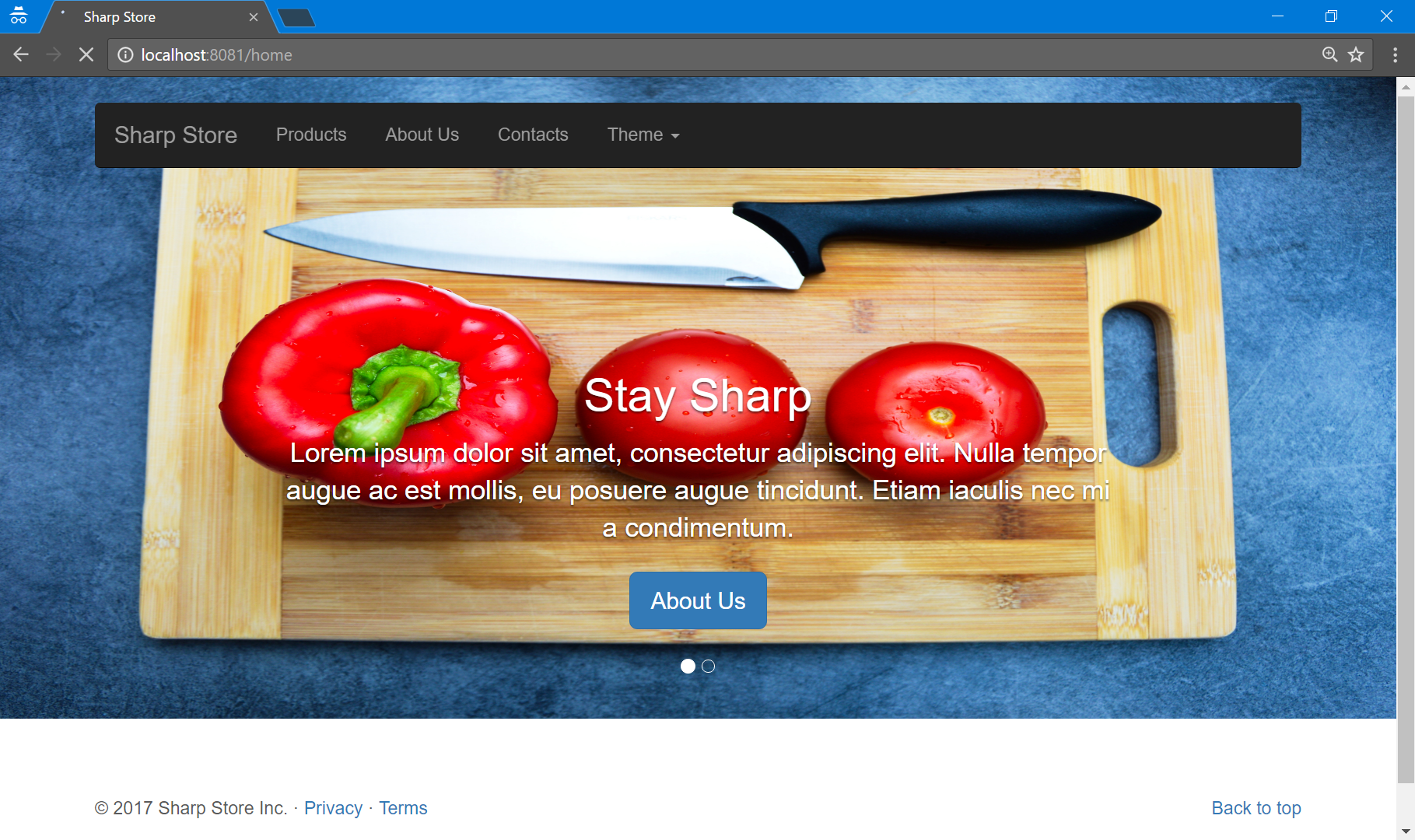
# Exercises: Handmade HTTP Server

Problems for exercises and homework for the [“C# Web Basics” course @ SoftUni](https://softuni.bg/courses/csharp-web-development-basics).

In the following exercises, you will be required to use the HTTP Server you created. (No more Apache ☺).

