

## 目录

基于 Web 客户端技术的个性化 UI 的设计和编程 .....	2
（Customized UI design and Programming based on Web client technology） .....	2
1. 前言 .....	2
1.1 毕设任务分析 .....	2
1.2 研学计划 .....	3
1.3 研究方法 .....	4
2. 技术总结和文献综述 .....	5
2.1 Web 平台和客户端技术概述 .....	5
2.2 项目的增量式迭代开发模式 .....	8
3. 内容设计概要 .....	9
3.1 分析和设计 .....	9
3.2 项目的实现和编程 .....	9
3.3 项目的运行和测试 .....	10
3.4 项目的代码提交和版本管理 .....	11
4. 移动互联时代的 UI 开发初步——窄屏终端的响应式设计 .....	12
5. 应用响应式设计技术开发可适配窄屏和宽屏的 UI .....	14
6. 个性化 UI 设计中对鼠标交互的设计开发 .....	16
7. 对触屏和鼠标的通用交互操作的设计开发 .....	18
8. UI 的个性化键盘交互控制的设计开发 .....	20
9. 谈谈本项目中的高质量代码 .....	21
10. 用 gitBash 工具管理项目的代码仓库和 http 服务器 .....	22
10.1 经典 Bash 工具介绍 .....	22
10.2 通过 gitHub 平台实现本项目的全球域名 .....	22
10.3 创建一个空的远程代码仓库 .....	22
10.4 设置本地仓库和远程代码仓库的链接 .....	23
参考文献: .....	27

# 基于 Web 客户端技术的个性化 UI 的设计和编程

(Customized UI design and Programming based on Web client technology)

科师大元宇宙产业学院 2021 级 XXX

**摘要:**近十年来,以 html5 为核心的 web 标准的软件开发技术在各个领域广泛应用。本项目选择 html5 的 web 客户端技术作为技术路线,通过对程序设计和软件开发的研究和实践,设计开发了一个个性化的用户界面(UI)的应用程序。在开发过程中,综合应用了 html 语言进行内容建模、css 语言进行外观设计、javascript 语言实现交互功能。值得一提的是,本项目的每条代码都是手工编写的,没有导入他人的任何框架和库。此外,本项目采用了响应式设计编程,可以智能地适应移动互联网时代用户屏幕多样化的需求。同时,面向对象的程序设计思想也广泛运用于本项目中,例如通过代码构建了一个通用的 pointer 模型,实现了对鼠标和触屏的控制。从工程管理的角度看,本项目采用了增量式开发模式,通过六次代码的增量式重构(A:Analysis, D:Design, I:Implementation, T:Testing)实现了项目的设计、开发和测试。此外,本项目还使用了 git 工具进行版本管理,并通过 github 进行代码分享和开源。最终,通过 gitbash 工具将代码上传到 github 上,并利用 github 提供的 http 服务器实现了全球互联网的部署,使得这个应用程序可以跨平台高效访问。

**关键词:** web, UI, git, JavaScript.

## 1. 前言

当设计和开发基于 Web 客户端技术的个性化用户界面(UI),我们置身于一个充满活力和不断演进的领域。随着互联网的普及和 Web 技术的日益成熟,个性化 UI 不仅仅是提升用户体验的手段,更是实现用户参与、满意度和留存率的关键因素之一。本论文旨在探讨在 Web 客户端环境下实现个性化 UI 的各种设计和编程技术。个性化 UI 不再局限于简单的颜色和布局选择,而是通过数据驱动的方式,根据用户的偏好、历史行为以及上下文信息,动态地调整和优化用户界面,从而提供更加个性化和精准的用户体验。我们将分析和比较目前流行的个性化 UI 技术,包括但不限于机器学习算法、数据挖掘技术、A/B 测试方法以及响应式设计的最新进展。此外,还将探讨如何利用现代前端开发框架和工具,如 React、Angular 和 Vue.js 等,实现高效且灵活的个性化 UI 编程。通过深入研究和案例分析,我们旨在揭示个性化 UI 对于提升用户满意度、增强品牌忠诚度以及优化业务成果的重要性。最终,本论文不仅为 Web 开发者和设计师提供了实用的指导和技术方案,也为未来个性化 UI 研究和应用的进一步探索提供了有价值的参考。在这个快速变化和竞争激烈的数字时代,理解和利用个性化 UI 技术不仅是一种创新的表现,更是企业和开发者赢得用户心的重要策略之一。本文的研究和讨论希望能够为此领域的研究者和实践者带来新的思路和启发,推动个性化 UI 在 Web 客户端应用中的广泛应用和进一步发展。

### 1.1 毕设任务分析

在设计和编程基于 Web 客户端技术的个性化用户界面(UI)时,任务分析是至关重要的步骤之一。任务分析帮助我们深入理解用户的需求、行为和期望,从而有效地设计和实现符合其预期的个性化 UI。以

下是任务分析的关键方面：

**用户需求分析：**收集用户信息：通过用户调研、数据分析等方式收集用户的基本信息和偏好。这包括年龄、性别、地理位置、兴趣爱好等。**理解用户目标：**确定用户使用 Web 应用的主要目标和动机。例如，购买产品、学习知识、社交互动等。

**任务建模：**识别主要任务：分析用户在 Web 应用中最常见的任务或活动，例如浏览产品、搜索信息、提交订单等。**任务流程分析：**细化每个任务的步骤和顺序，了解用户在完成任务时可能遇到的问题或需求。

**用户行为分析：**历史行为分析：利用用户的历史数据和行为模式，推断其喜好和习惯。例如，用户在特定时间段访问的页面、点击的内容等。**行为路径分析：**了解用户在 Web 应用中的典型行为路径，识别可能需要个性化的页面或功能点。

**上下文信息考虑：**设备和环境分析：考虑用户访问 Web 应用的设备类型（手机、平板、桌面电脑）和网络环境的差异。**时段和位置信息：**根据用户访问时的时间和地理位置，调整 UI 元素和内容的展示方式。

**个性化策略定义：**内容推荐：基于用户的兴趣和行为历史，推荐相关的内容或产品。例如，电子商务网站根据用户浏览历史推荐相关商品。**界面布局个性化：**根据用户的设备类型和偏好调整界面布局和元素的排列方式，提升用户的操作便利性和体验质量。通过深入的任务分析，设计团队能够更好地理解和响应用户的需求，从而设计出更加符合用户期望和行为习惯的个性化 UI。这不仅提升了用户满意度，还有助于提高 Web 应用的用户参与度和转化率。

本文的主要任务分为移动互联网时代的 UI 开发的初步阶段——窄屏阶段的响应式设计和应用响应式设计技术开发可适应宽屏和窄屏的 UI 和个性化 UI 设计中对鼠标交互的设计开发和对触屏和鼠标的通用交互操作的设计开发。最后再对 UI 的个性化键盘交互控制的设计开发。层层递进，最后形成整个项目的个性化 UI 的设计。

