Byte av plattform för webbplats från WordPress till Jekyll

Samuel Lönnfors

Examensarbete
Informationsteknik
2020

EXAMENSARBETE	
Arcada	
Utbildningsprogram:	Informationsteknik
Identifikationsnummer:	
Författare:	Samuel Lönnfors
Arbetets namn:	
Handledare (Arcada):	Dennis Biström
Uppdragsgivare:	languagecrashcourses.com

Sammandrag:
I detta arbete beskrivs hur ett plattformbyte för en webbplats för språkinlärning gjordes. Uppdragsgivaren (Language Crash Courses) ville byta plattform för webbplatsen för att den gamla var både ofullständig och svår att underhålla. Syftet med detta arbete är att beskriva ett plattformombyte av en halvfärdig webbplats. Arbetet behandlar hur plattformombytet gjordes, de verktyg som användes samt hur basen för nya webbplatsen skapades. Målet med arbetet är att visa hur plattformbyte av en webbplats kan gå till väga och hur man bygger en bas åt en webbplats så att den kan vidareutvecklas. Detta innehåller själva plattformutbytet, konvertering av data för nya plattformen samt val och konfigurering av webbhotell. Arbetet fokuserar endast på den tekniska utformningen av webbplatsen. Först presenteras webbplatsen i utgångsläget samt de verktyg som använts för att bygga upp den. Därefter presenteras målsättningen och verktygen för implementeringen av den nya webbplatsen. Sedan presenteras konverteringsprojektet där WordPress ersattes med Jekyll och Excel data konverterades till JSON. Till slut gjordes en värdering av alternativen för hosting d.v.s. GitHub pages vs Louhi. Tekniska projektet var framgångsrikt och nu är webbplatsen uppbyggd med Jekyll och för datainsättning används JSON. Angående webbplatsens hosting valde man att fortsätta med Louhi. Enligt uppdragsgivaren kan han nu lägga till mera funktionaliteter på webbplatsen samt fokusera mera på att lägga till nya språk till sidan och attrahera nya användare.

Nyckelord:	Language Crash Courses, Webbplats, Plattformbyte, Konvertering, Jekyll, WordPress, JSON, Excel
Sidantal:	41
Språk:	Svenska
Datum för godkännande:	

DEGREE THESIS	
Arcada	
Degree Programme:	Information Technology
Identification number:	
Author:	Samuel Lönnfors
Title:	
Supervisor (Arcada):	Dennis Biström
Commissioned by:	languagecrashcourses.com

Abstract:
This thesis describes how the change of a platform for a language learning website was conducted. The client (Language Crash Courses) wanted to change the platform of their website because the pre-existing website was both incomplete and difficult to maintain. The aim of this study is to describe the change of platform for an incomplete website. The study discusses how the change of platform was done, the tools that were used and how the base for the new website was created. The goal of this study is to show how a platform change of a website can be conducted and how to create a base for a website for further development. This includes the platform change itself, the conversion of data for the new platform and the selection of a web hosting provider. This study is focusing only on the technical design of the website. First, the pre-existing website is presented including the tools used for creating it. Second, the goal and the tools for the implementation of the new website is presented. Third, the conversion project where WordPress is replaced with Jekyll and Excel data is converted to JSON are also presented. Finally, an evaluation is made of the alternatives for hosting i.e. GitHub pages vs Louhi. The technical project was successful, and the website is currently created with Jekyll and JSON is used for data insertion. Regarding the hosting provider the decision was made to continue with Louhi. According to the client he can now add functionalities to the website and focus more on adding new languages to the website and attract new users.

Keywords:	Language Crash Courses, Website, Platform change, Conversion, Jekyll, WordPress, JSON, Excel
Number of pages:	41
Language:	Swedish
Date of acceptance:	

INNEHÅLL / CONTENTS

Figurer

Bilagor

1 INTRODUKTION

I detta arbete beskrivs hur ett plattformbyte för en webbplats för språkinlärning gjordes. Uppdragsgivaren (Language Crash Courses) ville byta plattform för webbplatsen för att den gamla var både ofullständig och svår att underhålla. Arbetet fokuserar endast på den tekniska utformningen av webbplatsen.

1.1 Bakgrund

Uppdragsgivaren upplever att det är problematiskt att modifiera WordPress webbplatsen. Därutöver upplever uppdragsgivaren att webbplatsen inte är komplett och den har olika fel i dess funktioner. Uppdragsgivaren hade konsulterat ett antal mjukvaruingenjörer och hade kommit till slutsatsen att WordPress inte är en lämplig plattform för webbplatsen. På grund av mängden data och behovet av tung databehandling till bra fungerande html för språkprojektet var det praktiskt taget omöjligt att hålla sig till WordPress enligt uppdragsgivaren. Det fanns inte en möjlighet att skapa kurser och efteråt ändra på html-layouten utan att vara tvungen att skapa om sidorna på nytt. Data för html strukturen var definierad i Excel makron. Om layouten på html sidorna måste ändras, så måste man också ändra på makrona och med samma återskapa allt kursmaterial. Uppdragsgivarens plan är att skapa flertal olika kurser och med WordPress skulle detta ha varit praktiskt taget omöjligt av ovannämnda orsaker.

1.2 Syfte och mål

Syftet med detta arbete är att beskriva ett plattformombyte av en halvfärdig webbplats. Arbetet behandlar hur plattformombytet gjordes, de verktyg som användes samt hur basen för nya webbplatsen skapades.

Målet med arbetet är att visa hur plattformbyte av en webbplats kan gå till väga och hur man bygger en bas åt en webbplats så att den kan vidareutvecklas. Detta innehåller själva

plattformutbytet, konvertering av data för nya plattformen samt val och konfigurering av webbhotell.

1.3 Avgränsing

Mitt uppdrag i detta projekt gäller främst backend delen (av webbplatsen), d.v.s. jag kommer inte att jobba så mycket med webbplatsens stil och design. Detta görs av uppdragsgivaren själv eller en annan part vald av honom.

Uppdragen är att få webbplatsen tekniskt färdig och funktionsduglig. En bas för webbplatsen byggs och som är lätt att vidareutveckla på. Därefter kan uppdragsgivaren öka funktionaliteter och utveckla layouten/designen på webbplatsen.

1.4 Language Crash Courses

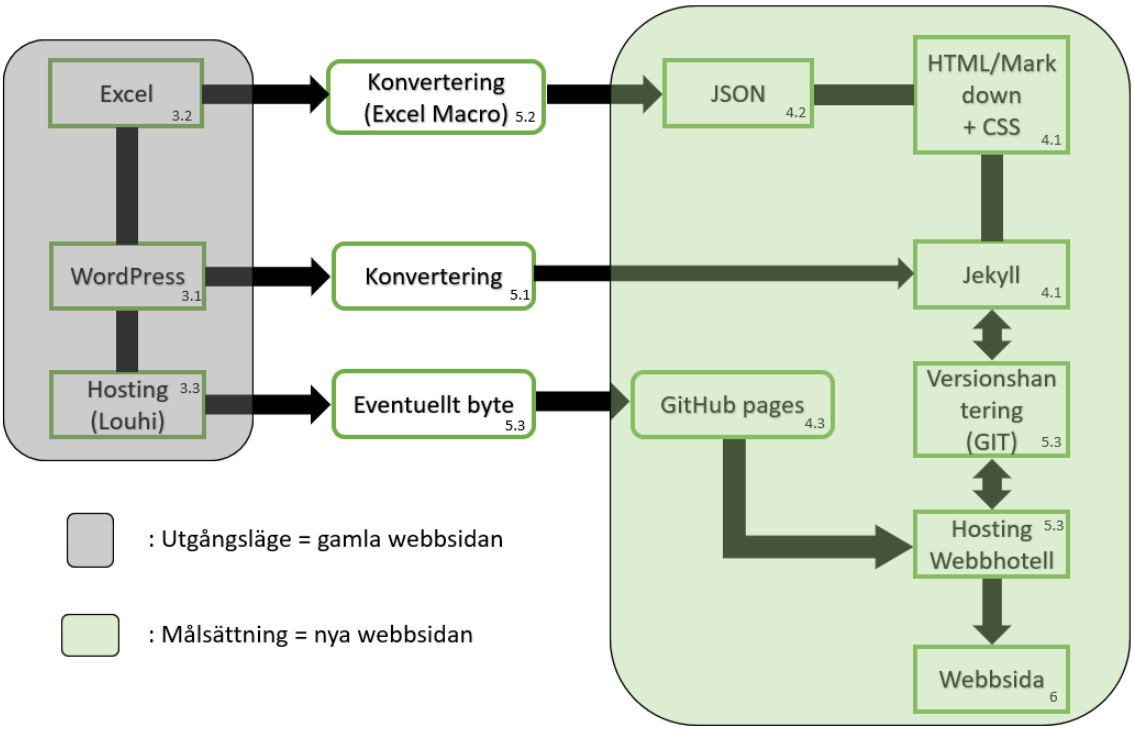
Language Crash Courses är en webbplats för språkinlärnings som presenterar en inlärningsmetod bestående av fem steg. Metoden baserar sig på att användaren läser, lyssnar och upprepar en berättelse som en berättarröst läser upp på sitt modersmål.

