# Exercises: The Right Way - Architecture

Problems for exercises and homework for the [“Java MVC Frameworks - Spring” course @ SoftUni](https://softuni.bg/trainings/1538/java-mvc-frameworks-spring-march-2017).

# Eventures

**Eventures** **Inc**. is a fast-rising newly made Start-Up Company, which specializes in **Event Tickets Sales**. It is said to be the killer of systems like Eventim, Eventbride, etc.

You have been appointed as the developer of the **main web application**. This is a great responsibility, so do your best and do not dissapoint your employers. The application functionality is not that complex, and it will be **separated** into **several parts**, each part consisting of **several tasks**.

Your current task is to create the **architecture** and **core logic** of the **application**, so get started.

## Data Storage

The core application logic requires **3 data models** to be implemented:

### User

Has the following properties:

* Username – a String.
* Password – a String.
* Email – a String.
* First Name – a String.
* Last Name – a String.
* Unique Citizen Number (UCN) – a String.

### Event

Has the following properties:

* Name – a String.
* Place – a String.
* Start – a LocalDateTime object.
* End – a LocalDateTime object.
* Total Tickets – an integer.
* Price Per Ticket – a **decimal** value.

### Order

Has the following properties:

* Ordered On – a LocalDateTime object.
* Event – an Event object.
* Customer – a User object.
* Tickets Count – an integer.

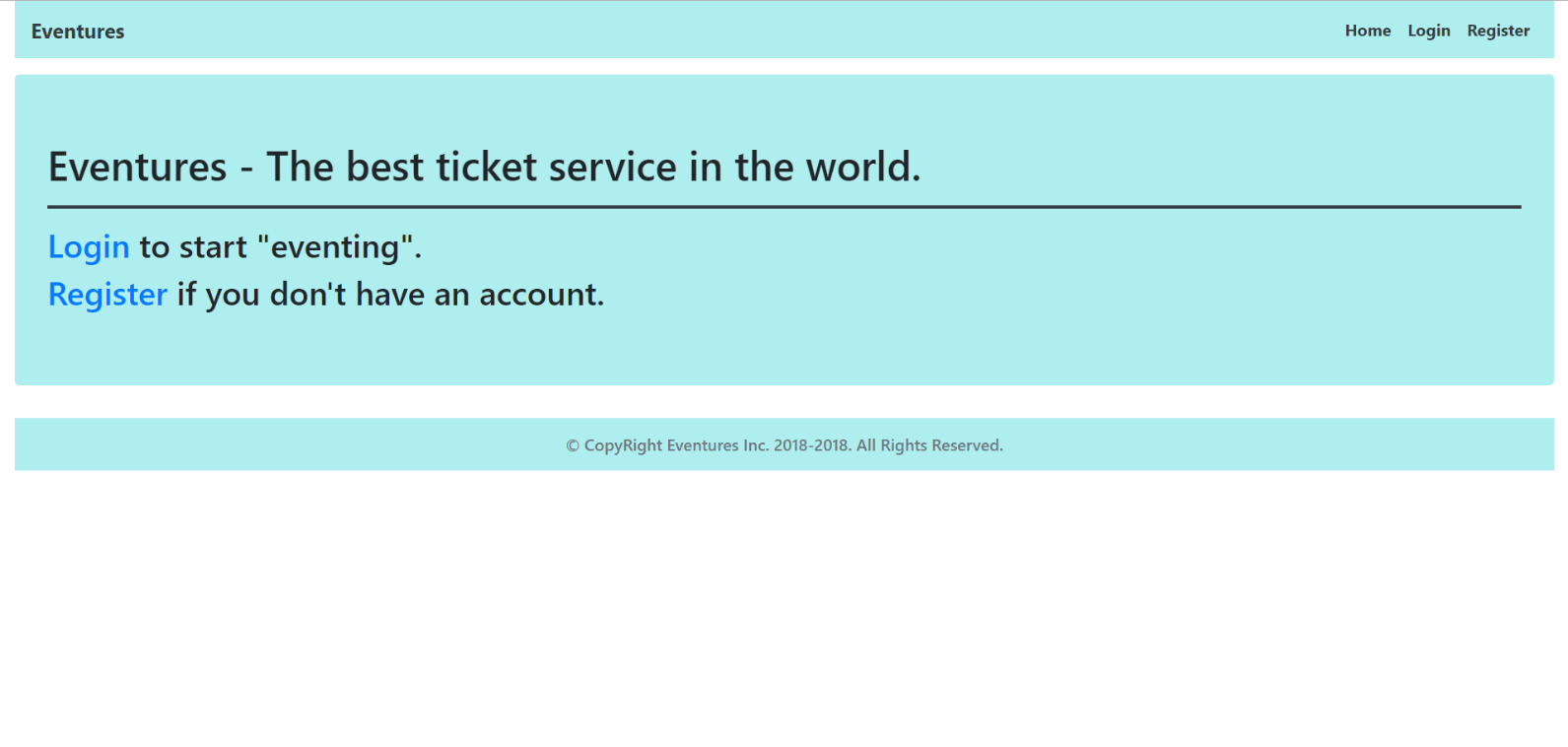
Each of the **data models**, also has an **Id**, which should be a **UUID**.