## 1.2 研学计划

要学习制作应用响应式设计，开发能适配窄屏和宽屏的 UI，需要掌握以下核心知识和技能，学习计划如下：

1. **HTML 和 CSS 基础：**确保对 HTML 和 CSS 有扎实的理解和掌握，包括盒模型、浮动、定位等基础知识。
2. **CSS 媒体查询：**学习使用媒体查询来根据设备的不同特性（如屏幕宽度、分辨率）应用不同的 CSS 样式。媒体查询是响应式设计的基础。
3. **响应式网格系统：**掌握响应式网格系统（如 Bootstrap 的栅格系统），这样可以更方便地构建响应式布局。
4. **Flexbox 布局：**熟悉 Flexbox 布局模型，它提供了更灵活和简洁的方式来排列、对齐和分布容器中的项目。
5. **CSS 网格布局：**学习使用 CSS 网格布局（Grid Layout），它提供了更强大和复杂的布局能力，能够更精确地控制页面的结构。
6. **移动优先设计策略：**采用移动优先的设计原则，即首先针对移动设备设计页面，再逐步增强以适应更大的屏幕尺寸。
7. **图片和多媒体的响应式处理：**学习如何制作响应式图片和视频，确保它们能够根据屏幕大小和分辨率进行适当的调整，以提高页面加载速度和用户体验。
8. **视口（Viewport）设置：**使用视口元标签（Viewport Meta Tag）来控制页面在不同设备上的显示和缩放效果。
9. **响应式框架的使用：**掌握流行的响应式框架（如 Bootstrap、Foundation 等），利用它们提供的组件和样式快速搭建响应式设计。
10. **测试和调试：**使用浏览器的开发者工具和移动设备模拟器进行测试和调试，确保设计在各种设备上表现一致和优秀的用户体验。

通过掌握以上技术和方法，你将能够有效地开发出适配不同屏幕大小和类型的响应式应用界面，提升用户体验并确保应用在各种设备上的可访问性和可用性。

### 1.3 研究方法

制作应用的响应式设计，能够适配窄屏和宽屏的 UI，需要结合多种技术和采用一定的思路：技术和实现思路：

1. 使用媒体查询：媒体查询是 **CSS3** 的一部分，允许你针对不同的媒体类型和设备特性（如屏幕宽度、高度、分辨率等）应用不同的样式表。通过定义不同的 **CSS** 规则，可以根据设备的尺寸和方向（横向或纵向）来优化布局和显示效果。

```
css
/ 示例：针对不同屏幕宽度应用不同的样式 /
@media screen and (max-width: 768px) {
    / 窄屏设备样式 /
}
@media screen and (min-width: 768px) {
    / 宽屏设备样式 /
}
...
```

2. 使用流动性布局：使用百分比或者弹性单位（如`em`、`rem`等）来定义宽度，使得元素能够随着屏幕大小的变化而自动调整大小。这种方法可以确保内容在不同屏幕尺寸下保持比例和排布的合理性。

```
css
/ 示例：使用百分比定义宽度 /
.container {
    width: 100%; / 或者使用 max-width: 100%; /
}
```

3. 响应式网格系统：使用现成的响应式网格系统（如 **Bootstrap** 的栅格系统），可以快速地构建响应式布局。这些系统允许你定义多列布局，并在不同的屏幕尺寸下进行适当的调整。

```
html
<div class="container">
  <div class="row">
    <div class="col-sm-6 col-md-4">
      <!-- 内容 -->
    </div>
    <div class="col-sm-6 col-md-8">
      <!-- 内容 -->
    </div>
  </div>
</div>
```

4. Flexbox 布局：Flexbox 布局模型提供了更灵活的方式来排列、对齐和分布容器中的项目，特别适合用于构建响应式布局。

```
css
.container {
```

```

display: flex;
flex-wrap: wrap;
}

.item {
flex: 1 1 300px; / 可根据需要调整 /
}
...

```

5. 视口（Viewport）设置：使用视口元标签（Viewport Meta Tag）可以控制设备上网页的显示方式和缩放比例，确保内容能够适应不同的屏幕尺寸和方向。

```

...html
<meta name="viewport" content="width=device-width, initial-scale=1.0">
...

```

6. 图片和多媒体的响应式处理：使用响应式图片和多媒体技术，如`<picture>`元素、`srcset`和`sizes`属性，可以根据设备的屏幕大小和分辨率加载适当的图像版本，提高页面的加载速度和用户体验。

```

...html
<picture>
  <source srcset="small.jpg" media="(max-width: 600px)">
  <source srcset="medium.jpg" media="(max-width: 1024px)">
  
</picture>

```

7. 移动优先设计策略：采用移动优先的设计方法，首先为较小的设备（如手机）设计和优化页面布局和功能，然后逐步增强以适应更大屏幕的设备。通过结合以上技术和思路，可以有效地开发出能够在各种设备上流畅运行并提供良好用户体验的响应式应用界面。

## 2. 技术总结和文献综述

### 2.1 Web 平台和客户端技术概述

Web 之父 **Tim Berners-Lee** 在发明 Web 的基本技术架构以后，就成立了 W3C 组织，该组织在 2010 年后推出的 HTML5 国际标准，结合欧洲 ECMA 组织维护的 ECMAScript 国际标准，几乎完美缔造了全球开发者实现开发平台统一的理想，直到今天，科学家与 Web 行业也还一直在致力于完善这个伟大而光荣的理想<sup>[1]</sup>。学习 Web 标准和 Web 技术，学习编写 Web 程序和应用有关工具，最终架构一套高质量代码的跨平台运行的应用，是我的毕设项目应用的技术路线。

#### 2.2.1 History

In 1989, Sir Tim Berners-Lee invented the World Wide Web (see the [original proposal](#)). He coined the term "World Wide Web," wrote the first World Wide Web server, "httpd," and the first client program (a browser and editor), "WorldWideWeb," in October 1990. He wrote the first version of the "HyperText Markup Language" (HTML), the document formatting language with the capability for hypertext links that became the primary publishing format for the Web. His initial specifications for URIs, HTTP, and HTML were refined and discussed in larger circles as Web technology spread.

### **2.2.2 A Consortium for the World Wide Web**

In 1994, the decision to form the World Wide Web Consortium came at the urging of many companies investing increasing resources into the web. Sir Tim Berners-Lee started leading the essential work of the Web Consortium team to foster a consistent architecture accommodating the rapid pace of progress in web standards for building websites, browsers, devices to experience all that the web has to offer.

In founding the World Wide Web Consortium, Sir Tim Berners-Lee created a community of peers. Web technologies were already moving so quickly that it was critical to assemble a single organization to coordinate web standards. Tim accepted the offer from MIT, who had experience with consortia, to host W3C. He required from the start that W3C have a global footprint.

