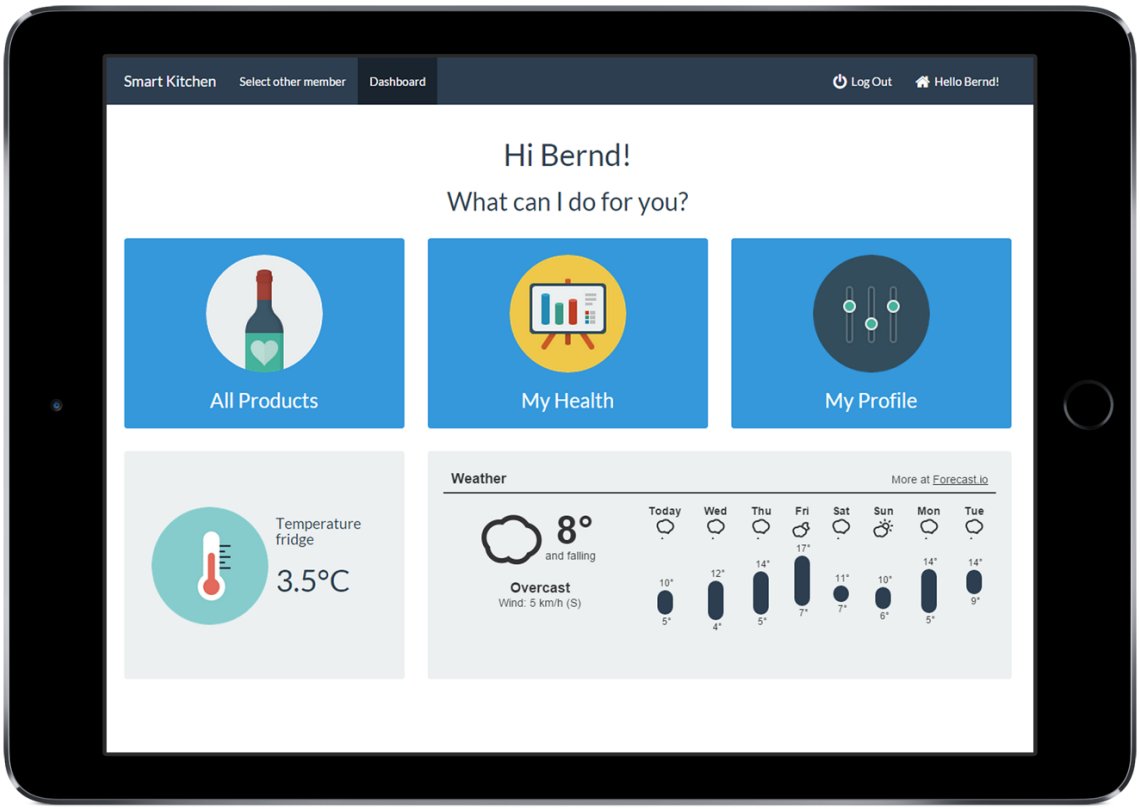
Project Sesam - Smart Kitchen

Smart Fridge

Spring 2015



Student
Bernd Verhofstadt
Heiligenweg 29
D-47626 Kevelaer
bernd.verhofstadt@outlook.com

Sending institution
Artesis Plantijn University College
Paardenmarkt 92
2180 Antwerp
Belgium

Receiving institution
OAMK - Oulun seudun ammattikorkeakoulu
Kotkantie 1
90250 Oulu
Finland

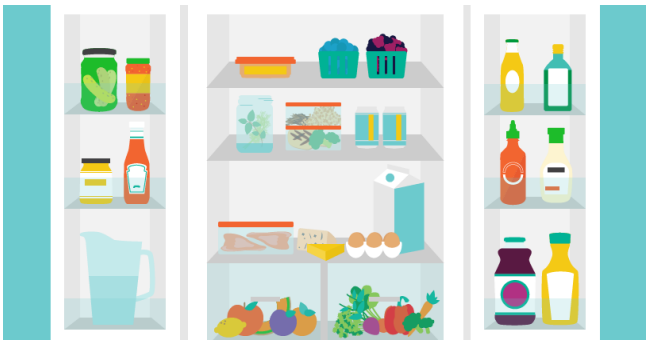
Table of contents

1. Introduction

Technology is not standing still. We want more and more automated devices. These devices will be used on a daily basis. A house is a place where you can find a lot of these things, for example the kitchen. The Sesam project is based on a voice character. When you call Sesam, he/she can do something like “Sesam open”. The Sesam project is a smart kitchen where you could do the same. For this project I worked with Tom Mampaey on the Smart Fridge, a spin-off from the Smart Kitchen. In the beginning we had only two words. Smart and Kitchen.

