# Spring Fundamentals Exam

# LikeBook Application

Exam for the ["Spring Fundamentals" course @ SoftUni](https://softuni.bg/trainings/3710/spring-fundamentals-may-2022).

With its easy-to-use interface and detail-rich profiles, **LikeBook** App makes it clear why it's one of the most enduring social network apps. This is the service to try if you're looking for a connection with friends and new communities. There are several requirements you must follow in the implementation:

*\*An example skeleton is included in the resources, which can be used as desired.*

## Database Requirements

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

### User

* Has an **Id – “UUID-String” or Long**
* Has a Username (unique, not null)
  + Username length must be between 3 and 20 characters (inclusive of 3 and 20).
* Has a Password (not null)
  + Password length must be between 3 and 20 characters (inclusive of 3 and 20).
* Has an Email (unique, not null)
  + Must contain '@'.

### Post

* Has an **Id – “UUID-String” or Long**
* Has a Content (not null)
  + Content length must be between 2 and 150 characters (inclusive of 2 and 150).
* Has a User (not null)
  + The creator of the post. One post can have only one user and one user may have many posts.
* Has a User Likes
  + The user likes contains users. One user may like many posts and one post can be liked by many users.
* Has a Mood (not null)
  + One post has one mood and one mood can have many posts.

### Mood

* Has an **Id** – **“UUID-String” or Long**
* Has a **Mood name** (unique, not null)
  + an option between (Happy, Sad and Inspired)
* Has a Description (optional)

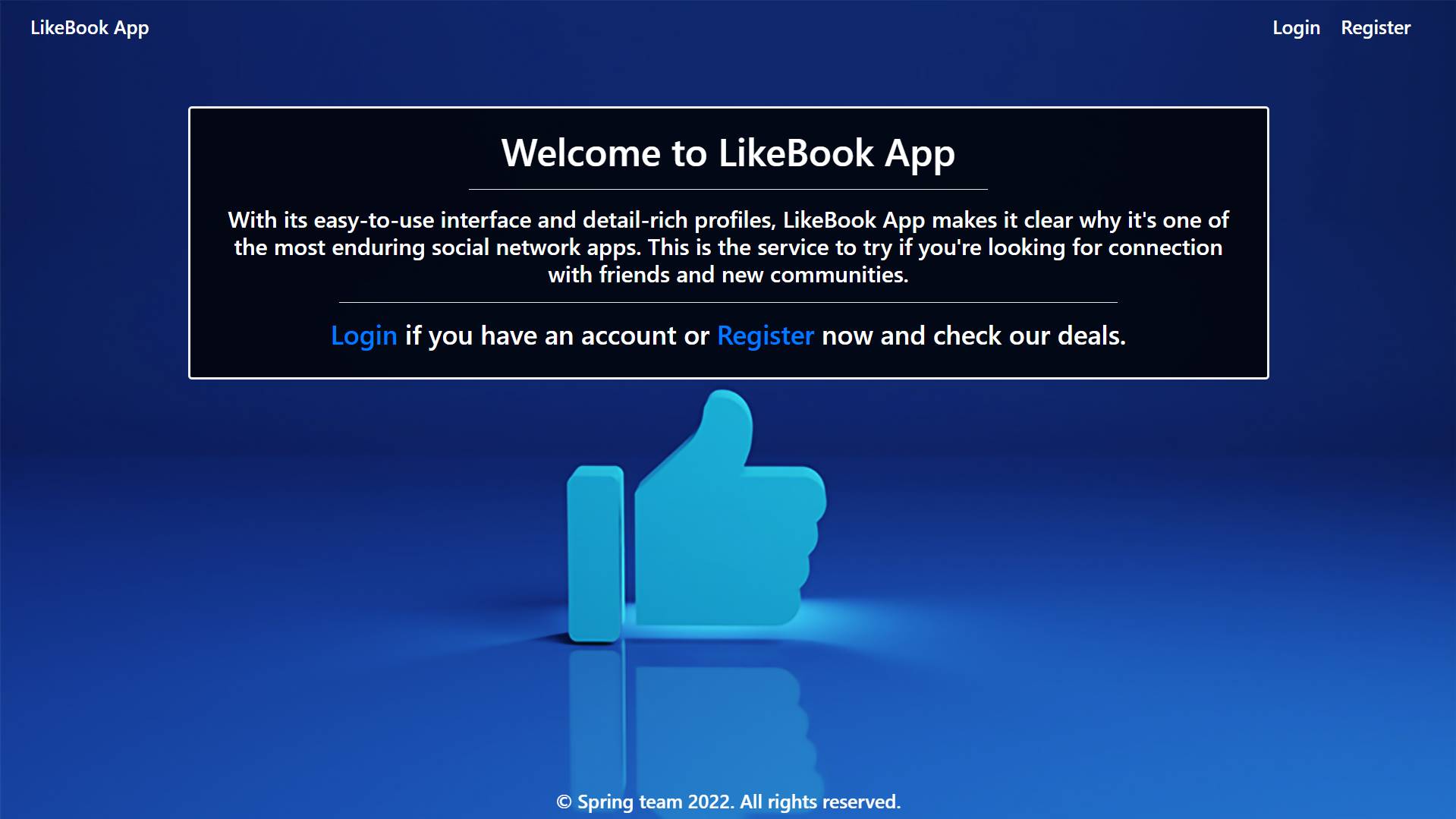
Implement the entities with the **correct data types** and implement **repositories** for them.