### **2.2.3 Web platform and Web Programming**

Let's start with a brief description of the Web, which is short for World Wide Web. Most people say "Web" instead of "World Wide Web," and we'll follow that convention. The Web is a collection of documents, called web pages, that are shared (for the most part) by computer users throughout the world. Different types of web pages do different things, but at a minimum, they all display content on computer screens. By "content," we mean text, pictures, and user input mechanisms like text boxes and buttons.<sup>[2]</sup>

Web programming is a large field, with different types of web programming implemented by different tools. All the tools work with the core language, HTML, so almost all the web programming books describe HTML to some extent. This textbook covers HTML5, CSS, and JavaScript, all in depth. Those three technologies are known to be the pillars of client-side web programming. With client side web programming, all web page calculations are performed on end users' computers (the client computers).<sup>[3]</sup>

1989 年，蒂姆·伯纳斯-李爵士发明了万维网（见原始提案）。1990 年 10 月，他创造了“万维网”一词，编写了第一台万维网服务器“httpd”和第一个客户端程序（浏览器和编辑器）“万维网。他编写了“超文本标记语言”（HTML）的第一个版本，这是一种具有超文本链接功能的文档格式化语言，成为网络的主要发布格式。随着 Web 技术的普及，他对 URI、HTTP 和 HTML 的最初规范得到了改进，并在更大的圈子里进行了讨论。

### 2.2.2 万维网联盟

1994 年，在许多公司的敦促下，成立了万维网联盟，向网络投入了越来越多的资源。Tim Berners-Lee 爵士开始领导网络联盟团队的重要工作，以促进一致的架构，适应网络标准的快速发展，从而构建网站、浏览器和设备，体验网络所提供的一切。蒂姆·伯纳斯-李爵士在创建万维网联盟时创建了一个同行社区。Web 技术已经发展得如此之快，因此组建一个单一的组织来协调 Web 标准至关重要。蒂姆接受了麻省理工学院的邀请，他在联盟方面有丰富的经验，主持 W3C。他从一开始就要求 W3C 具有全球影响力。

### 2.2.3 Web 平台和 Web 编程

让我们先简单介绍一下 Web，它是万维网的缩写。大多数人说“网络”而不是“万维网”，我们将遵循这一惯例。网络是一组文档，称为网页，由世界各地的计算机用户共享（大部分）。不同类型的网页做不同的事情，但至少，它们都在电脑屏幕上显示内容。所谓“内容”，我们指的是文本、图片和用户输入机制，如文本框和按钮。2. Web 编程是一个很大的领域，通过不同的工具实现不同类型的 Web 编程。所有的工具都使用核心语言 HTML，所以几乎所有的网络编程书籍都在某种程度上描述了 HTML。这本教科书涵盖了 HTML5、CSS 和 JavaScript，所有这些都是深入的。众所周知，这三种技术是客户端 web 编程的支柱。使用客户端 web 编程，所有网页计算都在最终用户的计算机（客户端计算机）上执行。

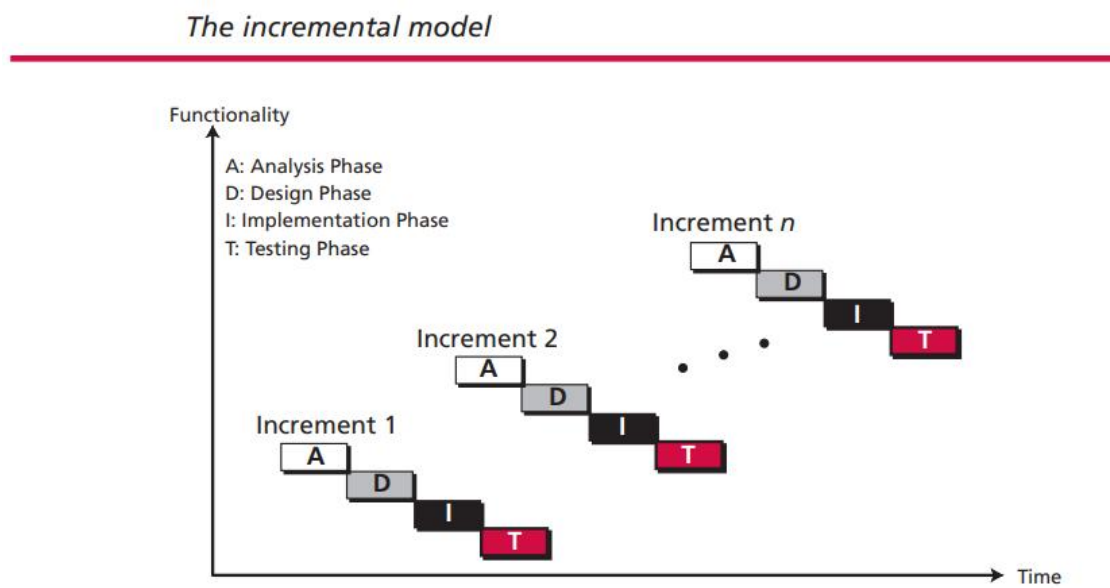
Web 应用的程序设计体系由三大语言有机组成：**HTML, CSS, JavaScript**。这三大语言的组合也体现了人类社会化大生产分工的智慧，可以看作用三套相对独立体系实现了对一个信息系统的描述和控制，可以总结为：**HTML** 用来描述结构（**Structure**）、**CSS** 用来描述外表（**presentation**）、**Javascript** 用来描述行为（**Behavior**）<sup>[3]</sup>；这也可以用经典的 MVC 设计模式来理解 Web 平台架构的三大基石，**Model** 可以理解为 HTML 标记语言建模，**View** 可以理解为用 CSS 语言来实现外观，**Controller** 则可理解为用 JavaScript 结合前面二个层次，实现了在微观和功能层面的代码控制。

2.2 项目的增量式迭代开发模式

本项目作为一个本科专业学生毕业设计的软件作品，与单一用途的程序相比较为复杂，本项目所涉及的手写代码量远超过简单一二个数量级以上，从分析问题的到初步尝试写代码也不是能在几天内能落实的，可以说本项目是一个系统工程，因此需要从软件工程的管理视角来看待和规范项目的编写过程。

而本项目考虑选择的软件工程开发过程管理模式有两种经典模型：瀑布模型（The waterfall model）和增量式迭代模型(The incremental model)。而任何开发模式则都必须同样经历四个阶段：分析（Analysis）、设计（Design）、实施（Implementation）、测试（test）。

瀑布模型需要专业团队完美的配合，从分析、设计到实施，最后到测试，任何阶段的开始必须基于上一阶段的完美结束。而这对于我们大多数普通开发者是不太现实的，作为小微开发者由于身兼数职，其实无法 1 次就能完美完成任何阶段的工作，比如在实施过程中，开发者会发现前面的设计存在问题，则必须在下一次迭代项目时改良设计。在当今开源的软件开发环境中，开发者在软件的开发中总是在不断地优化设计、重构代码，持续改进程序的功能和代码质量。因此在本项目的开发中，也采用了增量模型的开发模式<sup>[5]</sup>。本项目中我一共做了六次项目的开发迭代，如下图 2-1 所示：



The incremental model

In the incremental model, software is developed in a series of steps. The developers first complete a simplified version of the whole system. This version represents the entire system but does not include the details. Figure shows the incremental model concept.