## Front-End

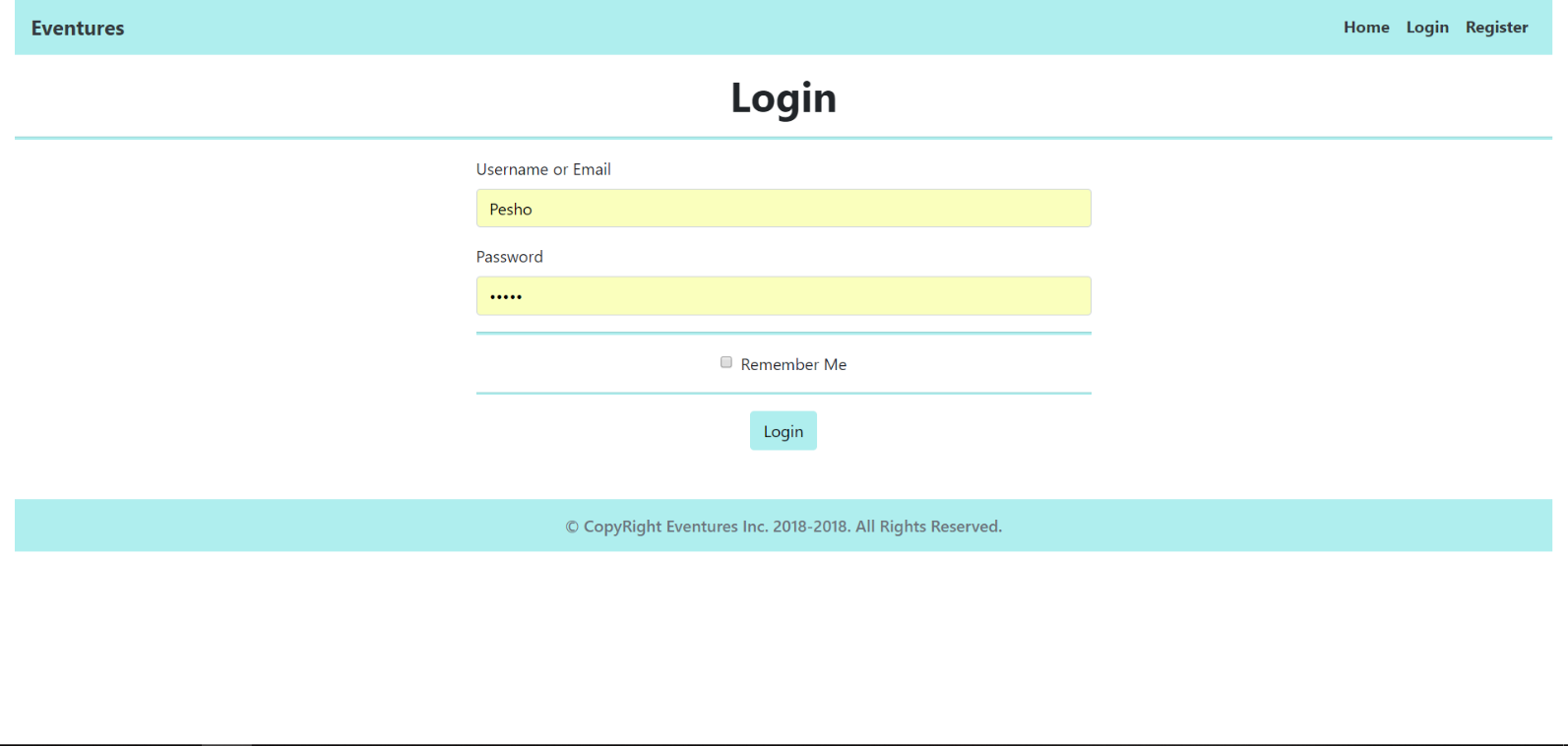
The Front-End might be a little different in some small components than the things you’ve done so far. However, do not panic, it is not that hard to do it. You can do it! 😉

There are a few templates you must implement, they are not that much, but they are tricky.

