Java Web Development Basics

Exam

Heroes

Exam for the <u>"Java Web Development Basics" course @ SoftUni</u>. Submit your solutions on the course page, so that you can be evaluated by your fellow colleagues.

Heroes is a web application . You have been tasked to implement this application for an unusually low price, by an unusually rich client. There are several requirements you must follow in the implementation.

1. Database Requirements

The **Database** of the **Heroes** application needs to support **2 entities**:

User

- Has an **Id – UUID-string**
- Has a **Username**
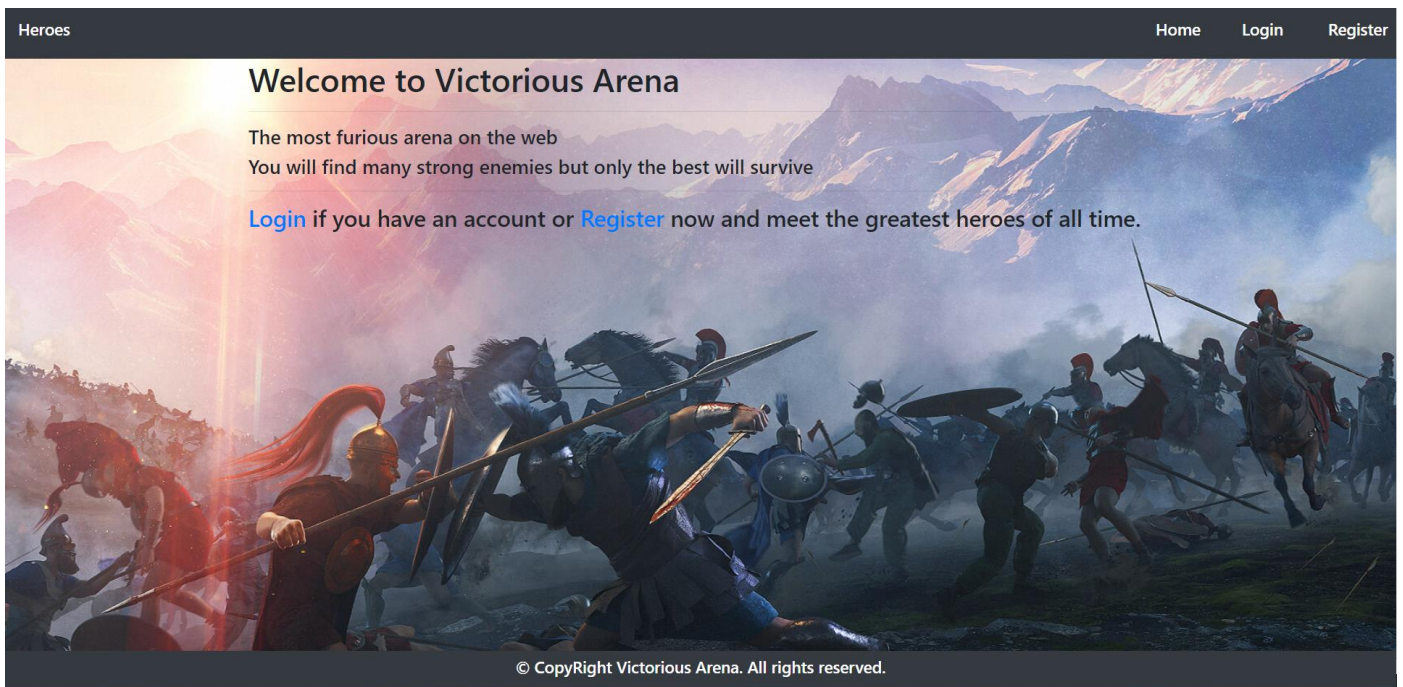- Has a **Password**
- Has an **Email**
- Has a **Country**

