CHALMERS

Plattformsoberoende teknik för webbsidor

Responsiv webbsida för Miljöbron

Examensarbete inom Data- och Informationsteknik

Jonas Arvidsson
David Fogelberg

Institutionen för Data- och Informationsteknik
CHALMERS TEKNISKA HÖGSKOLA
Göteborg, Sverige 2016

Responsive Website for Miljöbron

Jonas Arvidsson
David Fogelberg

Jonas Arvidsson

David Fogelberg

Supervisor at Chalmers: Uno Holmer
Supervisor at Miljöbron: Maria Rydberg

Examiner: Peter Lundin
Chalmers University of Technology
University of Gothenburg
Department of Computer Science and Engineering
SE-412 96 Göteborg
Sweden
Telephone + 46 (0)31-772 1000

Department of Computer Science and Engineering Göteborg, Sweden June 2016

Förord

Idén till projektet kom ursprungligen från en kontakt med Miljöbron, där deras behov av en modern webbsida diskuterades. Det visade sig att de såg en möjlighet att formulera ett examensarbete för att eventuellt skapa en ny och mer responsiv webbsida.

Vi vill tacka

Maria Rydberg för chansen att få bygga deras webbsida, och för alla synpunkter och engagemang i projektet.

Uno Holmer för råd och handledning under projektet.

Sebastian Lindroos för rådgivning vid val av CMS.

Sammanfattning

Miljöbron är en organisation som erbjuder projekt till studenter i samarbete med företag och på så vis ger studenter arbetslivserfarenhet. Miljöbron upplever att deras nuvarande webbsida är omodern, inte användarvänlig och tråkig att titta på. En ny webbsida har skapats från grunden åt Miljöbron som skall ge en mer modern bild av organisationen samt attrahera studenter och företag till att ta kontakt med organisationen. Webbsidan bygger på CMS:et Wordpress, då ett stort krav på webbsidan är att den ska vara lätthanterlig för personer som saknar programmeringskunskaper. Rapporten går igenom hur en responsiv webbsida utvecklas med hjälp av moderna webbutvecklingsverktyg.

Abstract

Miljöbron is an organisation that provides a way for students to gain work experience by providing projects for students in collaboration with companies. Miljöbron feel that their current website is outdated, not user-friendly and boring to look at. A new website was created from scratch for Miljöbron that will provide a more modern image of the organisation and connect students and companies by contacting the organisation. The website is based on the CMS called Wordpress since it should be easy to use by people who lack programming knowledge. The report covers how a responsive website is built with modern web development tools.

Beteckningar

AngularJS: Klientbaserat JavaScriptramverk som skapar struktur för dynamiska webbapplikationer.

API : Står för applikationsprogrammeringsgränssnitt och är ett gränsnitt mellan applikation och bibliotek.

AJAX : Asynkronisk JavaScript + XML, används till att binda ihop olika teknologier.

CMS: Innehållshanteringssystem som används till att redigera webbsidor.

CSS : Cascading Style Sheets, beskriver presentationsstilen för ett strukturerat dokument.

DOM : Document Object Model, ett applikationsprogrammeringsgränssnitt som bestämmer hur dokument får manipuleras, nås och hur dess logiska struktur får se ut.

HTML : HyperText Markup Language, webbens standard märkspråk för hypertext.

IDE : Integrerad utvecklingsmiljö.

JavaScript : Klientbaserat scriptspråk som främst används i webbapplikationer.

JSON : Textbaserat format som används för att utbyta data.

JQuery : JavaScript-bibliotek.

MVW : Model-view-whatever och är ett arkitekturmönster för ramverket AngularJS.

Partial : Gemensamt namn för de HTML-sidor som används till att uppdatera webbsidans centrala del.

Responsiv: en term som används för att beskriva webbdesign där innehållet på en webbsida anpassas beroende på enhetens skärmstorlek.

Scope : Ett objekt som referar till applikationens modell.

XHTML : Extensible Hypertext Markup Language och som är en vidareutvecklingen av märkspråket HTML.

XML : Extensible Markup Language, ett märkspråk som är universellt och utbyggbart.

XMLHttpRequest API : API som används till att skicka Http - och Https-meddelanden till servrar via klientbaserade scriptspråk t.ex. JavaScript.

Innehållsförteckning

Inledning

1.1 Bakgrund

Miljöbron är ett företag som fungerar som en länk mellan näringslivet och den akademiska världen. Miljöbron finns i Skåne och Västra Götaland, med kontor i Helsingborg, Lund, Göteborg, Trollhättan och Borås. Deras arbete går ut på att hjälpa studenter och företag att komma i kontakt med varandra genom att förmedla olika projekt. Projekten kan handla om en rad olika ämnen men den gemensamma faktorn är att projekten bidrar till att skapa en hållbar utveckling. De färdiga projekten läggs upp på Miljöbrons webbsida där studenter kan läsa om tidigare projekt för att inspireras och komma på nya idéer. Vi kommer utföra vårt uppdrag för hela Miljöbron Västra Götaland men kommer enbart ha kontakt med kontoret i Göteborg. Miljöbron har under den senaste tiden insett att deras nuvarande webbsida har problemen genom att den ej är responsiv, är omodern, inte särskilt användarvänlig och tråkig att titta på. Man bestämde sig för att skapa ett projekt, avsett för studenter, med målet att ta fram ett förslag till förbättring av den egna hemsidan.

1.2 Syfte

Syftet med projektet är att utveckla en ny webbsida för Miljöbron som är responsiv, modern, användarvänlig och väldesignad.

1.3 Mål

Projektet skall utveckla prototypen till en ny websida som skall vara responsive, modern och användarvänlig.

Webbsidan ska utvecklas med ett billigt CMS, gärna samma som man använder för dagens webbsida, så att Miljöbron kan använda webbsida i framtiden. Websidan ska fungera ihop med Miljöbrons portalwebbsida och ha en engelsk översättning.

1.4 Avgränsningar

Rapporten begränsas till att enbart diskutera och jämföra PHP baserade CMS. Samtidigt kommer vi endast använda gratis verktyg för att utveckla webbsidan. Själva mjukvaran kommer endast att testas och köras i Windowsmiljö.

2. Metod

För att uppnå målen kommer vi att använda oss av *Scrum* som är en metod för uppdelning av arbetsuppgifter inom systemutveckling. *Scrum* passar bra eftersom vår handledare på Miljöbron kommer att vara produktägare medan vi turas om att vara *Scrum* master varje vecka. Scrum mastern kommer att gå igenom varje veckas arbetsinsats och även hålla kontakt med produktägaren.

För att underlätta utförandet av *Scrum* kommer vi att använda oss av onlineverktyget *Trello*, där man enkelt kan skriva upp uppgifter man arbetar på, vad som måste göras, deadlines för sprintar och mycket annat som underlättar arbetet. Vi tänker även föra dagbok som ska fyllas i dagligen och försöka träffas fem gånger i veckan då vi jobbar tillsammans. För att säkra kvalitén på produkten och samtidigt göra kunden nöjd, kommer vi att dela upp produkten i tre olika prototyper. Varje prototyp representerar en komponent eller ett lager och kommer under fasta datum (se bilaga **Gant Schema**) att redovisas för kunden. Under demonstrationen kan kunden ge synpunkter och önskemål kring vad som bör förbättras eller ändras.

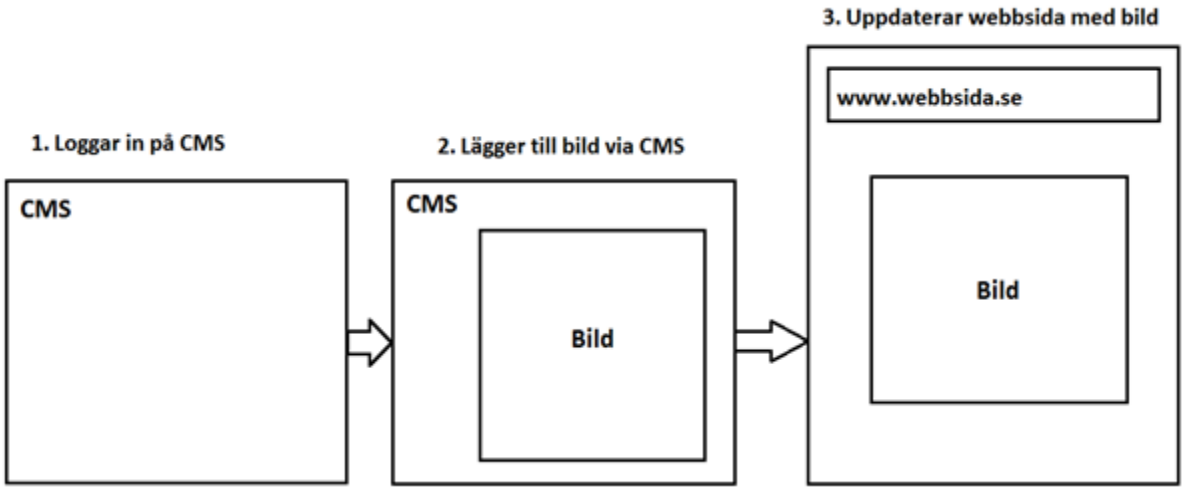
Den mjukvara som vi kommer använda är ett *Wordpress CMS* som diskuteras senare i rapporten om varför vi valde den och en *Wampserver* där en apache server väljs för lagring av data. Apache servern kommer att hanteras med databashanteraren MySQL. Mjukvaran kommer att köras som ett lokalt projekt i *Netbeans*. Det kommer att krävas en del tid för att sätta ihop mjukvaran till ett körbart projekt eftersom det behöver läsas på om hur mjukvaran fungerar och testa den med exempelkod. För versionshantering av projektkoden kommer vi använda *Git*.

3. Teori

Detta kapitel kommer ta upp vad ett CMS är för något och vad det används till. Kapitlet kommer att fördjupa sig inom CMS:en *Wordpress*, *Drupal* och *Joomla*. Det kommer även att tas upp olika ramverk, arkitekturmönster och programmeringsspråk som används ihop med CMS:en, samt tas upp en miljöaspekt i slutet av kapitlet.

