# Workshop: Blog

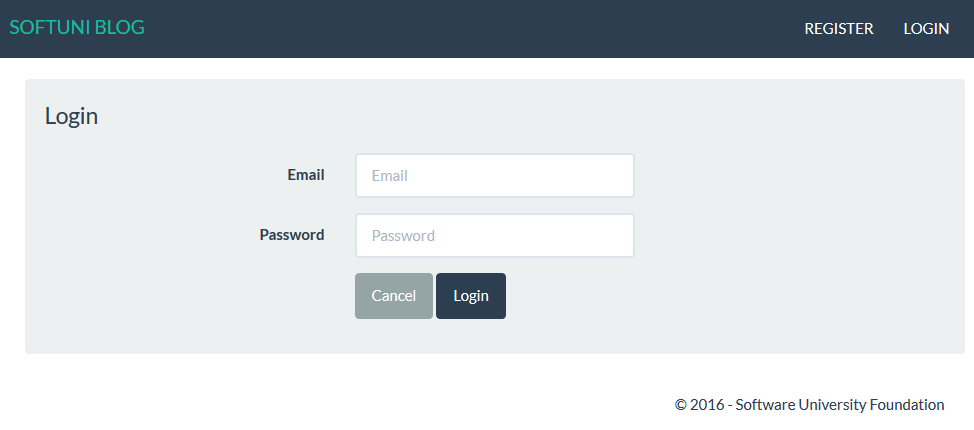
This document defines a complete walkthrough of creating a **Blog** application with the custom MVC Framework and TomEE application server.