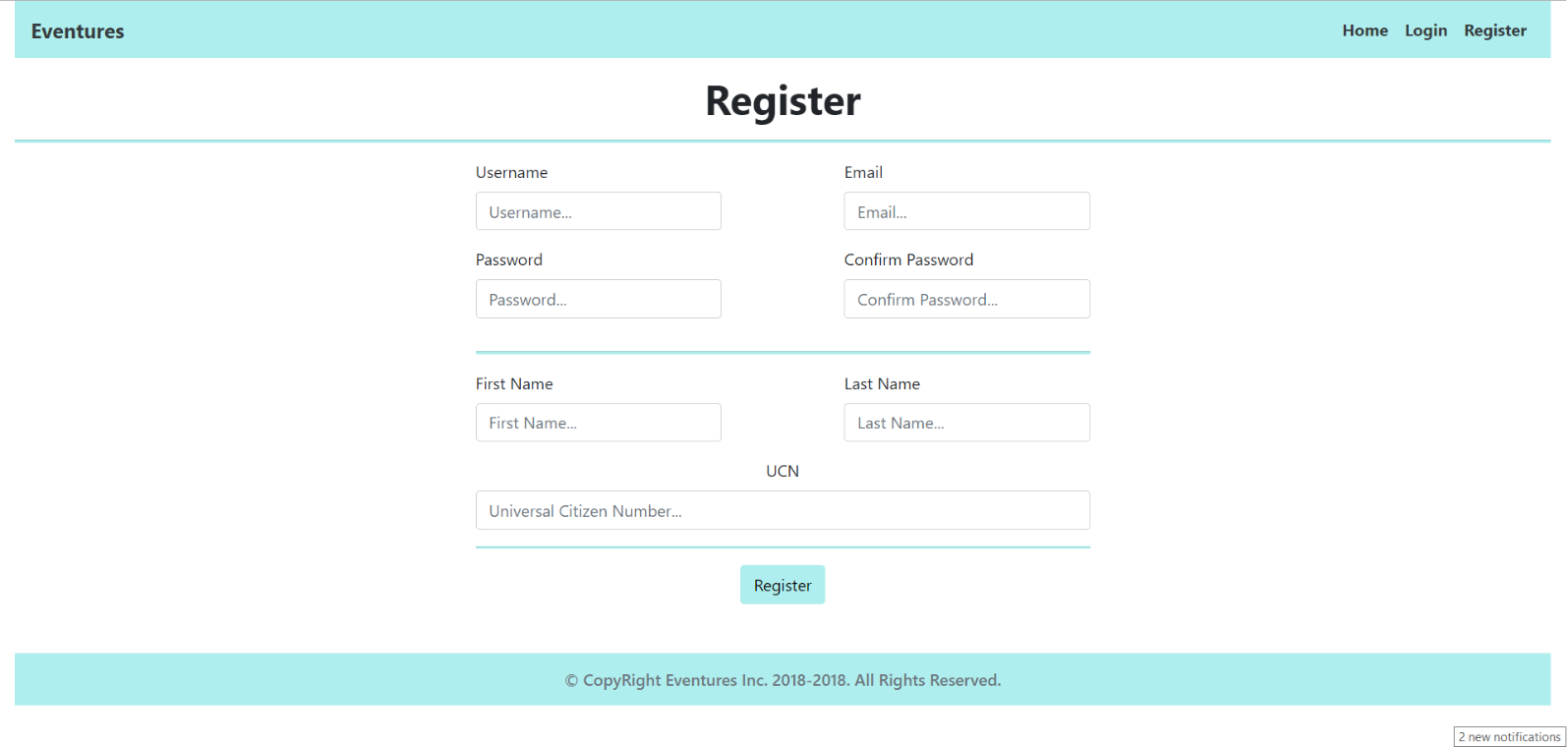
### Index Template



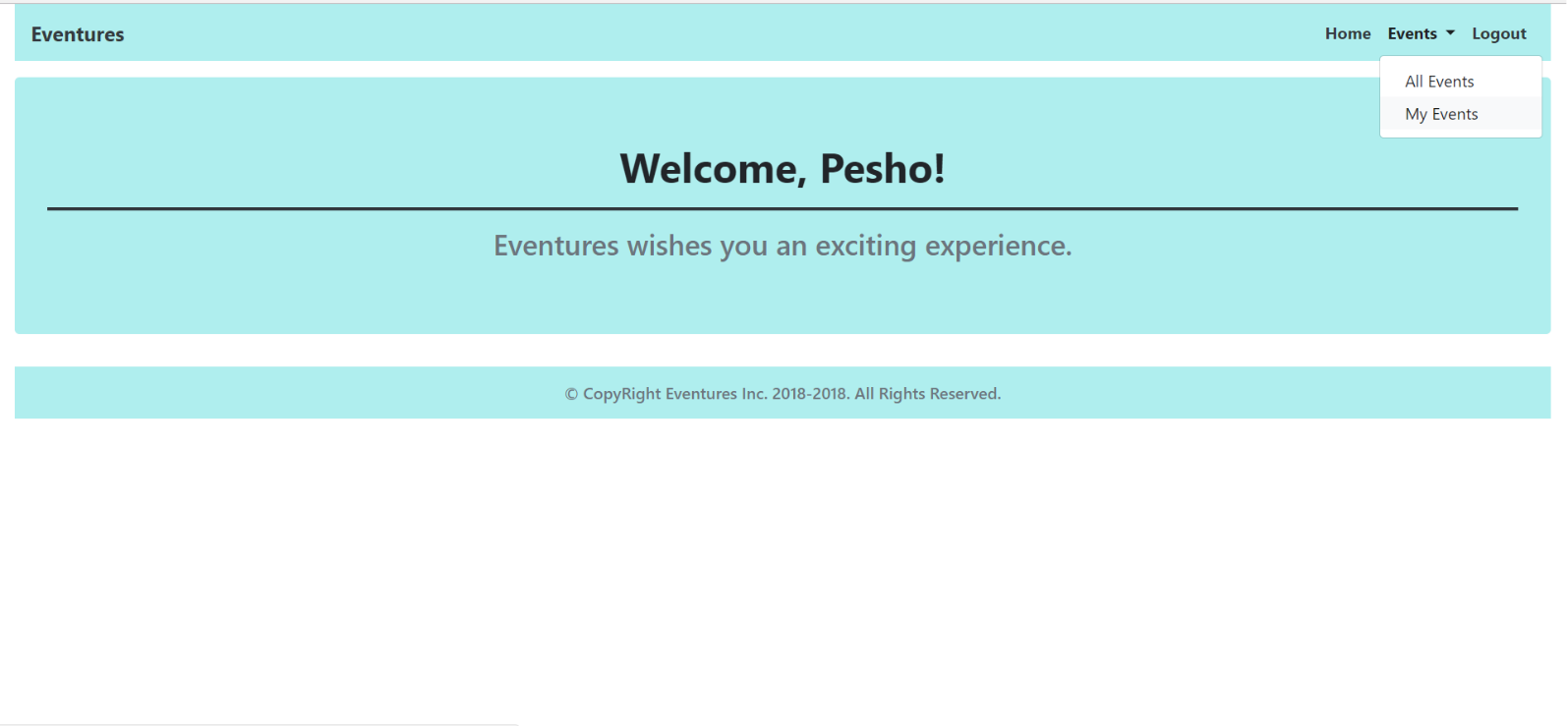
