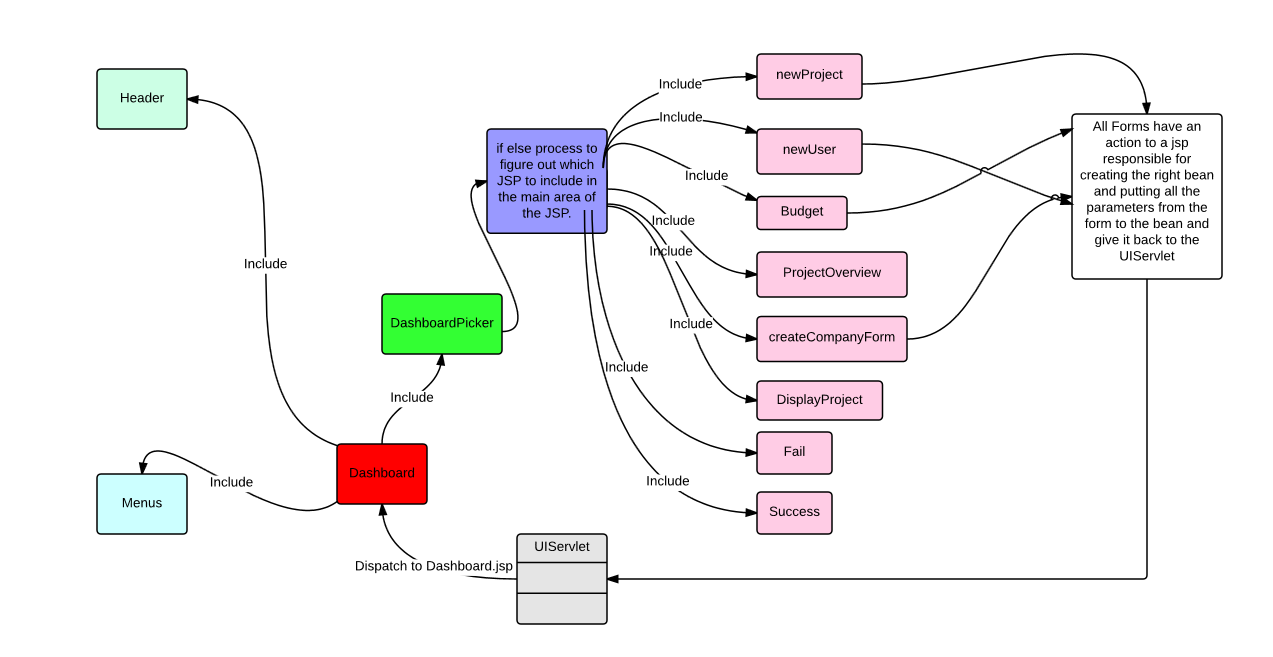
**HTML/CSS/JSP**  
  
Using JSP for this project:

Java Server Pages (JSP) in an amazing tool that can help you develop webpages in plain java. The upside is that we already know java and it was a good practice and a solid step forward in implementing Java in different environments. At start the java/HTML code seemed abstract with all the new annotations (for example : <% %> ) but once you learn/realize that this file will be converted into a servlet reverting the code and leaving java code as-is and placing HTML code with an OutputStream, it makes total sense.  
  
During the project we figured out more ways to solve a problem in a JSP and through many resources on the internet we learned more clever/efficient ways to improve our code.  
  
As a solid example of a case like that the way we represent the Dashboard after the log-in screen. In the beginning we had a discussion on how we could handle all the different pages that will obviously have the same menu item, header and footer actually the only thing that was supposed to change was the main area of the Dashboard JSP. So we agreed to use include functions in JSP.  
  
A small note on how this function works and what its syntax looks like.  
Syntax:   
<jsp:include page=”” /> And inside the page attribute goes the path(only if the JSP in not in the default folder ) and the name of the JSP we want to include in a “Parent” JSP.

The Dashboard(Picture 1) is a perfect example of a good use of that command were the code is minimized to just few lines and it gives the opportunity to split the complexity to smaller files :

(Picture 1: *Body element of the Dashboard*)

Here is a graphical representation of how our Dashboard looks and works like (Picture 3) and what the basic idea behind our design is.

The UIServlet dispatches to the Dashboard with the only exception when the the log-in process fails.

**The Dashboard**

The Dashboard (RED box) includes 3 different JSP files, the menus.jsp , Header.jsp and DashboardPicker.jsp. The first two (CYAN boxes) are concrete components of the Dashboard Page

So every time a JSP is included in another page the compiler disregards the child’s page title and basically the parent takes only the code and tags that exist inside the <body> tag(If there are commands in the <head> like imports and links to Cascade style sheet they will be executed).

