# Spring Fundamentals Exam

# Planner Application

Exam for the ["Spring Fundamentals" course @ SoftUni](https://judge.softuni.org/Contests/3987/Spring-Fundamentals-Retake-Exam-13-Apr-2023).

**Planner App** by Softuni is designed to help individuals and teams manage their workflow efficiently. It provides a user-friendly interface with a customizable Kanban board that allows users to create and move cards representing their tasks or work items across different stages of their workflow.

The App offers features such as the ability to set due dates, add description or comments, and assign tasks. It incorporates different priorities of the tasks, such as "Urgent" "Important" and "Low". The application provides real-time collaboration, enabling team members to work together more efficiently and effectively.

There are several requirements you must follow in the implementation:

## Database Requirements

The **Database** of the **Reseller** 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 '@'.
* Has assignedTasks
  + The assignedTasks is a collection that contains users tasks that are assigned to him. One user may have many tasks and one task can be assigned to only one user.

### Task

* Has an **Id – “UUID-String” or Long**
* Has a Description (not null)
  + Description length must be between 2 and 50 characters (inclusive of 2 and 50).
* Has a dueDate (not null) - date
  + The dueDate must be a positive in the future.
* Has a Priority (not null)
  + One task has one priority and one priority can have many tasks.
* Has a User
  + The user who has assigned the task to him so he can work on the task. One task has one user, but one user may have many tasks.

### Priority

* Has an **Id** – **“UUID-String” or Long**
* Has a **Priority name** (unique, not null)
  + an option between (URGENT, IMPORTANT, LOW)
* Has a Description (not null)
  + For URGENT - "An urgent problem that blocks the system use until the issue is resolved."
  + For IMPORTANT – "A core functionality that your product is explicitly supposed to perform is compromised."
  + For LOW – "Should be fixed if time permits but can be postponed."
* Has collection of **Tasks**
  + One priority may **have many** tasks, but one task has **only one** priority.

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

## Initialize priorities

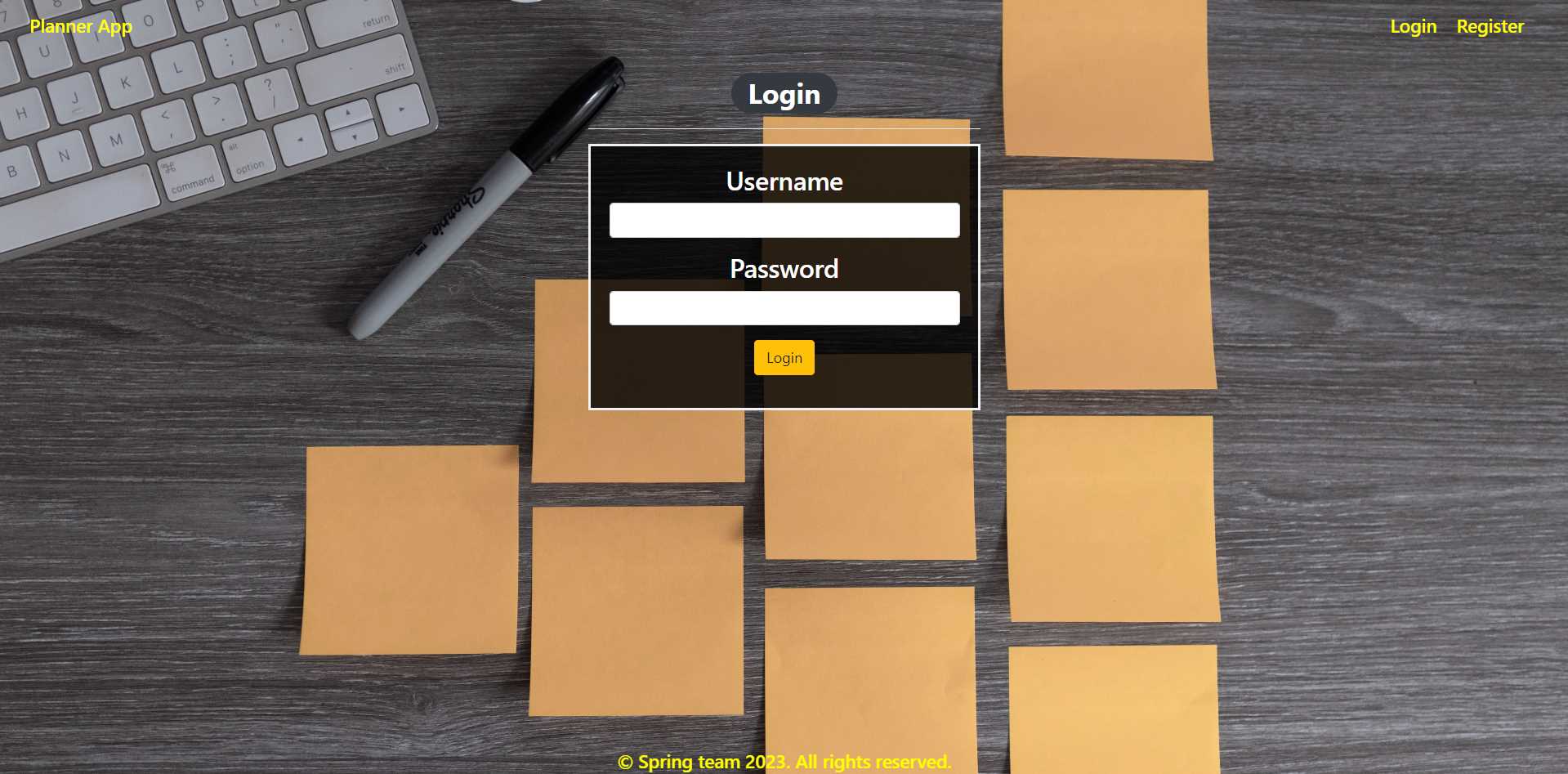
* Implement a method that checks (when the app started) if the database does not have **priorities** and   
  initialize them
  + **You are free to do it in different ways.**