### Login Template



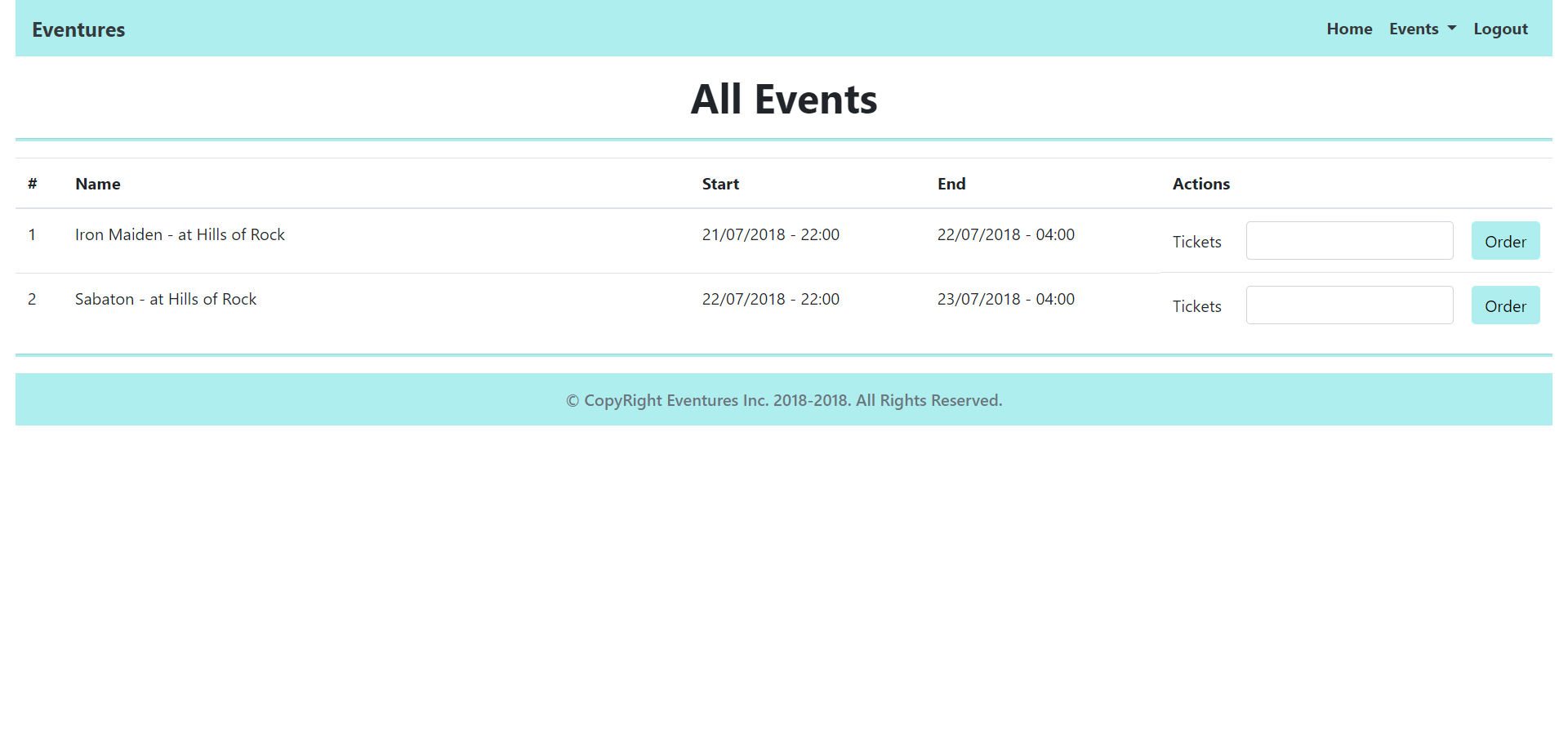
