

Spring Fundamentals Retake Exam

Coffee Shop Application

Exam for the ["Spring Fundamentals" course @ SoftUni](#).

The **Coffee Shop Application** is here to help with the normal work of the cafe. The application records all orders that are not yet ready and reminds the employees of them by visualizing a picture of the category, the product's name and its price. It also calculates the approximate time that will be required for the execution of the orders. As an additional functionality, statistics are kept, on which employees can process how many orders at that moment. At this stage, the functionality of the application is small but later it can be expanded upon.

1. Database Requirements

The **Database** of the **Coffee Shop** application needs to support **3 entities**:

User

- Has an **Id** – **UUID-string** or **Long**
- Has a **Username** (**unique**)
 - The **length** of the **username** must be **between 5** and **20** characters (both numbers are **INCLUSIVE**).
- Has a **First Name**
 - Can be **null**.
- Has a **Last Name**
 - The **length** of the **last name** must be **between 5** and **20** characters (both numbers are **INCLUSIVE**).
- Has a **Password**
 - The **length** of the **password** must be more than 3 (**INCLUSIVE**).
- Has an **Email**
 - Must contain a '@' symbol.
 - The email must be **unique**.

Order

- Has an **Id** – **UUID-string** or **Long**
- Has a **Name**
 - The **length** of the **name** must be **between 3** and **20** characters (both numbers are **INCLUSIVE**).
- Has a **Price**
 - The **price** must be a **positive number**

- Has an **Order time**
 - The **date and time** that **cannot** be in the **future**
- Has a **Category**
 - Has **ONLY ONE** category
 - This is the relation with categories.
- Has a **Description**
 - The **length** of the **description** must be **at least 5 (INCLUSIVE)** characters
 - The description is a **long text field**.
- Has a **Employee (user)**
 - A user that **adds this order** to the **Coffee Shop** application