## Page Requirements

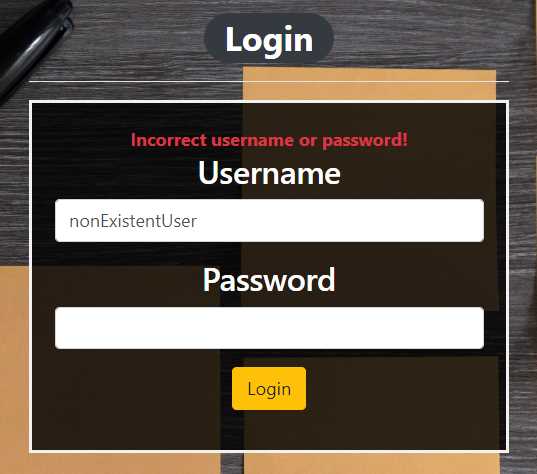
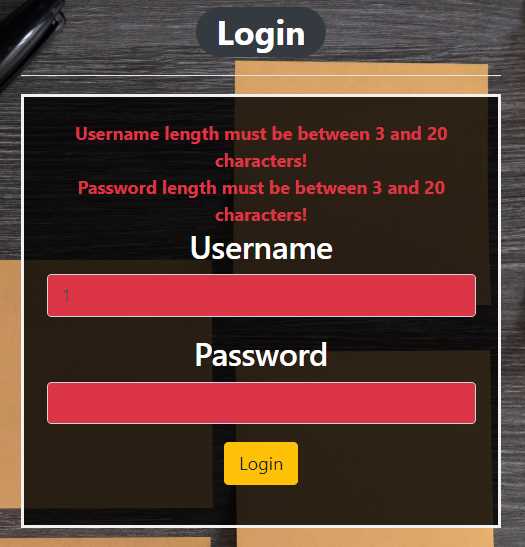
### Index Page (logged out user)



