

# Exam – SoftUni Bazar

You need to create a **web application** where users **see and trade items**. 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 items** listed. They can either **add** a new item, **view all their** items or **edit** or **delete** existing, only if **they** added it.

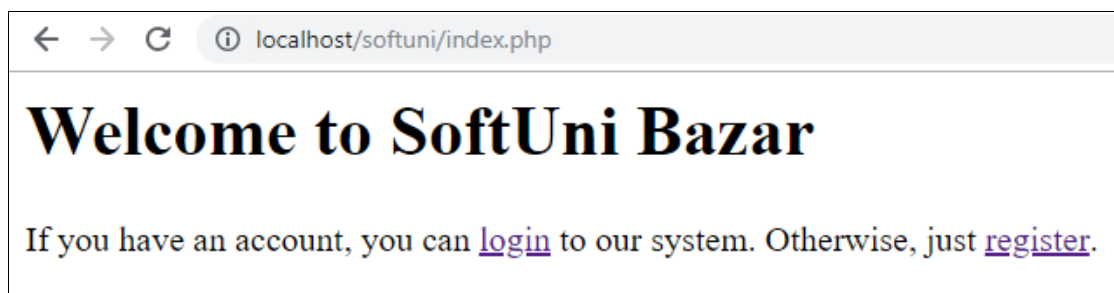
## 1. Expected Result

General rules:

1. Each **title** is **Heading 1**
2. All **attribute values and tags** are **lowercase**
3. All name fields are **lower\_snake\_case** variant of the text/value
4. All **tables** have **head** and if they are filled with data, they have a **body**
5. 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**.
6. All **error messages** are visualized on the **next line after the title** in a **paragraph** with style **color=red**.

