钮选择不同的主题（A、B、C、D、E、F）。

使用Bootstrap的栅格系统和Flexbox，确保按钮在不同屏幕尺寸下能够良好适应和排列。

使用Vue的响应式特性，确保状态变化时界面能够实时更新，提升用户体验。

1. 任务思考

**任务五 实现Photos Slideshow图片上传和照片排序功能**

1. 学习目标

* 学习批量上传图片。
* 学习拖拽图片至上传区域进行上传。
* 学习拖拽图片进行图片列表排序。
* 学习处理文件名。

1. 开发任务

图片文件的上传和图片拖拽排序功能。

将文件名字处理后，作为轮播图的标题。

1. 开发思路

* 在应用程序的功能公共状态管理文件中添加图片的数组。
* 在SettingArea组件中编写图片上传功能，去操作公共状态管理文件当中的值（此任务和后续任务会使用此值）。

1. 实现步骤
   1. 在store.js中添加当前轮播图的图片数组。

|  |
| --- |
| import {computed, ref} from "vue";  export const appMode = ref('RANDOM'); // MANUAL AUTO RANDOM  export const appTheme = ref("F"); // A B C D E F  export const appImages = ref([]); |

* 1. 在src目录下创建helper.js作为项目的工具类，里面分别写入随机生成ID的方法和处理文件名称的方法。

getId() 确保每个上传的图片有ID的标识符，方便后续操作。

convertFilename() 用于处理文件名称达到题目的要求。

|  |
| --- |
| export function getId() {  return ~~(Math.random() \* 10000000);  }  export function convertFilename(name) {  return name  .split(".")[0]  .replaceAll(/-/g, " ")  .split(" ")  .map(item => {  return item.charAt(0).toUpperCase() + item.slice(1).toLowerCase();  })  .join(" ");  } |

* 1. 在SettingArea组件中导入helper的两个方法，然后编写上传图片的功能，引入需要的状态管理和辅助函数。

|  |
| --- |
| import {appImages, appMode, appTheme} from "@/store.js";  import {convertFilename, getId} from "@/helper.js"; |

* 1. 单文件上传方法。

|  |
| --- |
| /\* upload single file by input form \*/  function changeSingleFile(e) {  const file = e.target.files[0];  if (!file) return;  appImages.value.push({  id: getId(),  image: URL.createObjectURL(file),  caption: convertFilename(file.name)  })  e.target.type = "text";  e.target.type = "file";  alert("Import photo");  } |

* 1. 拖拽文件上传方法，遍历文件列表，将每个文件的信息推送到 appImages 中。

|  |
| --- |
| /\* drag and drop many files \*/  function dropFiles(e) {  const files = e.dataTransfer.files;  if (!files.length) return;  [...files].forEach(file => {  appImages.value.push({  id: getId(),  image: URL.createObjectURL(file),  caption: convertFilename(file.name)  })  })  } |

* 1. 在模板中定义操作模式和主题切换的按钮区域，使用 Bootstrap 的栅格系统确保布局响应式。

|  |
| --- |
| <p class="mb-2">Import Photo</p>  <div class="dropArea centerBox flex-grow-1" @dragstart.prevent="" @dragover.prevent="" @drop.prevent="dropFiles">  <div class="">Photo drag and drop</div>  </div>  <label class="mt-2 bg-light rounded p-3">  <small>Single File</small>  <input type="file" class="form-control" @change="changeSingleFile">  </label> |

* 1. SettingArea完整代码。

|  |
| --- |
| <script setup>  import {appImages, appMode, appTheme} from "@/store.js";  import {convertFilename, getId} from "@/helper.js";  /\* selected btn and unselected btn class \*/  function btnClass(a, b) {  if (a === b) {  return "btn-primary";  }  return "btn-fill text-primary";  }  /\* upload single file by input form \*/  function changeSingleFile(e) {  const file = e.target.files[0];  if (!file) return;  appImages.value.push({  id: getId(),  image: URL.createObjectURL(file),  caption: convertFilename(file.name)  })  e.target.type = "text";  e.target.type = "file";  alert("Import photo");  }  /\* drag and drop many files \*/  function dropFiles(e) {  const files = e.dataTransfer.files;  if (!files.length) return;  [...files].forEach(file => {  appImages.value.push({  id: getId(),  image: URL.createObjectURL(file),  caption: convertFilename(file.name)  })  })  }  </script>  <template>  <div class="d-flex flex-column h-100">  <p class="mb-2">Switching operating</p>  <div class="border rounded-pill p-2 d-flex align-items-center mb-4">  <div class="row gx-1 w-100">  <div class="col">  <button class="btn w-100 text-center" :class="btnClass(appMode, 'MANUAL')" @click="appMode = 'MANUAL'">  MANUAL  </button>  </div>  <div class="col">  <button class="btn w-100 text-center" :class="btnClass(appMode, 'AUTO')" @click="appMode = 'AUTO'">AUTO  </button>  </div>  <div class="col">  <button class="btn w-100 text-center" :class="btnClass(appMode, 'RANDOM')" @click="appMode = 'RANDOM'">  RANDOM  </button>  </div>  </div>  </div>  <p class="mb-2">Switching theme</p>  <div class="border rounded-pill p-2 d-flex align-items-center mb-4">  <div class="row gx-1 w-100">  <div class="col">  <button class="btn w-100 text-center" :class="btnClass(appTheme, 'A')" @click="appTheme = 'A'">A</button>  </div>  <div class="col">  <button class="btn w-100 text-center" :class="btnClass(appTheme, 'B')" @click="appTheme = 'B'">B</button>  </div>  <div class="col">  <button class="btn w-100 text-center" :class="btnClass(appTheme, 'C')" @click="appTheme = 'C'">C</button>  </div>  <div class="col">  <button class="btn w-100 text-center" :class="btnClass(appTheme, 'D')" @click="appTheme = 'D'">D</button>  </div>  <div class="col">  <button class="btn w-100 text-center" :class="btnClass(appTheme, 'E')" @click="appTheme = 'E'">E</button>  </div>  <div class="col">  <button class="btn w-100 text-center" :class="btnClass(appTheme, 'F')" @click="appTheme = 'F'">F</button>  </div>  </div>  </div>  <p class="mb-2">Import Photo</p>  <div class="dropArea centerBox flex-grow-1" @dragstart.prevent="" @dragover.prevent="" @drop.prevent="dropFiles">  <div class="">Photo drag and drop</div>  </div>  <label class="mt-2 bg-light rounded p-3">  <small>Single File</small>  <input type="file" class="form-control" @change="changeSingleFile">  </label>  </div>  </template>  <style scoped>  .dropArea {  border: 1px solid var(--bs-dark);  border-radius: .75rem;  }  .centerBox {  width: 100%;  height: 100%;  display: flex;  justify-content: center;  align-items: center;  }  </style> |

* 1. 在OrderingArea组件中引入store里的appImages来获取图片数据。
  2. 在OrderingArea组件中定义响应式变量
     1. moveX 和 moveY 用于跟踪拖动过程中鼠标移动的距离。
     2. getId 保存当前正在拖动的图片的 ID。
     3. targetId 保存目标图片的 ID，以便在放置时进行交换。
  3. 实现拖动功能
     1. grabDown(item)：在鼠标按下时调用，设置当前拖动的图片 ID，并添加 mousemove 和 mouseup 事件监听器。

|  |
| --- |
| /\* mouse down for drag and drop \*/  function grabDown(item) {  getId.value = item.id;  moveX.value = 0;  moveY.value = 0;  window.addEventListener("mousemove", grabMove);  window.addEventListener("mouseup", grabUp);  } |

* + 1. grabMove(e)：在鼠标移动时调用，更新 moveX 和 moveY 的值，反映当前鼠标的位移。

|  |
| --- |
| /\* mouse move for drag and drop \*/  function grabMove(e) {  moveX.value += e.movementX;  moveY.value += e.movementY;  } |

* + 1. grabUp(e)：在鼠标松开时调用，检查是否有目标 ID。如果有，交换当前图片与目标图片的位置，并清理状态。

|  |
| --- |
| /\* mouse up for drag and drop \*/  function grabUp(e) {  if (targetId.value) {  const grabIndex = appImages.value.findIndex(item => item.id === getId.value);  const targetIndex = appImages.value.findIndex(item => item.id === targetId.value);  const tmp = {...appImages.value[targetIndex]};  appImages.value[targetIndex] = appImages.value[grabIndex];  appImages.value[grabIndex] = tmp;  getId.value = null;  targetId.value = null;  moveX.value = 0;  moveY.value = 0;  }  window.removeEventListener("mousemove", grabMove);  window.removeEventListener("mouseup", grabUp);  } |

* + 1. targetOver(item)：当鼠标悬停在其他图片上时调用，记录目标图片的 ID。

|  |
| --- |
| /\* other item mouse over for drag and drop \*/  function targetOver(item) {  targetId.value = item.id;  } |

* + 1. grabStyle(item)：为当前正在拖动的图片应用样式，调整其位置（使用 transform）和层级（zIndex），以确保在拖动时不会被遮挡。

|  |
| --- |
| function grabStyle(item) {/\* moving item style \*/  if (getId.value !== item.id) return;  return {  transform: `translate(${moveX.value}px, ${moveY.value}px)`,  pointerEvents: "none",  zIndex: 2,  }  } |

* + 1. 模板构建，使用一个包含所有图片的列（v-for）来显示 appImages 中的每个图片，在每个图片的外层 <div> 上添加 mousedown 和 mouseover 事件，以便于拖动和识别目标。

|  |
| --- |
| <div class="h-100 d-flex flex-column">  <p class="mb-2">Ordering photos</p>  <div class="flex-grow-1 position-relative">  <div class="scrollBox">  <div class="row gy-2">  <div class="col-12" v-for="item in appImages">  <div class="item" @mousedown="grabDown(item)" @mouseover="targetOver(item)" :style="grabStyle(item)">  <img :src="item.image" alt="image">  <div>{{ item.caption }}</div>  </div>  </div>  </div>  </div>  </div>  </div> |

* + 1. 样式设计，使用 CSS 来设置图片的外观和布局，确保图片具有适当的尺寸、边角圆润、背景色及文本可读性。

|  |
| --- |
| .item {  position: relative;  aspect-ratio: 16/9;  display: flex;  justify-content: flex-start;  align-items: flex-end;  -webkit-user-drag: none;  overflow: hidden;  border-radius: .5rem;  user-select: none;  }  .item div {  background: rgba(0, 0, 0, .5);  color: #fff;  padding: .5rem;  z-index: 2;  left: 0;  top: 0;  width: 100%;  height: 100%;  position: absolute;  }  .item img {  width: 100%;  height: 100%;  border-radius: .5rem;  }  .scrollBox {  position: absolute;  left: 0;  right: 0;  top: 0;  bottom: 0;  overflow-x: hidden;  overflow-y: auto;  } |

1. 任务总结

成功实现了图片上传功能，包括单文件选择和拖放多个文件的支持。

为上传的每个图片生成ID，并将文件名格式化为用户友好的形式。

成功实现了图片拖放排序功能，用户可以通过鼠标拖动图片来调整顺序，提升了用户交互体验。

1. 任务思考

* 这样随机生成ID是否会有重复的风险呢？

**任务六 实现Photos Slideshow主要功能**

1. 学习目标

* 实现轮播图主要轮播功能。
* 将模式切换的值和主题切换的值进行使用。

1. 开发任务

制作个别主题并且实现其功能。

1. 开发思路

* 将每个主题分为一个单独的组件，使用单独的样式。
* 将每个主题中公用的css样式提炼出来作为一个commonTheme.css。
* 把主题组件全部导入到SlideController组件中根据当前主题的状态切换不同的主题组件。
* 在公共管理状态文件中创建一个当前轮播的索引和根据所以获取当前图片的方法。

1. 实现步骤
   1. 在公共状态文件中增加当前轮播索引和根据索引获取图片的方法。

|  |
| --- |
| import {computed, ref} from "vue";  export const appMode = ref('RANDOM'); // MANUAL AUTO RANDOM  export const appTheme = ref("A"); // A B C D E F  export const appImages = ref([]);  export const currentImageIndex = ref(0);  export const currentImage = computed(() => {  return appImages.value[currentImageIndex.value];  }) |

* 1. 创建commonTheme.css放入src/assets/commonTheme.css,并在main.js中引入。

|  |
| --- |
| .themeContainer {  perspective: 1000px;  width: 100%;  height: 100%;  position: relative;  background: #222;  color: #fff;  display: flex;  justify-content: center;  align-items: center;  border-radius: .5rem;  overflow: hidden;  }  .themeContainer img {  width: 100%;  height: 100%;  left: 0;  top: 0;  position: absolute;  border-radius: .5rem;  }  .captionBox {  display: flex;  justify-content: flex-start;  align-items: flex-end;  position: absolute;  left: 0;  right: 0;  top: 0;  bottom: 0;  padding: 1rem;  }  .caption {  background: #fff;  color: #000;  padding: 1rem;  z-index: 1;  font-weight: bold;  border-radius: .5rem;  } |

* 1. 在components文件夹中创建每个主题的组件文件，EffectA...EffectF。

EffectA：主题A直接显⽰照⽚和标题，不带任何效果。

|  |
| --- |
| <script setup>  import {currentImage} from "@/store.js";  </script>  <template>  <img :src="currentImage.image" alt="image">  <div class="captionBox">  <div class="caption">{{currentImage.caption}}</div>  </div>  </template>  <style scoped>  </style> |

* 1. 在SlideController组件中导入所需的主题组件、状态管理（appImages、appMode、appTheme、currentImageIndex）、其他功能（computed、ref、watch）

|  |
| --- |
| import {appImages, appMode, appTheme, currentImageIndex} from "@/store.js";  import {computed, ref, watch} from "vue";  import EffectA from "@/components/EffectA.vue"; |

* 1. 设置幻灯片时间、主题组件Key(每次切换模式或者切换主题或者图片数组发生变化都会+1)、轮播图计时器id（方便clearInterval使用）。

|  |
| --- |
| const SLIDE\_TIME = 3000;  const slideKey = ref(0);  let slideInterval = null; |

* 1. 滑动定时器的设置
     1. 创建 setSlideInterval 函数，负责根据当前模式设置滑动定时器。
     2. 自动播放：每 3 秒切换到下一张图片，如果到达最后一张，则回到第一张。
     3. 随机播放：从当前图片以外的图片中随机选择一张进行展示。

|  |
| --- |
| /\* run slide \*/  function setSlideInterval() {  clearInterval(slideInterval);  currentImageIndex.value = 0;  slideKey.value++;  /\* Auto Playing Type \*/  if (appMode.value === "AUTO") {  slideInterval = setInterval(() => {  /\* check exists images \*/  if (!appImages.value.length) return;  /\* check last turn \*/  if (currentImageIndex.value + 1 === appImages.value.length) {  currentImageIndex.value = 0;  } else {  currentImageIndex.value += 1;  }  }, SLIDE\_TIME)  }  /\* Random Type \*/  if (appMode.value === "RANDOM") {  slideInterval = setInterval(() => {  /\* check exists images \*/  if (!appImages.value.length) return;  const randoms = appImages.value  .map((a, i) => i) // get only index  .filter(item => item !== currentImageIndex.value); // filter without current index  if(!randoms.length) return;  /\* set index \*/  currentImageIndex.value = randoms[~~(Math.random() \* randoms.length)];  }, SLIDE\_TIME)  }  } |

* 1. 手动控制轮播图的切换，监听键盘事件（左箭头和右箭头），仅在手动模式下允许用户切换图片。

|  |
| --- |
| /\* manual control event \*/  addEventListener("keydown", function (e) {  if (appMode.value !== "MANUAL" || !appImages.value.length) return;  if (e.code === "ArrowLeft" && currentImageIndex.value !== 0) {  currentImageIndex.value -= 1;  }  if (e.code === "ArrowRight" && currentImageIndex.value !== appImages.value.length - 1) {  currentImageIndex.value += 1;  }  }) |

* 1. 动态加载主题组件，使用计算属性 themeComponent 根据当前主题动态选择和渲染不同的组件。

|  |
| --- |
| /\* theme component \*/  const themeComponent = computed(() => {  return {  A: EffectA  }[appTheme.value];  }) |

* 1. 监视依赖变化，使用 watch 监听 appMode、appTheme 和 appImages 的变化，及时调用 setSlideInterval 更新定时器。

|  |
| --- |
| watch([appMode, appTheme, appImages], setSlideInterval, {deep: true, immediate: true}); |

* 1. 定义 HTML 结构，包括标题、当前图片索引和图片展示区，当没有图片时，显示提示信息；否则显示当前图片组件。

|  |
| --- |
| <section class="section h-100 d-flex flex-column">  <div class="d-flex justify-content-between align-items-center mb-2">  <div class="d-flex align-items-center">  <h1 class="fw-bold mb-0">SlideView</h1>  <div class="ms-3 badge text-primary bigBadge">  {{ currentImageIndex + 1 }} / {{ appImages.length }}  </div>  </div>  </div>  <div class="flex-grow-1 position-relative">  <div class="staticBox">  <div class="themeContainer">  <div class="text-white" v-if="!appImages.length">Add photos!</div>  <component :is="themeComponent" v-else :key="slideKey"></component>  </div>  </div>  </div>  </section> |

