ElanWave Bookstore – task for applicants

# Introduction

This is a simple ASP.NET task whose purpose is to check an applicant’s knowledge of backend using .NET Core and frontend using one of the following Blazor/Angular/React. It is also a nice introductory task for people who have recently started learning C# and ASP.NET.

To successfully do the task, an applicant must score at least 60% of the points. Different parts of task are worth different amount of points. Additional tasks at the end can bring extra points. In the grading process, we will grade both functionality and code quality.

# Task

The task is very simple – make an ASP.NET Core application for administration of books in a bookstore. User must log in (or sign up) before using the application. Once the user is logged in, he/she can browse the existing books in the bookstore, edit or delete them and add new ones. After using the application the user can log out.

Mockups of all pages are supplied in this document. Please follow the ‘design’ as closely as possible. It doesn’t have to be pixel-perfect, just to look similar.

For connection with the database you can use framework of your choice (Entity framework and such) or simple connections and transactions.

## Requirements

1. Visual Studio 2019
2. SQLite database or similar – we need a server-less db solution, so that the application is easily transported and run
3. Blazor/React/Angular on frontend

***IMPORTANT: Please note that reviewers will not look into tasks which cannot be unzipped and run in Visual Studio 2019 without additional software and setup.***

## Home page

Maximum points: 15(page layout – 5 points; login/logout functionality – 10 points)

Home page has very little functionality – the user can just log in or go to sign up page. The text in the middle is a static message. The footer contains static copyright text and a link to ElanWave site.

The log in will try to find non deleted (more info on this in the following section) user accounts with specified username and password in the database. After a successful login, the user data should be put in a session variable, and the user is redirected to the Books page.

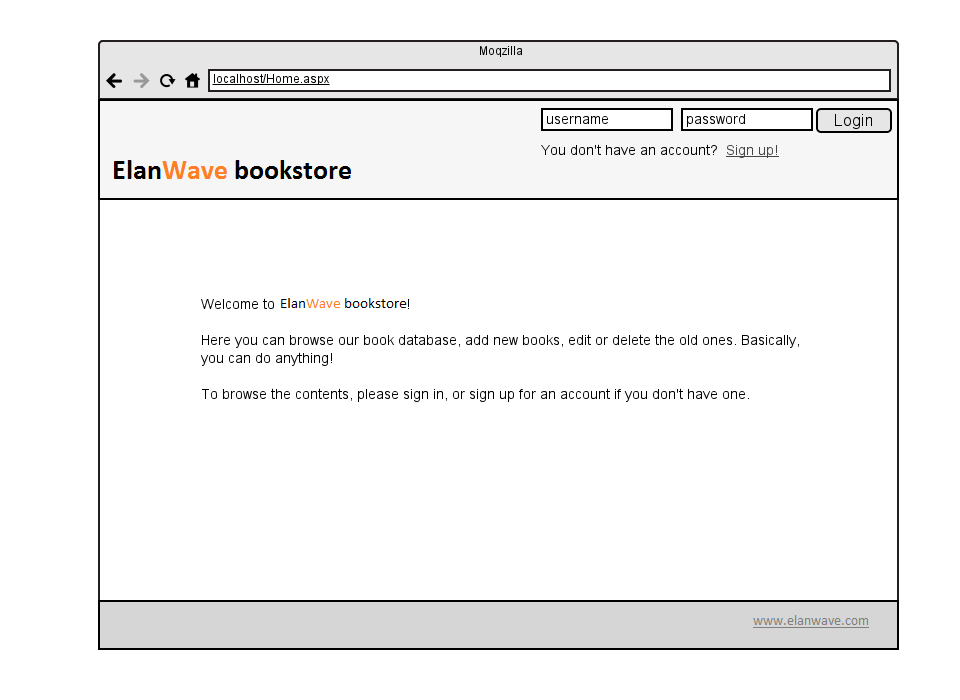


Figure 1 - Home page

## Sign Up page

Maximum points: 25 (page layout – 5 points; data persisting – 10 points; data validation – 10 points)

User gets to this page by clicking on the Sign Up link in the header. Sign up page should look as follows:

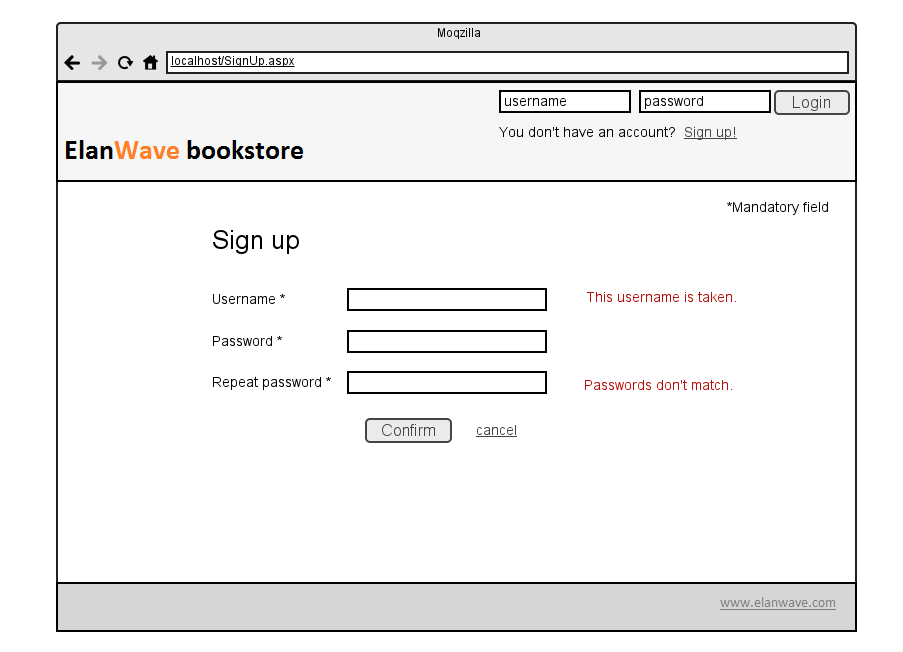


Figure 2 - Sign up page

The page is used for making new user accounts. A user account in the database have the following structure:

|  |  |
| --- | --- |
| UserAccount | |
| UserAccountID | The value is [Guid](http://en.wikipedia.org/wiki/Globally_unique_identifier). You can store it as char(36) or binary(16). |
| Username | varchar(45) |
| Password | varchar(45). The password can be stored as plain text or hash. The choice is yours. |
| IsDeleted | tinyint/boolean . We will be using soft deletion, so no entries will be permanently deleted from the database. |

The UserAccountID is assigned programmatically, the user only enters desired username and password, and repeat password. See the data validation section.

After successful registration, the user should be automatically logged in and redirected to the Books page.

### Data validation

**Username** – must be entered (client side) and must be unique (server side using postback) amongst non-deleted records in the database. These checks are performed on focus lost and on save button click.

**Password** – must be entered (client side). This check is performed on focus lost and on save button click.

**Repeated password** – must be entered and must be the same as password (client side). This check is performed on focus lost and on save button click.

The content of error messages is up to you!

## Books page

Maximum points 25 (page layout – 10 points, data retrieval – 10 points, deletion – 5 points)

Books page is an overview of all non-deleted book records in the database. It should look as follows:

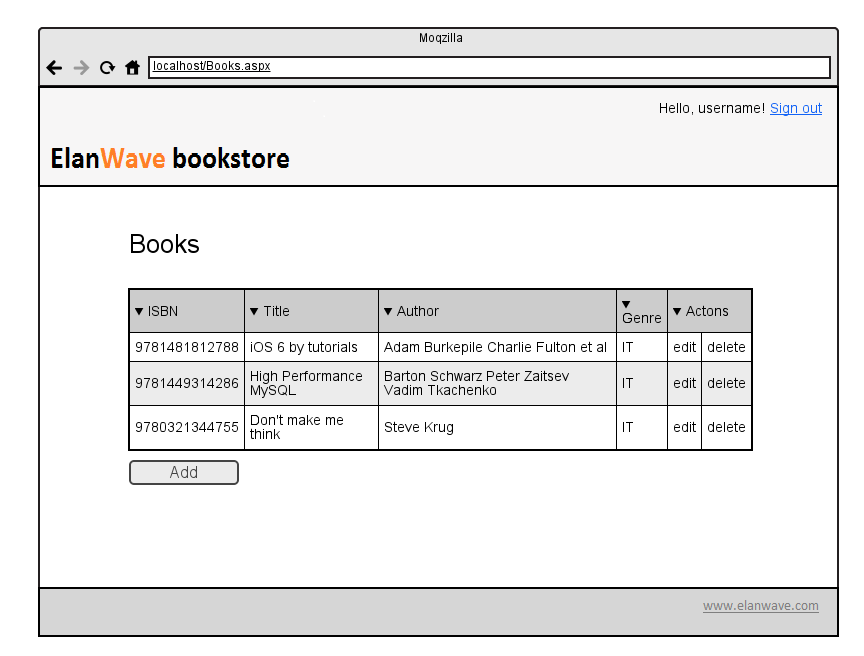


Figure 3 - Books page

On this page a user can view all the (non-deleted) books in the database, initiate editing and deletion of the books or add new ones. This page should NOT be accessible without being signed in.