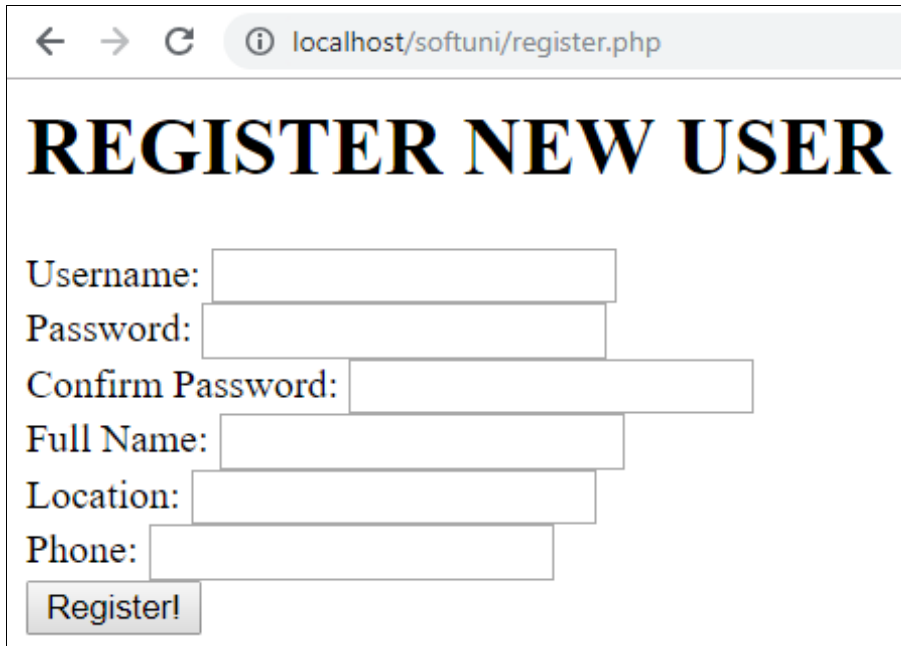
**index.php:**

- When user is **not** logged



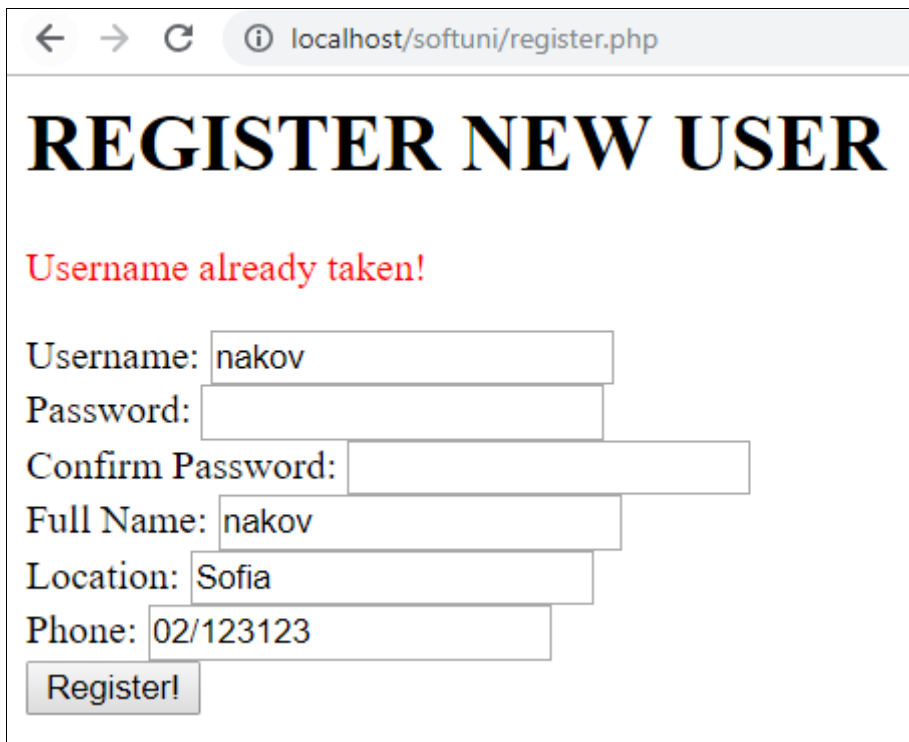
### register.php:

- Before form is submitted



A screenshot of a web browser showing the 'REGISTER NEW USER' form. The browser's address bar displays 'localhost/softuni/register.php'. The form title is 'REGISTER NEW USER' in a large, bold, black serif font. Below the title, there are six input fields: 'Username:', 'Password:', 'Confirm Password:', 'Full Name:', 'Location:', and 'Phone:'. Each field is followed by a text input box. At the bottom of the form is a 'Register!' button with a grey gradient.

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



A screenshot of the same 'REGISTER NEW USER' form after submission. The browser's address bar still shows 'localhost/softuni/register.php'. The form title remains 'REGISTER NEW USER'. A red error message, 'Username already taken!', is displayed above the input fields. The input fields now contain the following values: 'Username:' is 'nakov', 'Password:' is empty, 'Confirm Password:' is empty, 'Full Name:' is 'nakov', 'Location:' is 'Sofia', and 'Phone:' is '02/123123'. The 'Register!' button is still present at the bottom.

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

← → ↻ ⓘ localhost/softuni/register.php

# REGISTER NEW USER

Password mismatch!

Username:

Password:

Confirm Password:

Full Name:

Location:

Phone:

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

← → ↻ ⓘ localhost/softuni/register.php

# REGISTER NEW USER

Username must be between 4 and 255 characters!

Username:

Password:

Confirm Password:

Full Name:

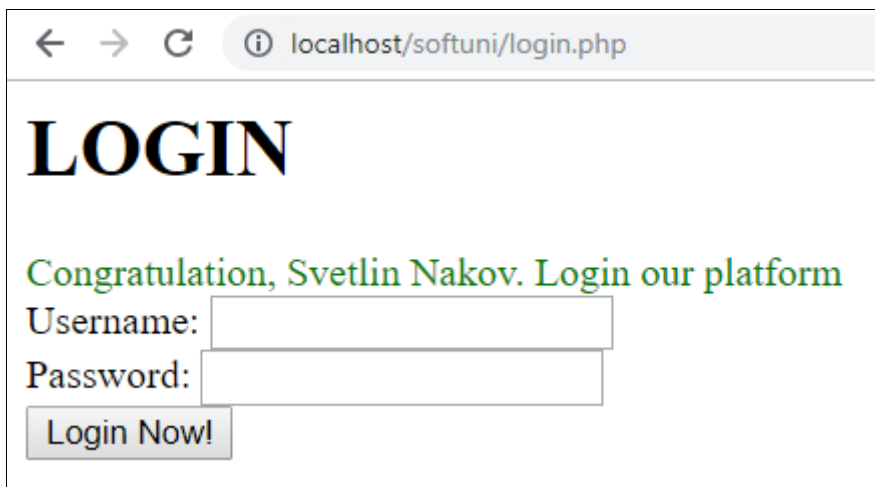
Location:

Phone:

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}!”