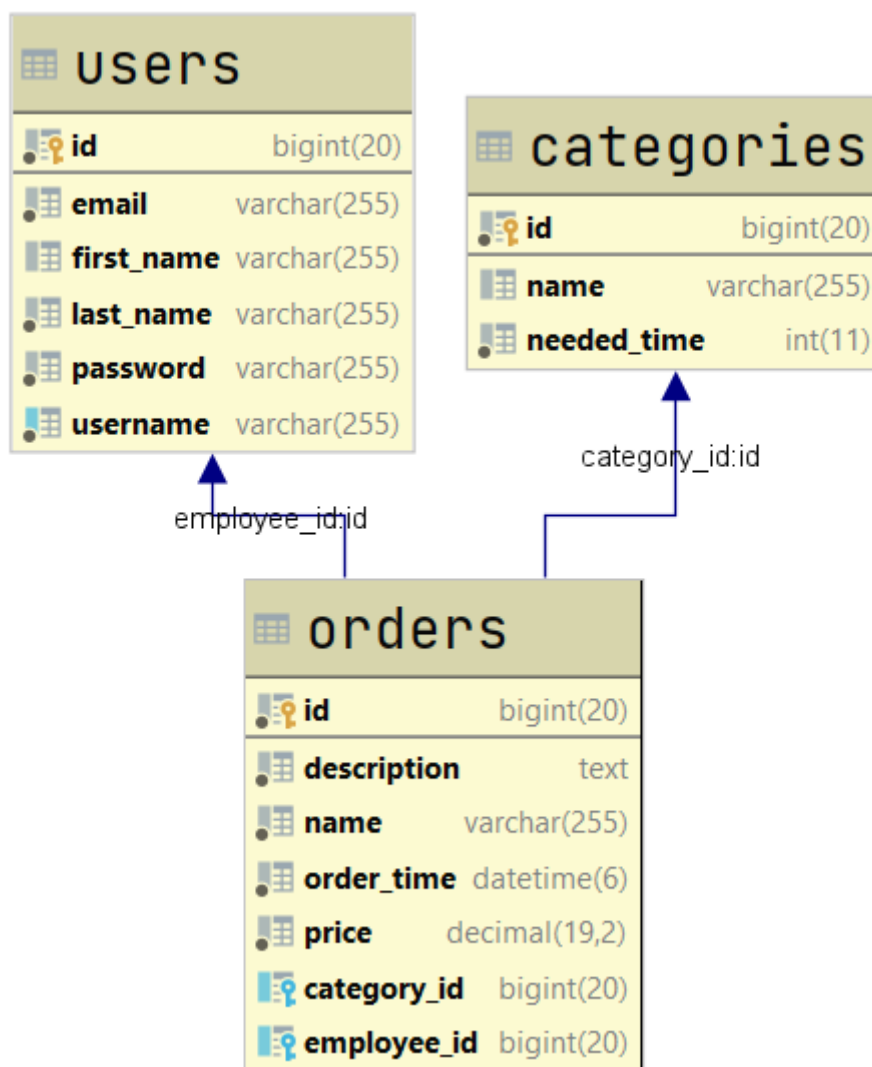
Category

- Has an **Id – UUID-string or Long**
- Has a **Name**
 - An option between (**Coffee, Cake, Drink, Other**)
- Has a **Needed Time (just an integer)**
 - The approximate **time in minutes** needed for the **preparation** of a product of the **specified category**.
 - The needed time for a **Drink** is **1** min.
 - The needed time for **Coffee** is **2** min.
 - The needed time for an **Other** is **5** min.
 - The needed time for a **Cake** is **10** min.

Nullable/Empty strings are not allowed unless explicitly mentioned.

Implement the entities with the **correct datatypes** and implement **repositories** for them.

Here is the ER Diagram:

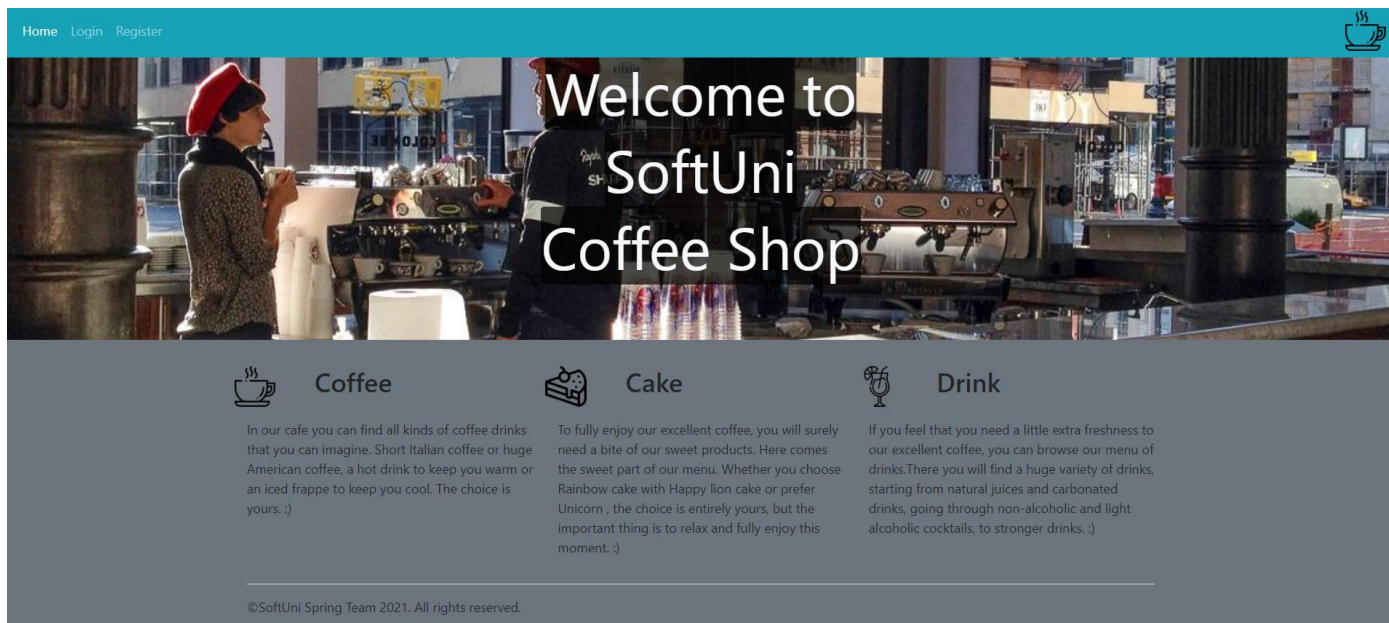


2. Initialize categories

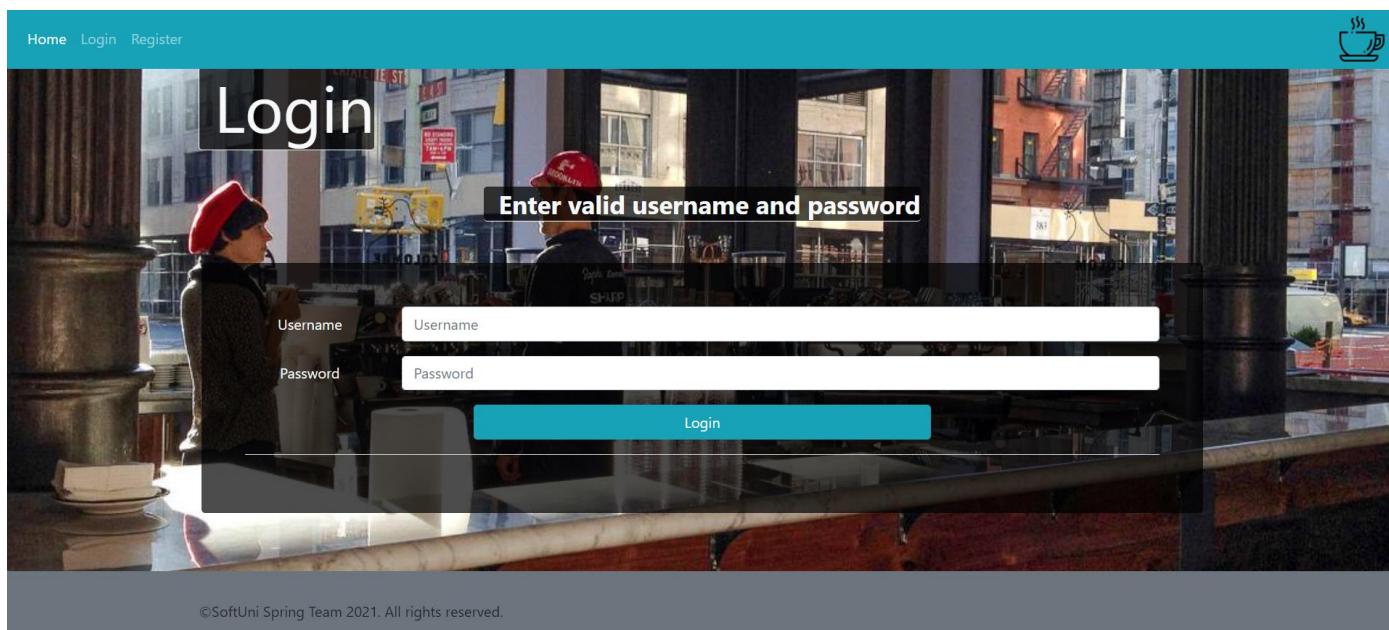
- Implement a method that checks (when app started) if the database does not have any category and initialize them
 - You are free to do this in some different ways.
 - Don't **forget** to add the **needed Time**

3. Page Requirements

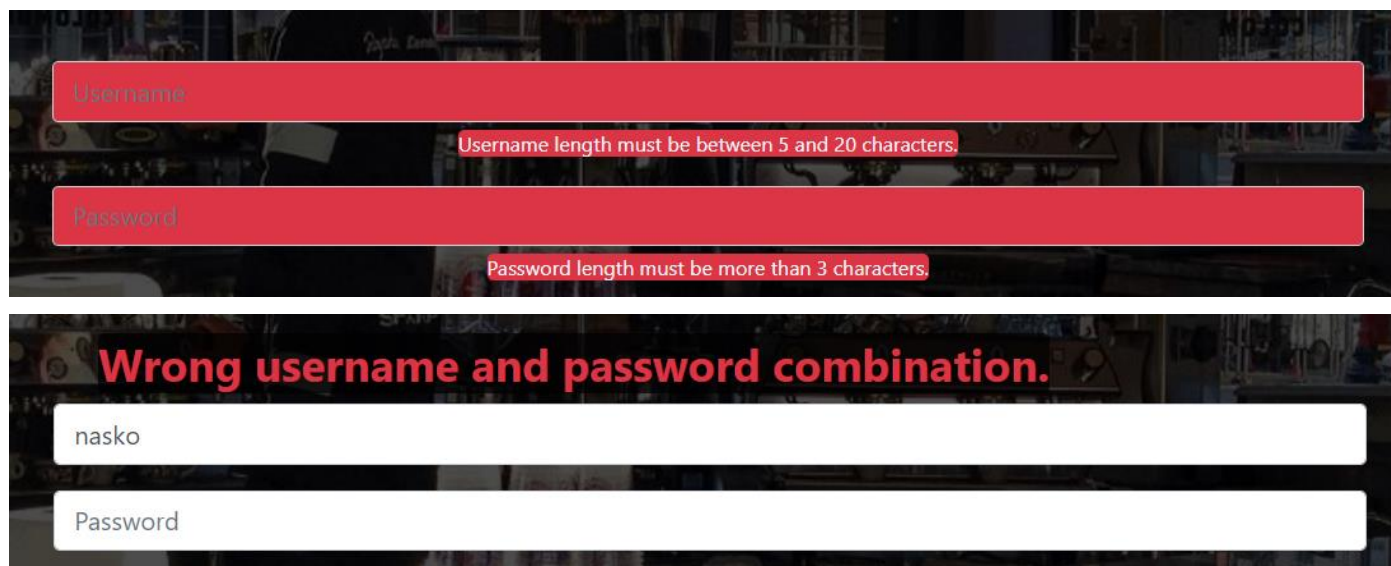
Index Page (logged out user)