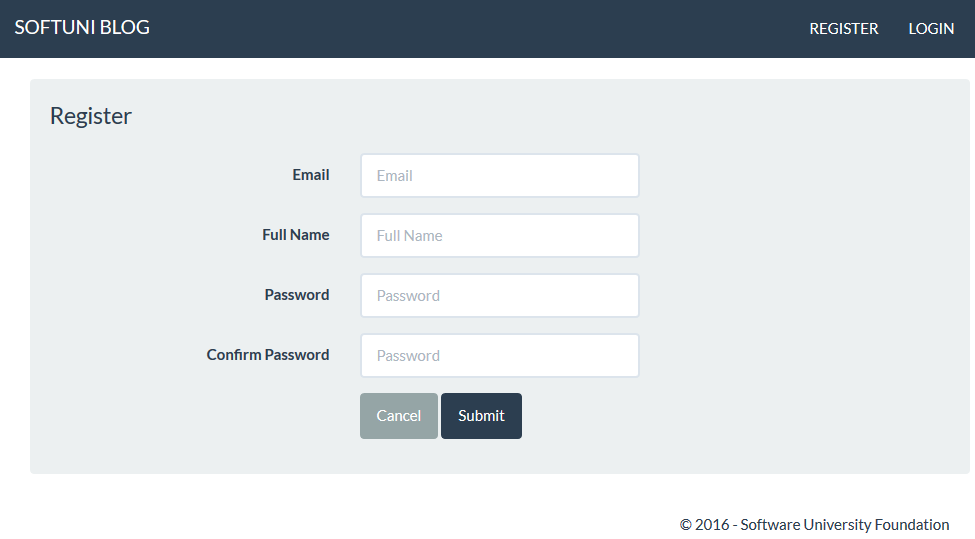
# What Will You Create

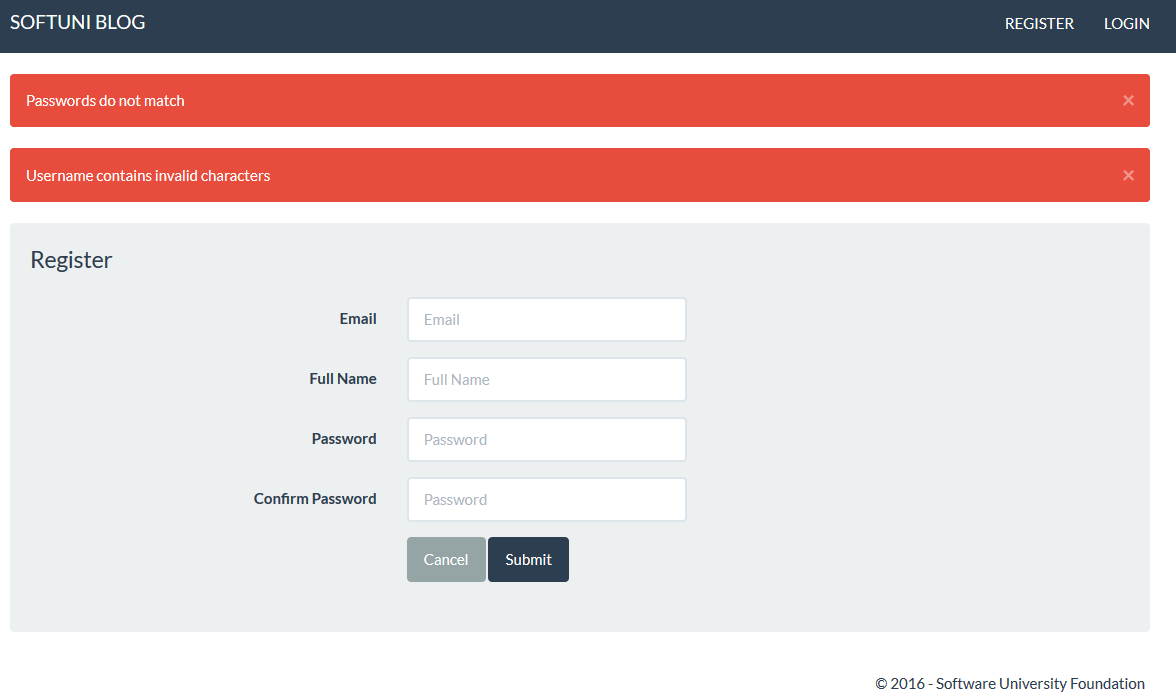
In the end of the whole tutorial, you should have blog which supports the following functionality:

* **Register** and **Login Users**
* **Create**, **Update**, **Read** and **Delete** **Articles**
* **Responsive Design** using **Bootstrap**
* **More…**

Pictures:







# Creating Register and Login

Today in the **first part**, you will only create the **user registration** and **login**.

## User Register

You are given a skeleton, containing templating system. There are TODO tags in some of the files, so you better get familiar with the project. You are given an example form called register.jsp for the user registration. Create a new UserController. You need to implement the methods for GET and POST request for user registration. You need to create connection with Hibernate and create Model, Repository and eventually a Service layer. The **user model** should contain the following fields:

* Id – Identity
* Full name – min length – 5, max length – 50 and, only containing English alphabet characters
* Email – unique, not null
* Password – not null, min length – 3, max length – 30

When registering a new user, you should have the following **validations**:

* The Password and the Confirm-Password values should be equal
* The Length of the password should be correct
* The Email must contain "**@**" and "**.**"
* The Name should have the correct length

If there are any errors in the form, they should be passed to the view. The notifications are handled in the error.jsp file, so you only need to **include it in your view**. It's **important** that your error collection is called errors.

## User Login

Using the register form as a template, **create a login form**, that looks like in the pictures above. You should create the routings **handling** the **GET** and **POST** **requests**. On successful login, you should **save the user id** in the **session storage**.