## Initialize moods

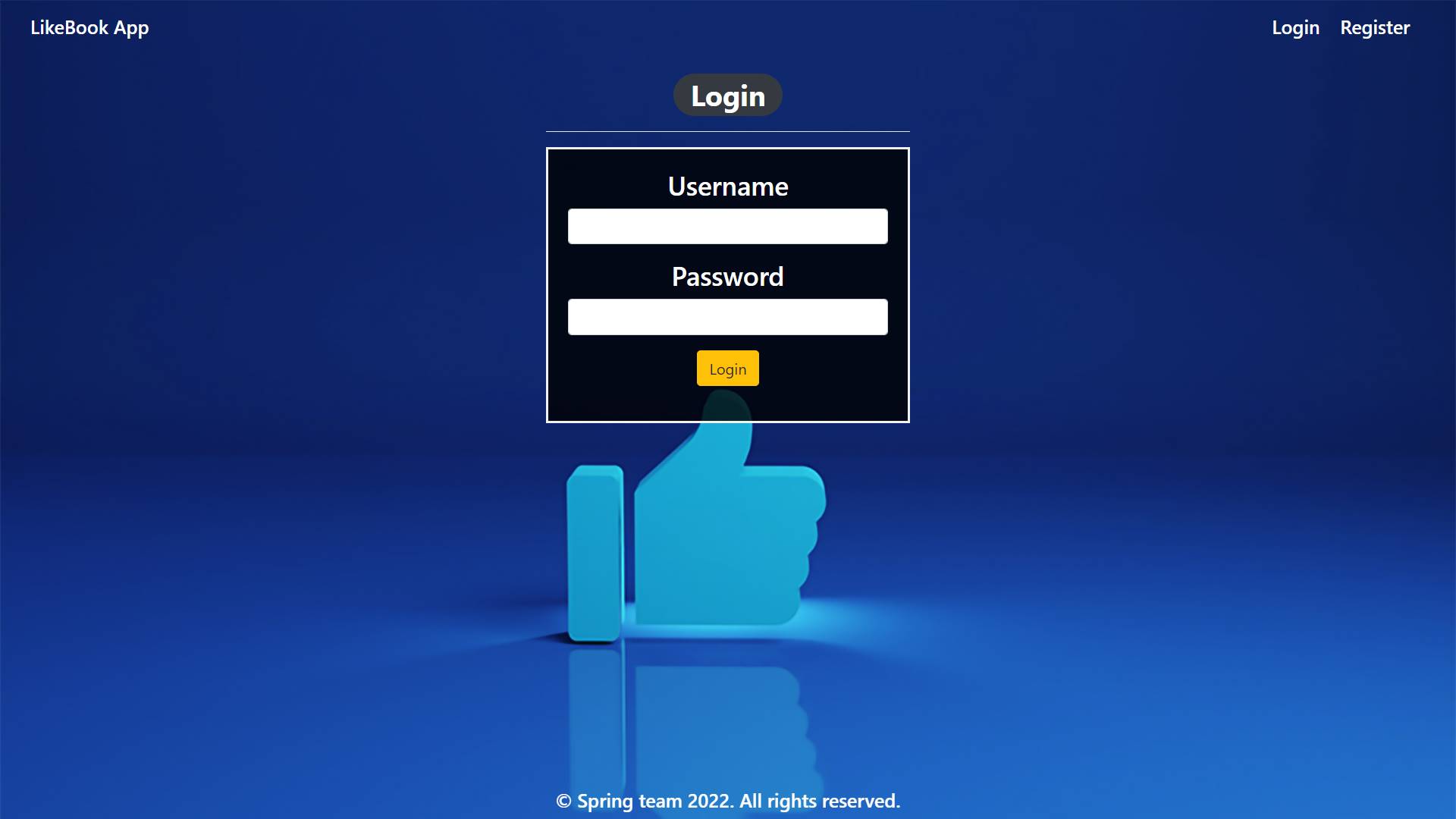
* Implement a method that checks (when the app started) if the database does not have moods and   
  initialize them
  + **You are free to do it in different ways.**