### Register Template



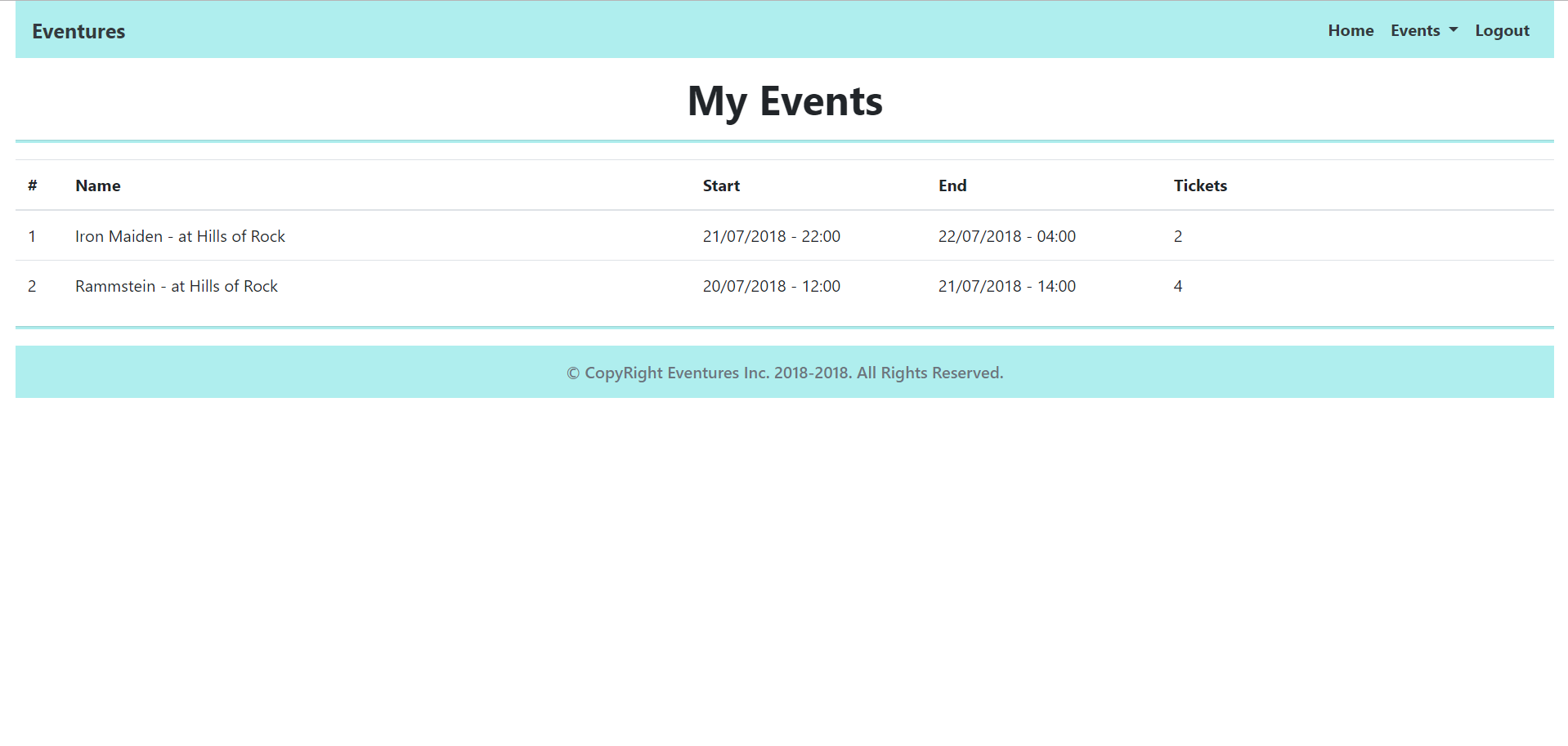
### Home (logged-in users)