Login Page (logged out user)

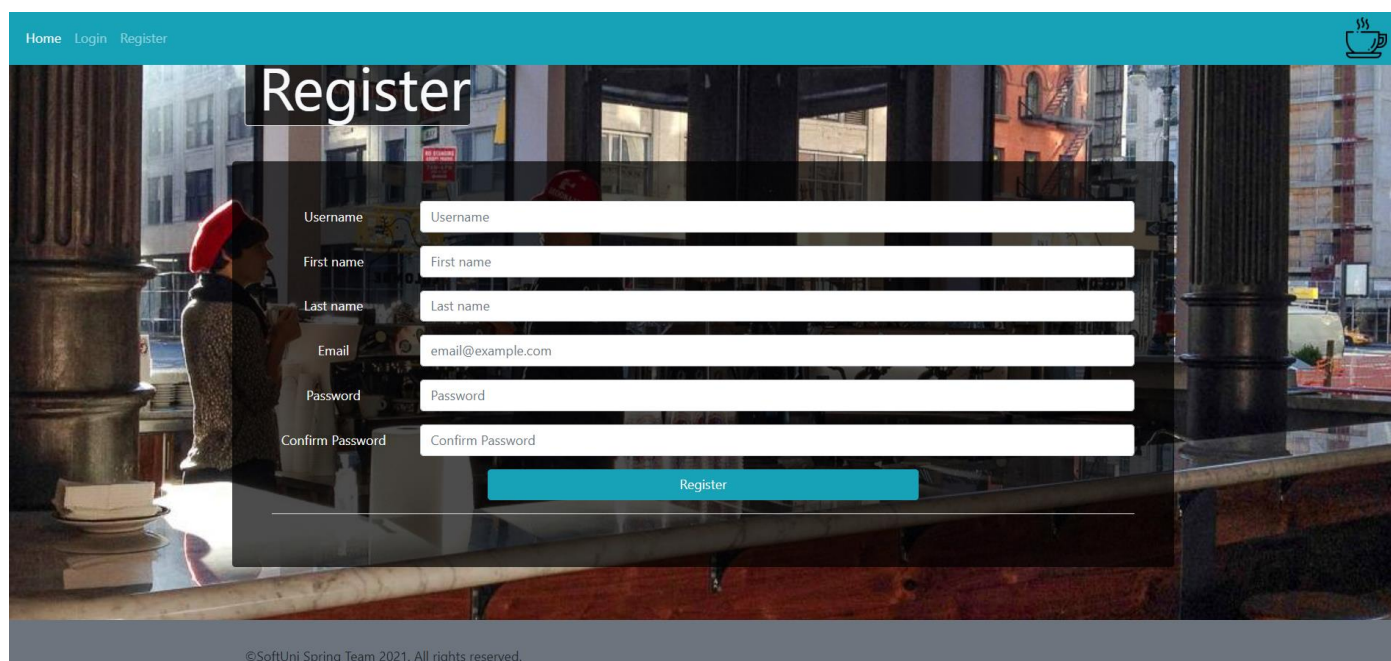


Login Page validations



The image shows two examples of login form validations. The first example shows a form with a red border and a red background. The 'Username' field has a red border and a red background, and the 'Password' field has a red border and a red background. A red message box says 'Username length must be between 5 and 20 characters.' and another red message box says 'Password length must be more than 3 characters.' The second example shows a form with a white border and a white background. The 'Username' field contains the text 'nasko' and the 'Password' field is empty. A red message box says 'Wrong username and password combination.'