3.1 Vad är ett CMS ?

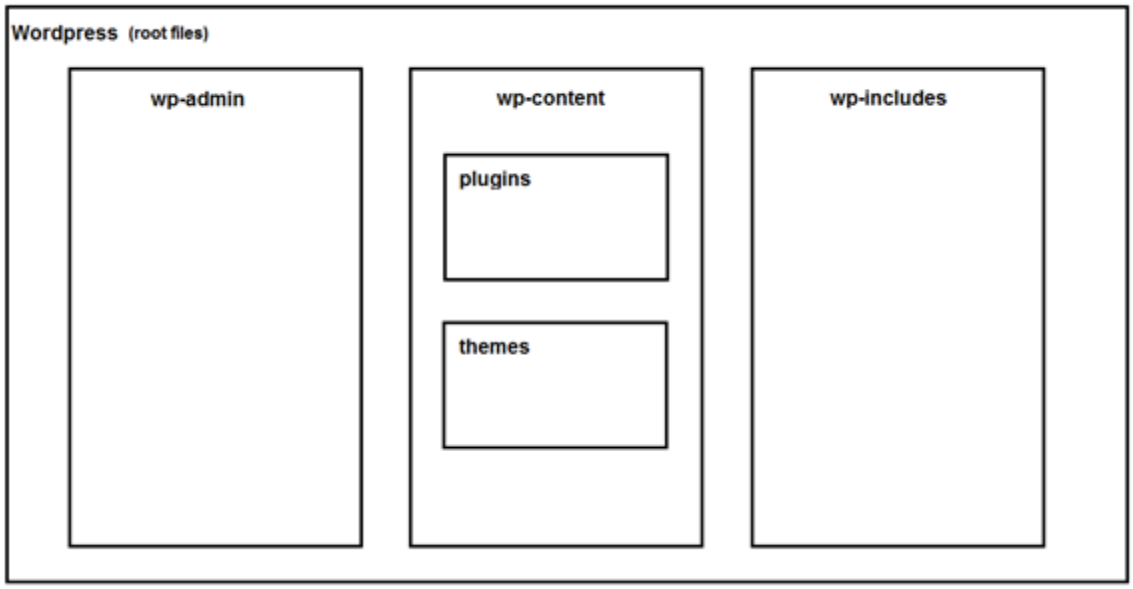
CMS står för *Content Management System* eller innehållshanteringssystem, och låter en användare redigera och uppdatera en webbsida utan att behöva kunna programmera. Istället för att använda sig av HTML5, JavaScript och PHP, kan användaren lägga till och ta bort både text och bilder utan att behöva röra koden [1] (se figur 3.1).



Figur 3.1 Exempel på hur en bild läggs till på en webbsida via ett CMS.

3.2 Wordpress

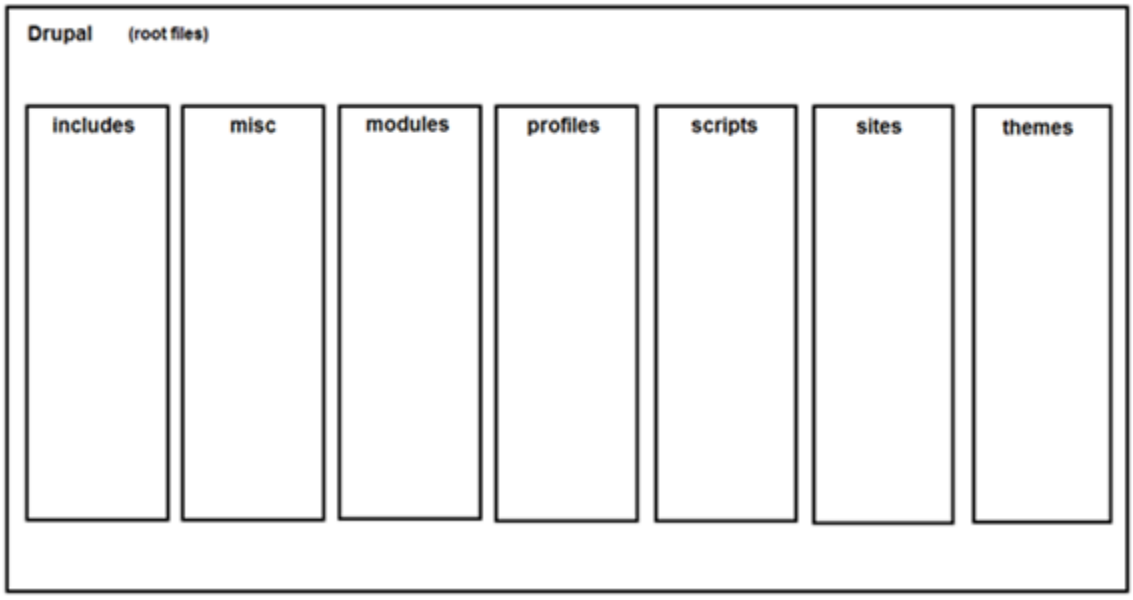
Wordpress grundades år 2003 och har sedan dess vuxit till världens mest använda CMS med ett community som består av flera miljoner användare. Det är känt för sin enkla användning och är uppdelat i två tjänster utifrån användarens behov. Wordpress.com ger användaren möjligheten att skapa en ny gratis webblogg direkt på nätet genom att skapa ett konto och välja ett färdigt tema. Wordpress.org fokuserar istället på att användaren ska få tillgång till Wordpresskoden och utveckla sitt eget tema från grunden genom att ladda ner ett projekt lokalt på datorn. Wordpressprojektet är uppdelat i mapparna wp-admin, wp-content och wp-includes (se figur 3.2). I wp-admin finns alla administrativa kodfiler som anger uppbyggnadsstrukturen av CMS:et och alla funktioner som användaren kan använda t.ex. göra ett nytt inlägg. Wp-content innehåller alla plugins som användaren lägger till separat och alla standard teman. Det är i wp-content inuti mappen themes som användaren kan skapa sitt eget tema och utveckla det från grunden. Wp-includes innehåller alla kodbibliotek som t.ex. JQuery, stödfunktioner som t.ex. databas hantering av inlägg och tema funktioner [30].
Det är också “open source” och följer GPL-licensen [5] vilket ger användaren friheten att få använda Wordpress projektet i kommersiella verksamheter och få modifiera projektkoden [2].
Wordpressprojekt kräver ett webbhotell som stödjer språket PHP och ett databas-hanteringssystem, antingen MySQL eller MariaDB. För att lagra data krävs också antingen en apache- eller Nginx server [3]. Wordpress erbjuder många användbara funktioner och räcker inte dessa kan användaren välja mellan mer än 20 000 plugin:er som utvidgar möjligheterna ytterligare [4].



Figur 3.2 WordPress-projektets mappfördelning.

3.3 Drupal

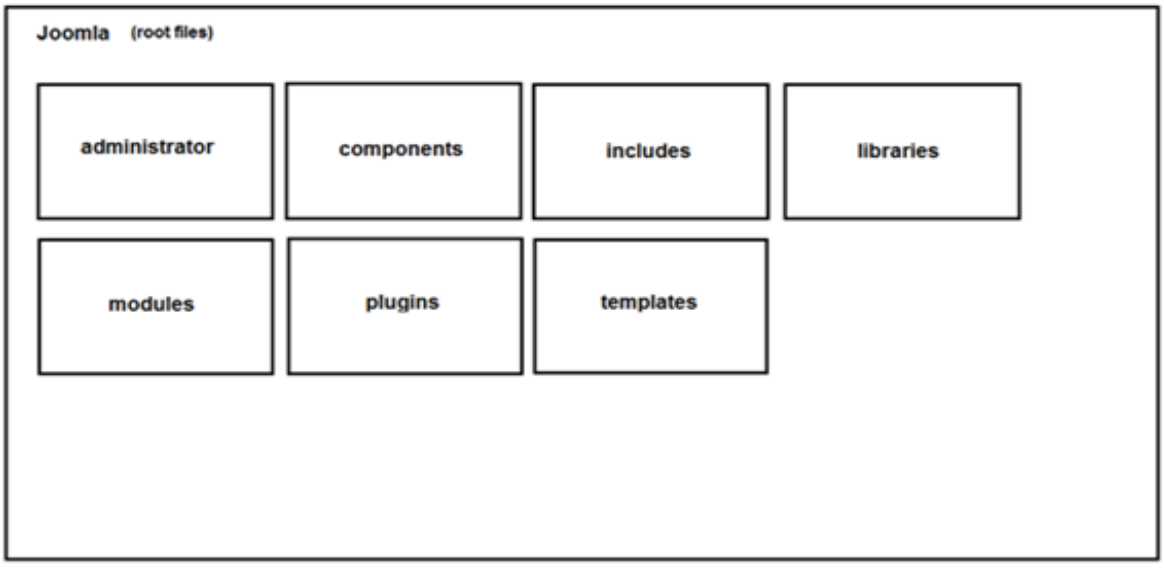
Drupal släpptes i januari år 2001 och är idag en av de mest använda CMS:en i världen. Det är “open source” och har ett community som består av cirka en miljon utvecklare, designers, utbildare, strateger, samordnare, redaktörer och sponsorer. CMS:et ebjuder ingen tjänst där användaren kan skapa en gratis webblogg direkt på nätet utan erbjuder istället en demo där själva CMS:et kan testas. Det krävs att användaren istället laddar ner Drupal som ett lokalt projekt vilket kräver programmeringskunskaper i PHP och HTML . Drupal-projektet är uppdelat i mapparna includes, misc, modules, profiles, scripts och themes (se figur 3.3). Includes innehåller alla hjälpfunktioner som t.ex. lösenordsgenerering, misc innehåller all JavaScripts och ikoner, modules innehåller alla moduler som utvidgar Drupals funktionalitet, profiles innehåller alla installationer för Drupals profiler, scripts innehåller olika skripter t.ex. för att generera hash för lösenord och themes innehåller alla standardteman. Det är i theme som användaren kan skapa sitt eget tema och utveckla det ifrån grunden [31]. Det krävs också ett webbhotell för att kunna lägga ut Drupal-webbsidan på nätet som t.ex. Bluehost [6], vilket kräver ett domännamn som kostar pengar. Projektet får användas enligt GPL-licensen vilket ger användaren möjligheten att få redigera koden och använda Drupal i kommersiella verksamheter. Drupal är känt för sin pålitliga prestanda, säkerhet, skalbarhet och flexibla moduler [7].



Figur 3.3 Drupal-projektets mappfördelning.