Clicking the edit link in a table row redirects the user to the Add / edit book page and prefills all the fields there.

Clicking the Add button below the table redirects the user to the Add / edit book page.

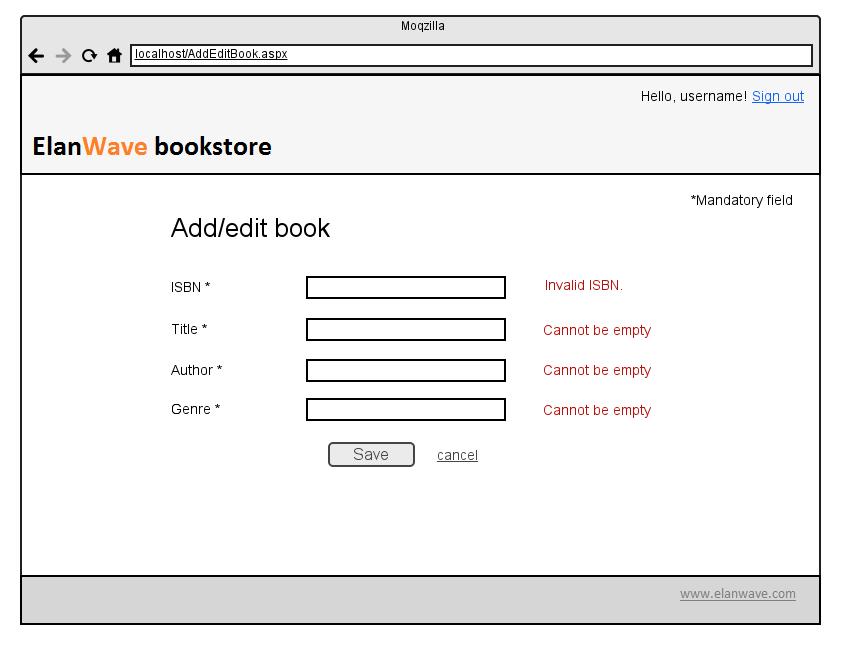
Clicking the delete link in a table row prompts the user to confirm his/her action before deleting the book record. Deletion is soft, no records are physically deleted from the database, just sets the IsDeleted field to true.

Tip: Use list view to display the books and to edit/delete them. (you can use existing patterns when creating view)

## Add/edit book page

Maximum points: 35 (page layout – 5 points; data validation – 10 points; data persistence – 10 points, data retrieval – 10 points)

Add / edit book page should look as follows:



The header content depends on the state of the page. If adding a new book record is in progress, the header should say “Add new book”. If editing of an existing book is in progress, the header should say “Edit <book title>”. This page should NOT be accessible without being signed in.

A book record in the database should be as follows:

|  |  |
| --- | --- |
| Book | |
| BookID | The value is [Guid](http://en.wikipedia.org/wiki/Globally_unique_identifier). You can store it as char(36) or binary(16). |
| ISBN | char(13). We will be using ISBN 13. Some info [here](http://en.wikipedia.org/wiki/International_Standard_Book_Number). |
| Title | varchar(256) |
| Author | varchar(256). If there are multiple authors they are all entered here as string |
| Genre | varchar(128) |
| IsDeleted | tinyint/boolean . We will be using soft deletion, so no entries will be permanently deleted from the database. |

Successful adding / editing redirects user back to the Books page. Cancel also redirects the user back to the book page, without any database updates.

### Data validation

ISBN – must be entered and must be valid ISBN number (client side). This check is performed on focus lost and on save button click.

**Title** – must be entered (client side). This check is performed on focus lost and on save button click.

**Author** – must be entered (client side). This check is performed on focus lost and on save button click.

**Genre** – must be entered and must be the same as password (client side). This check is performed on focus lost and on save button click.

The content of error messages is up to you!

# Additional tasks

Maximum points: 10

1. **Pager in table** – Enable paging in the Books table. *[2 point]*
2. **Sorting** – Enable sorting in the Books table. *[1 point]*
3. **Filter** – Add a filter function in the Books page which will display only records which match the filter query by ISBN, title, author or genre. *[2 points]*
4. **Detailed sign up checks** – check if the username is compliant to common rules (alphanumeric, hyphen, etc.) and if password is strong enough. *[2 points]*
5. **Server side checks** – add all data validation checks on server side too, before saving is performed. *[1 point]*
6. **Modal dialogs for Add/Edit books –** create simple modal dialog which will contain fields like on Add/Edit page and using that modal save new or edit existing book record (this functionality will take place instead of regular Add/edit page). *[2 points]*