## Page Requirements

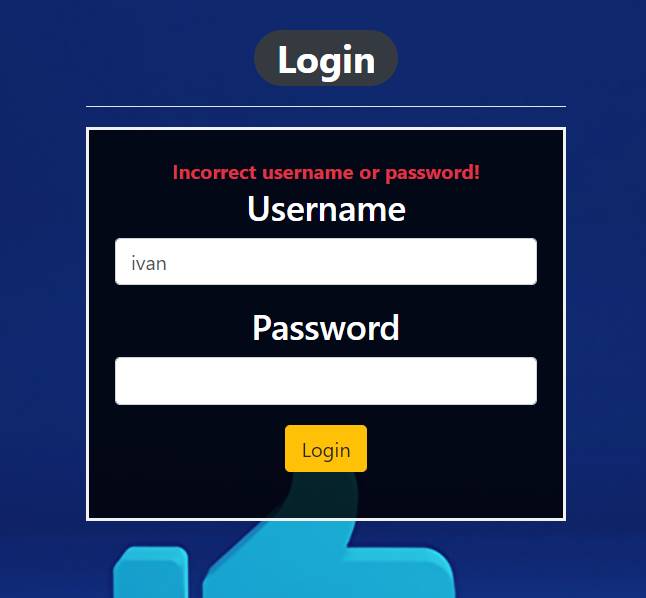
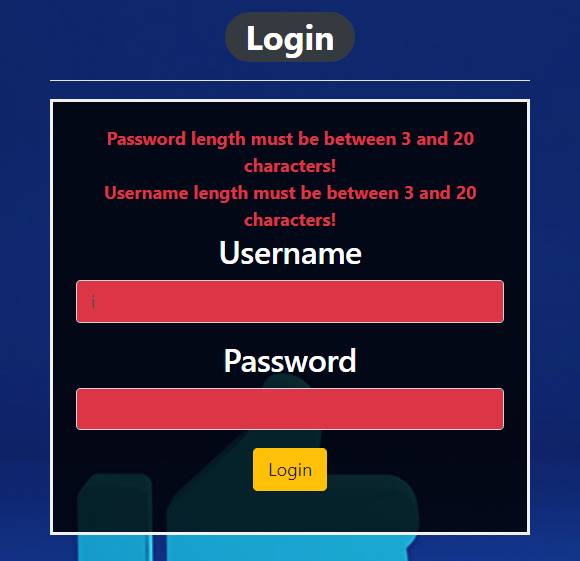
### Index Page (logged out user)