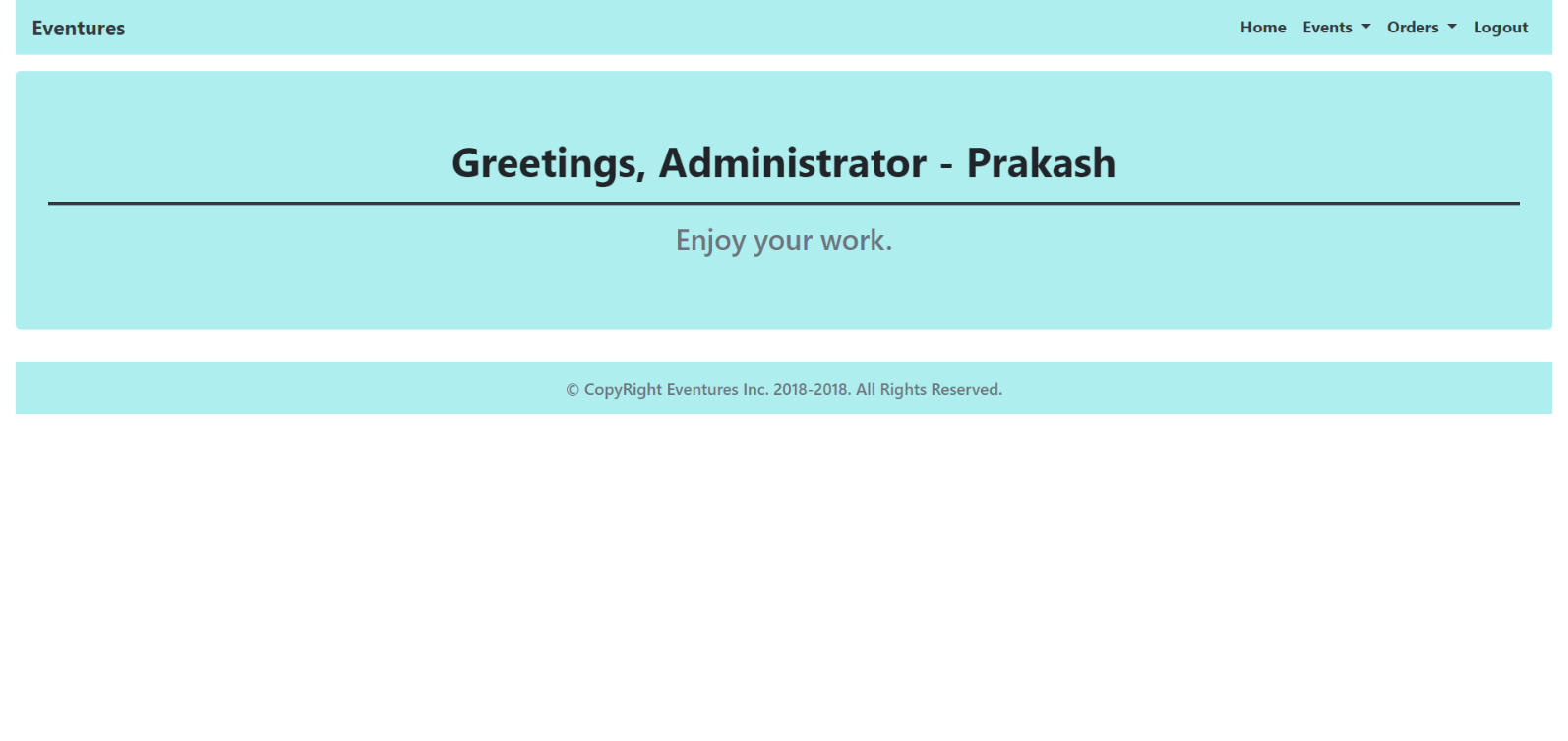
### All Events (logged-in)