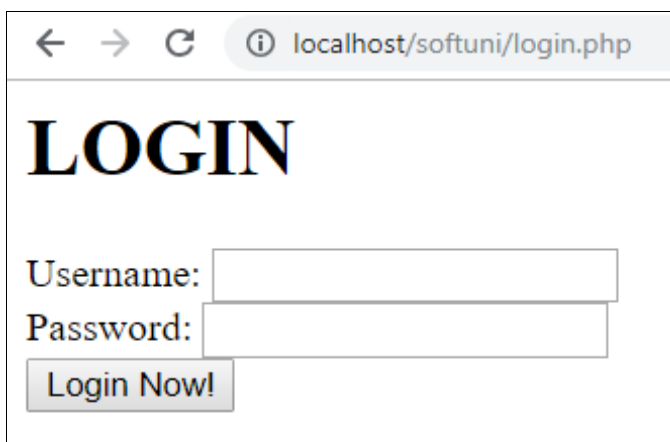
- **After** successful submission Congratulations, “Full Name”.



A screenshot of a web browser window. The address bar shows 'localhost/softuni/login.php'. The page has a large 'LOGIN' heading. Below it, a green message reads 'Congratulation, Svetlin Nakov. Login our platform'. There are two input fields: 'Username:' and 'Password:'. Below the password field is a 'Login Now!' button.

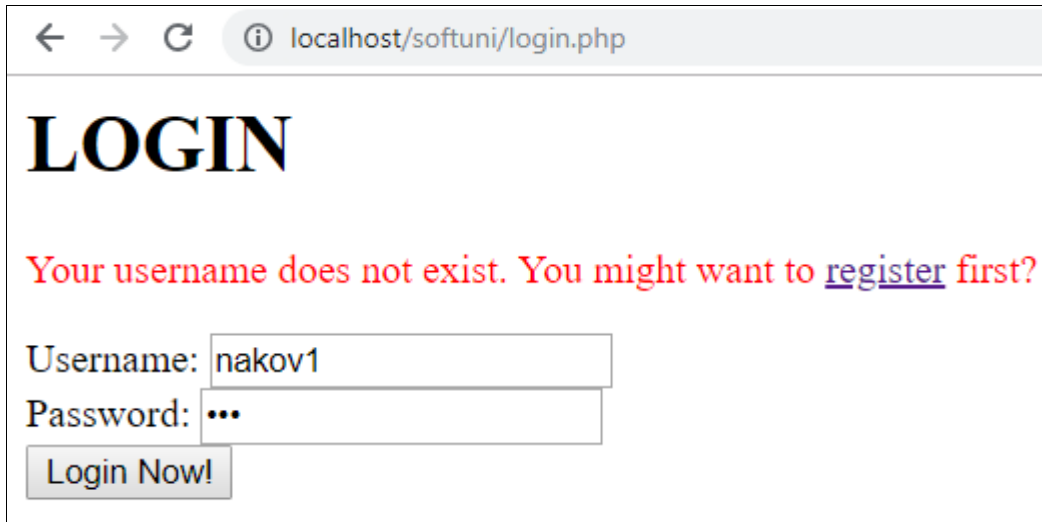
**login.php:**

- **Before** form is submitted



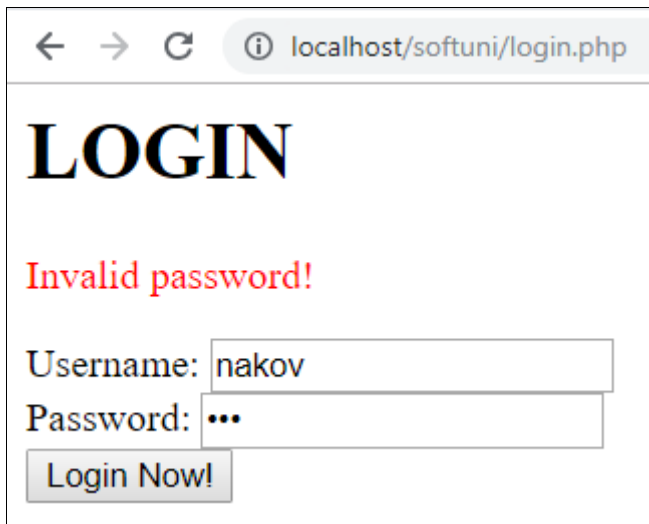
A screenshot of a web browser window. The address bar shows 'localhost/softuni/login.php'. The page has a large 'LOGIN' heading. Below it, there are two input fields: 'Username:' and 'Password:'. Below the password field is a 'Login Now!' button.