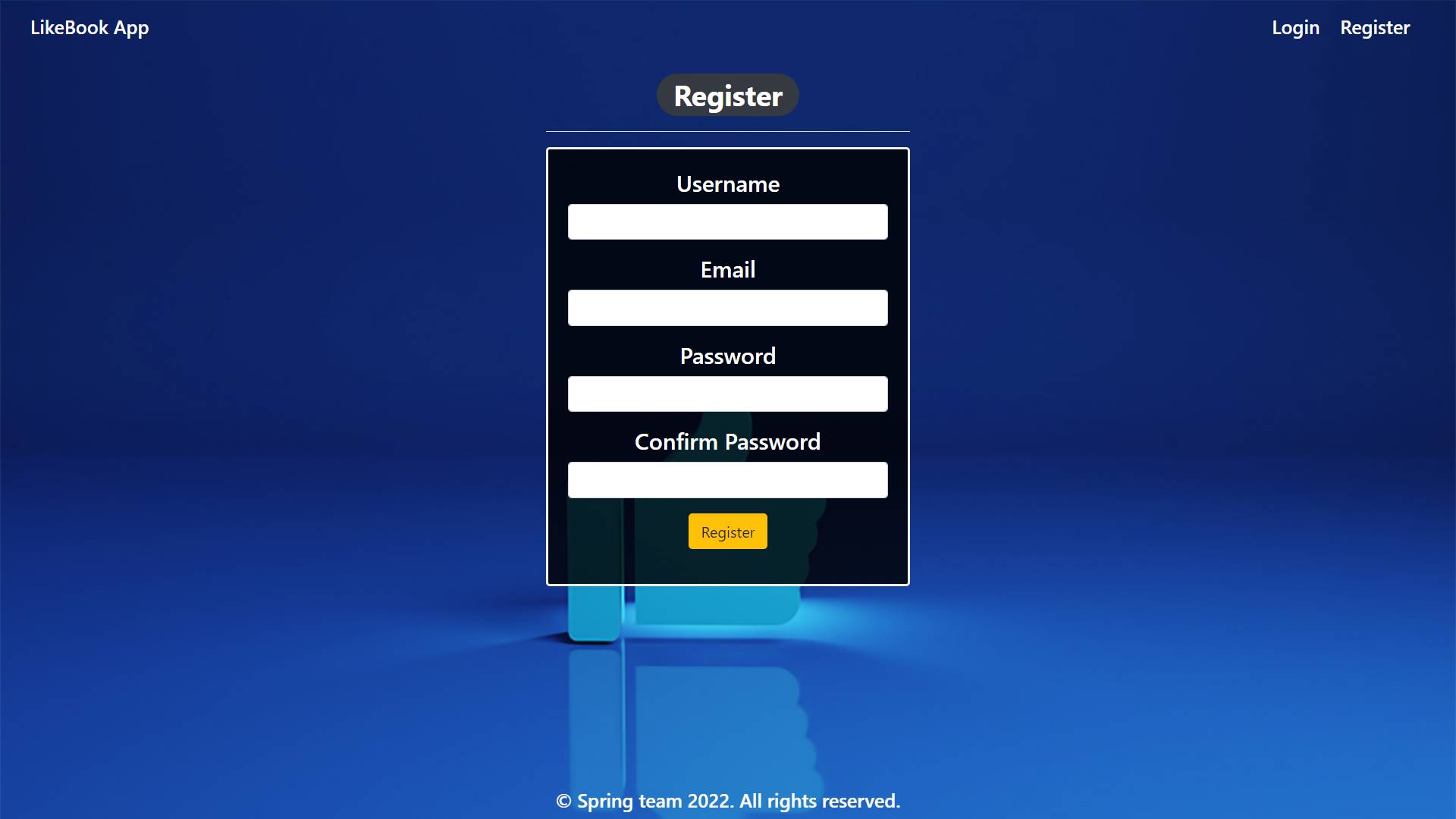
### Login Page (logged out user)