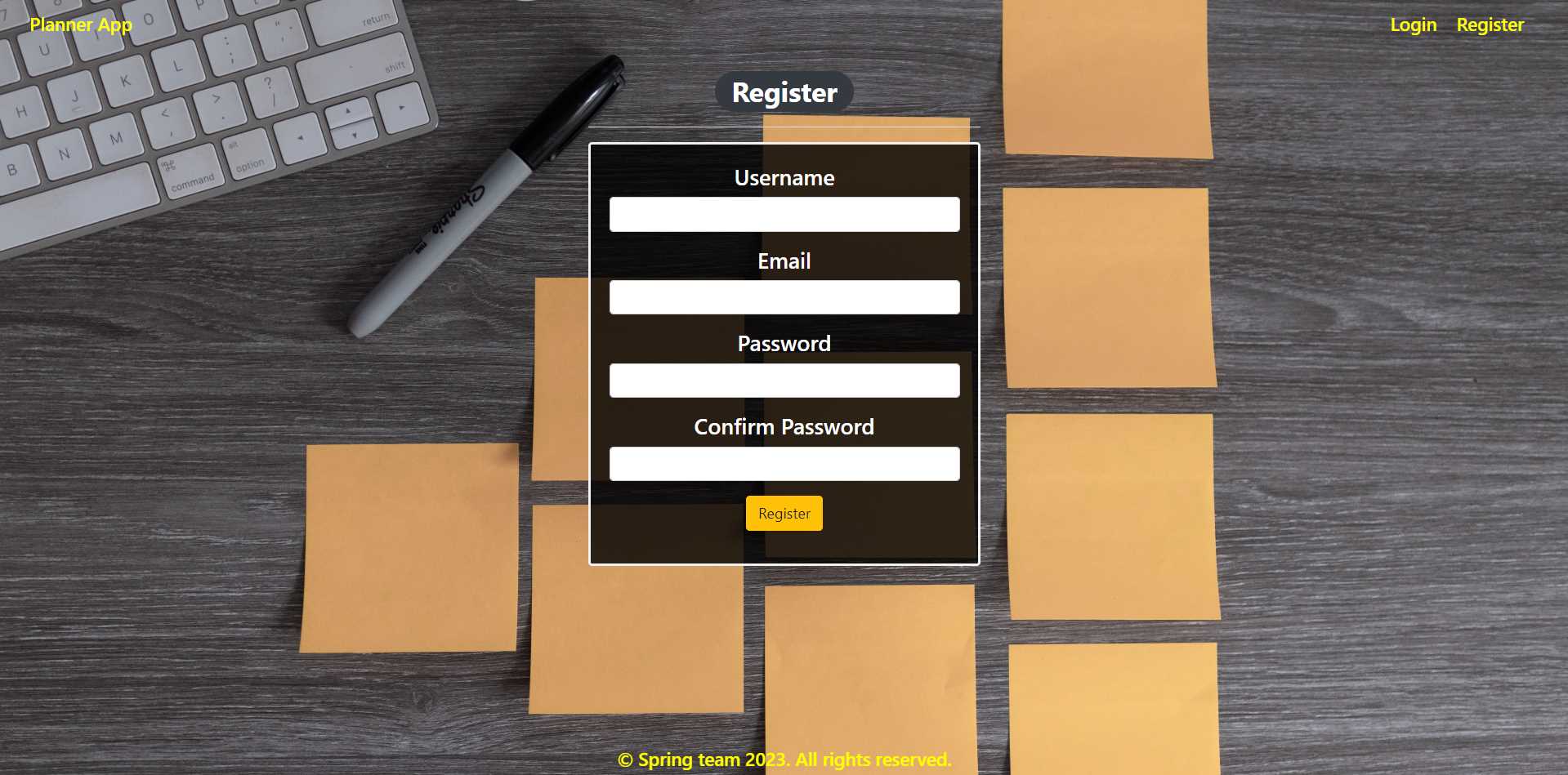
### Login Page (logged out user)



