Exam – SoftUni Library

You need to create a **web application** where users **see and manage list of books**. Usually the users will visit the page, navigate either to **signing up or sign in**. When they are logged, they will have the option to view **all books** listed. They can either **add** a new book, **view all their** books, or **edit** or **delete** existing, only if **they** added it.

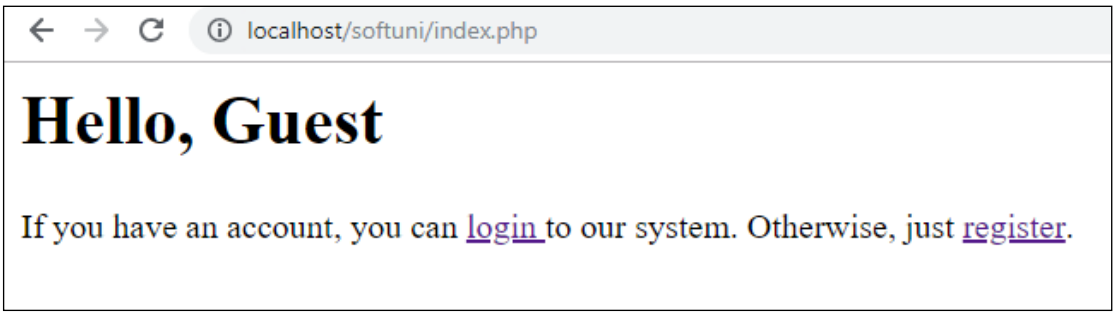
1.Expected Result

General rules:

1. **No blank lines** should be left
2. Going **on new line** (click ENTER) in the same text should be done, **only after the sentence ends** (after full stop)
3. Each **title** is **Heading 1**
4. All **attribute values and tags** are **lowercase**
5. All name fields are **`lower_snake_case`** variant of the text/value
6. All **tables** have **head** and if they are filled with data, they have a **body**
7. Only **direct child tags** go on **new line** e.g. **`<tr>` is** on new line after **`<thead>`,** but **`<a>`** in **`<li>` is not** on new line.
8. All **success messages** are stored in the **session** under the "**`success`**" key and are **immediately deleted** from session after visualization. They are visualized on the **next line after the title** in a **paragraph** with style **`color:green`**.
9. All **error messages** are visualized on the **next line after the title** in a **paragraph** with style **`color=red`**.
10. All **relational dropdowns** follow the convention: **`name_snake_case_id`**