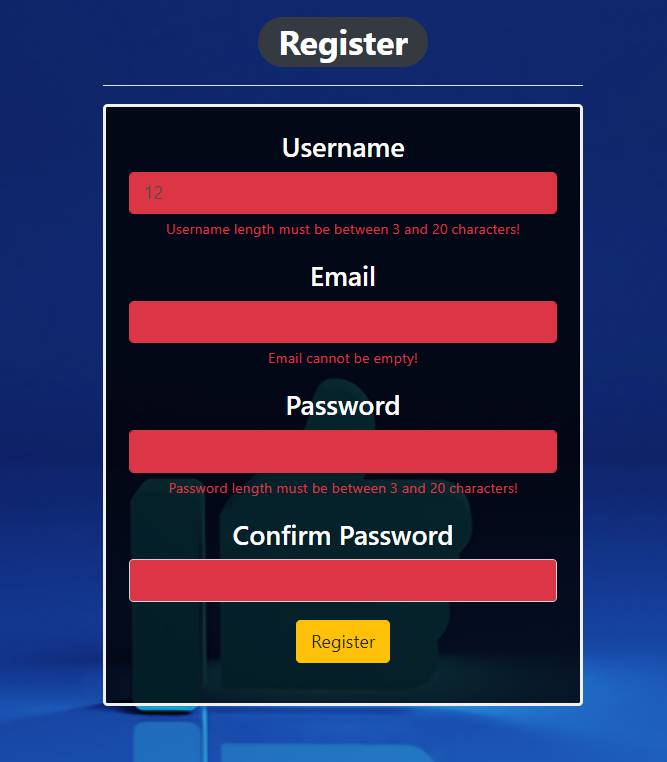
### Login Page validations

### Register Page (logged out user)

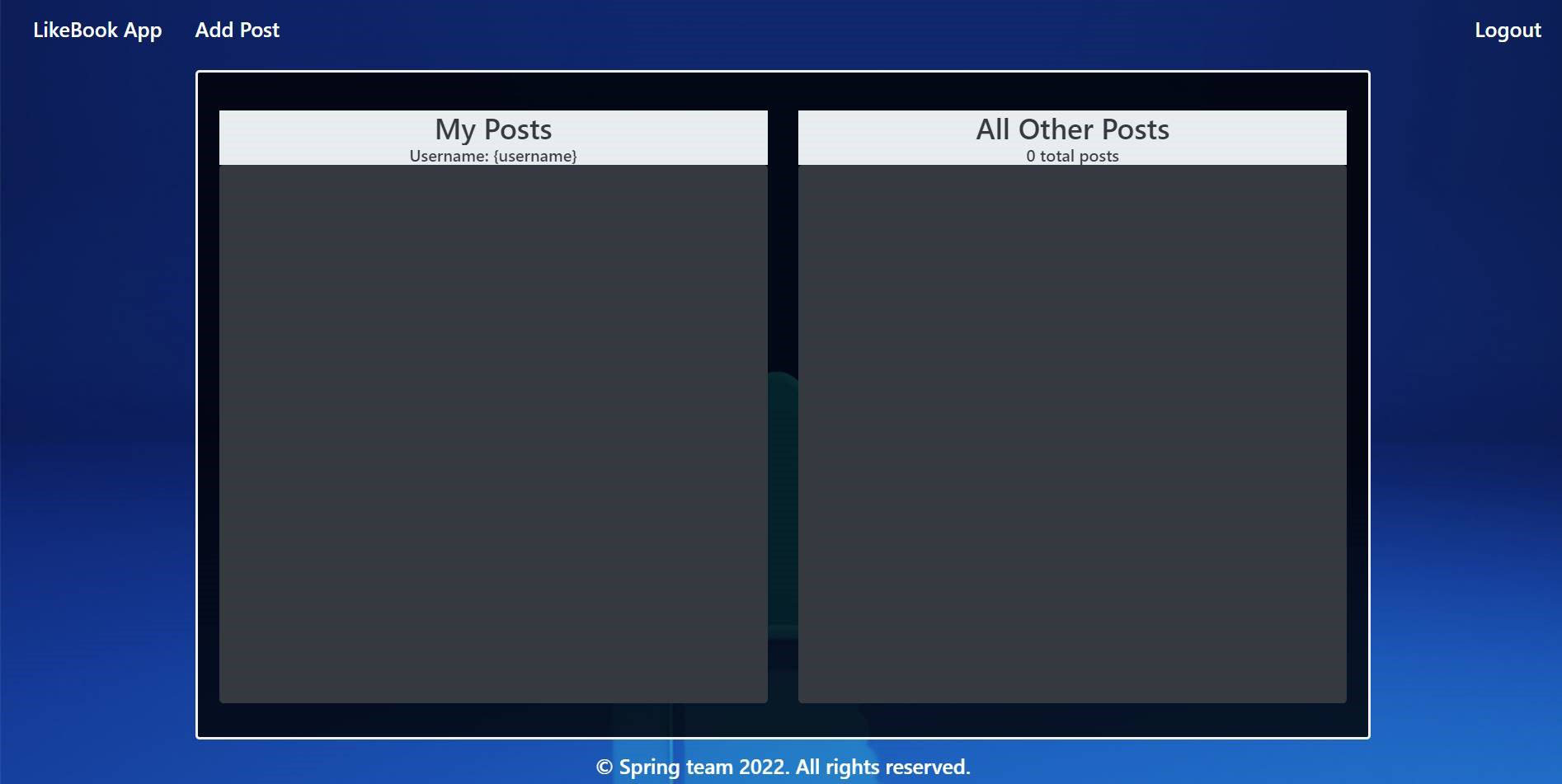