In the second version, more details are added, while some are left unfinished, and the system is tested again. If there is a problem, the developers know that the problem is with the new functionality. They do not add more functionality until the existing system works properly. This process continues until all required functionality has been added. <sup>[5]</sup>



在增量模型中，软件是按一系列步骤开发的。开发人员首先完成了整个系统的简化版本。此版本表示整个系统，但不包括详细信息。图显示了增量模型的概念。

在第二个版本中，添加了更多的细节，而有些细节尚未完成，系统将再次进行测试。如果出现问题，开发人员就会知道问题出在新功能上。在现有系统正常工作之前，它们不会添加更多功能。此过程一直持续到添加了所有必需的功能为止。5。

### 3. 内容设计概要

#### 3.1 分析和设计

这一步是项目的初次开发，本项目最初使用人们习惯的“三段论”式简洁方式开展内容设计，首先用一个标题性信息展示 logo 或文字标题，吸引用户的注意力，迅速表达主题；然后展现主要区域，也就是内容区，“内容为王”是项目必须坚守的理念，也是整个 UI 应用的重点；最后则是足部的附加信息，用来显示一些用户可能关心的细节变化。如图 4-1 用例图所示：

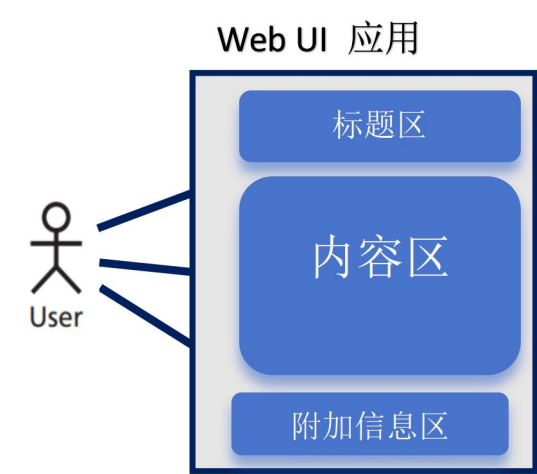


图 4-1 用例图

#### 3.2 项目的实现和编程

##### 一、HTML 代码编写如下：

```
<header>
    《 我的毕设题目 》
</header>
<main>
    我的主题内容：‘读好书、练思维、勤编程’  计算思维系列课程
</main>
<footer>
    Copyright 韩钦生  江西科技师范大学 2024-2025
</footer>
```

## 二、CSS 代码编写如下：

```
*{
    margin: 10px;
    text-align: center;
    font-size:30px ;
}
header{
    border: 2px solid blue;
    height: 200px;
}

main{
    border: 2px solid blue;
    height: 400px;
}
footer{
    border: 2px solid blue;
    height: 100px;
}
a{
    display: inline-block ;
    padding:10px ;
    color: white;
    background-color: blue;
    text-decoration: none ;
}
```

### 3.3 项目的运行和测试

项目的运行和测试至少要通过二类终端，本文此处仅给出 PC 端用 Chrome 浏览器打开项目的结果，如下图 4-2 所示。由于本项目的阶段性文件已经上传 [github](#) 网站，移动端用户可以通过扫描图 4-3 的二维码，运行测试本项目的第一次开发的阶段性效果。



图 4-2 PC 端运行效果图

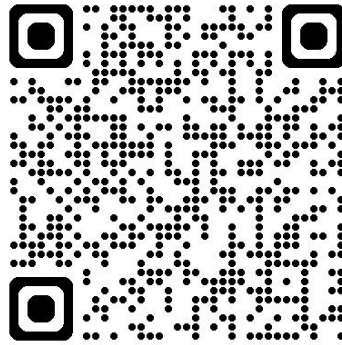


图 4-3 移动端二维码

### 3.4 项目的代码提交和版本管理

本项目的文件通过 `gitBash` 工具管理，作为项目的第一次迭代，在代码提交和版本管理环节，我们的目标是建立项目的基本文件结构，还有设置好代码仓库的基本信息：如开发者的名字和电子邮件。

进入 `gitBash` 命令行后，按次序输入以下命令：

```
$ cd /
$ mkdir webUI
$ cd webUI
$ git init
$ git config user.name
$ git config user.email marsterlijh@jxstnu.edu.cn
$ touch index.html myCss.css
```

编写好 `index.html` 和 `myCss.css` 的代码，测试运行成功后，执行下面命令提交代码：

```
$ git add index.html myCss.css
$ git commit -m 项目第一版：“三段论”式的内容设计概要开发
```

成功提交代码后，`gitbash` 的反馈如下所示：

```
$ git commit -m 项目第一版：“三段论”式的内容设计概要开发
[master (root-commit) 32de024] 项目第一版：“三段论”式的内容设计概要开发
2 files changed, 46 insertions(+)
create mode 100644 index.html
create mode 100644 myCss.css
```

项目代码仓库自此也开启了严肃的历史记录，我们可以输入日志命令查看，

```
$ git log
```

`gitbash` 反馈代码的仓库日志如下所示：

```
enovo@DESKTOP-98C266C MINGW64 /d/hqs_text/abc (master)
git log
commit e68c32b6a55d0a96f6235e752914d4f243682b1b (HEAD -> mas
author: unknown <2106150287@qq.com>
date: Wed Jun 19 13:46:53 2024 +0800

    项目第一版: '

enovo@DESKTOP-98C266C MINGW64 /d/hqs_text/abc (master)
```

#### 4. 移动互联时代的 UI 开发初步——窄屏终端的响应式设计

##### Responsive Design--Accommodating Display Hardware

The display hardware used with computers varies widely, with the size and resolution of a display depending on cost. Rather than have a version of each web page for each type of display, designers chose to make web pages give general layout guidelines and allow a browser to choose how to display the page on a given computer. Thus, a web page does not give many details. For example, the author of a web page can specify that a group of sentences form a paragraph, but the author cannot specify details such as the exact length of a line or whether to indent the beginning of the paragraph.<sup>[1]</sup>

Allowing a browser to choose display details has an interesting consequence: a web page may appear differently when viewed through two browsers or on two computers that have dissimilar hardware. If one screen is wider than another, the length of a line of text or the size of images that can be displayed differs. The point is: A web page gives general guidelines about the desired presentation; a browser chooses details when displaying a page. As a result, the same web page can appear slightly different when displayed on two different computers or by different browsers.<sup>[1]</sup>

响应式设计——适应显示硬件与计算机一起使用的显示器硬件变化很大，显示器的大小和分辨率取决于成本。设计师们没有为每种类型的显示器提供每个网页的版本，而是选择让网页提供总体布局指南，并允许浏览器选择如何在给定的计算机上显示页面。因此，网页并不能提供很多细节。例如，网页的作者可以定一组句子组成一个段落，但作者不能指定细节，如一行的确切长度或是否缩进段落的开头。1.允许浏览器选择显示详细信息有一个有趣的结果：当通过两个浏览器或在两台硬件不同的计算机上查看时，网页可能会出现不同的外观。如果一个屏幕比另一个屏幕宽，则可以显示的文本行的长度或图像的大小不同。重点是：网页提供了关于所需演示的一般指南；浏览器在显示页面时选择详细信息。因此，当在两台不同的计算机上或通过不同的浏览器显示时，同一个网页可能看起来略有不同。。