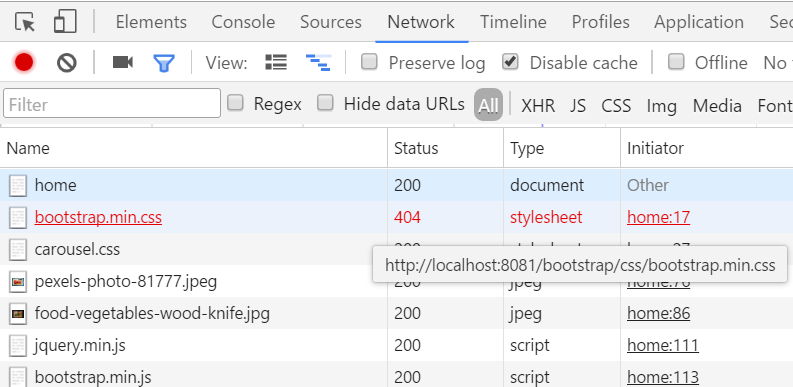
## Sharp Store – Home Page

In that and in the following task you will make a simple company website for company that sells knives. Check out the provided skeleton, look at the structure of the project and the files in the content folder. When you are familiar with the project replace the TODOs in the SharpStore.cs file with actual code.



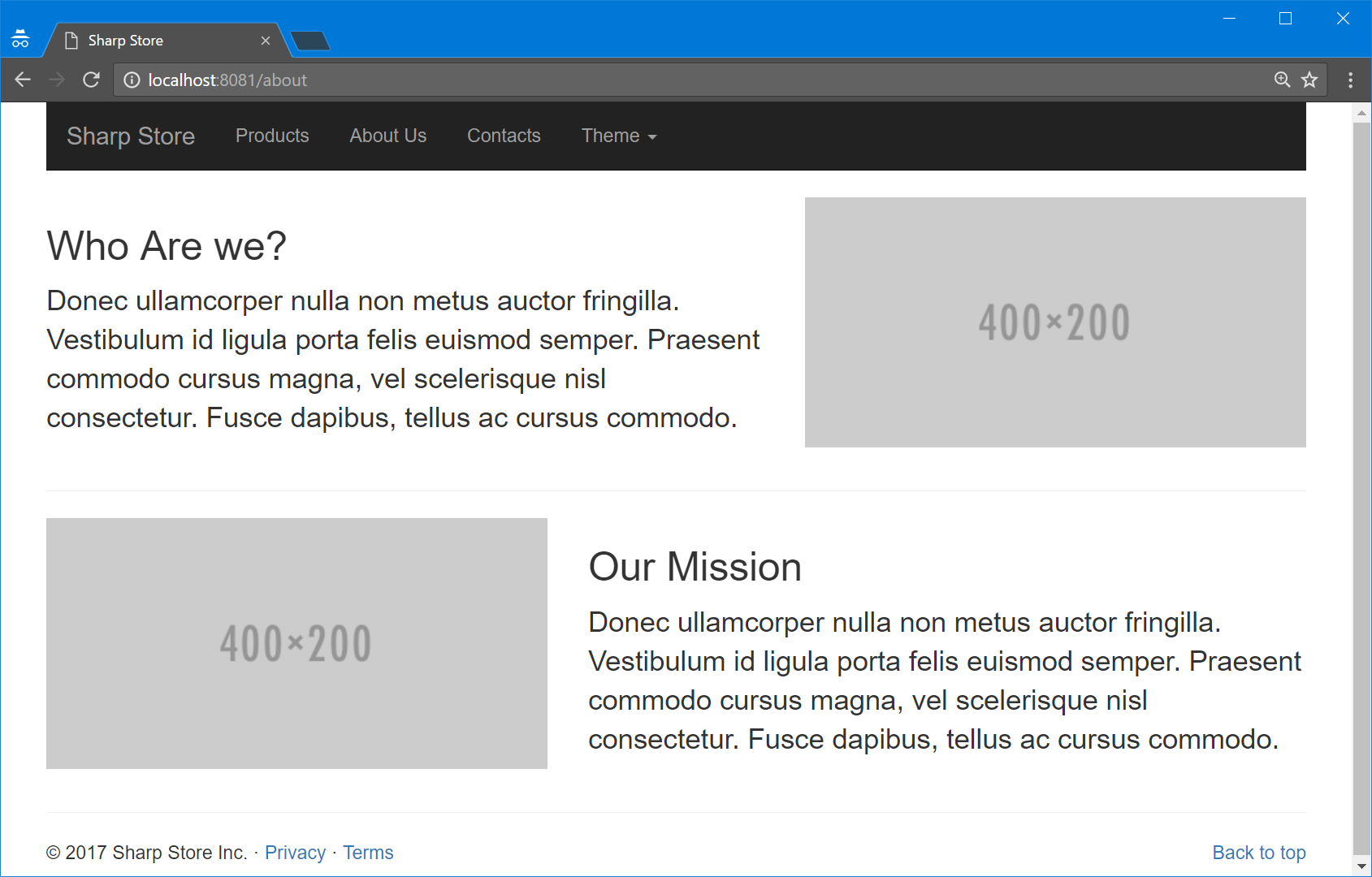
### Hints

* First create instance of HttpServer and run it to listen on any port (for example 8081).
* Open Chrome browser and Right click on the page and select Inspect. Then open the network tab.
* Try to open localhost:8081/home
* In the Network tab, you will notice that that the bootstrap.min.css file has status code of 404. When you hover over it you can see the route that the server is trying to find that resource. Add that route to the list of routes.