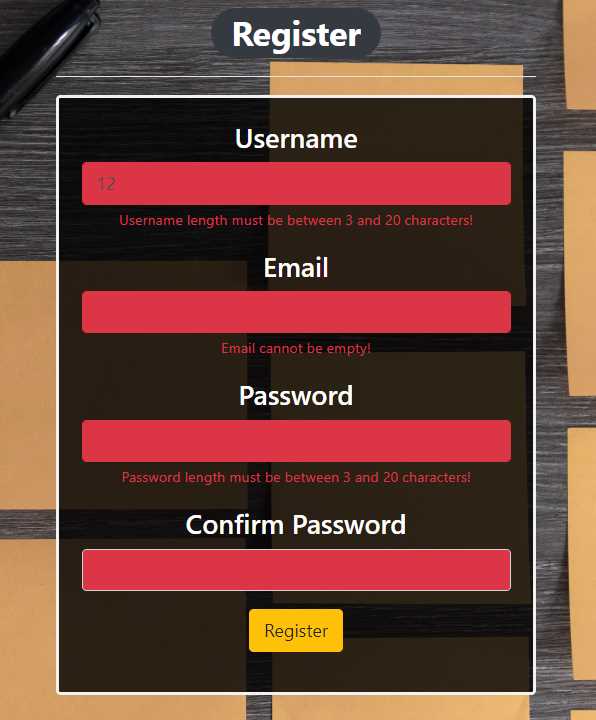
### Login Page validations

### Register Page (logged out user)

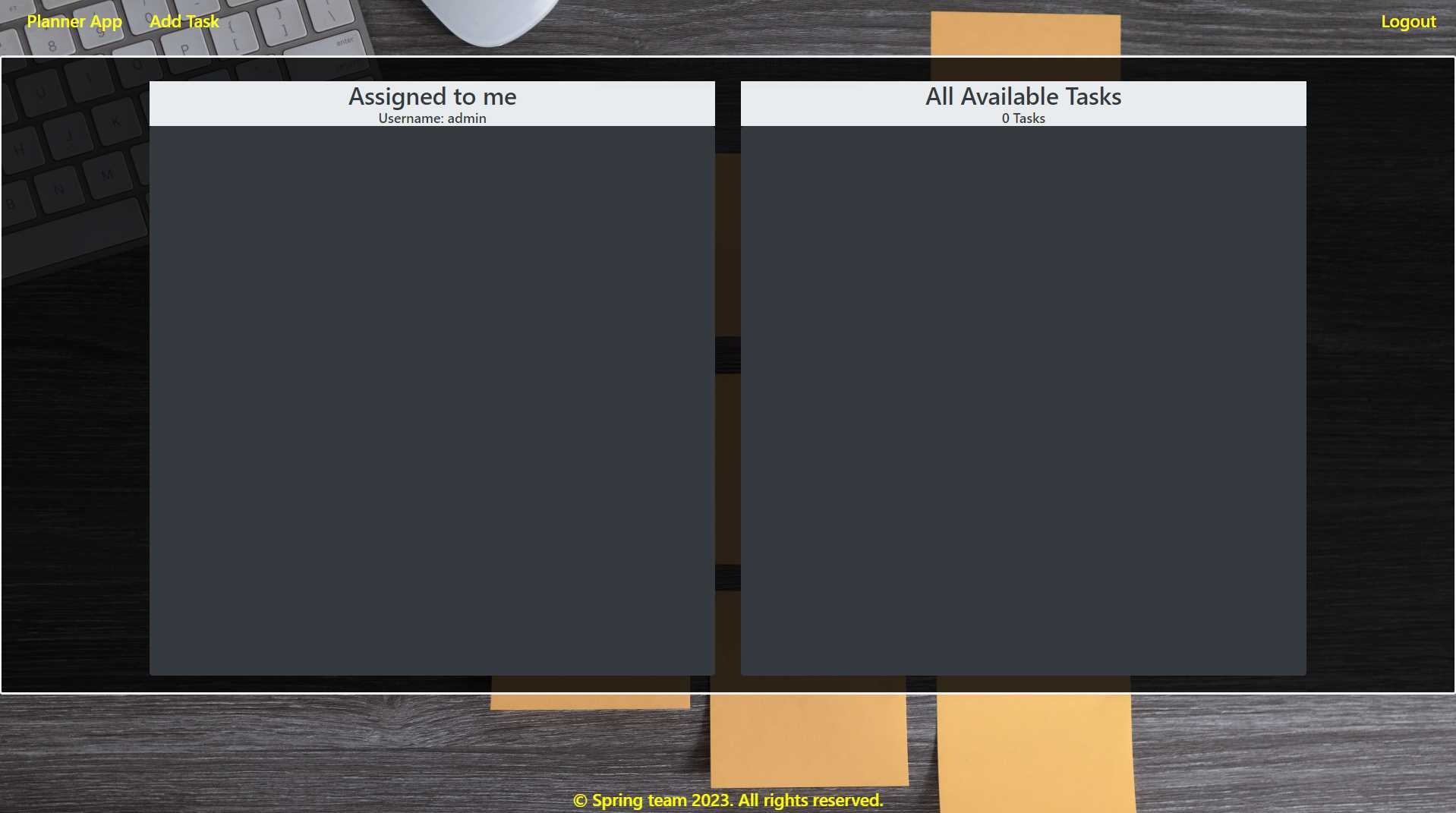


### Register Page validations

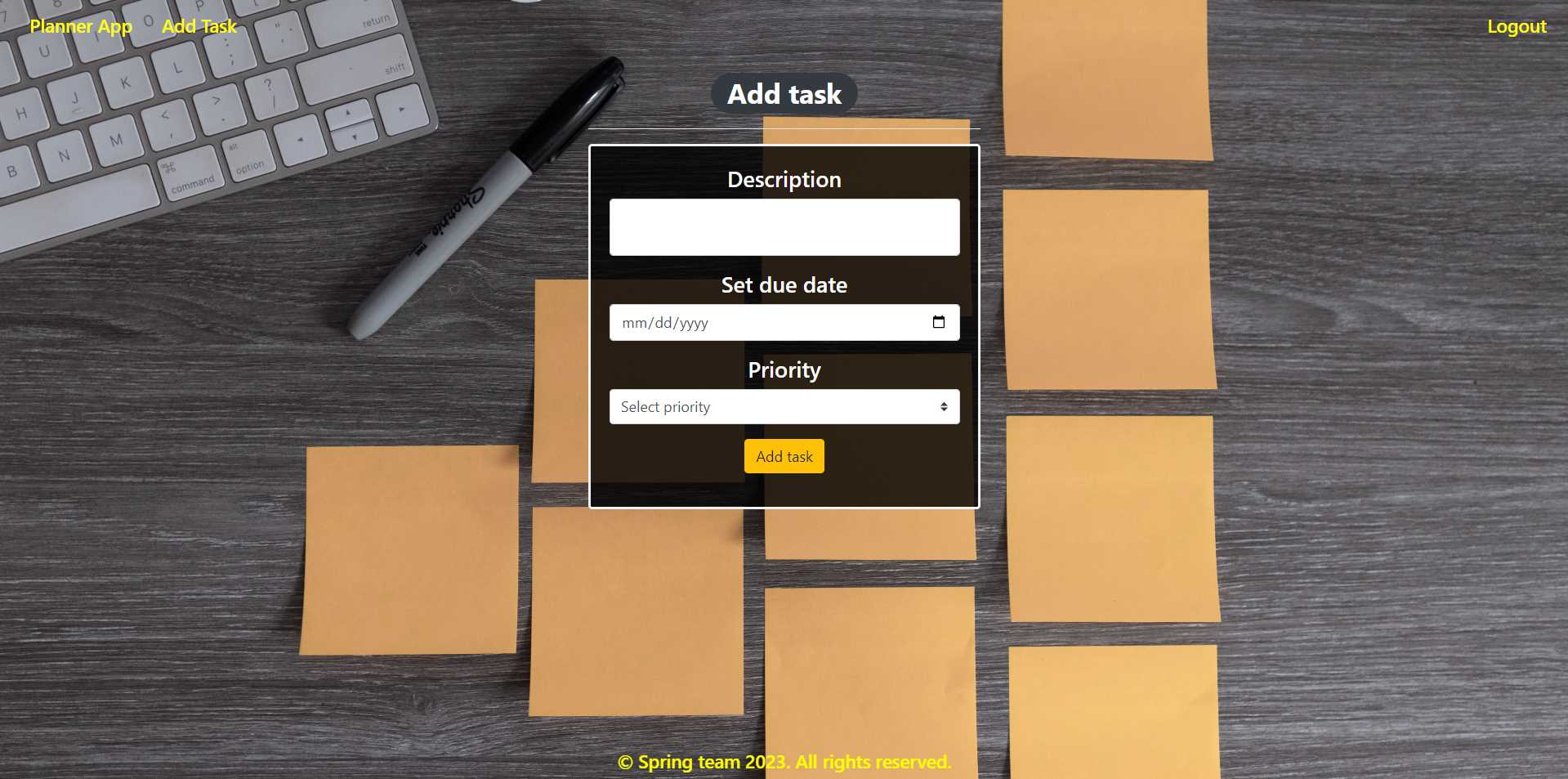


