1. 导言

基于 Express 后端，Pug 模板、bootstrap4框架前端以及 MongoDB 数据库构建的评书网站。该应用旨在提供一系列核心功能，以满足用户的基本需求。主要功能包括用户身份验证、图书发布和评价、图书分类查询等。

1. 背景

使用的公开数据源

APIs: 用户注册、用户登录、退出登录、分类查询图书、添加图书、删除图书、编辑图书、评论图书、查看我的图书

用户输入：  
 登录：用户名、密码

注册：用户名、昵称、密码、确认密码、角色

评论：评分、评价内容

竞争应用的都有用户注册、登录、图书信息查看等功能

1. 核心功能

What makes the proposed app relevant/distinctive?

提供用户友好的图书管理和分享平台。它可能在于用户能够轻松地搜索图书、发布评论、评价图书，并能与其他用户互动。它可能的特色之一是提供用户友好的界面和交互功能，使用户可以方便地管理图书、浏览他人的评价以及分享自己的阅读经验。

What business processes needed

用户注册和登录将是业务流程中的重要一环，用户需要能够注册账户并登录以使用应用程序的功能。另外，图书搜索、发布图书、评价图书等也是重要的业务流程。

Consider integration with existing services

后续可以接入第三方API，例如社交媒体分享、身份验证服务。这些服务可以增强用户体验并提供额外的功能。

What profile information is required? Where will it be stored?

用户可能需要提供基本的个人信息（例如用户名、密码、用户昵称等）。这些信息存储在mongodb数据库中。

Development time and cost

开发用了7天。

Will the proposed app meet the brief?