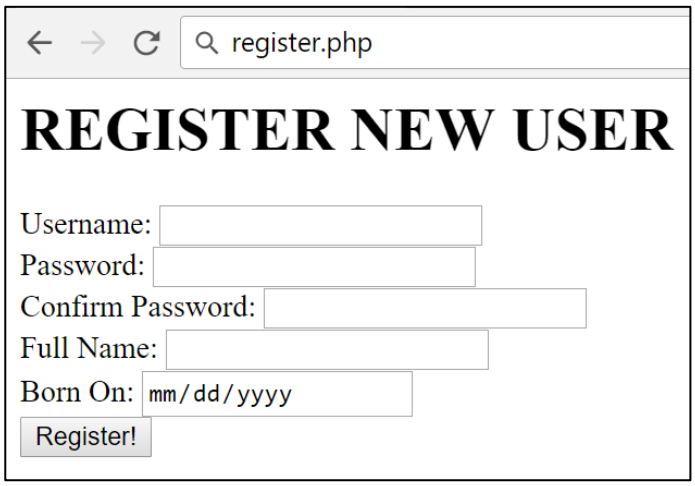
`index.php:`

- When user **is not** logged

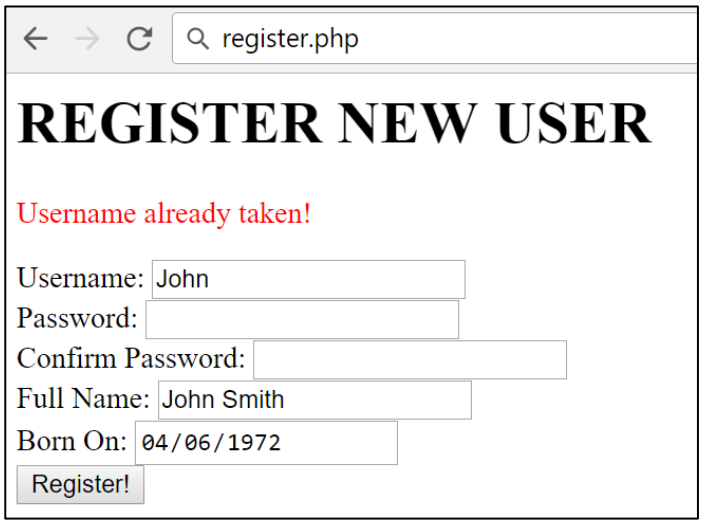


`register.php`:

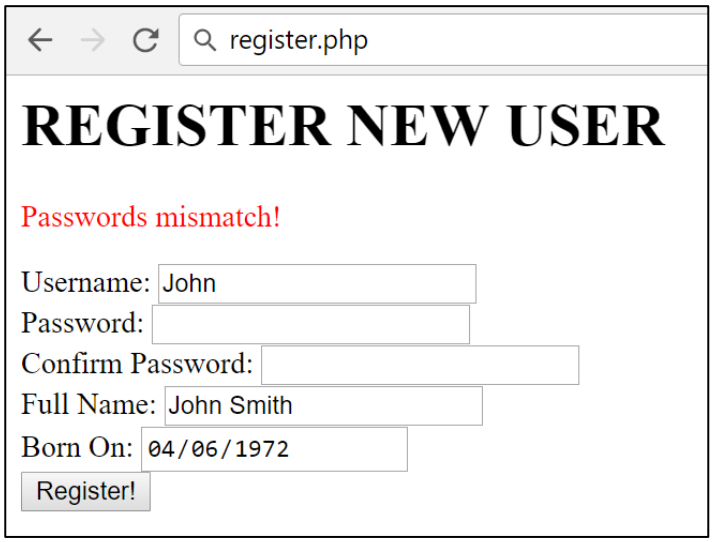
- **Before** form is submitted



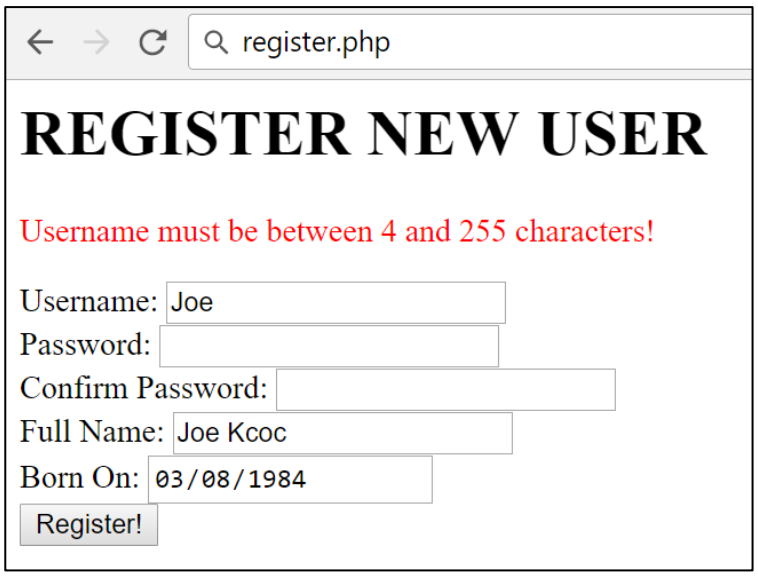
- Form **is submitted**, but there are **username duplicates**