## Changing the Navigation Bar

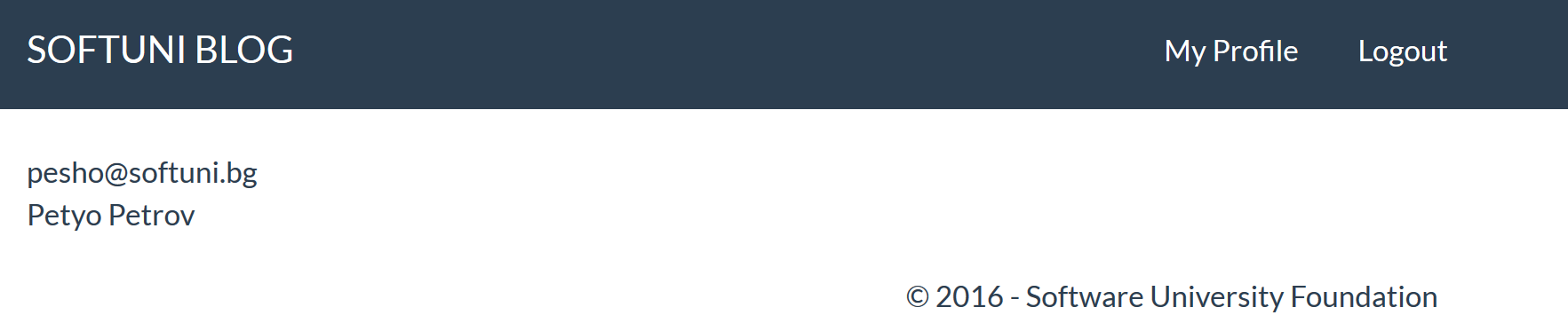
Implement a way to change the buttons in the nav-bar, based on the user authentication. If the user is guest, he should see the register and login buttons:



If he is logged in, he should see buttons to **logout** and see his **profile**:



## User Profile Page



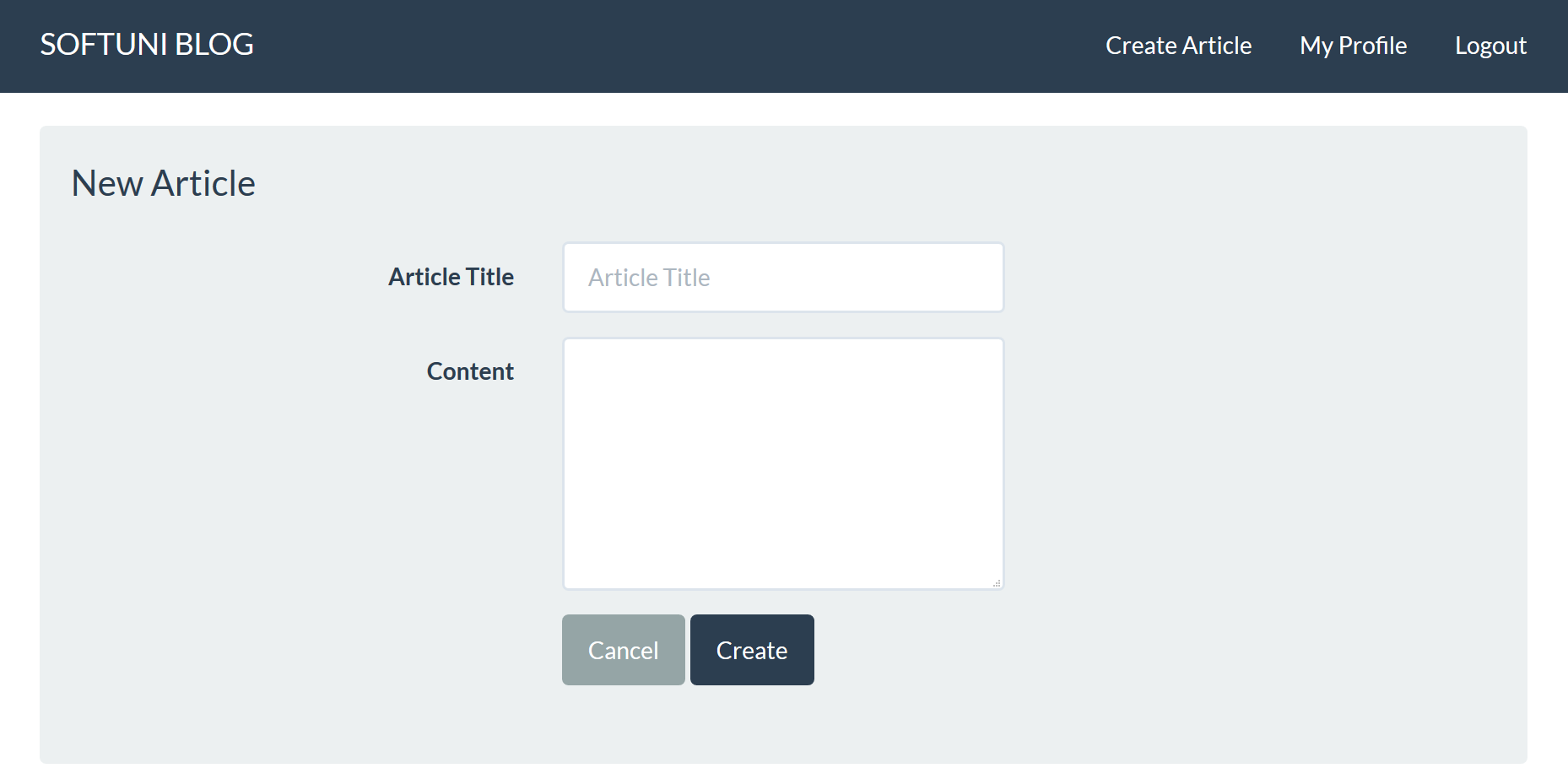
The picture above represents the profile page of a user. It contains only his email and full name. Later, you can extend it, listing all of the articles that the user has authored.

