

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 **** 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

The screenshot shows a web browser window with the URL "localhost/softuni/index.php" in the address bar. The page content is displayed in a white box. At the top, it says "Hello, Guest". Below that, there is a paragraph of text: "If you have an account, you can [login](#) to our system. Otherwise, just [register](#)".

register.php:

- Before form is submitted

The screenshot shows a web browser window with the URL 'register.php' in the address bar. The main content is a form titled 'REGISTER NEW USER'. It contains five input fields: 'Username' (empty), 'Password' (empty), 'Confirm Password' (empty), 'Full Name' (empty), and 'Born On' (empty). Below the form is a 'Register!' button.

Username:
Password:
Confirm Password:
Full Name:
Born On: mm/dd/yyyy
Register!

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

The screenshot shows a web browser window with the URL 'register.php' in the address bar. The main content displays the error message 'Username already taken!' above the registration form. The form fields are identical to the first one: 'Username' (John), 'Password' (empty), 'Confirm Password' (empty), 'Full Name' (John Smith), and 'Born On' (04/06/1972). Below the form is a 'Register!' button.

Username: John
Password:
Confirm Password:
Full Name: John Smith
Born On: 04/06/1972
Register!

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

The screenshot shows a web browser window with the URL 'register.php' in the address bar. The main content displays the error message 'Passwords mismatch!' above the registration form. The form fields are identical to the previous ones: 'Username' (John), 'Password' (empty), 'Confirm Password' (empty), 'Full Name' (John Smith), and 'Born On' (04/06/1972). Below the form is a 'Register!' button.

Username: John
Password:
Confirm Password:
Full Name: John Smith
Born On: 04/06/1972
Register!

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

← → C ⌂ register.php

REGISTER NEW USER

Username must be between 4 and 255 characters!

Username:

Password:

Confirm Password:

Full Name:

Born On:

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

← → C ⌂ login.php

Login

Congratulations, John. Login in our platform

Username:

Password:

login.php:

- **Before** form is submitted

← → C ⌂ login.php

Login

Username:

Password:

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

← → ⌛ 🔍 login.php

Login

Your username does not exist. You might want to [register first?](#)

Username:

Password:

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

← → ⌛ 🔍 login.php

Login

Wrong password. Did you forget it?

Username:

Password:

- After successful login:

← → ⌛ ⓘ localhost/softuni/profile.php

Hello, Svetlin Nakov

[Add new book](#) | [logout](#)

[My Books](#)
[All Books](#)

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:

localhost/softuni/add_book.php

ADD NEW BOOK

[My Profile](#)

Book Title:

Book Author:

Description:

Image URL:

Genre: Horror ▾

Add Horror
Educational
Adventure

my_books.php :

localhost/softuni/my_books.php

My books

[Add new book](#) | [My Profile](#) | [logout](#)

Title	Author	Genre	Edit	Delete
It	Stephen King	Horror	edit book	delete book
C# Programming	Svetlin Nakov	Educational	edit book	delete book
test	test	Adventure	edit book	delete book

all_books.php:

All Books

[Add new book](#) | [My Profile](#) | [logout](#)

Title	Author	Genre	Added by User	Details
It	Stephen King	Horror	pesho	Details
The Da Vinci Code	Dan Brown	Horror	pesho	Details
C# Programming	Svetlin Nakov	Educational	pesho	Details
Frankenstein	Mary Shelley	Horror	nakov	Details
Don Quixote	Miguel de Cervantes	Adventure	nakov	Details

[edit.php?id={book id}](#) (Edit existing book):

← → ⌛ i localhost/softuni/edit.php?id=18

EDIT BOOK

[My Profile](#)

Book Title:

Book Author:
This book will
teach you the

Description:

Image URL:

Genre:



The image shows the front cover of a book titled 'ПРИНЦИПИ на ПРОГРАМИРАНЕТО със C#' (Principles of Programming with C#). The cover is dark with white text and features a stylized illustration of a hand pointing at a computer monitor. The monitor displays a small version of the book cover itself. The author's name, 'Светлин Наков', is also mentioned on the cover.

[Edit](#)

- 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.

The screenshot shows a web browser window with the URL `localhost/softuni/view.php?id=4`. The page title is "VIEW BOOK". Below it, there is a link "My Profile". The book details are as follows:

- Book Title:** The Da Vinci Code
- Book Author:** Dan Brown
- Description:** The curator of the Louvre Museum in Paris is murdered among a number of mysterious clues, codes, and ciphers. Harvard symbologist, Robert Langdon, is summoned to help solve the mysterious murder. His investigation brings him together with French cryptologist, Sophie Neveu. Together, Langdon and Neveu embark on a high-paced, danger-around-every-corner adventure to discover the dead curator's prior involvement in a secret society known as the Priory of Sion.
- Genre:** Adventure

A thumbnail image of the book cover for "The Da Vinci Code" by Dan Brown is displayed, showing the title and a portrait of a man's eye.

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.