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



A screenshot of a web browser window. The address bar shows 'localhost/softuni/login.php'. The page has a large 'LOGIN' heading. Below it, a red error message reads: 'Your username does not exist. You might want to [register](#) first?'. There are two input fields: 'Username:' containing 'nakov1' and 'Password:' containing three dots. A 'Login Now!' button is at the bottom.

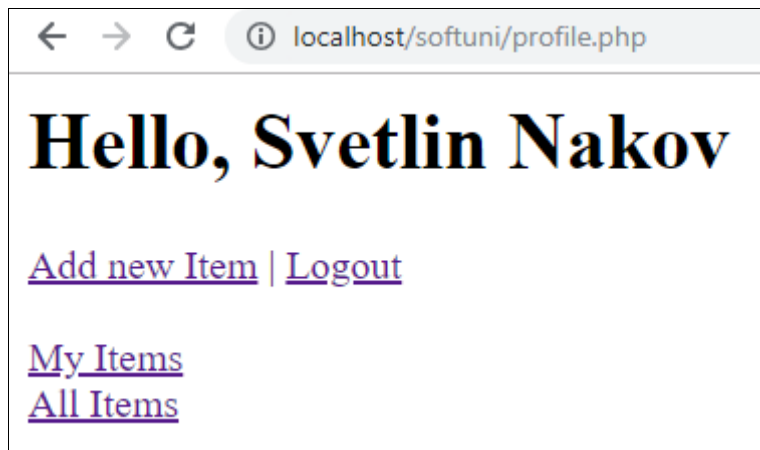
- Form is **submitted**, but the **password** is not the same:



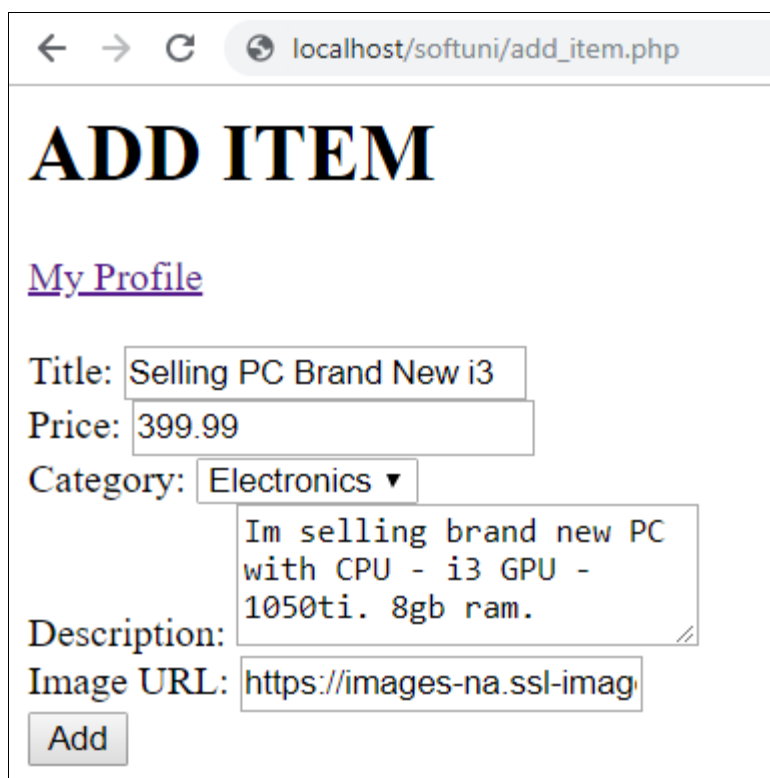
A screenshot of a web browser window. The address bar shows 'localhost/softuni/login.php'. The page has a large 'LOGIN' heading. Below it, a red error message reads: 'Invalid password!'. There are two input fields: 'Username:' containing 'nakov' and 'Password:' containing three dots. A 'Login Now!' button is at the bottom.

- **After** successful login:

You should get a greeting message with your **full name**. You must have 4 links for **adding new item**, for **logout** for viewing **your items** and for viewing **all items**



`add_item.php` should look like this:

A screenshot of a web browser window. The address bar shows 'localhost/softuni/add\_item.php'. The main content area has a large heading 'ADD ITEM'. Below the heading, there is a link 'My Profile'. The form contains the following fields: 'Title' with the value 'Selling PC Brand New i3', 'Price' with the value '399.99', 'Category' with a dropdown menu showing 'Electronics', 'Description' with a text area containing 'Im selling brand new PC with CPU - i3 GPU - 1050ti. 8gb ram.', and 'Image URL' with the value 'https://images-na.ssl-imag'. At the bottom of the form is an 'Add' button.