Register Page (logged out user)



The image shows a register page for a logged out user. The page has a teal header with links 'Home', 'Login', and 'Register'. A coffee cup icon is in the top right corner. The main content area has a background image of a person in a red beret. The 'Register' title is in a large, bold font. Below the title is a form with fields for 'Username', 'First name', 'Last name', 'Email', 'Password', and 'Confirm Password'. The 'Email' field contains the text 'email@example.com'. A teal 'Register' button is at the bottom of the form. The footer contains the text '©SoftUni Spring Team 2021. All rights reserved.'

Register Page validations

- Note: it is not necessary to show message for different passwords, just not save the user and redirect again to the register page.

Username Username length must be between 5 and 20 characters.

First name

Last name Last name length must be between 5 and 20 characters.

Email Enter valid email address

Password Password length must be more than 3 characters.

Confirm Password Password length must be more than 3 characters.

[Register](#)

Navigation (Guest user)

- Note: can access only to **Index**, **Login**, **Register** pages.



Navigation (Registered user)

- Note: can access only to **Home**, **Add Order**, **Logout**.



Add Order

Home Add Order Logout

Add Order

Name

Price

Order time

Category

Description

[Add order](#)

©SoftUni Spring Team 2021. All rights reserved.

Add Order validation

The screenshot shows a form for adding a new order. The form has five input fields: Name, Price, Order time, Category, and Description. Each field has a red border and a red error message below it. The 'Add order' button is at the bottom.

Field	Value	Error Message
Name	1	Name must be between 3 and 20
Price	-1	The price must be positive
Order time	03/20/2022 11:44 AM	Order time cannot be in the future
Category	Category	You must select the category
Description	1	Description size must be minimum 5 char

Add order

Home Page (with orders)

The screenshot shows the Home Page of the application. It features a header with navigation links (Home, Add Order, Logout) and a coffee cup icon. The main content area is titled 'All active Orders' and displays a list of orders with their preparation time (18 minutes). The orders are listed in a table with columns for icon, name, price, and status. To the right, there is a section titled 'Orders by employee' showing the number of orders for each employee.

Time to prepare all orders(in min): 18			
	Apple cake	4.00	Ready
	Mocachino	3.00	Ready
	Sandwich	2.00	Ready
	Cola 500ml	1.00	Ready

Orders by employee	
	Employee - gosho Number of orders: 3
	Employee - pesho Number of orders: 1

NOTE: The orders are ordered by their price in **descending** order

NOTE: The **picture** in front of every order **depends** of order's **category**.

NOTE: The employees are ordered by their count of orders in **descending** order.

The templates have been given to you in the application skeleton, so make sure you implement the pages correctly.

NOTE: The templates should look **EXACTLY** as shown above.

NOTE: The templates do **NOT** require **additional** **CSS** for you to write. Only **bootstrap** and the **given** **css** are enough.

4. Functional Requirements

The **Functionality Requirements** describe the functionality that the **Application** must support.

The **application** should provide **Guest** (not logged in) users with the functionality to **login**, **register** and **view** the **Index** page.

The **application** should provide **Users** (logged in) with the functionality to **logout**, **add an Order**, **view all Orders in DB (Home page)** and **Ready a single one** from orders.

In **Coffee Shop Application**, the navbar should redirect to appropriate **URL depending** on that if the user is logged in.

The **application** should provide **functionality** for **adding orders** with **category and employee**. Also can **Ready** them and remove from the database.

Ready button remove the selected order from the database and again **redirect** to the **home** page

The first **header** on the **home** page shows the **total needed time to prepare all orders in minutes(integer number)** in the database at this moment.

Order by Employee tab show **every** registered employee and his **unready orders** at this moment.

The **application** should **store** its **data** into a **MySQL** database.

5. Security Requirements

The **Security Requirements** are mainly access requirements. Configurations about which users can access specific functionalities and pages.

- **Guest** (not logged in) users can access **Index** page.
- **Guest** (not logged in) users can access **Login** page.
- **Guest** (not logged in) users can access **Register** page.
- **Users** (logged in) can access **Home** page.
- **Users** (logged in) can access **Add Order** page.
- **Users** (logged in) can access **Logout** functionality.

6. Scoring

Database – 10 points.

Pages – 25 points.

Functionality – 35 points.

Security – 5 points.

Validations – 15 points.

Code Quality – 10 points.