



## Reseller Application

### Regular Exam [Spring Fundamentals]

**ResellerApp** is the newest online marketplace dedicated to buying and selling new or used items. At any given time, the site has a wide variety of listings ranging from hand tools, clothes and books to accessories, pro audio gear, bikes, smartphones, and more...

There are several requirements you must follow in the implementation:

#### 1. Database Requirements

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

##### User

- **Id** – Accepts **UUID-String** or **Long** values
- **Username**
  - The **length** of the **values** should be **between 3 and 20** characters long (both numbers are **INCLUSIVE**)
  - The values should be **unique** in the database
- **Password**
  - The **length** of the **values** should be **between 3 and 20** characters long (**INCLUSIVE**)
- **Email**
  - The values should contain a '@' symbol)
  - The values should be **unique** in the database
- **Created Offers**
  - The offers contains users offers. One **user** may have **many offers** and one **offer** can be **created by only one user**
- **Bought Offers**
  - Collection with the **offers** that the **user** bought. **One user may have many offers** in the collection.

##### Offer

- **Id** – Accepts **UUID-String** or **Long** values
- **Description**
  - The **length** of the **values** should be **between 2 and 50** characters long (both numbers are **INCLUSIVE**)
- **Price** – floating point number
  - The price must be a **positive number**
- **Condition**
  - One offer **has one** condition and one condition can have **many offers**

##### Condition

- **Id** – Accepts **UUID-String** or **Long** values
- **Condition name**
  - an option between (**EXCELLENT**, **GOOD** and **ACCEPTABLE**)
- **Description**
  - For EXCELLENT - "In perfect condition"
  - For GOOD - "Some signs of wear and tear or minor defects"
  - For ACCEPTABLE - "The item is fairly worn but continues to function properly"

**Nullable/Empty/Blank values are not allowed unless explicitly mentioned.** Implement the entities with the **correct data types** and implement the **repositories** for them.

## 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
  - You can skip the description if you want

## 3. Page Requirements

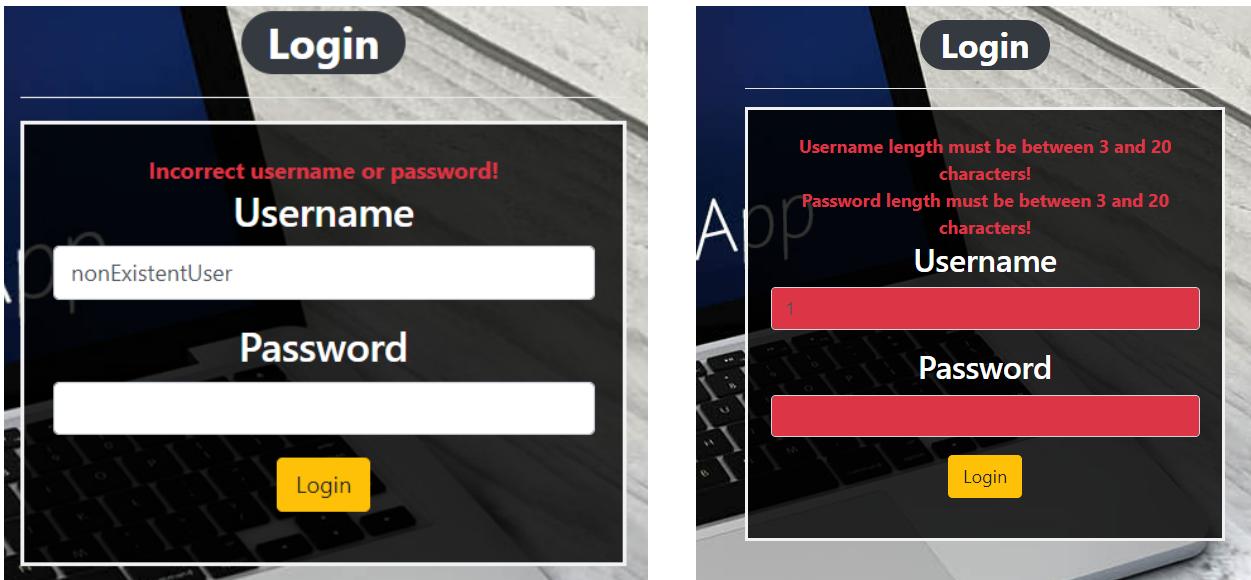
### Index Page (logged out user)