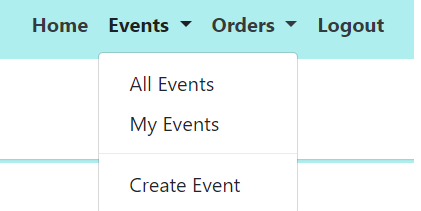
### My Events (logged-in)



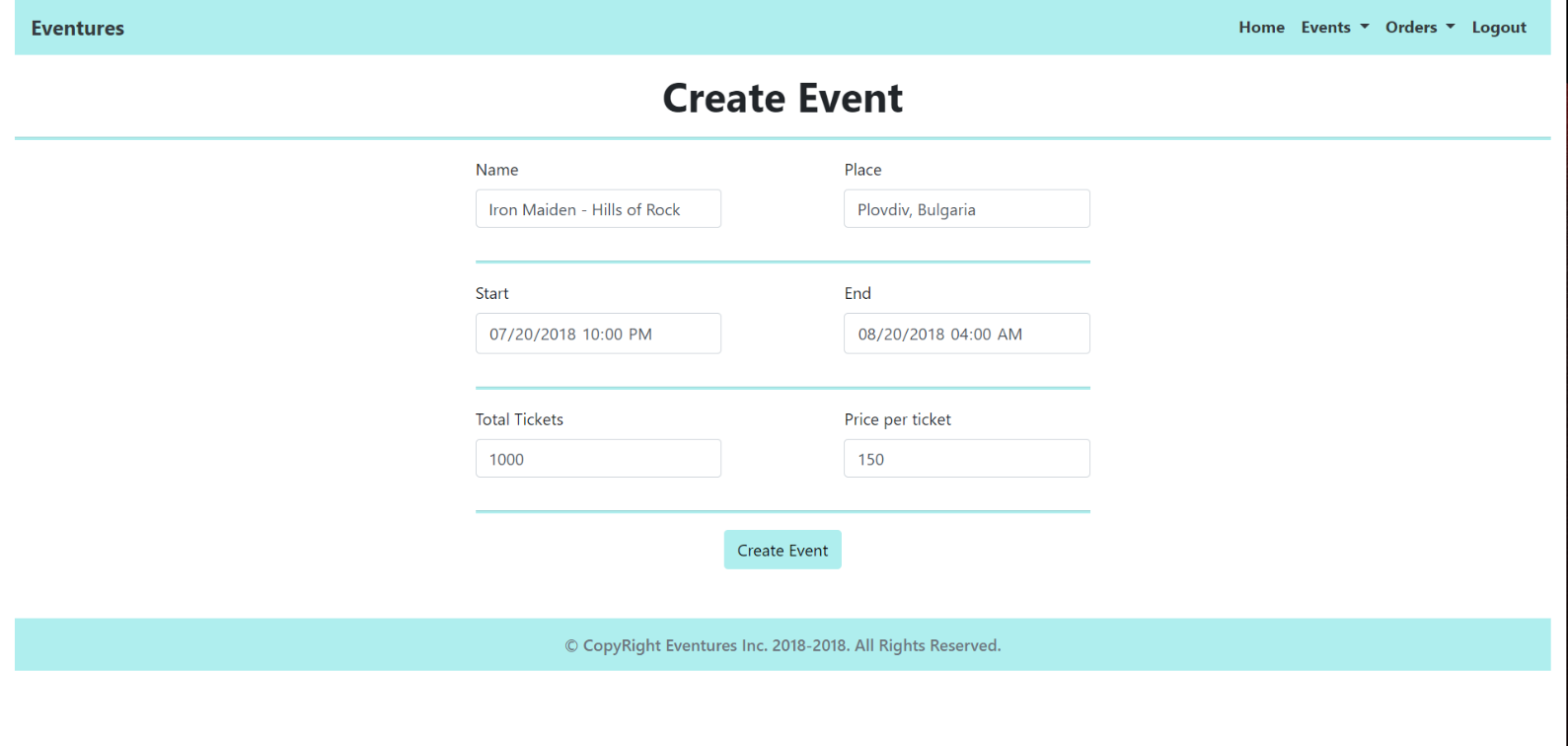
### Admin Home (logged-in Admin)



