Homework 1

Networkprogramming ID1212

Diaco Uthman diaco@kth.se

10/12-2017

# Introduction

The goal of this homework was to develop a Client-Server distributed application using Java. The application chosen was the Hangman game.

# Literature Study

The literature study consisted of watching the tutorial videos provided on Canvas, along with discussion with other students, in how to solve the problems that occurred.

# Method

To solve this project, NetBeans IDE was used mainly. Although when trying out different examples, Notepad++ was used.

Before starting the project, it was important to have an idea of the design. The videos that were given, along with looking at the source files of the provided codes, gave a good understanding of what was to be done.

To make sure that the application fulfils the requirements that were given, every requirement was thoroughly tested and approved, using a checklist to sign off every requirement.

# Result

The source codes have been submitted on GitHub in the repository given in the link below:

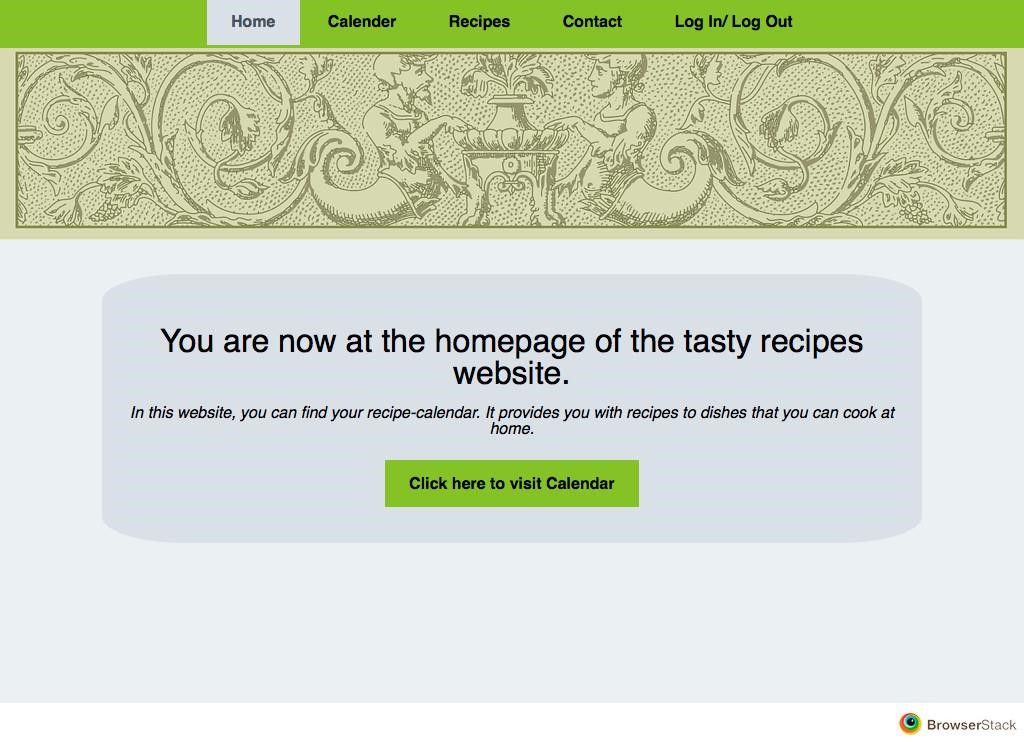
Figure 1: <https://github.com/FlyHighXX/ID1212-Networkprogramming>

*Link to the source codes of the application.*

Some of the requirements will be shown and proven below

 **Client and server must communicate by sending messages over a TCP connection using blocking TCP sockets.**

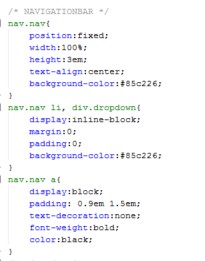
To fulfil this requirement, a set of sockets were created both on client and server side. These sockets communicate