可以实现用户注册、按照不同类型书籍进行搜索、发布图书、评价图书等。

1. 高级功能

计划后续添加的功能：

个性化推荐： 根据用户的兴趣和评价向用户推荐图书。

多语言支持： 提供多语言界面，以满足国际用户的需求。

1. 数据保护

用户可见的个人数据：

管理员可以查看自己发布的书籍，并可以对其进行编辑、删除操作

数据安全：

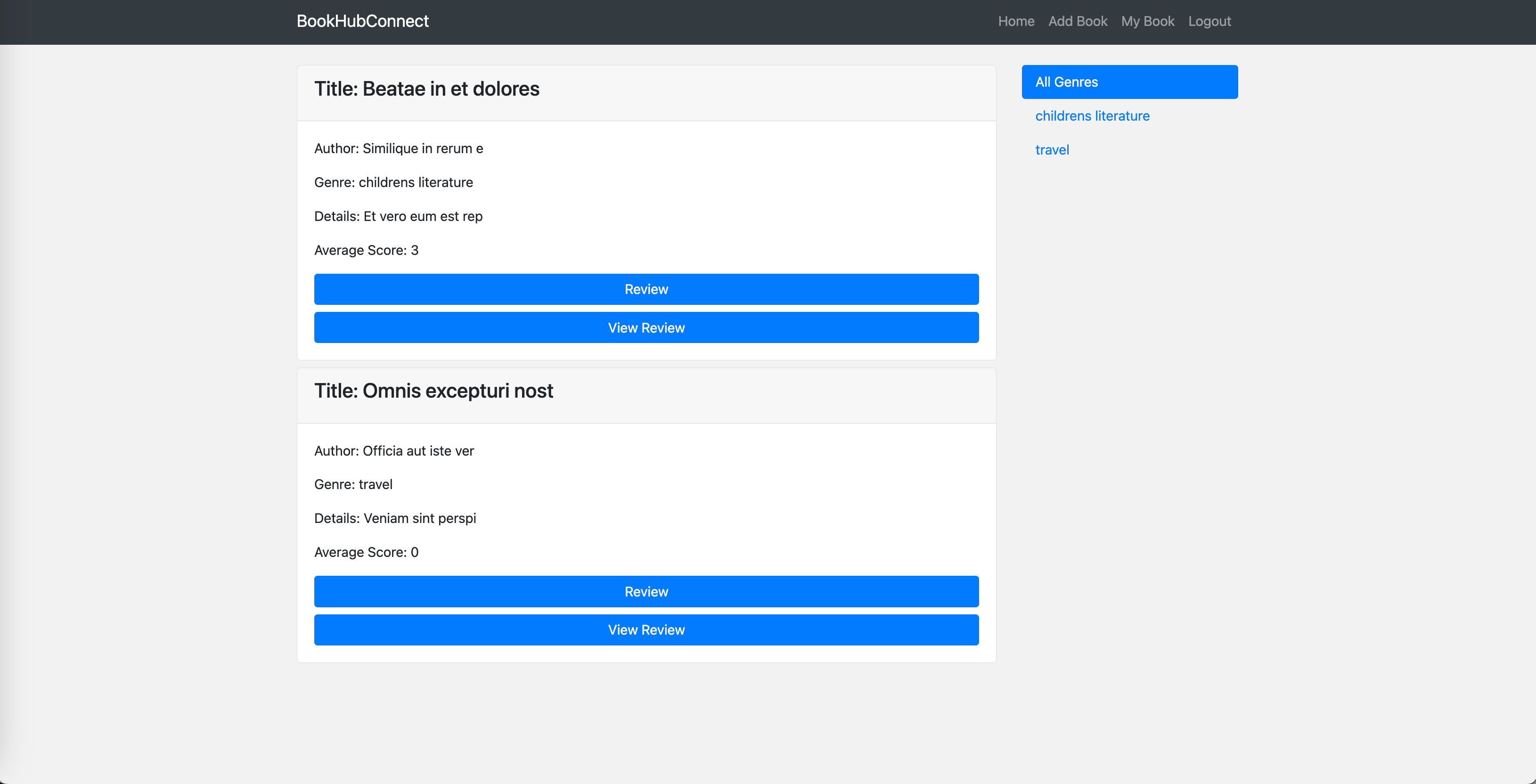
用户注册的数据，对密码进行md5加密

1. 实现

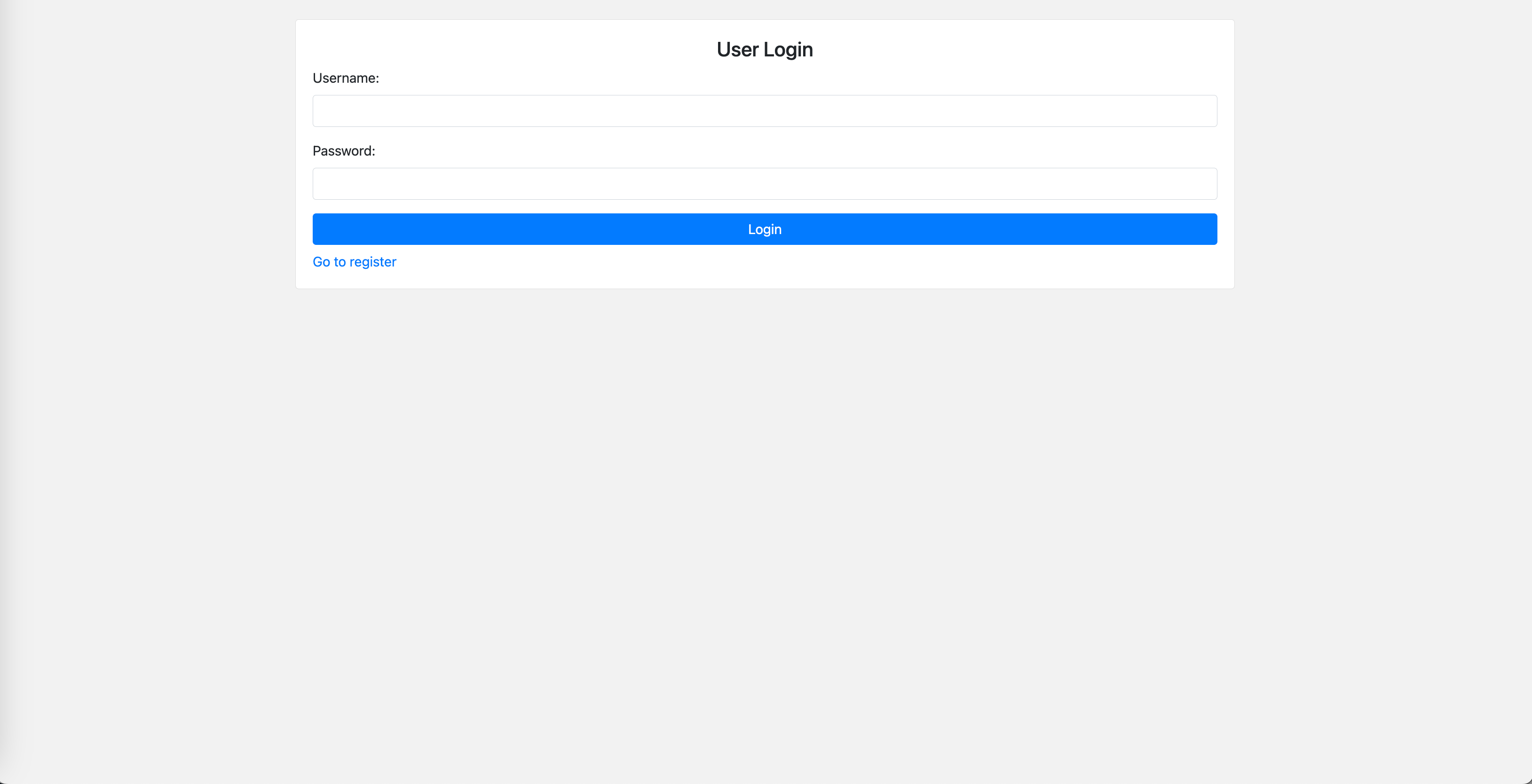
网站布局描述：

顶部为导航栏，左边为网站名称BookHubConnect,右边为菜单栏，菜单栏在未登录状态显示Home、Login、Register选项，登录后用户显示为Home、Logout选项，管理员显示为Add Book、My Book、Logout，首页左边为图书列表，右边为分类列表，图书项中有查看评论按钮和添加评论按钮，添加评论按钮需要在登录后才会显示