* 1. SlideController完整代码

|  |
| --- |
| <script setup>  import {appImages, appMode, appTheme, currentImageIndex} from "@/store.js";  import {computed, ref, watch} from "vue";  import EffectA from "@/components/EffectA.vue";  const SLIDE\_TIME = 3000;  const slideKey = ref(0);  let slideInterval = null;  /\* run slide \*/  function setSlideInterval() {  clearInterval(slideInterval);  currentImageIndex.value = 0;  slideKey.value++;  /\* Auto Playing Type \*/  if (appMode.value === "AUTO") {  slideInterval = setInterval(() => {  /\* check exists images \*/  if (!appImages.value.length) return;  /\* check last turn \*/  if (currentImageIndex.value + 1 === appImages.value.length) {  currentImageIndex.value = 0;  } else {  currentImageIndex.value += 1;  }  }, SLIDE\_TIME)  }  /\* Random Type \*/  if (appMode.value === "RANDOM") {  slideInterval = setInterval(() => {  /\* check exists images \*/  if (!appImages.value.length) return;  const randoms = appImages.value  .map((a, i) => i) // get only index  .filter(item => item !== currentImageIndex.value); // filter without current index  if(!randoms.length) return;  /\* set index \*/  currentImageIndex.value = randoms[~~(Math.random() \* randoms.length)];  }, SLIDE\_TIME)  }  }  /\* manual control event \*/  addEventListener("keydown", function (e) {  if (appMode.value !== "MANUAL" || !appImages.value.length) return;  if (e.code === "ArrowLeft" && currentImageIndex.value !== 0) {  currentImageIndex.value -= 1;  }  if (e.code === "ArrowRight" && currentImageIndex.value !== appImages.value.length - 1) {  currentImageIndex.value += 1;  }  })  /\* theme component \*/  const themeComponent = computed(() => {  return {  A: EffectA  }[appTheme.value];  })  watch([appMode, appTheme, appImages], setSlideInterval, {deep: true, immediate: true});  </script>  <template>  <section class="section h-100 d-flex flex-column">  <div class="d-flex justify-content-between align-items-center mb-2">  <div class="d-flex align-items-center">  <h1 class="fw-bold mb-0">SlideView</h1>  <div class="ms-3 badge text-primary bigBadge">  {{ currentImageIndex + 1 }} / {{ appImages.length }}  </div>  </div>  </div>  <div class="flex-grow-1 position-relative">  <div class="staticBox">  <div class="themeContainer">  <div class="text-white" v-if="!appImages.length">Add photos!</div>  <component :is="themeComponent" v-else :key="slideKey"></component>  </div>  </div>  </div>  </section>  </template>  <style scoped>  .bigBadge {  font-size: 1.2rem;  }  </style> |

1. 任务总结

幻灯片展示：根据不同的模式（自动、随机、手动）实现了幻灯片的切换效果。

主题切换：支持多种主题的切换，用户可以通过选择不同的主题来改变幻灯片的显示效果。

状态管理：利用 Vue 的响应式系统，确保所有状态变化（如图片、模式、主题）都能正确反映在界面上。

1. 任务思考

* 为什么要加slideKey作为组件的key不加会怎么样？

**项目五**

**世界技能大赛典型网站设计与开发**

**后端应用程序开发**

**项目需求**

我们将为一个办公室管理员（简称为“管理员”）创建一个管理系统，来管理一些在法国制造的产品。产品属于公司。管理员还将管理关联的公司数据。只有管理员可以查看和管理这些公司、产品清单和编辑视图。

每个产品记录都有一个唯一的标识符号，我们称之为全球贸易项目编号（GTIN）。作为背景信息，GTIN是由国际组织GS1开发的。

另一方面，也有面向公众的页面。它们是：

* 公共GTIN批量验证页面
* 公共产品页面

在这个项目中，我们创建了一个公司和产品管理系统来管理这些法国的产品。

它包含用于列出和编辑记录的管理网页。和JSON API，也用于查询和读取数据。

**管理员访问**

登录页面要求使用密码短语来进行身份验证。可以使用密码短语“管理”来访问管理管理系统。

作为一个3个小时工作的原型产品。我们不需要强大的身份验证。该管理系统使用简单的基于密码短语的检查。

其任务是创建一个公司和产品管理系统。

考虑到时间常数，我们将重点关注公司和产品记录管理的原型，以及在公共页面上显示产品的概念验证。

访问产品编辑和管理功能而不登录会导致401错误。

**管理公司**

只有管理员可以上市和管理公司。

管理员可以点击公司列表并查看特定的公司。

在每个公司页面中，管理员可以查看相关的产品。

管理员还可以创建新的公司或更新现有的公司信息。

以下是我们希望为每个公司存储和管理的数据：

* company name
* company address
* company telephone number
* company email address
* owner information:
  + - * owner's name
      * owner's mobile number
      * owner's email address
* contact information:
  + - * contact's name
      * contact's mobile number
      * contact's email address

**停用公司**

管理员可以将一个公司标记为已停用的公司。

当公司停用时，所有相关产品都被标记为隐藏。

应该有一个独立的上市未上市的公司的名单。

任何人都不应该能够从网络界面上删除任何公司的记录。

**管理产品**

每个产品都有不同的领域，值得注意的是GTIN。

在这个项目中，我们将GTIN简化为具有13或14位数字的任意数字。它可以是任何数字的序列，只要它们对每个产品都是唯一的。请在提交表单后，通过服务器端验证来验证此GTIN格式。

**上市产品**

管理员应该能够上市和管理产品。

当管理员访问/XX\_module\_b/products时，管理员可以看到所有产品的列表。

管理员可以通过单击列表中的产品记录或通过访问以下路径访问和管理特定产品：/XX\_module\_b/products/[GTIN].[GTIN]是GTIN字段的占位符。

例如，给定一个使用GTIN 3000123456789的产品，访问/XX\_module\_b/products/3000123456789将是该产品的产品管理页面。

**隐藏和删除产品**

管理员可以标记产品为隐藏的。

或者，当关联的公司被标记为停用时，一个产品就会变得隐藏起来。

管理员可以永久删除隐藏的产品。

**创建新产品**

管理员可以通过访问/products/new来查看创建的产品表单，并可以将新创建的产品保存到数据库。

产品有两种信息语言，英语和法语。

对于产品，我们要存储以下数据：

* name
* name in French
* GTIN (Global Trade Item Number)
* description, can be multiple lines of text.
* description in French
* product brand name
* product country of origin
* product gross weight (with packaging)
* product net content weight
* product weight unit

**产品图像上传**

每个产品都可能有一个图像相关联。管理员可以上传和更改此图片，或管理员可以删除上传的图片。

当没有上传图像时，会有一个默认的占位符图像。

**JSON API输出**

当访问GET /XX\_module\_b/products.json时，数据可以以JSON格式输出。当列出产品时，API将显示一个分页结构。

单个产品可以通过GET /XX\_module\_b/products/[GTIN].json的JSON进行查询，其中的GTIN是动态的。

当访问不存在的产品或隐藏的产品时，产品API返回404网络状态。

产品列表JSON允许使用关键字进行查询，通过GET /XX\_module\_b/products.json?query=KEYWORD。该列表应该显示包含关键字的名称、法语名称、描述或法语描述的产品。

**JSON API示例输出**

示例公司输出

|  |
| --- |
| {  "companyName": "Euro Expo",  "companyAddress": " Boulevard de l'Europe, 69680 Chassieu, France",  "companyTelephone": "+33 1 41 56 78 00",  "companyEmail": "mail.customerservice.hdq@example.com",  "owner": {  "name": "Benjamin Smith",  "mobileNumber": "+33 6 12 34 56 78",  "email": "b.smith@example.com "  },  "contact": {  "name": "Marie Dubois",  "mobileNumber": "+33 6 98 76 54 32",  "email": "m.dubois@example.com "  }  } |

产品输出示例

|  |
| --- |
| {  "name": {  "en": "Organic Apple Juice",  "fr": "Jus de pomme biologique"  },  "description": {  "en": "Our organic apple juice is pressed from 100% fresh organic apples, with no added sugars or preservatives. Rich in vitamin C and  antioxidants, it's an ideal choice for your daily healthy diet.",  "fr": "Notre jus de pomme biologique est pressé à partir de 100% de pommes biologiques fraîches, sans sucre ajouté ni conservateurs.  Riche en vitamine C et en antioxydants, c'est le choix idéal pour votre alimentation quotidienne saine."  },  "gtin": "03000123456789",  "brand": "Green Orchard",  "countryOfOrigin": "France",  "weight": {  "gross": 1.1,  "net": 1.0,  "unit": "L"  },  "company": {  "companyName": "Euro Expo",  "companyAddress": " Boulevard de l'Europe, 69680 Chassieu, France",  "companyTelephone": "+33 1 41 56 78 00",  "companyEmail": "mail.customerservice.hdq@example.com",  "owner": {  "name": "Benjamin Smith",  "mobileNumber": "+33 6 12 34 56 78",  "email": "b.smith@example.com "  },  "contact": {  "name": "Marie Dubois",  "mobileNumber": "+33 6 98 76 54 32",  "email": "m.dubois@example.com "  }  }  } |

示例产品JSON

|  |
| --- |
| {  "data": [  {  (Same as single product JSON structure)  },  {  (Same as single product JSON structure)  },  ],  "pagination": {  "current\_page": 1,  "total\_pages": 5,  "per\_page": 10,  "next\_page\_url": "http://wsXX.worldskills.org/XX\_module\_b/products.json?page=2",  "prev\_page\_url": null  }  } |

**面向公众的页面**

有两个面向公共页面的页面，分别是GTIN批量验证页面和公共产品页面。

**公共GTIN批量验证页面**

如果给定的GTIN编号已注册并有效，则有一个页面用于批量验证。任何用户都可以输入多个GTIN代码，并提交以查看验证结果。

当GTIN存在于数据库中且不被隐藏时，它是有效的。

该页面使用一个文本区域，允许用户批量输入GTIN号码。这些数字用换行符分隔。然后，页面应该检查每个GTIN号，并显示一个结果列表，显示每个GTIN号是否有效。

如果所有给定的GTIN数字都是有效的，那么在结果页面的顶部会显示一个分隔的“All valid”文本和绿色的勾号。



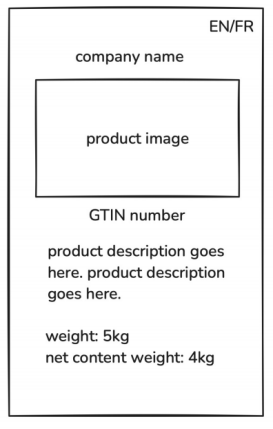
**公共产品页面**

可以通过访问/XX\_module\_b-/01/[GTIN]来访问产品的公共页面。

其中的“01”是静态的。公共产品页面显示了以下必填字段。并且是在移动友好的布局。

需要显示的字段为：公司名称、产品名称、GTIN编号、产品描述、产品图像、重量与单位、净内容重量与单位。

用户可以选择公共产品页面的英语和法语，该页面的lang属性、法语部分和英语部分已正确配置。



**数据库和模型创建的考虑因素**

我们想将这些公司和产品的数据存储在数据库中。

请考虑到产品的多语言信息的灵活性。

请提供数据库转储文件。数据库转储中应该包含FK约束和正确的列。柱的类型定义应保持合理。

请提供ER图模式。

请考虑数据库的正常形式。

GTIN字段应该被编入索引。

已经提供了一些样本数据。请使用它们，并在数据库中有一些数据以方便评估。

**任务一** **设计数据库表结构**

1. 学习目标

* 掌握MySQL数据库连接方式。
* 掌握MySQL数据库的创建方式。
* 掌握MySQL数据表的设计结构。

1. 开发任务
2. 根据实体类model创建MySQL数据库，设计数据表及表结构。以下为必须的实体字段。
3. Users（管理员）：可以登录到管理员门户并使用其所有功能。

* name 用户名
* mobile 手机号码
* email 邮箱
* created\_at 创建时间
* updated\_at 更新时间

1. Brands（品牌）：每个产品需要绑定一个品牌。

* name 名称
* created\_at 创建时间
* updated\_at 更新时间

1. Countries（国家）：每个产品需要绑定一个国家。

* name 名称
* created\_at 创建时间
* updated\_at 更新时间

1. Companies（公司）：每个产品需要绑定一个公司。

* name公司名称
* address 地址
* telephone 联系电话
* email 电子邮件
* owner\_id 公司法人id
* contact\_id 联系人id
* deleted\_at 删除时间
* created\_at 创建时间
* updated\_at 更新时间

1. Langusges（语言）：每个产品的名称和描述需要设置的语言类型。

* code 编码（目前分为en和fr）
* created\_at 创建时间
* updated\_at 更新时间

1. Products（产品）：核心产品。

* GTIN 全球贸易项目编号
* brand\_id 品牌id
* country\_id 国家id
* product\_image\_id 产品图片id
* company\_id 公司id
* gross\_weight 总重量
* net\_weight 净重量
* weight\_unit 重量单位
* deleted\_at 删除时间
* created\_at 创建时间
* updated\_at 更新时间

1. Product\_names（产品名称）：每个产品的名称，包含英语和法语名称。

* Language\_id 语言id
* product\_id 产品id
* contents 产品名称
* created\_at 创建时间
* updated\_at 更新时间

1. Product\_descriptions（产品描述）：每个产品的描述，包含英语和法语描述。

* Language\_id 语言id
* product\_id 产品id
* contents 产品描述内容
* created\_at 创建时间
* updated\_at 更新时间

1. 开发思路
2. 创建MySQL数据库：

* 在 Laravel .env 文件中设置数据库连接参数,进行数据库连接并创建数据库模型和迁移文件和控制类。

1. 设计数据表字段及表结构
2. 数据填入
3. 实现步骤
4. 配置数据库.env

|  |
| --- |
| DB\_CONNECTION=mysql  DB\_HOST=127.0.0.1  DB\_PORT=3306  DB\_DATABASE=example\_db  DB\_USERNAME=root  DB\_PASSWORD=your\_password |

1. 创建数据模型和资源控制器

* 用户（User）模型

|  |
| --- |
| php artisan make:model User -mr |

* 语言（Language）模型

|  |
| --- |
| php artisan make:model Language -m |

* 品牌（Brand）模型

|  |
| --- |
| php artisan make:model Brand -m |

* 国家（Country）模型

|  |
| --- |
| php artisan make:model Country -m |

* 产品图片（ProductImage）模型

|  |
| --- |
| php artisan make:model ProductImage -m |

* 公司（Company）模型

|  |
| --- |
| php artisan make:model Company -m |

* 产品（Product）模型

|  |
| --- |
| php artisan make:model Product -m |

* 产品名称（ProductName）模型

|  |
| --- |
| php artisan make:model ProductName -m |

* 产品描述（ProductDescription）模型

|  |
| --- |
| php artisan make:model ProductDescription -m |

1. 设计数据表字段及表结构

* User

|  |
| --- |
| <?php  use Illuminate\Database\Migrations\Migration; use Illuminate\Database\Schema\Blueprint; use Illuminate\Support\Facades\Schema;  return new class extends Migration {  /\*\*  \* Run the migrations.  \*/  public function up(): void  {  Schema::create('users', function (Blueprint $table) {  $table->id();  $table->string('name');  $table->string('mobile');  $table->string('email');  $table->timestamps();  });  }   /\*\*  \* Reverse the migrations.  \*/  public function down(): void  {  Schema::dropIfExists('users');  } }; |

* Language

|  |
| --- |
| <?php  use Illuminate\Database\Migrations\Migration; use Illuminate\Database\Schema\Blueprint; use Illuminate\Support\Facades\Schema;  return new class extends Migration {  /\*\*  \* Run the migrations.  \*/  public function up(): void  {  Schema::create('languages', function (Blueprint $table) {  $table->id();  $table->string("code");  $table->timestamps();  });  }   /\*\*  \* Reverse the migrations.  \*/  public function down(): void  {  Schema::dropIfExists('languages');  } }; |

* Brand

|  |
| --- |
| <?php  use Illuminate\Database\Migrations\Migration; use Illuminate\Database\Schema\Blueprint; use Illuminate\Support\Facades\Schema;  return new class extends Migration {  /\*\*  \* Run the migrations.  \*/  public function up(): void  {  Schema::create('brands', function (Blueprint $table) {  $table->id();  $table->string("name");  $table->timestamps();  });  }   /\*\*  \* Reverse the migrations.  \*/  public function down(): void  {  Schema::dropIfExists('brands');  } }; |

* Country

|  |
| --- |
| <?php  use Illuminate\Database\Migrations\Migration; use Illuminate\Database\Schema\Blueprint; use Illuminate\Support\Facades\Schema;  return new class extends Migration {  /\*\*  \* Run the migrations.  \*/  public function up(): void  {  Schema::create('countries', function (Blueprint $table) {  $table->id();  $table->string("name");  $table->timestamps();  });  }   /\*\*  \* Reverse the migrations.  \*/  public function down(): void  {  Schema::dropIfExists('countries');  } }; |

* ProductImage

|  |
| --- |
| <?php  use Illuminate\Database\Migrations\Migration; use Illuminate\Database\Schema\Blueprint; use Illuminate\Support\Facades\Schema;  return new class extends Migration {  /\*\*  \* Run the migrations.  \*/  public function up(): void  {  Schema::create('product\_images', function (Blueprint $table) {  $table->id();  $table->string("name");  $table->timestamps();  });  }   /\*\*  \* Reverse the migrations.  \*/  public function down(): void  {  Schema::dropIfExists('product\_images');  } }; |

* Company

|  |
| --- |
| <?php  use Illuminate\Database\Migrations\Migration; use Illuminate\Database\Schema\Blueprint; use Illuminate\Support\Facades\Schema;  return new class extends Migration {  /\*\*  \* Run the migrations.  \*/  public function up(): void  {  Schema::create('companies', function (Blueprint $table) {  $table->id();  $table->string("name");  $table->string("address");  $table->string("telephone");  $table->string("email");  $table->foreignId("owner\_id")->constrained("users")  ->cascadeOnDelete()->cascadeOnUpdate();  $table->foreignId("contact\_id")->constrained("users")  ->cascadeOnDelete()->cascadeOnUpdate();  $table->softDeletes();  $table->timestamps();  });  }   /\*\*  \* Reverse the migrations.  \*/  public function down(): void  {  Schema::dropIfExists('companies');  } }; |