Hero

- Has an **Id – UUID-string**
- Has a **Name**
- Has a **Class – option between (Warrior, Archer, Mage)**
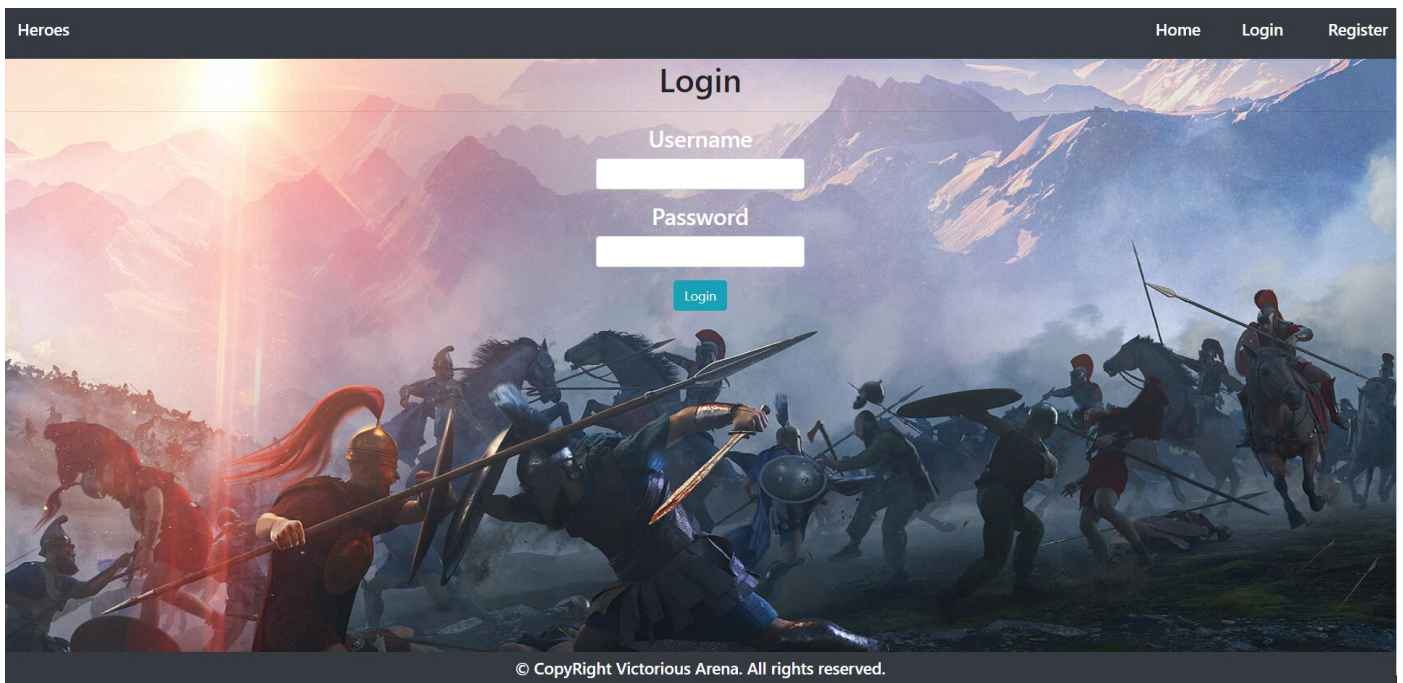- Has a **Level**

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

Page Requirements

Index Page (logged out user)

Login Page (logged out user)

Register Page (logged out user)

| Heroes | | Home | Login | Register |

Register

Username

Email

Password

Confirm Password

Country

[Register]

Home Page (logged in user)

| Heroes | | Home | Create Hero | Logout |
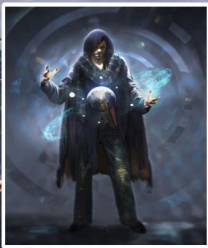
Welcome to the arena, $(username)

SoulTaker
[Details]
[Delete]

MindControl
[Details]
[Delete]