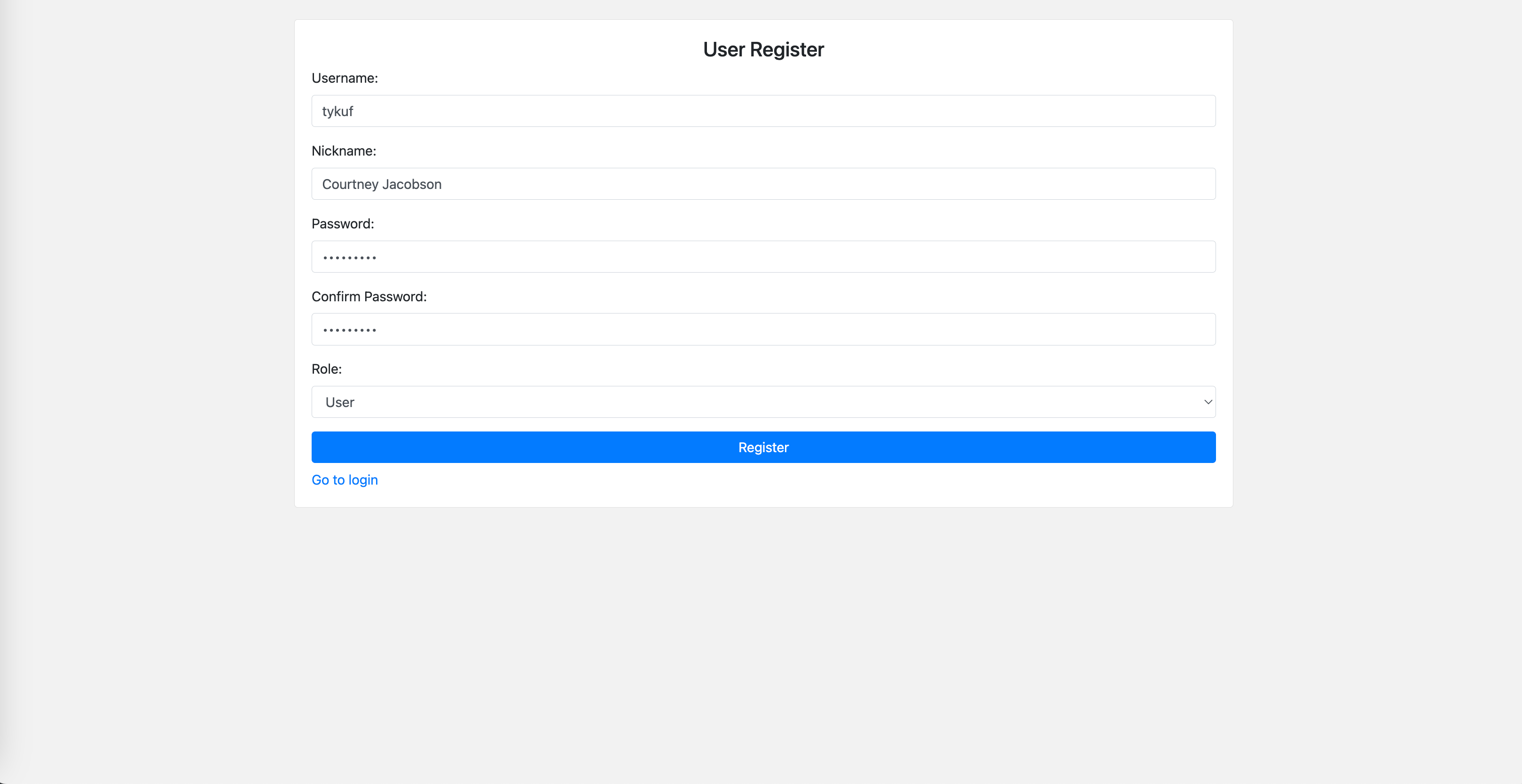
首页效果图：



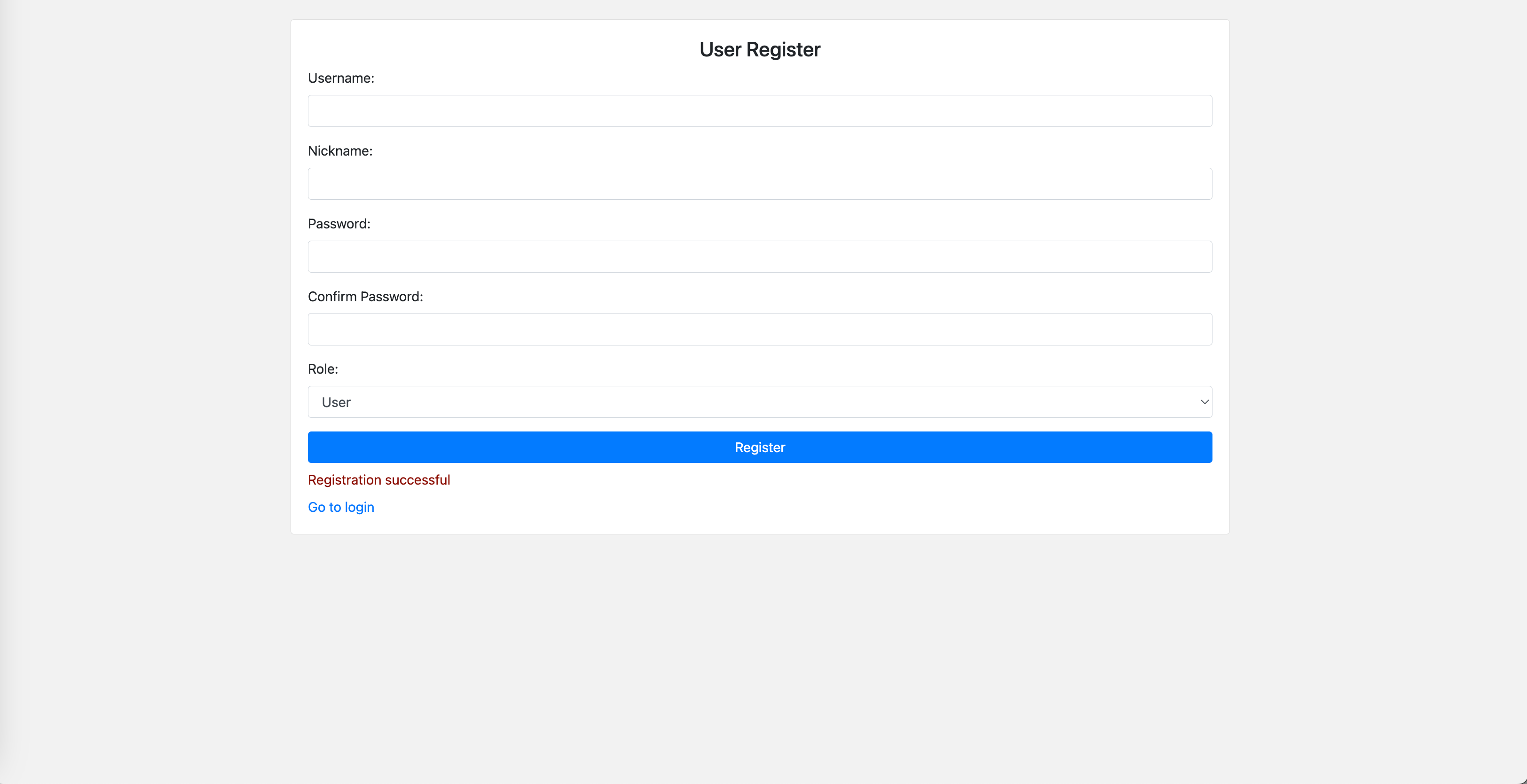
登录效果图：



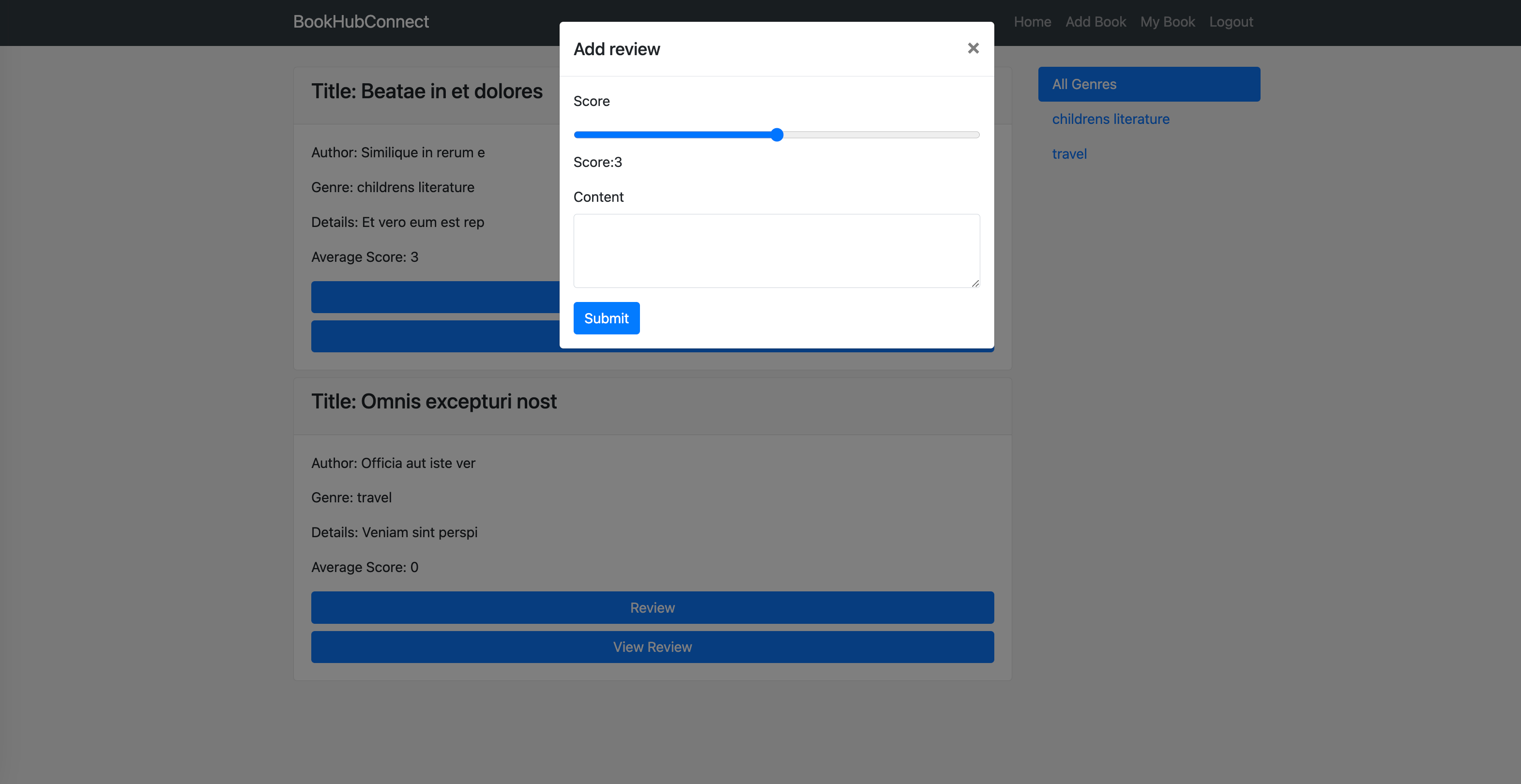
用户注册：



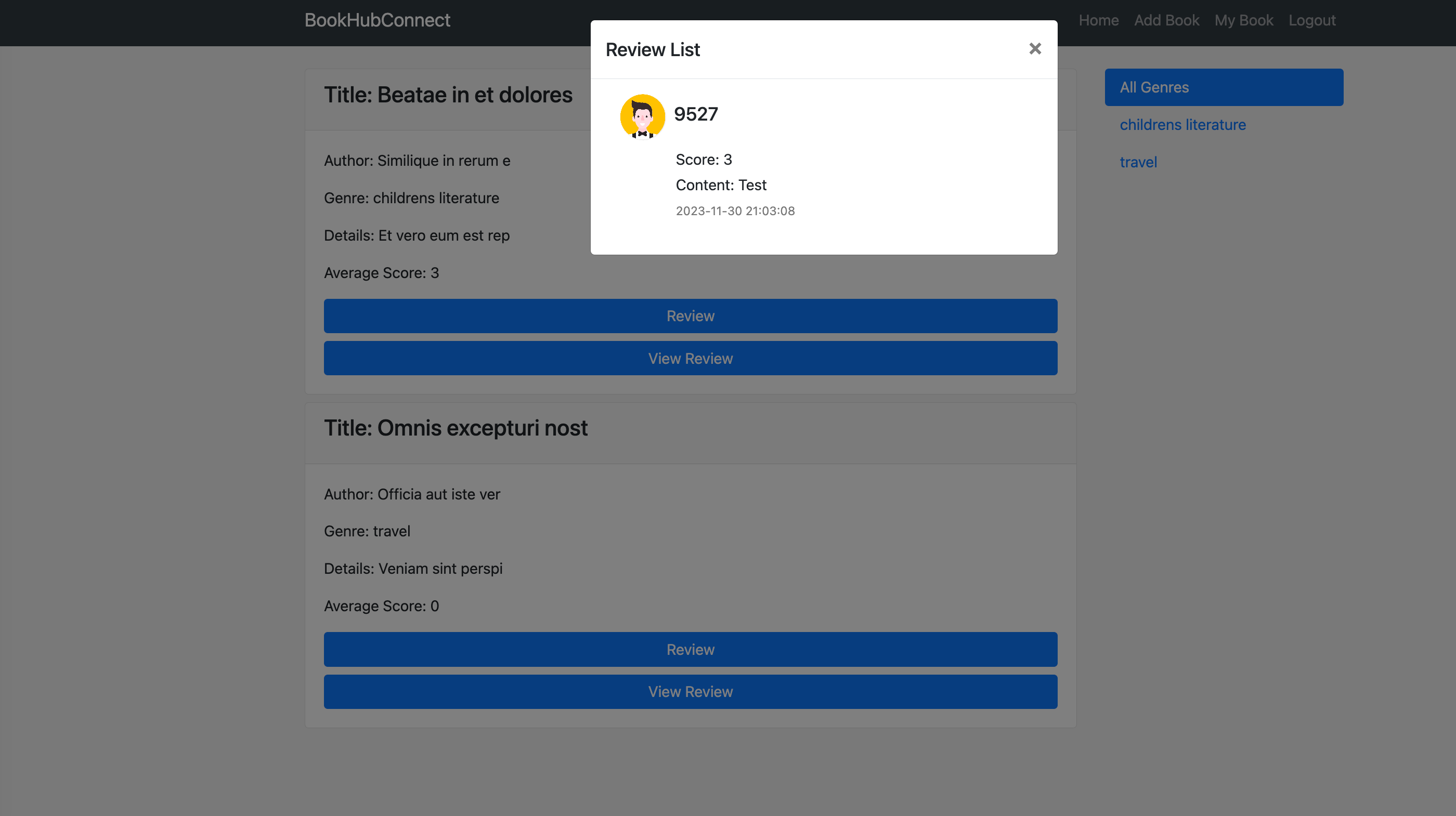
注册成功：



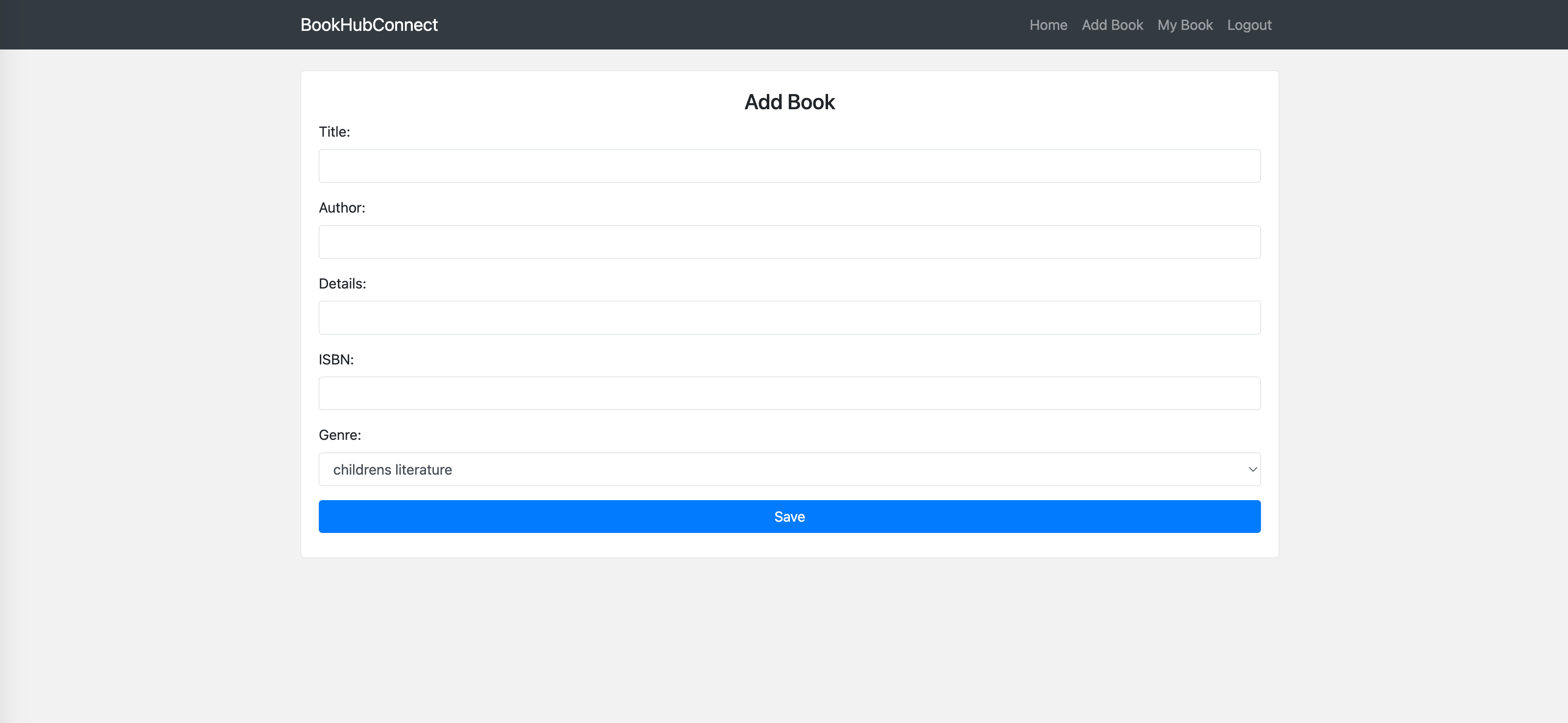
添加评论：



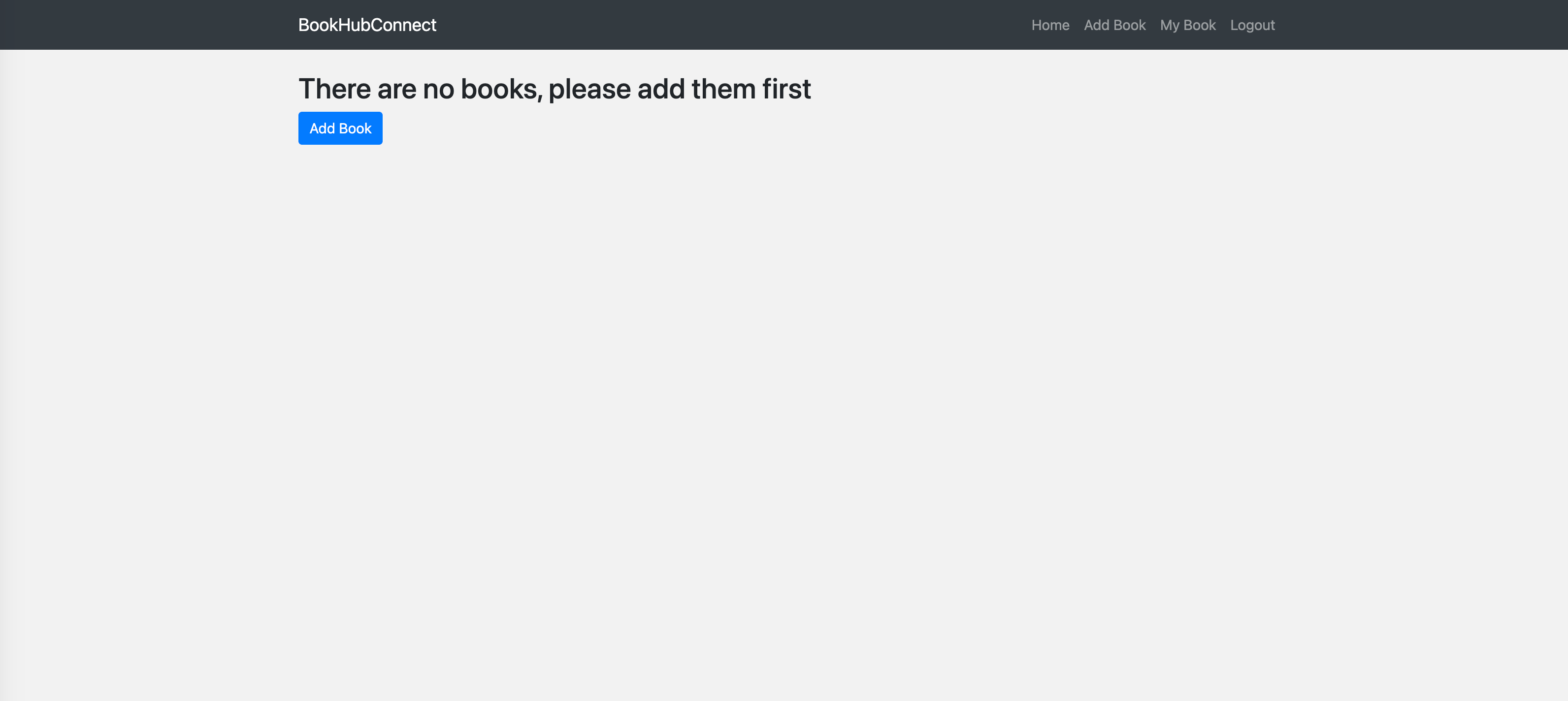
查看评论：

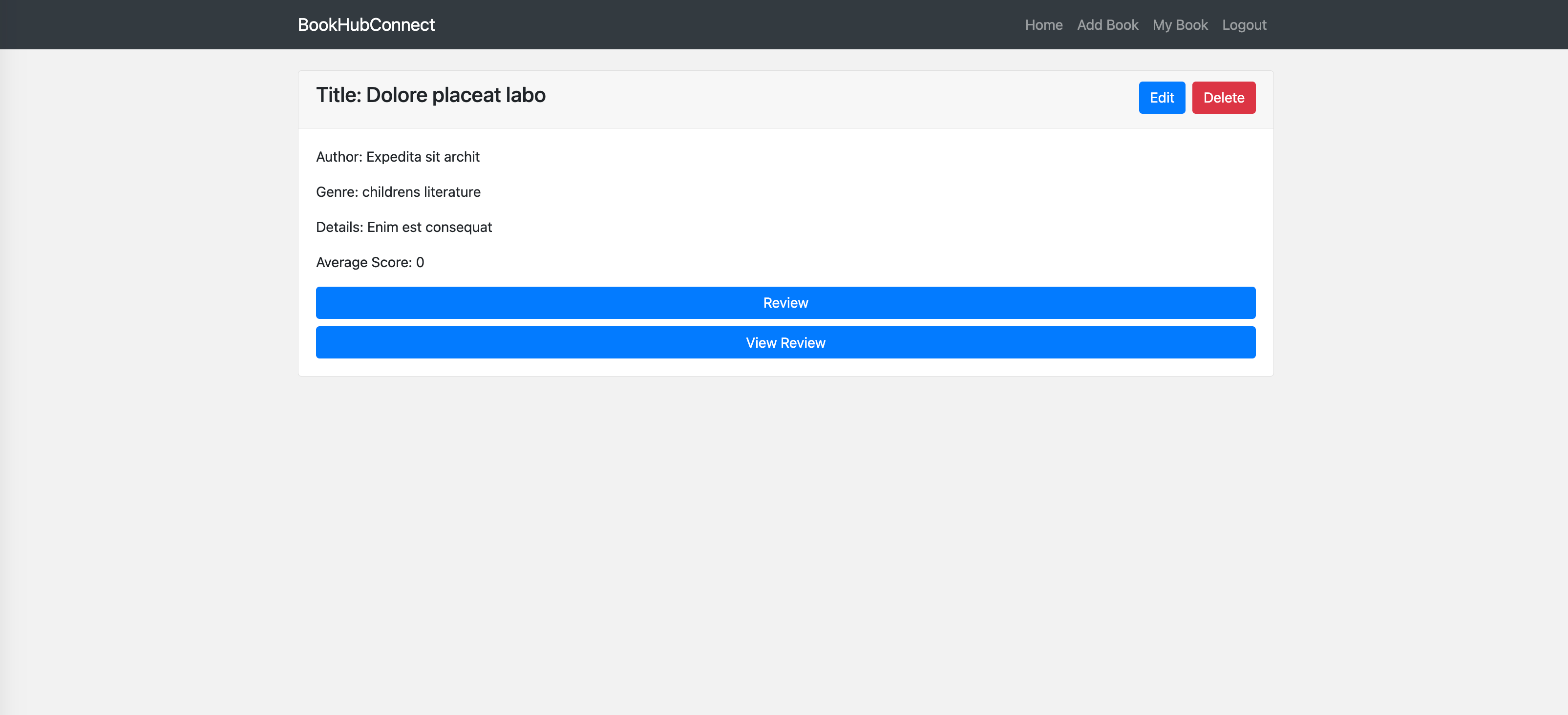


添加图书：



我的图书：





1. 技术堆栈