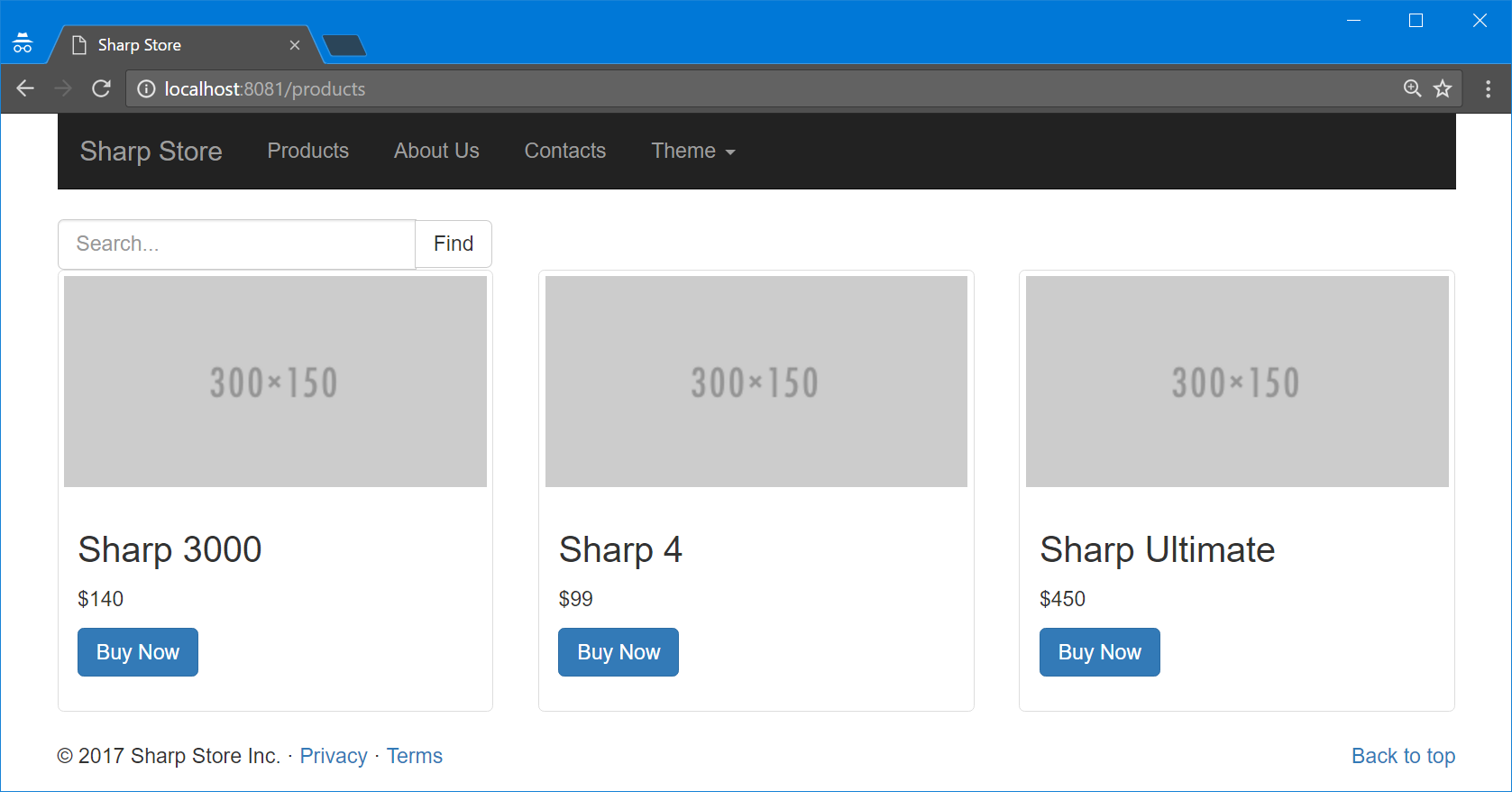
## Sharp Store – About us

Extend the website using the provided about.html in the resources create a page that would look like the one below. Make the links from the menu of the home page lead to that page and make the home page accessible from about us page.