分析移动互联时代的多样化屏幕的需求。

用 JavaScript 开动态读取显示设备的信息，然后按设计，使用 js+css 来部署适配当前设备的显示的代码。实现代码

用汉语言来描述我们是如何实现的，与上一阶段比较，本阶段初次引入了 em 和 % ，这是 CSS 语言中比较高阶的语法，可以有效地实现我们的响应式设计 。如代码块 4-1 所示：

```
<style>
*{
  margin: 10px;
  text-align: center;
}

header{
  border: 2px solid blue;
  height: 15%;
  font-size: 1.66em;
}

main{
  border: 2px solid blue;
  height: 70%;
  font-size: 1.2em;
}

nav{
  border: 2px solid blue;
  height: 10%;
}

nav button{
  font-size: 1.1em;
}
```

```

    footer{
        border: 2px solid blue;
        height: 5%;
    }
</style>

```

代码块 4-1

用汉语言来描述我们是如何实现的：与上一阶段比较，本阶段首次使用了 JavaScript，首先创建了一个 UI 对象，然后把系统的宽度和高度记录在 UI 对象中，又计算了默认字体的大小，最后再利用动态 CSS，实现了软件界面的全屏设置。如代码块 4-2 所示：

```

<script>
    var UI = {};
    UI.appWidth = window.innerWidth > 600 ? 600 : window.innerWidth ;
    UI.appHeight = window.innerHeight;
    const LETTERS = 22 ;
    const baseFont = UI.appWidth / LETTERS;

    //通过更改 body 对象的字体大小，这个属性能够遗传其子子孙孙
    document.body.style.fontSize = baseFont + "px";
    //通过把 body 对象的宽度和高度设置为设备/屏幕的宽度和高度，实现全屏。
    //通过 CSS 对子对象百分比（纵向）的配合，从而实现响应式设计的目标。
    document.body.style.width = UI.appWidth - 2*baseFont + "px" ;
    document.body.style.height = UI.appHeight - 4*baseFont + "px";
</script>

```

代码块 4-2

## 5. 应用响应式设计技术开发可适配窄屏和宽屏的 UI

适用移动互联时代阐述移动互联时代的用户终端的多样性，用 css 语言和 JavaScript 语言实现响应式设计实现代码：<body >

```

<header>
    <p id="book">
        有待商榷
    </p>
</header>
<nav>
    <button>向前</button>
    <button>向后</button>
    <button>暂停</button>
</nav>

```

```

<main id="main">

```

```

<div id="bookface">
  书的封面图
</div>
</main>

```

```

<footer>

```

```

  Copyright from 江西科技师范大学 2022--2025

```

```

</footer>
<div id="aid">
  <p>用户键盘响应区</p>

```

```

</div>
<script>
var UI = {};
if(window.innerWidth>600){
  UI.appWidth=600;
}else{
  UI.appWidth = window.innerWidth;
}

```

```

UI.appHeight = window.innerHeight;

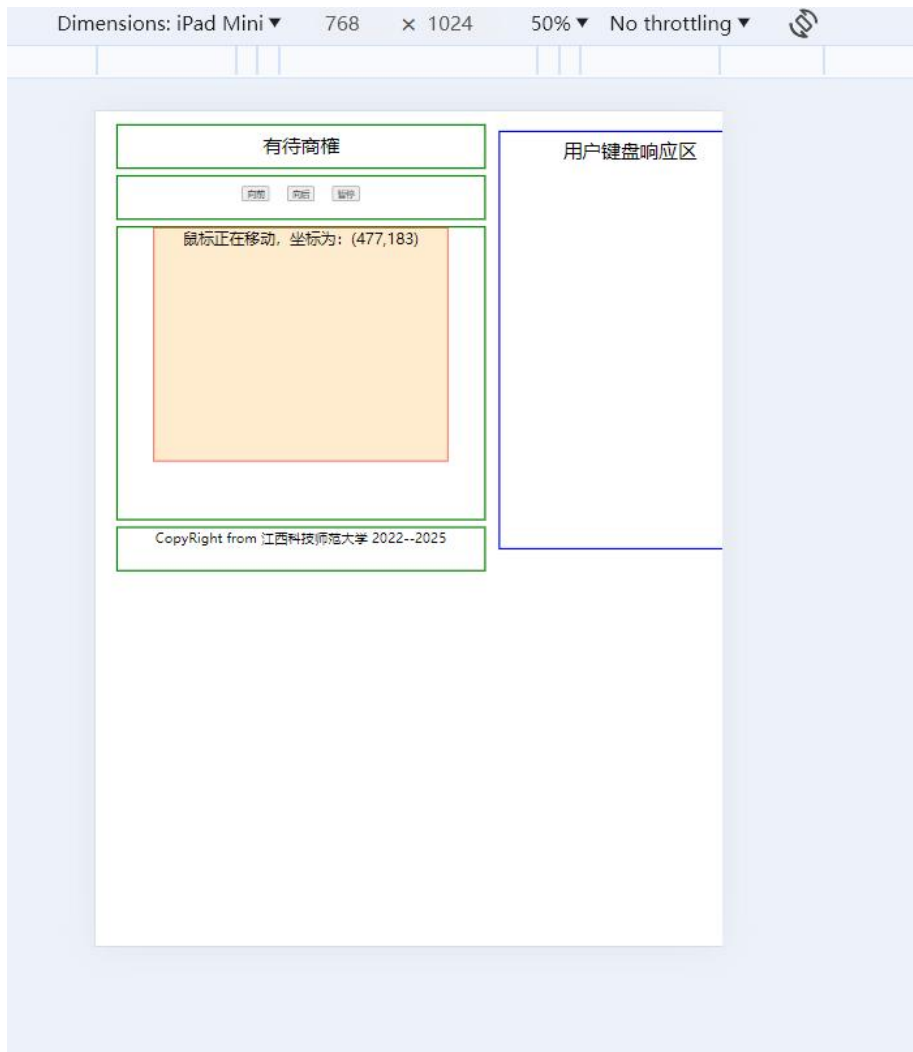
```

```

let baseFont = UI.appWidth /20;
//通过改变 body 对象的字体大小，这个属性可以影响其后代
document.body.style.fontSize = baseFont + "px";
//通过把 body 的高度设置为设备屏幕的高度，从而实现纵向全屏
//通过 CSS 对子对象百分比（纵向）的配合，从而达到我们响应式设计的目标
document.body.style.width = UI.appWidth + "px";
document.body.style.height = UI.appHeight - 62 + "px";
if(window.innerWidth<1000){
  $("aid").style.display='none';
}
$("aid").style.width=window.innerWidth-UI.appWidth-30+'px';
$("aid").style.height= UI.appHeight-62+'px';