### Register Page validations

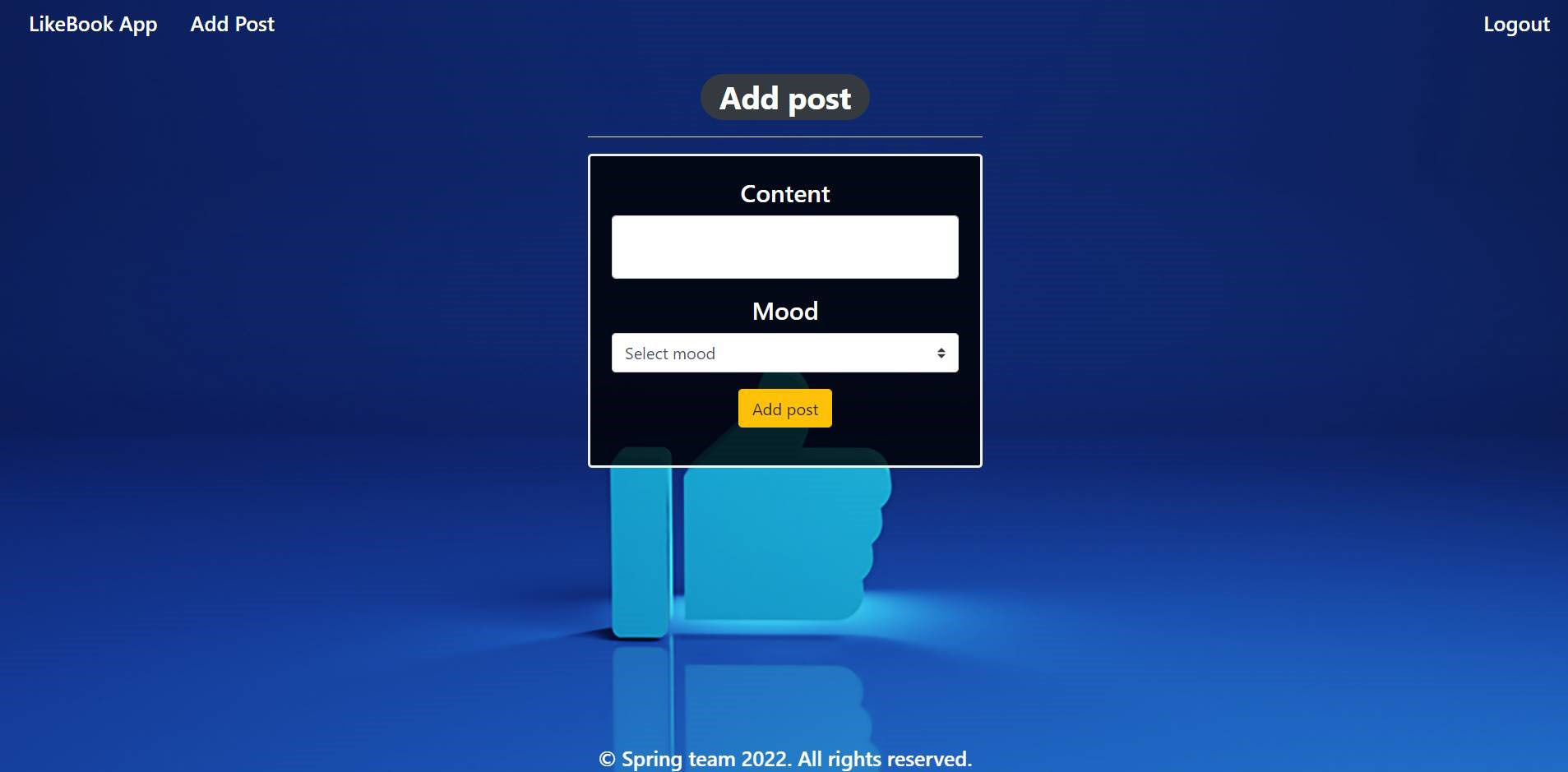


### Home Page (without having any posts)

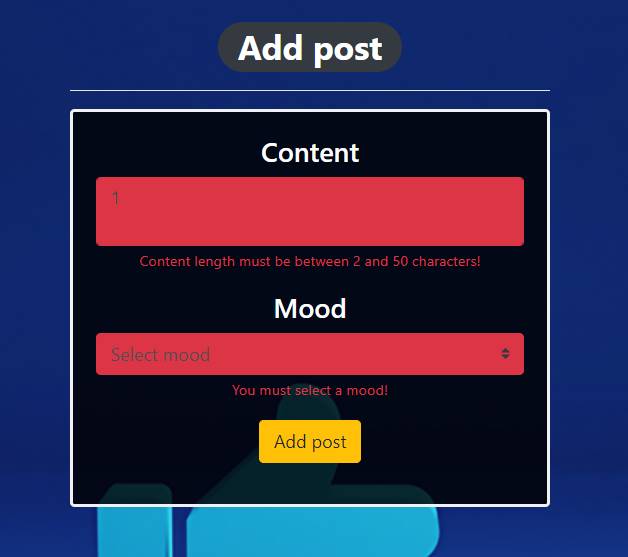
* Note: The left section of the page should visualize **current logged user’s posts** from the database.
* Note: The right section (All Other Posts) of the page should visualize **posts** created by other users.