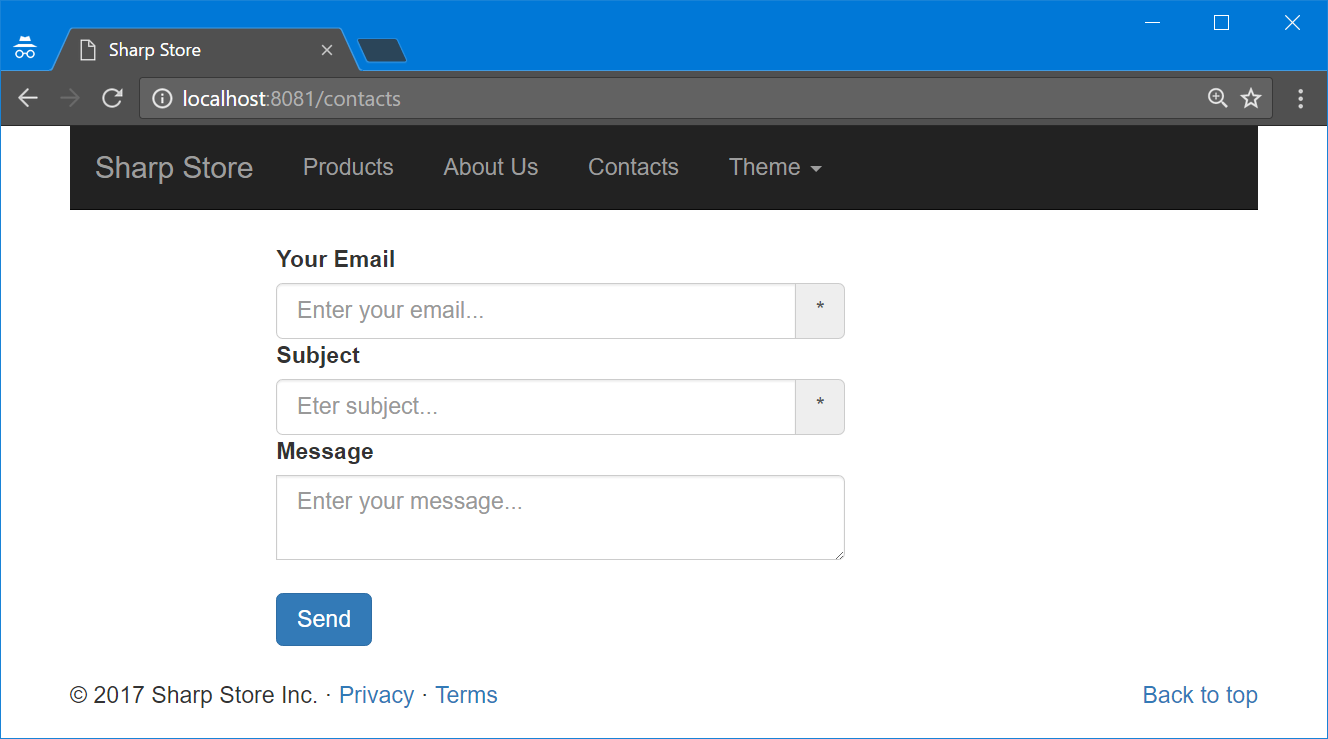
## Sharp Store – Products

Time to add new page all by yourself. You can use the initial about.html page as template and make page similar to that. The final result should be like the picture below and again the navigational links should work.



## Sharp Store – Contacts

Similar to the previous exercise create a contatcs.html page like the one in the picture below.



## Sharp Store – Products from Database

Create database **SharpStore** add table **Knives** in it. Each knife has **name**, **price** and **image URL**. Populate the table with some sample records. Then **modify your application** so when /products page is loaded it should load thumbnails for **all knives listed in the database** instead of just the dummy placeholders you created in Problem 3.

## Sharp Store – Send Message

In the database, you created in the previous problem add new table **Messages**. Each message has **sender**, **subject** and **message.** Modify your application so when the form from the /contacts page sends a POST request the **message would be saved to the database.**

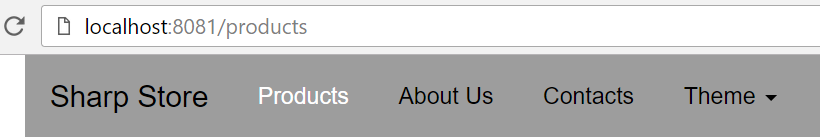
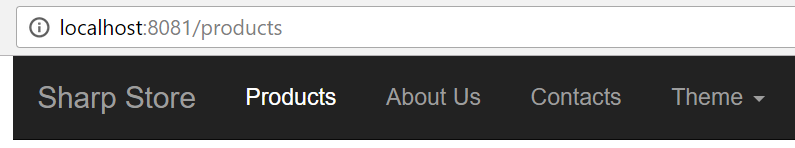
## Sharp Store – Search

Add functionality to the /products page so the visitors of the site can search a knife by its name (or part of it). When the **Find** button is clicked show only the knifes that contains the word provided by the search form.

## Sharp Store – Change Site Theme

Add another functionality to the user. When Dark/Light option from the navigational bar is selected **change the color of the navigational bar** (black/light gray). Make sure it is changed when trying to visit **all pages**.