2. Brainstorm

The first step to the project was our brainstorming session. A blank paper on the table with only the words ‘Smart Fridge’. The ideas that where written on the paper where very different. Some major ideas:



- When standing before the fridge, you’ll be recognized
- Voice control like Siri, Google now, … -> Text-To-Speech and Speech-To-Text
- Glass screen with touch functionality
- See what products are in the fridge
- Get recipes suggestions what’s in the fridge
- Challenge your family with points to eat healthy
- Order food online on your fridge

With these ideas we made a selection, what looked possible in a time period of four months. To discover the possibilities we did some research and tests.

3. Research

The first thing that I wanted to make, was a user interface. Easiest thing that I could do was grab a computer, screen, keyboard and mouse and build it in the fridge. This solution is not the best one… It’s expensive and overkill on hardware. So we decided that we wouldn’t use a desktop computer in the fridge.

As mentioned above, a full size computer isn’t the best option. But when there are computers with limited hardware which are less expensive, like Raspberry Pi it could be a great choice.

I started with doing tests on the Raspberry Pi. Voice control was on the list so I did a lot of research on this subject. I found some companies that offered ‘free’ STT (Speech to Text). I looked into it but the results were not satisfactory. Google chrome was a solution but as it won’t work on Raspberry pi, this wasn’t an option. Because of the limited budget and time that we had to spend, I decided not to use STT.

I was testing with Raspberry Pi to make a user interface but it was not user friendly, you only can use limited graphics due to the limited hardware.

I needed another solution, something more flexible to run. Everyone is using a smartphone or tablet these days, thus I was thinking about a mobile application. But the question was for which OS. Because we didn’t have enough time, we had to make a discussion. I was thinking about a responsive web application, in this way you can use it on any device with a modern browser and internet connection. The web application should be compatible with different devices and easy to use. We decided that I would make a web app, which at the end can be converted to a mobile application without a lot of coding.

To run a web application, there are two things that you need. A web server and data storage.
The advantage of an offline web application is that you don’t need an internet connection. The downside is that It’s more difficult to deploy and you should make an offline data storage. To make it more

dynamical and have more options, I decided to make everything in the cloud. Luckily a web server wasn't a problem, since I have my personal one.
For the data storage, I had to choice the most suitable technology for the back-end.

Next step was to do some tests with different technologies.

Framework	Database
.NET MVC	MySQL
Web API 2	MS SQL
NodeJS	MongoDB

In the first place, I wanted to learn something new. I wanted to use NodeJS with MongoDB. With this technology I had no experience or idea how it works. I did some small projects but it seemed difficult to debug etc.

I read an article about MVC and Web API and they said that Web API 2 is the future, but MVC will disappear in a few years. So I choose the best one, Web API 2. For the database, I took Microsoft SQL Server 2014. To make it more flexible, I will use the Microsoft Entity Framework.

4. Work Methods

To realize this project, we used a few methods. The most important one was SCRUM.