- Form **is submitted**, but the **password** and the **confirm password** field differs:



- Form **is submitted**, but **validation** did not pass:



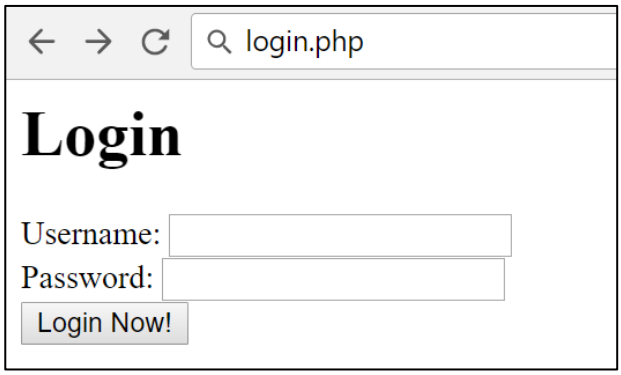
Show the **error only for the first field the validation did not pass** (they might all be out of range). The message is in the format:

- For **text** fields: **"`{Field Name} must be between {MIN_LENGTH} and {MAX_LENGTH} characters!`"**.
- For **number** fields: **"`{Field Name} must be between {MIN_VALUE} and {MAX_VALUE}!`"**

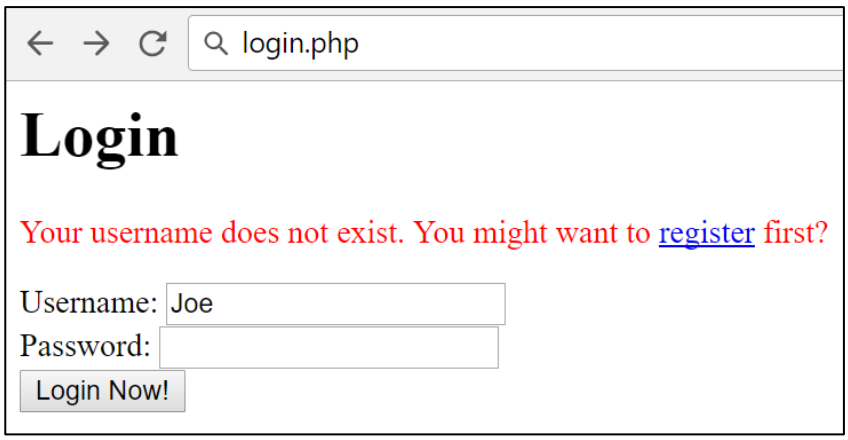
- **After** successful submission

`login.php`:

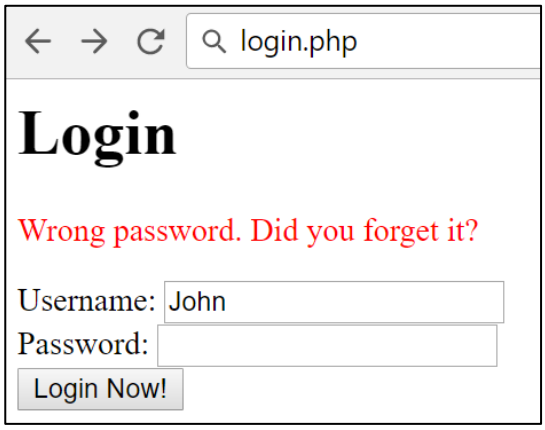
- **Before** form is submitted



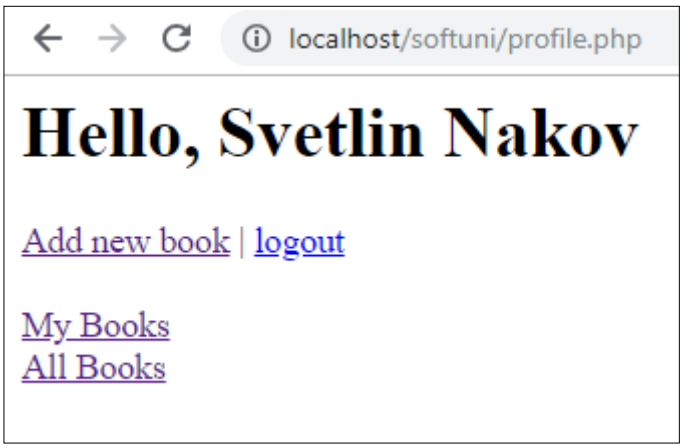
- Form **is submitted**, but **username** does not exist:



- Form **is submitted**, but the **password** is not the same:



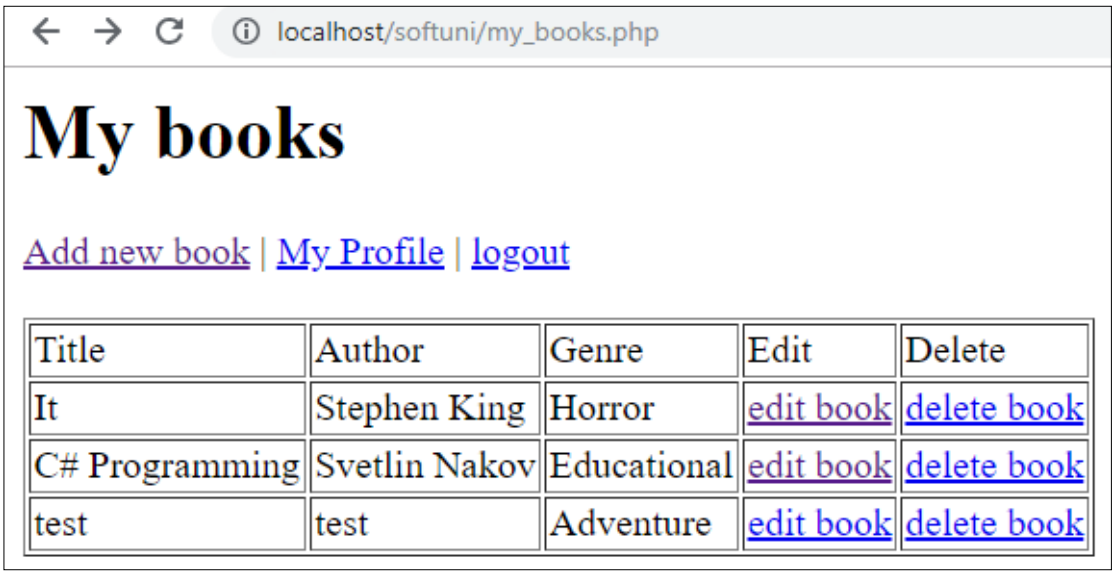
- **After** successful login:



You should get a greeting message with your first and your last name. You must have 4 links for **`adding new book`**, for **logout** for viewing **`your books`** and for viewing **`all books`**