**Figure 1:** This shows the homepage using *Safari 9.1* on OS X El Capitan



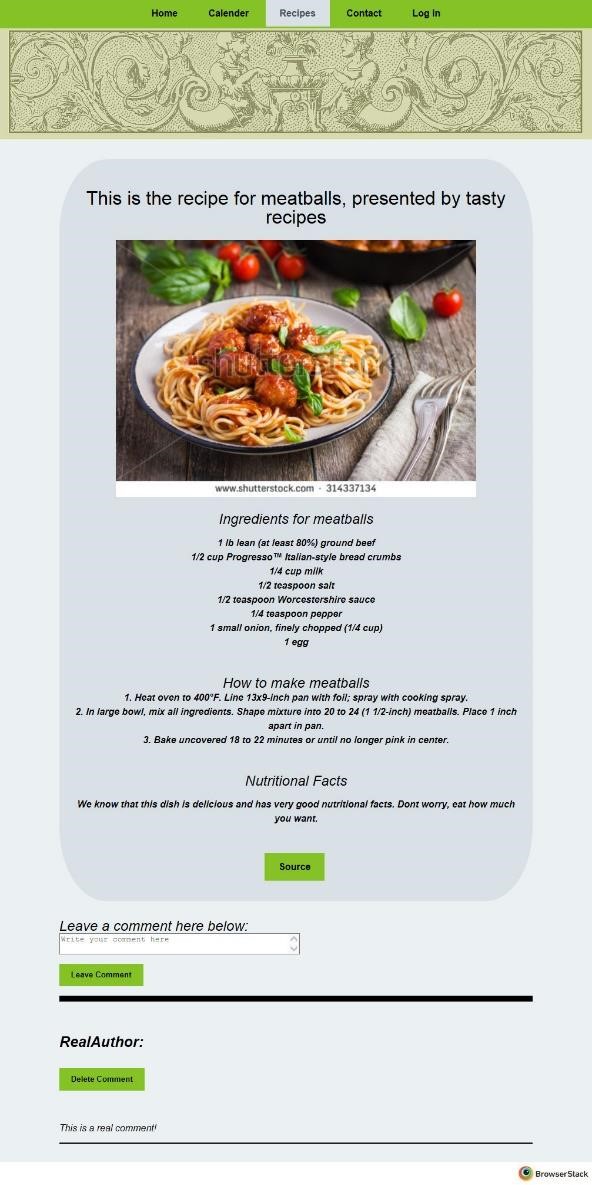
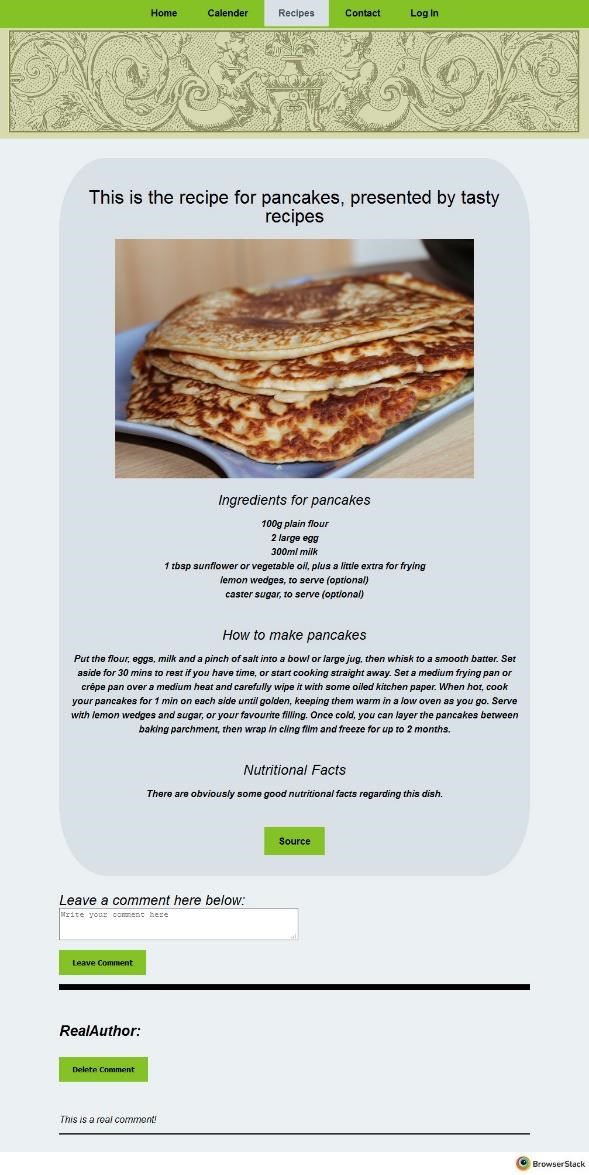
**Figure 2:** The HTML code for the header menu.



**Figure 3:** The CSS code for the header menu.

* **“A recipe page shall contain the name of the dish, an image of the prepared meal, a list of ingredients, instructions, nutritional facts and user comments.”**

As Figure 4 shows, there are two different recipe pages, accessible through the dropdown menu that was described in Figure 2 and Figure 3. These include all the requirements that were given. Since these requirements are only written text into a container, there is no need to include the codes for them. It would only be a big chunk of code that provides nothing interesting.