* Product

|  |
| --- |
| <?php  use Illuminate\Database\Migrations\Migration; use Illuminate\Database\Schema\Blueprint; use Illuminate\Support\Facades\Schema;  return new class extends Migration {  /\*\*  \* Run the migrations.  \*/  public function up(): void  {  Schema::create('products', function (Blueprint $table) {  $table->id();  $table->string("GTIN")->unique();  $table->foreignIdFor(\App\Models\Brand::class)->constrained()  ->cascadeOnUpdate()->cascadeOnUpdate();  $table->foreignIdFor(\App\Models\Country::class)  ->constrained()->cascadeOnUpdate()->cascadeOnUpdate();  $table->foreignIdFor(\App\Models\ProductImage::class)  ->nullable()->constrained()->cascadeOnUpdate()->nullOnDelete();  $table->foreignIdFor(\App\Models\Company::class)  ->constrained()->cascadeOnUpdate()->cascadeOnUpdate();  $table->decimal("gross\_weight");  $table->decimal("net\_weight");  $table->string("weight\_unit");  $table->softDeletes();  $table->timestamps();  });  }   /\*\*  \* Reverse the migrations.  \*/  public function down(): void  {  Schema::dropIfExists('products');  } }; |
|  |

* ProductName

|  |
| --- |
| <?php  use Illuminate\Database\Migrations\Migration; use Illuminate\Database\Schema\Blueprint; use Illuminate\Support\Facades\Schema;  return new class extends Migration {  /\*\*  \* Run the migrations.  \*/  public function up(): void  {  Schema::create('product\_names', function (Blueprint $table) {  $table->id();  $table->foreignIdFor(\App\Models\Language::class)  ->constrained()->cascadeOnUpdate()->cascadeOnDelete();  $table->foreignIdFor(\App\Models\Product::class)  ->constrained()->cascadeOnUpdate()->cascadeOnDelete();  $table->string("contents");  $table->timestamps();  });  }   /\*\*  \* Reverse the migrations.  \*/  public function down(): void  {  Schema::dropIfExists('product\_names');  } }; |

* ProductDescription

|  |
| --- |
| <?php  use Illuminate\Database\Migrations\Migration; use Illuminate\Database\Schema\Blueprint; use Illuminate\Support\Facades\Schema;  return new class extends Migration {  /\*\*  \* Run the migrations.  \*/  public function up(): void  {  Schema::create('product\_descriptions', function (Blueprint $table) {  $table->id();  $table->foreignIdFor(\App\Models\Language::class)  ->constrained()->cascadeOnUpdate()->cascadeOnDelete();  $table->foreignIdFor(\App\Models\Product::class)  ->constrained()->cascadeOnUpdate()->cascadeOnDelete();  $table->text("contents");  $table->timestamps();  });  }   /\*\*  \* Reverse the migrations.  \*/  public function down(): void  {  Schema::dropIfExists('product\_descriptions');  } }; |

1. 填充数据库数据模型设计与关系定义

* Brand 模型

|  |
| --- |
| <?php  namespace App\Models;  use Illuminate\Database\Eloquent\Factories\HasFactory; use Illuminate\Database\Eloquent\Model;  class Brand extends Model {  use HasFactory;   protected $guarded = []; } |

* Company 模型

|  |
| --- |
| <?php  namespace App\Models;  use Illuminate\Database\Eloquent\Factories\HasFactory; use Illuminate\Database\Eloquent\Model; use Illuminate\Database\Eloquent\SoftDeletes;  class Company extends Model {  use HasFactory, SoftDeletes;   protected $guarded = [];   public function products()  {  return $this->hasMany(Product::class)->withTrashed();  }   public function owner()  {  return $this->belongsTo(User::class, "owner\_id");  }   public function contact()  {  return $this->belongsTo(User::class, "contact\_id");  } } |

* Country 模型

|  |
| --- |
| <?php  namespace App\Models;  use Illuminate\Database\Eloquent\Factories\HasFactory; use Illuminate\Database\Eloquent\Model;  class Country extends Model {  use HasFactory;   protected $guarded = []; } |

* Language 模型

|  |
| --- |
| <?php  namespace App\Models;  use Illuminate\Database\Eloquent\Factories\HasFactory; use Illuminate\Database\Eloquent\Model;  class Language extends Model {  use HasFactory; } |

* Product 模型

|  |
| --- |
| <?php  namespace App\Models;  use Illuminate\Database\Eloquent\Factories\HasFactory; use Illuminate\Database\Eloquent\Model; use Illuminate\Database\Eloquent\SoftDeletes; use Illuminate\Support\Facades\Storage;  class Product extends Model {  use HasFactory, SoftDeletes;   protected $guarded = [];   public function getIsHiddenAttribute()  {  return $this->trashed() || $this->company->trashed();  }   public function getImageUrlAttribute()  {  if(!$this->productImage) {  return asset("public/placeholder.jpg");  }  return Storage::url($this->productImage->name);  }   public function getNameAttribute()  {  $en = Language::query()->where("code", "en")->first();  return $this->names()->where("language\_id", $en->id)->first()->contents;  }   public function getNameFrAttribute()  {  $en = Language::query()->where("code", "fr")->first();  return $this->names()->where("language\_id", $en->id)->first()->contents;  }   public function getDescriptionAttribute()  {  $en = Language::query()->where("code", "en")->first();  return $this->description()->where("language\_id", $en->id)->first()->contents;  }   public function getDescriptionFrAttribute()  {  $en = Language::query()->where("code", "fr")->first();  return $this->description()->where("language\_id", $en->id)->first()->contents;  }   public function names()  {  return $this->hasMany(ProductName::class);  }   public function description()  {  return $this->hasMany(ProductDescription::class);  }   public function company()  {  return $this->belongsTo(Company::class)->withTrashed();  }   public function productImage()  {  return $this->belongsTo(ProductImage::class);  }   public function brand()  {  return $this->belongsTo(Brand::class);  }   public function country()  {  return $this->belongsTo(Country::class);  } } |

* ProductDescription 模型

|  |
| --- |
| <?php  namespace App\Models;  use Illuminate\Database\Eloquent\Factories\HasFactory; use Illuminate\Database\Eloquent\Model;  class ProductDescription extends Model {  use HasFactory;  protected $guarded = []; } |

* ProductImage模型

|  |
| --- |
| <?php  namespace App\Models;  use Illuminate\Database\Eloquent\Factories\HasFactory; use Illuminate\Database\Eloquent\Model;  class ProductImage extends Model {  use HasFactory;   protected $guarded =[]; } |

* ProductName 模型

|  |
| --- |
| <?php  namespace App\Models;  use Illuminate\Database\Eloquent\Factories\HasFactory; use Illuminate\Database\Eloquent\Model;  class ProductName extends Model {  use HasFactory;   protected $guarded = []; } |

* User 模型

|  |
| --- |
| <?php  namespace App\Models;  // use Illuminate\Contracts\Auth\MustVerifyEmail; use Illuminate\Database\Eloquent\Factories\HasFactory; use Illuminate\Foundation\Auth\User as Authenticatable; use Illuminate\Notifications\Notifiable; use Laravel\Sanctum\HasApiTokens;  class User extends Authenticatable {  use HasApiTokens, HasFactory, Notifiable;   protected $guarded =[]; } |

1. 在DatabaseSeeder类中写入数据填充

|  |
| --- |
| <?php  namespace Database\Seeders;  // use Illuminate\Database\Console\Seeds\WithoutModelEvents; use App\Models\Brand; use App\Models\Company; use App\Models\Country; use App\Models\Language; use App\Models\Product; use App\Models\ProductDescription; use App\Models\ProductName; use App\Models\User; use Illuminate\Database\Seeder;  class DatabaseSeeder extends Seeder {  /\*\*  \* Seed the application's database.  \*/  public function run(): void  {  /\* Create the language \*/  $en = Language::create([  "code"=> "en",  ]);  $fr = Language::create([  "code"=> "fr",  ]);   /\* Create the company \*/  $companyData = $this->getCsv(base\_path("database/seeders/companies.csv"));  foreach($companyData as $item) {  $owner = User::create([  "name" => $item[4],  "mobile" => $item[5],  "email" => $item[6],  ]);  $contact = User::create([  "name" => $item[7],  "mobile" => $item[8],  "email" => $item[9],  ]);   Company::create([  "name" => $item[0],  "address" => $item[1],  "telephone" => $item[2],  "email" => $item[3],  "owner\_id" => $owner->id,  "contact\_id" => $contact->id,  ]);  }   /\* Create the products \*/  $productData = $this->getCsv(base\_path("database/seeders/products.csv"));  foreach($productData as $item) {  /\* Check already exists GTIN \*/  $checkAlreadyProduct = Product::query()->where("GTIN", $item[0])->exists();  if($checkAlreadyProduct) {  continue;  }   /\* brand \*/  $brand = Brand::where("name", $item[5])->first();  if(!$brand) {  $brand = Brand::create([  "name"=> $item[5]  ]);  }  /\* country of origin \*/  $country = Country::where("name", $item[6])->first();  if(!$country) {  $country = Country::create([  "name"=> $item[6]  ]);  }   $company = Company::query()->inRandomOrder()->first();  $product = Product::create([  "GTIN" => $item[0],  "brand\_id"=> $brand->id,  "country\_id"=> $country->id,  "gross\_weight" => $item[7],  "net\_weight" => $item[8],  "weight\_unit" => $item[9],  "company\_id" => $company->id,  ]);  /\* name \*/  ProductName::create([  "product\_id" => $product->id,  "language\_id" => $en->id,  "contents" => $item[1]  ]);  ProductName::create([  "product\_id" => $product->id,  "language\_id" => $fr->id,  "contents" => $item[4]  ]);  /\* description \*/  ProductDescription::create([  "product\_id" => $product->id,  "language\_id" => $en->id,  "contents" => $item[2]  ]);  ProductDescription::create([  "product\_id" => $product->id,  "language\_id" => $fr->id,  "contents" => $item[3]  ]);  }  }   /\* Get csv contents for data import \*/  private function getCsv($path)  {  /\* get file contents \*/  $str = file\_get\_contents($path);  /\* split line \*/  $data = explode("\n", $str); // dd(str\_getcsv($data[0]));  /\* delete title \*/  array\_shift($data);  /\* split item \*/  $data = collect($data)->map(function ($row) {  return str\_getcsv($row);  });   return $data;  } } |

* 终端输入指令进行数据填充

|  |
| --- |
| php artisan db:seed |

1. 任务总结

在任务一中，我们成功创建了MySQL数据库，并设计了数据库表结构，同时根据任务要求，填入了一些基础数据。

本任务中的开发过程展示了如何连接MySQL数据库、创建数据库、设计数据库表结构和数据表之间的关联关系。通过本任务的完成，开发者掌握了MySQL数据库的核心技术，为后续开发部分提供了数据支撑。

1. 任务思考

* 在本任务中，通过为每个表设置外键约束（如 foreignId 和 constrained）实现了表之间的关联。使用级联操作（cascadeOnUpdate 和 cascadeOnDelete）保证了相关数据的同步更新和删除。这种设计是否在数据量很大的情况下能高效执行？是否有其他方法能进一步优化数据的一致性和完整性？
* 通过 CSV 文件和 Seeder 类实现数据填充，虽然提高了开发效率，但在实际使用中是否会出现 CSV 文件格式不统一、字段缺失等问题？如果数据源是实时变化的，是否可以设计更灵活的机制来处理数据导入，确保数据的动态更新和准确性？

**任务二 实现Lravel Web路由**

1. 学习目标

* 掌握如何在 Laravel 框架中设计和实现 Web 路由。
* 理解如何利用中间件保护路由，确保登录用户才能访问特定资源。
* 学会如何为资源（如公司、产品）设计 RESTful 路由并进行增删改查操作。
* 理解如何处理文件上传和删除操作，尤其是与产品图片相关的操作。

1. 开发任务
2. Web 路由设计

* 通过 Laravel 路由定义页面展示、API 接口及资源管理。

1. 用户认证与授权

* 使用 Laravel 中的认证中间件来保护敏感路由，确保只有登录用户能够访问。

1. 资源管理

* 实现公司与产品的增删改查功能，且支持软删除 。
* 实现产品图片的上传与删除功能。

1. 开发思路
2. 路由分层设计

* 需要认证的路由（如公司、产品管理）使用 auth 中间件 。

1. 资源控制器的使用

* 利用 Laravel 的 resource 路由简化 RESTful 路由的编写。例如：Route::resource("company", CompanyController::class) 。

1. 软删除与恢复

* 使用 Eloquent 的软删除功能，允许对删除的数据进行恢复。

1. 资图片上传与管理

* 使用 Laravel 的文件上传功能，将图片与产品关联，并提供删除功能 。

1. 实现步骤
2. 设计路由

在 routes/web.php 文件中，你需要定义所有的路由。路由将决定应用程序如何处理用户请求。具体的步骤如下

* 登录与注销路由:

|  |
| --- |
| // 登录页面  Route::get("/login",[\App\Http\Controllers\AuthController::class,"loginPage"])->name("auth.login.page");  // 登录操作  Route::post("/login",[\App\Http\Controllers\AuthController::class,"loginAction"])->name("auth.login.action");  // 注销操作  Route::get("/logout",[\App\Http\Controllers\AuthController::class,"logout"])->name("auth.logout"); |

* 公共页面路由 :

这部分处理所有公开的页面访问，如首页和产品详情页面。

|  |
| --- |
| // 首页  Route::get("/", [\App\Http\Controllers\PublicController::class, "index"])->name("public.index");  // 根据 GTIN 显示单个产品  Route::get("/01/{product:GTIN}",[\App\Http\Controllers\PublicController::class,"show"])->name("public.show"); |

* 需要登录的路由（保护路由） :

使用中间件 auth 来保护公司和产品相关的路由，确保只有登录的用户才能访问。

|  |
| --- |
| Route::middleware(["auth"])->group(function(){  // 公司管理路由  Route::resource("company",\App\Http\Controllers\CompanyController::class)->withTrashed()->except(["destroy"]);  Route::delete("/company/{company}",[\App\Http\Controllers\CompanyController::class, "destroy"])->name("company.destroy")->withTrashed();  // 产品管理路由  Route::get("/products",[\App\Http\Controllers\ProductController::class,"index"])->name("product.index");  Route::get("/products/new",[\App\Http\Controllers\ProductController::class,"create"])->name("product.create");  Route::post("/products/new",[\App\Http\Controllers\ProductController::class,"store"])->name("product.store");  Route::get("/products/{product:GTIN}",[\App\Http\Controllers\ProductController::class,"show"])->name("product.show")->withTrashed();  Route::put("/products/{product:GTIN}",[\App\Http\Controllers\ProductController::class,"update"])->name("product.update")->withTrashed();  Route::delete("/products/{product:GTIN}", [\App\Http\Controllers\ProductController::class, "destroy"])->name("product.destroy")->withTrashed();  // 产品图片管理  Route::post("/products/{product:GTIN}/image",[\App\Http\Controllers\ProductImageController::class, "upload"])->name("product.image.upload")->withTrashed();  Route::delete("/products/{product:GTIN}/image",[\App\Http\Controllers\ProductImageController::class, "destroy"])->name("product.image.destroy")->withTrashed();  }); |

1. 创建控制器

创建我们所需要的其他资源控制器

* AuthController 负责处理登录、登出和显示登录页面

|  |
| --- |
| php artisan make:controller AuthController |

* PublicController 负责展示公开页面（如首页和产品详情)

|  |
| --- |
| php artisan make:controller PublicController |

* ApiController 负责处理 API 请求，提供产品数据

|  |
| --- |
| php artisan make:controller ApiController |

1. 实现登录功能

|  |
| --- |
| <?php  namespace App\Http\Controllers;  use Illuminate\Http\Request;  class AuthController extends Controller {  public function loginPage()  {  /\* return login view \*/  return view("auth.login");  }   public function loginAction(Request $request)  {  /\* check admin password \*/  if($request->password !== "admin")) {  return redirect()->back()->with("alert", "Wrong password");  }  /\* save login session \*/  session()->push("login", true);  /\* redirect company list page \*/  return redirect()->route("company.index")->with("alert", "Success login");  }   public function logout()  {  /\* delete login session \*/  session()->remove("login");  /\* redirect login page \*/  return redirect()->route("auth.login.page")->with("alert", "Success logout");  } } |