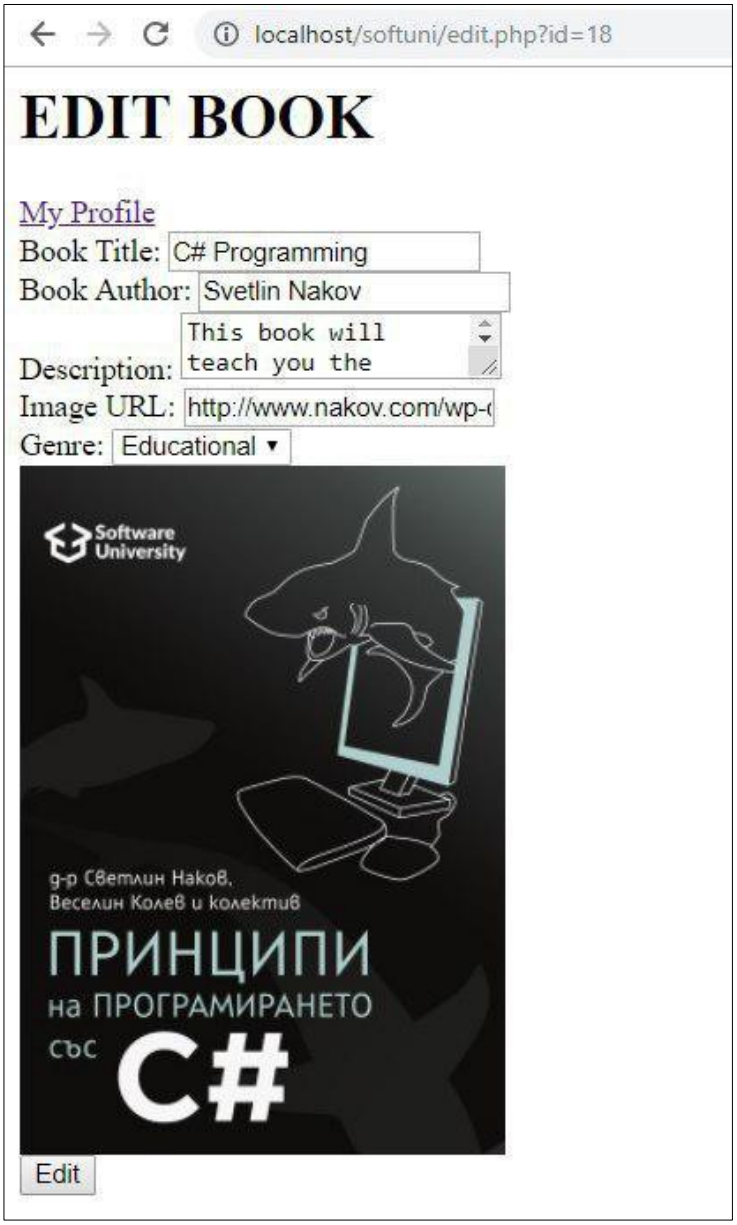
`add_book.php should look like this:`

my_books.php :



all_books.php:

`edit.php?id={book id}` (Edit existing book):