### Example



### Hint

Use a cookie ☺

## PizzaMore Again

Design and implement a **“PizzaMore” web application** with the following functionality:

* **Home**
  + Home page of the pizza site
  + Should be able to redirect to all other pages
  + Should send a **cookie** to the browser
  + Option to change language via **cookie**
* **Sign Up**
  + Register a new user in a database
* **Sign In**
  + In case of success, a session key should be stored in the database
* **Menu**
  + List of all pizza suggestions with ability to vote for them
  + Should be accessible only when signed in
  + Option to Sign Out

Menu page should also give redirects to 2 other pages – **Your suggestions** and **Add Suggestion**.

* **Your Suggestions**
  + Should be accessible only when signed in
  + He can see a list of all suggestions for pizzas and delete any if he wants to
* **Add Suggestion**
  + Should be accessible only when signed in
  + user can add new suggestion for pizza by specifying its name, recipe and image URL.

Use the lab ([part one](https://softuni.bg/trainings/resources/officedocument/12328/pizzamore-%D0%BB%D0%B0%D0%B1-back-end-part-1-csharp-web-development-basics-january-2017), [part two](https://softuni.bg/trainings/resources/officedocument/12414/pizzamore-lab-back-end-part-2-description-csharp-web-development-basics-january-2017)) from the previous lecture for reference about details to all functionalities.

\*Bonus Tasks

* Make sure there are **no bugs** (for example, you cannot log in again when there is already a logged in user).
* Prevent the user from voting more than 1 time for a suggested pizza.