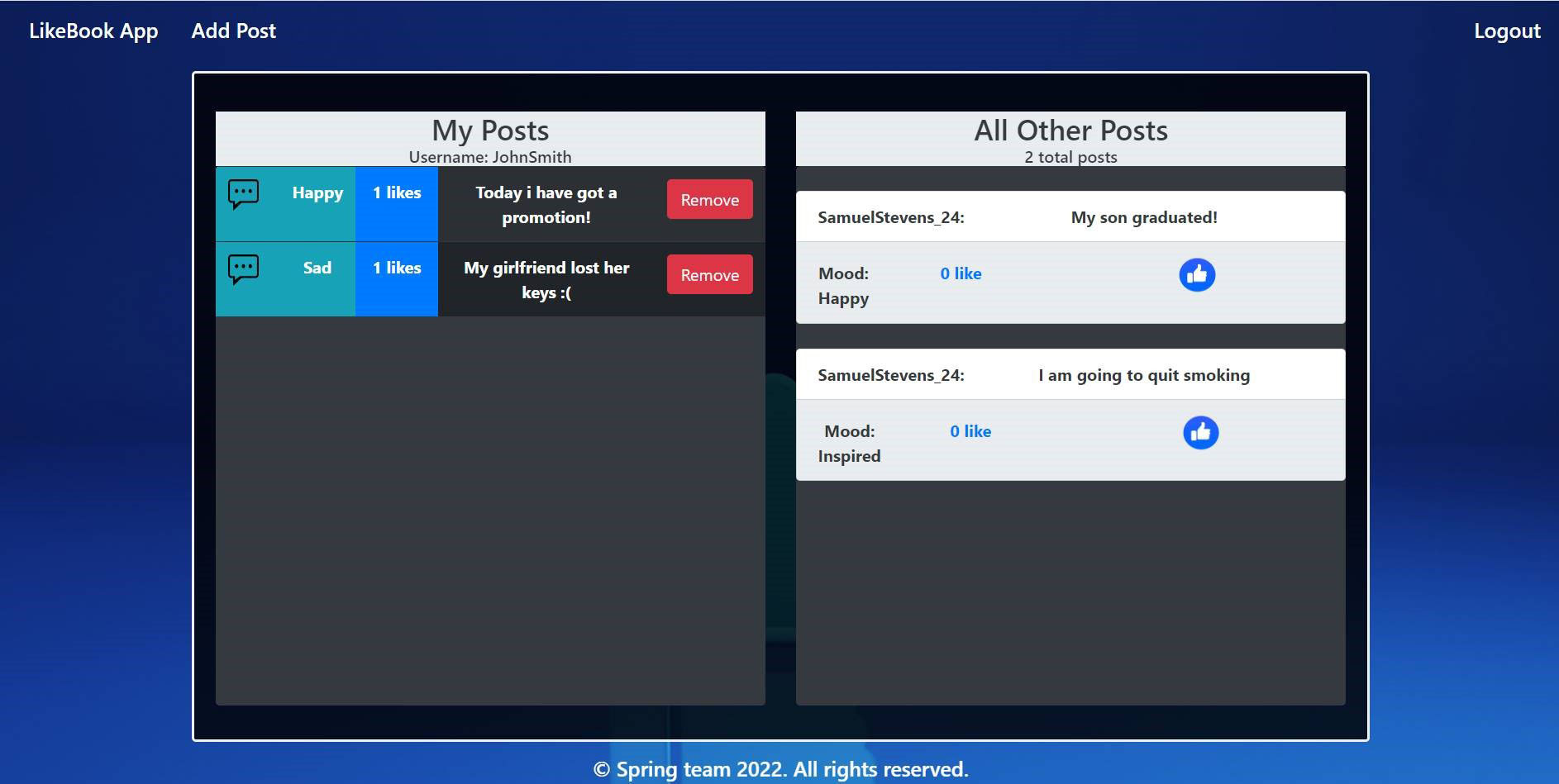
### Add post



### Add post validation



### Home Page (with posts)



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.

## 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 post (Add Post page), view all posts (Home page) and like other people’s posts from the All Other Posts section and remove their posts from the My Posts section.**

LikeBook App in the navbar should **redirect** to the appropriate URL depending on that if the **user is logged in**.

The application should provide functionality for **adding posts** with moods of **Happy, Sad and Inspired**.

The posts should be separated into different sections according to their creator.

When the user clicks on the **Like button of some post**, he likes the post. When he clicks on the **Remove** button, just delete the post from his **posts** in DB and redirect to the home page.

Bellow the All Other Posts banner is located an info bar that shows the **sum of all posts made by other users**.

The application should store its data in a MySQL database.

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

## Scoring

### Database – 10 points.

### Pages – 25 points.

### Functionality – 35 points.

### Security – 5 points.

### Validations – 15 points.

### Code Quality – 10 points.