1. 实现公司管理功能

|  |
| --- |
| <?php  namespace App\Http\Controllers;  use App\Models\Company; use App\Models\User; use Illuminate\Http\Request;  class CompanyController extends Controller {  /\*\*  \* Display a listing of the resource.  \*/  public function index(Request $request)  {  /\* get list type: all || delete(deactivated) \*/  $type = $request->get("type", "all");  /\* get company list \*/  if ($type === "all") {  $items = Company::query()->withTrashed()->get();  } else {  $items = Company::query()->onlyTrashed()->get();  }  /\* return company index view \*/  return view("company.index", compact(["items", "type"]));  }   /\*\*  \* Show the form for creating a new resource.  \*/  public function create()  {  /\* return company create view \*/  return view("company.create");  }   /\*\*  \* Store a newly created resource in storage.  \*/  public function store(Request $request)  {  /\* insert row \*/  $owner = User::create([  "name" => $request->owner\_name,  "mobile" => $request->owner\_mobile,  "email" => $request->owner\_email,  ]);  $contact = User::create([  "name" => $request->contact\_name,  "mobile" => $request->contact\_mobile,  "email" => $request->contact\_email,  ]);  $company = Company::create([  "name" => $request->name,  "address" => $request->address,  "telephone" => $request->telephone,  "email" => $request->email,  "owner\_id" => $owner->id,  "contact\_id" => $contact->id,  ]);  /\* redirect company detail page \*/  return redirect()->route("company.show", $company)->with("alert", "Success create company");  }   /\*\*  \* Display the specified resource.  \*/  public function show(Company $company)  {  /\* return company show view \*/  return view("company.show", compact(["company"]));  }   /\*\*  \* Show the form for editing the specified resource.  \*/  public function edit(Company $company)  {  /\* return company edit view \*/  return view("company.edit", compact(["company"]));  }   /\*\*  \* Update the specified resource in storage.  \*/  public function update(Request $request, Company $company)  {  /\* Check changed owner data \*/  $owner = $company->owner;  if(  $owner->name !== $request->owner\_name ||  $owner->mobile !== $request->owner\_mobile ||  $owner->email !== $request->owner\_email  ) {  $owner = User::create([  "name" => $request->owner\_name,  "mobile" => $request->owner\_mobile,  "email" => $request->owner\_email,  ]);  }  /\* Check changed contact data \*/  $contact = $company->contact;  if(  $contact->name !== $request->contact\_name ||  $contact->mobile !== $request->contact\_mobile ||  $contact->email !== $request->contact\_email  ) {  $contact = User::create([  "name" => $request->contact\_name,  "mobile" => $request->contact\_mobile,  "email" => $request->contact\_email,  ]);  }  $company->update([  "name" => $request->name,  "address" => $request->address,  "telephone" => $request->telephone,  "email" => $request->email,  "owner\_id" => $owner->id,  "contact\_id" => $contact->id,  ]);  /\* redirect company detail page \*/  return redirect()->route("company.show", $company)->with("alert", "Success update company");  }   /\*\*  \* Remove the specified resource from storage.  \*/  public function destroy(Company $company)  {  /\* Toggle company activate status \*/  if($company->trashed()){  $company->restore();  } else {  $company->delete();  }  /\* Redirect company detail page \*/  return redirect()->route("company.show", $company)->with("alert", "Success change company activate status");  } } |

1. 实现产品管理功能

|  |
| --- |
| <?php  namespace App\Http\Controllers;  use App\Models\Brand; use App\Models\Company; use App\Models\Country; use App\Models\Language; use App\Models\Product; use App\Models\ProductDescription; use App\Models\ProductImage; use App\Models\ProductName; use Illuminate\Http\Request;  class ProductController extends Controller {  /\*\*  \* Display a listing of the resource.  \*/  public function index()  {  /\* get product items \*/  $items = Product::query()->withTrashed()->get();  /\* return product index view \*/  return view("product.index", compact(["items"]));  }   /\*\*  \* Show the form for creating a new resource.  \*/  public function create()  {  /\* get companies \*/  $companies = Company::withTrashed()->get();  /\* return product create view \*/  return view("product.create", compact(["companies"]));  }   /\*\*  \* Store a newly created resource in storage.  \*/  public function store(Request $request)  {  /\* validate request value \*/  $request->validate([  "gtin" => ["required", "unique:products,GTIN", "regex:/^(\d{13}|\d{14})$/"],  "image" => ["nullable", "file", "image"],  ]);  /\* insert row \*/  $brand = Brand::where("name", $request->brand)->first();  if(!$brand) {  $brand = Brand::create([  "name"=> $request->brand,  ]);  }  $country = Country::where("name", $request->country)->first();  if(!$country) {  $country = Country::create([  "name"=> $request->country,  ]);  }  $image = null;  if($request->has("image")) {  $image = ProductImage::create([  "name"=> $request->file("image")->store(),  ]);  }  $product = Product::create([  "GTIN" => $request->gtin,  "brand\_id" => $brand->id,  "country\_id" => $country->id,  "product\_image\_id" => $image?->id,  "company\_id" => $request->company\_id,  "gross\_weight"=> $request->gross\_weight,  "net\_weight"=> $request->net\_weight,  "weight\_unit"=> $request->weight\_unit,  ]);  $en = Language::where("code", "en")->first();  $fr = Language::where("code", "fr")->first();  /\* name \*/  ProductName::create([  "product\_id" => $product->id,  "language\_id" => $en->id,  "contents" => $request->name  ]);  ProductName::create([  "product\_id" => $product->id,  "language\_id" => $fr->id,  "contents" => $request->name\_fr  ]);  /\* description \*/  ProductDescription::create([  "product\_id" => $product->id,  "language\_id" => $en->id,  "contents" => $request->description  ]);  ProductDescription::create([  "product\_id" => $product->id,  "language\_id" => $fr->id,  "contents" => $request->description\_fr  ]);  /\* redirect product page \*/  return redirect()->route("product.index")->with("alert", "Success create product");  }   /\*\*  \* Display the specified resource.  \*/  public function show(Product $product)  {  /\* return product detail view \*/  return view("product.show", compact(["product"]));  }   /\*\*  \* Show the form for editing the specified resource.  \*/  public function edit(Product $product)  {  //  }   /\*\*  \* Update the specified resource in storage.  \*/  public function update(Request $request, Product $product)  {  /\* Toggle product visible status \*/  if($product->trashed()) {  $product->restore();  } else {  $product->delete();  }  /\* redirect product detail page \*/  return redirect()->route("product.show", $product)->with("alert", "Success change product visibility");  }   /\*\*  \* Remove the specified resource from storage.  \*/  public function destroy(Product $product)  {  /\* Check visibility \*/  if(!$product->is\_hidden) {  return redirect()->back()->with("alert", "This product is visible");  }  /\* force delete \*/  $product->forceDelete();  /\* redirect product index page \*/  return redirect()->route("product.index")->with("alert", "Success force delete product");  } } |

1. 任务总结

* 通过本任务，我们学习了如何在 Laravel 中设计和实现 Web 路由，并结合中间件保护敏感路由。
* 我们了解了如何通过资源控制器简化 CRUD 操作，并学会了如何处理文件上传与删除。
* 本任务帮助我们理解了 RESTful 路由的设计理念，以及如何使用 Laravel 的内置功能处理常见的开发任务。

1. 任务思考

* 如何优化路由设计，减少冗余？例如，是否可以通过命名空间或路由组来进一步简化代码？
* 对于文件上传和删除操作，有没有更好的方式来管理存储空间，避免频繁操作时引发性能问题？
* 如何实现产品与公司之间的关联管理，确保数据一致性？

**任务三 用户端API接口开发**

1. 学习目标

* 理解如何在 Laravel 中设计和开发用户端 API 接口。
* 掌握分页查询、多语言支持和数据格式化输出的实现方法。
* 学习如何使用路由模型绑定与中间件处理逻辑。

1. 开发任务
2. 开发一个支持分页和关键词搜索的产品列表接口 /products.json。
3. 开发一个查询单一产品详情的接口 /products/{product:GTIN}.json。
4. 确保隐藏产品（is\_hidden = true）不会被返回。
5. 格式化数据输出，包括品牌、公司信息及多语言支持。
6. 开发思路
7. 路由设计：

* 定义两个 API 路由，一个用于产品列表查询，一个用于单一产品详情查看。

1. 控制器逻辑：

* 实现搜索、分页和数据过滤逻辑，以及详细数据的格式化输出。

1. 模型关系：

* 通过模型关系访问产品的品牌、公司、联系人等信息。

1. 响应数据结构：

* 确保返回的数据结构清晰且具备扩展性，便于前端解析。

1. 隐藏产品过滤：

* 添加逻辑避免隐藏的产品被查询到。

1. 实现步骤
2. 路由设计：

在 web.php 文件中添加以下路由：

|  |
| --- |
| Route::get("/products.json",[\App\Http\Controllers\ApiController::class,"index"])->name("api.index");  Route::get("/products/{product:GTIN}.json",[\App\Http\Controllers\ApiController::class,"show"])->name("api.show"); |

1. 产品列表接口 (index 方法) ：

|  |
| --- |
| public function index(Request $request) {  $keyword = $request->get("query");  $page = +$request->get("page", 1);  $products = Product::all()  ->filter(fn($item) => !$item->is\_hidden)  ->filter(function ($item) use ($keyword) {  $keywordReg = '/' . $keyword . '/';  return !$keyword || (  preg\_match($keywordReg, $item->name) ||  preg\_match($keywordReg, $item->name\_fr) ||  preg\_match($keywordReg, $item->description) ||  preg\_match($keywordReg, $item->description\_fr)  );  })  ->values();  $totalCount = $products->count();  $perCount = 10;  $products = $products->slice(($page - 1) \* $perCount, $perCount)->values();  $total\_pages = (int)ceil($totalCount / $perCount);  if (!$total\_pages) {  $total\_pages = 1;  }   $extraQuery = [];  if ($keyword) {  $extraQuery = ["query" => $keyword];  }   /\* return api response \*/  return response()->json([  "data" => $products->map(function ($product) {  return [  "name" => [  "en" => $product->name,  "fr" => $product->name\_fr,  ],  "description" => [  "en" => $product->description,  "fr" => $product->description\_fr,  ],  "gtin" => $product->GTIN,  "brand" => $product->brand->name,  "countryOfOrigin" => $product->country->name,  "weight" => [  "gross" => +$product->gross\_weight,  "net" => +$product->net\_weight,  "unit" => $product->weight\_unit,  ],  "company" => [  "companyName" => $product->company->name,  "companyAddress" => $product->company->address,  "companyTelephone" => $product->company->telephone,  "companyEmail" => $product->company->email,  "owner" => [  "name" => $product->company->owner->name,  "mobileNumber" => $product->company->owner->mobile,  "email" => $product->company->owner->email,  ],  "contact" => [  "name" => $product->company->contact->name,  "mobileNumber" => $product->company->contact->mobile,  "email" => $product->company->contact->email,  ],  ]  ];  }),  "pagination" => [  "current\_page" => $page,  "total\_pages" => $total\_pages,  "per\_page" => $perCount,  "next\_page\_url" => $page > $total\_pages || $page === $total\_pages ? null : route("api.index", [...$extraQuery, "page" => $page + 1]),  "prev\_page\_url" => $page > $total\_pages || $page === 1 ? null : route("api.index", [...$extraQuery, "page" => $page - 1])  ]  ]); } |

1. 单一产品详情接口 (show 方法) ：

|  |
| --- |
| public function show(Product $product) {  /\* if hidden item, throw 404 \*/  if($product->is\_hidden) {  abort(404);  }  /\* return response \*/  return response()->json([  "name" => [  "en" => $product->name,  "fr" => $product->name\_fr,  ],  "description" => [  "en" => $product->description,  "fr" => $product->description\_fr,  ],  "gtin" => $product->GTIN,  "brand" => $product->brand->name,  "countryOfOrigin" => $product->country->name,  "weight" => [  "gross" => +$product->gross\_weight,  "net" => +$product->net\_weight,  "unit" => $product->weight\_unit,  ],  "company" => [  "companyName" => $product->company->name,  "companyAddress" => $product->company->address,  "companyTelephone" => $product->company->telephone,  "companyEmail" => $product->company->email,  "owner" => [  "name" => $product->company->owner->name,  "mobileNumber" => $product->company->owner->mobile,  "email" => $product->company->owner->email,  ],  "contact" => [  "name" => $product->company->contact->name,  "mobileNumber" => $product->company->contact->mobile,  "email" => $product->company->contact->email,  ],  ]  ]); } |

1. 任务总结

* 通过本次开发，完成了用户端 API 的核心功能。
* 实现了支持关键词搜索与分页的产品列表接口，优化用户体验。
* 提供了详细的产品信息查询接口，增强了数据可用性。
* 确保隐藏产品不会暴露，提升了数据安全性。

1. 任务思考

* 当产品数据量大时，如何通过缓存或数据库索引提升查询效率？

**项目六**

**世界技能大赛典型网站设计与开发**

**前端应用程序开发**

**项目需求**

一家新成立的公司正在寻找全堆栈开发者来创建一个在线浏览器游戏平台。游戏开发者可以将游戏上传到该平台上，用户也可以在浏览器中在线玩游戏。

该平台有三个部分：

* 游戏开发者门户网站：一个供游戏开发者将游戏上传到该平台上的网络应用程序。
* 管理员门户：一个供管理员使用用户和游戏来管理平台的web应用程序。
* 游戏门户网站：一个供玩家在浏览器中在线玩游戏的网络应用程序。

该公司希望为该平台创建一个最低可行的产品（MVP）。MVP应该已经包含了上述的部分，但是游戏开发者门户和管理员门户还没有充实起来是可以接受的。游戏门户应该功能齐全，这样用户就可以在浏览器中在线玩游戏。

**任务一** **搭建前端工程**

1. 学习目标
2. 开发任务
3. 开发思路
4. 实现步骤
5. 任务总结
6. 任务思考

**任务二 游戏首页面实现**

1. 学习目标
2. 开发任务
3. 开发思路
4. 实现步骤
5. 任务总结
6. 任务思考

**任务三 用户注册登录页面实现**

1. 学习目标
2. 开发任务
3. 开发思路
4. 实现步骤
5. 任务总结
6. 任务思考

**任务四 游戏排行榜页面实现**

1. 学习目标
2. 开发任务
3. 开发思路
4. 实现步骤
5. 任务总结
6. 任务思考

**任务五 个人中心页面实现**

1. 学习目标
2. 开发任务
3. 开发思路
4. 实现步骤
5. 任务总结
6. 任务思考

**任务六 管理端页面实现**

1. 学习目标
2. 开发任务
3. 开发思路
4. 实现步骤
5. 任务总结
6. 任务思考

**项目七**

**世界技能大赛典型网站设计与开发**

**竞速赛介绍**

**项目需求**

在速度测试项目中。你只需要在比赛中完成随机挑选的一个。请按照说明去知道要实施哪一个。

对于每一个简单的问题，它都值0.5分，⼯作时间少于 15 分钟。

对于每个介质（正常），它值1分，⼯作时间少于 30 分钟。。

对于每一个困难（困难），它需要1.5分，预计需要⼤约 30 分钟的⼯作。

请将您的文件放入/XX\_module\_a/中，并且将任务ID作为文件夹名称，例如/XX\_module\_a/A1/XX是工作站编号。

**竞速赛主要知识点介绍**

理解并应用相关领域的基本理论和知识。使用HTML，CSS，JAVASCRIPT在有效的时间内对任务优先级进行排序，快速且准确地完成模块的编写,参赛者需在比赛中熟练运用这些技能以取得好成绩

**WSC2024-竞速赛A13：滴⽔（中等）**

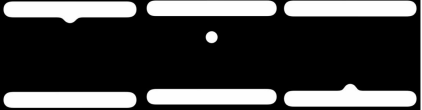
### 学习目标

➢ 理解CSS动画的基本概念及应用场景,学习如何使用伪元素 ::before 和 ::after 创建附加装饰性效果。

➢ 掌握 keyframes 的使用以创建重复性动画。

➢ 学习 SVG 过滤器（feGaussianBlur 和 feColorMatrix）的基本用法和视觉效果。

### 开发任务



1. 创建一个带有中心“水滴”动画效果的页面，背景色为黑色。
2. 使用 CSS 伪元素和定位，创建容器内的装饰性边框。
3. 实现一个模拟水滴落下的效果。
4. 添加 SVG 滤镜，使动画看起来更加柔和。

### 开发思路

在开发中，我们将使用 CSS Flexbox 设置页面居中，以确保容器和动画位于中央。

首先，创建一个 div 容器来包含“水滴”动画，并通过伪元素添加上下的白色修饰条。接着，利用 @keyframes 动画让“水滴”在容器内上下运动，模拟滴水效果。此外，通过使用 SVG 滤镜中的高斯模糊和颜色矩阵，我们可以为“水滴”效果添加柔和的滤镜，提升整体视觉美感。

### 实现步骤

1. **设置基本HTML结构**
   1. 在页面主体内，使用 <div class="container"> 标签包裹一个“水滴” <div class="ball"> 元素，模拟滴水动画的主体。

|  |
| --- |
| <!doctype html>  <html lang="en">  <head>  <meta charset="UTF-8">  <meta name="viewport" content="width=device-width, initial-scale=1.0">  <title>滴水动画</title>  <link rel="stylesheet" href="./style.css">  </head>  <body>  <div class="container">  <div class="ball"></div>  </div>  </body>  </html> |

* 1. 在 HTML 中嵌入 SVG 滤镜代码，用于稍后在 CSS 中添加视觉效果。

|  |
| --- |
| <svg xmlns="http://www.w3.org/2000/svg">  <filter id="filter">  <feGaussianBlur in="SourceGraphic" stdDeviation="10" />  <feColorMatrix values="1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 18 -7" />  </filter>  </svg> |

1. **设置CSS样式**
   1. 设置页面的整体样式，包括背景颜色、容器大小、以及水平和垂直居中。

|  |
| --- |
| body {  background: black;  height: 100vh;  display: flex;  justify-content: center;  align-items: center;  }  .container {  width: 300px;  height: 300px;  position: relative;  filter: url(#filter);  } |

* 1. 使用伪元素 ::before 和 ::after 为容器创建顶部和底部边框。

|  |
| --- |
| .container::before,  .container::after {  content: "";  position: absolute;  left: 0;  right: 0;  background: #fff;  border-radius: 50rem;  height: 30px;  }  .container::before {  top: 0;  }  .container::after {  bottom: 0;  } |