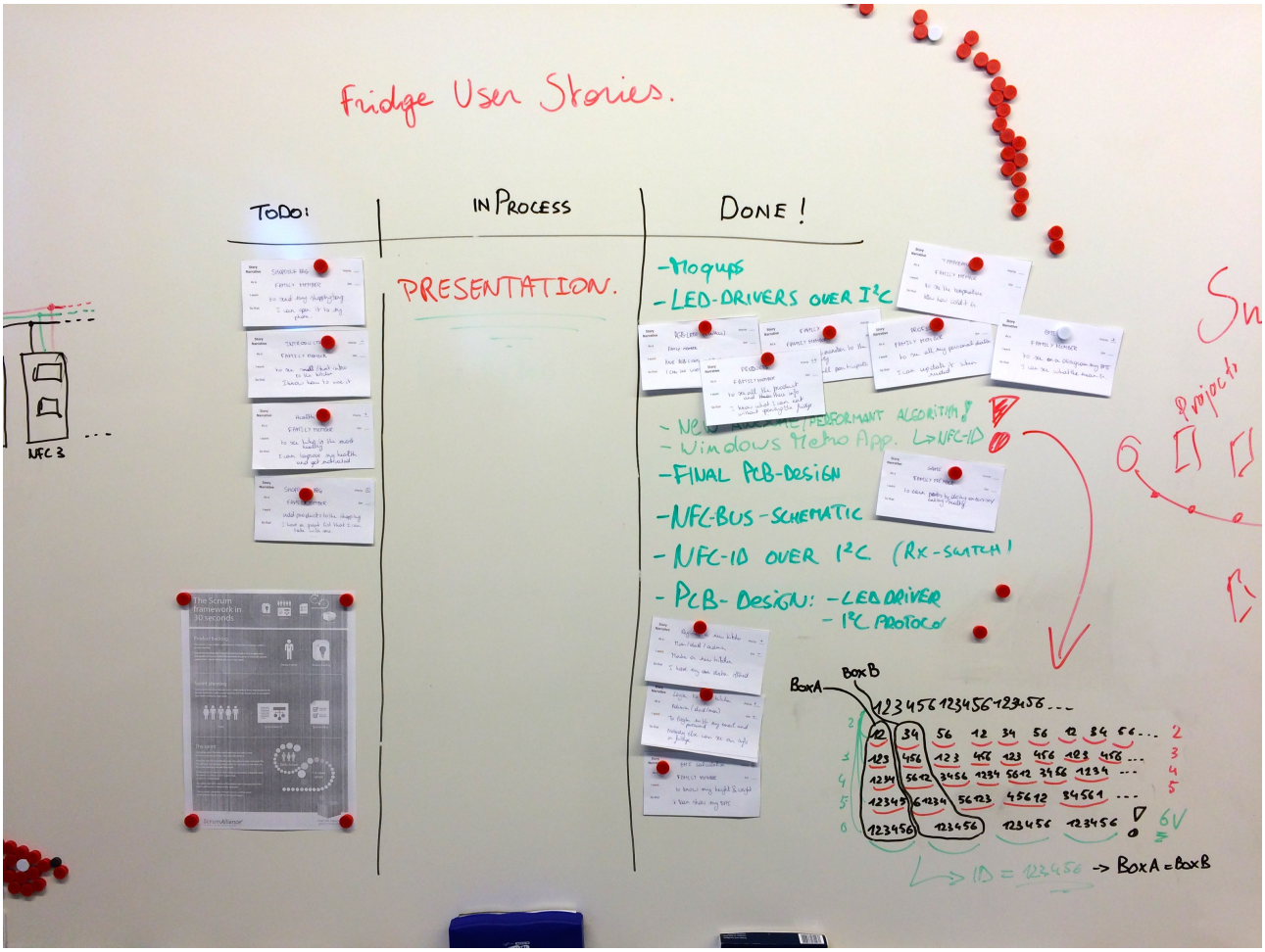
Scrum is a work method that is used for project management. When using scrum there are a lot of possibilities, we did a selection of the most important features.

User requirements

I made some user requirements for the software part. What does the end customer want ? For example: 'As a family member I want to see which products are in the fridge. Off course we added some priority to see what we had to do first.

Scrum board

We had a whiteboard where we made a table with following columns: to do, in progress and done.



Weekly meeting

We had a weekly meeting with our supervisors about the project progress. What we had achieved so far, what the next sprint is about etc.

Standup meeting

The standup meetings where daily and were possible on every moment of the day.

Sprint

The sprint was every two weeks. Where we had some deadlines that we have to finish by then. We discussed the sprint with the supervisors.

Git

We worked with GitHub. It is a web-based Git repository hosting service, which offers the full distributed revision control and source code management (SCM) functionality of Git as well as adding its own features.

I created an organization named 'OAMK Smart Kitchen' on GitHub. All the people that will work on this project can work on there own repository within the Smart Kitchen organization.

https://github.com/OAMK-Smart-Kitchen

5. Software structure

The software can be divided in two parts. The back-end and the front-end.

Front end

The front end is the layer between the user interface and the back end. In the front end, the client application that is running, uses Angular / javascript to communicate with the back end. The tablet is connected over the home WiFi router with the browser to the internet.