-----

**Figure 4:** The meatballs page on *Internet Explorer 10* and pancakes page on *Firefox 44.*0 running on Windows 10.

* **“The calendar shall be a visual representation of one month, with clickable images of the month’s dishes. These images shall be links to corresponding recipes. Your calendar shall have dishes two days in the month, the meatballs day and the pancake day.”**

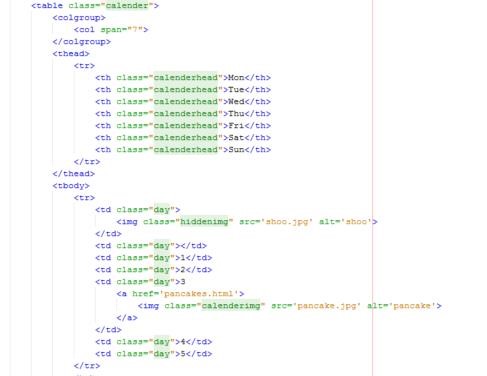
The images in the calendar are clickable. By clicking a picture, you will be redirected to the recipe page of that dish. The calendar itself, was a bit tricky to create, but after some study, it was created. The calendar page is shown in Figure 6 along with the code for the calendar itself in Figure 7 and Figure 8. This was the most time-consuming part of the project, but also the most interesting in my opinion.

The calendar was created using a table tag. This way, the head of the calendar provided the width of the columns and the squares below were vertically aligned accordingly. This way, there was no need to worry about the widths. They were always equal.

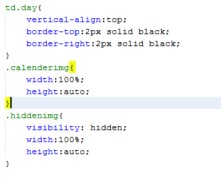
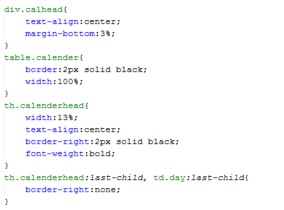
The reason for having an image that has the class name “hiddenimg”, which is a hidden image, was to make all the rows have the same height. It may be an ugly solution, but at the time of making the webpage, this was the only solution that I could come up with.



**Figure 6:** The calendar page shown in Chrome 50 running on windows 10.



**Figure 7:** The HTML code for the calendar. To save space, only first week was included. The other weeks are like the <tr> tag and the other image is the same as the one with pancakes.



**Figure 8:** The CSS code for the calendar.

* **The report must show that your web site is identical in the specified browsers.**

As shown throughout the report, with the user interface screenshots, the pictures are taken on the different browsers, and give the same layout. Each picture was taken on a different browser using the modern.ie webpage. To give some more proof that the pages do look the same, the next pictures will be of the pages, in other browsers than those previously shown.

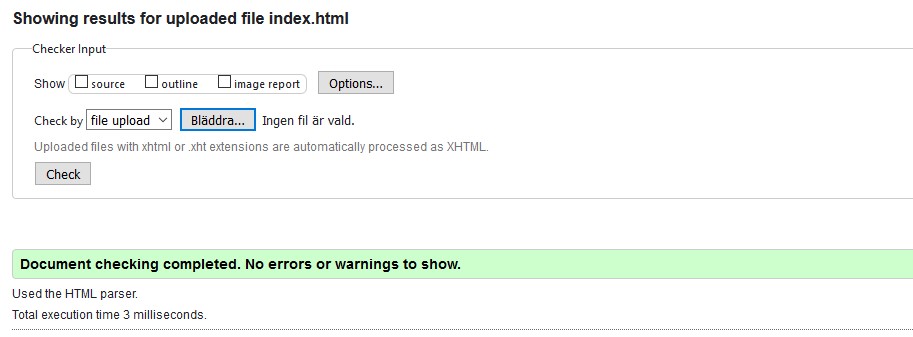
* **The report must show that a reset CSS is used, and that all files have passed W3C validation.**

To show that a reset CSS file was used in this project, Figure 9 will show the head of the homepage HTML file. Row 2 of that figure will show that a reset.css file have been used. The same row is included in all HTML files.

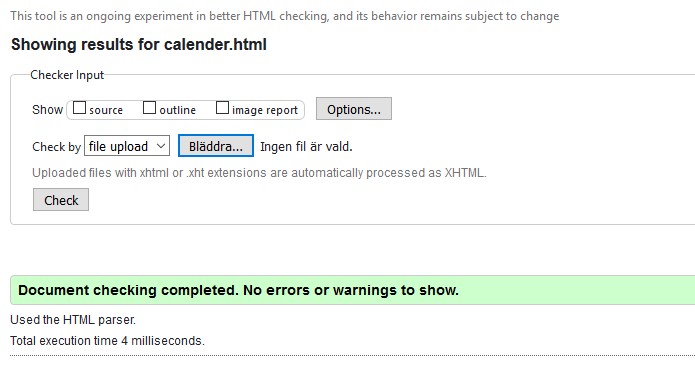
All files were successfully validated using the W3C validation. Figures 10-14 will show the validations of the files. Because the CSS validation did not provide information about which file was validated, I have decided to include one picture of how the page looks, if there are no issues with the file. All .css files passed the validation.