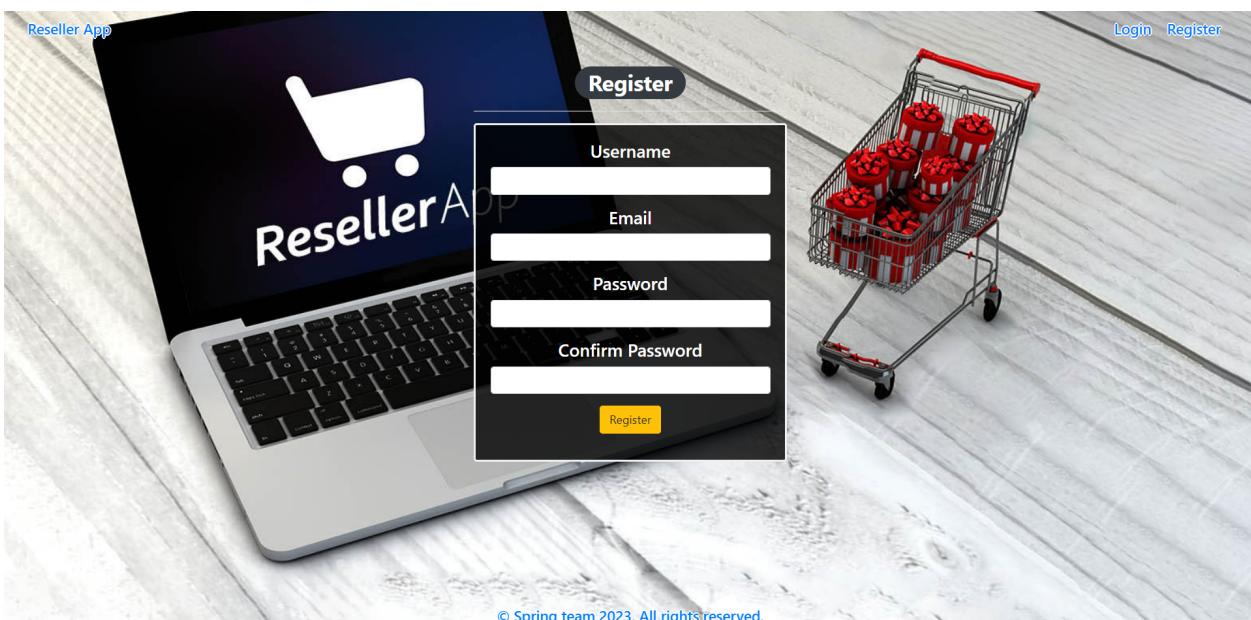
### Login Page (logged out user)



