# Workshop: Eventures Part Two

Workshop description for the [“ASP.NET Core” course @ SoftUni](https://softuni.bg/courses/asp-net-mvc).

### 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 extend the **business logic** of the **application**, by adding some convenient features. Also, add **social login**, **auto mapping** etc.

## Pagination

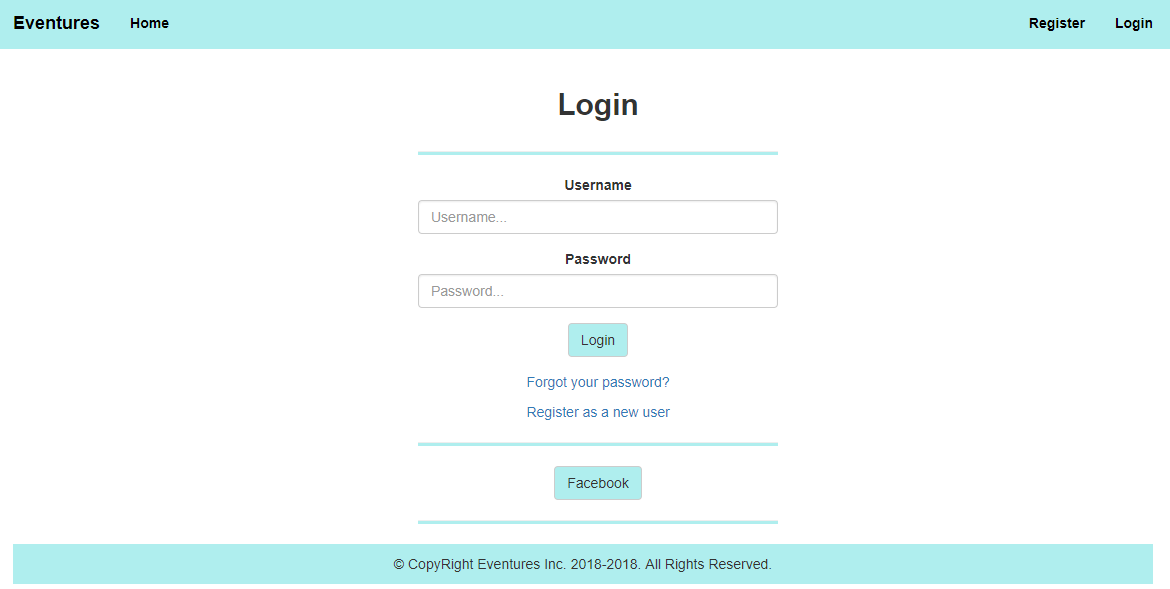
Implement a pagination for the Events page. This is a bit complex task, but you should be able to manage it. The pagination can be indexed, or just **Next / Previous** type. The element count on a page is up to your choice. You will also have to create quite a number of Events in order to test it.

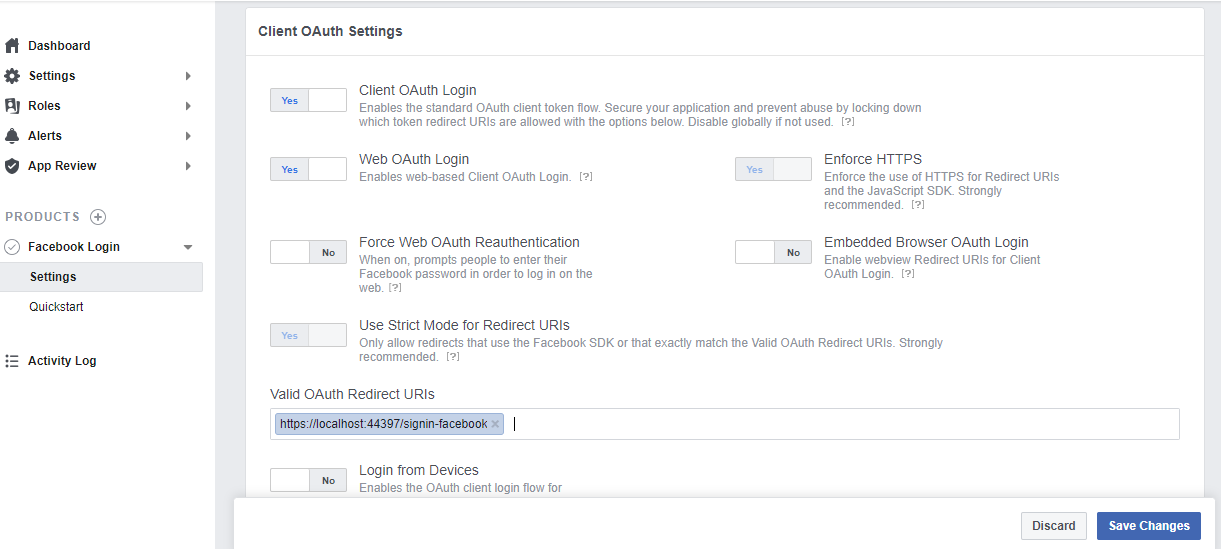
## User Administration

Implement a simple page for **User administration**. Where **Administrators** can **promote** **Users** to **Administrators**, or the can **demote** **Administrators** to **Users**. The page should represent a simple list, it can be a bootstrap table, with a simple button [**Promote**], [**Demote**]. Include a link to the page on the **Admin navbar**.

## Social Accounts

Add functionality to **login with Facebook**.





**NOTE:** Go to the Facebook Login product Settings and make sure to add the following redirect URIs in Client OAuth Settings. By default **ASP.NET Core** is using the /signin-facebook redirect URI. Be sure to add for **HTTPS** as well, because **the ASP.NET Core 2.1 application is running in HTTPS**.

**HINT:** Check the code in the default Login.cshtml and customize it.

## AutoMapper

Replace all your manual object mappings to use AutoMapper. Everywhere in your project.

## Available Events

As you know, the Events have a **total ticket count**. Last time we implemented Orders. But these Orders weren’t fully functional. Here’s a little change to that functionality.

Each time a certain amount of **tickets** is **ordered**, the **ticket count** of the corresponding Event, should be **reduced** by the Order’s **ticket amount**. When an Event’s **tickets** are **reduced** to **0**, it should **not** be **visualized** on the All Events page (it is no longer available).

**Note**: If an Event has **5 tickets**, and an Order with **10 tickets** is attempted, it should throw an **error**. The **error** is up to you.