3.4 Joomla

Joomla släpptes år 2005 och är baserat på PHP. Det används till att utveckla alla typer av webbsidor och internetapplikationer. CMS:et är en av de populäraste idag och har ett community som består av ca tvåhundratusen utvecklare. Användaren kan välja mellan två tjänster antingen “Joomla.com” eller “Joomla.org”. “Joomla.com” ger användaren möjligheten att skapa en gratis webblogg direkt på nätet utan kännedom om programmering. Användaren behöver endast registrera ett konto och välja ett färdigt utvecklat tema [8]. “Joomla.org” ger istället användaren möjligheten att skapa sin egen webbsida genom att installera Joomla som ett lokalt projekt. Joomlaprojektet är uppdelat i mapparna administrator, components, includes, libraries, modules, plugins och templates (se figur 3.4). I administrator finns alla filer som bygger upp Joomla:s webbgränssnitt. Components innehåller komponenter [32]. Komponenter kan beskrivas som komplexa mjukvarutillägg och används till att utvidga en webbsidas funktioner som t.ex. att lägga till ett nyhetsflöde. Modules innehåller moduler som också är mjukvarutillägg men är enklare än komponenter t.ex. för att visa bara de senaste nyheterna. Plugins innehåller avancerade mjukvarutillägg men inte lika komplexa som komponenter [33]. Includes innehåller Joomlas kärnfiler och libraries innehåller alla kodbibliotek som t.ex. JQuery. Templates innehåller alla standardteman som följer med och det är även här användaren skapar sitt egna tema [32]. Projekten är “open source” och laddas ner lokalt på datorn. Användaren kan redigera och använda Joomla-projektet enligt GPL licensen, vilket innebär att projektkoden får modifieras och användas i kommersiella verksamheter. Räcker inte resurserna som erbjuds i projektet kan användaren ladda ner Joomlas ramverk som utvidgar möjligheterna. Ramverket erbjuder bland annat lagerstyrningssystem, rapportering av data, applikationsbroar, anpassade produktkataloger, integrerade e-handelssystem, bokningssystem, kommunikationsverktyg och företagskataloger [9].



Figur 3.4 Joomla-projektets mappfördelning.

3.5 HTML

HTML står för HyperText Markup Language, och används för att utveckla webbsidor. HTML, tillsammans med CSS och JavaScript utgör de vanligaste språken som används för att utveckla webbsidor. HTML kan använda JavaScript för att förändra en webbsidas beteende, och CSS för att ändra utseendet på en webbsida. HTML kan skrivas i vilket textredigeringsprogram som helst och kan tolkas av webbläsare. Ett HTML-dokument består av element som representerar olika delar av en webbsida, som visas i figur 3.5 [19].

```
<!DOCTYPE html>
<html>
    <head>
        <title>Page title</title>
    </head>
    <body>
        <div>Page content</div>
    </body>
</html>
```

Figur 3.5. Grundläggande HTML-kodstruktur för en hemsida.

3.6 CSS

CSS står för Cascading Style sheets, eller *stilmall* på svenska, och används för att beskriva hur innehållet på en webbsida ska se ut och presenteras. CSS är designat för att separera ett dokuments innehåll från dess utseende, som textstorlek, färger och fonter. CSS kan användas i samband med HTML och kan styra utseendet på flera webbsidor samtidigt, och därmed spara mycket tid. CSS möjliggör anpassningen av webbsidor till olika plattformar, som telefoner, surfplattor och datorer. CSS kan användas oberoende av HTML och sparas då i .css filer. Figur 3.6 visar hur CSS kan användas för att ändra färg och storlek på olika delar av en webbsida [20].

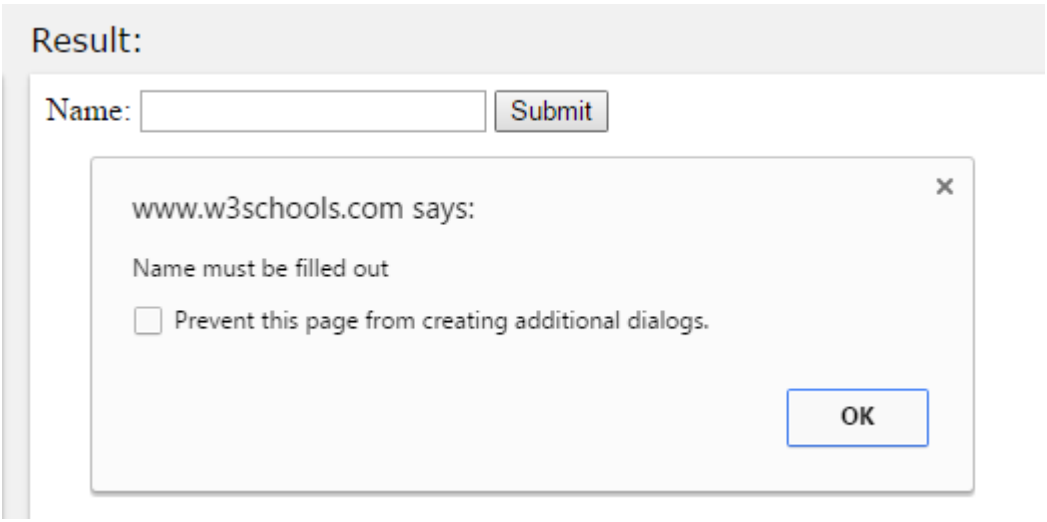
```
header {
    background-color: gray;
}

p {
    font-size: 10px;
    color: gray;
}
```

Figur 3.6 CSS-kod som ändrar färg och storlek på två olika element.

3.7 JavaScript

JavaScript är ett scriptspråk som används för att ändra beteendet på en webbsida men kan användas för andra syften än webb. JavaScript är klientbaserat, vilket betyder att koden exekveras på klientens dator och tar därmed inte resurser från webbservern. Detta gör att webbsidor med JavaScript är snabba för användarna och sparsamma för webbservrar [21], eftersom koden inte behöver ta kontakt med någon server.
Då JavaScript exekveras direkt på klientens dator kan skadlig kod köras direkt på datorn, vilket utgör en stor säkerhetsrisk. Webbläsare erbjuder visst skydd mot skadlig JavaScript-kod, men det är fortfarande möjligt för sådan kod att implementeras på webbsidor [22]. Att JavaScript utförs på användarens dator betyder även att man som utvecklare inte vet exakt hur koden och webbsidan kommer att uppföra sig på alla datorer.
JavaScript gör det även möjligt för användaren att interagera med en webbsida utan att behöva ladda om sidan. I exemplet som syns i figur 3.7 har användaren tryckt på *Submit* utan att ha fyllt i någon information, ett JavaScript aktiveras då på webbsidan och varnar användaren utan att behöva ladda om sidan. Detta sparar på bandbredd och gör webbsidan snabbare genom att inte behöva hämta sidan på nytt [23].



Figur 3.7 Varningsruta skapad genom JavaScript.

3.8 JQuery

JQuery är ett gratis JavaScript-bibliotek som underlättar användningen av JavaScript för webbapplikationer. JQuery är tänkt att göra skrivandet av JavaScript enklare genom att ta vanliga JavaScript-funktioner som i vanliga fall skulle kräva flera rader kod att implementera och istället skriva de som enstaka metoder [39]. Med den enkla JQuery-syntaxen kan även utvecklare med mindre kodkunskap använda JavaScript. Att JQuery tillåter webbsidan att ha mindre kod gör även att webbsidan laddas snabbare.
I figur 3.8 visas hur man med JavaScript kan ändra färgen på en del av webbsidan. I figur 3.9 visas hur samma sak kan göras med JQuery [40].

```
function changeBackground(color) {
   document.head.style.background = color;
}

 onload="changeBackground('green');"
```

Figur 3.8 JavaScript-kod som ändrar bakgrundsfärg

```
$('head').css('background','green');
```

Figur 3.9 JQuery-kod för att ändra bakgrundsfärg

3.9 PHP

PHP är ett skriptspråk som kan användas vid webbutveckling för att göra webbsidor interaktiva, men kan även användas för andra syften. PHP stod från början för *Person Home Page*, men står nu för *Hypertext Preprocessor*[24]. PHP är ett scriptspråk på serversidan vilket betyder att till skillnad från JavaScript så utförs koden på servern, istället för hos klienten. Det betyder att PHP underlättar överföringen av data från server till webbläsare och kan användas för funktioner i webbapplikationer som verifiering, sparande och hämtning av information och navigering [25].

3.10 Responsiv design

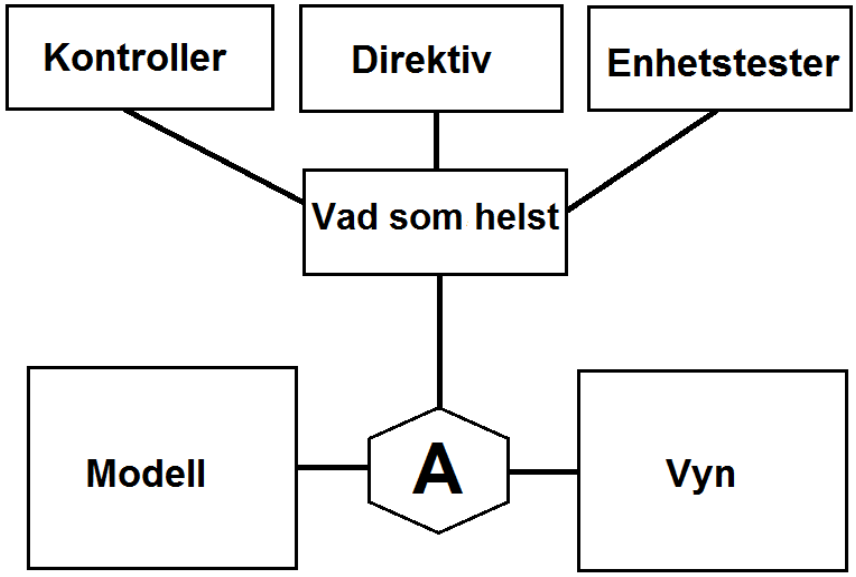
Responsiv design är en term som används för att beskriva webbdesign där innehållet på en webbsida anpassas beroende på skärmstorlek på den enhet som användaren har, exempelvis en dator, mobil eller surfplatta. Istället för att skapa olika varianter av en webbsida för olika skärmstorlekar så skapas webbsidan från början med responsiv design. När man bygger en webbsida med responsiv design går man från att använda pixlar för att definiera avstånd och använder istället procent, för att webbsidan skall anpassa sig efter olika skärmstorlekar [26].

3.11 Bootstrap

Bootstrap är ett gratis ramverk baserat på HTML, CSS och JavaScript. Ramverket används för att underlätta utvecklingen av responsiva webbsidor, vilket betyder att Bootstrapkomponenterna automatiskt justeras efter den apparat som används för att visa innehållet. Bootstrap är även anpassat för att personer med hög såväl som låg kunskap skall kunna använda det [27]. Istället för att själv lägga stor tid på att göra varje komponent på webbsidan responsiv så kan Bootstrap användas för att underlätta jobbet. Bootstrapbiblioteket innehåller HTML- och CSS- former, knappar, navigationsverktyg och andra gränssnittskomponenter. Dessa komponenter är sådana som en webbsida kan förväntas ha, exempelvis navigeringsfält, varningsmeddelanden och förloppsindikatorer. Bootstrap ger en del stilmallar som definierar stilen på HTML-komponenterna, vilket gör att en webbsida byggd på Bootstrap kommer ha ett konsekvent och modernt utseende. I Bootstrapbiblioteket finns även JavaScript-komponenter som exempelvis dialogrutor och bildspel. Dessa komponenter är också mobilanpassade, vilket gör implementeringen av exempelvis responsiva bildspel eller andra dynamiska element väldigt enkel [42].