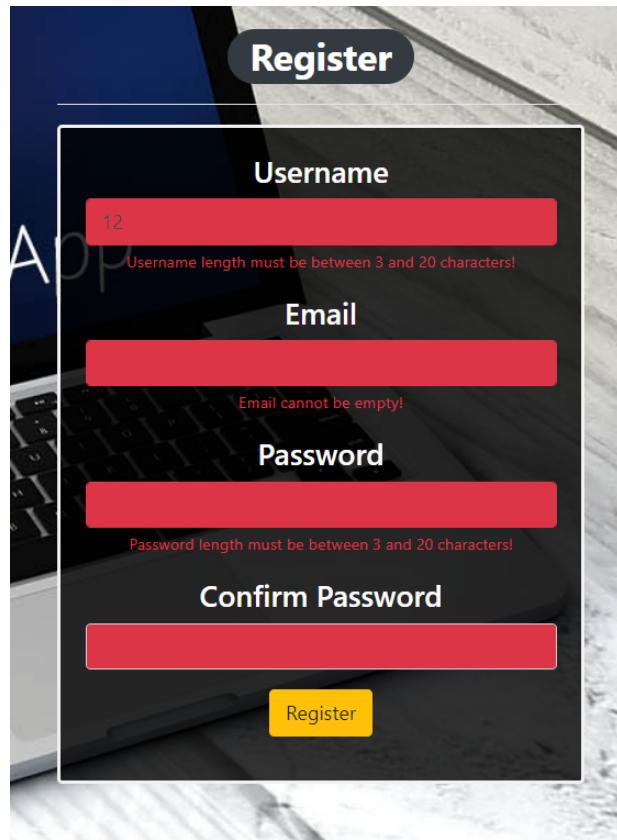
## Login Page validations



## Registration Page (logged out user)

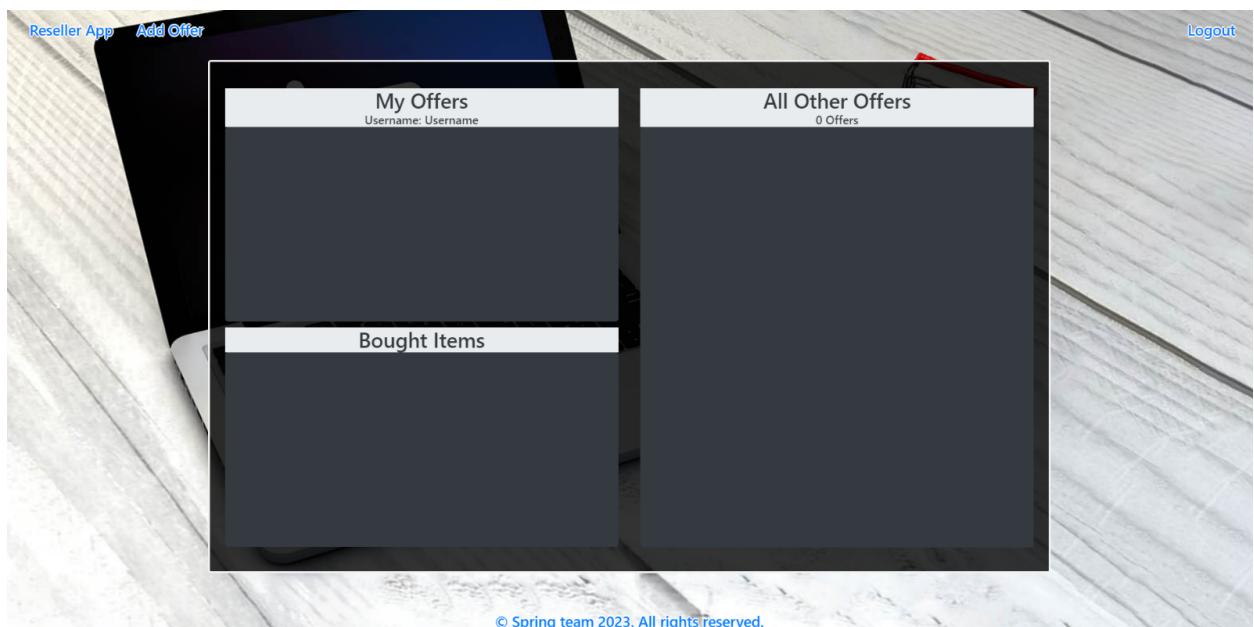


## Registration Page validations



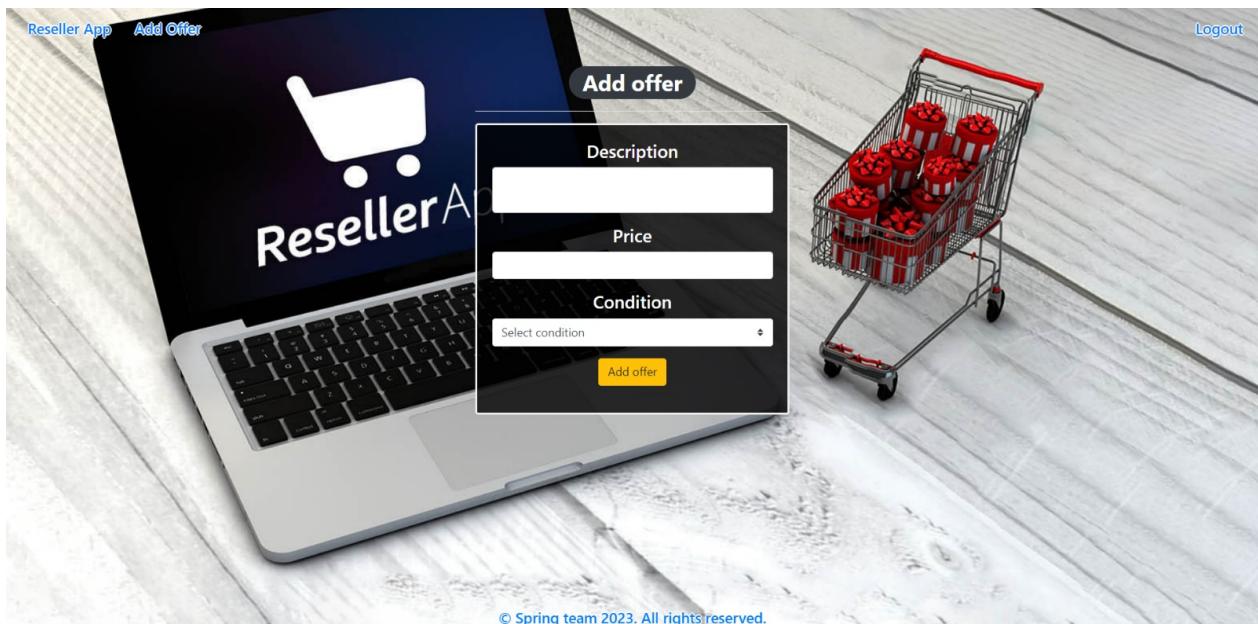
## Home Page (without having any offers)

- The left and upper section (My Offers) of the page should visualize **all of the current logged user offers** from the database which he owns.