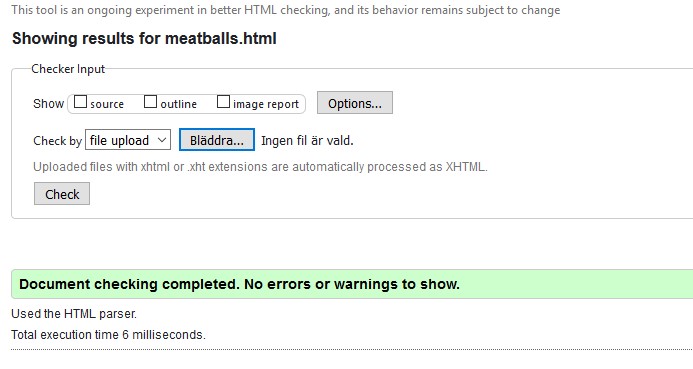
**Figure 9:** Shows the <head> of the homepage HTML file.



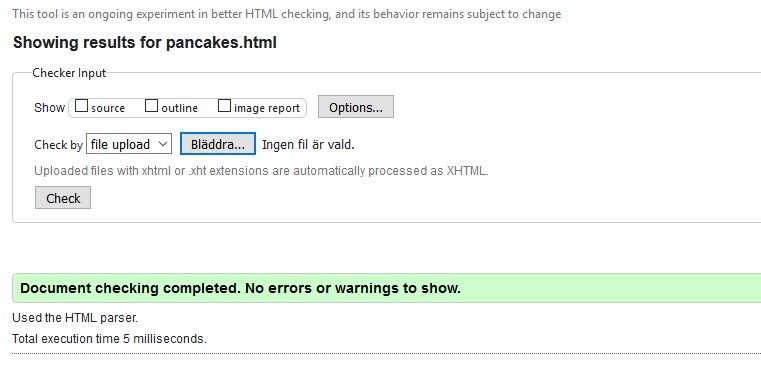
**Figure 10:** Validation of index.html file showing the firstpage of the Tasty Recipes webpage.



**Figure 11:** Shows the validation for calender.html which shows the calendar of the webpage.



**Figure 12:** Shows the validation for the meatballs.html file. This shows the meatballs recipe.



**Figure 13:** Shows the validation for pancakes.html recipe page.



**Figure 14:** Shows a sample of how the confirmation of a successfully validated CSS file looks in the W3C Validation Service.

* **Visibility of System Status**

Because this version of the page is still very basic, the only important information that the user needs to know about the page, is which page is being seen. Therefore, a white rectangle is seen in the Page Header. In further developments to the webpage. Also, the page changes within one second of when a user clicks on any of the links.

* **Match between system and the real world**

All text on the page, such as the welcoming of the user in the first page, are given in a way that is understandable for an arbitrary user. Although, since there is no implementation of error messages as such, there is no big need to emphasise this matter, in the Tasty Recipes webpage. Later, when there is a possibility to write comments, or to log in, there will be features needed to make the webpage understandable for the user.

* **Consistency and Standards**

This topic is, as many of the other topics, very subjective. In my opinion, the requirement has been met. The webpage has a standard look. The login link is located at the top along with links to all the other pages of the webpage. This way, the user always knows that he should look at the top whenever he/she needs to navigate to another page. This is consistent and follows the standards. Whenever I want to navigate to another page, my instinct is to go to the top of the window and look for either a search-area or the page that I want to navigate to.

* **Recognition rather than Recall**

In this basic webpage, the only thing that a user needs to remember, is which page he/she is located in. There is not anything to do within the webpages. This is achieved by the page header that highlights which page is currently seen. This way, the user does not have to recall what he/she pressed in the previous page, but can simply look at the page header.

In further developments, it is important to be informative about what the user have done, because there might be many pages introduced, such as a log in page. This topic will be much more important then, rather than now.

* **Aesthetics and Minimalist Design**

Once again, this is a subjective topic, which might be interpreted differently, by different users, but in my opinion, the page is minimalistic. There is not anything on any of the pages, that is unnecessary or unneeded. One could argue that a link to the calendar is unnecessary when there already is a link in the page header, but that was a requirement of the assignment, and therefore implemented in the page. The page header, however was not a requirement, but still implemented because it made sense, to achieve some of the requirements such as keeping the user informed about which page he/she is visiting.