Frontend：

Pug模板引擎： 用于在服务器端生成HTML页面的模板引擎，简化了HTML的创建过程。

Bootstrap 4： 前端UI框架，提供响应式设计、组件和样式，用于创建现代化的用户界面。

Backend：

Express.js： 轻量级的Node.js Web应用框架，用于构建可靠和高性能的后端服务。

MongoDB： 非关系型数据库，用于存储和管理应用程序的数据。

其他工具和技术：

Mongoose： 在Node.js环境中与MongoDB数据库进行对象模型化的库。

1. 数据组织

Collections:

users：包含用户的个人信息。

books：存储书籍的详细信息。

reviews：记录用户对图书的评价和评分等信息。

genres：存储书籍的类别信息，name。

结构：

users: username、password、nickname、role。

books: name、author、details、isbn、user、genre。

reviews: book、user、score、content、createTime。

genres: name

集合之间的关系：

users和reviews之间是一对多的关系，一个用户可以对多本书进行评价。

books和reviews之间是一对多的关系，一本书可以有多个评价。

books和genres之间是多对一的关系，一个分类可以有多本图书

1. API设计

GET /user/login 查看登录页面

POST /user/login 登录操作

GET /user/register 查看注册页面

POST /user/register 注册操作

GET /user/logout 退出登录

POST /review/add 评论操作

GET /review/list 查看评论列表

GET /book/list 图书列表

GET /book/myself 查看我发布的图书

GET /book/add 查看添加图书页面

POST /book/add 添加图书操作

GET /book/remove/:id 删除图书操作

GET /book/update/:id 编辑图书页面  
 POST /book/update/:id 编辑图书操作

查询和UI表单：

书本分类查询可以通过右侧分类菜单栏

设计过程支持的模拟服务器：

没有使用mock数据，测试的时候使用Fake Filler浏览器插件快速填充表单

1. 托管