```

实现页面：



## 6. 个性化 UI 设计中对鼠标交互的设计开发

对于个性化中对于鼠标交互的设计开发，是在应用响应式设计开发可适配宽屏和窄屏 UI 的基础上进行进一步的修改和精进而成，本节主要是研究在上节的基础上加上 UI 的鼠标交互。本节的相关逻辑代码如下所示：

```
//尝试对鼠标设计 UI 控制
var mouse={};
mouse.isDown= false;
mouse.x= 0;
mouse.y= 0;
mouse.deltaX=0;
$("#bookface").addEventListener("mousedown",function(ev){
    mouse.isDown=true;
    mouse.x= ev.pageX;
    mouse.y= ev.pageY;
```

```
$("#bookface").textContent= "鼠标按下了, 坐标为: "+"("+mouse.x+", "+mouse.y+")";
```



```
});
$("bookface").addEventListener("mouseup",function(ev){
    mouse.isDown=false;
    mouse.x= 0 ;
    mouse.y= 0 ;
    $("bookface").textContent= "鼠标松开!";
    if(Math.abs(mouse.deltaX) > 50){
        $("bookface").textContent += "，这是有效拖动! " ;
    }
});
```

```
});
$("bookface").addEventListener("mousemove",function(ev){
    ev.preventDefault();
```

```
if (mouse.isDown && Math.abs($('bookface').offsetLeft) < UI.appWidth / 5){
    console.log("认可的 mouse 事件: down and moving");
    mouse.deltaX = ev.pageX - mouse.x ;
    $('bookface').style.left = $('bookface').offsetLeft + mouse.deltaX + 'px' ;
    mouse.deltaX = 0 ;
}
```

```
});
```

实现画面如下：



## 7. 对触屏和鼠标的通用交互操作的设计开发

设计开发支持触屏和鼠标的通用交互操作时，需要考虑以下关键事项：

1. 大小和间距的适应性目标大小： 触摸屏幕上的可点击区域应足够大，使用户能够轻松触摸而不会出现误触。推荐的触摸目标大小为至少 7mm x 7mm（48px x 48px）。间距： 在触摸屏上，元素之间的间距应该足够大，以免用户不小心点击了不想点击的元素。同时，也需要在鼠标操作上保持合理的间距，避免元素过于密集而导致误点。
2. 触摸手势和鼠标操作的兼容性手势支持： 对于触摸屏幕，支持常见的手势操作如滑动、捏合和双击。确保这些手势的响应和效果与用户期待的一致。鼠标操作： 鼠标操作通常包括悬停、点击和拖拽等，需要确保这些操作在鼠标上的体验与触摸屏上的手势操作一样流畅和直观。
3. 反馈和动画效果点击反馈： 触摸屏幕上，点击时需要提供明确的视觉反馈，如按钮的按下效果或者颜色变化，以确保用户知道他们点击的是正确的区域。动画效果： 在交互过程中适度使用动画效果可以增强用户体验，但要确保这些动画不会影响操作的流畅性，尤其是在触摸屏上。
4. 文本输入和编辑虚拟键盘： 对于需要文本输入的场景，如搜索框或表单，触摸屏幕需要弹出合适的虚拟键盘，并且可以自由调整大小和位置。编辑操作： 在文本编辑时，支持光标控制、文本选择和剪切/复制/粘贴等鼠标和触摸操作，确保操作的便捷性和一致性。
5. 可访问性和无障碍设计导航和焦点管理： 确保键盘焦点和触摸操作之间的切换流畅，以支持无障碍用户。字体和对比度： 文本应该具有足够的大小和对比度，以便用户可以轻松阅读。在触摸屏上特别需要注意字体大小的适应性。
6. 设备适配性响应式设计： 确保界面和布局在不同大小的屏幕和设备上都能良好显示和操作，包括智能手机、平板电脑和桌面电脑。浏览器兼容性： 测试和确保在不同的主流浏览器中的兼容性，包括 Chrome、Safari、Firefox 和 Edge 等。
7. 用户反馈和测试

用户体验测试： 进行定期的用户体验测试，尤其是针对触摸屏和鼠标操作的用户反馈，以便不断优化和改进交互设计。错误处理和调整： 当用户反馈或测试中发现问题时，及时进行调整和修复，以提高整体的用户满意度和使用便捷性。综上所述，设计开发触摸屏和鼠标通用交互操作时，关注大小和间距、手势和鼠标操作的兼容性、反馈和动画效果、文本输入和编辑、可访问性、设备适配性以及用户反馈和测试是确保良好用户体验的关键步骤。

本节的关键代码如下：

```
let handleMoving = function(ev) {
    ev.preventDefault();
    if (ev.touches) {
        if (Pointer.isDown) {
            console.log("Touch is moving");
            Pointer.deltaX = parseInt( ev.touches[0].pageX - Pointer.x );
            $("bookface").textContent= "正在滑动触屏，滑动距离：" + Pointer.deltaX + "px 。";
            $('bookface').style.left = Pointer.deltaX + 'px' ;
        }
    } else {
        if (Pointer.isDown) {
            console.log("Pointer isDown and moving");
            Pointer.deltaX = parseInt( ev.pageX - Pointer.x );
```

```
        $("bookface").textContent= "正在拖动鼠标, 距离: " + Pointer.deltaX + "px 。";
        $('bookface').style.left = Pointer.deltaX + 'px' ;
    }
}
};
```

```

$("bookface").addEventListener("mousedown",handleBegin );
$("bookface").addEventListener("touchstart",handleBegin );
$("bookface").addEventListener("mouseup", handleEnd );
$("bookface").addEventListener("touchend",handleEnd );
$("bookface").addEventListener("mouseout", handleEnd );
$("bookface").addEventListener("mousemove", handleMoving);
$("bookface").addEventListener("touchmove", handleMoving);
} //Code Block  end
function $(ele) {
    if (typeof ele !== 'string') {
        throw("自定义的$函数参数的数据类型错误, 实参必须是字符串!");
        return
    }
    let dom = document.getElementById(ele) ;
    if(dom) {
        return dom ;
    }else{
        dom = document.querySelector(ele) ;
        if (dom) {
            return dom ;
        }else{
            throw("执行$函数未能在页面上获取任何元素, 请自查问题!");
            return ;
        }
    }
}
```

实现画面如下:



## 8. UI 的个性化键盘交互控制的设计开发

阐述探索和利用 `keydown` 和 `keyup` 键盘底层事件，为未来 UI 的键盘功能提供底层强大的潜力。

因为系统中只有一个键盘，所以我们在部署代码时，把键盘事件的监听设置在 DOM 文档最大的可视对象——`body` 上，通过测试，不宜把键盘事件注册在 `body` 内部的子对象中。代码如下所示：

```
$("body").addEventListener("keydown",function(ev){
    ev.preventDefault(); //增加“阻止事件对象的默认事件后”，不仅 keypress 事件将不再响应，
    而且系统的热键，如“F5 刷新页面/Ctrl+R ”、“F12 打开开发者面板”等也不再被响应
    let k = ev.key;
    let c = ev.keyCode;
    $("keyStatus").textContent = "按下键 : " + k + " , "+ "编码 : " + c;
});
```

```
$("body").addEventListener("keyup",function(ev){
    ev.preventDefault();
    let key = ev.key;
    $("keyStatus").textContent = key + " 键已弹起" ;
    if (printLetter(key)){
        $("typeText").textContent += key ;
    }
}
```

```
function printLetter(k){
    if (k.length > 1){ //学生须研究这个逻辑的作用
        return false ;
    }
    let puncs =
    ['~','`','!','@','#','$','%','^','&','*','(',')','-','_','+','=',' ','.',';','<','>','?','/',' ','\','\"'] ;
    if ( (k >= 'a' && k <= 'z') || (k >= 'A' && k <= 'Z')
    || (k >= '0' && k <= '9')) {
        console.log("letters") ;
        return true ;
    }
    for (let p of puncs ){
        if (p === k) {
            console.log("puncs") ;
            return true ;
        }
    }
    return false ;
}
//提出更高阶的问题，如何处理连续空格和制表键 tab?
```

```
    }    //function printLetter(k)
  });
```

## 9. 谈谈本项目中的高质量代码

This is a book about instructing computers. Computers are about as common as screwdrivers today, but they are quite a bit more complex, and making them do what you want them to do isn't always easy.

If the task you have for your computer is a common, well-understood one, such as showing you your email or acting like a calculator, you can open the appropriate application and get to work. But for unique or open-ended tasks, there probably is no application.

That is where programming may come in. Programming is the act of constructing a program--a set of precise instructions telling a computer what to do. Because computers are dumb, pedantic beasts, programming is fundamentally tedious and frustrating. Fortunately, if you can get over that fact, and maybe even enjoy the rigor of thinking in terms that dumb machines can deal with, programming can be rewarding. It allows you to do things in seconds that would take forever by hand. It is a way to make your computer tool do things that it couldn't do before. And it provides a wonderful exercise in abstract thinking.<sup>[6]</sup>

这是一本关于指导计算机的书。今天，计算机和螺丝刀一样常见，但它们要复杂得多，让它们做你想让它们做的事情并不总是那么容易。如果你的电脑任务是一个常见的、众所周知的任务，比如给你看电子邮件或像计算器一样工作，你可以打开相应的应用程序开始工作。但对于独特或开放式的任务，可能没有应用程序。这就是编程的用武之地。编程是构建程序的行为——一组精确的指令告诉计算机该做什么。因为计算机是愚蠢、迂腐的野兽，编程从根本上来说是乏味和令人沮丧的。幸运的是，如果你能克服这一事实，甚至可以享受愚蠢机器所能处理的思维的严谨性，那么编程是有回报的。它可以让你在几秒钟内完成那些需要你用手才能完成的事情。这是一种让你的计算机工具做以前做不到的事情的方法。它提供了一个很好的抽象思维练习。

创建一个 `Pointer` 对象，践行 MVC 设计模式，设计一套代码同时对鼠标和触屏实现控制。

面向对象思想，封装，抽象，局部变量，函数式编程，逻辑。

（围绕着抽象定义函数、代码块、模型设计以及降低全局变量的使用来写）

## 10. 用 gitBash 工具管理项目的代码仓库和 http 服务器

### 10.1 经典 Bash 工具介绍

When we speak of the command line, we are really referring to the shell. The shell is a program that takes keyboard commands and passes them to the operating system to carry out. Almost all Linux distributions supply a shell program from the GNU Project called bash. The name is an acronym for bourne-again shell, a reference to the fact that bash is an enhanced replacement for sh, the original Unix shell program written by Steve Bourne.<sup>[7]</sup> Like Windows, a Unix-like operating system such as Linux organizes its files in what is called a hierarchical directory structure. This means they are organized in a tree-like pattern of directories (sometimes called folders in other systems), which may contain files and other directories. The first directory in the file system is called the root directory. The root directory contains files and sub directories, which contain more files and sub directories, and so on.<sup>[7]</sup>

当我们谈到命令行时，我们实际上指的是 shell。shell 是一个接受键盘命令并将其传递给操作系统执行的程序。几乎所有的 Linux 发行版都提供了一个名为 bash 的 GNU 项目的 shell 程序。这个名字是 bourne-reagain shell 的首字母缩写，指的是 bash 是由 Steve bourne 编写的原始 Unix shell 程序 sh 的增强替代品。像 Windows 一样，像 Linux 这样的类 Unix 操作系统将其文件组织在所谓的分层目录结构中。这意味着它们以树状目录模式组织（有时在其他系统中称为文件夹），其中可能包含文件和其他目录。文件系统中的第一个目录称为根目录。根目录包含文件和子目录，其中包含更多的文件和子文件夹，依此类推。

### 10.2 通过 gitHub 平台实现本项目的全球域名


### 10.3 创建一个空的远程代码仓库

## Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Required fields are marked with an asterisk (\*).

Owner \*

 masterLijh ▾

Repository name \*

/ userName.github.io

✓ userName.github.io is available.

Great repository names are short and memorable. Need inspiration? How about **expert-rotary-phone** ?

Create repository

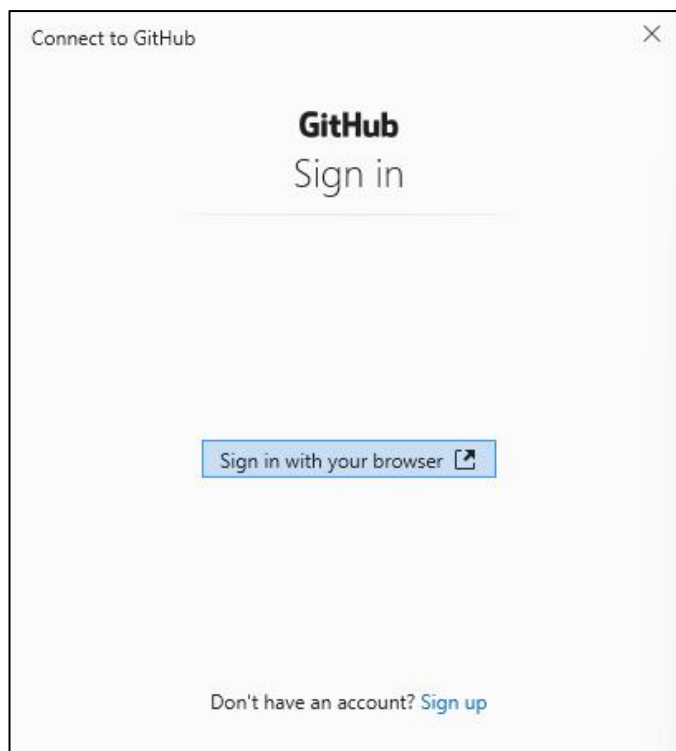
点击窗口右下角的绿色“Create repository”，则可创建一个空的远程代码仓库。