3.12 MVW

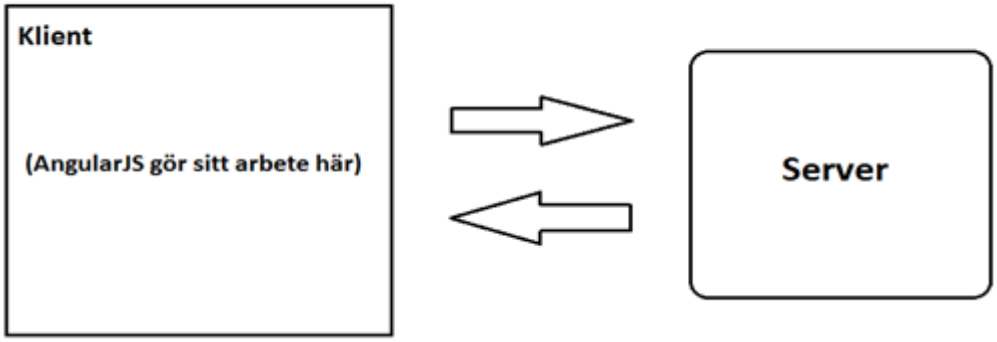
MVW står för model-view-whatever och är ett arkitekturmönster för ramverket AngularJS som kommer att diskuteras senare. Arkitekturmönster är uppdelat i tre delar, det vill säga en modelldel, en vy-del och en vad som helst-del (se figur 3.10). En vy presenterar data och är uppbyggt med HTML som baseras på DOM-strukturen. En modell är ett JavaScriptobjekt som uppdaterar vy-delen med data. “Vad som helst”-delen är ett gemensamt namn för de komponenter som kan manipulera modelldelen t.ex. kontroller, direktiv och enhetstester. En kontrolls uppgift är att skapa modellen och binda den till vy-delen. Ett direktiv används till att skapa nya beteenden eller transformera DOM-trädet och enhetstester används till att testa individuella kodenheter [41].



Figur 3.10 Arkitekturmönstret MVW för AngularJS.

3.13 AngularJS

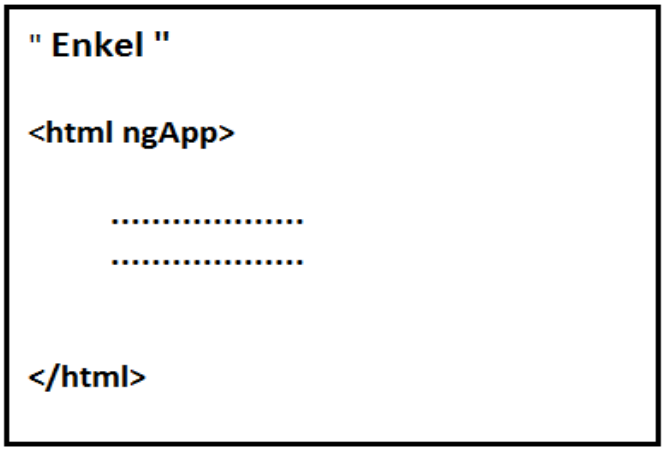
AngularJS är ett “open source” JavaScript-ramverk som är klientbaserat och skapar struktur för dynamiska webapplikationer. Ramverket kan kombineras med HTML till att skapa applikationsmallar och dess komponenter. Allt AngularJS gör som t.ex. databindning sker inom webbläsaren vilket minskar påfrestningen på servern och gör ramverket kompatibelt med vilken server som helst (se figur 3.11) [10].



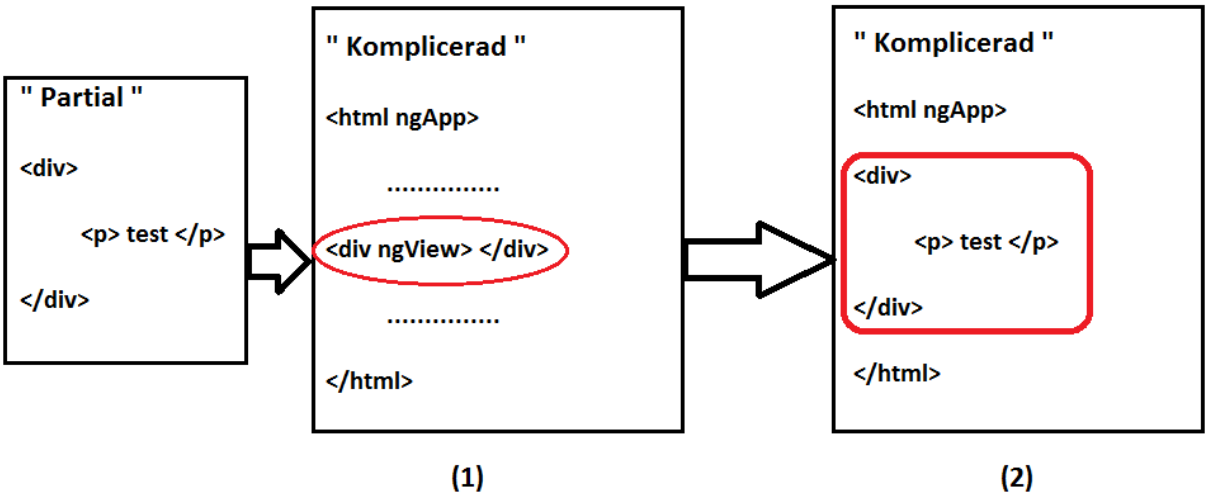
Figur 3.11 AngularJS är klientbaserad.

Mallar

AngularJS använder sig av mallar som utgör vyn i dynamiska webbapplikationer vilket innebär att de kombinerar information ifrån modell- och kontrolldelarna som därefter visas i webbläsaren. En mall konstrueras med CSS, direktiver, markups och HTML som följer DOM-mönstret. En enkel webbapplikation använder sig oftast bara av en mall, det vill säga en HTML-fil som utgör en vy-del (se figur 3.12). En komplicerad webbapplikation består däremot av flera vy-delar genom att kombinera en mall med olika segment som kallas för partials. Partials är separata HTML sidor som anges med webbapplikationens url och matas in i mallens kod vid direktivet ngView som uppdaterar vyn (se figur 3.13) [38].



Figur 3.12 En enkel webbapplikationsmall.



Figur 3.13 En partial matas in i mallen vid direktivet ngView som därefter uppdaterar vyn.

Direktiv

Direktiven anger olika egenskaper åt olika attribut eller element t.ex. ngApp och ngView. Direktivet ngApp placeras oftast innanför <body> eller <html> taggarna och anger automatisk bootstrap vilket innebär automatisk initieringen av webbapplikationen. Den bestämmer även rotelementet för webbapplikationen, det vill säga punkten där initieringen börjar (se figur 3.14) [11]. Direktivet ngView placeras innanför <body> taggarna och används till att ändra utseendet på applikationen baserat på dess url. När url:en ändras inkluderar ngView den HTML kod (partial) som gäller för den url:en och uppdaterar sidans utseende [12].

```
< html ngApp >

<body>

    <div ngView> </div>

</body>

</html>
```

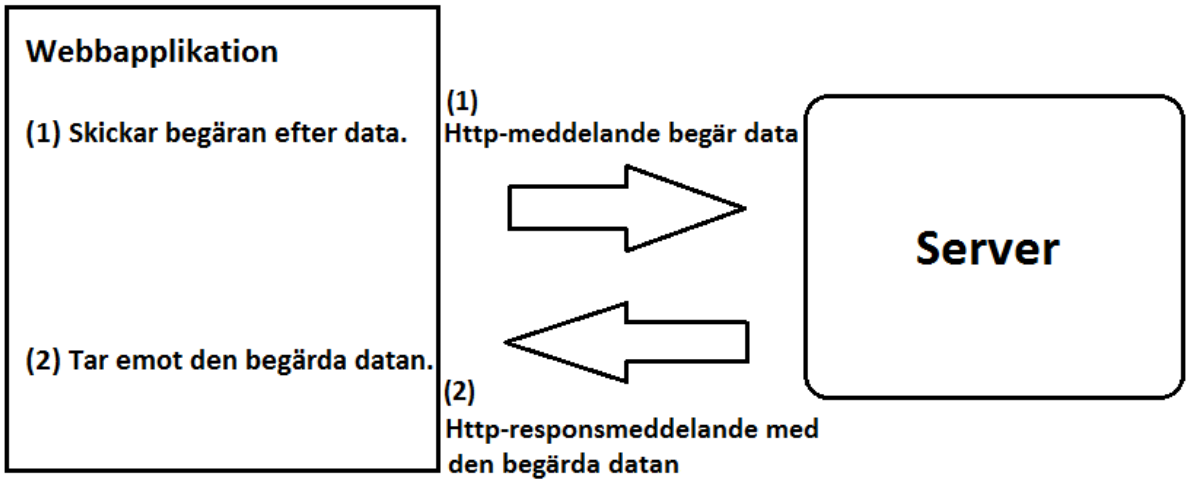
Figur 3.14 ngApp placeras innanför html-taggen och ngView placeras mellan body-taggarna.

Modul

Efter att AngularJS initierats kommer AngularJS att leta efter ngApp-direktivet och ladda dess tillhörande modul och kompilera DOM-trädet. En modul är en behållare för alla komponenter som t.ex. services, kontroller och filter[13].

Services

Services är ersättningsbara objekt som används till att dela och organisera kod mellan applikationer. De binds ihop genom att använda “dependency injection” (DI) som är ett mjukvarumönster som bestämmer hur komponenter får tag på sina beroende [14]. Ett exempel på en service är Http som används till att förmedla och begära information med JSONP eller med XMLHttpRequest mellan externa Http servrar (se figur 3.15). Servicen skickar ett Http-meddelande till servern i begäran av information som därefter servern tolkar och svarar på genom att skicka tillbaka ett Http-responsmeddelande med den begärda informationen [15].



Figur 3.15 Http-service som kommunicerar med en extern server.