### Home Page (without having any tasks)

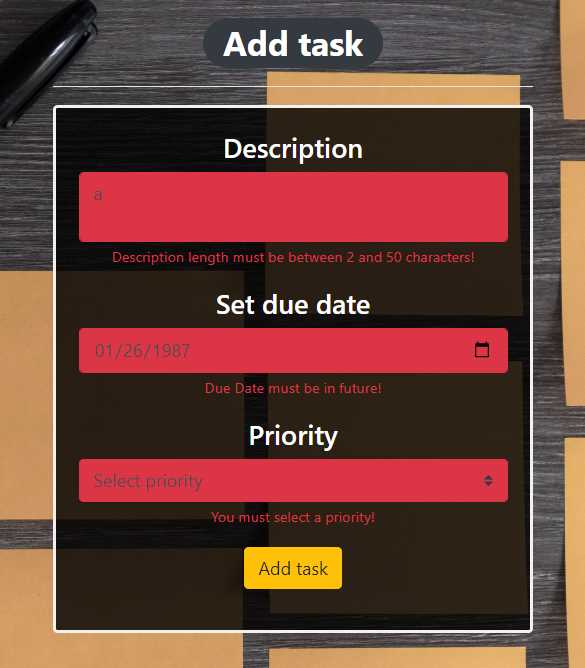
* Note: The left section (Assigned to me) of the page should visualize **all of the current logged user tasks** from the database which he assigned to himself.
* Note: The right section (All Available Tasks) of the page should visualize the available(not assigned) **tasks** (created tasks but not assigned).