* 1. 应用 @keyframes 动画让“水滴”反复从顶部降落到底部

|  |
| --- |
| .ball {  position: absolute;  left: 50%;  transform: translateX(-50%);  width: 30px;  height: 30px;  background: #fff;  border-radius: 50%;  animation: drop 2s infinite;  }  @keyframes drop {  from {  top: 0;  }  to {  top: calc(100% - 30px);  }  }  svg {  display: none;  } |

### 任务总结

在这个项目中，我们成功开发了一个带有水滴动画效果的网页。整个开发过程从以下几个方面展开：

➢ **页面结构搭建**：使用 HTML 创建了基础的页面结构，包括一个容器和水滴元素，同时嵌入了 SVG 滤镜以提升视觉效果。

➢ **样式设计**：通过 CSS 设置了页面的黑色背景和居中布局，使用伪元素为容器添加了装饰性边框，并通过 CSS 网格布局保持页面的简洁性。

➢ **动画效果实现**：在 CSS 中实现了水滴的上下运动效果，通过关键帧动画使水滴在容器内平滑移动，增加了页面的动态感。应用 SVG 滤镜中的高斯模糊和颜色矩阵，使水滴动画看起来更加柔和，提升了整体的视觉美感。

**WSC2024-竞速赛A22：CSS 条形图（简单）**

### 学习目标

➢ 掌握使用 CSS 创建条形图的基本技能。

➢ 理解 CSS 动画和过渡效果的实现原理。

➢ 学习如何使用 Flexbox 布局提升页面结构的灵活性。

### 开发任务



构建一个简单的条形图，当用户将鼠标悬停在条形图上时，条形图会从底部向上展开，展现不同的高度和颜色。

### 开发思路

通过 HTML 和 CSS 构建一个包含多个条形图的容器。利用 CSS 的过渡和变换属性，实现动态效果，让条形图在鼠标悬停时显示更高的状态。

### 实现步骤

1. **设置基本 HTML 结构**：
   1. 创建一个 container 容器，并在其中添加三个 bar 条形图元素

|  |
| --- |
| <!doctype html>  <html lang="en">  <head>  <meta charset="UTF-8">  <meta name="viewport" content="width=device-width, initial-scale=1.0">  <meta http-equiv="X-UA-Compatible" content="ie=edge">  <title>Document</title>      <link rel="stylesheet" href="./style.css">  </head>  <body>  <div class="container">  <div class="bar bar-1"></div>  <div class="bar bar-2"></div>  <div class="bar bar-3"></div>  </div>  </body>  </html> |

1. **添加 CSS 样式**：
   1. 设置全局样式以清除默认的边距和填充，设置 box-sizing: border-box 确保元素尺寸计算一致。

|  |
| --- |
| \* {  margin: 0;  padding: 0;  box-sizing: border-box;  }  body {  height: 100vh;  display: flex;  justify-content: center;  align-items: center;  } |

* 1. 将 body 设置为 100vh 高度，使用 Flexbox 布局使容器内的条形图垂直对齐。

|  |
| --- |
| .container {  width: 400px;  height: 400px;  display: flex;  justify-content: space-around;  align-items: flex-end;  border: 1px solid #333;  padding: 2rem;  } |

* 1. 为每个条形图设置不同的背景颜色和高度,同时设置 transition-delay 控制动画的出现顺序。初始状态通过 transform: scaleY(0) 隐藏元素，并设置 transform-origin: bottom 使动画从底部开始

|  |
| --- |
| .bar {  width: 4rem;  transition: transform 1s;  transform-origin: bottom;  transform: scaleY(0);  }  .bar-1 {  background: #fe5c5d;  height: 100%;  transition-delay: 0s;  }  .bar-2 {  background: #81fd58;  height: 36%;  transition-delay: .3s;  }  .bar-3 {  background: #758ffc;  height: 72%;  transition-delay: .6s;  } |

* 1. 添加 .container:hover .bar 样式，通过取消 transform 的 scaleY 设置，使条形图在鼠标悬停时展开，实现平滑过渡动画。

|  |
| --- |
| .container:hover .bar {  transform: none;  } |

1. **效果预览**： 将以上代码放入 HTML 文件中，打开浏览器查看效果。在鼠标悬停时，条形图会展示出预设的高度。

### 任务总结

在这个项目中，我们成功实现了一个简单的条形图，通过 CSS 动画和 Flexbox 布局提升了用户体验。

创建了一个包含多个条形图的容器，为每个条形图元素定义了唯一的类名，以便于后续样式的应用。

使用全局样式清除了默认边距和填充，并利用 Flexbox 布局使条形图垂直对齐。为每个条形图设置了不同的背景颜色和初始高度，同时应用了过渡效果，以便在用户鼠标悬停时能够平滑展开。通过设置 transform 属性和 transform-origin，条形图在鼠标悬停时会从底部向上展开，展示出预设的高度和颜色，增强了视觉效果。

**WSC2024-竞速赛B35：分形三角形（简单）**

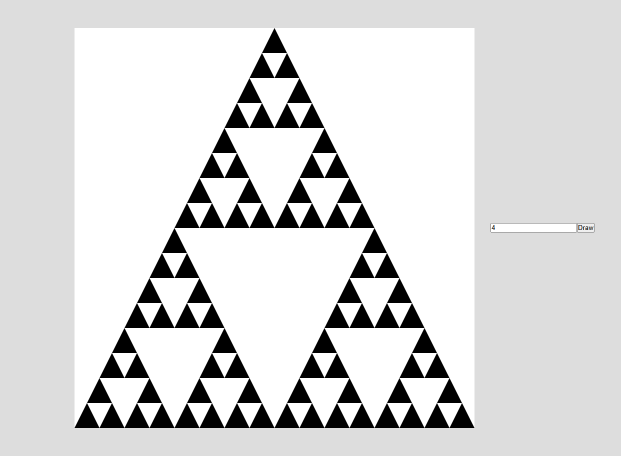
### 学习目标

➢ 理解分形图形的概念及其特点。

➢ 掌握使用 HTML5 Canvas 绘制图形的方法。

➢ 学会利用递归算法生成分形图形。

### 开发任务



1. 本项目的任务是使用 HTML5 Canvas 绘制分形三角形，并实现用户输入最大递归深度的功能，以控制分形的复杂程度。
2. 设计简单的用户界面，包括输入框和按钮，使用户能够交互式地生成不同复杂度的分形图形。

### 开发思路

为了实现功能，我们将开发过程分为几个关键部分。首先，设计 HTML 结构，包括一个 600x600 像素的 Canvas 元素和一个表单用于获取用户输入，并使用简单的 CSS 样式使其居中显示。接着，利用 JavaScript 处理表单提交事件，读取并验证用户输入的深度值，确保其在 1 到 10 之间，不合法时通过弹窗提示用户。然后，定义绘图逻辑，创建递归函数 drawTriangle，根据当前深度决定绘制完整三角形或更小的子三角形，直到达到深度为零时绘制基础三角形。

### 实现步骤

1. **基础结构：**

* 创建一个 Canvas 和一个表单用于输入递归深度，设置 Canvas 的宽高为 600 像素，确保绘图区域足够大，能够展示分形图案。

|  |
| --- |
| <!DOCTYPE html>  <html lang="en">  <head>      <meta charset="UTF-8">      <meta name="viewport" content="width=device-width, initial-scale=1.0">      <title>B35 Title</title>      <link rel="stylesheet" href="./style.css">  </head>  <body>      <canvas width="600" height="600" id="canvas"></canvas>      <form id="form">          <input type="number" id="depth" />          <button>Draw</button>      </form>      <script src="./script.js"></script>  </body>  </html> |

1. **样式设计：**

* 使用 CSS 设置页面布局，使 Canvas 居中并添加背景色，增强用户体验。#canvas 设置白色背景，使分形图案清晰呈现。

|  |
| --- |
| \*,        \*::before,        \*::after {          box-sizing: border-box;          margin: 0;          padding: 0;        }        body {          display: flex;          align-items: center;          justify-content: center;          min-height: 100vh;          background-color: #ddd;          gap: 1rem;        }        #canvas {          background-color: #fff;        } |

1. **用户输入处理：**

* 使用 JavaScript 获取 Canvas 元素和表单，设置提交事件处理，校验递归深度输入值，确保用户输入的深度在 1 到 10 之间，输入不合法时给予提示。

|  |
| --- |
| const canvas = document.getElementById("canvas");  const ctx = canvas.getContext("2d");  document.getElementById("form").onsubmit = (e) => {  e.preventDefault();  const depth = Number(document.getElementById("depth").value);  if (depth <= 0 || depth > 10) {  alert("Depth must be between 1 and 10");  return;  }  ctx.clearRect(0, 0, canvas.width, canvas.height);  const triangle = [  { x: canvas.width / 2, y: 0 },  { x: 0, y: canvas.height },  { x: canvas.width, y: canvas.height },  ];  drawTriangle(triangle, depth);  }; |

1. **递归绘图逻辑：**

* 定义 drawTriangle 函数，绘制三角形并当 depth 为 0 时，绘制当前三角形并返回。根据当前深度递归绘制子三角形，达到分形效果。

|  |
| --- |
| function drawTriangle(triangle, depth) {  if (depth === 0) {  ctx.beginPath();  ctx.moveTo(triangle[0].x, triangle[0].y);  ctx.lineTo(triangle[1].x, triangle[1].y);  ctx.lineTo(triangle[2].x, triangle[2].y);  ctx.closePath();  ctx.fill();  return;  }  const a = { x: (triangle[0].x + triangle[1].x) / 2, y: (triangle[0].y + triangle[1].y) / 2 };  const b = { x: (triangle[1].x + triangle[2].x) / 2, y: (triangle[1].y + triangle[2].y) / 2 };  const c = { x: (triangle[0].x + triangle[2].x) / 2, y: (triangle[0].y + triangle[2].y) / 2 };  drawTriangle([triangle[0], a, c], depth - 1);  drawTriangle([a, triangle[1], b], depth - 1);  drawTriangle([b, c, triangle[2]], depth - 1);  } |

### 任务总结

➢ 输入验证：在表单提交时，添加了对用户输入的有效性检查，确保输入值在 1 到 10 之间，提升了应用的健壮性。

➢ 图形绘制：利用 HTML5 Canvas API 实现了分形三角形的绘制，展示了递归算法的效果通过创建递归函数 drawTriangle，实现了分形三角形的动态绘制逻辑，使用户可以直观观察不同深度的分形效果。

**WSC2024-竞速赛C11：图像拼接（简单）**

### 学习目标

➢ 学习如何处理图像上传并生成马赛克效果。

➢ 理解图像处理的基本概念，例如像素、颜色分配和色彩平均计算。

➢ 掌握使用 PHP 的 GD 库进行图像处理的基本操作。

### 开发任务



1 ) 开发一个图像处理工具，能够接受用户上传的 PNG 图像，计算其分块区域的平均颜色，并生成马赛克效果的图像，供用户下载。

### 开发思路

本项目的目标是通过 PHP 的 GD 库处理用户上传的图像并生成马赛克效果。首先，创建一个 HTML 表单，允许用户上传 PNG 格式的图像，确保表单设置了 enctype="multipart/form-data"。上传后，使用 $\_FILES 检查图像是否成功上传，并处理可能的错误。

接着，利用 imagecreatefrompng() 加载图像，并获取其宽度和高度。然后，创建一个与原图相同大小的新画布。定义一个函数计算每个马赛克单元的平均颜色，通过遍历像素并累加 RGB 值。通过双重循环，按照指定的单元格大小遍历图像，利用计算出的平均颜色在新画布上绘制相应的矩形。最后，输出生成的马赛克图像，完成整个处理过程。

### 实现步骤

1. 首先，创建一个名为 index.php 的文件，并在其中嵌入 PHP 代码和 HTML 表单。HTML 表单允许用户上传图像，使用 POST 方法，并设置 enctype 为 multipart/form-data 以支持文件上传。

|  |
| --- |
| <?php  // 图片处理逻辑  ?>  <!DOCTYPE html>  <html lang="en">  <head>  <meta charset="UTF-8">  <meta name="viewport" content="width=device-width, initial-scale=1.0">  <title>C11</title>  </head>  <body>  <form action="#" method="POST" enctype="multipart/form-data">  <input type="file" accept="image/\*" name="image" id="image" required>  <br><br>  <button>Generate Mosaic</button>  </form>  </body>  </html> |

1. 在 PHP 代码块中，添加处理上传图像的逻辑。首先检查请求方式是否为 POST，并验证上传的文件是否存在和没有错误。可以通过检查 $\_FILES['image']['error'] 的值来确认上传是否成功。

|  |
| --- |
| if ($\_SERVER['REQUEST\_METHOD'] == 'POST' && isset($\_FILES['image'])) {  // 检查上传错误  if ($\_FILES['image']['error'] == 1) {  echo 'Error. Try a different image.';  return;  }  // 获取临时文件路径  $file = $\_FILES['image']['tmp\_name'];  $cellSize = isset($\_GET['cell\_size']) ? intval($\_GET['cell\_size']) : 50;  // 创建图像资源  $image = imagecreatefrompng($file);  $width = imagesx($image);  $height = imagesy($image);  } |

1. 定义一个函数来计算每个单元格的平均颜色。以下是 getAverageColor 函数的实现，接收单元格位置、大小、原图及其尺寸。使用双重循环遍历单元格内的像素，累加 RGB 值。计算平均值并返回 imagecolorallocate 创建的颜色。

|  |
| --- |
| function getAverageColor($x, $y, $cellSize, $image, $width, $height, $result) {  $r = 0;  $g = 0;  $b = 0;  $count = 0;  // 遍历单元格内的每个像素  for ($i = $x; $i < ($x + $cellSize) && $i < $width; $i++) {  for ($j = $y; $j < ($y + $cellSize) && $j < $height; $j++) {  $rgb = imagecolorat($image, $i, $j);  $r += ($rgb >> 16) & 0xFF;  $g += ($rgb >> 8) & 0xFF;  $b += $rgb & 0xFF;  $count++;  }  }  // 避免除以零  if ($count > 0) {  return imagecolorallocate($result, round($r / $count), round($g / $count), round($b / $count));  } else {  return imagecolorallocate($result, 0, 0, 0); // 默认颜色  }  } |

1. 回到之前的判断中，创建新的图像资源 $result使用双重循环遍历图像的每个单元格，调用 getAverageColor 获取颜色，使用 imagefilledrectangle 绘制填充矩形。如果上传了图片并通过验证则生成马赛克图像，输出结果图像

|  |
| --- |
| $result = imagecreatetruecolor($width, $height);  for ($i = 0; $i < $width; $i += $cellSize) {  for ($j = 0; $j < $height; $j += $cellSize) {  $averageColor = getAverageColor($i, $j, $cellSize, $image, $width, $height, $result);  imagefilledrectangle($result, $i, $j, $i + $cellSize - 1, $j + $cellSize - 1, $averageColor);  }  }  header('Content-Type: image/png');  imagepng($result);  exit; |

### 任务总结

在这个项目中，我们成功开发了一个图像处理工具，能够生成马赛克效果的图像。整个开发过程可以总结如下：

➢ 后端服务搭建：使用 PHP 的 GD 库处理用户上传的 PNG 图像。通过创建一个 HTML 表单，用户可以方便地上传图像文件，并确保文件上传过程的可靠性和有效性。

➢ 图像处理实现：首先，加载用户上传的图像并获取其宽度和高度。随后，定义了一个函数用于计算每个马赛克单元的平均颜色。通过遍历每个像素并累加 RGB 值，成功实现了颜色的平均计算。

➢ 马赛克效果生成：通过双重循环遍历图像，根据指定的单元格大小生成新的马赛克图像。在新的画布上绘制矩形，填充计算出的平均颜色，从而创建出马赛克效果。最终，通过适当的 HTTP 头部输出生成的马赛克图像，供用户下载，完成了整个处理过程。

**WSC2022SE-竞速赛D3 井字游戏（困难）**

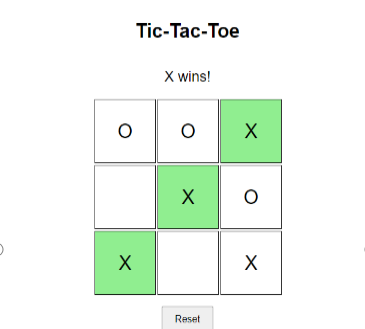
### 学习目标

➢ 理解如何使用 Node.js 和 Express 构建基本的后端服务器，处理 HTTP 请求和响应。

➢ 掌握前后端如何通过 HTTP 请求进行通信，特别是使用 Fetch API 来发送和接收数据。

➢ 学习基本的 JavaScript 游戏开发逻辑，包括用户交互处理和游戏状态更新。

### 开发任务



1）使用 Node.js 和 Express 创建一个后端服务器，处理井字棋游戏的状态。

2）实现 API 端点，以获取、保存和重置游戏状态。

3）开发前端页面，允许用户通过点击操作来玩井字棋。

4）使用 JavaScript 控制游戏逻辑，如检测游戏结束状态（胜利或平局）和切换玩家回合。实现一个简单的 AI 对手来与玩家竞争。

### 开发思路

1）后端服务搭建

后端主要负责管理和维护游戏的状态。使用 Node.js 和 Express 框架来搭建后端服务。

* 初始化 Node.js 项目并安装 Express 和 CORS，用于处理服务器和跨域请求。
* 定义一个全局对象来保存游戏状态，包括棋盘布局、游戏是否结束、当前玩家等信息。
* 使用 Express 启动服务器，监听特定端口以接收来自前端的请求。