Kontroller

En kontroll är en komponent som skapar modellen och binder den till vyn vilket introducerades i rapportens MVW kapitel. I AngularJS initieras den automatiskt genom att ange ett scopeobjekt som parameter i dess konstruktor funktion (se figur 3.16). Scopeobjektet används till att ordna modellen till vyn och skicka händelser till kontrollen [16].

```
angular.module('app').controller('Main', function($scope, $http)
{
        // vad kontrollen ska göra anges här!

        $http.get('api/get_posts').success(function(res){
        $scope.postss = res.posts;

        }
}
```

Figur 3.16 Kodexempel på en AngularJS kontroll där kontrollen hämtar aktuella inlägg på en WordPress webbsida med hjälp av tjänsten http.

Filter

Ett filter används till att formatera och ändra data som t.ex. att alla länkar ska öppnas i nya fönster istället för att öppnas på den aktuella sidan. De kan användas i mallar, kontroller och services. Det är enkelt att skapa egna filter och därefter registrera dem i modulen [17].

3.14 Miljöaspekten

Det första man tänker på när man ser en webbsida är antagligen inte att den är miljövänlig, men det finns sätt för en webbsida att minska sin miljöpåverkan. Det är beräknat att IT sektorn står för 2-2.5% av det globala utsläppet av växthusgaser[28]. För att hjälpa till att minska detta kan man göra sin webbsida mer miljövänlig med en del olika metoder. Man kan tex hålla webbsidan uppdaterad, ta bort gamla bilder eller annat innehåll som inte längre används. Genom att hålla webbsidan fri från gammalt innehåll kan serverutrymme sparas vilket leder till att färre servrar behövs för att lagra webbsidor. Ett annat sätt att spara energi är att göra sin webbsida snabb. Varje förfrågan till en server kräver energi, och ju längre tid det tar, desto mer energi förbrukas av servern [29]. Alltså genom att ha en snabb webbsida sparas en liten del energi vilket ger en lägre miljöpåverkan.

4. Genomförande

4.1 Problemanalys

Miljöbrons krav är att få en responsiv webbsida som stödjer de populäraste plattformarna och som samtidigt ska kunna underhållas utan programmeringskunskaper. Webbsidan ska också vara enkelt att använda, fungera på deras portalsida och ha en engelsk översättning. Några övriga önskemål är att webbsidan ska kännas modern, enkel att publicera nya inlägg i, kunna sortera olika studentprojekt, visa Instagrambilder och ha ett nyhetsflöde på första sidan.

4.2 Planering

Själva planeringen av arbetet skedde genom att ha möten ungefär fyra dagar i veckan där uppgifter tilldelades och gemensam programmering förekom. Det användes en del verktyg för att underlätta planeringen av arbetet. Såsom Github, Trello, Gant Schema och Google drive. Ett privat Github skapades för versionshantering av koden. Trello användes för att skapa användar fall tillsammans med deluppgifter som underlättade utförandet av Scrum-modellen. Deluppgifterna skulle vara klara vid varje sprint som var två veckor långa och utgjorde en prototyp. Gant Schema användes till att uppskatta tiden som behövdes för utvecklingen av varje prototyp, testning, säkerhet och daglig dokumentation. Google drive användes till att dela och lagra alla dokument och text filer. Olika mappar användes till att skapa struktur för dokumentationfilerna som gamla filer, inlämnade filer och daglig dokumentation.

4.3 Utveckling

4.3.1 Val av plattform

Vid utvecklingen av webbsidan behövdes ett CMS för att kunna redigera webbsidans innehåll utan att ändra direkt i koden. Inledningsvis gjordes en teoretisk jämförelse mellan tre olika system (Se 6.1). Valet blev Wordpress eftersom Miljöbron hade erfarenhet av verktyget och för att det uppfyller övriga krav som ställdes. Det behövdes också en IDE som var enkel att använda ihop med Wordpress. Valet blev Netbeans eftersom den har ett Wordpresstillägg och stöd för MySQL-databaser, som är standard för Wordpress. Netbeans stödjer också PHP och AngularJS vilket var ett krav som ställdes.

4.3.2 Installation av plattform

Utvecklingen började med att sätta upp ett projekt i Netbeans tillsammans med Wordpress- mjukvaran. Ett Netbeans-tillägg installerades för att göra Wordpress kompatibelt med Netbeans. För att kunna köra projektet lokalt installerades en WampServer. I WampServern skapades en MySQL databas via verktyget phpMyAdmin. Vid första körningen av projektet skapades ett personligt användarkonto till Wordpress

admin för att få tillgång till alla redigeringsverktyg (skriva inlägg, ta bort inlägg, ändra inlägg, lägg till ny meny etc..).

4.3.3 Prototyp UI

I denna prototyp skapades webbsidans tema med dess kodstruktur och dess visuella design. Det var även under denna utvecklingsfas som alla externa ramverk importerades.

- Tema

Ett nytt blankt tema skapades inuti Wordpress-projektets “theme mapp” som fick namnet MiljobronTheme. Standardfiler skapades inuti temat, det vill säga index, style och functions för att temat skulle kännas igen av Wordpress, där det manuellt aktiverades som det aktuella utvecklingstema.

- Kodstruktur

Webbsidans kodstruktur delades upp i olika HTML-sektioner. En header-sektion, en container-sektion, en sidebar-sektion och en footer-sektion. Varje sektions namn representerades som ett unikt ID i koden. Detta ID användes också inuti CSS-filen till att justera sektionernas positioner och deras utseende. Alla sektionerna placerades inuti huvudsektionen structure som begränsar positionerna av de andra sektionerna. Detta gjorde det enkelt att ändra storlek och position på de andra sektionerna utan att förstöra webbsidans struktur (se figur 4.1).

```
<body>

<div id = "structure">

<div id = "header"> kod för header här </div>

<div id = "container"> kod för container här </div>

<div id = "sidebar"> kod för sidebar här </div>

<div id = "footer"> kod för footer här </div>

</div>

</body>
```

Figur 4.1 Webbsidans kodstruktur.

Efter två veckors utveckling ändrades kodstrukturen för webbsidan eftersom den gamla strukturen inte fungerade för AngularJS (se figur 4.2). Istället för ha en container-sektion och en sidbar-sektion sattes de ihop till en gemensam sektion. Med hjälp av en länk användes AngularJS till att byta mellan container och sidebar på första sidan.

```
<html ngApp>
<body>

<div id = "structure">

<div id = "header"></div>

<div id = "container" ngView></div>

<div id = "footer"></div>

</div>
</body>

</html>
```

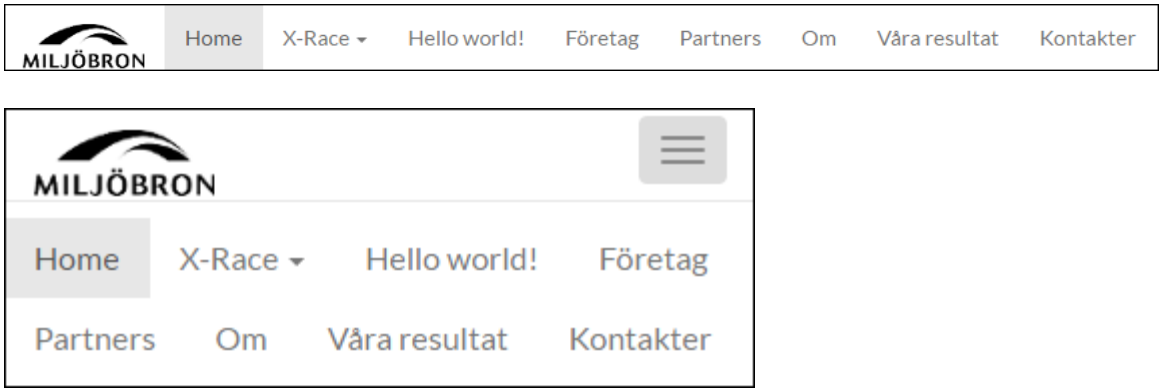
Figur 4.2 Nya kodstrukturen för AngularJS.

- Ramverk

Olika ramverk användes till att underlätta utvecklingen av webbsidan. Skärmanpassning var högt prioterat och behövde lösas tidigt. Med hjälp av *WC3.css* ramverket kunde detta lösas snabbt genom dess inbyggda responsiva grid-system som anpassar webbsidan direkt efter skärmens storlek. Bootstrap användes också till att mobilanpassa olika komponenter på webbsidan vilket diskuteras i designdelen. AngularJS kombinerades med JQuery till att skapa en snabb och effektiv navigeringsmeny men implementerades inte förrän Prototyp Service och kommer därför att diskuteras senare [37].

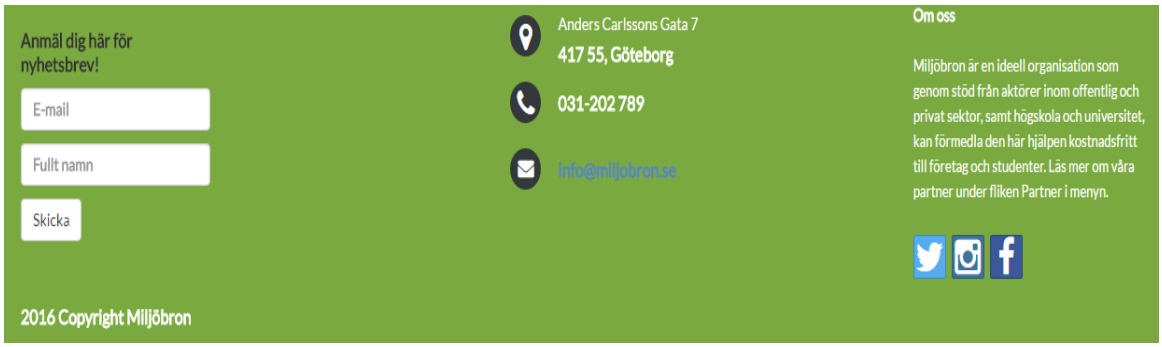
- Design

Webbsidans design följde Miljöbrons grafiska profil (se bilaga) och bestod även av Bootstrap-komponenter. Webbsidans bakgrundsfärg blev enligt den grafiska profilen limegrön och typsnittet blev Gill Sans MT. Header-sektionen var uppbyggt med *Wp-bootstrap-navwalker* [34] vilket var en färdig Bootstrap-navigeringsmeny anpassad för Wordpress och olika skärmstorlekar (se figur 4.3). I Wordpress tillkom även funktionen att menyflikarna kunde tas bort eller läggas till.



Figur 4.3 Navigeringsmenyn vid olika skärmstorlekar.