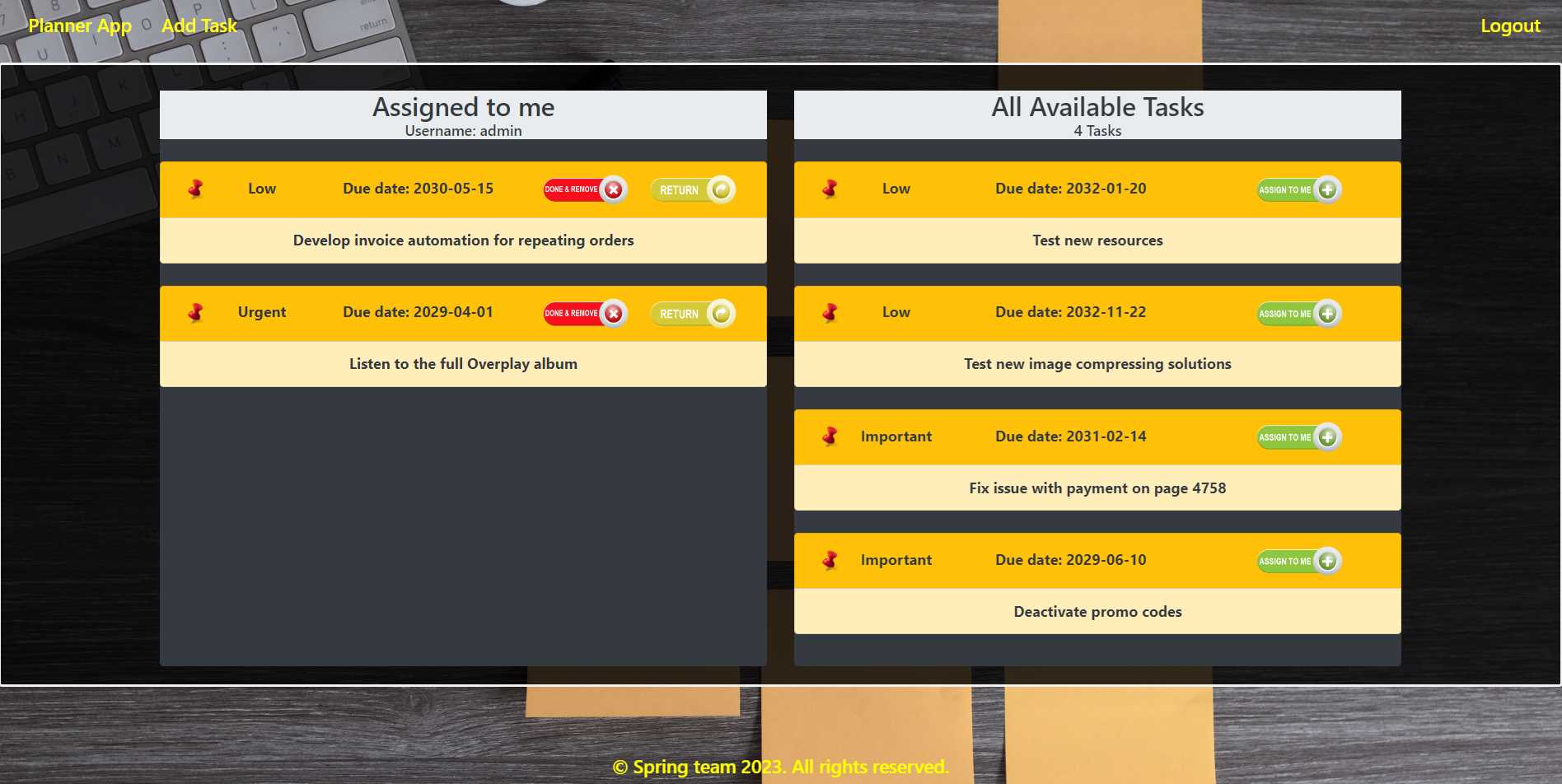
### Add task



### Add task validation



### Home Page (with tasks)



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 task (Add Task page), view all tasks (Home page) and assign a task from "All Available Tasks" to "Assigned to me".**

Planner 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 tasks** with priorities of **Urgent, Important or Low**.

The tasks should be separated into "Assigned to me" and "All Available Tasks".

When the user clicks on the **ASSIGN TO ME button of some task**, he is assigning the tasks to **his collection(Assigned to me)**. You should **not delete** this task from DB. The task should be removed from the collection with **All Available Tasks** and should be added to the collection with **assignedTasks** of the user.

The **Done & Remove** button in "Assigned to me" section **should remove the task** from DB.

By using the **Return** button in "Assigned to me" section the user is abandoning the task and is returning it to the **All Available Tasks.** The user should unassign this task from his collection of **assignedTasks** and the **task** should no longer have **user.**

Bellow the "All Available Tasks" banner is located an info bar that shows the **sum of the** **unassigned** **tasks in this field**.

Bellow the "Assigned to me" banner is located an info bar that shows the **username of the current logged in user**.

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 Task page.
* Users (logged in) can access Logout functionality.

## Scoring

### Database – 10 points.

### Pages – 25 points.

### Functionality – 35 points.

### Security – 5 points.

### Validations – 15 points.

### Code Quality – 10 points.