单服务器: 使用Apache, Nginx, IIS单服务器，这种方式是成本比较低的一种，不过有较大的风险，如果出现故障，可能会导致整个服务停止运行

多服务器： 通过设置多个服务器来提高应用程序的稳定性和可靠性。可以通过负载均衡来分发流量，但维护成本较高。

云服务的优势：

可扩展性和弹性： 云服务允许根据需求快速扩展和收缩资源。

高可用性： 云提供商通常提供具有高可用性的基础设施，降低了应用中断的风险。

云服务比较：

AWS (Amazon Web Services)： 提供广泛的云服务，包括计算、存储、数据库等。按需付费，可根据需求选择各种服务。

Azure (Microsoft Azure)： 微软的云服务平台，提供与AWS类似的服务。与其他微软产品集成良好。

Google Cloud Platform (GCP)： Google提供的云服务平台，具有高度的可扩展性和性能。

1. 扩展性

可能遇到的问题：

用户数量： 随着用户数量的增加，系统需要处理更多的请求。如果系统架构不具备水 平扩展性，可能会导致性能下降和响应时间延迟。

数据量： 随着数据量的增加，数据库查询和处理可能变得更加复杂和耗时。如果数据 库设计不合理或者没有合适的索引，可能导致性能瓶颈。

解决方案：

水平扩展： 通过增加更多的服务器实例来分担负载，可以采用负载均衡技术将流量分 发到不同的服务器上。

1. 跟踪设计

可以进行跟踪网站性能数据，例如页面加载时间、服务器响应时间等，用于优化性能

不能跟踪用户在网站上访问的个人隐私信息等

1. 结论

使用了bootstrap框架和pug模版引擎，大大的增加开发效率，同时bootstrap自带的样式让网页看起来更美观