2）前端界面设计

前端使用 HTML 和 CSS 创建界面，通过 JavaScript 实现交互。

* 创建页面元素：包括标题、消息显示区域、3x3 的棋盘网格、重置按钮。
* 使用 CSS 定义棋盘和单元格的样式，包括网格布局和单元格的大小、边框等。

3）实现游戏逻辑

JavaScript 用于处理用户交互、更新游戏状态、以及与后端的通信。

定义变量保存当前的棋盘状态。

* 使用 JavaScript 从后端获取游戏状态并渲染到页面上
* 监听棋盘单元格的点击事件，根据当前玩家在单元格中放置标记。
* 每次移动后检查游戏结束条件（胜利或平局），更新状态并显示相应的消息。
* 监听重置按钮的点击事件，调用后端 API 重置游戏状态，刷新前端显示。

1. 前后端通信

使用 Fetch API 与后端通信，获取和更新游戏状态。确保前端和后端数据同步。

### 实现步骤

1. 后端服务搭建
2. 打开CMD视窗，初始化项目和安装依赖,express 用于创建服务器，cors 用于处理跨域请求：
   1. npm init -y : 快速创建一个 package.json 文件，使用默认配置。
   2. npm install express cors
3. 创建app.js文件，设置 Express 服务器并应用CORS,app.use(express.json()) 解析 JSON 请求体

const express = require("express");

const cors = require("cors");

const app = express();

const port = 3000;

app.use(express.json());

app.use(cors()); // Enable CORS for all routes

1. 在 app.js 中，首先定义游戏的初始状态，随后设置相关的 API 路由，以便前端能够获取和更新游戏状态。

|  |
| --- |
| let gameState = {      board: Array(9).fill(""),      isGameOver: false,      isPlayerTurn: true,      message: "",  }; |

* 获取游戏状态的路由

|  |
| --- |
| app.get("/api/getGameState", (req, res) => {      res.json(gameState);  }); |

* 提交游戏状态的路由

|  |
| --- |
| app.post("/api/saveGameState", (req, res) => {      gameState = req.body;      res.sendStatus(200);  }); |

* 重置游戏状态的路由

|  |
| --- |
| app.post("/api/resetGameState", (req, res) => {      gameState = {          board: Array(9).fill(""),          isGameOver: false,          isPlayerTurn: true,          message: "",      };      res.sendStatus(200);  }); |

1. 最后，设置服务器在指定端口运行,使用 app.listen 启动服务器，控制台将输出运行状态，确认服务器已成功启动。

app.listen(port, () => {

console.log(`Server is running on http://127.0.0.1:${port}`);

});

1. 打开CMD视窗，运行以下命令启动服务器

node .\app.js

成功启动后，控制台将显示 Server is running on http://127.0.0.1:3000，表明服务器已在指定端口监听请求。

1. 前端用户界面设计
2. 页面结构创建 使用基本的 HTML 标签创建页面结构，包括标题、消息显示区域、棋盘容器和重置按钮

|  |
| --- |
| <!DOCTYPE html>  <html lang="en">  <head>  <meta charset="UTF-8" />  <meta name="viewport" content="width=device-width, initial-scale=1.0" />  <title>Tic-Tac-Toe</title>  </head>  <body>  <h1>Tic-Tac-Toe</h1>  <div id="message"></div>  <div id="board"></div>  <button id="reset">Reset</button>  </body>  </html> |

1. 创建棋盘 布局使用 CSS 的 display: grid 属性，定义一个 3x3 的棋盘布局。使用 grid-template-columns 和 grid-template-rows 设置每个格子的大小和间隔，确保棋盘在页面上有合适的视觉效果。

|  |
| --- |
| #board {  display: grid;  grid-template-columns: repeat(3, 100px);  grid-template-rows: repeat(3, 100px);  gap: 5px;  margin-top: 20px;  } |

1. 棋盘单元格 .cell 设置大小、边框、字体和居中布局。使用 flex 布局确保单元格内容居中对齐，并为单元格添加鼠标悬停效果，以便用户能够直观地看到可点击状态

|  |
| --- |
| .cell {  width: 100px;  height: 100px;  display: flex;  align-items: center;  justify-content: center;  font-size: 2em;  border: 1px solid #000;  cursor: pointer;  }  .cell.disabled {  cursor: not-allowed;  } |

1. 页面提示与重置按钮样式 设置消息区域和重置按钮的样式，使其在用户操作时具有良好的可读性和可用性。通过 margin 和 padding 控制元素之间的间隔。确保整体界面布局清晰

|  |
| --- |
| #message {  margin-top: 20px;  font-size: 1.5em;  height: 30px;  }  #reset {  margin-top: 20px;  padding: 10px 20px;  font-size: 1em;  cursor: pointer;  }  body {  display: flex;  flex-direction: column;  align-items: center;  margin-top: 50px;  font-family: Arial, sans-serif;  }  h1 {  margin-bottom: 20px;  } |

1. 最后，引入同级目录下的index.js文件

|  |
| --- |
| <script src="./script.js"></script> |

1. 实现游戏逻辑
2. 游戏逻辑通过 index.js 实现，负责处理用户交互、游戏状态更新以及与后端的通信
3. 游戏变量定义 获取棋盘、消息区域和重置按钮的引用，并定义游戏变量：棋盘数组board 和胜利概率winningCombinations

|  |
| --- |
| const boardElement = document.getElementById('board');  const messageElement = document.getElementById('message')  const resetButton = document.getElementById('reset')  const api = 'http://127.0.0.1:3000'  let board = Array(9).fill('');  let isGameOver = false,      isPlayerTurn = true;  const winningCombinations = [      [0, 1, 2],      [3, 4, 5],      [6, 7, 8],      [0, 3, 6],      [1, 4, 7],      [2, 5, 8],      [0, 4, 8],      [2, 4, 6]  ]; |

1. 初始化游戏状态 fetchGameState 函数用于通过异步请求从后端获取当前游戏状态。函数内部使用 fetch 方法请求 API，并在成功响应后解析 JSON 数据。如果返回的 data.board 存在，解构赋值将当前棋盘状态、游戏结束标志和玩家回合状态赋值给相应的变量。接着调用 renderBoard 函数渲染棋盘，并更新提示信息。若游戏未结束且是 AI 回合，调用 getBestMove 函数让 AI 执行一步，并检查其状态。

|  |
| --- |
| // 获取初始状态并开始游戏  const fetchGameState = async () => {      try {          const response = await fetch(`${api}/api/getGameState`);          const data = await response.json();          if (data.board) {              ({                  board,                  isGameOver,                  isPlayerTurn              } = data);              renderBoard();              updateMessage(data.message);            data.message && highlightWinningCells(data.message.split(' ')[0])              //刷新过快会导致机器人没有思考时间              const isBoardNotEmpty = board.some(cell => cell !== '');              if (!isGameOver && isBoardNotEmpty && !isPlayerTurn) {                  // AI (O) 执行一步                  const bestMove = getBestMove();                  if (bestMove !== -1) {                      makeMove(bestMove, 'O');                      checkGameStatus('O');                      isPlayerTurn = true; // 切换到用户                  }              }          }      } catch (error) {          console.error('Failed to fetch game state:', error);      }  };  fetchGameState(); |

1. **渲染棋盘** renderBoard 函数负责动态生成棋盘的 HTML 结构，并将其插入到页面中。使用 map 方法遍历 board 数组，为每个格子创建对应的 <div class="cell"> 元素。根据 isGameOver 和单元格值的状态动态添加 disabled 类，以控制是否可以点击。最后，为每个格子添加点击事件监听器，响应用户的点击操作，确保用户可以与游戏进行交互。

|  |
| --- |
| const renderBoard = () => {      boardElement.innerHTML = board.map((value, index) =>          `<div class="cell ${isGameOver == true ?'disabled':(value ? ' disabled' : '')}" data-index="${index}">${value}</div>`      ).join('');      boardElement.querySelectorAll('.cell').forEach(cell =>          cell.addEventListener('click', handlePlayerMove)      );  }; |

1. **状态更新函数** 编写 updateGameState 和 updateMessage 函数，确保游戏状态和提示信息能够及时更新和同步到服务器。updateGameState 函数在棋盘状态发生变化时向服务器发送更新请求。

|  |
| --- |
| const updateMessage = (message) => {      if (messageElement) messageElement.textContent = message;  };  const updateGameState = async () => {      try {          await fetch(`${api}/api/saveGameState`, {              method: 'POST',              headers: {                  'Content-Type': 'application/json'              },              body: JSON.stringify({                  board,                  isGameOver,                  isPlayerTurn,                  message: messageElement.textContent              })          });      } catch (error) {          console.error('Failed to update game state:', error);      }  }; |

1. **页面状态保存** 在页面刷新或关闭时，使用 beforeunload 事件提交当前游戏状态，以便在下次打开页面时能够恢复上次的棋盘状态。

|  |
| --- |
| window.addEventListener('beforeunload', function (event) {      updateGameState();  }); |

1. **胜利状态检查** 实现 checkGameStatus 函数来判断当前棋盘状态是否有玩家胜利。通过遍历胜利组合数组，检查是否有任意一组组合的格子被同一玩家填满。如果检测到胜利，调用 endGame 函数显示获胜信息并高亮显示获胜的格子。若棋盘已满且无人获胜，显示平局信息。

|  |
| --- |
| const checkGameStatus = (player) => {      if (winningCombinations.some(combination => combination.every(index => board[index] === player))) {          endGame(`${player} wins!`, player);          return true;      }      if (board.every(cell => cell)) {          endGame("It's a draw!");          return true;      }      updateMessage(player === 'X' ? "AI's turn (O)" : "Player's turn (X)");      return false;  };  const endGame = (message, player) => {      isGameOver = true;      updateMessage(message);      if (player) highlightWinningCells(player);  };  const highlightWinningCells = (player) => {      winningCombinations.forEach(combination => {          if (combination.every(index => board[index] === player)) {              combination.forEach(index => {                  const cell = document.querySelector(`[data-index='${index}']`);                  if (cell) cell.style.backgroundColor = '#90EE90';              });          }      });  }; |

1. **玩家操作处理** handlePlayerMove 函数处理用户点击棋盘的事件。在用户点击有效格子时，更新棋盘并调用 checkGameStatus 检查是否获胜。若用户操作后未结束游戏，AI 通过调用 getBestMove 函数进行下一步，并检查 AI 是否获胜。

|  |
| --- |
| const handlePlayerMove = (event) => {      const index = event.target.dataset.index;      if (isPlayerTurn == true && !isGameOver && !board[index]) {          makeMove(index, 'X');          if (!checkGameStatus('X')) {              isPlayerTurn = false;              setTimeout(() => {                  const bestMove = getBestMove();                  if (bestMove !== -1) {                      isPlayerTurn = true;                      makeMove(bestMove, 'O');                      checkGameStatus('O')                  }              }, 500);          }      }  };  const getBestMove = () => board.findIndex(cell => !cell);  const makeMove = (index, player) => {      board[index] = player;      const cell = document.querySelector(`[data-index='${index}']`);      if (cell) {          cell.textContent = player;          cell.classList.add('disabled');          cell.removeEventListener('click', handlePlayerMove);      }  }; |

1. 最后为重置按钮添加事件监听器，用户可以重置游戏状态。点击重置按钮时，将棋盘重置为初始状态，设置消息为空，并调用后端 API 清除之前的游戏记录，确保新游戏从干净的状态开始。

|  |
| --- |
| resetButton.addEventListener('click', async () => {      try {          await fetch(`${api}/api/resetGameState`, {              method: 'POST'          });          messageElement.textContent = ""          board.fill('');          isGameOver = false;          isPlayerTurn = true;          renderBoard();      } catch (error) {          console.error('Failed to reset game state:', error);      }  }); |

### 任务总结

在这个项目中，我们成功开发了一个简单但功能齐全的井字棋游戏。整个开发过程从以下几个方面展开：

➢ 后端服务搭建：使用 Node.js 和 Express 搭建了一个服务器，用于管理游戏状态。通过 RESTful API，我们可以获取、保存和重置游戏状态，这确保了游戏的持久性和前后端的同步。

➢ 前端界面设计：通过 HTML 和 CSS 创建了一个用户友好的界面，包含一个 3x3 的棋盘、消息显示区域和重置按钮。使用 CSS 网格布局使得棋盘的布局简洁而直观。

➢ 游戏逻辑实现：在 JavaScript 中实现了核心的游戏逻辑，包括用户交互、胜负判定和状态更新。玩家的每次移动都会实时更新游戏状态，并通过前后端的通信保持数据的一致性。

➢ 前后端通信：使用 Fetch API 实现了前后端的通信，使得前端能够获取和保存游戏状态。这确保了玩家在不同设备或页面刷新后的游戏连续性。

**竞速赛赛题和规模介绍**

在WorldSkills竞速赛中，Web Technologies是一个备受关注的领域，参赛者需要在规定的时间内完成与网页开发和设计相关的任务。以下是Web Technologies竞速赛的赛题和规模介绍

### 竞速赛赛题：Web Technologies

Web Technologies赛项的竞速赛题通常围绕前端和后端开发展开，任务设置模拟了真实的开发环境，考核参赛者在Web开发中的核心技能。常见的赛题类型包括：

1. 前端开发：

网页设计与实现：要求参赛者在限定时间内设计并开发一个响应式网页，需使用HTML、CSS、JavaScript等技术，确保页面在不同设备上的兼容性和视觉效果。

用户界面优化：参赛者需要优化给定的网页，使其加载速度更快、交互更加流畅，同时提升用户体验。

1. 后端开发：

服务器端编程：任务可能要求参赛者开发一个简单的Web应用程序或API，需使用Node.js、PHP、Python等后端语言，确保功能实现的同时，注重代码的效率和安全性。

数据库集成：参赛者需要设计和实现数据库结构，并将其与Web应用集成，实现数据的存储、检索和展示。

1. 全栈开发：

全栈应用开发：参赛者需要在规定时间内，从前端到后端完整开发一个Web应用，涵盖用户界面设计、功能实现、数据管理等多个方面。

问题解决与调试：参赛者可能会面临故障排除任务，要求在短时间内定位并修复现有代码中的问题，确保应用程序正常运行。

### 竞速赛规模：Web Technologies

Web Technologies竞速赛的规模在WorldSkills竞速赛中具有显著影响力，以下是该赛项的规模特点：

1. **参赛人数**：Web Technologies吸引了来自全球各地的年轻开发者参加，每次竞速赛都有大量选手展示他们的Web开发技能。
2. 赛项内容多样性：竞速赛内容覆盖了Web开发的各个方面，从前端设计到后端编程，再到全栈开发，全面考核参赛者的技术水平。
3. 全球影响力：Web Technologies赛项在全球范围内备受关注，是展示和比拼Web开发技能的重要平台，比赛结果往往反映了各国在数字技术领域的发展水平。
4. 裁判与评判：裁判团队由Web开发领域的专家组成，他们将根据严格的标准对参赛作品进行评分，重点考察代码质量、功能实现、用户体验以及时间管理能力。

**附录**

**项目需求**

WorldSkills将2024年9月在法国里昂举办全球世界技能大赛。作为此次比赛的组织者，WorldSkills希望开发相关应用程序，为关注此次比赛的观众服务。观众可以坐在家里，了解 WorldSkills，感受比赛的气氛，并通过应用程序中的图片和视频了解比赛的动态。

## 安装网站设计工具

### 学习目标

➢ 掌握GIMP的安装步骤，包括下载、安装和配置选项

➢ 独立完成GIMP的安装和基本设置

### 关于设计工具

网站设计工具是帮助设计师创建、编辑和优化网页及其内容的软件或平台

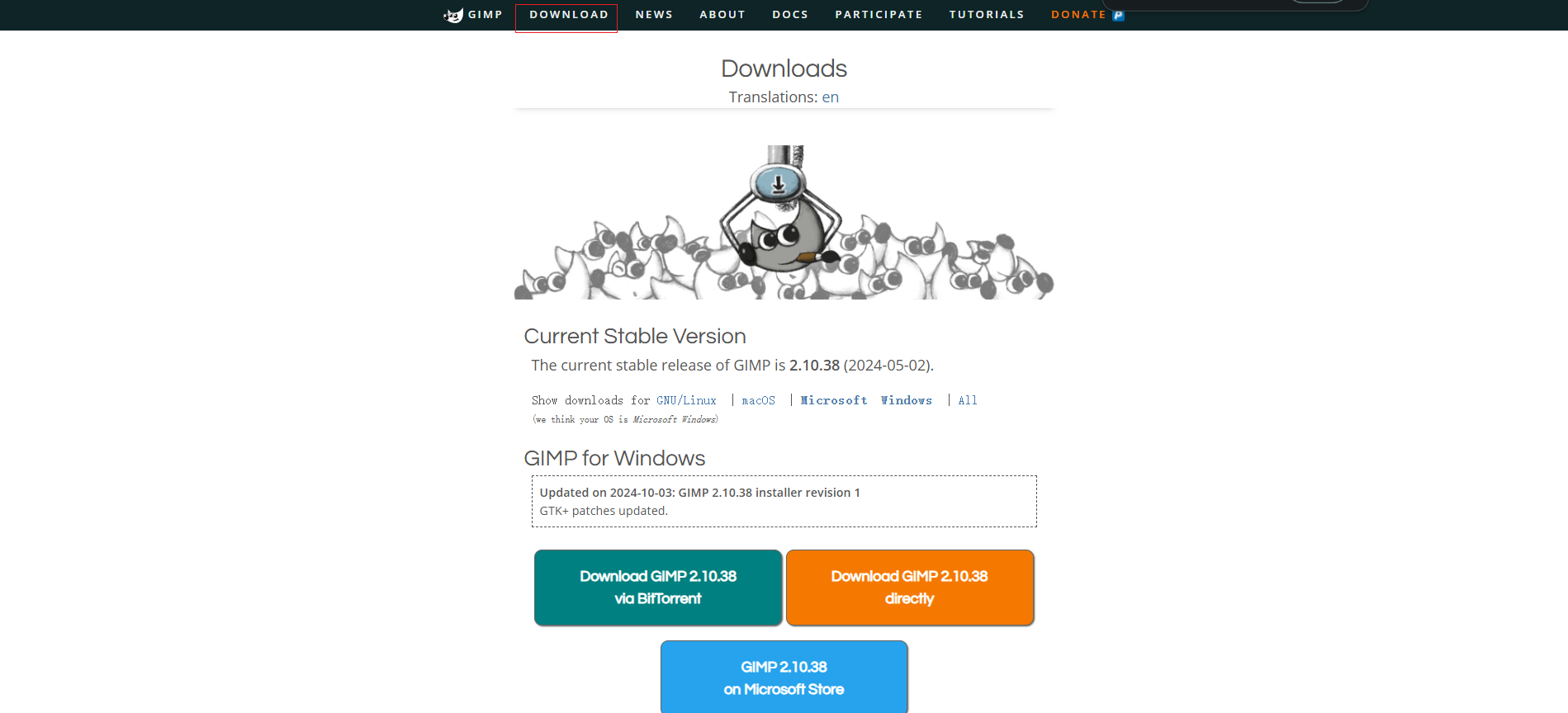
在本次比赛中使用GIMP进行图像编辑：

GIMP作为一个功能强大的图像编辑工具，尤其是在图形创作和图像处理方面。它适合设计师和开发者使用，与其他专业设计工具结合使用时，可以有效提升网站设计的质量和效率。通过学习和掌握GIMP，你可以提高自己的图像处理能力，为网站设计提供更好的视觉效果

### 实现步骤

* 1. 下载GIMP

1. 访问GIMP官方网站 <https://www.gimp.org/> ，在主页上找到“Download”按钮，点击进入下载页面。选择适合你的操作系统下载最新版本



* 1. 安装GIMP

1. 下载完成后，双击打开下载的.exe文件（Windows）或.dmg文件（macOS）。
2. 根据安装向导的提示完成以下步骤：
3. 选择语言：选择你希望使用的语言，然后点击“OK”。
4. 接受许可证协议：阅读并接受GNU通用公共许可证协议，点击“Next”。
5. 选择安装目录：你可以使用默认安装目录，或点击“Browse”选择一个自定义目录，然后点击“Next”。
6. 选择组件：可以选择安装附加组件（如插件和图标），默认选中即可，点击“Next”。
7. 开始安装：确认所有设置无误后，点击“Install”开始安装。安装过程可能需要几分钟。
8. 完成安装：安装完成后，点击“Finish”关闭安装向导。

## 安装PHP开发工具

### 学习目标

➢ 学习PHP开发环境的搭建和配置

➢ 设置PHP开发环境

➢ 使用PHP开发工具

➢ 初始化PHP框架

### 了解php DevTools

PHP是一种功能强大的开源服务器端脚本语言，专为Web开发设计，它与HTML紧密集成，允许在同一文件中混合使用PHP代码和HTML标签，并且在服务器端执行，返回已处理的结果给客户端。PHP跨平台兼容，能够在Linux、Windows、macOS等多种操作系统上运行，且与主流Web服务器（如Apache、Nginx）无缝配合使用。PHP社区庞大，拥有大量文档和扩展库，使开发者能够快速构建高效、安全的Web应用。

### 实现步骤

* 1. **安装PHP运行环境**

**XAMPP**（跨平台）：访问XAMPP官网https://www.apachefriends.org/zh\_cn/index.html，下载适用于您操作系统的版本（Windows、Linux或macOS）运行安装程序并选择PHP、Apache和MySQL等组件。安装完成后，启动Apache和MySQL服务，确保本地服务器运行。

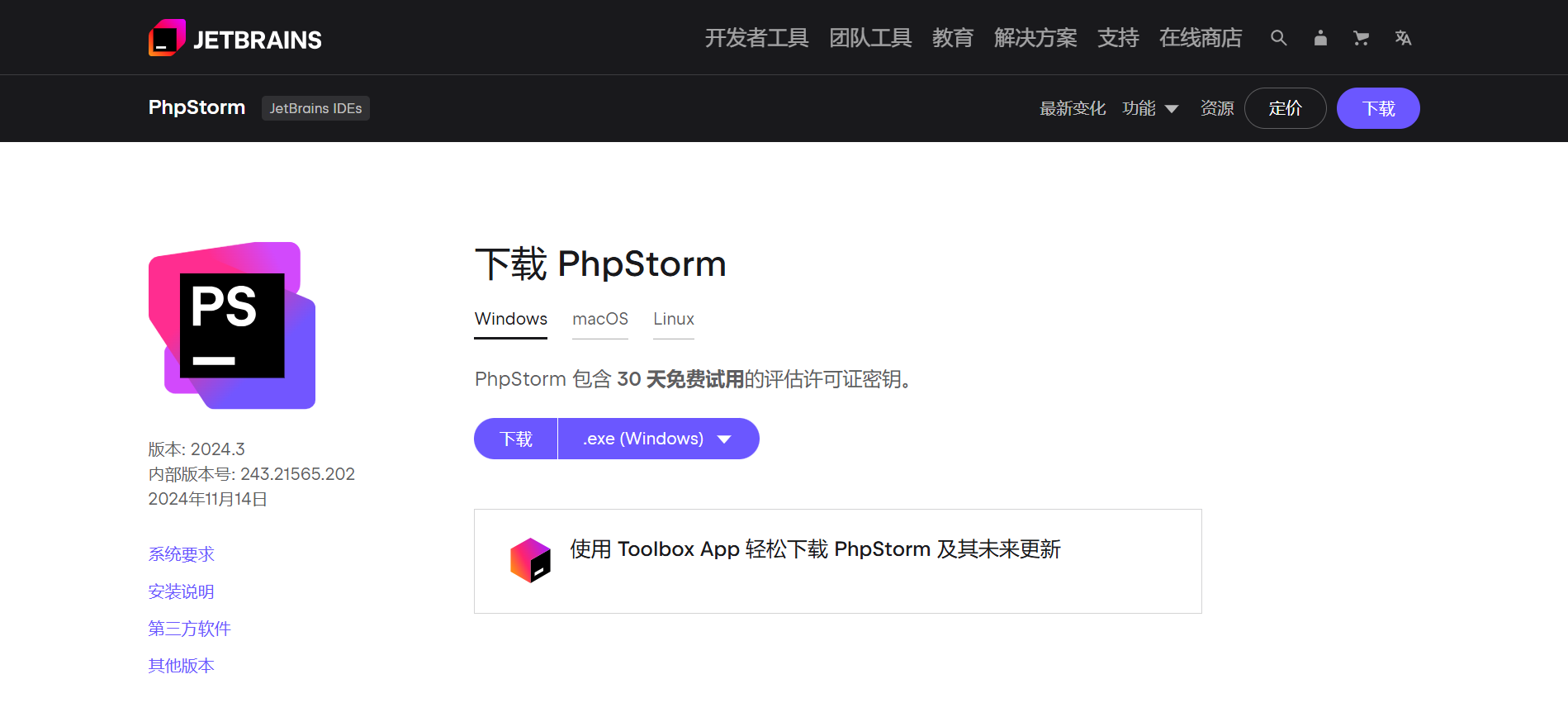


* 1. **下载并安装PHPSTORM**

PHPSTORM是由JetBrains开发的一款强大的专业PHP IDE，提供深度的代码理解、顶尖的编码辅助、和对PHP工具的全面集成。

* 1. **下载PHPStorm**
* 访问 [JetBrains官方网站](https://www.jetbrains.com/phpstorm/download/" \t "_new) 下载PHPStorm

<https://www.jetbrains.com.cn/phpstorm/download/?fromMenu#section=windows>

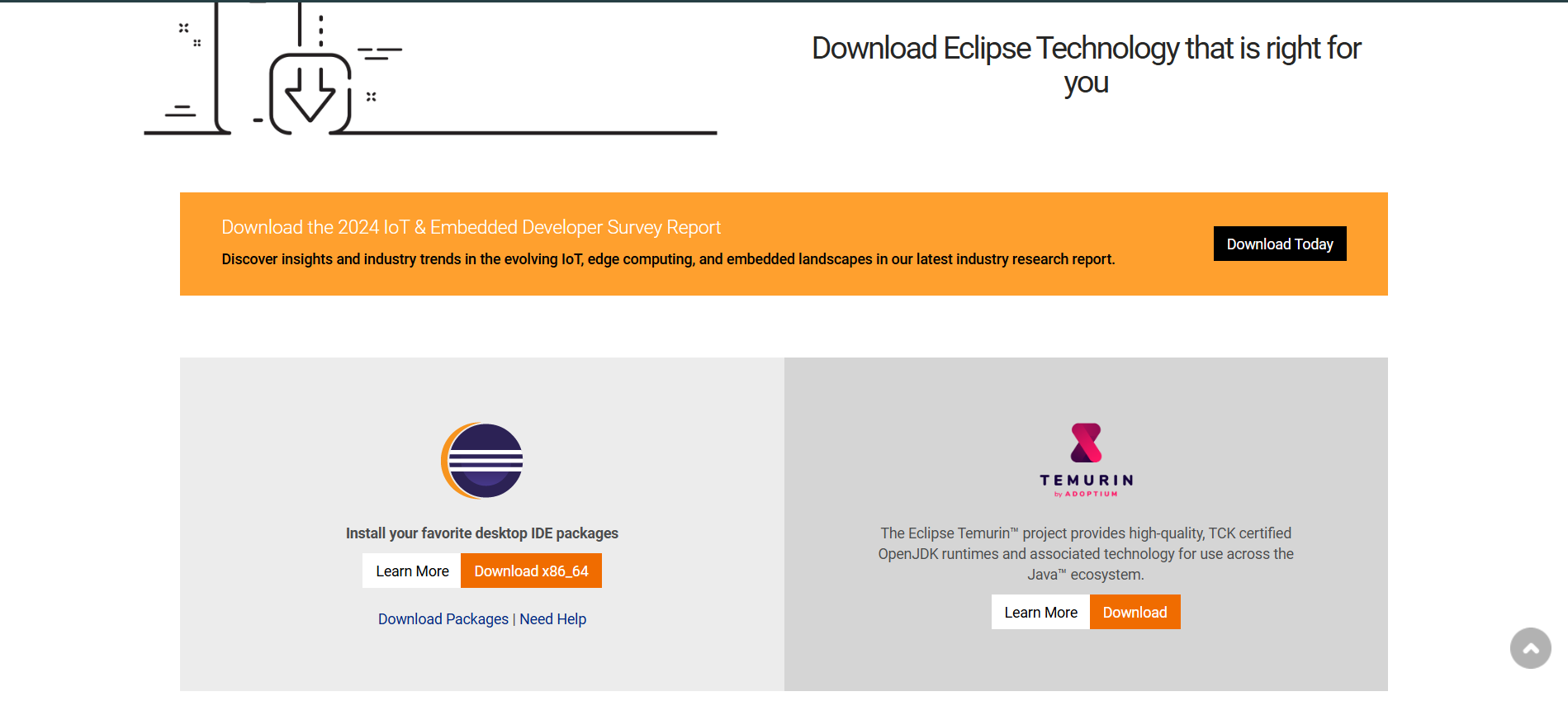


* 选择适合你操作系统的版本（Windows、Mac或Linux）安装PHPStorm
* 按照屏幕指示完成安装过程。
  1. **安装Eclipse**

Eclipse 是一个适用于多种编程语言的集成开发环境（IDE），尤其适用于Java项目。

1. **下载Eclipse**

* 访问[Eclipse官方下载页面](https://www.eclipse.org/downloads/" \t "_new) <https://www.eclipse.org/downloads/>选择适合你的操作系统



* 安装Eclipse

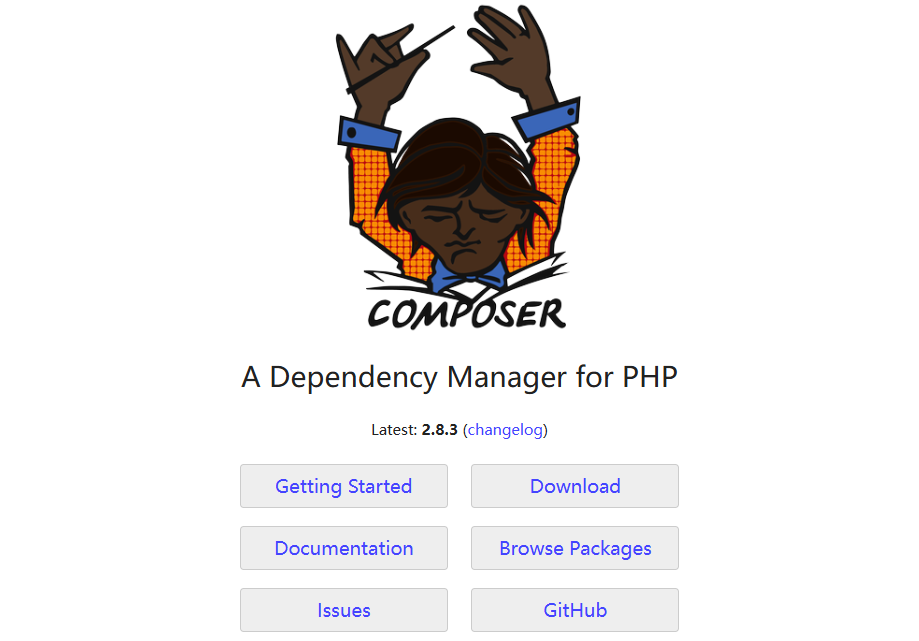
使用Eclipse安装器选择“Eclipse IDE for Java Developers”或根据你的需求选择其他适用于Web开发的包

* 1. **安装和配置Laravel**

Laravel 是一个使用PHP语言的Web应用框架，它为开发复杂的PHP应用提供了简洁、优雅的工具。

1. **安装Composer**

* Composer是PHP的依赖管理工具，Laravel的安装需要依赖Composer。
* 访问[Composer官方网站](https://getcomposer.org/" \t "_new)https://getcomposer.org/下载并安装Composer



1. **通过Composer安装Laravel**

* 打开命令行工具,使用 cd 命令切换到你希望创建新项目的目录
* 使用 Composer 创建新的 Laravel 项目
* 运行以下命令来通过 Composer 创建一个新的 Laravel 项目：

composer create-project --prefer-dist laravel/laravel myproject

这条命令会从 Laravel 的官方仓库下载最新版本的 Laravel 框架，并创建一个名为 myproject 的新目录，里面包含了所有的框架文件。记得将 myproject 替换为你的项目名。

* 1. **测试和调试环境**

确保一切设置正确，你可以在本地服务器上运行并测试你的Laravel项目。

在命令行中使用以下命令启动Laravel内置的开发服务器：

* php artisan serve

打开浏览器访问 http://localhost:8000，你应该能看到Laravel的默认首页。

## 安装nodejs开发工具

### 学习目标

➢ 了解Node.js的环境配置

➢ 熟悉常见的Node.js开发框架

➢ 学会搭建开发项目的基础架构

### 关于nodejs

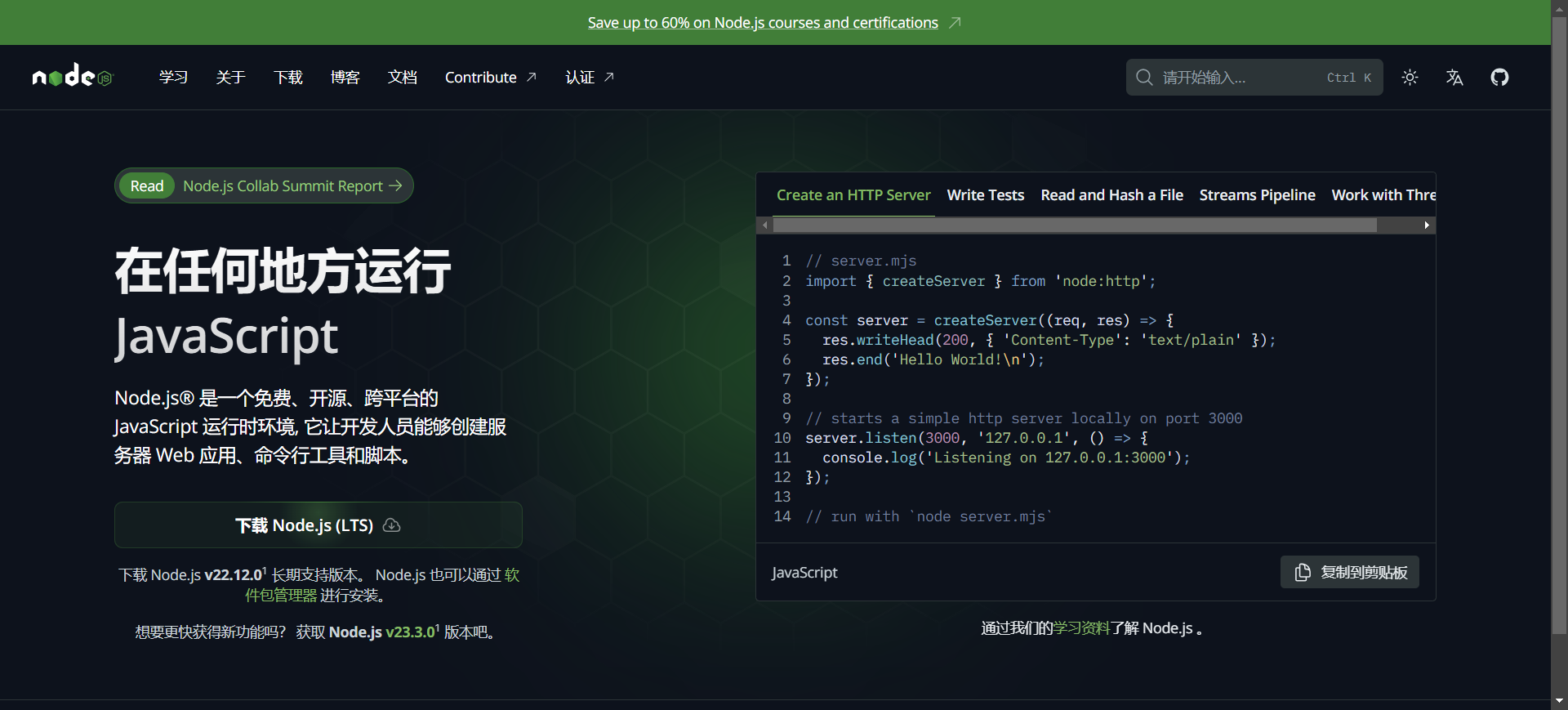
Node.js 是一个基于 Chrome V8 引擎 的开源、跨平台 JavaScript 运行时环境。它使得 JavaScript 不再局限于浏览器端，而可以在服务器端运行，支持构建高性能的网络应用。Node.js 通过事件驱动、非阻塞 I/O 模型，使得其在处理大量并发请求时具有极高的效率，非常适合用于构建实时应用，如聊天服务、在线游戏、实时数据分析等。

它是一个强大的JavaScript运行环境，而npm是世界上最大的软件注册处，用于管理JavaScript代码的依赖关系。

### 实现步骤

* 1. **下载并安装Nodejs和npm**

访问Node.js官网，选择LTS版本下载 <https://nodejs.org/zh-cn>安装Node.js和npm



* 1. 打开下载的.msi文件，跟随安装向导步骤进行安装
  2. 在安装过程中，确保勾选了包括npm的所有必要组件，并允许安装程序将Node.js添加到PATH环境变量中
  3. 检查Node.js版本 node -v,检查npm版本 npm -v
  4. **创建Node.js项目**
     1. 在命令行中进入项目目录，初始化Node.js项目

npm init -y

* + 1. 创建项目的文件结构如：

├── app.js

├── package.json

└── routes/

└── index.js

* + 1. 安装必要的依赖库，如express或其他框架进行 node.js的开发

npm install express

## 安装前端开发工具

### 学习目标

➢ 了解前端开发工具的重要性

➢ 掌握基础的前端开发工具安装与配置

### 关于前端DevTools

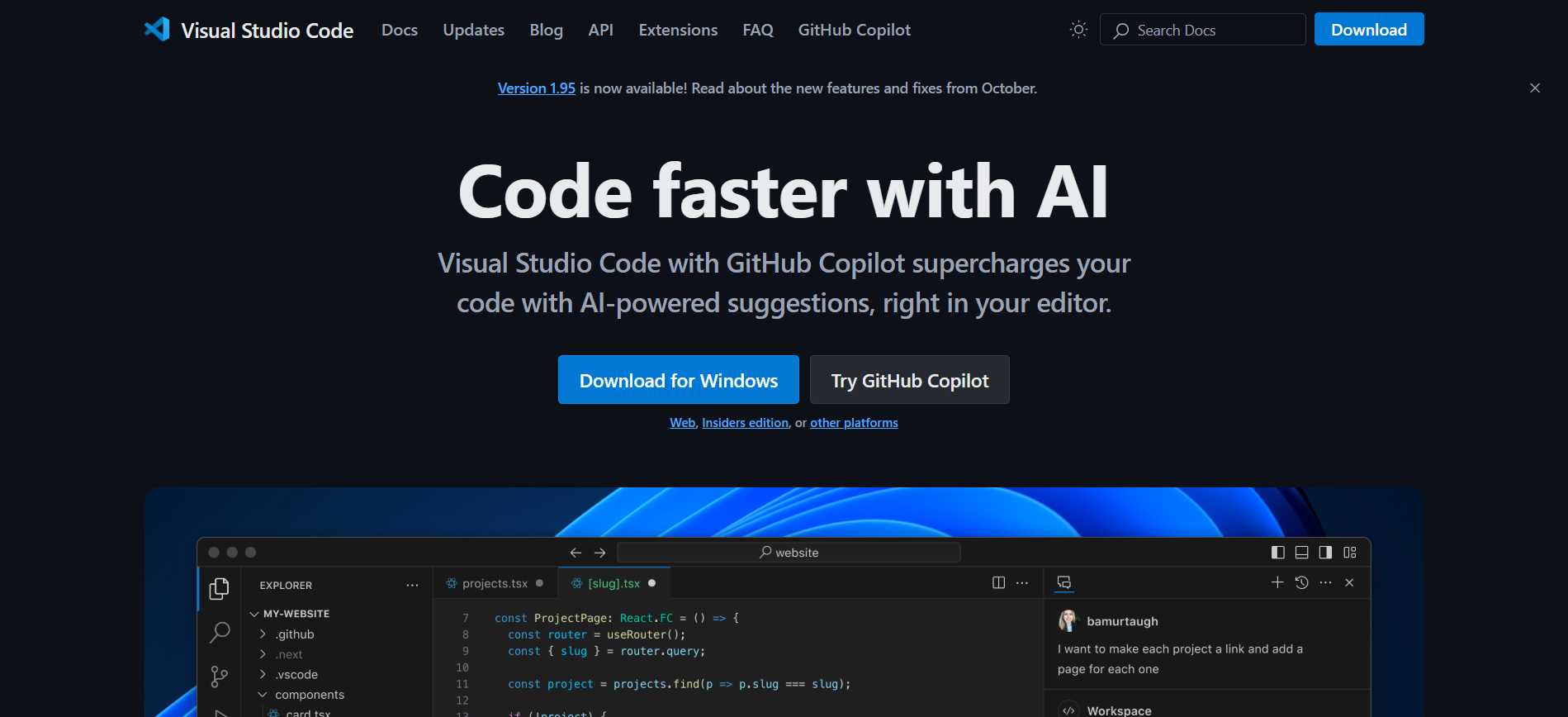
**前端开发工具**可以帮助开发者更高效地编写、调试和优化代码。常见的工具包括代码编辑器、版本控制系统、包管理器、预处理器等。学会安装常用的前端开发工具，并了解其基本配置。常用工具包括：VS Code、Git等。

* 1. VS Code 是一种开源的代码编辑器，支持多种编程语言，广泛应用于软件开发。它由微软开发，提供了丰富的功能和扩展支持，适合前端和后端开发。VS Code 具有轻量级、跨平台等特点，可以通过安装插件增强功能。
  2. Git 是一种分布式版本控制系统，用于跟踪代码变化，方便团队协作。它允许开发者在多个分支上并行工作，具有强大的合并和冲突解决能力。Git 由 Linus Torvalds 开发，并被广泛应用于开源项目和企业开发中。通过与 GitHub、GitLab 等平台集成，Git 成为现代软件开发的重要工具。

### 实现步骤

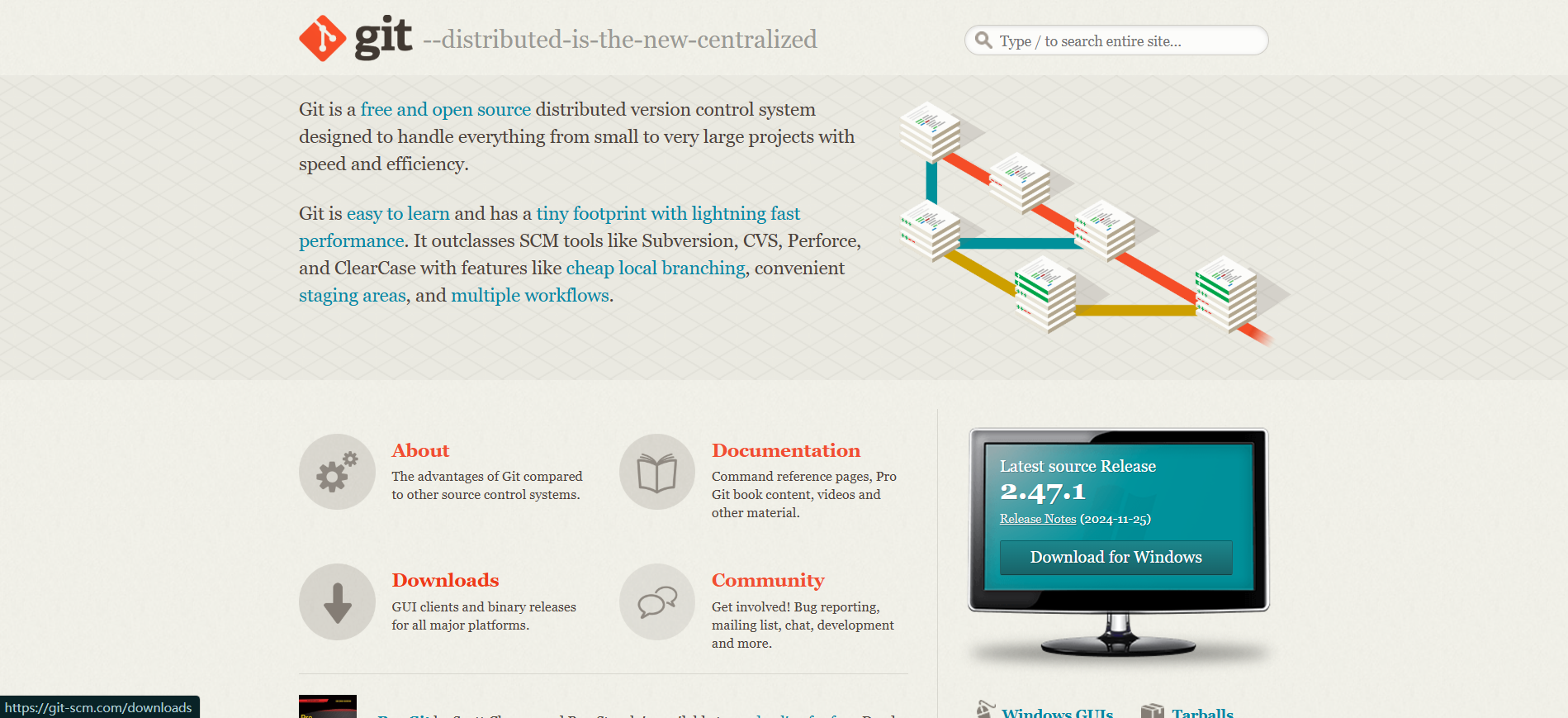
* 1. **安装 VS Code**

访问 [VS Code 官网](https://code.visualstudio.com/" \t "_new)https://code.visualstudio.com/。点击“Download”按钮，根据操作系统选择合适的版本（Windows、macOS 或 Linux）。



* + 1. 双击下载的安装文件（VSCodeSetup.exe）。
    2. 按照安装向导的指示，接受许可协议并选择安装位置。
    3. 选择附加选项（如创建桌面图标），然后点击“Install”开始安装。
  1. **安装 Git**

访问 [Git 官网](https://git-scm.com/" \t "_new)。点击“Downloads”按钮，根据操作系统选择合适的版本。



* + 1. 双击下载的安装文件（Git-x.y.z.exe）。
    2. 按照安装向导的指示，接受许可协议并选择安装位置。
    3. 保持默认选项，直到点击“Install”开始安装，安装完成后点击“Finish”

****配置 Git：****

* **打开终端**（Command Prompt、Terminal 或 Git Bash）。
* **设置用户信息**：
  + - git config --global user.name "Your Name"
    - git config --global user.email ["your.email@example.com"](mailto:\"your.email@example.com\")
* **设置默认文本编辑器**（可选）：
  + - git config --global core.editor "code --wait"
* **查看当前配置**：
  + - git config --list

## 安装MySQL数据库

### 学习目标

➢ 理解 MySQL 数据库的基本概念和功能

➢ 掌握 MySQL 数据库的安装过程

➢ 配置 MySQL 数据库环境

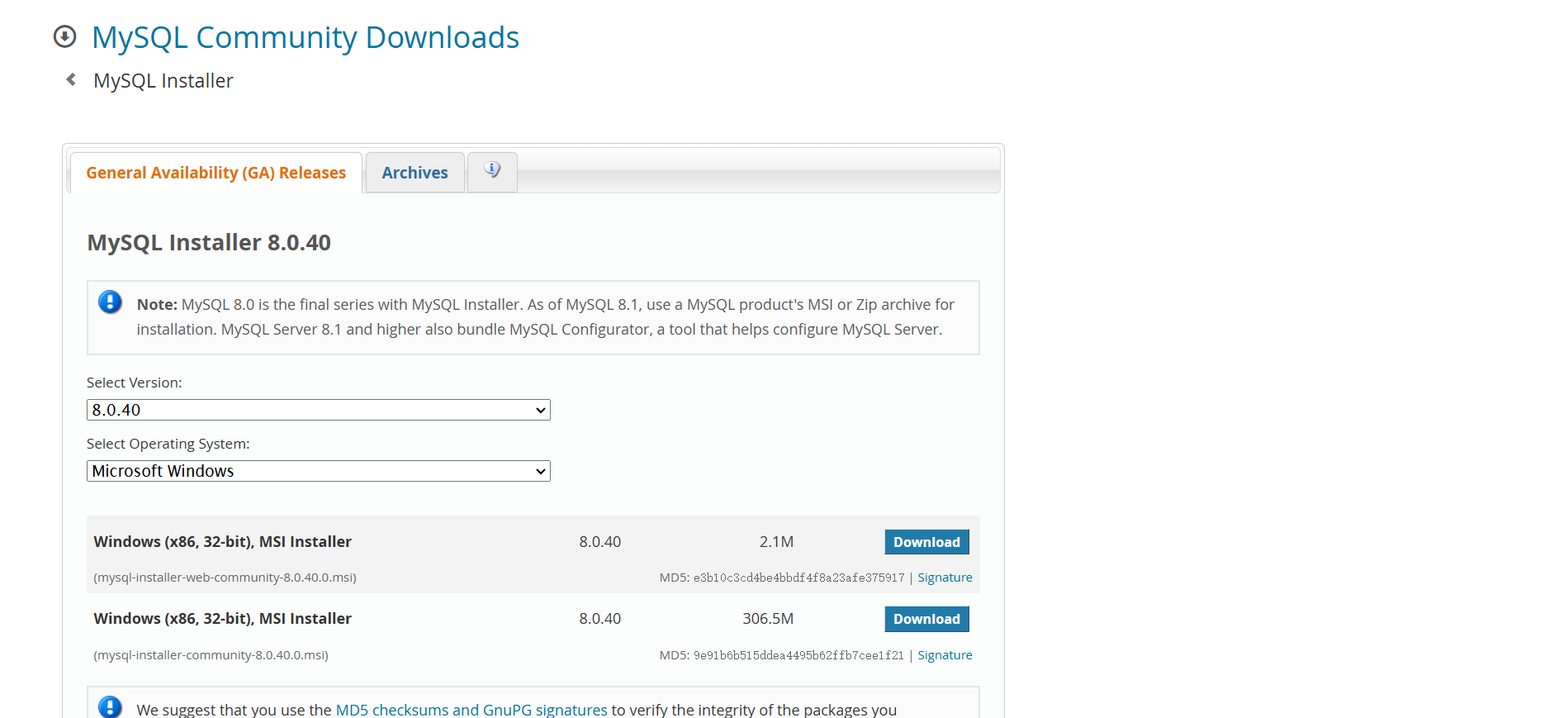
### 关于Mysql

MySQL 是一种高效的关系型数据库管理系统（RDBMS），其基础是 SQL（Structured Query Language，结构化查询语言）。支持大量结构化数据的存储与管理、复杂的 SQL 查询、事务处理、多用户访问、安全的权限管理、数据备份与恢复，并具备高性能和可扩展性。

它是目前世界上最流行的开源数据库之一，广泛应用于互联网、企业应用、数据存储等多个领域。MySQL 由瑞典公司 MySQL AB 开发，现由 Oracle 公司维护。

### 实现步骤

* 1. **下载MySQL安装包**

****

* 访问 https://dev.mysql.com/downloads/installer/。选择 MySQL Installer for Windows，根据系统类型选择合适的版本（32位或64位）。
* 下载“Web Community”版本（轻量版）或“Full”版本（包含所有MySQL组件）。
  1. **运行安装程序**
* 下载完成后，双击安装程序启动安装。
* 选择安装类型：
  + **Developer Default**：适合开发人员，包含MySQL Server、Workbench、MySQL Shell等工具。
  + **Server Only**：仅安装MySQL数据库服务器。
  + **Custom**：自定义安装内容，选择需要的组件。
  1. **配置MySQL安装**
* 选择安装路径和数据目录，默认路径通常为 C:\Program Files\MySQL\MySQL Server。
* 在产品和功能页面，点击 Next，再点击 Execute 开始安装所选组件。
  1. **初始化配置**
* 安装完成后，进入MySQL配置步骤：
  + **配置类型**：选择 Standalone MySQL Server。
  + **服务器类型**：根据使用场景选择 Development Machine（轻量配置）或 Server Machine（适合生产环境）。
  + **端口设置**：默认端口为 3306，若有冲突可以更改为其他端口。
  1. **设置MySQL Root密码**
  2. **选择MySQL身份验证方式**
  3. **启动MySQL服务**
  4. **完成安装并测试**
  5. **启动、停止和重启MySQL服务**

**通过服务管理器**：打开 服务（Windows键 + R，输入 services.msc），找到 MySQL 服务，右键点击可以启动、停止或重启服务。