## User Logout

The logout should invalidate the current session.

# Articles

## Creating Articles



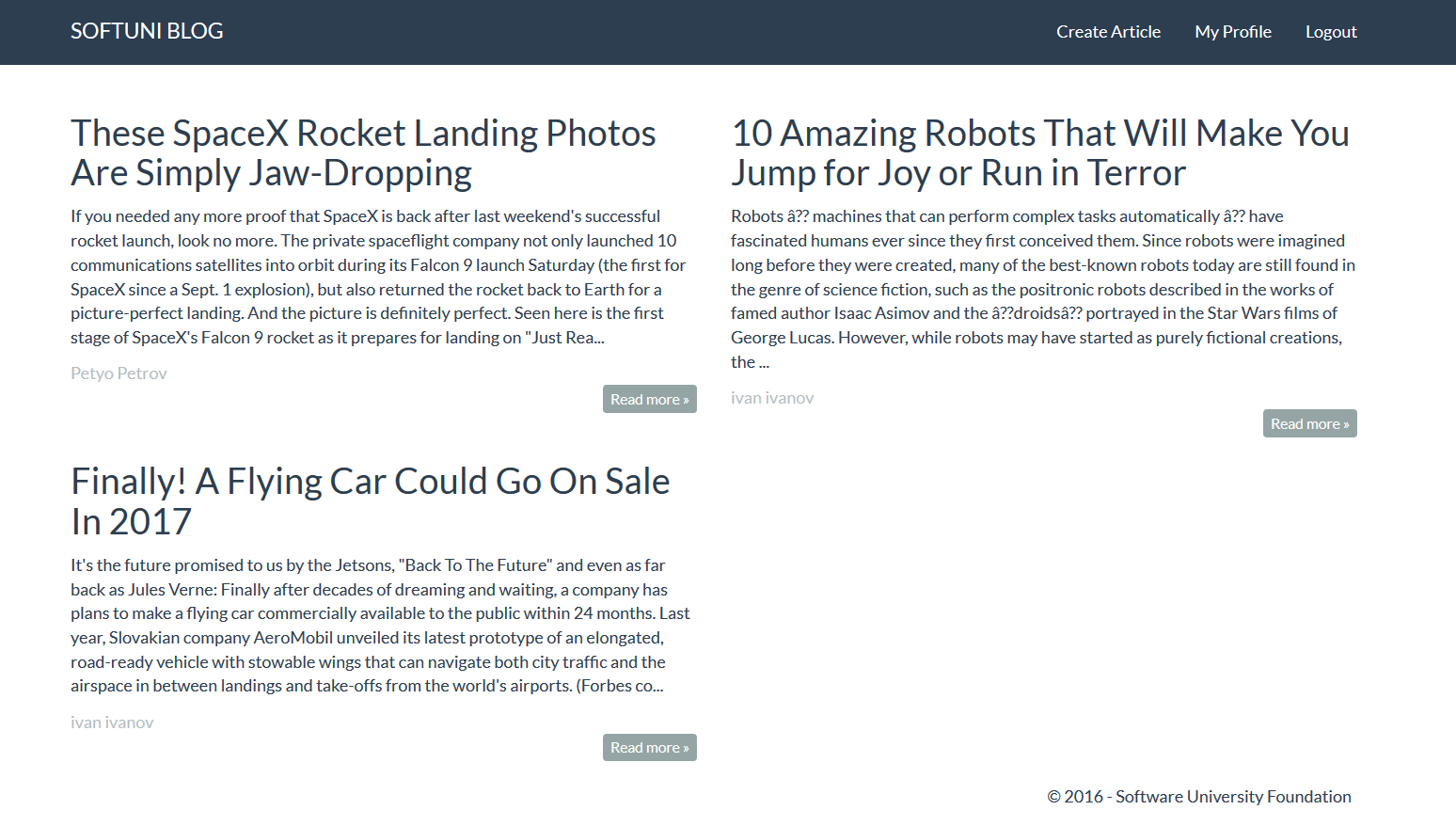
After **user is logged in**, he should be able to create an article. Create the form shown above, and use the following **validations**:

* The Title should be at least 30 symbols long
* The Content should be at least 50 symbols long

The article model should contain the following fields:

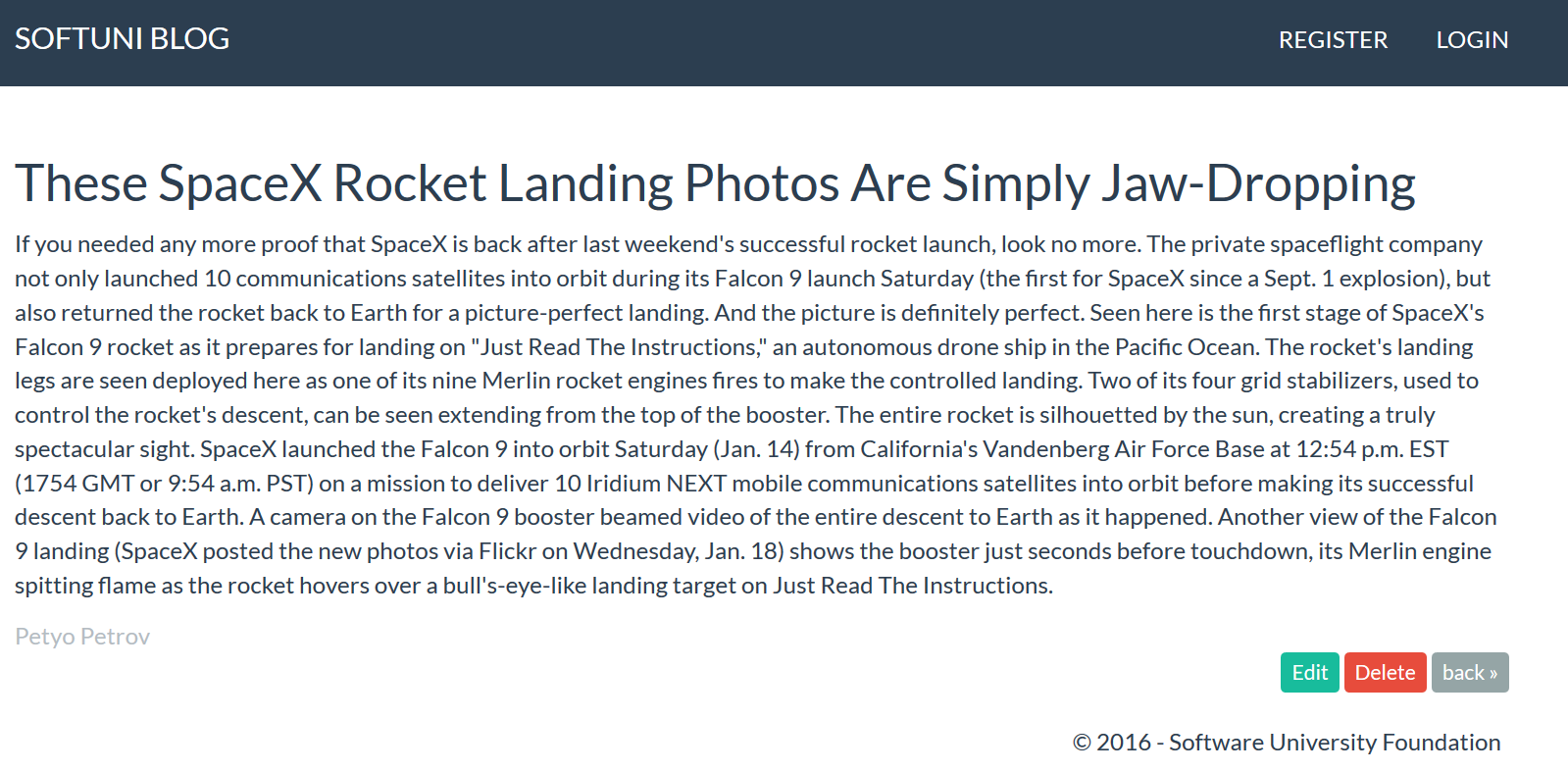
* Id – Identity
* Title – not null
* Content – not null, column type - text
* Author – not null, many to one

## Listing All Articles



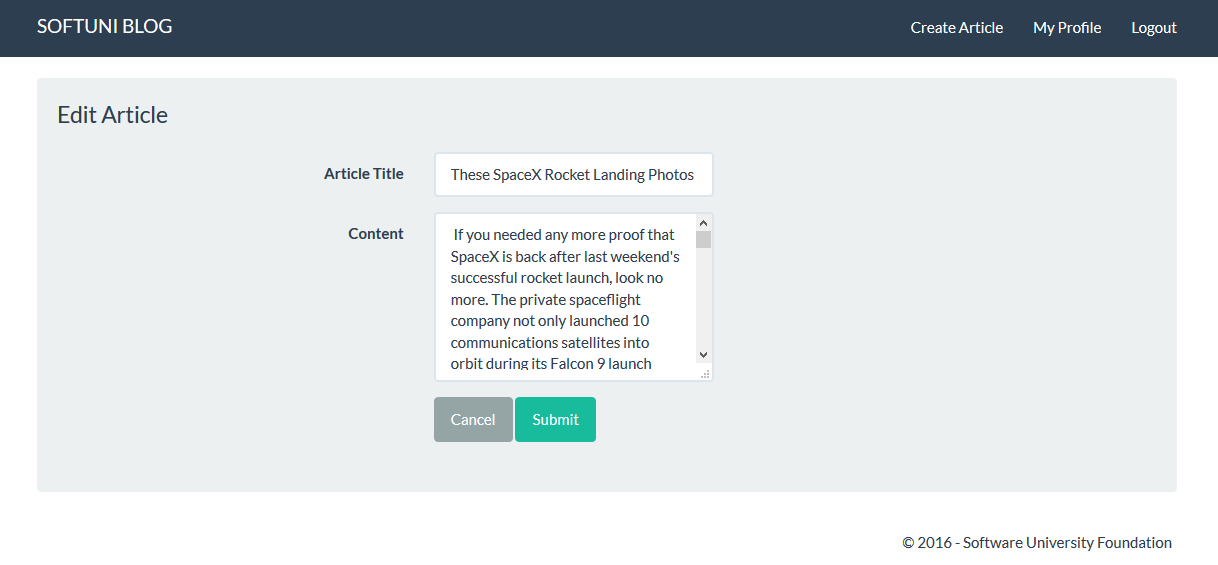
You will be given example HTML page called "index.html", but you need to edit it, to your needs. On that page, you need to show only the **first 500 symbols of the content**. The **read more** button should lead you to the article details. You need to create a **view model** for the article, that will keep only the information that you want to show.