Back end

The back end is the place where the data is stored on the server, the point where the user can interact or request data from the server. The back end is placed in a datacenter. There are several server types, the Web server will store the client application. When the user enters the URL from the application, the web server will send the relevant data back to the client.

Windows Server contains different parts, Sql Database, WebServer that runs Web API 2 application. It's possible to deploy with Microsoft Deployment in Visual Studio. The windows server has a layer between the Web API 2 application and the Sql 2014 database, Entity Framework. This framework can generate a database from a model, controller or reverse.

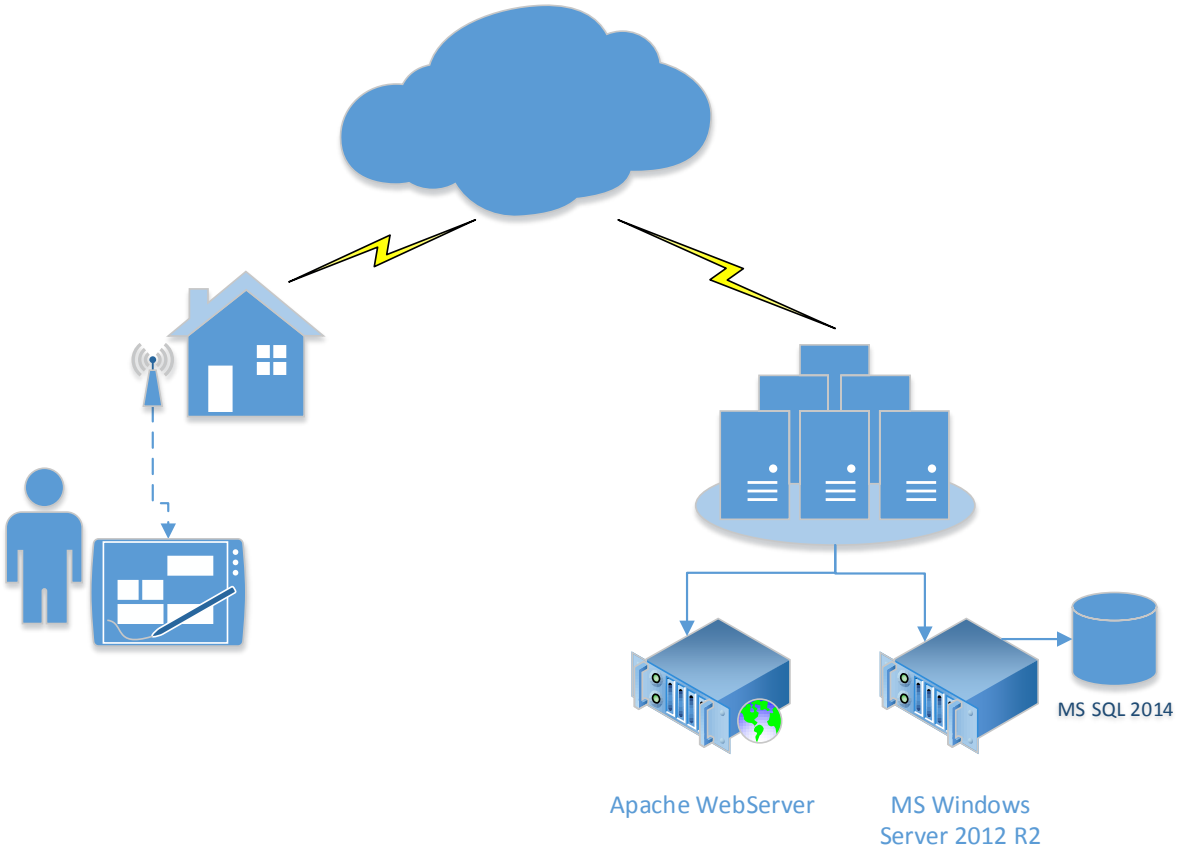


FIGURE 1: SOFTWARE STRUCTURE. LEFT SIDE FRONT END AND RIGHT SIDE BACK END

6. Application

A web application can run on any devices with a internet connection, a browser with support for javascript, HTML 5 and CCS3. Without a lot of coding it's possible to make an application in the app stores for multiple

OS with a WebView/browser embedded. The web app coding is for every app the same, you just have to embed it in the code of the OS.