### 10.4 设置本地仓库和远程代码仓库的链接

进入本地 webUI 项目的文件夹后，通过下面的命令把本地代码仓库与远程建立密钥链接

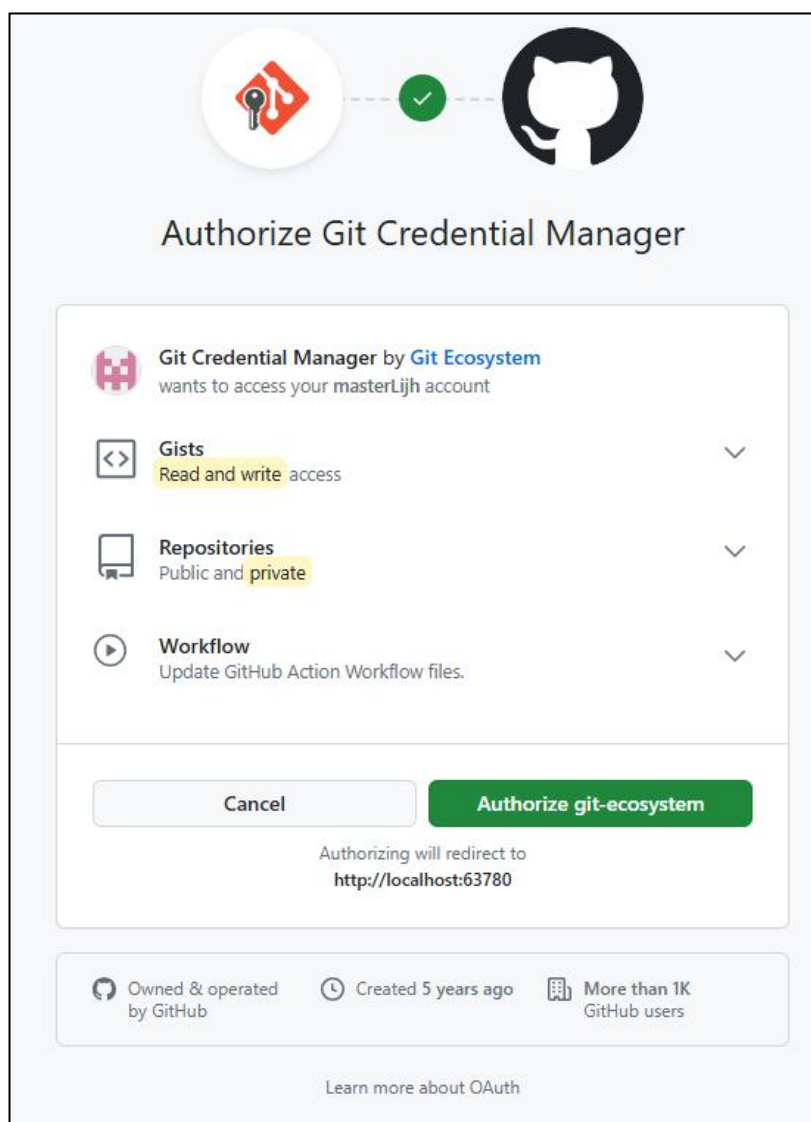
```
$ echo "WebUI 应用的远程 http 服务器设置" >> README.md
$ git init
$ git add README.md
$ git commit -m "这是我第一次把代码仓库上传至 gitHub 平台"
$ git branch -M main
$ git remote add origin
    https://github.com/H-Q-S/hqs.git
$ git push -u origin main
```

本项目使用 window 平台，gitbash 通过默认浏览器实现密钥生成和记录，第一次链接会要求开发者授权，如下图所示：

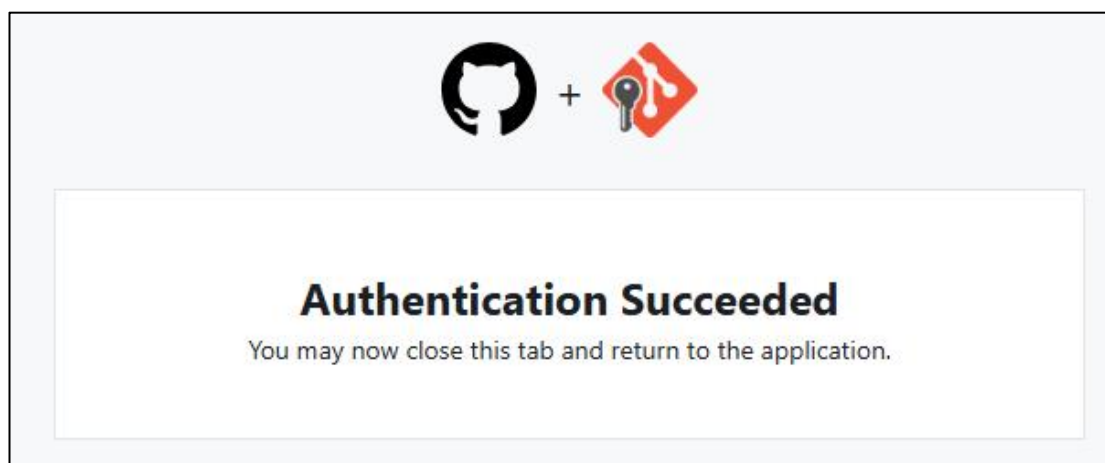


再次确认授权 `gitBash` 拥有访问改动远程代码的权限，如下图所示：





最后，GitHub 平台反馈：gitBash 和 gitHub 平台成功实现远程链接。



从此，我们无论在本地做了任何多次代码修改，也无论提交了多少次，上传远程时都会把这些代码和修改的历史记录全部上传 github 平台，而远程上传命令则可简化为一条：`git push`，极大地方便了本 Web 应用的互联网发布。

远程代码上传后，项目可以说免费便捷地实现了在互联网的部署，用户可以通过域名或二维码打开，本次使用 PC 的微软 Edge 浏览器打开，本文截取操作中间的效果图，如下所示：

 有待商榷

导航一

导航二

导航三

鼠标松开! 本次算无效拖动!

韩钦生 江西科技师范大学 2022--2025

用户键盘响应区

按下键：Alt

全文完成，谢谢！

## 参考文献:

- [1] W3C. W3C's history. W3C Community. [EB/OL]. <https://www.w3.org/about/>.  
<https://www.w3.org/about/history/>. 2023.12.20
- [2] Douglas E. Comer. The Internet Book [M] (Fifth Edition). CRC Press Taylor & Francis Group, 2019: 217-218
- [3] John Dean,PhD. Web programming with HTML5,CSS,and JavaScript[M]. Jones & Bartlett Learning,LLC. 2019: 2
- [4] John Dean,PhD. Web programming with HTML5,CSS,and JavaScript[M]. Jones & Bartlett Learning,LLC. 2019: xi
- [5] Behrouz Forouzan. Foundations of Computer Science[M](4th Edition). Cengage Learning EMEA,2018: 274--275
- [6] Marijn Haverbeke. Eloquent JavaScript 3rd edition. No Starch Press,Inc, 2019.
- [7] William Shotts. The Linux Command Line, 2nd Edition [ M ]. No Starch Press, Inc, 245 8th Street, San Francisco, CA 94103, 2019: 3-7