## Showing Article Details



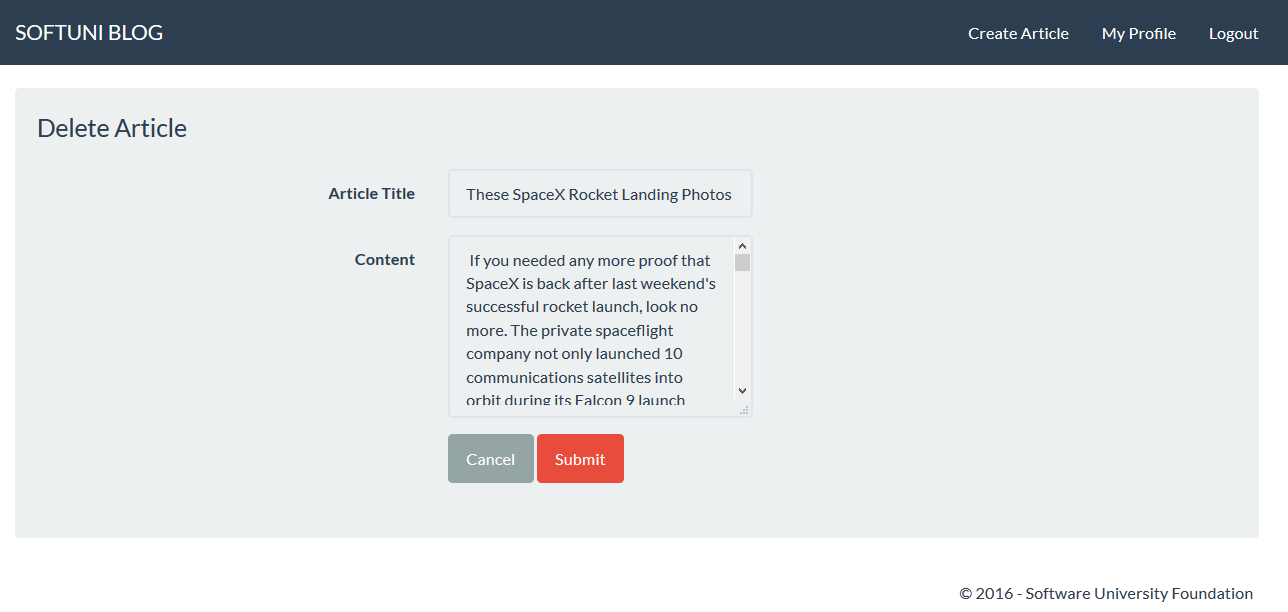
You will be given a page called "details.html". Use it to create the page, you're looking at. You can use the **view model**, that you've created for the previous step.

## Editing Articles



**If a user is logged in**, he should be able to **edit** the **existing** **articles**.

## Deleting Articles



**If a user is logged in**, he should be able to **delete** the **existing** **articles**. He shouldn't be able to change the contents of the fields.

## \*Advanced Functionality

Play around and test your skills. You can create the following functionalities:

* Only the author of an article should be able to edit it
* Add admins, which can edit all articles
* Create categories for the articles
* Create comments for the articles
* Create tags for the articles
* Create a search by tags
* Edit and Delete users by admins

Pictures:



