En webbapplikation för Pargas stad

Utveckling av funktionerna

Duncker Benjamin

Rantanen Cedric

Examensarbete för Tradenom (YH)-examen

Utbildningsprogrammet i Informationsbehandling

Raseborg 2015

EXAMENSARBETE

Författare: Duncker Benjamin & Rantanen Cedric

Utbildningsprogram och ort: Informationsbehandling, Raseborg

Handledare: Gammals Rolf

Titel: En webbapplikation för Pargas stad – Utveckling av funktionerna

---

Datum: 13.11.2015 Sidantal: 63 Bilagor: 3

Abstrakt

Detta examensarbete beskriver utvecklingen av en mobil webbapplikation till Pargas stad. Applikationens uppgift är att förbättra kommunikationen inom Pargas genom att innehålla väsentlig information för Pargasbor, turister och lokala företag. För att applikationen skall innehålla det material som anses vara väsentligt skall den fungera som ett informationsflöde där invånarna finner exempelvis företags öppethållningstider samt deras geografiska läge, arbetsplatsannonser, aktiviteter, nyheter samt tidtabeller till bussar, tåg och färjor. Det skall vara möjligt att uppdatera detta informationsflöde och ändringarna skall ske i realtid. Allt detta skall leda till bättre samhörighet mellan företag och invånare inom Pargas stad.

Examensarbetet kommer att fokusera på applikationens uppbyggnad och funktionalitet. Vi valde att arbeta med innehållshanteringssystemet Drupal 7 och med dess moduler för att uppfylla de krav som applikationen skall ha för att fungera. Med hjälp av moduler kan vi skapa avancerade funktioner till applikationen på ett mindre komplicerat sätt.

Slutprodukten av detta examensarbete är en användarvänlig webbapplikation vars uppgift är att förbättra kommunikationen inom Pargas stad. Applikationen omfattar alla de nödvändiga funktionerna för att möjliggöra detta. Dessa funktioner är: användarinloggning, språkbyte, skapandet av nytt innehåll och en företagsprofil.

Språk: Svenska Nyckelord: Pargas, Drupal, webbapplikation, PHP, moduler

---

BACHELOR'S THESIS

Author: Duncker Benjamin & Rantanen Cedric

Degree Programme: Business Information Technology, Raasepori

Supervisors: Gammals Rolf

Title: A web application for the town of Parainen – Development of the functions

---

Date: 13.11.2015 Number of pages: 63 Appendices: 3

Abstract

This thesis describes the development of a mobile application for the town of Parainen. The objective of the application is to improve communication by containing material deemed to be essential and because of this it will act as an information point where citizens can find e.g. company opening hours as well as their geographic location, job advertisements, activities, news and timetables for buses, trains and the ferries. It will be possible to update all of this information and the changes will take place in real time. This should result in a better connectedness between businesses, tourists and the citizens in the town of Parainen.

This thesis will focus on the structure and functionality of the application. We chose to work with the content management system Drupal 7 and with its modules to meet the requirements of the application. With the help of modules, we created advanced functions which the application must have with a method that is less complicated.

The final product is a user-friendly web application whose purpose is to improve the communication within the town of Parainen. The application consists of all the required functions to make this possible. These functions include: a login system, an option for multiple languages, a creation of new content and a company profile.

Language: Swedish Key words: Parainen, Drupal, web application, PHP, modules

---

Innehållsförteckning

1 INLEDNING

Nuförtiden använder vi oss av applikationer i allt större omfång. Applikationerna har varierande syften och funktioner som strävar till att hjälpa användaren med vardagliga arbeten. I detta examensarbete kommer vi att presentera en mobil applikation för Pargas stad. Examensarbetet beskriver hur vi har arbetat, vilka teoretiska delar som är centrala och hur vi utfört den praktiska delen.

1.1 SYFTE OCH MÅLSÄTTNING

Projektets bakgrund härstammar från ett större projekt vid namn ”Den levande skärgården i Pargas Stad” och har som uppgift att förbättra Pargasbornas levnadskvalité. Ändamålet är att skapa en mångsidig applikation vars användningsområde är Pargas stad. Denna applikation skall förbättra informationsflödet inom hela staden och genom detta gynna de lokala tjänsterna och öka samhörigheten mellan småföretagarna. Det är också planerat att slutprodukten kunde användas till andra orter och städer, genom att skapa applikationen i ett malliknande format.

Målet med detta examensarbete är att skapa grunderna till en användarvänlig webbapplikation för Pargas stad. Applikationen omfattar alla de nödvändiga funktionerna för att möjliggöra detta.

De slutliga effektmålen med denna applikation är att innehålla väsentlig information för Pargasborna och turister. För att applikationen skall innehålla det som räknas som väsentligt skall den fungera som ett informationsflöde där invånarna finner exempelvis företags öppethållningstider och deras geografiska läge, arbetsplatsannonser, aktiviteter, nyheter samt tidtabeller till bussar, tåg och färjor. Det skall vara möjligt att uppdatera detta informationsflöde och ändringarna skall ske i realtid. Allt detta skall leda till bättre samhörighet mellan företag inom Pargas stad. Applikationen kommer också att vara till stor nytta för turister som besöker staden eftersom den skall bland många andra funktioner innehålla en karta på var allting är beläget. Dessutom skall den ha information om när alla båtar och färjor kör mellan öarna.

1.2 BAKGRUND

Egentliga behovet för denna produkt anmärktes av våra beställare Elvström och Norrman som varit i kontakt med Pargas stad. Robin Elvström och Jonas Norrman är tradenomstuderande på Yrkeshögskolan Novia i Åbo och medlemmar i projektet Den levande skärgården i Pargas Stad. Elvström och Norrman tog kontakt med Novia Raseborg och utbildningsavdelningen Informationsbehandling och frågade efter personer på tredje årskursen som påbörjat planeringen av examensarbetet. Vi har anförtrotts uppdraget att skapa en applikation för Pargas stad.

Vi fick veta att Pargas var intresserad av en applikation av denna typ men önskar se resultat innan de är beredda att investera i vidareutveckling av produkten. Det finns liknande applikationer som exempelvis Foursquare som kan uppfylla en hel del av kriterierna som detta examensarbete har, men ingen applikation som är specifikt skräddarsydd för Pargas stad.

1.3 OMFATTNING

I detta examensarbete presenterar vi enbart de väsentligaste funktionerna hos applikationen. Vi behandlar applikationens mest centrala koder. Vi utgår också från att läsaren har baskunskaper om olika tekniska termer vilket innebär att vi inte kommer att beskriva vanligt förekommande termer. Mer avancerade och upprepade termer som används i detta examensarbete är beskrivna i Bilaga 1.

1.4 TILLVÄGAGÅNGSSÄTT

Vid projektets startpunkt hade vi planerat att indela arbetet i tre stycken huvudkategorier. Den ena kategorin var applikationens funktioner, den andra var databas, sekretess och säkerhet. Den tredje och sista kategorin var design och layout. Robert Lönnberg ansvarar för applikationens utseende, vilket skulle betyda att vi två skulle få varsin kategori att ansvara över. Men under projektets gång lade vi märke till att programmeringstillvägagångssättet vi använde oss av inte kräver dessa två skilda kategorierna. Detta resulterade till att kategorin databas, sekretess och säkerhet föll bort. Då bestämde vi oss för att samarbeta med gemensamma krafter för att nå slutresultatet. Detta betyder att vi gemensamt arbetade på applikationens funktioner.

Under projektets utvecklingsskede har vi arbetat på distans. För att detta skall fungera har vi använt oss av olika chattprogram för att vi två skall kunna kommunicera med varandra i realtid. Dessa chattprogram är bland annat Skype och Steam. Vi delade upp arbetet genom att ge varandra delområden vilka vi var ansvariga för att förverkliga på applikationen. Rantanen tog sig an uppdraget att förverkliga en fungerande karta med alla sina tillhörande funktioner och Duncker såg till att användarna har en egen profil med alla väsentliga verktyg för innehållsredigering. Under projektets gång meddelade vi varandra om hur allting framskrider, ifall det finns problem och idéer på hur eventuella problem kan lösas. Till en stor del arbetar vi på projektet samtidigt och under längre perioder för att projektet skulle framskrida enligt tidsschemat.

Vårt första möte med Elvström och Norrman var den 30 mars 2015, varefter vi har haft regelbundna möten med en till tre veckors mellanrum. Vanligtvis var det Elvström som föreslog ett datum och ett klockslag för nästa möte via en grupp-chatt på Facebook. Det egentliga mötet hålls via Skype programmet. Under dessa möten presenteras nya idéer för applikationen, vad alla medlemmar arbetat med sedan förra mötet och vad de skall fortsätta med. Även problem, förslag på lösningar och beslutsfattande diskuteras under dessa möten. Individuella möten med beställaren och/eller med andra teammedlemmar har också gjorts av varierande skäl. De första gemensamma mötena är dokumenterade i mp3 filformat. Detta har senare uteblivit då Elvström har presenterat behövligt material som vi diskuterat under möten på Google Drive. Dessa material är bland annat enstaka funktioner som skall finnas med, visualisering över hur applikationen kunde se ut och en förteckning på applikationen i ett översikt (sitemap) format.

1.5 PARGAS STAD

Pargas är en tvåspråkig stad med 15 000 invånare, där de flesta talar svenska. Staden är belägen i västra Finland, precis söder om Åbo. Pargas har en naturskön miljö som består av en omfattande mängd av öar, holmar och grönskande växtlighet. För att hitta fram mellan alla öppna öar finns det ett dussintals vidsträckta broar samt enstaka mindre varianter som binder samman de mest centrala öarna. Därtill finns det färjor och förbindelsefartyg som transportmedel, vilka åker med jämna intervall mellan specifika öar. För de personer som besöker Pargas med egen båt finns det tio välkomnande gästhamnar och mindre hamnar för att ta sig i land. (Pargas, (u.å.)).

Pargas skärgård erbjuder en mysig småstadsmiljö nära havet vilket lockar till sig sommarbesökare samt turister. Pargas strävar till att erbjuda tjänster, boende, arbete och transport oberoende av var i staden man befinner sig. Till industriellt arbete är nedbrytning av kalk, produktion av isoleringsmaterial, cement och spackel samt putsmedel de största i Pargas.

2 PROGRAMMERINGSMILJÖ OCH VERKTYG

Programmeringsmiljön vi använder oss av är innehållshanteringssystemet (CMS) Drupal 7 och för att få vår webbapplikationstyp till ett distributionsbart filformat krävs det att vi tar i bruk tjänsterna som PhoneGap erbjuder. Utöver detta behandlas även ett flertal andra punkter som har betydelse för applikationsprojektet.

2.1 DRUPAL

Drupal är ett gratis CMS som baserar sig på öppen källkod. Drupal är skapad av Dries Buytaert och fungerade år 2000 som ett personligt experiment. Experimentet gick ut på att fungera som en sorts anslagstavla, dit han och hans vänner kunde skriva vad de sysslar med för tillfället. Domännamnet de använde då var drop.org, men vid en punkt även dorp.org. Drupal blev ett offentligt projekt år 2001 med syftet att låta användare få testa och experimentera sig fram med denna plattform för att möjligtvis finna nya vägar för utveckling. Namnet Drupal, kommer från det engelska uttalet av det holländska ordet "druppel", vilket betyder "drop". (Drupal, (u.å.)a).

Drupal är nuförtiden i användning på miljontals olika webbplatser samt applikationer, och själva systemet används och upprätthålls med ständiga uppdateringar av en stor mängd människor. Innehållshanteringssystemet baserar sig på programmeringsspråket PHP och kan köras på olika operativsystem som Windows, Mac och Linux. Det enda som krävs är att datorn upprätthåller en webbserver och en databas. Webbservern och databasen är väsentliga delar, då webbservern används för distribuering av information till Internet och databasen för att lagra innehåll och olika slags konfigurationsmöjligheter. Drupal har hög flexibilitet eftersom dess stomme är byggd för användning av moduler. Användaren kan anpassa sin webbplats enligt egna behov och kan smidigt installera moduler med avancerade funktioner för att erbjuda besökarna till webbplatsen en angenäm upplevelse.

Det finns flera olika sorter av utgåvor över programmet Drupal, varav version 7.39 (oktober 2015) är den senaste. Med denna version kommer det automatiskt med olika slag av kärnfunktioner under namnet Drupal Core, vilket kan anses vara de väsentligaste funktionerna som alla CMS program erbjuder. Dessa funktioner kommer bland annat i formen av moduler allt från hantering av block, skapandet av kontaktformulär, registreringsmöjligheter, bild

redigering och taxonomi. Därtill kommer det ytterligare funktioner som exempelvis rollhantering, byte och hantering av teman, administrering av systemet och menyhantering. Redan tack vare dessa kärnfunktioner kan användaren skapa förslagsvis en enkel personlig webbplats, ett forum eller en blogg anpassad för flera användare.

2.1.1 Installation av Drupal

Installationen av innehållshanteringssystemet Drupal på en dator är relativt rakt på sak. De första skeden av installationsprocessen är dock lite varierande beroende på vilket operativsystem man kommer att installera Drupal på. I vårt fall installerar vi Drupal på operativsystemet Linux och eftersom det är av servertypen saknar den ett GUI som innebär att vi är tvungna att använda oss av kommandon.

Man börjar med att ladda ner Drupal paketet, packa upp alla filer och sedan flytta innehållet till www-mappen. Detta kan ses från nedanstående Kod 1, kommandona 1 – 3. Därefter används kommandona 4 – 6, som består av att kopiera default.settings.php filen, ändra namnet på den till settings.php och ge tillräckligt med skrivrättigheter till filen. Tillräckligt med skrivrättigheter syftar i detta fall på att ge fulla rättigheter åt alla personer som har inloggningskonto på servern. När väl detta är gjort bör man radera alla rättigheter hos filen default.settings.php den får inte ändras på överhuvudtaget. Orsaken till att default.settings.php inte får ändras är för att det skall vara möjligt att göra en fullständigt rollback till den ifall något oplanerat händer. Den används ytterligare också då man flyttar Drupal-installationen till en ny databas. Settings.php ansvarar bland annat för att skapa en länk mellan koden och databasen, vilket i praktiken betyder att den innehåller databas namn, användare och lösenord. Utöver detta innehåller den också information som bland annat webbsidans krypteringssalt och andra användbara inställningar. Exempel på inställningar man enbart kan redigera via settings.php är varaktigheten på cookies och namnet på rollen anonymous. När Drupal hemsidan är färdig installerad skall man ta bort alla skrivrättigheter på filen och göra den till en skrivskyddad (read only) fil (McCourt, 2013). (Batigolix, Boggs, Calebtr och HongPong, 2014).

Efter detta följer steget till att skapa en Drupal databas. Vi skapar en tom databas och en användare med administratörs rättigheter till denna databas. Dock ändrar man namnet på ”username” och ”databasename” till det namn som man vill att användaren respektive databasen skall ha. Direkt efter att man skapat en användare kommer systemet att kräva ett

lösenord till användaren. Allt detta görs i Kod 1, kommandona 7 – 9. (Batigolix, Boggs, Calebtr och HongPong, 2014).

Nästa steg kommer att handla om att köra installationsskriptet. Man går med en valfri webbläsare till sin webbplats URL, vilket i vårt fall är http://91.150.11.4/. Det kommer att initieras en installationsprocess som behöver enstaka information innan Drupal är färdig för användning. Men om installationsprocessen inte initieras automatiskt skall man skriva /install.php efter domännamnet i URL:en. Information som skall ges är hurdan installationstyp man vill använda sig av, språket för webbplatsen och information över den databas som man nyligen skapade. Därtill skall man även ge information om webbsidans namn, e-post för notifieringar och tidszon, dessa kan redigeras vid ett senare skede från webbsidans konfigurationsinställningar. Då alla nämnda steg är avklarade borde installationsprocessen vara färdig och man kan börja använda sin nya webbplats som är uppbyggd på Drupal.

Kod 1. Kommando tabell för installation av Drupal

```
1. wget https://www.drupal.org/files/projects/drupal-7.36.tar.gz
2. tar -zxvf drupal-7.36.tar.gz
3. mv drupal-x.x /var/www/
4. cp sites/default/default.settings.php sites/default/settings.php
5. chmod a+w sites/default/settings.php
6. chmod a+w sites/default
7. mysqladmin -u username -p create databasename
8. mysql -u username –p
9. GRANT SELECT, INSERT, UPDATE, DELETE, CREATE, DROP, INDEX, ALTER,
CREATE TEMPORARY TABLES, LOCK TABLES
  ON databasename.*
  TO 'username'@'localhost' IDENTIFIED BY 'password';
```

2.1.2 Installation av moduler

Augusti 2015 finns det över 31´000 moduler att välja mellan. Dessa moduler har alla i en viss omfattning varierande egenskaper och tillger användaren olika slag av funktioner som kan implementeras till webbsidan. Det kan vara bland annat att ändra beteendet på webbsidan, utvidga kärnfunktionerna och/eller ändra utseendet. Det finns i princip två olika sätt för att installera moduler till Drupal. Detta görs antingen manuellt eller via Drupals gränssnitt.

När man väl hittat en modul från Drupals hemsida är det rekommenderat vid manuell installation av moduler att ladda ner den nyaste versionen från rekommenderade utgåvor kategorin samt en version som stöds på sin nuvarande Drupal utgåva. Man packar upp alla

filer och ansluter sig till webbplatsen med en FTP-klient. Därefter söker man sig till mappen vid namn modules, som befinner sig i sites/all/modules. Det är till denna mapp man överför sin uppackade modul. Efter detta är installationen i princip klar. Man kan därefter gå till sin webbplats för kategorin moduler http://example.com/admin/modules och aktivera den nyligen installerade modulen, dock ändrar man URL:en från example.com till det domännamn man använder.

Man installerar moduler via Drupal gränssnittet på ett lite annorlunda sätt än det manuella, då man kan slopa uppackningen och anslutningen av FTP-klienter. Man börjar med att gå till http://example.com/admin/modules och aktiverar en modul vid namn Update Manager. Därefter kommer det att finnas på samma sida en länk för att installera moduler. När man väl klickat sig in blir man ombedd till att antingen ladda upp en nedladdad zar.zip modulfiltyp eller skriva in länken för var denna zar.zip filtypen finns från Drupals egna modulsidor. Man får länken genom att högerklicka på tar.zip filtypen och välja ”Kopiera länkadress”. Efter att man gjort någondera av dessa alternativ kommer en installationsprocess att påbörjas. Efter installationen är det enbart att aktivera modulen, som sker på samma sätt som vid den manuella metoden. (Falk, 2011, s. 237).

2.1.3 Säkerhet

Förutom att Drupal är flexibel med tanke på det enorma modulutbudet, är den även bevisad till att vara ett säkert CMS som står upp mot de mest kritiska internetsårbarheterna. Tack vare att Drupal använder höga kodningsstandarder och noggranna processer för kodgranskning är programmet utformat med stabilitet, robusthet och säkerhet i åtanke.

Lösenord som sparas till databasen via Drupal är krypterade genom salt och upprepande gånger hashade. Salt är slumpvistillagt data, som exempelvis text, siffror och/eller symboler, som läggs till med lösenordet. Hash är en typ av matematisk algoritm som strävar till att få lösenordet till ett förhållandevis litet heltal, för att det skall bli relativt enkelt att spara lösenordet till databasen. Förutom dessa krypteringsmetoder erbjuder även Drupal diverse principer gällande säkerheten för lösenord genom att erbjuda krav för minimilängder, komplexitet och när lösenorden skall upphöras. För att ännu höja på webbplatsens säkerhet har Drupal till sitt förfogande en mängd diverse funktioner med konfigurationsmöjligheter, som bland annat administration över rättigheter, rollhantering och IP-blockering. (Drupal, (u.å.)b).

2.1.4 Drupal begrepp

Då man startar ett nytt Drupal projekt medföljer det enstaka nya begrepp som man bör känna till för att komma igång. En enda webbplats kan innehålla många olika typer av innehåll som exempelvis nyheter och aktiviteter. Dessa kan kategoriseras var för sig som egna content types eller på svenska innehållstyper. Då man skapar ett nytt aktivitetsinnehåll är det i Drupal termer frågan om en nod. Noden kommer i detta fall att tillhöra innehållstypen aktiviteter. Varje innehållstyp har egna inställningar över hurdana fält (fields) de innehåller och hur de skall presenteras (Falk, 2011, s. 63). Till fältkategorin kan det finnas många olika slag av fälttyper (field types). Dessa fälttyper definierar hurdan typ av data som skall lagras och med hurdan widget som elementen skall redigeras med. Widget är en datateknisk term över ett litet program eller formulärelement som innehåller omväxlande innehåll, och som vårt fall ansvarar för hur fälttypen skall uppvisas för användaren som skapar ett nytt innehåll. Det finns enstaka fälttyper som kommer med Drupal installationen, men med externa moduler har man till förfogande ett betydligt större utbud och därigenom mera varierande funktioner. Konfigurering över hur alla noder slutligen skall presenteras sker från innehållstypens hantera visnings-fliken (manage display). Vid hantera visnings-fliken har man möjlighet till att administrera hur nodernas fälttyper skall presenteras, i vilken rangordning de uppenbarar sig i noden och ifall man vill att någon skall vara dold.

För att ännu ha möjligheten till att samla ihop olika slag av innehållstyper och specifika noder använder vi oss av Views. Views är en separat modul vars huvudfunktion vid hemsidor som är programmerade med Drupal är att skapa innehållsrika listor, tabeller, bildspel och diverse andra format. Under administrationsmenyn för struktur finner man Views där det finns en översikt över alla tillgängliga vyer som finns på webbsidan. Utseendet och hur listorna fungerar går att redigera radikalt. En View kan exempelvis innehålla en tabell av fälttyper från flera olika innehållstyper som går att filtrera i bokstavsordning. (Falk, 2011, s. 105–106).

Beroende på hurdant Drupal tema som används har man ett flertal standardregioner utplacerade på sin webbplats. Dessa regioner definierar områden på vart innehåll kan läggas. Vanliga områden är ett mittenfält, sidobarer och en eller flera sidfötter. Ifall man önskar fler regioner än de som kommer med ett tema, är det möjligt editera i PHP-filerna page.tpl.php och [tema_namn]_theme.info för att antingen lägga till, ta bort eller justera på var regionerna borde befinnas. Till dessa regioner finns det möjlighet till att placera block. Block liknar till en del widget, men är trots det totalt två separata saker. Block fungerar som en låda, som visar

fram innehåll på en valfri tillgänglig region. Oftast är block skapade automatiskt tillsammans med olika moduler, men man kan trots det skapa egna block för varierande syften.

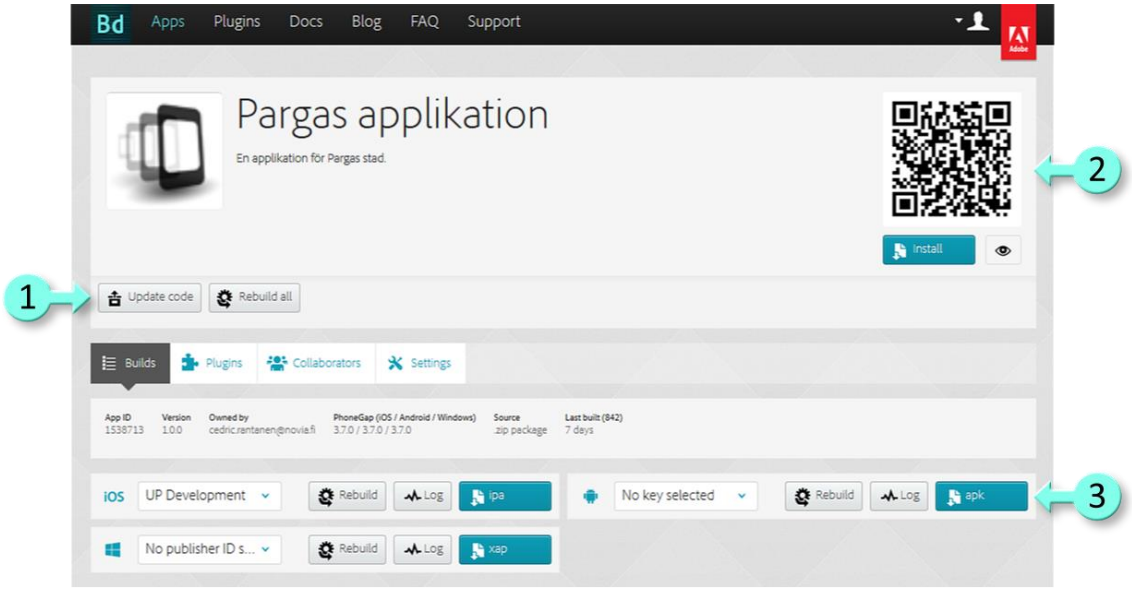
2.2 PHONEGAP

PhoneGap är ett ramverk (framework) som ägnar sig åt att skapa mobila applikationer. Programmet använder programmeringsspråken HTML, CSS och JavaScript, vilket är de tre populäraste programmeringsspråken på webben. PhoneGap kompilerar allt innehåll och som slutresultat får användaren färdigt användbara mobila applikationer för distribution i rätta filformat för enheter som iOS, Android och Windows. Filformaten kommer då att ha filändelserna .ipa, .apk respektive .xap. (PhoneGap, 2015).

Ifall man har planer på att lansera en applikation till de tre ovannämnda marknaderna, då behöver man i vanligt fall skapa tre skilda applikationer. Dessa tre applikationer har exakt samma syfte och innehåll, men använder tre skilda programmeringsspråk som Objective-C, Java och C++. Detta kommer att vara tidskrävande och kräver mycket förkunskap av programmeraren för att förverkliga applikationerna. Då man handskas med applikationsförverkligande med hjälp av PhoneGap behöver man enbart skapa en enda fungerande applikation, vilket i sin tur kommer att fungera på alla tre marknader. PhoneGap erbjuder en stor variation av diverse insticksmoduler som man kan använda sig av. Dessa förenklar programmeringsarbetet då man får tag på mobila enhetens API, vilket är exempelvis kameran, användarens kontaktlista eller lagringsutrymmet på enheten.

Man kan använda PhoneGap på två olika sätt, antingen laddar man ner programvaran och installerar det på en dator eller använda deras molntjänst PhoneGap Build. Till detta applikationsprojekt använder vi främst oss av molntjänsten. Från denna tjänst är det enkelt att ladda upp nya versioner av applikationen samt testköra dem på riktiga mobila enheter. Nedanstående Figur 1 visar användargränssnittet över PhoneGap Build. Användaren komprimerar applikationsmappen till ett ZIP filformat och laddar upp filen genom att klicka på Update code-knappen, som siffran 1 indikerar. Filen kommer att kompileras på PhoneGaps servrar. När kompileringen är avslutad och skett utan felanmärkningar, då kommer färdiga applikationstyperna överensstämma med siffran 3 från bilden. I detta skede kan man antingen ladda ned filformaten genom att klicka på någondera av filtyperna från siffran 3 eller skanna

QR-koden med en mobil enhet och installera applikationen direkt på enheten, som siffran 2 syftar på.



Figur 1. PhoneGap Build användargränssnitt.

Man installerar insticksmoduler till PhoneGap Build genom att man refererar deras namn i en konfigurationsfil vid namn config.xml. PhoneGap Build kommer vid kompileringsskedet granska denna konfigurationsfil och importera de nämnda insticksmoduler till den färdiga applikationen. Denna konfigurationsfil kan även innehålla information över applikationens namn, beskrivning, versionsnummer, rättigheter och andra nyttiga attribut. Ett exempel på hur man installerar en insticksmodul för att granska användarens nätverksanslutning kan ses från Kod 2. Versionsnumret kan dock utelämnas, då kommer PhoneGap Build att leta efter den nyaste versionen och installera den. Kod 2 visar enbart hur man installerar en specifik insticksmodul och visar inte hur, när och vilka kriterier som skall uppfyllas för att initiera funktionen på applikationen.

Kod 2. config.xml, installation av en insticksmodul för PhoneGap Build

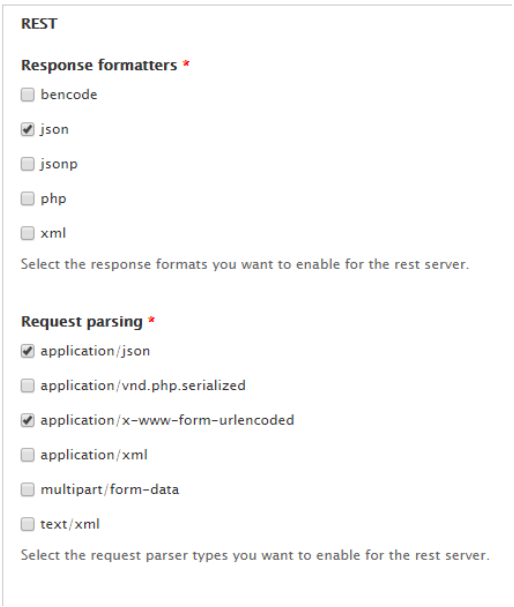
```
<gap:plugin name="org.apache.cordova.network-information"
version="0.2.12" />
```

2.3 DRUPALGAP

DrupalGap är ett utvecklingsverktyg för Drupal hemsidor. Det används för att skapa mobila applikationer som använder Drupal som bas. Eftersom verktyget använder öppen källkod är det också gratis att ladda ner. DrupalGap är vid nuläget 2015 enbart användbart på de mobila operativsystemen iOS och Android.

2.3.1 Skapa koppling mellan Drupal och DrupalGap

Då man arbetar med DrupalGap krävs det två stycken moduler på Drupal-sidan. De här två modulerna är Services och DrupalGap. Med hjälp av Service modulen går det att skapa en REST-server. Första steget då man startar med denna sammankoppling är att lägga ’Path to endpoint’ till ett logiskt namn som man enkelt förstår vad servicen tillhör och dess eventuella inställningar. I vårt fall heter den drupalgap. När detta väl är gjort bör man ge CRUD-rättigheter åt mobila applikationer, utan dessa rättigheter går det inte att göra något från den mobila enheten. Rättigheterna som CRUD står för är: create, retrieve, update och delete. När väl detta är utfört, går man över till det sista steget vilket är att konfigurera enstaka Service inställningar. De inställningar vi lagt upp för Service kan ses från nedanstående Figur 2.



Figur 2. DrupalGap Service inställningar.

Med hjälp av DrupalGap modulen kan man kontrollera att allting fungerar som det skall och med en enda knapptryckning skapa en DrupalGap SDK. Men om man vill installera det manuellt går detta genom att ladda ner filen från DrupalGaps hemsida. Om installationen sker

den manuella vägen krävs det dock extra konfigurationer inne i settings.js filen. Kod 3 visualiserar dessa ändringar.

Kod 3. settings.js, Till vilken webbplats applikationen är kopplad till, dess endpoint och typ av applikation.

```
Drupal.settings.site_path = 'http://91.150.11.4';
Drupal.settings.endpoint = 'drupalgap';
drupalgap.settings.mode = 'phonegap';
```

Som Kod 3 ovan visualiserar hade vi vår DrupalGap installation omställd till PhoneGap, vilket innebär att vi hade tillgång till alla de insticksmoduler som de erbjuder. Detta kräver förstås också att vi använder config.xml filen som medföljer PhoneGap. Kod 3 visar också att det är viktigt att veta namnet på endpoint, det namnet som vi namngav i ett tidigare skede.

2.3.2 DrupalGap moduler

Till skillnad från Drupal är alla funktioner hos DrupalGap moduler av någon typ. DrupalGap modulerna går att avskiljas genom att dela in dem till två skilda kategorier, contributed modules och custom. De moduler som tillhör contributed modules kategorin är i princip omgjorda Drupal moduler med syftet att fungera tillsammans med DrupalGap. De resterande modulerna faller till kategorin custom, vilket även funktioner för att exempelvis visa fram innehåll faller till.

Vanligtvis består custom moduler i DrupalGap av enbart en enda JavaScript fil. På hemsidan för DrupalGap finns det instruktioner över tillvägagångssätt på hur man skall skapa egna moduler. Dessa instruktioner ger enbart anvisningar för de viktigaste stegen och lämnar ute några viktiga små detaljer. Det som vi beskriver gäller enbart då en koppling mellan en View till DrupalGap skapas, de viktiga små detaljerna som lämnades bort i guiden är också beskrivna. Guiden kan man finna från länken http://drupalgap.org/node/219.

Första steget är att ställa in Views i Drupal till att skriva ut resultatet i JSON format, då DrupalGap enbart kan läsa dessa format. Nästa steg är att skapa en katalog och inuti den en JavaScript fil av exakt samma namn. Dessa skall vara belägna inne i katalogen app/modules/custom i DrupalGaps filstruktur. I exemplet som kan kopieras från DrupalGaps hemsida heter denna katalog my_module, men i vårt fall namnger vi katalogen till articleviewmenu. Denna katalog med namnet articleviewmenu kommer att bli vår custom modul. Då vi ändrat namnet på DrupalGaps exempel, kommer det att innebära att vid varje kodsträng där my_module definieras skall det ersättas i vårt fall med articleviewmenu. Därtill

vid path skall man lägga in sökvägen för den Drupal View som skall presenteras. (Tyler.frankenstein, 2014).

Då en modul installeras på DrupalGap måste man komma ihåg att contributed och custom moduler installeras på lite olika sätt. Kod 4 visar hur moduler av båda typerna installeras. Om det ytterligare krävs att man länkar modulens innehåll till en menyknapp skall man göra enligt Kod 5. Ifall det är frågan om en contributed modul krävs det utanpå detta även att installera samma modul på Drupal webbsidan för att det skall fungera.

Kod 4. settings.js, installation av modul

```
Drupal.modules.contrib['logintoboggan'] = {};
Drupal.modules.custom['my_module'] = {};
```

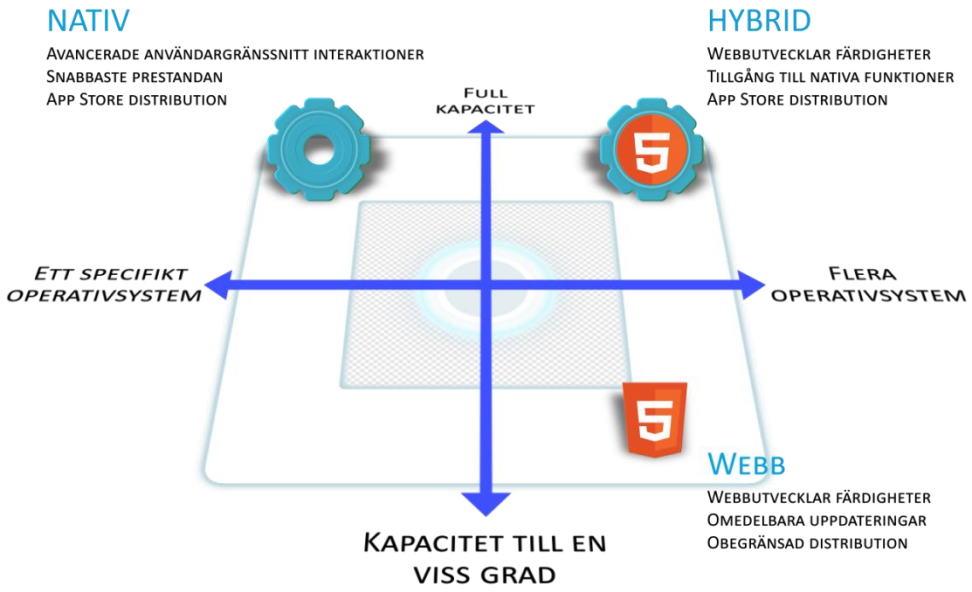
Kod 5. settings.js, Införing av modulfunktion till meny

```
drupalgap.settings.menus['main_menu'] = {
  options: menu_popup_get_default_options(),
  links: [
{
      title: 'Aktuellt',
      path: 'articles', /* The items['articles'] from
artricleviewmenu.js */
       options:{
        attributes:{
          'data-icon':'grid'
        }
      }
    }
  ]
};
```

2.4 APPLIKATIONSTYPER

Smarttelefoner har börjat bli allt vanligare i dagsläget, det är snart en norm för personer att äga en smarttelefon. Som en följd av att populariteten bland smarttelefoner har ökat mångfaldigt har även applikationssidans marknad ökat markant under de senaste åren. Då man skall skapa en applikation måste man tänka på vilka funktioner den skall ha och på vilket sätt man effektivast åstadkommer dem. De vanligaste applikationstyperna är nativ och webbapplikationer. En applikationstyp som ökat i popularitet på grund av PhoneGap är hybridapplikationer. En illustrering över de essentiella skillnaderna mellan dessa tre skilda applikationstyperna kan ses från nedanstående Figur 3. Figuren förklarar var nativ-, hybrid-

och webbapplikationer befinner sig i jämförelse med ifall de stöder flera operativsystem och hur mycket kapacitet de erbjuder till användaren.



Figur 3. En illustrering över de tre applikationstyperna (Korf och Oksman, 2015, omarbetad av skribenterna).

2.4.1 Nativapplikation

En nativapplikation är programmerad och optimerad för ett visst operativsystem, vilket betyder att applikationen kan ta full nytta av telefonens egna funktioner. Till dessa funktioner hör funktioner som bland annat kamera, GPS och användarens kontaktlista. Man kan säga att de här applikationerna har tillgång till allting på telefonen eftersom de är installerade och fungerar direkt på telefonen.

Ett par exempel på specifika operativsystem med sina egna programmeringsspråk och program. Operativsystemet iOS använder programmeringsspråket Objective-C vilket programmeras i programmet Xcode. Medan Android operativsystemet använder programmeringsspråket Java i programmet Eclipse eller Android Studio. (Korf och Oksman, 2015).

Som redan nämndes tidigare är fördelen med att programmera nativa applikationer att man får tillgång till alla funktioner som finns på en mobil enhet. Egenskaper för dessa applikationer är att de är snabba vid responstider för klickningar och har betydligt mjukare animeringar, vilket är viktigt ifall man skapar en applikation avsedd som ett spel. Därtill startar applikationen

nästan omedelbart, de har ett konsekvent utseende och presterar ytters bra (Korf och Oksman, 2015).

Nackdelen är dock att den enbart fungera på det operativsystem man programmerat den för. Ifall man vill överföra (to port) den till ett annat system krävs det att man programmerar om stora delar av applikationen, allt eftersom det är andra programmeringsspråk som är aktuella på de andra systemen (Budiu, 2013*)*. Utanpå detta kommer det även att orsaka en hel del besvär för programmerarna ifall man måste uppdatera applikationen. Om man har applikationen på tre olika marknader krävs det att uppdateringarna skickas skilt till alla tre.

2.4.2 Webbapplikation

Webbapplikationer är inte äkta mobila applikationer utan är i själva verket vanliga webbsidor som har gjorts om för att anpassas till mobila enheter med mindre skärm storlekar (Korf och Oksman, 2015). Programmeringsspråken är inte baserade sig på operativsystemet, vilket innebär att de här applikationerna inte använder Java, C eller något av den typen. De programmeringsspråk som används är webbspråken HTML5, CSS och JavaScript. Med hjälp av programmeringsspråken har man förmågan att göra applikationen mycket lik en nativ-applikation, men som är i själva verket bara en webbsida som användaren bläddrar igenom på webbplatsen medan man använder den.

Fördelen med den här applikationstypen är att det går mycket lätt att överföra den till andra operativsystem. Man kan kalla dessa applikationer för plattformsoberoende mobila applikationer. Dessutom om man har en enkel webbsida kan man under en kort stund från en nollpunkt få den att fungera bra för mobila enheter med mindre skärmar. Då ett företag egentligen bara är ute efter ett lätt sätt för mobilanvändare att surfa på webbsidan är webb-applikation det bästa alternativen.

Det finns dock tre stora nackdelar med webbaserade applikationer. Den första nackdelen är att det krävs alltid en internetanslutning för att använda applikationen, då applikationen egentligen är ett fönster in till en webbsida. Den andra är att det är omöjligt att få tillgång till de flesta funktioner som finns på en mobil enhet, det går bland annat inte att använda andra applikationer som är belägna på telefonen. Slutligen den tredje, applikationen kan bli avvisad från applikationsbutiker ifall de är programmerade för att vara webbapplikationer. Vilket

gäller bland annat för operativsystemet iOS som har strikta regler för vad som får finnas på deras applikationsbutik.

2.4.3 Hybridapplikation

De applikationstyper som är klassificerade som hybrid är varken genuint nativa eller webb baserade, utan de är mittemellan. PhoneGap applikationer hör till denna gren för hybridapplikationstyper. Eftersom all rendering av dess layout sker som om det vore en webbsida och man använder programmeringsspråken HTML, CSS och JavaScript, istället för att använda det egna programmets användargränssnitt och programmeringsspråk. Trots detta är dessa applikationer inte bara webbsidor, utan PhoneGap innehållet kompileras till applikationer avsedda för distribution till en applikationsbutik och har även tillgång till den mobila enhetens API.

Till en stor del innehåller hybridapplikationer det bästa från webb- och nativapplikations-typerna. Som nämndes tidigare behöver man bara skapa en applikation som sedan kan fungera på flera olika marknader och operativsystem. Tack vare detta sparas en hel del tid då omprogrammering till ett nytt operativsystem inte krävs. En extra fil behövs dock för operativsystemet Android. Filen går under namnet AndroidManifest.xml. Orsaken till att AndroidManifest.xml krävs är för att den innehåller alla de rättigheter applikationen skall ha tillgång till. De vanligaste är internet och kamera rättigheterna.

Beroende på avsikten över hurdan applikation som skapas kan hybridapplikationer ha allt innehåll sparat på den installerade applikationen, på servern eller delvis på båda. Ifall man bestämmer sig för att lagra allt innehåll på applikationen kan man till en stor del använda applikationen utan internetanslutning, beroende på applikationens huvudsakliga syfte, men det blir krångligt ifall innehållet skall uppdateras. Om man bestämmer sig på att ha allt innehåll samlat på en server är det enkelt att uppdatera allt innehåll men detta kräver internetanslutning. (Korf och Oksman, 2015).

Nackdelar med hybridapplikationer är att deras prestanda är en aning långsammare än en nativapplikation och de stöder inte programmeringsspråket PHP på applikationssidan.

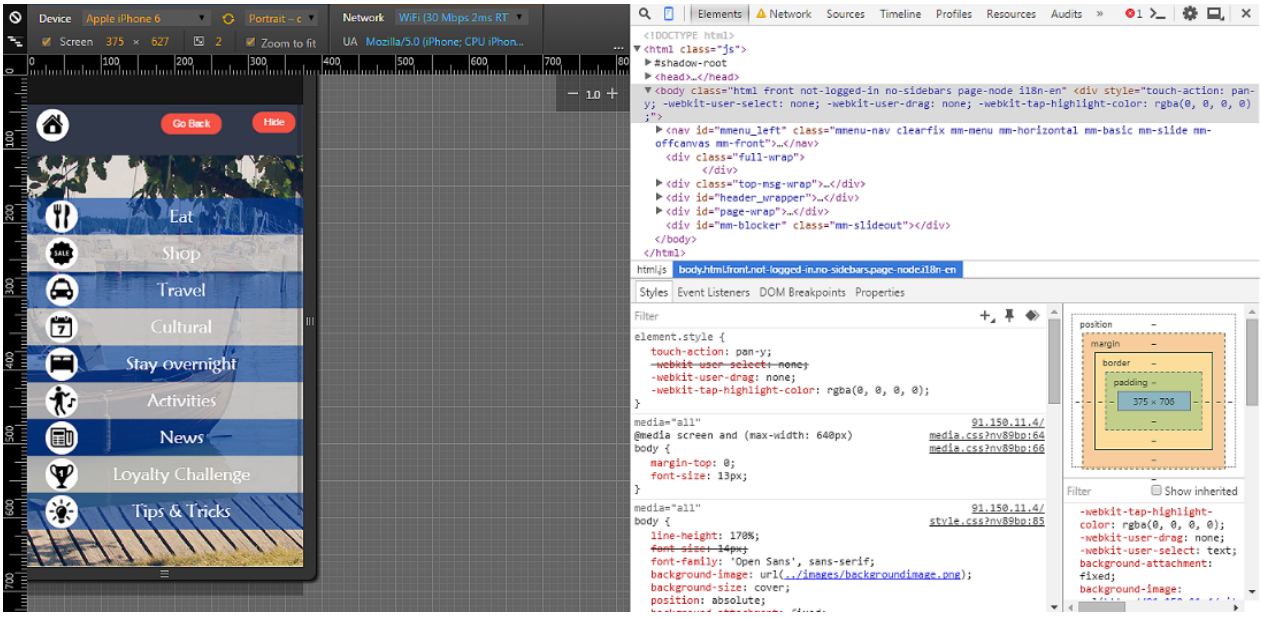
2.5 VERKTYG SOM ANVÄNTS

Då man utför ett större webb- eller applikationsprojekt krävs det ett flertal programmerings-verktyg för att avklara de olika uppgifterna. Till dessa verktyg inkluderas felsökningsverktyg, kodredigeringsprogram, serverkommunikationsprogram och databashanteringsverktyg.

2.5.1 Google Chromes ”Verktyg för programmerare”

Google Chrome är en av de populäraste webbläsarna som används då man surfar på Internet. Med denna webbläsare finns ett utvecklarverktyg som är till stor hjälp då man vill se eller snabbt test editera webbsidans kod. Vilket kallas för: Verktyg för programmerare.

Verktyg för programmerare kan enkelt initieras med hjälp av att trycka på tangentbords-knappen ”F12”, tangentbordskombinationen ”CTRL + Shift + I” eller högerklicka på sidan och välja ”Granska komponent”. Webbsidan förminskas temporärt och en massa verktyg för webbmanipulering dyker upp, vilket kan ses från Figur 4.



Figur 4. Google Chromes verktyg för programmerare.

Denna nya vy kan användas för att se hur mottaglig webbsidan är till olika enheters skärmbredd. Vilket görs genom att man antingen drar på skärmens yttre kanter men det går även att välja en enhet som exempelvis Apple iPhone 6 eller Google Nexus 10 från kategorin ”Device”. Förutom detta är vårt användningsområde med denna vy även till för att granska ifall det finns felmeddelanden, som kan tyda på programmeringsfel och för att granska

identiteten på växlande föremål som bland annat knappar. Att veta identiteten på föremål underlättar designarbetet på CSS delen.

2.5.2 Notepad++ & Adobe Dreamweaver CS6

Notepad++ och Adobe Dreamweaver CS6 är båda textbehandlare eller kodredigeringsprogram som stöder ett stort omfång av diverse programmeringsspråk. Dessa verktyg används av programmerare som skall editera kod.

Notepad++ är en lätt och gratis version av textbehandlare. Nyttan med att använda Notepad++ är att programmet inte kräver alltför mycket prestanda från datorn. Man kommer snabbt igång med programmet och dess enkla användargränssnitt ger mer utrymme för koden. Notepad++ har också funktionen att se var parenteser börjar och slut, utöver detta går det också att se var andra programmeringstaggar börjar och slutar.

Adobe Dreamweaver CS6 är sen igen ett allt-i-ett-verktyg som är avgiftsbelagt. Men programmet har en framstående fördel då den kan föreslå vad användaren kan skriva in i koden för programmeringsspråken HTML, CSS och JavaScript. Därtill alarmerar programmet ifall koden innehåller ogiltiga eller felaktiga kodsträngar och rapporterar var de befinner sig. Denna funktion existerar inte i Notepad++.

Orsaken till att vissa specifika kodredigeringsprogram används då man programmerar är att de kommer med inbyggda färdigheter som underlättar programmeringsarbetet. Både Adobe Dreamweaver CS6 och Notepad++ färgar koden på ett sådant sätt att man enkelt vet vad som är vad. Färgkombinationerna varierar beroende på vilket programmeringsspråk som används.

2.5.3 PuTTY

PuTTY är en terminalemulator och har sitt primära användningsområde till att ansluta sig till andra sammankopplade datorer via ett datornätverk. Med hjälp av programmets fjärrinloggning kan man utföra funktioner och starta upp program från en annan dator. När vi arbetar på distans underlättar PuTTY verktyget ifall man vill redigera rättigheter, ändra i filer och/eller installera program till servern.

Programmet stöder krypterade och klartextprotokoll för att utföra denna anslutning. I vårt fall använder vi oss huvudsakligen av ett protokoll vid namn SSH. SSH eller Secure Shell är ett krypterat nätverksprotokoll som används då man behöver starta en textbaserad anslutning till

en annan dator. I detta fall handlar det om en anslutning till servern. På en Linux server kallas programmet som skall installeras för OpenSSH. OpenSSH kan installeras samtidigt som operativsystemet men det går också i ett senare skede att installeras genom att skriva ”sudo apt-get install openssh-server openssh-client”. Porten som används till SSH är port nummer 22.

2.5.4 FileZilla & WinSCP

FileZilla och WinSCP är båda gratis FTP-klienter som används i huvudsak för att överföra filer från en dator till en annan. Vi använder oss av programmen för att överföra material till och från servern. Dessutom kan man enkelt navigera sig till olika filer på servern tack vare programmens GUI, och när man väl har behov av att redigera i en fil kommer textbehandlings-programmen väl till hands.

FTP är ett nätverksprotokoll, men det vi huvudsakligen använder oss av kallas för SFTP. SFTP står för Secure File Transfer Protocol eller SSH File Transfer Protocol. Inloggningen med standard FTP är osäker och utförs i klar text vilket gör detta otroligt osäkert. SFTP använder sig däremot av SSH vilket ger den förmågan att kryptera inloggningsuppgifterna som gör det mycket svårt för attackerare att få de data som de är ute efter. SFTP kör likväl som SSH på port nummer 22. Medan standard FTP körs på port nummer 21.

2.5.5 phpMyAdmin

phpMyAdmin är ett gratis verktyg för hantering och administrering av MySQL. När man väl installerat och modifierat phpMyAdmin på servern kan man nå det grafiska gränssnittet med hjälp av en webbläsare. Man kan utföra växlande handlingar med hjälp av phpMyAdmin. De vanligaste är att hantera databaser, tabeller, användare och rättigheter. Verktyget underlättar också säkerhetskopiering av databasen.

2.6 DEFINITIONER

Programmeringsspråk kommer i olika slags av filformat, vilka var för sig innehåller information i ett textformat. Textbehandlarprogrammen avkodar informationen och gör det möjligt för användaren att avläsa texten.

2.6.1 HTML

HTML står för ”Hyper Text Markup Language” och använder .html som sin filändelse. HTML har en stor betydelse för alla webbsidor som befinner sig i dagens läge på Internet. Med HTML kan man ange sidans struktur och till en viss del hur den skall se ut, meta information, lägga in bilder och infoga andra programmeringsspråk som till exempel JavaScript och CSS.

Man kännetecknar att filformatet är HTML i början tillsammans med DOCTYPE-deklarationen. Därefter följer en variation av flera olika markeringar eller taggar, som oftast kommer i par med en starttagg och en sluttagg. Mellan dessa taggar förekommer ett element, vars funktion är beroende på hurdana taggar som används (Hagberg & Hellström, 2002, s. 38–39).

Det finns en stor variation på HTML taggar. Vilka är exempelvis html, head och body. Taggarna <html> och </ html> beskriver att detta är ett HTML-dokument och samtidigt visar var HTML koden börjar och avslutas. Innanför de här taggarna finner man vanligtvis head- och body-taggar. Av princip skall varje webbsida ha <head> och </ head> taggar i sig. Det som finns innanför de här taggarna är osynligt för besökare, men innehåller väsentlig information om själva webbsidan. Webbsidans titel och länkar till externa globala formatmallar eller filtyper förekommer omgivna av head-taggarna. Själva innehållet för webbsidan placeras innanför <body> och </ body>. Allt det material som skall visuellt presenteras till besökaren finns innanför dessa body-taggar. Materialet kan variera allt från vanlig text till bilder och från radbrytningar till tabeller. (Hagberg & Hellström, 2002, s. 53-54).

2.6.2 PHP

Namnet PHP är en förkortning av ”PHP: Hypertext Preprocessor”, och har sitt användnings-område på servern. Filformatet PHP är populär inom webbutveckling och kan köras som ett eget filformat, med sin unika filändelse .php. Man kan använda PHP tillsammans med HTML vilket också är ett av de mest vanliga webbprogrammeringsspråken i användning.

Användningsområden för PHP är mångfaldiga, men de vanligaste är bland annat att generera dynamiskt innehåll till webbsidan, samla ihop information från olika formulär, skicka och ta emot cookies, skapa, öppna, läsa, skriva, stänga och ta bort filer från servern, kryptera data,

lägga till, ta bort eller ändra data från databasen, och styra användaråtkomst. PHP är ett programmeringsspråk som arbetar enligt uppfattningar och tydningar, vilket betyder att koden inte behöver kompileras vid testning. Detta gör programmeringsspråket snabbt och enkelt att använda eftersom man inte behöver vänta på att koden kompileras mellan varje testkörning. Som tidigare nämnt baserar sig Drupal på PHP, men även en stor del av de moduler som används är skrivna med PHP. (Brown, 2015).

Programmeringsspråket PHP är innesluten i speciella start- och sluttaggar. Med starttaggen <?php inleds PHP och användaren kan påbörja programmeringen, medan med sluttaggen ?> avslutas det hela. Ifall man behagar till att infoga ett PHP-skript in till en HTML filtyp kan de förslagsvis utföras enligt Kod 6. Nedanstående exemplet kommer att infoga texten ”Hello World” innanför paragraftecknet <p>. Ifall detta granskas med en webbläsare kommer det att stå ”Hello World” på webbsidan.

Kod 6. Exempel på HTML inuti PHP, vilket är återigen inuti HTML.

```
<html>
  <body>
    <?php
      echo ”<p>Hello World</p>”;
    ?>
  </body>
</html>
```

2.6.3 JavaScript

JavaScript är ett programmeringsspråk som kan justera beteendet och interaktiviteten på webbsidor. JavaScript kännetecknas med .js filändelsen. Allt från funktioner, kontroll granskningar, animeringar, ändringar på sidans design och spel är möjliga att göra med hjälp av JavaScript. Till skillnad från PHP har JavaScript sitt användningsområde på klientsidan, men har dock möjligheten till att tala med servern tack vare AJAX.

Man kan länka externa JavaScript filer och/eller bädda in JavaScript till HTML filformat. Länkningen av externa filer kan se ut enligt Kod 7. Då man bäddar in JavaScript till HTML använder man script-taggar, JavaScript-koden kommer då alltså att befinna sig mellan <script> och </script>. (Hagberg & Hellström, 2002, s. 407).

Kod 7. Exempel på länkningen mellan en extern JavaScript fil till HTML

```
<script type="text/javascript" src="myforms.js"></script>
```

Till JavaScript finns det olika typer av bibliotek som har egna specifika funktioner som kan underlätta diverse uppgifter. De typer av bibliotek som vi har tagit oss bekantskap med detta projekt är AJAX, JSON, jQuery och jQuery Mobile.

AJAX är en förkortning av namnet ”Asynchronous Javascript And XML” och med hjälp av detta är det möjligt att ta emot ny information från servern utan att behöva ladda om sidan. Allt från olika chatt-system till realtidsuppdateringar kan utföras med hjälp av AJAX. JSON står för ”JavaScript Object Notation” och tar hand om överföring av läsbartext till och från en server eller webbsida.

Meningen med jQuery är att förenkla olika händelser, animeringar, HTML-manipulation och AJAX. Från Kod 8 kan man observera ett exempel på hur skillnaden kan se ut mellan vanlig JavaScript är med jQuery, då uppdraget är att ändra textens färg till vitt på ett objekt med namnet test.

Kod 8. Ett exempel på skillnaden mellan JavaScript och jQuery

JavaScript

```
document.getElementById("test").style.color = "#FFF";
```

jQuery

```
$("#test").css("color", "#FFF");
```

jQuery Mobile är utvecklad från jQuery vilket också är HTML5 och CSS3 baserat. jQuery Mobile har tagit med sig från jQuery dess fina egenskap till förminskade koduppbyggnader. Med hjälp av jQuery Mobile kan man utforma vanliga webbsidor till att vara allt mer mottagligare för varierande skärmbredd, speciellt för mobila enheternas skärmar. Detta gör att jQuery Mobile är ett perfekt verktyg för webb och hybridapplikationstyper.

2.6.4 CSS

CSS är en förkortning av Cascading Style Sheets och har sin egen unika .css filändelse. CSS står för all design och layout som en webbsida använder. Liknande som för JavaScript har man även möjlighet till att bädda in CSS och/eller referera externa .css filer till HTML filformat. När man refererar till externa CSS filer kan det se ut på följande sätt som Kod 9 visar. Men ifall man behagar till att bädda in CSS till HTML, då använder man style-taggar. Då skall CSS-koden befinna sig innanför följande <style> och </style> taggarna. (Hagberg & Hellström, 2002, s. 338).

Kod 9. Exempel på länkningen mellan en extern CSS fil till HTML

```
<link type="text/css" rel="stylesheet" href="style.css" />
```

Med CSS kan man manipulera hur webbsidan ser ut på mobila enheter med varierande skärmbredd genom att använda Media Queries. Dessa Media Queries skrivs ut i CSS filer med taggen @media, varefter det medföljer en eller flera kriterier. Detta kan bland annat visas enligt nedanstående exempel. Ifall enhetens skärm är åtminstone 500 pixel eller mera bred uppfylls kriterierna i exemplet och då kommer body i det här fallet att ändra sin bakgrundsfärg till svart.

Kod 10. Exempel på Media Query

```
@media screen and (min-width: 500px) {
  body {
    background-color: #000;
  }
}
```

2.7 SERVER SIDANS STRUKTUR

För att kunna sätta igång med ett applikationsprojekt där en databas existerar bör man först lägga upp en server som lagrar alla inställningar och data som behövs. Ett vanligt operativsystemsalternativ till en webbserver är Linux. På operativsystemet Linux är det relativt okomplicerat att installera de program som krävs för att det skall fungera. De följande programmen är FTP, SSH och paketet LAMP. Utöver mjukvaran krävs det också tillgång till mobila enheter. Dessa används för att köra regelbundna test på applikationen.

2.7.1 LAMP

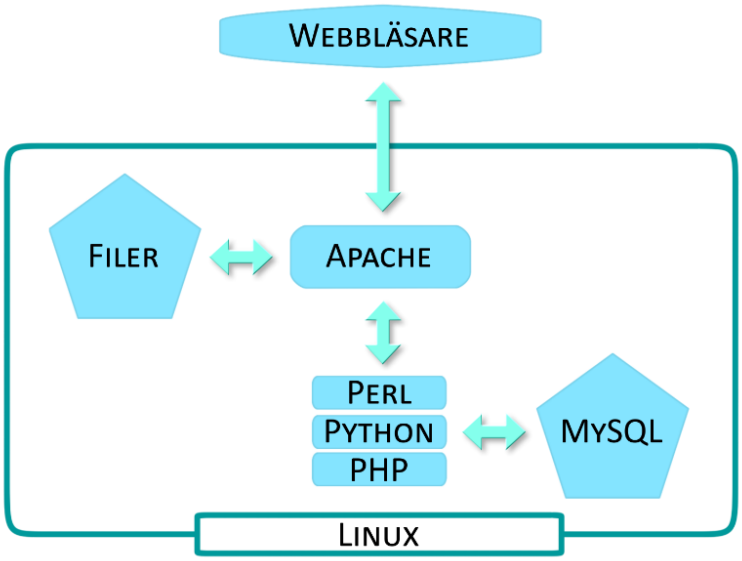
LAMP står för Linux, Apache, MySQL och PHP. Tillsammans kallas dessa också för en ”Web stack” då man använder Linux som operativsystem, genom att Apache fungerar som webbservern, MySQL som databas och PHP som programmeringsspråk. PHP kan vid behov bytas ut mot programmeringsspråken Python eller Perl enligt användarens behov.

Det finns tre stora fördelar med att använda LAMP som en utvecklingsplattform.

- För det första är att alla komponenter är gratis och/eller använder öppen källkod. Eftersom de är kostnadsfria utökar det mottagligheten och lockar till sig ett stort antal användare. Alla personer är inte beredda att betala stora summor för att få en fungerande webbserver.

- För det andra är öppen källkod licenser nästan restriktionslösa, vilket betyder att personer kan utveckla program som använder LAMP utan att behöva betala licenskostnader vid distributionen av användarens färdigt utvecklade program.
- Den tredje och största fördelen med LAMP är att alla personer har tillgång till källkoden, detta betyder att alla användare kan fixa buggar och förbättra programmen. Det här medför att LAMP är otroligt flexibelt vilket inte är tillgängligt vid kommersiella program.

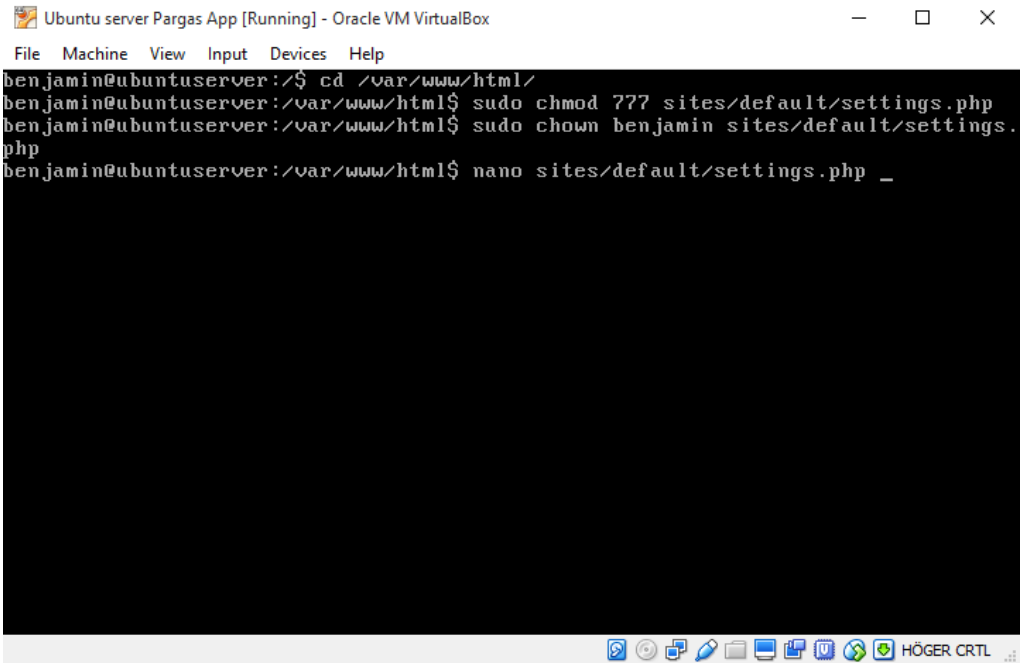
Hur alla delar i LAMP fungerar tillsammans visas på Figur 5. MySQL är det som är bakom allting. PHP, Python eller Perl kommunicerar med MySQL databasen och med Apache HTTPD servern som i sin tur har tillgång till filerna som är i användning. Allt detta är installerat och befinner sig på Linux operativsystemet. Apache är den vanligaste webbservern som används till webbsidor och webbapplikationer, vilket slutligen illustrerar helheten till webbläsaren. (Brown, 2005).



Figur 5. Illustration på hur LAMP programmen fungerar tillsammans (Brown, 2015, omarbetad av skribenterna).

Den version av Linux som används inom projektet är Ubuntu server 15.04. Det är ett operativsystem (OS) som inte har ett grafiskt användargränssnitt (GUI). Det liknar mycket den Command Prompt som finns i Windows. Figur 6 visar hur användargränssnittet ser ut hos en Ubuntu server, vilket är installerat på en virtuell maskin. Programmet som används för den virtuella datorn är VirtualBox av Oracle. Det finns flera andra program som också kan utföra

liknande virtuella miljöer, exempelvis VMware, men enligt våra personliga erfarenheter är VirutalBox ett bättre alternativ än de andra eftersom dess installationsprocess är minst frustationstillkallande. Vid användning av VMware krävs det extra konfiguration för att allting skall fungera. Ett viktigt område att komma ihåg vid installationsprocessen då man vill få en egen IP-adress av modemet till den virtuella maskinen är ställa in nätverket att vara ansluten i ett bryggat format. Om man inte utför detta krävs det manuell konfiguration av nätverks-inställningarna.



Figur 6. Ubuntu server gränssnitt.

Vanliga kommandon som krävs för att använda en Ubuntu OS som saknar GUI är *cd*, *sudo*, *chmod*, *chown* och *nano*.

- **Cd** används då man navigerar från mappar till filer på operativsystemet. Om man exempelvis vill gå in till mappen html som finns inne i var/www/ måste man skriva ”cd /var/www/html”.
- **Sudo** används för att få root privilegier. Med de här förhöjda privilegierna går det att göra ändringar som vanligtvis endast användaren *root* skulle ha tillgång till. Hit hör bland annat att ändra rättigheter på mappar.

- **Chmod** och **chown** används inte lika mycket som cd eller sudo men de är mycket viktiga att känna till eftersom det är ofta som användaren www-data eller root automatiskt skapar en mapp som ingen annan användare har tillgång till. När vi är flera personer som arbetar på samma projekt är det också viktigt att använda dessa då man behöver få rättigheter till en fil som är skapad av en medarbetade. Med hjälp av chmod och chown går det att ändra vem som äger mappen och/eller ändra skrivrättigheterna till den. Skrivrättigheterna i Ubuntu är indelade i läs-, skriv- och exekverarättigheter. Därtill finns det också ägare, grupp, publika rättigheter. För att exempelvis ändra att alla användare som finns på servern skall ha tillgång till att ändra i en mapp skall man skriva ”*sudo chmod 777 <mappens namn>”*, och om man vill att alla dessa rättigheter skall placeras till alla filer som mappen innefattar genomförs det genom att lägga till ett *-R* efter mappens namn. För att ändra mappens ägare skall man skriva ”*sudo chown <användare> <mappens namn>*”. Det rekommenderas dock att inte ha fulla skrivrättigheter på filer. Just enough principen är en bra vägledare vid denna punkt också.
- **Nano** är ett textbehandlingsprogram som finns från början med i bland annat Ubuntu. Det är lätt att använda eftersom allt man behöver veta för att använda står skrivet i programmets nedre kant. Programmet startas genom att skriva ”nano <filens namn>”. Om användaren som används inte har skrivrättigheter till filen går det att skriva *sudo* framför *nano* för att få tillgång.

MySQL Relational Database Management Systemet är ett enkelt men kraftigt program som direkt fungerar efter installationen utan några extra konfigurationer. Dess standardinställningar är mycket bra men mera avancerade användare kan konfigurera den för att höja på prestandan. Vi använder dock standardinställningarna eftersom de fungerar bra till det ändamål som vi har planerat använda applikationen och databasen till.

MySQL stöder SQL, som står för Structured Query Language, vilket i sin tur gör det möjligt att hantera relationsdatabaser. SQL gör att manövreringen över relationsdatabaserna blir lätthanterbart. Med hjälp av relationsdatabaser kan man skapa och utveckla databaser på företagsnivå. På grund av detta använder både stora och små företag MySQL. Databas motorn som används till applikationen är InnoDB som är speciellt bra då det handlar om höga nivåer av transaktioner (Brown, 2005).

2.7.2 Uncomplicated Firewall

Uncomplicated Firewall (UFW) är en brandvägg som automatiskt kommer med på alla Ubuntu server installationer. Då man har en webbserver krävs det att det finns en brandvägg som har möjligheten att blockera trafik på portar som man inte vill att personer skall kunna komma åt. De vanligaste portarna som används vid vardaglig Internet användning är 21, 22, 80 och 443. Port 21 används av standard FTP och port 22 är för SFTP och SSH. Då man besöker webbsidor på Internet använder man port 80 eller 443, där port 80 är det normala HTTP och port 443 är den säkrare versionen HTTPS. Portarna går att ändra till andra men i vårt fall använder vi standardportarna.

Vanliga kommandon som används för att komma igång med UFW brandväggen åskådliggörs i kommando format från Kod 11.

Kod 11. Kommando tabell för Uncomplicated firewall

```
1. sudo ufw enable
2. sudo ufw default deny
3. sudo ufw allow <port nummer>
4. sudo ufw allow from <IP adress> to any port <port nummer>
5. sudo ufw delete <regel typ> from <IP adress> to any port <port
nummer>
```

Vid ny Ubuntu-installation är brandväggen avstängd, vilket innebär att man måste manuellt starta den. Man startar UFW med kommando nummer 1, från Kod 11. Därefter lönar det sig att köra kommando nummer 2 för att blockera all trafik till servern. När trafiken väl är blockerad går det att endast tillåta trafik på de portar som man vet att används. Öppning av portar sker genom kommando 3 eller genom kommando 4 beroende på om man bara tänker öppna det till en specifik IP eller till hela världen. Via kommando 5 går det sedan att ta bort dessa regler.

2.8 TEST AV APPLIKATIONENS FUNKTIONSDUGLIGHET

Under en programmeringsprocess är det viktigt att testa koden och de funktioner som har skrivits för att helt enkelt se ifall de fungerar. Då det handlar om programmering av en mobil-applikation finns det två sätt att testa den. Det ena sättet är att ha en emulator som emulerar den mobila enheten och andra är att ha en fysisk maskin som applikationen installeras på efter varje ändring.

Det finns enskilda emulatorer för varje enskilt operativsystem då man håller på att utveckla en nativapplikation. Nativapplikationer som skall lanseras till Android och som använder programmet Android Studio har tillgång till en färdigt utrustad Android emulator. Apples motsvarighet för iOS är Xcode och Microsofts för Windows är Visual Studio, där var och en har sin egen emulator. Eftersom vi använder oss av PhoneGap går det inte att använda de ovannämnda programmen för emulering. Om man använder PhoneGap har man tillgång till användningen av ett program vid namn Ripple. Ripple än enbart tillgängligt hos webbläsaren Google Chrome och fungerar som ett tillägg till denne. Med hjälp av detta skall det gå att provköra applikationen medan programmeraren kodar, men eftersom Ripple inte har uppdaterats på en lång tid är det inte mera ett förnuftigt alternativ på grund av att pålitligheten hos den funktionella delen är föråldrat. PhoneGap uppdateras med jämna mellanrum, varav en av uppdateringarna var en som inte Ripple kan emulera till 100 %.

Som redan nämndes är inte Ripple ett säkert kort då man provar en applikation, vilket betyder att det enda förnuftiga utvägen är att använda en fysisk mobil enhet som en smarttelefon och läsplatta. Nackdelen är dock att det krävs tillgång till alla de tre vanliga operativsystemen. Varje gång en ändring hos den mobila applikationsdelen har gjorts måste man skapa en ny färdigt kompilerad applikation från PhoneGap. Fördelen med att testa applikationen på en fysisk enhet är att om den fungerar där kommer den med stor sannolikhet också fungera på den distribuerade applikationsfilen.

2.9 SÄKERHETSKOPIERING

Då ett programmeringsprojekt utförs kan det flera gånger uppstå stora problem som kräver att man bör återställa (rollback) hela systemet till ett tidigare definierat tillstånd för att få projektet till ett funktionerande läge. Utöver detta kan det också ske dataförlust ifall systemet som filerna finns på kraschar. För att kunna utföra återställningar krävs det att filerna, systemet och databasen säkerhetskopieras.

Under vårt projekt utfördes säkerhetskopieringar dagligen då vi arbetar. Eftersom vi hade allting på en VirtualBox var säkerhetskopieringsprocessen lätthanterbar att utföra då det endast krävs kopiering av en mapp. Vi använder molnlagringstjänster eftersom filerna då är enkelt tillgängliga för andra datorer som har tillgång till mappen.

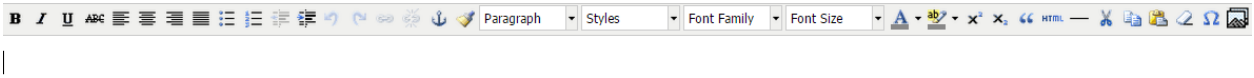
3 APPLIKATIONENS FUNKTIONER

Vi förstod direkt att denna applikation är avsedd till att bli använd av många personer samtidigt som betyder att applikationen skall vara robust och se till att skicka innehålls-uppdateringar till mobila enheterna i realtid. Därtill skall applikationen även förse användarna med funktioner som inloggnings system. För att kunna utföra allt detta är vi behov av en server, vilket resulterar till att det krävs en databas. Databasen kommer att vara belägen på en virtuell maskin som använder operativsystemet Linux server. Vi valde oss att använda Linux över Windows eftersom vi har mera erfarenhet med att använda Linux för liknande syften. Att lägga upp en webbserver på Linux är mindre komplicerat än hos Windows eftersom det går att göras direkt via operativsysteminstallationen.

Nästa val vi hade framför oss var att välja mellan de tre olika applikationstyper: nativ, hybrid eller webb. Vi bestämde oss direkt att applikationen skulle vara av hybridformatet via PhoneGap, eftersom vi då skulle ha tillgång till ett stort antal externa insticksmoduler från varierande källor. I några veckors tid arbetade vi intensivt med utvecklingsverktyget DrupalGap. DrupalGap använder en Drupal webbsida som bas medan den bibehåller sin hybrid-applikationskaraktär. Eftersom verktyget är otroligt nytt och har enbart några år bakom sig, vilket är ungt för en SDK, har det inte ännu alla de funktioner som krävs för att vi skall kunna uppnå de funktionella kraven som har satts på Pargas applikationen. När vi kom fram till denna insyn var vi redan långt på vägen med utvecklingen av webbsidan som skulle fungera som bakgrundsprogram (backend) att vi direkt bestämde oss för att övergå till en webb-applikationstyp istället för att avkasta allt vi åstadkommit med. Applikationstypen skulle ha förmågan att uppfylla de funktionella krav som applikationen har. Kraven är, som redan har nämnts tidigare, att innehålla ett nyhetsflöde, tidtabeller för transport inom Pargas stad, evenemang och aktiviteter. Utöver dessa krav skall även företag kunna skapa företagsprofiler där de kan lägga relevant information om sitt företag.

Eftersom Drupal används som grund kräver det att dess tema eller layout är fullständigt omgjord till det mobila formatet. Robert Lönnberg har anförtrotts uppdraget till att skapa det grafiska utseendet, vilket innebär applikationens design och layout. Drupal valdes till det CMS vi använder oss av eftersom vi har lärt oss grunderna till det under studietiden vid Yrkeshögskolan Novia.

Användningen av Drupal ger tillgång till ett stort antal moduler som utökar dess funktionalitet mångfaldigt. En essentiell modul som nästan alltid skall finnas med på en Drupal 7 webbsida är Wysiwyg. Wysiwyg är förkortningen av ”What you see is what you get” och ger textredigeringsverktyget på webbsidan en uppgradering genom att den förstår formateringarna ny rad och annat nödvändigt automatiskt. Figur 7 visualiserar vilka funktioner som blir tillgängliga genom modulen Wysiwyg. Förutom den tidigare nämnda modulen används ett flertal andra moduler och de är nämnda i Bilaga 2.



Figur 7. Wysiwyg visualiseras.

3.1 INNEHÅLLSTYPER

Applikationen skall fungera som ett informationsflöde för Pargasborna och turister. Informationen skall vara relevant för besökarna och vara kategoriserat på ett logiskt sätt till bland annat nyheter, tidtabeller, evenemang och aktiviteter.

3.1.1 Nyheter

Nyhetsfunktionen skall fungera som en metod för företag och andra personer att förmedla om vad som är aktuellt inom staden. Företag kan också använda denna funktion för att upplysa kunder om aktuella erbjudanden eller händelse som gäller företaget. Det skall vara möjligt att binda nyhetsinnehållet till företagen. Nyheterna skall inte vara indelade i skilda kategorier och skall inte heller ha några specialfunktioner.

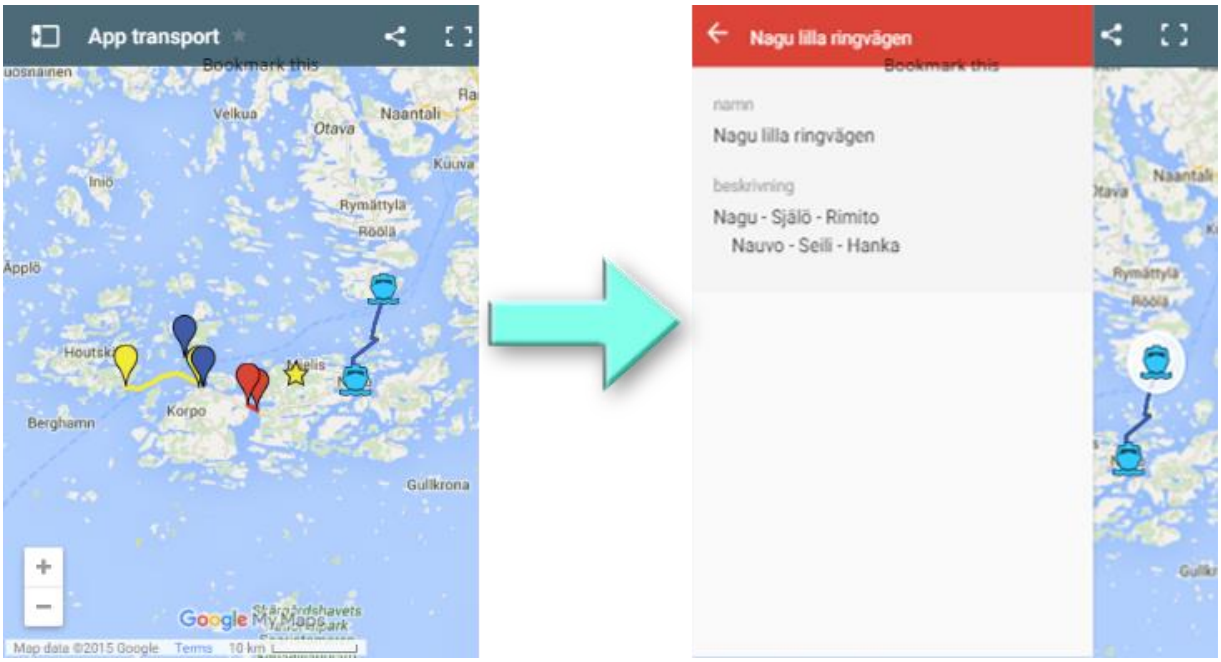
3.1.2 Tidtabeller

Pargas är en skärgårdsstad som betyder att den har ett stort antal öar. För att personer lätt skall kunna finna transportmetoder från ett ställe till ett annat krävs det att all information befinner sig på ett ställe. Till kategorin tidtabeller skall besökaren få information som tidtabellerna för färjor och bussar. Men information om taxitrafiken och beställningsbussar skall också vara tillgänglig.

Färjetrafiken i Pargasskärgård är något som är aktiv användning, men vid nuvarande tillfället finns det ingen metod att på ett enkelt sätt ta reda på när nästa färja går från punkt A till B.

Den nuvarande webbsidan som har information om alla dess rutter är endast en mellanhand till ett flertal andra webbsidor som enbart tar upp de färjor som företaget skör.

En av applikationens uppgifter är att komprimera alla kartor och rutter till en enda plats. En tidtabellslista finns också till hands. Den komprimerade kartan är uppbyggd av Googles My Maps där vem som helst har möjlighet att skapa innehåll till en karta. Från att enbart lägga in vanliga platsmarkörer går det också att rita in streck på kartan. My Maps är uppbyggd med en baktanke att det skall vara möjligt implementera kartan till en webbsida eller andra plattformar som har IFrame som en funktionell lösning. Figur 8 är en illustration hur en IFrame ser ut då den är implementerad in till webbapplikationen. Information om rutten kommer fram då användaren klickar på någon av de ikonerna.



Figur 8. Implementerad IFrame till webbapplikationen.

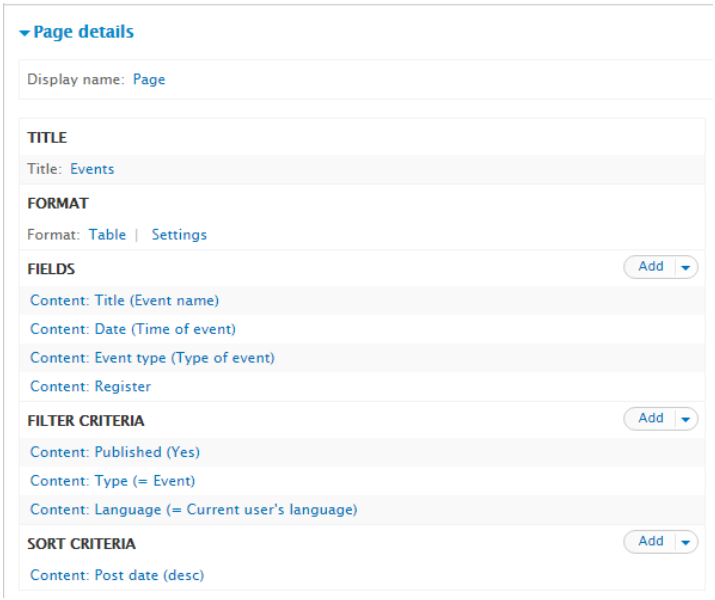
Busstrafiken inom Pargas och till grannkommunerna går att granskas från Matkahuoltos hemsida, men det finns inget lätt sätt att se när de kör omkring innanför Pargas området utan att veta var man är och vart man skall. Applikationens bussfunktion skall visa alla rutter och tidtabeller som körs innanför Pargas stads gränser och till nära platser. Funktionen kommer att precis som färjetrafiken att använda Google My Maps som en bas.

Till skillnad från färja och buss innehåller taxi kategorin inte några tidtabeller eller kartor. Detta kommer enbart att vara en informativ sida om de olika företag som kör taxi i Pargas och skärgården.

3.1.3 Evenemang

Evenemang är en händelse som exempelvis en konsert eller utflykt, det vill säga något som inte händer regelbundet. De fälttyper som används för att skapa ett evenemang är titel, brödtext, datum, bild, evenemangstyp och registrering. Med hjälp av dessa fält går det att ge evenemanget ett namn och även enkelt kategorisera liknande evenemangstyper med varandra. Utöver detta kan man ställa in vem och under vilka tider det går att registrerar sig till evenemangen.

De moduler som används vid evenemangdelen är Views och Date. Views används för att skapa en lista som visar alla de evenemang som skapats på hemsidan, medan Date krävs till att få in ett datumfält på innehållstypen. Figur 9 åskådliggör vilka inställningar som används vid Views.



Figur 9. Överskådning av Evenemangs Views inställningar.

Inställningarna från Figur 9 kommer att ge en lista på alla de evenemang som har skapats. Listan innehåller dess titel, datum och hurdan evenemangtyp det kommer att vara. Utöver detta finns det även en registreringsknapp som möjliggör att besökare kan behändigt registrera sig till evenemangen.

För att arrangören till evenemangen skall kunna veta hur stort intresset är på ett specifikt evenemang är det viktigt att personer har möjligheten att registrera sig till dem. Det går att registrera sig på flera olika sätt, antingen med det användarnamn som man har på sidan eller via epost. Det finns också en funktion att registrera andra personer, denna funktion lades in för att en person skall ha möjligheten att registrera en hel grupp till ett evenemang utan att varje deltagare behöver göra det individuellt. När man registrerar andra går det att göras via epost eller med deras användarnamn på applikationen.

Registreringen sköts genom en modul som går vid namnet Entity Registration. Med hjälp av modulen kan administratörer skapa nya registreringstyper som man därefter kan använda som ett fält vid innehållstyper. Till applikationen används enbart en registreringstyp eftersom det inte krävs registrering vid någon annan funktionell del av applikationen.

3.1.4 Aktivitet

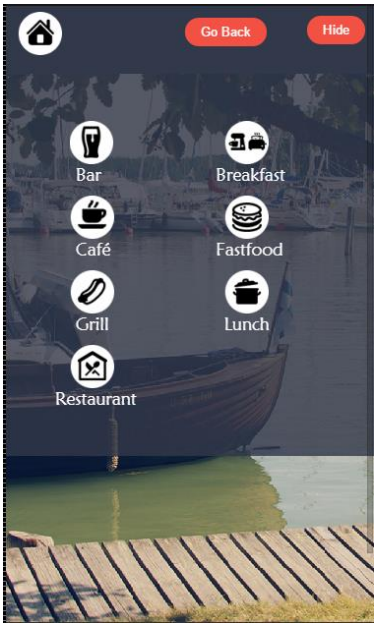
Till skillnad från evenemang är en aktivitet en händelse som är regelbundet återkommande. En aktivitet kan vara en biljardkväll på någon ungdomssal eller innebandyträningar vid en gymnastiksal. Meningen med dessa är att man inte skall kunna registrera sig till dem utan det skall enbart fungera som en informativ upplysning om de olika aktiviteter som händer. Deltagandet till aktiviteterna torde öka då flera personer blir upplysta eftersom då är det flera som har vetskapen om dem.

Vid skapningen av en aktivitet är det möjligt att definiera hurdan typ av aktivitet det handlar om, på liknande sätt som för evenemang. Man kan enkelt se på den typen av aktiviteter man är intresserad över genom att filtrera innehållet med bland annat underhållning, kurser, kultur och idrott.

3.2 TAXONOMI

Applikationens sökfunktion fungerar genom Drupals inbyggda taxonomi egenskap. På applikationen skall det finnas ett dussintal olika kategorier av företag som skall gå att söka mellan. Detta förverkligas genom att bygga upp vokabulärer som lagrar taxonomi termer. Vi har planerat att alla innehållstyper får sin egen vokabulär som innehåller relevanta söktermer (Falk, 2011, s. 80). Exempelvis skall innehållstypen företag innehålla en vokabulär där alla de olika företagstyperna och deras tjänster skall finnas. Om ett företag skapar en profil och väljer

termerna restaurang, bar och breakfast kommer det att vara möjligt att finna detta företag från de tre menyerna som går under termernas namn. Figur 10 illustrerar dessa menyer eller termer som finns under matkategorin.



Figur 10. Illustration av applikationens menyer från matkategorin.

Utöver detta kommer alla företag också ha möjligheten att lägga in egna sökord, eller som det kallas taggar, till sitt företag. Detta implementerades för att ge en mera frihet åt företag och det underlättar också vår arbetsbörda eftersom vi inte behöver tänka på alla möjliga sökord. Sökorden går sedan att användas i Drupals inbyggda sökfunktion. (Falk, 2011, s. 79, 85).

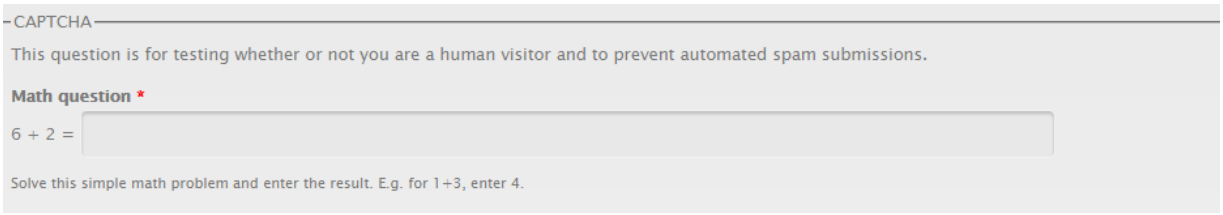
3.3 INLOGGNING OCH REGISTRERING TILL APPLIKATIONEN

Applikationen skall ha en funktion för inloggning och registrering av nya användare. Inloggning krävs för att en användare skall ha tillgång till de interaktiva funktionerna på applikationen. Det enda man kan göra utan att vara inloggad är att läsa publicerade artiklar och liknande innehåll.

Inloggning och registreringsfunktionen sköts på webbsidans sida genom en modul som går under namnet LoginToboggan. Modulen tillåter användare att logga in med både sin registrerade epost eller användarnamn. Samtidigt möjliggör det att ställa in minimumlängden på lösenordet. Utöver detta är det också rekommenderat att det finns ett robotskydd vid inloggningen så att inte spamrobotar och dylikt skapar konton på applikationen. För detta

använder vi oss av modulen Captcha som är en av de vanligaste metoderna för att skapa ett sådant skydd. Captcha tvingar registreraren att klara av något logiskt test. Den här logiska handlingen kan ta flera olika former. Den vanligaste är dock att man skall skriva in de siffror och bokstäver som visas på en bild. En annan typ som Facebook använder sig av är att man skall välja alla bilder av samma typ. Detta kan handla om att välja alla bilder som har ett lejon på sig.

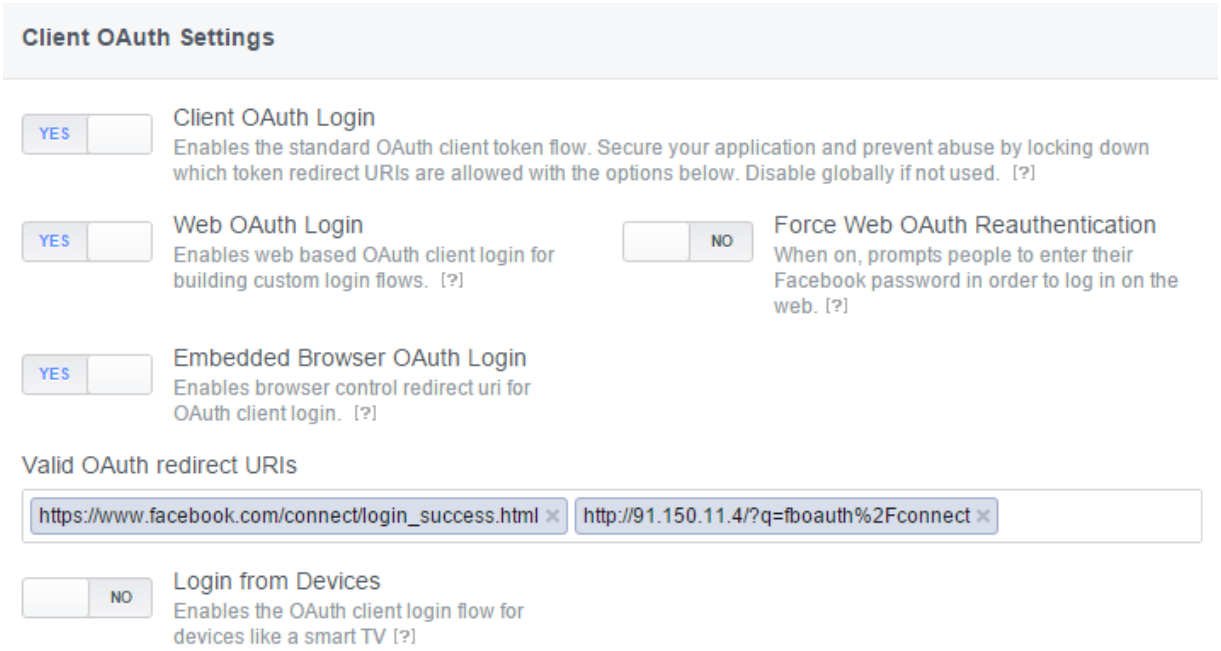
Den version av Captcha som används på applikationen är en matematisk kalkylation. Captcha frågar användaren vad svaret blir då man adderar två slumpmässigt valda tal. Figur 11 visar hurdan fråga det kan handla om.



Figur 11. Matematisk kalkylation med Captcha.

På applikationen går det även att logga in med Facebook. Funktionen implementerades för att det finns ett stort antal personer som har ett Facebook konto och vill använda det för att logga in på diverse saker. Facebook inloggning är mycket lätthanterat och säkert. Ifall personer är redan färdigt inloggad till Facebook på telefonen då behöver man endast trycka på en knapp för att logga in på Pargas applikationen. FBOauth är den modul som möjliggör inloggningen via Facebook. Då man loggar in på applikationen via Facebook kommer användarens e-post adress, profilbild och namnet på personen att bli importerade.

För att Facebook-inloggningen skall fungera på en Drupal hemsida krävs det inte bara en modul som ansvarar för allting, utan man måste även skapa en applikation på Facebooks developer hemsida. Adressen till hemsidan är *developers.facebook.com*. När väl en applikation är skapad är nästa steg att gå in till avancerade inställningar där man kan lägga in *Valid OAuth redirect URIs*. Beroende på om det handlar om en nativ- eller webbapplikation skall man lägga in två olika URL adresser, men om man är osäker går det att lägga in båda för systemet väljer det alternativet som går att använda automatiskt. Adresserna skrivs in enligt Figur 12 nedan.



Figur 12. Exempel på Facebooks Valid OAuth redirect URIs

Utöver dessa konfigurationer krävs det också att man använder två stycken identifikations-koder vid namn APP ID och App Secret. Dessa två koder finner man under Basic Meny rubriken på den Facebook applikation man skapat. Koderna skrivs in på Drupal webbsidan för att skapa en förbindelse mellan Facebook och webbsidan.

3.4 PROFILER

Företagsprofilerna är en central del av applikation. Den skall fungera som huvudmediet för företag genom att förmedla information om företaget till sina kunder och potentiella kunder. Till detta hör bland annat kontaktinformation och företagets geografiska position på en karta. Detta innebär att vi som programmerare måste ge möjligheten för dem att ha en exklusiv profil.

För att möjliggöra skapandet av en hyfsad profil använde vi oss först av en modul som heter Profile2. Den har egenskapen att lägga till informationsfält till profiler. Det går också att skapa flera olika profiler på en administratorsnivå men eftersom medlemmarna av applikationen inte har de rättigheter som krävs för att skapa en profil på den här vägen måste en annan metod användas. Metoden som används är att använda en vanlig innehållstyp som en profil. Genom den här metoden är det möjligt för en vanlig användare att skapa flera profiler utan att bli blockerad av brist på rättigheter.

Profilerna kommer att vara sorterade enligt hurdan produkttyp och/eller tjänst de erbjuder, som till exempel butik, frisör och restaurang. En applikationsanvändare kommer själv att kunna kontrollera till vilken kategori som företaget placeras till. Kategorierna görs genom att använda Drupals inbyggda taxonomi funktion där varje enskild kategorityp tillhör en taxonomi entitet.

Profilerna är programmerade på ett sådant sätt att de fungerar som grupper personer kan gå med i. Administratörerna i de olika grupperna har förmågan att skicka massmeddelanden till alla medlemmar. Modulen som används för att skapa denna gruppmiljö heter Organic Group. Organic Group möjliggör att alla innehållstyper kan bli grupper som användare kan gå med i. Applikation använder dock funktionen från Organic Group enbart till innehållstypen profil.

Med hjälp av Organic Group modulen är det möjligt att skapa flera företagsprofiler per användarkonto och ha olika administrativa medlemmar i dem alla. En administrativ medlem i gruppen kan gruppera allt nytt innehåll som skapas till gruppen. Kopplingen medför privilegiet att alla administrativa medlemmar i gruppen kan redigera detta nya innehåll utan begräsningar. Genom denna funktion kan flera medlemmar i samma företag ha möjligheten att redigera företagets innehåll. Drupals inbyggda rättigheter tillåter normalt enbart webbsidans administratörer och moderatorer att redigera andras innehåll men genom denna funktion förbigås detta och gruppens administratör kan själv välja vem som skall ha möjlighet till att redigera innehållet som är grupperat till gruppen. Figur 13 illustrerar vilka menyer det finns vid Organic Group-innehålls administrativa fönster.

Figur 13. Illustration the administrativa fönstren vid Organic Group-innehåll.

Utöver gruppfunktionen skall också profilen ha två flikar. där ena fliken innehåller all information om företaget i full text medan den andra fliken enbart innehåller kontakt-information och öppethållningstider. Funktionen är problematisk att genomföra med Drupal eftersom innehållshanteringssystemet inte är uppbyggd att fungera på detta sätt. Vi löste dock detta problem genom att använda modulerna Node Subpages och Field Collection. Node Subpage gör det möjligt att skapa block som visar flikar nedanför nodens innehåll. Fliken är länkad till en ny sida som kan innehålla en View eller ett fält. View inställningen skulle ha uppfyllt våra krav ifall vi hade fått den att enbart visat de fält som tillhör den aktuella profilen, men eftersom det är otroligt komplicerat, om inte omöjligt, att få till ett fungerande skick övergick vi till att använda fältinställningen istället. Fältinställningen hade dock problemet att den enbart kan visa ett fält. Detta problem löstes med modulen Field Collection som tillåter oss att sätta flera fält inne i ett fält som sedan via Node Subpage användar-gränssnitt går att länkas till. Slutligen fick vi ett flertal fält på den andra fliken.

3.5 KARTA

Vi använder oss av en karta för att man skall kunna visuellt och geografiskt visa var alla företag, butiker, attraktioner, aktiviteter eller dylikt befinner sig.

Den karta som vi använder oss till vår applikation är en modul vid namn Get Locations. Vi använder oss av en blandning mellan två olika versioner: den nyaste rekommenderade versionen 7.x-1.16 och utvecklings versionen 7.x-2.x-dev (laddat ner 28.8.2015). Modulen visar upp positioner på en Google Maps API version 3 karta och kräver enbart att modulerna Libraries och System skall vara aktiverade. Orsaken varför vi använder oss av denna modul för att fungera som applikationens karta är tack vare dess otroligt omfattande möjligheter för konfigurationer och inställningar.

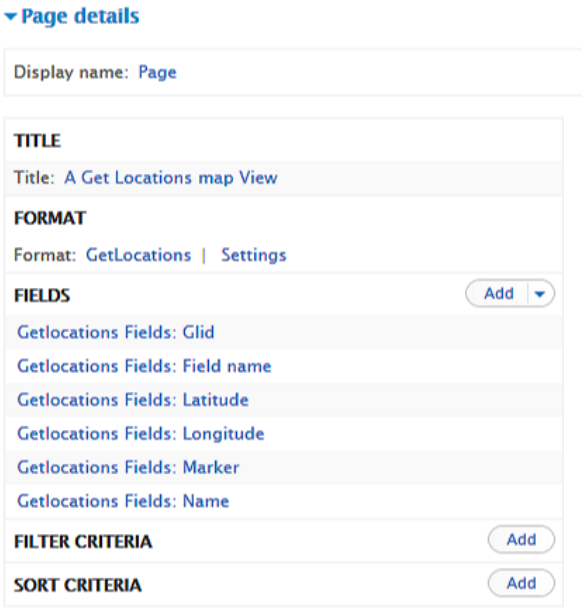
Get Locations har ett präktigt och intressant sätt på hur man kan ställa in de markörer som presenteras på kartan. Man har möjligheten till att välja ifall man vill att vissa innehållstyper har egna markörer eller ifall man vill att markörerna blir separerade och får unika markörer från taxonomin. Man kan skapa en ny taxonomigrupp som innehåller enstaka namn för en kategori. Exempelvis taxonomi gruppen ”Företagstyp” innehåller då bland annat Restaurang, Boende, Aktivitet, Transport och Butik. Som en sid not, man har då denna taxonomi kategorisering som ett obligatoriskt fält tillsammans med kartan då användaren skapar nytt

innehåll. På detta sätt kan administratören ställa in från sidans config/services/getlocations att Restaurang, Boende, Skola, Bibliotek, Kiosk, Hotell, Supermarket etc., får egna unika markörer.

De markörer som man har att välja mellan skall placeras in till mappen sites/all/libraries/getlocations/markers. Man kan även skapa egna markörer vilket betyder att möjligheterna är oändliga för de som vill ha mer än det som erbjuds. Huvudsaken är att man skapar en ny mapp, lägger markörerna dit och inkluderar en konfigurationsinställnings-formatfil med .ini som filändelse, annars förstår inte Get Locations att den kan använda de egna gjorda markörerna.

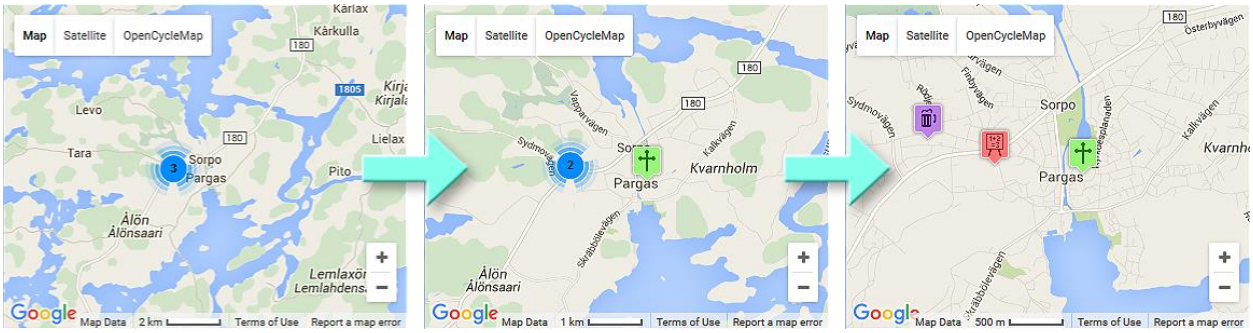
När väl användaren skall exempelvis skapa nytt innehåll, där kartan är aktiverad, kan användaren välja mellan att skriva in gatuadressen eller vänsterklicka/trycka på kartan för att markera eller flytta positionen på markören. Då man håller på att skriva in en gatuadress kommer systemet att föreslå finska orter, adresser eller företag som redan är registrerade till Google Maps. Användaren har även möjlighet till att zooma in eller ut och byta kartan till satellitvy. Kartan är därför enkel att använda och det är ett av de största kriterierna för valet av en karta till denna applikation.

Då man skall skapa en Get Locations karta med flera markörer, kommer detta att göras med hjälp av Views. Nedanstående Figur 14 visualiserar de inställningar som används för att skapa en Get Locations karta, då dess uppgift är att samla ihop alla kartors markörer oberoende till vilken innehållstyp de tillhör. Denna Views karta har dock inga filter i användning, men man kunde filtrera noderna enligt taxonomin eller till vilken innehållstyp de tillhör.



Figur 14. Överblick på en View för en Get Locations karta.

Ifall det finns alltför många markörer på ett enda ställe, då alla markörer är så pass tätt intill varandra och man ser direkt inget på kartan på grund av detta, då kommer funktionen vid namn Marker Cluster i bruk. Markörerna samlas ihop i en enda klunga representerad med ett siffervärde över hur många markörer som befinner sig i klungan. Hur Marker Cluster fungerar kan ses från Figur 15. Figuren markerar att det finns tre stycken markörer i klungan. Vid förstorning kommer markörerna fram en och en. När det inte finns behov av en klunga, då kommer dess existens att upphöras.



Figur 15. Get Locations Marker Cluster-funktion.

3.5.1 GPS

Med modulen medföljer en GPS-funktion, varav namnet GPS är en förkortning av ”Global Positioning System”. Syftet är alltså att visa på kartan var användarens nuvarande position befinner sig. Denna funktionalitet fungerade dock inte i den rekommenderade versionen för modulen, därför blev vi tvungna att använda oss av tillägg från utvecklings versionen och egen editering för att få det att fungera. Från utvecklings versionen importerades två hela mappar: getlocations_gps och getlocations_smartip. Tills vidare visar kartorna användarens position med hjälp av en omvänd geokodning, detta kräver dock att användaren tillåter överlåtande av information om sin nuvarande position, som sidan vackert kommer att fråga efter.

För att få problemet med positioneringen åtgärdat och andra utökningar från det som finns tillgängligt krävdes det enstaka editering på vissa filer. Det gjordes omformuleringar och utökningar i följande filer: getlocations.module, getlocations_plugin_style_map.inc, getlocations_fields.js, getlocations_gps.module och getlocations_gps.js. Största delen hör till att importera GPS och Smart IP ändringar från utvecklar versionen till den rekommenderade. Smart IP är en extern modul, men som under utvecklingsversionen för Get Locations har projektet påbörjats för att implementera Smart IP som en egen undermodul.

Design och layout ändringar gjordes i filerna getlocations_fields.css och getlocations_gps.css. Alla färdigt skrivna kodsträngar kommenterades bort från filen getlocations_gps.css, då deras funktioner körs nuförtiden i bakgrunden istället för att man skall trycka på en knapp för att initiera funktionen. Filen getlocations_fields.css ansvarar för att en latitud och en longitud textfält är visuellt synliga då en markör skall utplaceras för nytt innehåll. Allting förknippat med dessa textfält skrivs om vid Kod 12 för att bli osynligt små, förutom en klass som har uppgiften till att skapa ett litet mellanrum mellan kartan och nästa fälttyp. Denna ändring är enbart till för att tillfredsställa det visuella ögat. Latituden och longituden används av själva modulen för att placera ut markören och därför kan man inte radera textfälten bort, men man kan gömma dem.

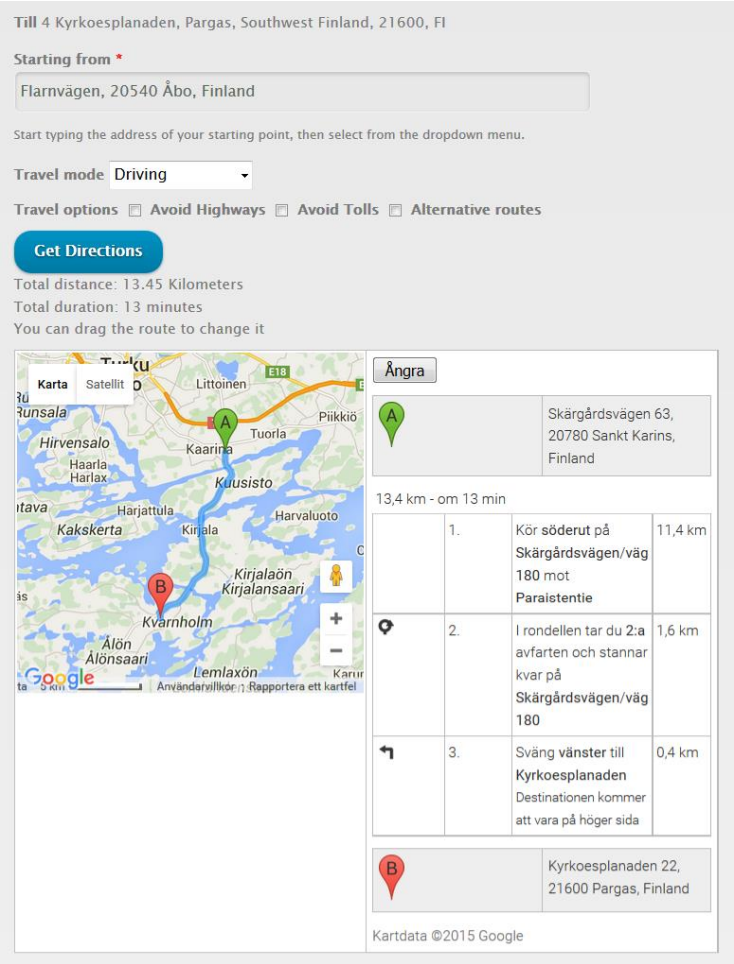
Kod 12. getlocations_fields.css. Visuella ändringar.

```
.getlocations_fields_latlon_wrapper, .getlocations_fields_lat_wrapper
, .getlocations_fields_lon_wrapper, .getlocations_fields_latlon_wrapp
er_themed, .getlocations_fields_latlon_wrapper_themed_nomap, .getloca
tions_fields_lat_wrapper_themed_nomap, .getlocations_fields_lon_wrapp
er_themed_nomap {
  opacity: 0;
  width: 0px;
  height: 0px;
  padding: 0px;
  margin: 0px;
}

.getlocations_fields_latlon_wrapper_themed_nomap{
  margin-bottom: 8px;
}
```

3.5.2 Get Directions

Get Locations modulen stöder bland annat en tilläggsmodul vid namn Get Directions. Med hjälp av modulen kan man få vägbeskrivning till en av de markörer som finns utplacerade på de kartor som listar upp enskilda innehållstyper eller deras underkategorier. Man väljer sin egen startpunkt på liknande sätt som då man skapar nytt innehåll och skall placera ut en ny markör. Därefter behöver användaren bara klicka på ”Get Directions”-knappen och ifall det finns en giltig kör rutt kommer det en beskrivning på hur man skall ta sig till väga, detta kan ses från nedanstående Figur 16. Användaren kan då se hur länge det tar att köra och den totala sträckan. Därtill har användaren även en möjlighet till att ändra på kör rutten genom att dra på kartans inmatade kör förslag. Kartans vägbeskrivningar kommer automatiskt att bli uppdaterade och visar upp ny information till den ändrade rutten. Användaren har även möjlighet till att ändra transportmetoden från att köra med bil till att gå, cykla eller använda sig av kollektivtrafik.



Figur 16. Vägbeskrivning med Get Directions.

3.5.3 Bortgallrade kartor

Före vi bestämde oss att använda Get Locations modulen som applikationens karta och alla funktioner som den skall ha, hade vi gått igenom och bekantat oss med en hel del andra moduler med liknande egenskaper och aspekter. Modulernas namn och deras versionsnummer är presenterade i Bilaga 3.

De moduler som är mest lik Get Locations modulen är Leaflet och OpenLayers. Leaflet och OpenLayers ger användaren möjligheten till att placera ut en karta som en fälttyp, dit användaren kan placera ut markörer samt rita ut sträckor, cirklar och polygoner tack vare dess lagersystem. Leaflet och OpenLayers erbjuder en stor variation på kartor att välja emellan, exempelvis MapQuest, Thunderforest, Yandex och Bing. Tack vare detta är man inte längre bunden till att endast använda kartor från Google. Båda modulerna stöds på Drupal och DrupalGap, medan Get Locations modulen enbart stöds på Drupal. Skillnaderna mellan

modulerna är att Leaflet är en lätthanterbar modul men som tillger användaren relativt få konfigurationsmöjligheter, medan OpenLayers är praktiskt taget det motsatta. Till Leaflet finns det dock tilläggsmoduler och bibliotek som ökar på administrativa inställningar för hur kartan skall fungera. Trots allt detta finns det för Leaftlet modulen inte en möjlighet till att centrera kartan med flera noder eller användare och inga alternativ för att okomplicerat separera noder från taxonomin. Detta resulterade till att Leaflet avkastades. Medan för OpenLayers fanns det strul med att helt enkelt få kartan visuellt synlig då man skulle skapa nytt innehåll. Varken den rekommenderade eller utvecklings versionen gav något hjälp till detta problem.

Förutom stora och kompletta moduler som strävar till att ge alla tänkbara funktioner man kan tänka sig behöva då man handskas med en karta, finns det även mindre varianter som kan antingen köras ensamma eller kombineras för att få fram mera funktioner. Modulen Address Field är en fälttyp och ger en möjlighet till att lägga ut fält för gatuadress, ort, postadress och land för innehåll. Fungerar bra som en tilläggsfunktion med en separat kartmodul, som exempelvis Geofield. Geofield kan behandla informationen som tilldelas från ifyllda Address Field-fält och genomföra geokodning vilket kommer att resultera till en markör på en karta. Modulerna Geolocation och Simple Gmap är relativt liknande som Geofield, då de behöver information för att visa fram en karta. Geolocation har därtill en funktion som möjliggör emottagning av IFrame kartor.

Det finns dock ett antal mindre kartmoduler som inte kräver andra moduler för att vara fungerande. Bland dessa moduler finns Google Map Field och Staticmap. Båda modulerna kan visa fram en Google Maps karta som ett nytt fält. Användaren kan placera ut en markör på valfritt ställe på kartan och ändra karttypen från vanlig till exempelvis satellit vy. Modulen Google Map Field erbjuder dessutom en funktion som tillåter användaren att placera ett standardiserat zoomavstånd för alla besökare som öppnar innehållet och ser på kartan.

För att genomföra geokodning eller omvänd geokodning krävs det att man kan förknippa geografiska positioner med Drupal noder, platser och annat innehåll. Modulen Location tillåter administratorn till att samla in adresser, genomföra geokodning och associera platserna med noder. Vissa kartmoduler kräver att denna modul är färdigt installerad för att allting skall fungera, medan andra moduler kan ha en liknande modul inpaketerat i själva kartmodulen. Exempelvis Get Locations har en undermodul som baserar sig på Location modulen.

Det finns en modul vid namn IPGV&M eller “IP Geolocation Views & Maps som kan visa på en View redan färdigt utplacerade markörer. IPGV&M stöder markördata från bland annat GetLocations, Geofield, Geolocation och Location moduler. Modulen samlar ihop data som till exempel positionen för en eller flera användare och/eller noder, varefter den slutligen sammanfogar informationen till en enda View. IPGV&M kan visa fram markörerna på kartor som Google, Leaflet och OpenLayers. Man kan ställa in bland annat hur markörerna presenteras på kartan, centrerings möjligheter vid användarens position och diverse olika filter.

Då man handskas med DrupalGap är det viktigt att man förstår att man inte kan handskas med vilka moduler som helst, då PHP inte direkt stöds. Modulen GeoJSON ger en möjlighet att visa geografiskt innehåll från Views som JSON data. När applikationer med PhoneGap stöder enbart HTML, CSS och JavaScript uppstår det problem då Views använder sig till en stor del av PHP för att visa fram innehåll som kartor, markörer, positioner och helt enkelt sådant material som kan associeras med geografiskt innehåll. Detta är en lysande modul att installera på Drupal sidan ifall man överväger till att skapa applikationen med hjälp av DrupalGap. Då man söker efter kartmoduler som stöds på DrupalGap, hittar man bland annat Address Field, Geotracker, Geofield och Geofield Gmap. Modulerna kan kombineras för att visa upp en karta med användarens nuvarande position. Med redigering av modulernas kod och View gjord i JSON format, kan man framlägga en karta med flera markörer och en nedanstående förteckning som rangordnar alla noder enligt vilken av dessa som är närmast användarens nuvarande position. Detta skapades till applikationen innan DrupalGap idén avkastades.

3.6 BETYGSÄTTNINGSSYSTEMET

Betygsättning av innehåll på Internet har blivit ett allt vanligare fenomen. Den tar sig många olika skepnader men till de vanligaste tillhör det att man endast ger ett plus då man anser att något är bra. Dock med denna metod går det inte att ge negativ feedback, därför används oftast en pil upp och/eller pil ner eller ett femstjärnsystem.

På Pargasapplikationen vill vi dock ha ett unikt betygsättningssystem som inte finns någon annanstans. Inspirationen har tagits från webbsidan feber.se som använder ett system som ger värme åt artiklar när man betygsätter dem. En Artikel kan exempelvis ha 120° vilket betyder att den är otroligt populär med tanke på dess värmegrad. Utöver att den säger temperaturen är bakgrunden på graderna också antingen blå eller röd, vilket visualiserar temperaturen. Ett

liknande betygsättningssystem skall implementeras till applikationen för Pargas stad. Istället för värme använder vi oss av ett vädersystem, då havet och den grönskande naturen är nära till hands för Pargasskärgård. När man ger plus poäng blir vädret finare där solklart är det bästa, medan åska eller regn är det sämsta vilket en artikel får då den har mycket negativ feedback. Hurdant väder som syns bestäms av medeltalet av rösterna. Enligt beställarna skall en nedröstning (downvote) vara värd -3 poäng och en uppröstning (upvote) +2. Detta gjordes för att beställarna anser att en negativ feedback skall ha mycket starkare påverkan än en positiv. Enligt dem är en missnöjd kund mera benägen till att berätta om upplevelsen till ett stort antal personer, medan en nöjd kund kanske enbart berättar till några få vänner. Poängsättnings-systemet skall reflektera detta.

En nackdel med ett system som ger poäng åt företag är att de som har dåligt betyg troligen inte vill vara med i fortsättningen på applikationen, eftersom det ger dålig publicitet. Det finns flera personer och företag som inte kan ta emot kritik utan att gå i försvar. Att kunna ta konstruktiv kritik är dock en av de viktigaste egenskaperna hos en bra företagare, för om man arbetar vidare och eliminerar orsaken till att man fick dålig kritik blir företaget bara bättre. God feedback är också bra eftersom det höjer moralen och personer blir mer motiverade till att arbeta. Trots allt detta finns det alltid osofistikerade personer som kommer att nedrösta företag och innehåll av hänsynslösa skäl. För att kunna rösta krävs det dock att personen ifråga måste vara inloggad. Inloggningen har utsatts som ett krav eftersom då torde tröskeln vara högre till att använda oacceptabelt beteende.

För att möjliggöra betygsättningssystemet använde vi i ett tidigt skede modulerna Fivestar och Rate. Det visade sig att de saknade den förmåga som krävs för att få ett skräddarsytt betygsättningssystem.

Fivestar är en modul som enbart har möjligheten att visa det vanliga femstjärniga systemet, vilket medföljde att den direkt blev avinstallerad efter vi hade bekantat oss med den. Den andra modulen Rate var mycket lovande, om den inte hade haft stora brister med att spara röstningsresultaten i realtid. Ibland sparades rösterna på Rate utan problem men för att applikationens betygsättningssystem skall vara robust krävs det att den alltid fungerar.

Modulen som slutligen används är Vote Up/Down. Den har samma funktioner som Rate men rösterna blir alltid sparade. Modulen visar som standardläge enbart siffror som resultat då man betygsätter. Varje röst är från början värd +1 eller -1 poäng. För att redigera detta krävs det att

lägga till och editera på kodsträngar i modulen. För att kunna förklaras krävs det vetskap om vilka filer i modulen som skall redigeras, som i det här fallet är tre stycken. *Vud.theme.inc* som är belägen i modulens källkatalog, den andra är *widget.tpl.php* finns i modulens widget/updown katalog och den en tredje och sista är updown.css befinner sig i samma katalog som *widget.tpl.php*.

I vud.theme.inc ändras hur mycket varje röst skall vara värd. Detta görs i Kod 13 genom att redigera siffrorna i följande kodsträngar till 2 där det står 1 och till -3 där det står -1.

Kod 13. vud.theme.inc ändring av röstvärdet

```
if (!$readonly) {
  if ($up_access) {
    $token_up =
drupal_get_token("vote/$type/$entity_id/1/$tag/$widget_theme");
    $variables['link_up'] =
url("vote/$type/$entity_id/1/$tag/$widget_theme/$token_up/nojs");
    $variables['link_class_up'] .= ' use-ajax';
    }
    if ($down_access) {
      $token_down = drupal_get_token("vote/$type/$entity_id/-
1/$tag/$widget_theme");
      $variables['link_down'] = url("vote/$type/$entity_id/-
1/$tag/$widget_theme/$token_down/nojs");
      $variables['link_class_down'] .= ' use-ajax';
    }
}
```

Följande fil som skall redigeras är widget.tpl.php som är huvudfilen för den widget som används inom. Filen bestämmer hur poängen skall synas. Resultatet skall kunna bli representerad med fem olika sorter av väder och utöver detta skall det också finnas ett NULL läge som existerar då det inte finns några röster att räkna. För att kunna räkna medeltalet av rösterna och därefter kunna använda detta på ett effektivt sätt krävs det tre variabler. Två av dem existerar inne i modulen men den tredje måste vi skapa. Variablerna är $unsigned_points, $vote_count och $voteaverage. $unsinged_points säger hur mycket poäng som har kommit in från rösterna. $vote_count är antalet röster och $voteaverage är medeltalet av de två andra variablerna. Summan dividerat med antalet. Följande Kod 14 är ett exempel på hur detta utförs för det bästa och det sämsta resultatet.

Kod 14. widget.tpl.php Visning av vädret efter kalkylation och logisk kontroll

```
<?php
if ($voteaverage >= 1.35) {
            echo "<div class='level5'>";
            echo "<body>  </body>";
            echo "</div>";
      }
.
.
elseif ($voteaverage < 0) {
            echo "<div class='level1'>";
            echo "<body>  </body>";
            echo "</div>";
      }
.
.
?>
```

Kod 14 berättar att om medeltalet av rösterna är över eller lika med 1.35 kommer den att skriva ut div klassen level5, men om medeltalet är under 0 kommer den att skriva ut div klassen level1. Klassen level5 är det bästa resultatet medan level1 är det sämsta som man kan få. Vädrets design kommer att få sitt utseende från den tredje och sista filen updown.css. Bilderna som används av modulen är belägen i en mapp som har namnet weather.

Kod 15. updown.css Vädrets design

```
.level5{
background-image: url("weather/level5.png");
background-repeat: no-repeat;
height: 32px;
width: 32px;
}
.
.
.level1 {
background-image: url("weather/level1.png");
background-repeat: no-repeat;
height: 32px;
width: 32px;
}
```

3.7 SPRÅKBYTE

Pargasstad är en turist ort på sommaren vilket innebär att applikationen kräver ett system för språkbyte. Det skall finnas tre olika språk, vilka är svenska, finska och engelska. Som utvecklare av applikationen har vi till uppgift att skapa en metod för personer att enkelt översätta sina artiklar till de ovannämnda språken, men ifall det finns något innehåll som inte behövs översättas eller av någon annan orsak inte kommer att bli översatt finns det ett språkneutralt läge som innehåll kan ha.

Översättningen av själva CMS Drupal sker genom att man aktiverar kärnmodulen Locale och laddar ned från Drupals hemsidor en språkfil med .po filformatet, varefter filformatet importeras till Drupal antingen manuellt eller genom det grafiska gränssnittet (Falk, 2011, s. 258–259). Översättningen kommer då inte att översätta innehåll, enbart CMS Drupal kommer att bli översatt. Man kan välja mellan en stor variation av olika språk men vi vårt fall valde vi oss av svenska och finska, då engelska språket kommer med som standarsspråk hos Drupal installationen.

Modulen som hjälper till med översättningen av webbsidan är i18n eller ett annat namn för den är Internationalization. Modulen ger möjligheten att översätta menyer, block och taxonomi. Utöver det här bidrar den också med möjligheten att byta språk. För att byte av språk skall vara enkelt rekommenderas det även att man installerar en tilläggsmodul som förbättrar utseendet på språkbytemenyn. Modulen vi använder för detta är Language Switcher Dropdown, men det finns ett stort utbud av moduler med liknande funktion. Modulen förser sidan med en rullgardinsmeny (drop down menu) för att byta webbsidans aktuella språk. Det går också att ställa in webbsidan till att byta standardspråket automatiskt enligt användarens språkpreferenser. Detta sker via administrationsvyn för språkdetektering och urval från språkkategorin som befinner sig i webbplatsens konfigurationssida.

3.8 RÄTTIGHETER OCH ROLLHANTERING

För att en applikation skall fungera och vara robust krävs det att medlemmar inte har alltför mycket rättigheter. Principen ”just enough” implementeras till alla rättighetsnivåer förutom för administratorn. Denna princip handlar om att användarnas rättighetsnivå skall representera hur mycket rättigheter de har tillgång till. Ifall användaren inte är inloggad till applikationen då skall personen ifråga inte ha rättigheter till att verkställa destruktiva arbeten på applikationen.

Användningen av olika rättighetsnivåer skall användas på allting som har ett inloggnings-system där de inloggade kan lägga till innehåll och dylikt. Applikationen för Pargas stad kommer att använda tre olika rättighetsnivåer som kräver inloggning. Dessa nivåer kallas för roller. De tre rollerna är Administrator, Moderator och medlem, på sidan av de här rollerna finns det också den anonyma icke-inloggade sektorn.

- **Administrator** är den roll som har alla rättigheter. Den har ansvarar för att saker och ting på programmeringsnivå fungerar och att nya funktioner kommer till. Detta innebär att en administrator kan göra allting på applikationen som exempelvis att ändra roll-rättigheterna, vilka innehållstyper som finns och vilka fält som används. De utnämner också vilka personer som skall vara moderatorer. Endast de mest pålitliga kan vara en sådan eftersom de kan med sina rättigheter förstöra allting på applikationen.
- **Moderatorns** huvuduppgift är att överse allting som sker på applikationen och ingripa ifall problem på användarnivå uppstår. De kommer att ha rättigheter att blockera användare som bryter mot regler. Moderatorer har också rättigheter till att radera och editera innehåll som är insatta av andra användare. Moderatorerna kommer också att ha tillgång till en medlemslista för att se vilka personer som är registrerade på applikationen.
- **Medlemmarna** är de personer som kommer att använda applikationen och framförallt lägga in nytt innehåll. De har enbart rättigheter att publicera innehåll på sidan som exempelvis profiler, evenemang och nyheter. Till en början var det planerat att denna roll skulle delas in i två delar, företag och vanliga privat personer. Beställaren hade dock önskemål att användarna endast skall ha en användarroll, därmed blev detta utfört på det här sättet.

Tilldelningen av en roll sker direkt när en person registrerar till applikationen. Modulen som sköter att personen som får den korrekta rollen heter Auto Assign Role. Modulen kan också utföra en hel del andra funktioner, exempelvis går det att ställa in att personer som registrerar sig kan välja vilken roll som de vill ha, istället för att det sker automatiskt. Redan i ett tidigt skede bestämde vi oss för att endast använda en användarrollstyp på applikationen. Eftersom alla användare med rollen medlem kommer att ha samma rättigheter måste det avvaktas ifall det missbrukas eller inte.

3.9 BOKMARKERING OCH SPAM-RAPPORTERING

Applikationen skall också ha funktioner som ger användarna möjlighet att rapportera olämpliga kommentarer och bokmarkera intressant innehåll och profiler (Falk, 2011, s. 164–165). För att uppnå funktionerna har vi tagit i bruk modulen Flag.

Bokmarkeringsfunktionen kom med automatiskt då den aktiverades. Bokmarkeringen av innehåll sker genom att användaren trycker på knappen ”Bokmarkera detta” som finns vid nedre kanten av det publicerade innehållet, vilket illustreras vid nedanstående Figur 17. Det bokmärkta innehållet kan användaren se vid sin personliga profil, där de också har möjligheten att radera sitt bokmärke.

Figur 17. Bokmarkering av publicerat innehåll och spam rapportering över kommentarer.

Funktionen för att anmäla ovidkommande kommentarer eller spam använder samma princip som bokmarkeringsfunktionen men till skillnad från att det syns i profilen kommer nu enbart administratörer och moderatorer att kunna se sidan där allt rapporterat innehåll befinner sig.

Denna administrativa sida är gjord med hjälp av modulen Views. Listan visar innehållets titel, kommentarens titel, när och av vilken användare den rapporterades. Användarens unika identifikationsnummer som rapporterat innehållet syns i tabellen för att motverka missbruk av funktionen. Listan kan ses från Figur 18.

Innehållets titel	Kommentarens titel	Tiden för anmälning	Användarens uid
Knäcka upp ägget före kokningen!	Earn money by viisit website!	fredag, oktober 23, 2015 - 11:58	5

Figur 18. Spam-tabellen över ovidkommande kommentarer.

För att listan av spam-rapporterat innehåll inte skall bli överfull har en ”Inte Spam”-funktion satts till som enbart moderatorer och administratörer kan använda. Funktionen möjliggör borttagning av alla spam rapporteringar på en kommentar. Funktionen är programmerad med hjälp av modulen Rules. Rules är en av de kraftigaste modulerna som Drupal har att erbjuda eftersom genom den går det att göra en omfattande mängd tillämpningar som annars skulle kräva mycket programmering. Det som vi har gjort är att när man trycker på knappen ”Inte Spam” initieras en funktion som söker upp alla användare som har rapporterat kommentaren. Funktionen går i ett loop liknande händelseförlopp som kommer att ta bort flaggmarkeringen från var och en av användarna som markerat kommentaren som spam. Detta kommer att upprepas tills alla spamflaggor är borttagna från kommentaren. Rules inställningarna till funktionen illustreras i Figur 19. När väl alla spamflaggor är borttagna blir ”Inte Spam”-knappen gömd. Den kommer dock tillbaka då någon användare har rapporterat en kommentar som spam.

Figur 19. Illustration av ”Inte Spam” Rules inställningar.

3.10 KAMERAFUNKTION

På applikationer som är i hybrid- eller nativformat är processen för att få en kamera att fungera okomplicerat eftersom då har man direkt tillgång till funktionen. Det går dock inte att göra på samma sätt i en för en webbapplikationstyp eftersom den, som redan har nämnts, inte har tillgång till samma utsträckning av mobila den enhetens funktioner. Men med hjälp av HTML 5 är det möjligt att få tillgång till kameran direkt från ett bildfält från en webbsida uppbyggd med hjälp av Drupal. Koden som används kan ses från Kod 16.