Webbsidans footer skapades med *Footer-distributed-with-address-and-phones* [35] vilket var en färdig mobilanpassad Bootstrap-footer (se figur 4.4). Denna Bootstrap-footer anpassade sig automatiskt efter olika skärmstorlekar vilket underlättade utvecklingen av en responsiv webbsida. På vänster sida av footern tillkom det även ett nyhetsbrev som bestod av färdiga Bootstrap-textrutor och en Bootstrap-sökknapp. I mitten tillkom det allmänna kontaktuppgifter om Miljöbron och en e-mail länk. På höger sida tillkom det allmän information om Miljöbron och ikoner för sociala medier. Ikonerna utvecklades med Bootstrap-social [36] vilket var färdigdesignade Bootstrap-knappar och beroende vilken man klickade på, öppnades antingen Facebook, Twitter eller Instagram i en ny tab.



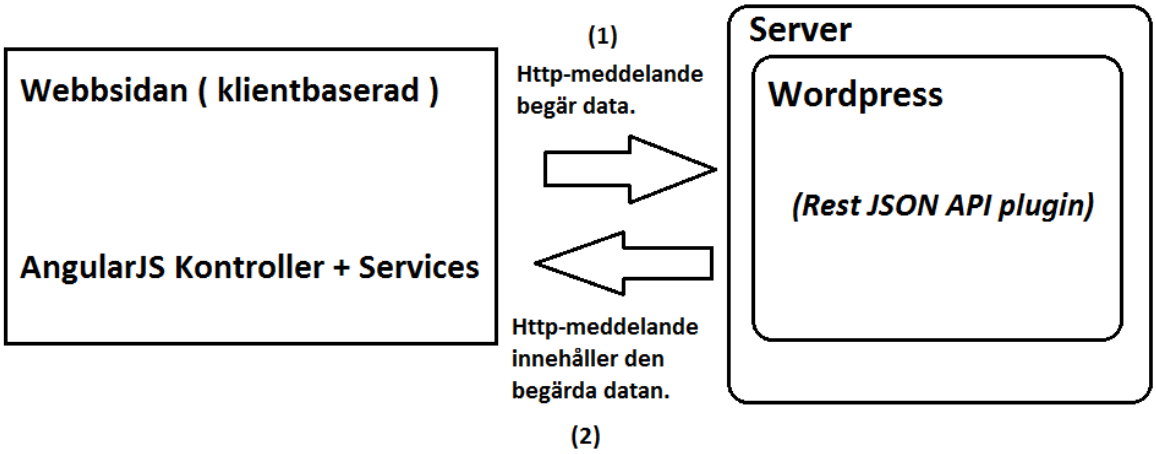
Figur 4.4 Footerns utseende vid olika skärmstorlekar.

4.3.4 Prototyp Service

I denna prototyp skapades all databaskommunikation och övriga funktioner.

- Databaskommunikation

Webbsidans databaskommunikation utvecklades med hjälp av AngularJS. Då Wordpress fungerar som backend tillsammans med en MySQL databas behövdes även ett Rest JSON API installeras för att göra AngularJS kompatibelt med Wordpress. Det skapades AngularJS kontroller som kombinerades med http-services för att kunna skicka och ta emot data ifrån databasen via http-meddelanden vilket gör webbsidan klientbaserad (se figur 4.5). Datan användes därefter på webbsidan till att byta sidornas innehåll.



Figur 4.5 AngularJS kontroller kommunicerar med servern genom att skicka och ta emot http-meddelanden via Services.

- Övriga funktioner

Det skapades ett AngularJS filter som användes till att filtrera alla länkar som ingick i respons-meddelanden från databasen. Alla länkar ändrades med filtret till att öppnas i en ny flik istället för att öppnas på den aktuella sidan vilket var smidigt när flera länkar skulle öppnas. Det skapades också ett sökfält med en knapp som var placerad överst i container sektionen på webbsidan (se figur 4.6).

Figur 4.6 Webbsidans sökfält där sökordet ”hello world” används som exempel.

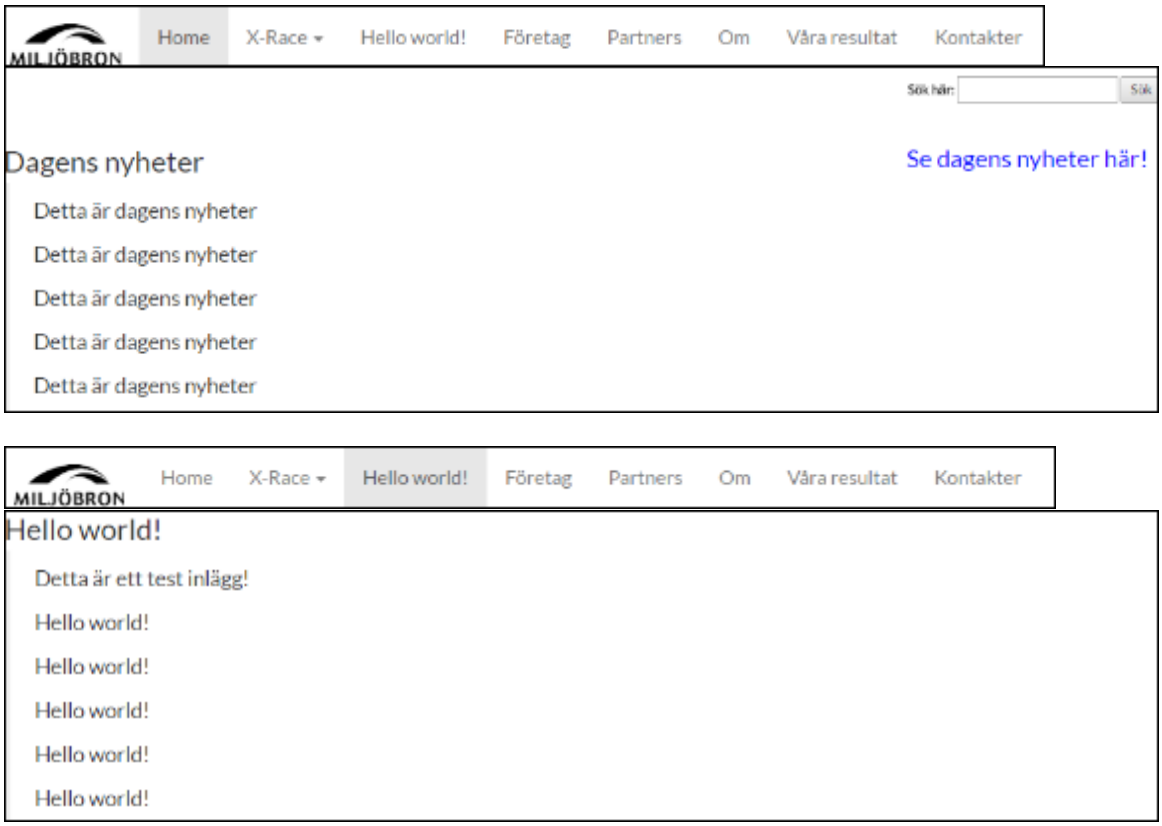
I headern skapades det ett bildspel som kallades för carousel och ingick i Bootstrap ramverket. Bildspelet justerades så att den ändrar bild automatiskt var femte sekund och det skapades även knappar så att bilderna kan ändras manuellt vid knapptryck. Det lades till fem stycken bilder i bildspelet som går endast att redigera via koden.
Det utvecklades även en JavaScripts funktion som gör att navigeringsmenyn försvinner efter att man bläddrar ner till ett visst läge vilket ger en bättre användarupplevelse av den information som erbjuds.

5. Resultat

Resultatet blev en responsiv webbsida som består av en structure-sektion, header-sektion, en container-sektion och en footer-sektion (se figur 5.1).

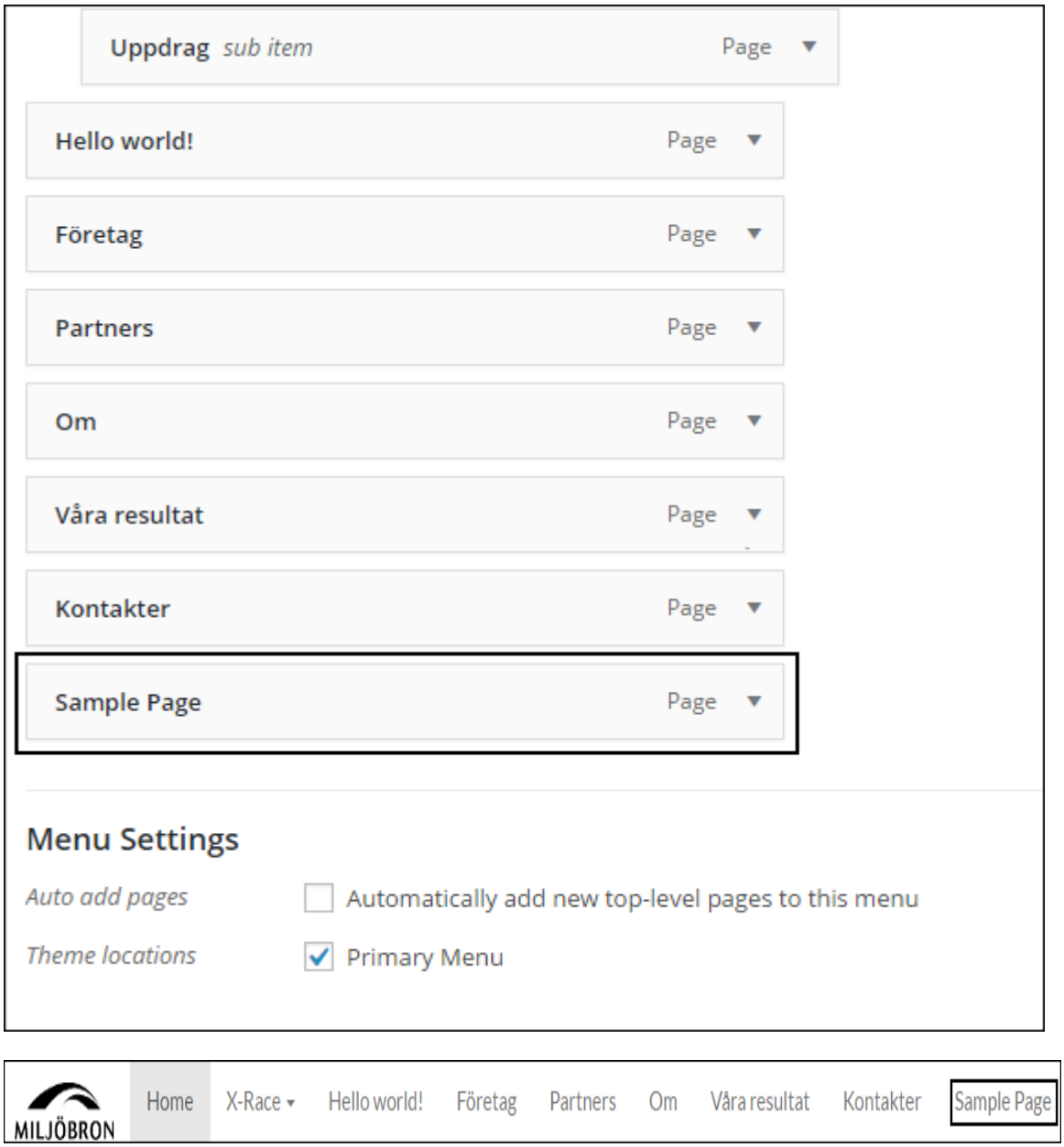
Figur 5.1 Webbsidans sektioner.