I created a Windows App for Windows 8 or higher. The Windows application contains a browser that will load the application. The advantages of a native application is that you have a nice icon and don't need any bookmarks etc. The native app will show the web application in full screen, in this way the end user will recognize it as a real Mobile application and not as an Website.

Frameworks and technologies

- AngularJS

  AngularJS lets you write client-side web applications as if you had a smarter browser. It lets you use good old HTML as your template language and lets you extend HTML's syntax to express your application's components clearly and instantly. It automatically synchronizes data from your UI (view) with your JavaScript objects (model) through 2-way data binding. To help you structure your application better and make it easy to test, AngularJS teaches the browser how to do dependency injection and inversion of control.

  Oh yeah and it helps with server-side communication, taming async callbacks with promises and deferreds. It also makes client-side navigation and deep linking with hash-bang URL's or HTML5 pushState a piece of cake. The best of all: it makes development fun!

- Html 5

  HTML 5 is the latest stable version of the standard. It has a support for geolocation, we use this for the Weather by Forecast.io.

- Bootstrap

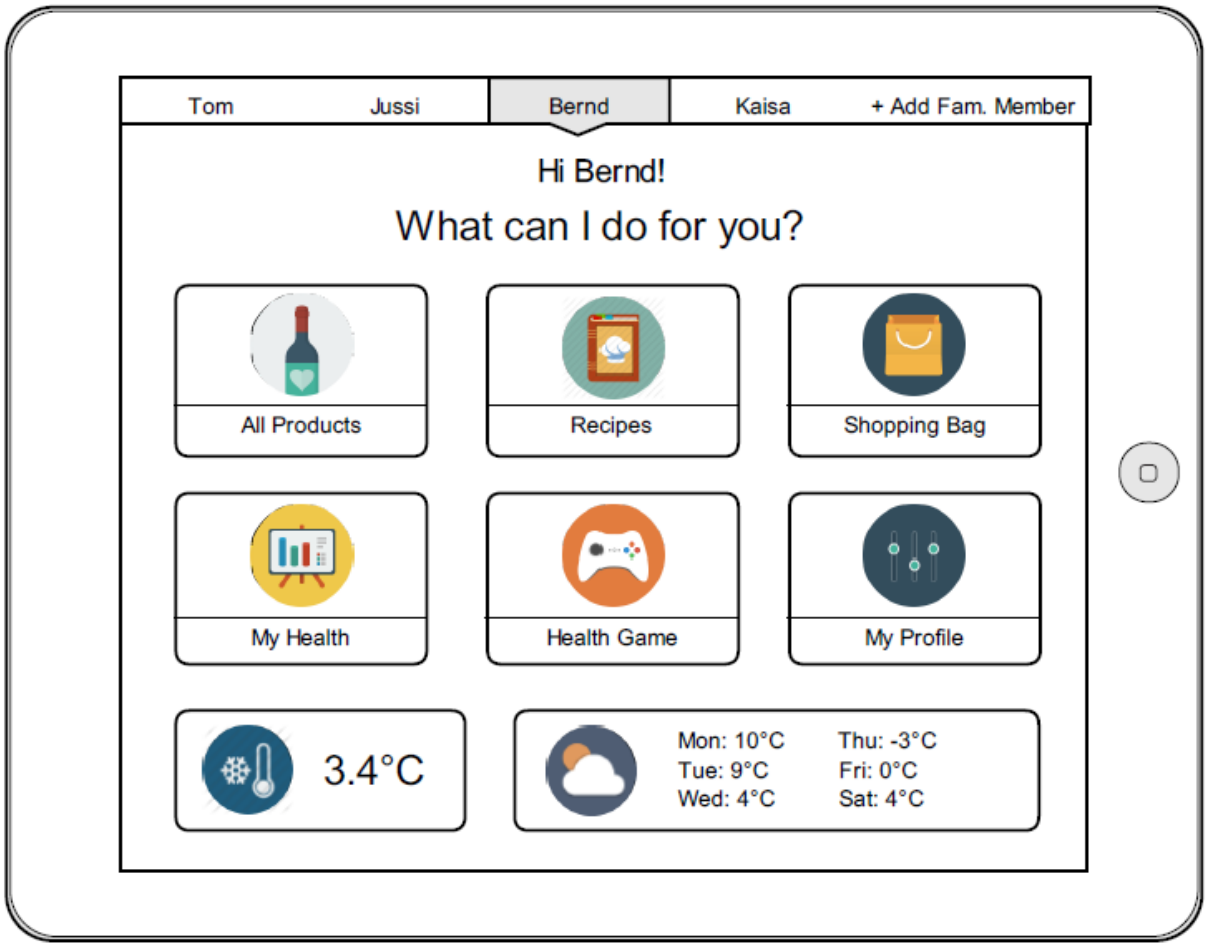
  Bootstrap is a sleek, intuitive, and powerful front-end framework for faster and easier web development, created by Mark Otto and Jacob Thornton, and maintained by the core team with the massive support and involvement of the community. Bootstrap is the most popular HTML, CSS, and JS framework for developing responsive, mobile first projects on the web.