* **The report must show that your web site is identical in the specified browsers.**

Throughout the results, pictures have been posted of the pages, running on different browsers. The pictures were taken using the modern.ie webpage, and when I used the page, the available versions were Firefox 44.0, Safari 9.1, Chrome 50.0, and Internet Explorer 10.0. Hopefully this is okay, because I could not install Chrome 49.0 and Safari 9.0 on my laptop.

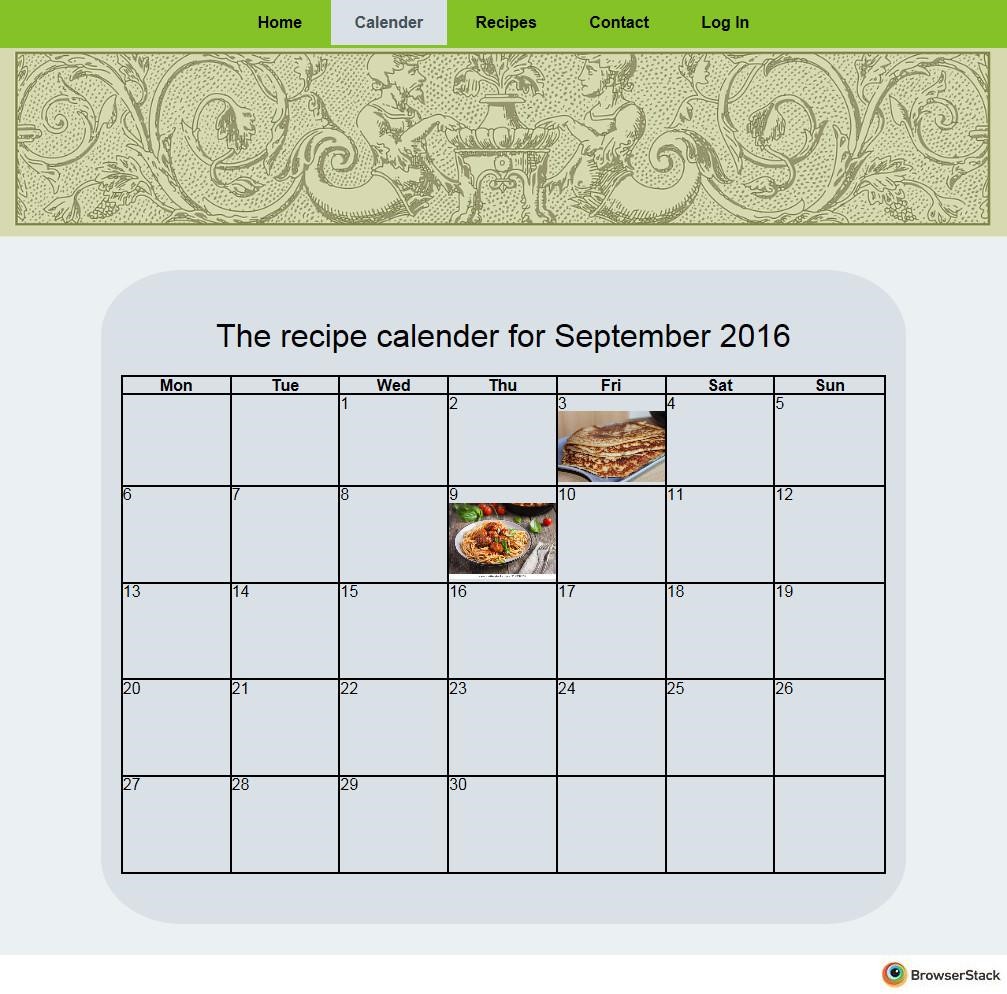
To show that the calendar page looks alike in different browsers, there are some more screenshots included in Figure 15, 16 and 17. The fourth screenshot of that page is shown in Figure 6.

The recipe pages have been shown in different browsers on Figure 4. Two more pictures are included in Figure 18 and 19, to prove that they look alike in different browsers.