Kod 16. kamera.js Kamerafunktion till ett bildfält

```
function(){
$('div.image-widget-data input[type="file"]').each(function(idx,
item) {
  $(item).attr('accept', 'image/*,capture=camera');
}); }
```

Koden kommer att söka efter <input type=”file”> inne i själva bild-widgeten och lägga till ett acceptera (accept) attribut som berättar åt den mobila enheten att den har rättigheter att ladda upp bilder direkt från den interna kameran. (Cafugo, 2013). Vi vill att kodskriptet skall köras på samma gång som webbsidan laddas fram. Det här går att åstadkomma med hjälp av att lägga in kod 17 som en referens i Drupal-temats .info fil.

Kod 17. mobile_responsive_theme.info referens till kamera.js

```
scripts[] = js/kamera.js
```

Kamerafunktionen krävs för att applikationen skall kunna uppfylla de minimikrav som ställs på applikationer som sätts ut på de olika plattformarnas egna butiker. Speciellt då när

webbapplikationer inte är av allmänhet accepterade på applikationsbutiker, men om de har nativa funktioner kan de bli godkända.

3.11 PROGRAMMERING PÅ APPLIKATIONSSIDAN

Fastän största delen av applikationens innehåll finns på webben krävs det ändå några programmeringsingrepp på webbapplikationen för att allting skall fungera. Som redan nämndes används PhoneGap som bas, därför är det då enkelt att skapa en applikation för utvecklings- och testningssyften samt för själva distribueringen av den färdiga applikationen. Utöver detta har man också tillgång till ett brett urval av insticksmoduler.

Insticksmodulen vi främst använder oss av heter Inappbrowser. Inappbrowser gör det lättare för oss att skapa en länk mellan webbsidan och den mobila applikationen. Vanligtvis installerars insticksmoduler med hjälp av att sätta en länk inne i applikationens config.xml fil, men i det här fallet installeras den genom att kopiera in den till applikationens mapp för insticksmoduler. Detta gjordes för att det krävdes några redigeringar i dess kod för att få en knapp som stänger av applikationen att fungera på Android operativsystemet istället som en tillbakaknapp. Det går att djupare bekanta sig med Inappbrowser ändringarna från källan. (Asfgit, 2015).

När insticksmodulen är installerad går det att använda dess funktioner för att få en webb vy av webbsidan. Detta utförs genom att skriva det som finns i Kod 18 in till applikationens index.html, noterat med att webbsidans URL-adress skrivs in istället för 'URL till webbsidan'. Med hjälp av detta kommer webbsidan att laddas fram då användaren öppnar applikationen på sin mobila enhet.

Kod 18. Applikationens index.html

```
<script type="text/javascript" charset="utf-8">
document.addEventListener("deviceready", onDeviceReady, false);
function onDeviceReady() {
var ref = window.open('URL till webbsidan', '_blank', 'location=no');
        ref.addEventListener('loadstart', function(event) { });
        ref.addEventListener('loadstop', function(event) { });
        ref.addEventListener('loaderror', function(event)
{ alert('error: ' + event.message); });
        ref.addEventListener('exit', function(event)
{ alert(event.type); });
    }
</script>
```

4 UTVECKLINGSMÖJLIGHETER

Webbapplikationen för Pargas stad är till för att demonstrera hurdan applikation som kunde skapas med tanke på att den skall fungera som en effektiv informationskälla för alla inom staden. Produkten uppfyller de minimikriterierna för att den överhuvudtaget kunde tänkas vara användbar. En produkt är trots det aldrig fullständigt färdig, då det finns otaliga idéer för utveckling och frågor om hur allting skall skötas då applikationen tas i bruk. Exempelvis kan det påvisas att applikationen behöver ett fjärde tilläggspråk, avsaknaden för ett bildgalleri är stor och/eller kategorierna över olika företag är alltför få/många.

4.1 LOJALITETSPROGRAMMET

Det fanns en tanke på att applikationen skulle innefatta ett lojalitetsprogram som skulle locka användarna till att dagligen besöka applikationen. Lojalitetsprogrammet skulle generera poäng till användarna då de utför varierande uppgifter som exempelvis betygsätter en måltid på en restaurang eller går till en lokal barberare för tionde gången. Med hjälp av poäng kan man rangordna användare och publicera på framsidan de mest framgångsrika användarna och eventuellt tilldela prestationsmedaljer till deras profil. En idé framställdes att även företag kunde förse med belöningar till aktiva användare. Exempelvis kunde det tionde besöket till barberaren vara gratis eller till ett förmånligare pris. Detta är förstås upp till var och en av företagen att bestämma över ifall de tillåter detta.

För att lojalitetsprogrammet skall fungera krävs det att metoden på bland annat hur man registrerar sitt besök till barberaren sker på ett säkert, enkelt och icke-manipulerbart tillvägagångssätt. Till detta var det planerat att använda oss av närfältkommunikationstaggar (NFC-taggar). NFC gör det möjligt till att upprätta en radioförbindelse mellan anordningen och den mobila enheten genom att de nästan vidrör varandra. På detta sätt kunde man registrera besöket och ladda upp informationen till applikationen. Dessa NFC-taggar kostar, vilket betyder att företagen skall ansöka om medlemskap för att vara med i detta lojalitets-program. Medlemmar i lojalitetsprogrammet skall förses med tilläggsfunktioner till deras profil, ökad publicitet och även möjlighet till att presentera sina produkter eller tjänster på applikationen. Med tanke på de tekniska kraven är de flesta smarttelefoner och lästabletter försedda med NFC teknologi. Äldre enheter som inte är försedda med denna teknologi kan inte använda dessa NFC-taggar.

4.2 ÖVERLÅTELSE, UPPRÄTTHÅLLANDE OCH UPPDATERINGAR

Alla filer som är relaterade till webbapplikationen kommer att överlåtas till beställaren. Följande filer är: databasen, webbsidans innehåll och själva applikationen. Databasen kommer att överlåtas i filformatet .sql, vilket exporteras med hjälp av phpMyAdmin. Webbsidans innehåll består av en mapp som går under namnet sites. Mappen kommer att föras över till beställarens webbserver. Applikationen kommer att överföras med hela sin mappstruktur tillsammans med en distributionsfärdig applikationsfilformat för Android, iOS och Windows.

I framtiden kommer det att skapas en användarmanual om hur man använder applikationen. Användarmanualen kommer att skapas tillsammans med Robert Lönnberg som är ansvarig för alla de grafiska elementen. Användarmanualen skall innefatta alla de administrativa verktyg som krävs för upprätthållning och utförandet av uppdateringar på applikationen.

4.3 ÖVERGÅNG TILL NATIV- ELLER HYBRIDAPPLIKATION

Eftersom applikationen är uppbyggd till att fungera som en webbapplikation är alla funktioner insatta på servern, vilket innebär att det inte finns några funktioner på självaste applikationen. Detta limiterar dess interna funktionella kapabilitet enormt eftersom åtkomst saknas till de flesta nativa funktionerna. För att vi skall kunna använda flera av den mobila enhetens egna funktioner krävs det att vi programmerar om den så att den antingen är hybrid eller nativ.

Om applikationen skall fungera som hybrid är det PhoneGap som används och utöver detta kan DrupalGap också användas då den har fått förmågan att utnyttja mer avancerade funktioner i framtiden. Om man dock enbart använder PhoneGap krävs det att hela applikationen byggs upp från grunden, vilket innebär att allt det innehåll som redan finns på Drupal hemsidan går förlorat. Samma gäller det om man gör om allting till nativ-applikation. Den nativa lösningen kräver att applikationen programmeras enskilt till varje plattform som är otroligt klumpigt eftersom applikation ändå kräver en Internetanslutning för att fungera.

Den mest logiska lösningen i framtiden är att övergå till en applikation som använder DrupalGap eftersom det då enbart krävs mindre konfigurationer på serversidan för att få det att fungera. Genom DrupalGap har vi också en möjlighet att använda PhoneGaps insticksmoduler vilket ger oss förmågan att på ett smidigt sätt använda en stor del av de nativa funktionerna hos den mobila enheten. DrupalGap tillåter oss också att använda allt innehåll som redan finns på webbsidan. Inget material går förlorat i processen.

5 SLUTDISKUSSION

Då examensarbetet anförtroddes till oss blev det klart att produkten skulle bli en otroligt omfattande applikation men trots det har projektet varit mycket givande och en lärorik upplevelse för oss. Under utvecklingsprocessen har vi fått nöjet till att bekanta oss med DrupalGap, webbapplikationsprogrammering och Drupals omfattande utbud på ständigt nya moduler, vilket till en stor del varit nya obekanta områden.

I ett tidigt skede på sommaren 2015 påbörjade vi projektet att skapa en applikation för Pargas stad. En hel del arbetstimmar gick åt till DrupalGap utveckling, vilket med tiden förkastades då vårt projekt kräver mer funktionalitet vilket DrupalGap tillsvidare inte kunde tillge då systemet är relativt nytt. Med beställarens tillåtelse övergick vi till att skapa en webb-applikation med Drupal som bakgrundsprogram (backend).

Informationsöverföring, händelseförlopp, problem, avklarade uppgifter och förslag på idéer har skett på ett smidigt sätt trots att vi har arbetat på skilda arbetsplatser. Problem har ständigt dykt upp under projektets gång, men man har löst dem genom att man noggrant går igenom vad och varför problemet uppstått. Vanligtvis kan det vara att lösningen till ett problem är alltför invecklad eller att det skett ett tankefel vilket har varit orsaken till att man inte kommit vidare. Därför har det varit till en stor hjälp att arbeta tillsammans då man inte behöver sitta ensam med sina problem, utan man kan fråga om hjälp och få en annan synvinkel på problemet.

Under själva arbetet använde vi oss av ett mobilt anpassat tema på Drupal, för att se hur innehållet skulle se ut på en mobil enhet med en mindre skärmbredd. Funktionerna lades in till webbsidan efter hand då de blev färdiga. Vi var dock fullständigt medvetna om att när applikationen får sin egentliga design och layout, vilket hade sin start på hösten 2015, blir funktionerna omplacerade. En del av funktionerna stängdes av för att underlätta design-processen, då det preliminära utkastet inte innefattade placeringen av alla funktionerna. Efterhand har enstaka funktioner funnit sina rättmätiga platser, men en stor del är tillsvidare avstängda eller omtumlade av designen och funktionsförhindrade.

Applikationen skall fungera som en grund till att flera städer och inte enbart Pargas stad tar i bruk en tjänst som tillåter företag, privatpersoner och turister på ett underlättat sätt skall kunna kommunicera med varandra. Framförallt skall företag ha möjligheten att förmedla information

effektivt till ett större antal användare genom applikationen. Förmedlingen av information skall förhoppningsvis ge en större kundkrets till företagen. Den nutida situationen vid städer är att personer föredrar att utföra sina ärenden vid större köpcenter än vid lokala företag. Applikationen skall göra lokala företag attraktivare för den potentiella kundkretsen genom att använda olika knep, som bland annat det planerade lojalitetsprogrammet.

Examensarbetets slutprodukt är en lätthanterlig webbapplikation. Applikationen omfattar alla de behövliga funktionerna för att möjliggöra detta, vilka är språkbyte, användarinloggning, skapandet av nytt innehåll och en företagsprofil.

KÄLLFÖRTECKNING

LITTERATUR

Hagberg, P. & Hellström A., 2002. *Kom igång med webbutveckling (Uppl. 2)*. Lund: Studentlitteratur.

Falk J., 2011, *Drupal 7: Börja Här*, Lund: Studentlitteratur

ELEKTRONISKA KÄLLOR

Asfgit, *[Android] add hardware back button support*. Uppdaterad 04.03.2015 https://github.com/apache/cordova-plugin-inappbrowser/pull/86/files [hämtat: 21.09.2015]

Batigolix, Boggs, G., Calebtr & HongPong, *Quick install for developers (command line)*. Uppdaterad 09.12.2014 https://www.drupal.org/documentation/install/developers [hämtat: 15.09.2015]

Brown, M., *Understanding LAMP*. Publicerat 01.12.2005 http://www.serverwatch.com/tutorials/article.php/3567741/Understanding-LAMP.htm [hämtat: 09.09.2015]

Budiu, R., *Mobile: Native Apps, Web Apps, and Hybrid Apps*. Publicerat 14.09.2013 http://www.nngroup.com/articles/mobile-native-apps/ [hämtat: 28.08.2015]

Cafugo, *Add images to Drupal from your mobile device*. Publicerat 26.4.2013 http://cafuego.net/2013/04/26/add-images-drupal-your-mobile-device [hämtat: 01.10.2015]

Drupal, *History*. (u.å.)a https://www.drupal.org/about/history [hämtat: 15.09.2015]

Drupal, *Security*. (u.å.)b https://www.drupal.com/feature/security [hämtat: 15.09.2015]

Korf, M., & Oksman, E., *Native, HTML5, or Hybrid: Understanding Your Mobile Application Development Options*. Uppdaterad april 2015 https://developer.salesforce.com/page/Native,_HTML5,_or_Hybrid:_Understanding_Your_Mobile_Application_Development_Options [hämtat: 09.09.2015]

McCourt, C., *A Guided Tour of Drupal's Settings.php*. Publicerat 21.4.2013 https://www.ostraining.com/blog/drupal/settings-php/ [hämtat: 23.09.2015]

Pargas, *Allmän information om staden*. (u.å.)
http://www.pargas.fi/web/kommuninfo/info_om_staden/sv_SE/info/ [hämtat: 21.09.2015]

PhoneGap, *About the Project*. Uppdaterad 2015
http://phonegap.com/about/ [hämtat: 09.09.2015]

Tyler.frankentstein, *Views Render Array*. Uppdaterat 15.09.2014
http://drupalgap.org/node/219 [hämtat: 07.10.2015]

FIGURFÖRTECKNING

KODFÖRTECKNING

BILAGA 1

TEKNISKA BEGREPP

API = Applikationsprogrammeringsgränssnitt kommer från det engelska ordet Application Programming Interface. Är en detaljerad förteckning om hur applikationer får kommunicera med programvara.

CMS = CMS står för Content Management System och på ren svenska betyder innehållshanteringssystem. Hanterar och publicerar olika slags av informationsrikt innehåll.

Cookies = Är en datafil och som används av webbplatser för att exempelvis upprätthålla besökarens temporära information eller att komma ihåg ett sparat lösenord.

Domännamn = Är en webbsidans namn. Exempelvis adress.com.

FTP = File Transfer Protocol är ett applikationsprotokoll och har sitt användningsområde då man skall föra över en fil från en dator till en annan via Internet.

GUI = Graphical User Interface eller grafiskt användargränssnitt. Underlättar samarbetet mellan dator och människa, genom att framlägga grafiska ikoner istället för rå text.

IFrame = IFrame betyder Inline Frame och handlar om att bädda in ett HTML-dokument till ett annat HTML-dokument. Dess huvudsakliga användningsområde är att infoga innehåll från en annan källa.

Kodsträng = En programmeringsterm över en liten region av kod.

Omvänd geokodning = Vid omvänd geokodning använder man geografiska koordinater för att finna en definition av positionen. Det mest typiska är ortnamn eller postadresser.

Insticksmodul (plugin) = Är en mjukvaru komponent som lägger till en funktion till ett program.

REST-server = Representational State Transfer-server redogör hur man kan vidareförmedla information och tjänster från en maskin till en annan.

Root = En speciell användare med administrationsrättigheter för systemet.

SDK = Software Development Kit är en samling av flera utvecklingsverktyg vilka möjliggör programmeringen av applikationer till ett specifikt programvarupaket eller utvecklingsplattform.

SSH = En förkortning för namnet Secure Shell. Är ett applikationsprotokoll och används för att koppla samman en säker förbindelse från en dator till en annan på Internet.

BILAGA 2

LISTA ÖVER MODULER I ANVÄNDNING

Modul namn	Version	Datum installerad	Kategori
Admin menu	7.x-3.0-rc5	01.05.2015	Administration
Autoassignrole	7.x-1.0-beta4	13.08.2015	Registrering
Captcha	7.x-1.3	02.05.2015	Säkerhet
Ctools	7.x-1.7	02.05.2015	Administrations verktyg
Date	7.x-2.8	02.05.2015	Datum och tid
Entity	7.x-1.6	10.08.2015	Enhets extension
Entityreference	7.x-1.1	13.08.2015	Entity som Field type
Fboauth	7.x-2.0-rc1	10.08.2015	Inloggning
File Entity	7.x-2.0-beta2	18.08.2015	Filhantering
Flag	7.x-3.6	13.08.2015	Flaggning
Flippy	7.x-1.4	08.09.2015	Fram och tillbaka navigation.
Field Collection	7.x-1.0-beta8	12.10.2015	Fältgruppering
Get Directions	7.x-3.2	28.08.2015	Vägbeskrivning
Get Locations	7.x-1.16 & 7.x-2.x-dev	19.08.2015 & 28.08.2015	Karta
i18n	7.x-1.12	02.05.2015	Översättning
jQuery update	7.x-2.5	02.05.2015	jQuery bibliotek uppdatering
Language Switcher Dropdown	7.x-2.5	13.08.2015	Språkhantering
Libraries	7.x-2.2	02.05.2015	Bibliotek
Logintoboggan	7.x-1.5	11.08.2015	Registrering/inloggning
Media	7.x-2.0-beta1	18.08.2015	Multimedia filhantering
Menu per role	7.x-1.x-dev	14.08.2015	Roll administration
Node Subpages	7.x-2.1	09.10.2015	Nodindelning
Organic group	7.x-2.7	13.08.2015	Profil och grupper
Registration	7.x-1.5	15.08.2015	Registrering till evenemang
Rules	7.x-2.9	02.05.2015	Onsite scripting
Smart IP	7.x-2.41	19.08.2015	IP positionering
Token	7.x-1.6	02.05.2015	API för hanteringstjänster
Variable	7.x-2.5	13.08.2015	API för variabler & metadata
Views	7.x-3.11	02.05.2015	Vymanipulation
Vote Up/Down	7.x-1.0-alpha1+3-dev	08.09.2015	Röstning
Votingapi	7.x-2.12	24.08.2015	API för röstning
Wysiwyg	7.x-2.2	02.05.2015	Textbehandlare

BILAGA 3

LISTA ÖVER BORTGALLRADE KARTMODULER

Modul namn	Version
Address Field	7.x-1.1
Geofield	7.x-2.3
Google Map Field	7.x-2.16
GeoJSON	7.x-1.0-beta2
Geolocation	7.x-1.6
IPGV&M	7.x-1.27
Leaflet	7.x-1.1
Location	7.x-3.7
OpenLayers	7.x-3.0-beta3 & 7.x-3.x-dev
Simple Gmap	7.x-1.2
Staticmap	7.x-1.x