I headern finns det en navigeringsmeny och ett bildspel. Navigeringsmenyn gör det enkelt att byta sida på webbsidan (se figur 5.2).



Figur 5.2 Innan och efter man byter webbsidan med navigeringsmenyn.

Det går även att lägga till nya menyflikar direkt från Wordpress (se figur 5.3).



Figur 5.3 Menyfliken ”Sample Page” läggs till och visas därefter på navigeringsmenyn.

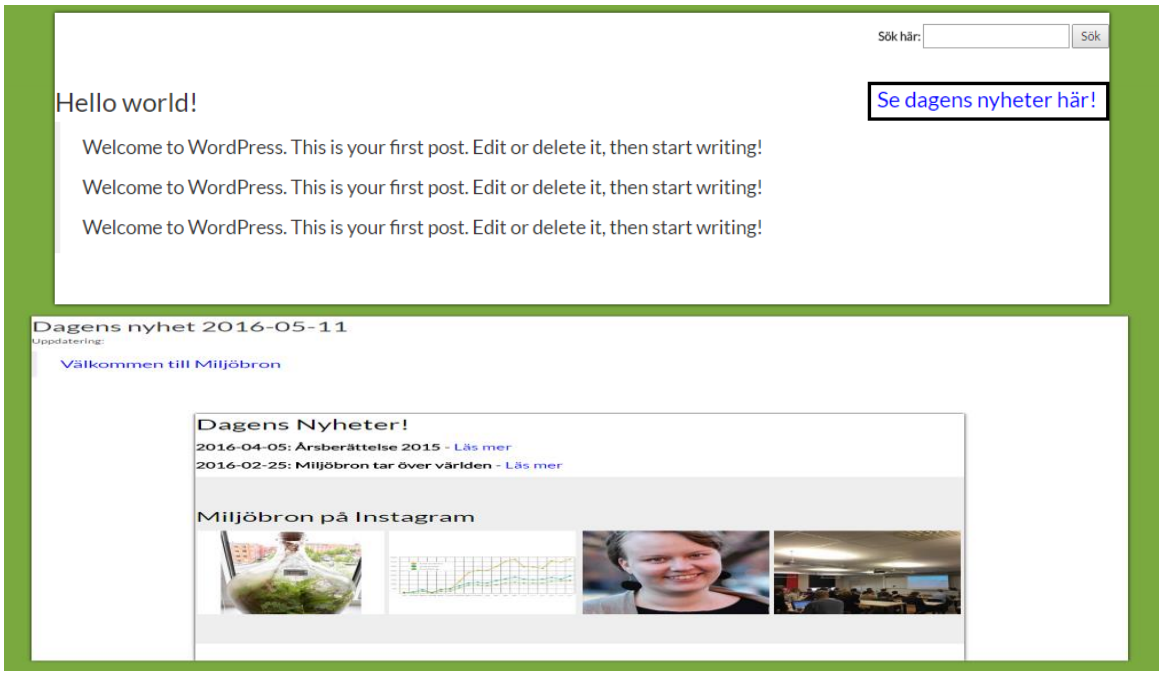
Bildspelet visar fem stycken bilder anknutna till Miljöbron som det enkelt går att ändra emellan (se figur 5.4).

Figur 5.4 Före och efter man ändrar bild på bildspelet.

Containern är placerad i mitten av sidan och visar webbsidans information. Det finns även ett sökfält och en länk för nyheter. I sökfältet kan man söka efter inlägg genom att skriva in inläggets titel (se figur 5.5).

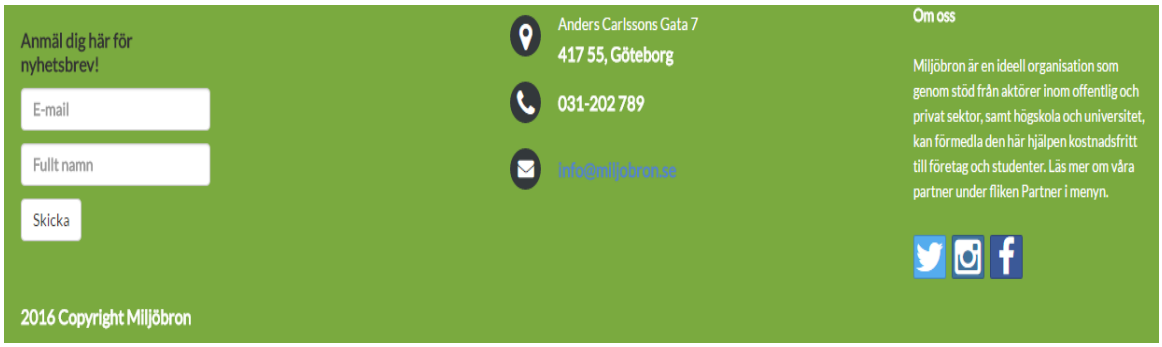
Figur 5.5 Webbsidans sökfält där sökordet ”hello world” används som exempel.

Om man klickar på länken öppnas dagens nyheter på första sidan (se figur 5.6).



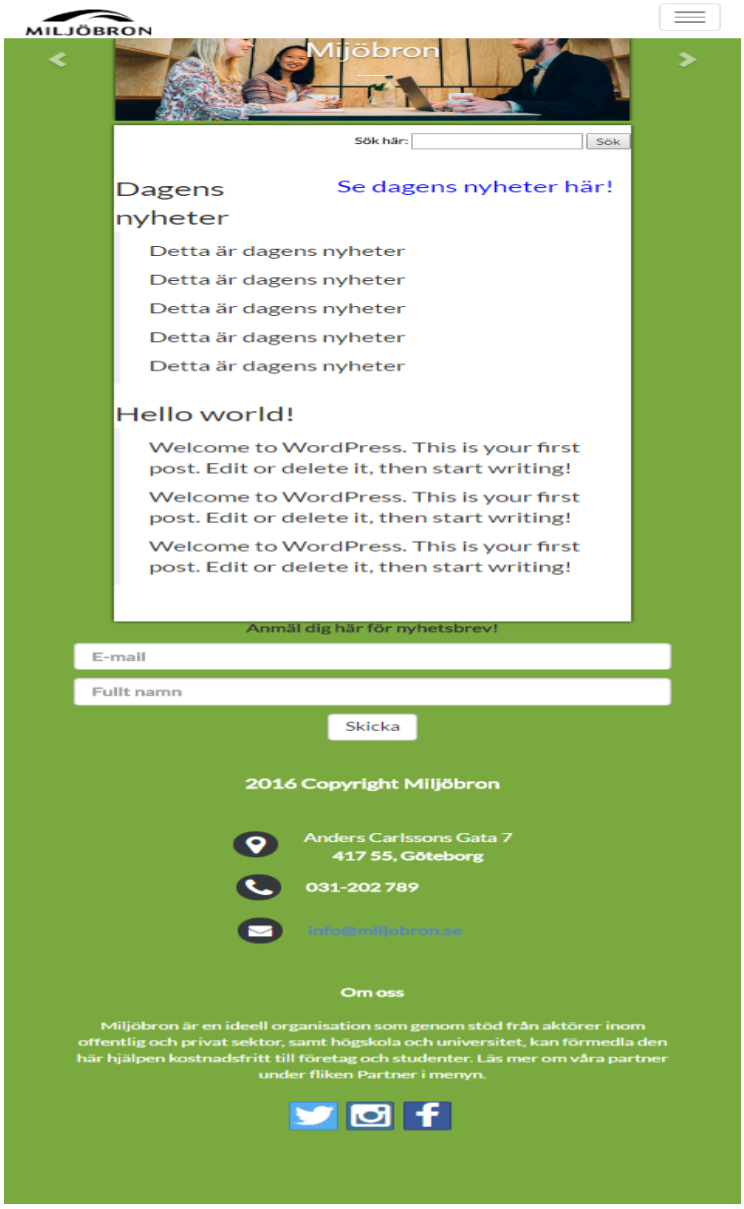
Figur 5.6 Länken öppnas dagens nyheter som ersätter första sidan.

Footern är placerad längst ner. Den består av ett nyhetsbrev, allmänna kontaktuppgifter och ikoner för sociala medier (se figur 5.7). Nyhetsbrevet fungerar inte i dagsläget men tanken är att besökare på webbsidan ska kunna skriva upp sig för Miljöbrons nyhetsbrev. Ikonerna öppnar Miljöbrons Facebook, Instagram och Twitter i en ny flik.



Figur 5.7 Footerns utseende.

Webbsidan är även responsiv vilket nämndes ovan (se figur 5.8).



Figur 5.8 Webbsidans utseende på mobilen.

6. Diskussion

6.1 CMS

Det finns både för- och nackdelar med Wordpress, Drupal och Joomla. Wordpress community är betydligt större än Joomla och Drupal vilket gör det lättare att få hjälp om problem uppstår. Ett större community innebär också att flera utvecklare väljer att skapa plugin:er till Wordpress istället för de andra CMS:er eftersom utvecklarna når ut till flera användare. När det kommer till enkelhet att skapa en webbsida utan programmeringskunskaper har Wordpress och Joomla övertaget. Det går enkelt skapa en webblogg direkt på nätet genom att välja ett färdigt tema, något som Drupal inte erbjuder, utan kräver programmeringskunskaper. I Drupal finns endast alternativet att användaren måste ladda ner Drupal som ett lokalt projekt och teckna ett webbhotell för att kunna få webbsidan på nätet. Wordpress och Joomla erbjuder också denna tjänst men riktas som en separat tjänst åt utvecklare. Fördelen med att ladda ner antingen Wordpress, Drupal eller Joomla som ett lokalt projekt är att användaren kan skapa ett unikt tema för sin webbsida. När det kommer till mängden resurser för att kunna köra CMS:en är det Joomla men främst Drupal som har övertaget. Drupal och Joomla laddar sina sidor snabbare än Wordpress vilket ger bättre prestanda men kan ändras vid installering av plugin:er (modules i Drupal). Alla har tillgång till Restfull API alternativ vilket är oerhört viktigt idag för att kunna sätta ihop separata komponenter som databas och klient. Programmuppdateringar sker automatiskt i Wordpress och Joomla vilket inte är fallet i Drupal som endast uppdaterar plugin:er automatiskt. Detta gör att Drupal kräver att en person med programmeringskunskaper går in i koden och uppdaterar webbsidan till den senaste Drupalversionen. Drupal blir dyrare att underhålla eftersom en person med programmeringskunskaper krävs för att underhålla sidan.