Devil
[Details]
[Delete]
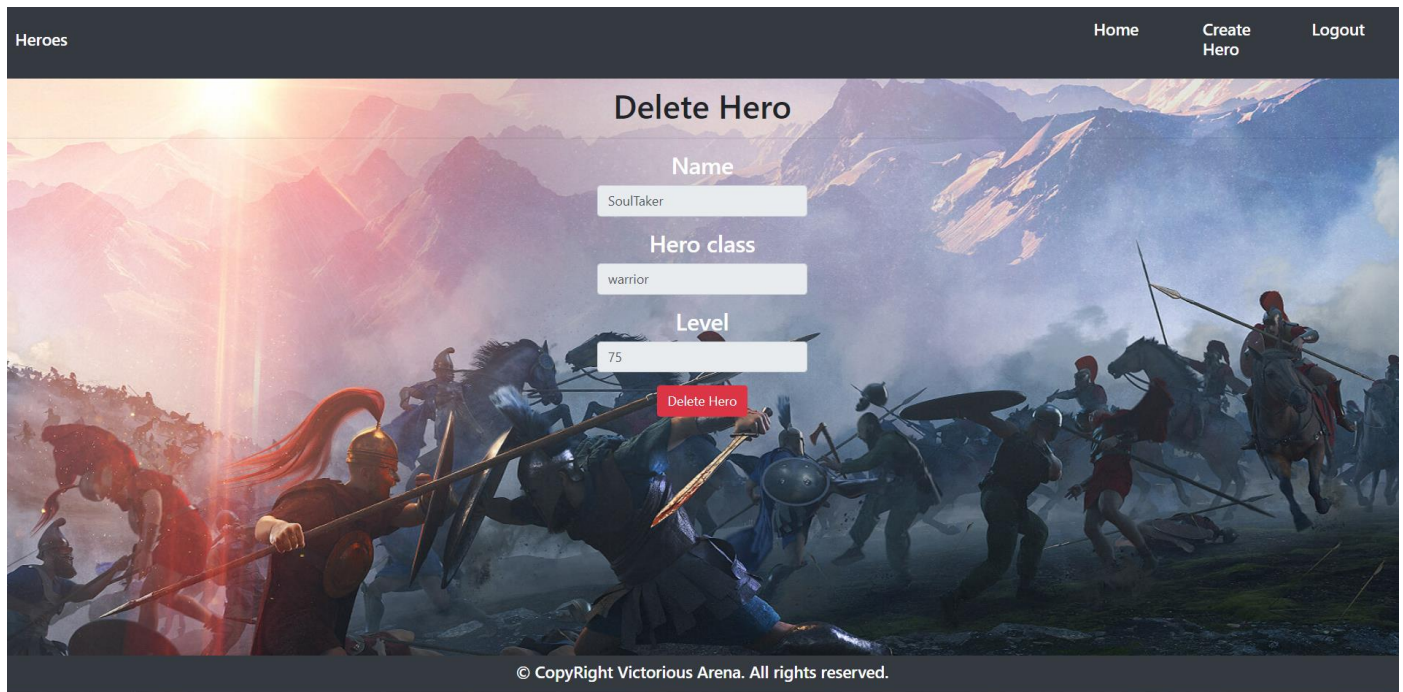
Add Hero (logged in user)

Hero Details Page (logged in user)

Hero Delete Page (logged in user)

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 `Functional 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`, **create** a **Hero**, **view details** about a **Hero**, **delete** a **Hero**, and `view all Heroes ordered buy level descending`(`Home page`). When you are at **Home Pag**e you should show "Welcome to the arena, " + current user, **username**.

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

The **application** should provide **functionality** for **adding heroes** with **only 3 classes** – "`Warrior`", "`Archer`" and "`Mage`".

The **application** should **store** its **data** into a **MySQL** database, using **Hibernate** native.

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 `Hero Details` page.

Follow us:

- **Users** (logged in) can access **Hero Delete** page.
- **Users** (logged in) can access **Add Hero** page.
- **Users** (logged in) can access **Logout** functionality.

Scoring

Database – 10 points.

Pages – 30 points.

Functionality – 40 points.

Security – 10 points.

Code Quality – 10 points.