- The the left and down section (Bought Items) of the page should visualize **the bought items (offers) by the current logged user from All Other Offers** section
- The right section (All Other Offers) of the page should visualize the available (not bought) **offers** from **all other users** (except the current logged one).

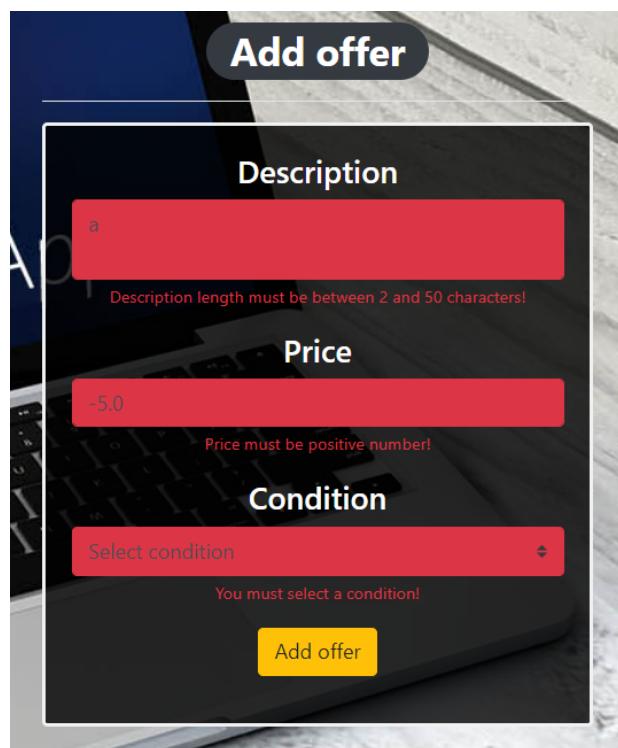


© Spring team 2023. All rights reserved.