6.2 Miljöaspekt

Genom att skapa en ny webbsida åt Miljöbron, som eventuellt kan göra fler studenter och företag intresserade av samarbete med Miljöbron, kan flera projekt utföras och på så vis gynna miljön. Skulle Miljöbron få fler kunder och besökare på sin webbsida skulle även webbsidan vara verktyget för att förmedla all information om Miljöbron och deras projekt. Det är därför viktigt att webbsidan är läsbar och lättanvänd så informationen kan förmedlas på ett bra och effektivt sätt. Genom att sprida information om miljöprojekt har detta också en positiv miljöpåverkan eftersom fler personer blir intresserade.

AngularJS används till att skapa struktur på webbsidans kod men också till att minska belastningen av servern. Istället för att ladda hela webbsidan varje gång en besökare klickar på en ny sida, uppdateras endast innehållet i containern. Genom att konstruera webbsidan på detta sätt gynnas miljön eftersom servern inte behöver hämta lika mycket data och därmed sparas energi. När en utvecklare skapar en webbsida ska denna person inte bara få alla webbsidans funktioner att fungera utan måste även tänka på att koda miljömässigt. Om alla webbsidor skulle konstrueras enligt ett miljömässigt mönster skulle det både gynna miljön och företag eftersom färre datahämtningar ifrån servrar skulle behövas för att underhålla webbsidor.

7. Slutsats

Målet med projektet var att utveckla en ny webbsida åt Miljöbron som har ett modernt utseende, enkel att använda och är responsiv. Hur vi skulle gå till väga med detta var dock inte lika tydligt. Tidigt i projektet kom en del krav och önskemål som vi kunde utgå från, men hur vi skulle uppnå dessa var helt upp till oss. Vi valde att använda det moderna ramverket AngularJS i CMS:et WordPress för att uppnå det resultatet vi ville. Dessa verktyg gav oss bra förutsättningar och kunskap inför framtida webbprojekt, då verktygen är oerhört populära och kunskaper om dessa är eftertraktade.

7.1 Uppfyllda krav

Ett av de främsta krav som Miljöbron ställde på webbsidan var att den skulle fungera i alla typer av surfverktyg, det vill säga att den är responsiv. Detta krav uppnåddes och webbsidan anpassar sig väl och är enkelt navigerbar och läsbar på dator, mobil och surfplatta. Då deras nuvarande webbsida använder sig av WordPress för publicering och hantering av webbsidan så valde vi också att använda WordPress för att kostnaden för deras webbsida inte skulle förändras och för att Wordpress har ett enormt community vilket gör det enkelt att få hjälp om det skulle uppstå problem med webbsidan i framtiden. Att det är samma CMS som styr webbsidan uppfyller också kravet att sidan ska fungera med deras portalwebbsida, det vill säga där man navigerar till antingen Miljöbron Västra Götaland eller Miljöbron Skåne.

7.2 Ej uppfyllda mål

Ett krav som vi inte hunnit med och som är väldigt viktig för Miljöbrons verksamhet är möjligheten att översätta webbsidan till engelska. Detta för att internationella studenter skall kunna ta del av de projekt som Miljöbron erbjuder. Om alternativet inte finns kan problem uppstå mellan Miljöbron och deras samarbete med skolor.
Deras nuvarande webbsida använder även en anpassad version av WordPress administrativa verktyg, för att göra det så enkelt som möjligt för de anställda att redigera webbsidan. Vår webbsida har inget anpassat WordPress verktyg och skulle därmed vara svårare för Miljöbrons anställda att använda, vilket är något de helst vill undvika.

7.3 Fortsatt arbete

Webbutveckling utvecklas konstant vilket betyder att webbsidor också behöver underhållas för att hänga med i tekniken, och för att förbli moderna. Det räcker oftast inte att bygga en webbsida och sedan sluta utveckla den helt och hållet. Det finns därför många sätt man kan fortsätta arbeta med Miljöbrons webbsida. Ett sätt att fortsätta arbeta är t.ex. att försöka göra den administrativa delen av webbsidan mer lätthanterlig. Detta var ett krav som ej uppnåddes till den nivå som var planerat från början, och därför något som kan utvecklas vidare. Nyhetsbrevet är även en funktion som behöver implementeras så att besökare kan prenumerera på Miljöbrons nyhetsbrev. Det är för tidigt att säga om Miljöbron kommer att använda denna webbsida som sin officiella, men det går att säga att webbsidan uppfyller många av kraven och önskemålen som ställdes från början.

8. Referenser

1. CMS Website Design (Content Management System), Nevega Bem, http://www.navegabem.com/cms-website-design.html (Acc 2016-04-30)

2. About WordPress, WordPress, https://wordpress.org/about/ (Acc 2016-04-30)

3. Requirements, WordPress, https://wordpress.org/about/requirements/ (Acc 2016-04-30)

4. WordPress Features, WordPress, https://codex.wordpress.org/WordPress_Features (Acc 2016-05-16)

5. GNU General Public License, GNU, 2007, http://www.gnu.org/licenses/gpl-3.0.html (Acc 2016-05-16)

6. Drupal Shared Hosting, Drupal, https://www.drupal.org/hosting (Acc 2016-05-17)

7. About, Drupal, https://www.drupal.org/about (Acc 2016-05-17)

8. Start Your Free Site With Joomla.com, Joomla, https://www.joomla.com/ (Acc 2016-05-17)

9. About Joomla!, Joomla, https://www.joomla.org/about-joomla.html (Acc 2016-05-17)
10. What is Angular?, AngularJS, https://docs.angularjs.org/guide/introduction (Acc 2016-05-08)

11. ngApp, AngularJS, https://docs.angularjs.org/api/ng/directive/ngApp (Acc 2016-05-08)

12. ngView, AngularJS, https://docs.angularjs.org/api/ngRoute/directive/ngView (Acc 2016-05-14)

13. Bootstrap, AngularJS, https://docs.angularjs.org/guide/bootstrap (Acc 2016-05-10)

14. Dependency Injection, AngularJS, https://docs.angularjs.org/guide/di (Acc 2016-05-11)

15. $http, AngularJS, https://docs.angularjs.org/api/ng/service/$http (Acc 2016-05-11)

16. Understanding Controllers, AngularJS, https://docs.angularjs.org/guide/controller (Acc 2016-05-12)

17. Filters, AngularJS, https://docs.angularjs.org/guide/filter (Acc 2016-05-12)

18. Providers, AngularJS, https://docs.angularjs.org/guide/providers (Acc 2016-05-14)

19. HTML Introduction, w3school, http://www.w3schools.com/html/html_intro.asp (Acc 2016-05-10)

20. CSS Introduction, w3school, http://www.w3schools.com/css/css_intro.asp (Acc 2016-05-10)

21. What can you do with JavaScript, W3C, 2011, https://www.w3.org/community/webed/wiki/index.php?title=What_can_you_do_with_JavaScript&mobileaction=toggle_view_desktop (Acc 2016-05-10)

22. JavaScript Security Breaches, Javascripters, http://www.jscripters.com/javascript-security-breaches/ (Acc 2016-05-11)

23. Tryit Editor, w3school, http://www.w3schools.com/js/tryit.asp?filename=tryjs_validation_js (Acc 2016-05-11)

24. PHP, Wikipedia, 2016, https://en.wikipedia.org/wiki/PHP (Acc 2016-05-12)

25. Carey Wodehouse, Server-Side Scripting: Back-End Web Development Technology, Upwork, https://www.upwork.com/hiring/development/server-side-scripting-back-end-web-development-technology/ (Acc 2016-05-12)

26. Responsiv webbdesign, Wikipedia, 2015, https://sv.wikipedia.org/wiki/Responsiv_webbdesign (Acc 2016-05-12)

27. Bootstrap, https://getbootstrap.com/ (Acc 2016-05-13)

28. ICTs and Energy Efficiency, ITU, http://www.itu.int/en/action/climate/Pages/energy_efficiency.aspx (Acc 2016-05-15)

29. Creating a responsible, earth-friendly website, Manoverboard, 2015, https://manoverboard.com/creating-a-responsible-earth-friendly-website/ (Acc 2016-05-15)

30. Codex, WordPress, https://codex.wordpress.org/WordPress_Files (Acc 2016-05-04)

31. Basic Directory Structure of a Drupal 7 Project, Drupal, https://www.drupal.org/node/2621480 (Acc 2016-05-17)

32. Pedrina Brasil, Creating a Template for Joomla 2.5 PART 2, Grimmster 2012, http://www.grimmster.com/author/pedrina (Acc 2016-05-17)

33. Extension Types, Joomla, https://docs.joomla.org/Extension_types (Acc 2016-05-17)

34. Edward McIntyre ,wp-bootstrap-navwalker, Github, 2014, https://github.com/twittem/wp-bootstrap-navwalker (Acc 2016-05-23)

35. Freebie: 5 Responsive Footer Templates, Tutorialzine, http://demo.tutorialzine.com/2015/01/freebie-5-responsive-footer-templates/footer-distributed-with-address-and-phones.html (Acc 2016-05-23)

36. Social Buttons for Bootstrap, Github, https://lipis.github.io/bootstrap-social/ (Acc 2016-05-23)

37. W3.CSS Tutorial, w3school, http://www.w3schools.com/w3css/default.asp (Acc 2016-05-23)

38. Templates, AngularJS, https://docs.angularjs.org/guide/templates (Acc 2016-05-24)

39. jQuery Introduction, w3school, http://www.w3schools.com/jquery/jquery_intro.asp (Acc 2016-05-25

40. Richard Hein, 6 reasons you should be using jQuery, JavaWorld, 2012, http://www.javaworld.com/article/2078613/java-web-development/6-reasons-you-should-be-using-jquery.html (Acc 2016-05-25)

41. AngularJS Superheroic JavaScript MVW Framework, edureka!, 2015, http://www.edureka.co/blog/angularjs-superheroic-javascript-mvw-framework (Acc 2016-05-27)

42. Bootstrap (front-end framework), Wikipedia, https://en.wikipedia.org/wiki/Bootstrap_(front-end_framework)#JavaScript_components (Acc 2016-05-13)

Bilagor

Bilden visar hur vi delade upp arbetet i början av projektet såsom kodning, planering, skriva rapport, testning, förberedelse inför presentation och prototyper.

Gant schema