- Det första steget går ut på att lyssna på berättelsen och upprepa orden efter modersmålstalaren.
- Det andra steget går ut på att lyssna på berättelsen och samtidigt läsa igenom berättelsen. Man kan välja en långsam eller normal läshastighet.
- Det tredje steget går ut på att översätta en mening utan att se på svaret. Man kan använda ljudinspelningar också. Rätta svaret ser man genom att flytta markören över textraden.
- Det fjärde steget är det samma som det tredje steget förutom att meningarna har bytt plats med varandra.
- I det femte steget lär man sig ordförrådslistor. Det finns cirka 350 ord/fraser. Detta steg går ut på att lyssna på ljudinspelningar och upprepa efter modersmålstalaren.

Istället för att lära sig ett ord åt gången, är idén att man redan i början går igenom och upprepar hela meningar. Tanken har varit att erbjuda kurser gratis utan distraherande reklam. Detta har varit möjligt med hjälp av frivilliga personer som översätter, korrekturläser, skapar ljudinspelningar och utvecklar webbplatsen. (Language Crash Courses 2020)

2 ÖVERGRIPLIG BILD AV ARBETET

Figuren nedan beskriver den övergripliga bilden av arbetet (se fig.1). I bildens ”lådor” anges ett referensnummer för det kapitel i arbetet där arbetsfasen och/eller programvaran beskrivs.



Figur 1. Övergriplig bild av plattformbyte för Language Crash Courses

I kapitel 3 presenteras webbplatsen i utgångsläget samt de verktyg som använts för att bygga upp den. Webbplatsen i utgångsläget var byggd med WordPress och använde Excel makron för att skapa html filer av data. I kapitel 4 presenteras målsättningen och verktygen för implementeringen av den nya webbplatsen. Uppdragsgivaren hade efter egna undersökningar valt att den nya webbplatsen skulle implementeras med Jekyll och använda makron för att konvertera Excel data till JSON-filer. I kapitel 5 presenteras konverteringsprojektet där WordPress ersattes med Jekyll och Excel data konverterades till JSON. Till slut gjordes en värdering av alternativen för hosting d.v.s. GitHub pages vs Louhi.

3 WEBBPLATSEN I UTGÅNGSLÄGET

I detta kapitel beskrivs teknologin som användes i utgångsläget för projektet innan projektet för plattformbyte påbörjades. Först beskrivs allmän teori om verktygen som användes och i sista kapitlet beskrivs hur dessa verktyg hänger ihop med varandra.