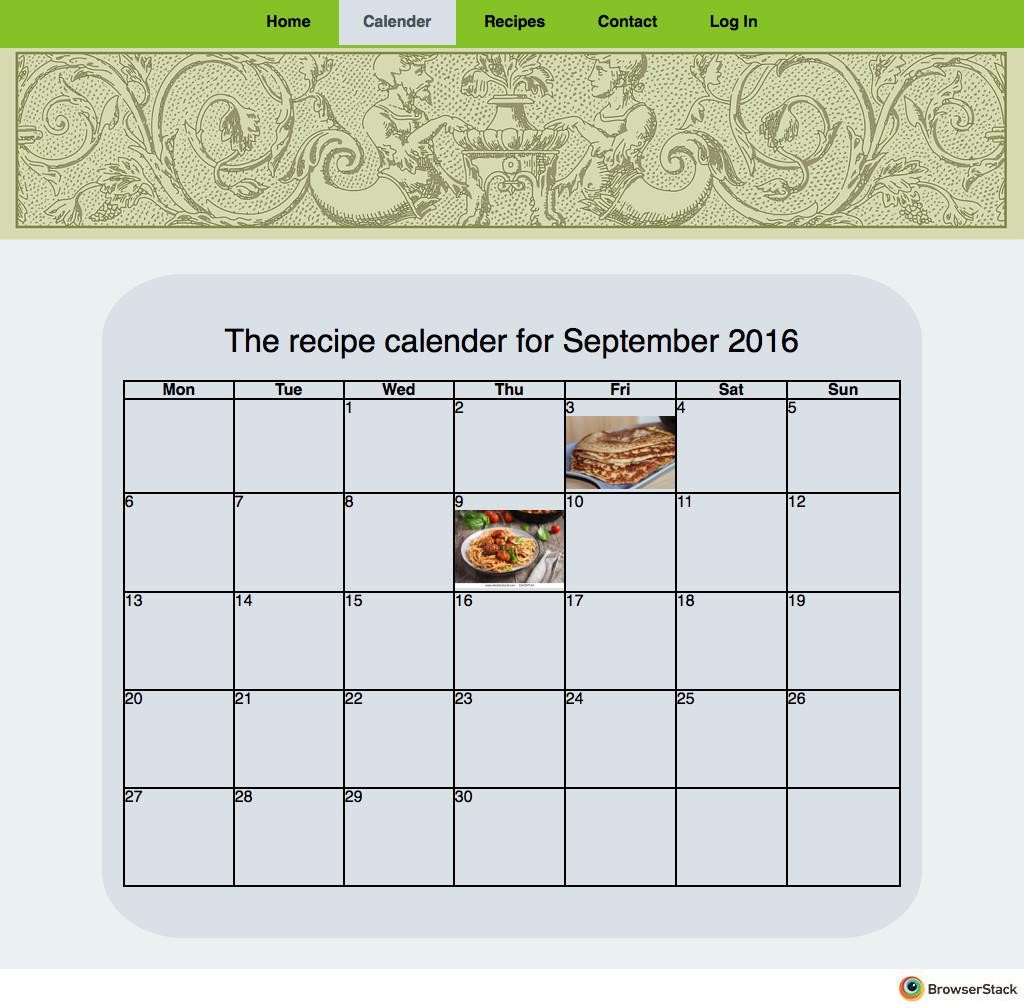
It has been decided not to give any more pictures of the home page, because it is such a basic page, which contains the same header as the other pages, and doesn’t have anything interesting in the page content.