User interface

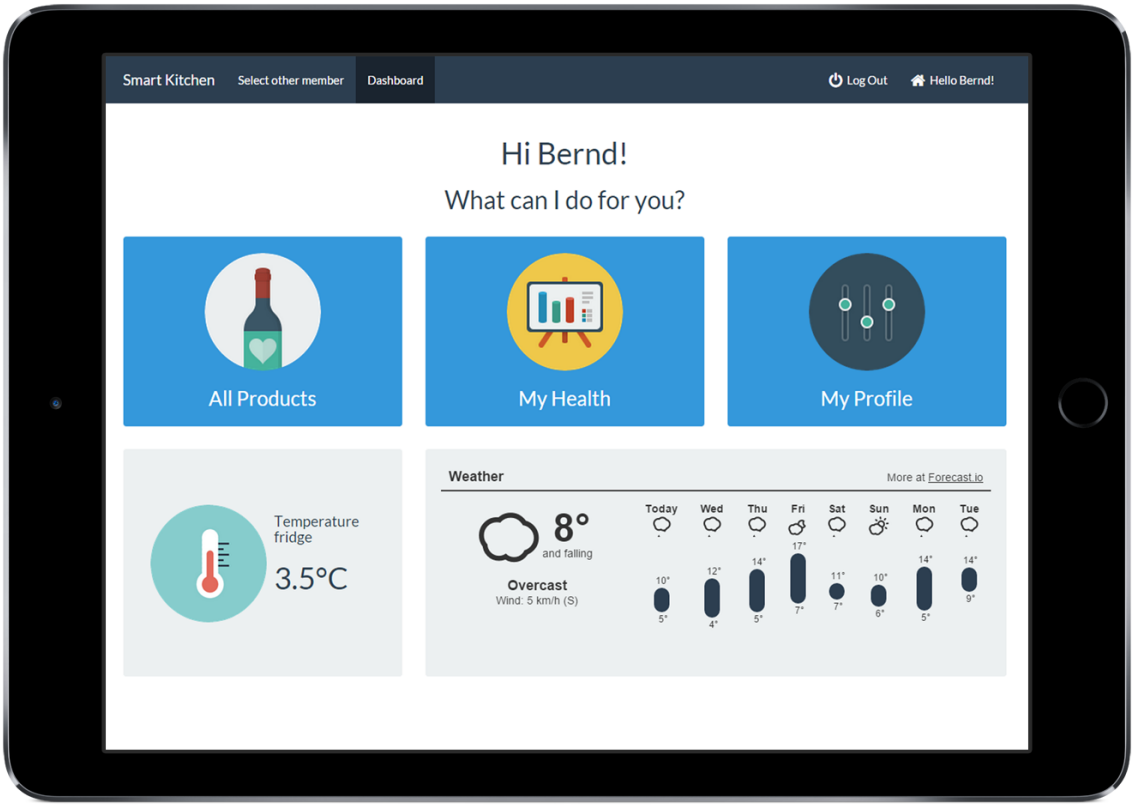
For the user interface, I started with a drawing. The drawing was a sketch with the most important elements and their shape.