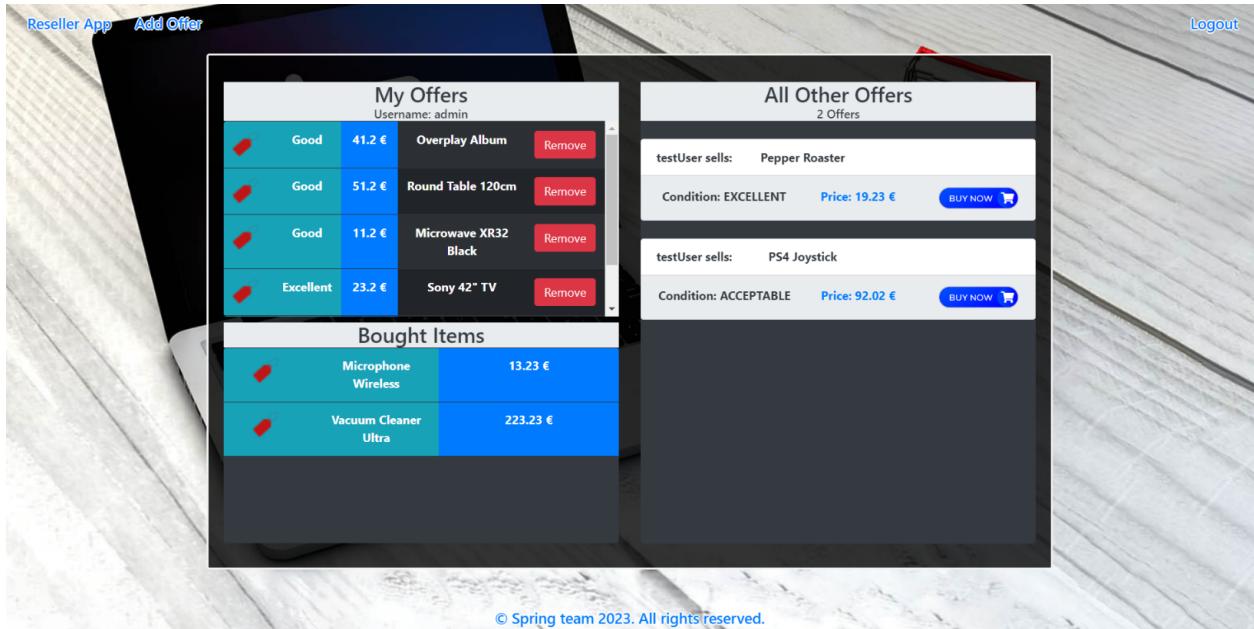
## Add Offer



## Add Offer Validation



## Home Page (with offers)



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

### NOTES:

- The templates should look **EXACTLY** as shown above
- 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 log in, register and view the Index page.

The application should provide **Users** (logged in) with the functionality to **log out, add a new offer (Add offer page), view all offers (Home page) and buy an offer from All Other Offers to Bought Items.**

Reseller Application in navbar should **redirect** to the appropriate URL depending on that if the **user is logged in**. The application should provide functionality for **adding offers** with conditions of **Excellent, Good or Acceptable**. The offers should be separated into different sections according to their ownership.

When the user clicks on the **BUY NOW button of some offer**, he buys the offer to **his Bought Items**. You should **not delete** this offer from DB. The offer should be removed from the collection with offers from the seller and should be added to the collection with boughtOffers of the buyer.

The **Remove** button in My Offers section **should remove the offer** from DB.

Bellow the All Other Offers banner is located an info bar that shows the sum of the offers in this field.

The application should store its data in 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;
  - **Login** page;
  - **Register** page.
- **Users** (logged in) can access:
  - **Home** page;
  - **Add Offer** page;
  - **Logout** functionality.