After adding an item you should be reditected to `my_items.php`  
`my_items.php`:

Sorted by **Category ID Ascending** then by **Price Descending**

<a href="#">←</a> <a href="#">→</a> <a href="#">↻</a> <a href="#">localhost/softuni/my_items.php</a>					
<h1>MY ITEMS</h1>					
<a href="#">Add new Item</a>   <a href="#">My Profile</a>   <a href="#">Logout</a>					
Title	Category	Price	Edit	Delete	
Selling PC Brand New i3	Electronics	399.90	<a href="#">Edit</a>	<a href="#">Delete</a>	
Selling PC Old Pentium 4	Electronics	99.99	<a href="#">Edit</a>	<a href="#">Delete</a>	
Selling Used Sneakers	Clothes	19.50	<a href="#">Edit</a>	<a href="#">Delete</a>	
Selling Car Tyres	Others	435.00	<a href="#">Edit</a>	<a href="#">Delete</a>	

**all\_items.php:**

Sorted by **Location Descending** then by **Category ID Ascending** then by **Price Ascending**

<a href="#">←</a> <a href="#">→</a> <a href="#">↻</a> <a href="#">localhost/softuni/all_items.php</a>						
<h1>ALL ITEMS</h1>						
<a href="#">Add new Item</a>   <a href="#">My Profile</a>   <a href="#">Logout</a>						
Title	Category	Price	Username	Location	Phone	Details
Buying 4gb DDR3 Ram	Electronics	85.00	Gosho	Sofia	02/3333333	<a href="#">Details</a>
Selling PC Old Pentium 4	Electronics	99.99	nakov	Sofia	02/123123	<a href="#">Details</a>
Selling PC Brand New i3	Electronics	399.90	nakov	Sofia	02/123123	<a href="#">Details</a>
Selling Used Sneakers	Clothes	19.50	nakov	Sofia	02/123123	<a href="#">Details</a>
Selling Winter Jacket	Clothes	100.00	Gosho	Sofia	02/3333333	<a href="#">Details</a>
Selling Car Tyres	Others	435.00	nakov	Sofia	02/123123	<a href="#">Details</a>
Selling Gaming PC	Electronics	1999.99	pesho	Pleven	088121212	<a href="#">Details</a>
Buying 3 Stripe Tracksuit	Clothes	95.00	pesho	Pleven	088121212	<a href="#">Details</a>
Selling Golf 2 Side Mirrors	Others	65.00	pesho	Pleven	088121212	<a href="#">Details</a>

**edit.php?id={item id}** (Edit existing item):

← → ↻ ⓘ localhost/softuni/edit.php?id=2

# EDIT ITEM

[My Profile](#)

Title:

Price:

Category:

Description:

Image URL:



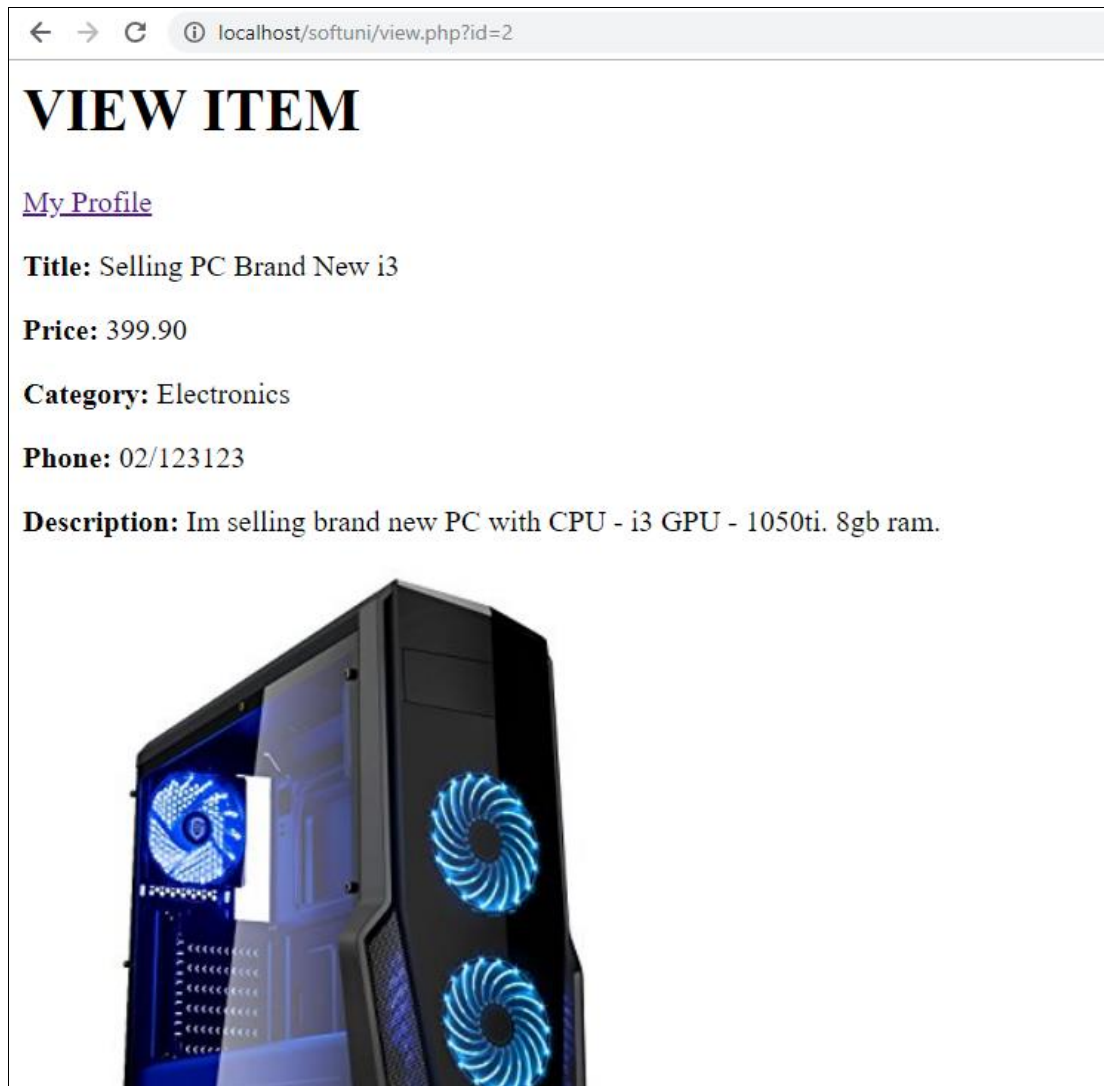
- After **successful submission**, the operation is **edited**, **redirect** to the list of my items **my\_items.php**

**delete.php?id={item\_id}** (Delete existing item):

- **Deletes** the item and **redirects** to of my items **my\_items.php**.

**view.php?id={item\_id}** When you click “details” from **all\_items.php** you should be able to see the **title**, **price**, **category**, **user’s phone**, **description** and **image** about the item you selected. This option is available only for **logged in** users.





## 1. Database

Create a database “softuni\_bazar” and add the following tables:

### Data Definition

#### 1.1. DDL – Users

users		
Column Name	Type	Constraints
username	String	4...255. Unique.
password	String	4...255
full_name	String	5...255
location	String	4...255
phone	String	6...255

## 1.2. DDL – Categories

Categories		
Column Name	Type	Constraints
name	String	3...50

## 1.3. DDL – Items

Items		
Column Name	Type	Constraints
title	string	3...255
price	decimal	1...50
description	string	10...10000
image	string	3...255
category_id	int	1...4294967295
user_id	int	1...4294967295

## 1.4. DML – Add Categories

**Add** the following **Categories**: Save the DDL in **export.sql** file in the top-level directory of your project.

name
Electronics
Clothes
Other

## 2. HTML

### 2.1. 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. **Location** with minimum length of **4** symbols
6. **Phone** with minimum length of **6** symbols
7. Button, submitting the form with value **“Register”**

All fields are **required**. Use a convention for name attributes: **lower\_snake\_case**.

## 2.2. 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**.

## 2.3. New Item Page

1. **Title** with minimum length of **3** symbols and max length – **255**
2. **Price** with minimum length of **1** symbols and max length – **50**
3. **Category** which is **empty dropdown** (will be later populated with PHP)
4. **Description** containing **10 to 10,000** symbols
5. **Image** string containing url
6. Button, submitting the form with text **“Add”**

All fields are **required**. Use a convention for name attributes: **lower\_snake\_case**.

## 3. 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.

### 3.1. 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\_item, edit, all\_items, my\_items, view** and **logout;**
4. Extract common operation such as autoloading classes in **“common.php”**.

### 3.2. 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.

### 3.3. Repository Layer

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

### 3.4. Service Layer

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

## 4. Item Trading Application

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