Tom
Bernd
Jussi
kaisa
add family member
Hi Bernd!
What can I do for you?
Products
Recipes
Shopping
Shopping Bag
My Health
Health game
My profile
My Health
Level : 14
bronze silver gold
Bernd
Jussi
Tom
points earned : 3482
Stats
Fridge temperature: 3,4 °C
Weather Oulu
Mo 9,4°C
Tu 3,2°C

When the sketches were finished, I made them digital on http://moqups.com and they looked like this:



This design is more similar to the end design in Bootstrap (HTML and CSS). Now it's time to make it in real code. The result won't look the same but the major parts will be similar to the previous drawings.



The final design is also responsive to multiple screen resolution, from a smartphone in landscape to a very large Screen size.

GitHub

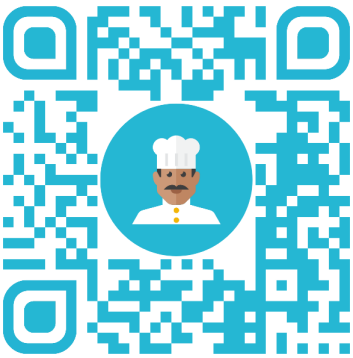
The Web Application can be found on GitHub.

https://github.com/OAMK-Smart-Kitchen/Smart-Fridge-OAMK

People can see and work on the code, add features, fix problems etc.

Windows Application

I published this project as a Windows 8 and 8.1 application to the Windows Store. You can find it on http://bit.ly/Smart-Fridge



7. Web Api 2

I did a lot of research about different back-ends, I decided to use Web API 2. This standard has a very good community platform. Microsoft has some examples and templates in Visual Studio.

The Web API 2 is running on a IIS 8.5 on a Windows Cloud Server 2012.

Data storage is on a Sql Server 2014 that runs on the same Windows Cloud Server 2012. When the Web API wants to get or store data to the Database, the Web API wants to do that with a SQL Query. It would have taken a lot of time to write all the queries. To save me some time, I used a layer between the Sql Server and the Web API controller, the Entity framework by Microsoft.

Frameworks and technologies

- Windows Server 2012 R2

  Windows Server is a stable server OS thats great to run applications of any scale. It's well supported by Microsoft. The only minor point is that you need a License to run this OS. Windows Server is known for the feature packages that can be installed, IIS web Server, Directory Server, Remote Desktop,… When you have some problems, you find a solution faster due to the verbose logs.

- Web API 2

  ASP.NET Web API 2 is a framework that makes it easy to build HTTP services that reach a broad range of clients, including browsers and mobile devices. ASP.NET Web API is an ideal platform for building RESTful applications on the .NET Framework.

- IIS 8.5

  IIS (Internet Information Server) is a group of Internet servers (including a Web or Hypertext Transfer Protocol server and a File Transfer Protocol server)

- Entity Framework

  Entity Framework (EF) is an object-relational mapper that enables .NET developers to work with relational data using domain-specific objects. It eliminates the need for most of the data-access code that developers usually need to write. I used the Model-First method where I designed a Model and converted it to a Database (T-SQL).

- SQL Server 2014

  SQL Server is a relational database management system (RDBMS) from Microsoft that's designed for the enterprise environment. SQL Server runs on T-SQL (Transact -SQL), a set of programming extensions from Sybase and Microsoft that add several features to standard SQL, including transaction control, exception and error handling, row processing, and declared variables.

GitHub

The Back end Application can be found on GitHub.

https://github.com/OAMK-Smart-Kitchen/Kitchen-WebService-OAMK

People can work on the code, add features, fix problems etc.

Technical information with problems and solutions

I created a wiki page on Github with the most important problems and the solution to those, for example how to Configure a Web API server with the Microsoft SQL Server.

https://github.com/OAMK-Smart-Kitchen/Kitchen-WebService-OAMK/wiki

8. Reference list

2015 Google - https://github.com/angular/angular.js

2015 Google - https://docs.angularjs.org/guide/introduction

2015 Margaret Rouse - http://searchsqlserver.techtarget.com/definition/SQL-Server

2015 Margaret Rouse - http://searchwindowsserver.techtarget.com/definition/IIS

2015 Microsoft - http://www.asp.net/entity-framework

2015 Microsoft - https://msdn.microsoft.com/en-us/library/dn448365(v=vs.118).aspx

2015 Microsoft - https://www.microsoft.com/en-us/server-cloud/products/windows-server-2012-r2/

2015 Twitter - https://github.com/twbs/bootstrap