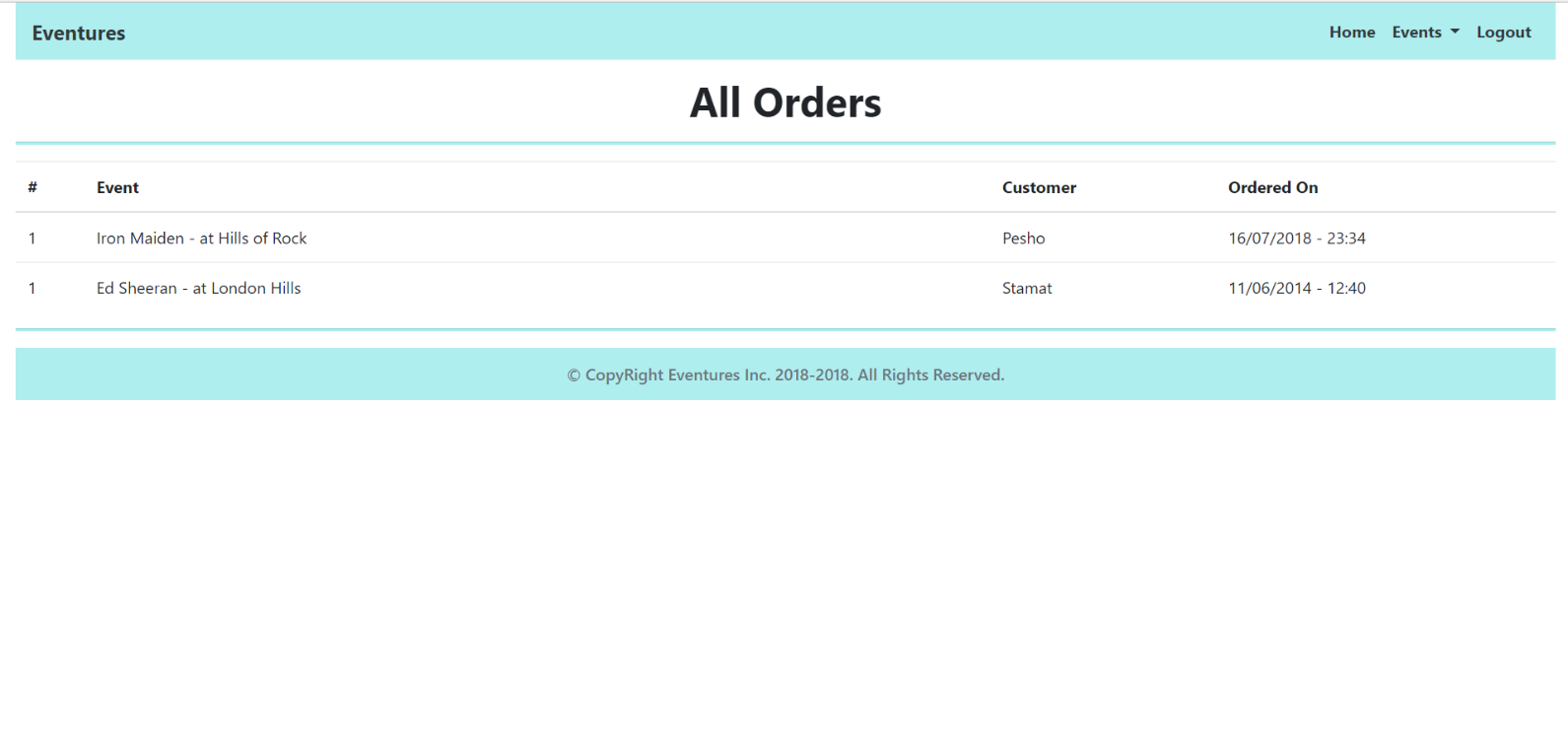
#### Admin Navigation



### Create Events



### All Orders



**Note**: The color for the application is #AFEEEE.

## Business Logic

### Technical Requirements

The application should be a **Spring Boot** app. As such it should use **the most** of the **Spring Framework**.

Use **Spring Boot** for application **development** and **deployment**.

Use **Spring Security** for authentication. Add the following **roles** to your User functionality – (‘USER’, ‘ADMIN’).

Use **Sprint Boot’s JPA Starter** for Persistence.

Use **Thymeleaf** for template development.

### Functionality

The application should provide its Guest users (**not logged-in**) the functionality to register and login.

The application should provide its Regular users (**logged-in** Users with Role – User) the functionality to **view all** Events, **order** **tickets** for them, and **view** all Events, **they’ve** ordered **tickets** for (**My Events**).

The application should provide its Admin users (**logged-in** Users with Role – Admin) the functionality to **create** **new** Events, **view all** Events, **order** **tickets** for them, **view** all Events, **they’ve** ordered **tickets** for (**My Events**), and **view** all Orders made.

### Messaging

Upon **clicking** the[Order] button, you should send a **Message** through JMS, containing the **order** data.

There should be a Listener, which is subscribed to that **message**, which **creates** and **persists** an Order, with the **order data**.