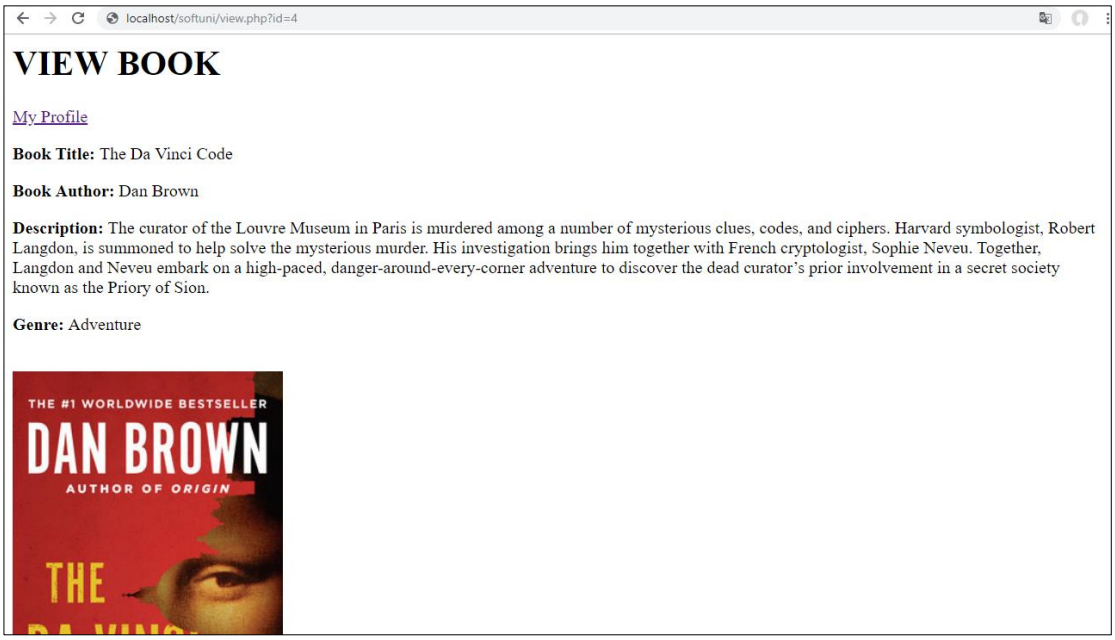


- After **successful submission**, the operation is **edited**, **redirect** to the list of my books **my_books.php**

`delete.php?id={book_id}` (Delete existing operation):

- **Deletes** the operation and **redirects** to of my books **my_books.php**.

`view.php?id={book_id}` When you click "details" from **`all_books.php`** you should be able to see the image and description about the books you selected. This option is available only for **`logged in`** users.



2.Database

Create a database "softuni_library" and add the following tables:

Data Definition

DDL – Users

users		
Column Name	**Type**	**Constraints**
username	String	4...255. Unique.
password	String	4...255
full_name	String	5...255
born_on	date	not null

DDL – Genres

genres		
Column Name	**Type**	**Constraints**

name	String	3…50

DDL – Books

books		
Column Name	**Type**	**Constraints**
title	String	3…50
author	string	3…50
description	string	10…10000
image	string	3…255
genre_id	int	1…4294967295
user_id	int	1…4294967295
added_on	TimeStamp	Current timestamp

DML – Add Genres

Adds the following **`genres`**: Save the DDL in **`export.sql`** file in the top-level directory of your project.

name
Drama
Educational
Adventure

HTML

Register Form

Create register form. Follow the constraints below:

1. **`Username`** with minimum length of **4** symbols
2. **`Password`** with minimum length of **4** symbols
3. **`Confirm password`** with minimum length of **4** symbols
4. **`Full Name`** with minimum length of **5** symbols
5. **`Born On`** with type date
6. Button, submitting the form with value "**Register**"

All fields are **required**. Use a convention for name attributes: **lower_snake_case**.

Login Page

1. **`Username`**
2. **`Password`**
3. Button, submitting the form with text "**Login**"

All fields are **required**. Use a convention for name attributes: **lower_snake_case**.

New Book Page

1. **`Title`** with minimum length of **3** symbols and max length – 50
2. **`Author`** with minimum length of **3** symbols and max length – 50
3. **`Description`** containing **10 to 10,000** symbols
4. **`Image string containing`** url
5. **`Genres`** which is **empty dropdown** (will be later populated with PHP)
6. Button, submitting the form with text "**Add**"

All fields are **required**. Use a convention for name attributes: **lower_snake_case**.

Structure

You are required to have **`index.php`** in the project, even if it is initially empty. Consider all **object oriented best practices** you have been taught during the course.

Initial File Structure

1. Create a file named **"`export.sql`"** holding CREATE DATABASE and all the CREATE TABLE statements. Put it in the top-level directory;
2. Create a directory named "**`Config`**" in the top-level directory. Put file named "**`db.ini`**" holding the database connection information under the following keys**`: dsn, user, pass`**;
3. Create endpoints (front layer) for all possible pages in the top-level directory: **`index, register, login, add_book, edit, all_books, my_books, view`** and **`logout`**;
4. Extract common operation such as autoloading classes in **"`common.php`".**

Database Abstraction Layer

Create an **abstraction layer over the database** operations in three stages – **prepare -> execute -> take result**. All objects should be in the **`\Database`** namespace.

Repository Layer

Create repository layer in the namespace **`\App\Repository`**

Service Layer

Create service layer in the namespace **`\App\Service`**

Book Management Application

Make **all the layers to work**, according to the structure and the business requirement in **1.** and submit the **zip archive** in judge.