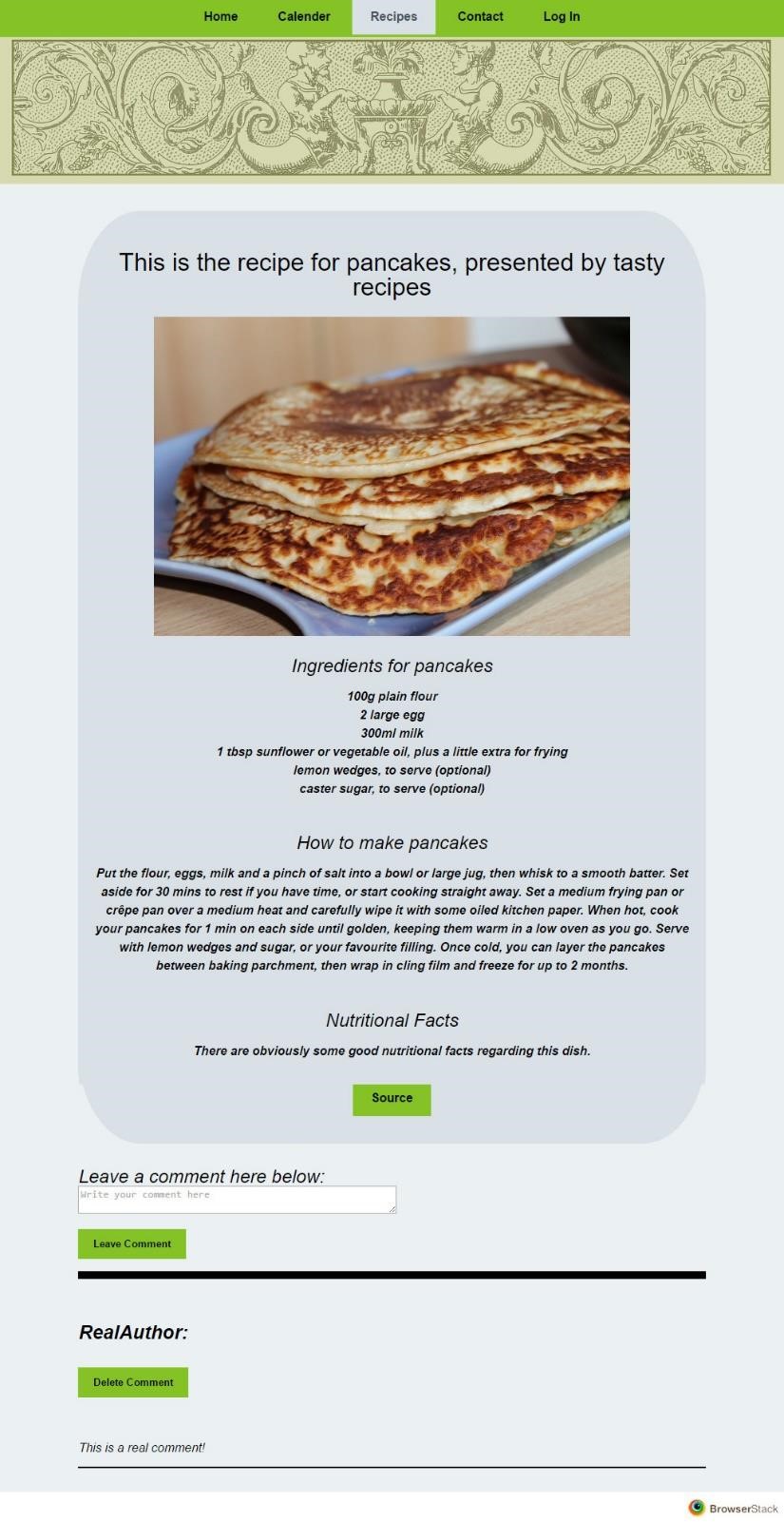
**Figure 15:** Shows a screenshot of Calendar.html in Firefox 44.0



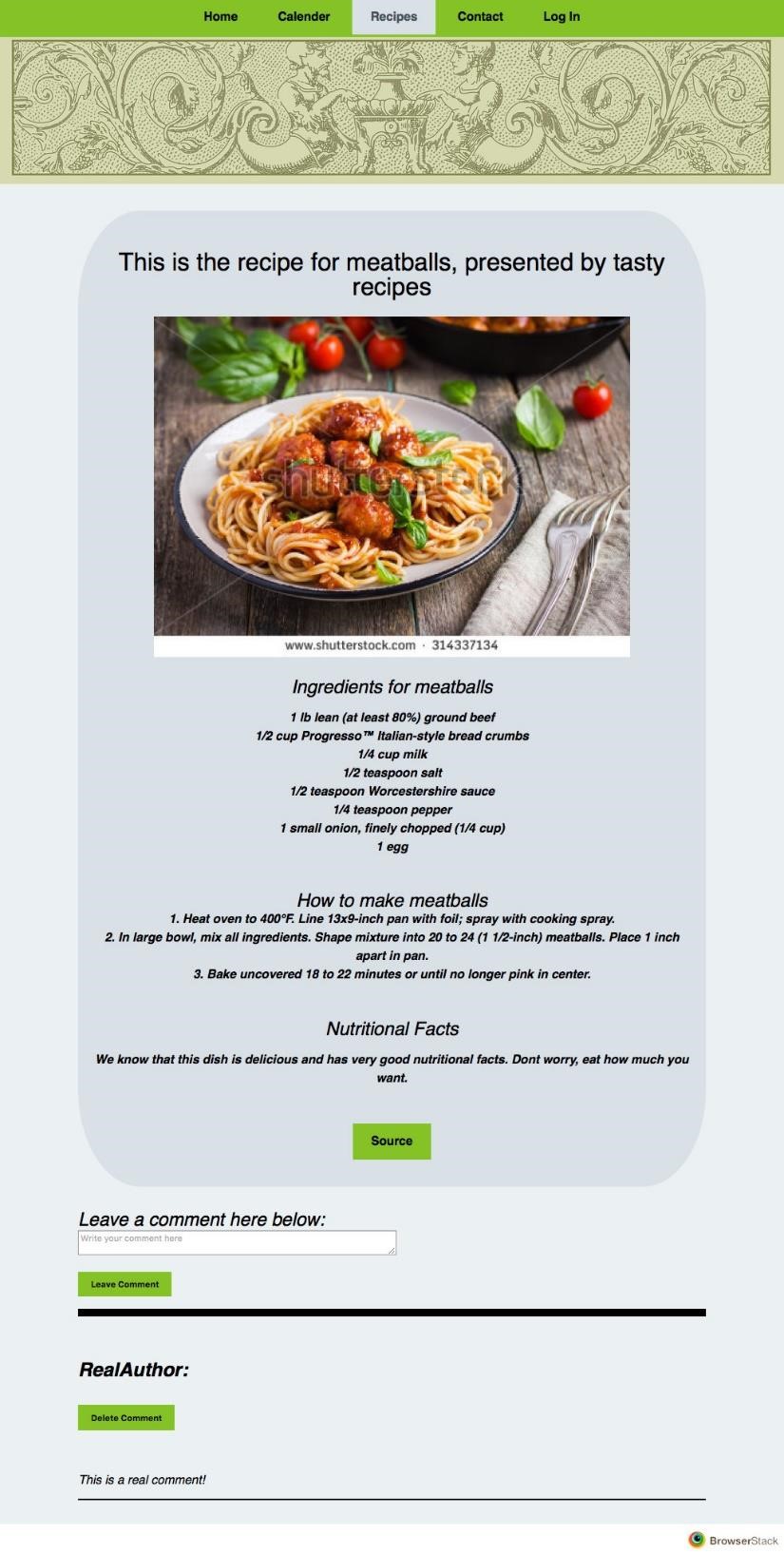
**Figure 16:** Shows a screenshot of Calender.html in Internet Explorer 10.



**Figure 17:** Shows a screenshot of Calender.html in Safari 9.1



**Figure 18:** Shows a screenshot of pancakes.html in Chrome 50.0



**Figure 19:** Shows a screenshot of meatballs.html in Safari 9.1

* **Use text alternatives**

Figure 20 shows a piece of code which shows that if the image cannot be found, then a text alternative will come in its place. The reason for implementing this, is that for some reason the image might not be available at the time of request. At that time, it is important to somehow show that there should be an element there. The text alternative provides us with this feature. It is implemented by the “alt” property. All images throughout the web page use this alt property in case the picture is not available.



**Figure 20:** Shows an image tag that provides a text alternative.

* **Don’t rely on colour alone.**

To avoid relying on colour alone, there are clear topics on every page which describe which page the user has entered. If the colours of the header are gone for some reason, then the user will still know which page he/she is visiting simply by having a look at the topic of the page. The figures throughout the report show how the views look, hence there are no new pictures included here.

* **Use HTML and CSS properly.**

The webpage has been developed using external style sheets to handle all the layout. HTML structural elements have been used to define the structure of the page in a proper way that makes the webpage work correctly and follow standards of the responsive design.

Figure 2 and 3 shows an example of how HTML and CSS has been used in relation to each other. We can see that there is no layout information in the HTML file, but only elements assigned class names. The elements are then handled in the external CSS stylesheet shown in Figure 3. This is the proper way to using HTML and CSS in relation to each other.

* **Provide clear navigation mechanisms**

Clearly, the page header that is shown in all pages of the website fulfils the requirement of having a clear navigation mechanism. The user will always know which page is the current one, and can easily navigate to the next one. The page header can be seen in the figures throughout the report. This is enough for the user to able to navigate between the pages of the website and understand what happens.

# Discussion

All mandatory tasks have been completed, and one of the optional tasks. The responsiveness does work since I have implemented the responsive design, but I have not implemented the CSS media queries to make things different on different resolutions. I could have done it, but decided not to. The page looks relevant on different resolutions, but since media queries are not used, I am not sure that it is acceptable.

The assignment was hard in the beginning, because I wanted it not only to work, but also to look good. I had to start over about 2 times, because I did not like the design, but in the end, I came up with a nice design that I could live with. The coding itself was not very hard, because the language is easy.

Some CSS properties were hard to understand, because they seemed similar, but still produced different results, but in the end, most of them were generally understood.

# Comments About the Course

Because I was ill during the first seminar, I could not participate. This made the whole process of completing the assignment, much longer. I would say that this seminar took about 40 hours to complete, because of this. This was much due to illness and because I had to prepare two times for the seminar.

[i http://www.color-hex.com/color-palettes/](http://www.color-hex.com/color-palettes/)