**Using HTML and CSS:**

The usage of HTML(Hyper Text Markup Language) was a pleasant think to use it Is easy, and you can create a really good webpage with it. And by adding css it becomes a piece of art and this actually an under underappreciated field. With the use of JSP you can generate HTML in a page like we do in Project Overview. The opportunities unfolding in front of us are so great and kind of endless.

We made use of all three ways to add CSS to a page, and we made use of Bootstraps as well in a small extend.

The idea of having a lot of CSS files in one project was up for discussions with two opinions one side said one CSS document is more than enough for the whole project and the other side said it is ok to have three different CSS for different cases. We do not use most of the Bootstraps CSS files, but some of the already existing once where modified to match our needs.

**USEBEAN –**

UseBean:

The way we create and use java beans is through the JSP syntax :

*<jsp:useBean id=”” class=”” score=”” />*

* The ID Attribute is the name of the new Object.
* The class Attribute is the import off the JAVA class so path has to be defined there
* And the scope defines the scope you want your object to be set

As the line of code only creates an empty object (Bean) , its attributes have to be set.

And the other option is to set them one by one like this

*<jsp:setProperty name=”” property=”title” />*

* The name attribute has to match the Id of the bean

If the bean has a set method for an attribute called title it will have the name

Like : setTitle(String title) so when the jsp page will compile to a servlet page and lets say the name of the bean is book, it will be :

Book.setTitle(request.getAttribute(“title”));

*And title has to be the name of the input field in a form for example*

There is the option to set them all in one go like this:

*<* *jsp:setProperty name=”” property=”\*” />*

Then it will go through the attributes in the request and see if there are matching set methods it will set them all in the same line.

All the forms in this project (excluding the file upload form) are using this syntax to create a new bean and store it to the session object. A good example would be the newProjectHandler (Picture 2) where a lot of information from a form is placed in a bean in just two lines :

  
(Picture 2 : *creating a new bean with the name newProject and storing it to the session*)

**The Backlog**

The backlog was created really fast just before the project started and after a sort discussion in the group it was presented at the product owner before the Easter Holy Days.  
The backlog way ordered to fit the owner’s priorities and it took its final form.

1) as an admin I want an ER-diagram of the database ....

2)as a re-seller I want to be able to start a new project so I can be approved

3)as a re-seller I want to be able to edit my project so I can get approval if I got declined

4)as an admin I want to be able to see a project so I can approve it

5)as an admin I want to sort the projects so I can have a better view

6)as an admin I want to see approved projects so I can supervise them

7) as an admin I want to see money left unspent so I can allocate them elsewhere

8)as an admin I want to log-in so I can access the dashboard

9)as an admin I want to see finished projects so I can see money spent

10)as an admin I want to filter projects per steps so I can see the progress

11)as an admin I want to be able to delete projects so I can clear junk or false projects

12)as an admin I want to able to pick the currency so I can compare more clearly

13)as an admin I want to have a report with statistics per country so I can see the progress

14)as a re-seller I want to able to add media files so I can prove my work

15)as an admin I want to set the max budget per quota so I can have a clear view

16)as a partner I want to access proof of execution so I can see that our agreement is fulfilled

17)as an admin I want to be able to back-up the database so I can recover in case of data loss

As the first days were passing we were adding some new cases to the side bar like :

18)as an admin I want to create new users so they can access the system

19) If a project is finished it should be possible to have a printable document (PDF)

20)As a user I want to receive a mail if a change was made to a project I am a part of.

But not all of them made it to the actual backlog as they were forgotten or there was not enough time to implement them in the project.

(((The user stories could have been made better and by that I mean that the User stories could have been more simple and broken down to simpler more understandable stories for an actual product owner.)))

Not all user stories were fulfilled in the project due to group issues.

**DESIGN AND ARCHITECTURE**

This part of the report will be presented in two layers, the view layer and the source layer.  
  
The view layer takes care of what should be presented on the screen and what are the patterns that we use in the project.

The source layer is split in three layers the presentation layer that speaks both to the view and the source, the domain layer and the Data Source layer.   
  
**Objectives**

The main objective is to create a Web application that is able to handle and document advertisement campaigns that Dell runs in different countries in the Nordic Area, and shrink the work load from Dell’s managers.

It is a platform that partners can go and start a project and request funding, presenting their idea and providing essential information like starting and finishing dates and a description. Then the region manager can read it and make comments if necessary and approve it or reject it.

There is communication tool in every project that keeps track of the conversation so the user can go back in time and read everything without having to search in emails witch is the way it works in the current moment.   
  
There is an upload function in each project so Proof of execution can be uploaded from the Partners side and be controlled by the supervisor of the project and if the proof that the project was executed as intended the project can be marked as finished and the fund is transferred to the Partner through e-banking or external services.

So the problems solved by using this web application are:

* One place for all users to communicate and act with.
* Book keeping to some extent.
* Easy overview of projects running.
* Partners are now responsible to start a project.
* Better and easier supervision of a vast workload for managers.
* Easier proof of execution handling (No more attached files in a mail :P ).