3.1 Wordpress

Wordpress är ett innehållshanteringssystem licenserat av GPLv2. Detta betyder att vem som helst kan modifiera på programvaran i WordPress. Innehållshanteringssystemet är ett verktyg som ger möjligheten att modifiera aspekter på din webbplats, fast du inte har färdigheter i programmering. På grund av detta är webbsideutveckling med WordPress tillgänglig för alla. Ungefär 35% av världens webbplatser är uppbyggda med WordPress, vilket betyder att det är världens populäraste innehållshanteringssystem.
Ursprungligen var WordPress primärt använt för att skapa bloggar, men i nuläget används det till alla typer av webbsidor. Nuförtiden kan kärnkoden (core code) modifieras och dessutom finns det många olika teman och instickningsprogram (plug-ins) med vilka man kan ändra på webbplats som är uppbyggt med WordPress. Med hjälp av WordPress kan man bygga många olika typer av webbsidor utöver bloggar och affärssidor. WordPress är också det populäraste sättet att bygga butikssidor för e-handel. (Kinsta 2020)

Exempel på populära webbsidetyper byggda med WordPress är:

- Butiker för E-handel
- Affärssidor
- Bloggar
- CV
- Portfolier
- Sociala nätverk
- Forum

Det finns två olika typer av WordPress: WordPress.com och WordPress.org. WordPress.org är en öppen källa (open-source) version av WordPress. Detta är versionen som vem som helst kan ladda ner och använda hur de vill. Användaren har kontroll över allting men måste köpa ett eget domännamn och webbhotell (web host). Det är alltså ett öppet källkodsprogram som hanteras och utvecklas av WordPress foundation som är en icke vinstdrivande organisation. (Jackson 2020)

WordPress.com är en implementation byggt på WordPress.org där WordPress upprätthåller användarens webbplats, vilket innebär att den underhåller allting för användaren på bekostnad av flexibilitet. WordPress.com är i sin del ett vinstdrivande företag som ägs av Automattic. (Jackson 2020)

3.2 Excel

Excel är en programvara utvecklad av Microsoft och används för att hantera data och automatisera olika funktioner. Med hjälp av Excel kan man samla stora mängder data som sätts in i form av rader och kolumner. Datatypen som sätts in i Excel kan vara i form av bokstäver, siffror, grafer, diagram och bilder. (French 2019)

Vanligen använder organisationer Excel för att organisera data och utföra finansiella analyser. Programmet används alltså i många affärsfunktioner i både små och stora företag. (Corporate finance institute 2020)

Enligt Corporate finance institute (2020) används Excel huvudsakligen till:

- Datahantering
- Datainmatning
- Bokföring
- Finansiell analys
- Programmering
- Kartläggning och diagram
- Tidsplanering
- Ekonomisk modellering

- Diverse data som behöver organiseras

Med Excel kan man spara många kalkylblad i en enda datafil. Datafilen som sparas kallas till en arbetsbok (workbook) och varje sida i arbetsboken är ett separat kalkylblad. I nyare versioner av Excel innehåller varje kalkylblad ungefär en miljon rader och mera än 16,000 kolumner. På grund av detta måste det finnas ett schema, för att hålla reda på var data finns. Horisontella raderna i Excel är identifierade med siffror (1,2,3,4,5, 6…) och vertikala kolumnerna identifieras med hjälp av bokstäver (A, B, C, D, E…). Då det finns mera än 26 kolumner så identifieras varje kolumn med två eller fler bokstäver till exempel: AA, AB, AAB osv. (French 2019)

Vid skärningspunkten mellan kolumnerna och raderna finns en rektangel som kallas för en cell. Cellerna används för att spara data i kalkylbladen. Eftersom varje kalkylblad innehåller miljontals celler har varje cell en egen cellreferens. Referensen är kombinationen av kolumn-bokstaven och radnumret där cellen ligger. Några exempel är A1, D26, EE627. I cellreferensen kommer bokstaven alltid först. (French 2019)

3.3 Louhi hosting

För att få en webbplats att synas på internet måste man använda sig av hosting. Företag inom webhosting erbjuder tjänster som möjliggör publicering av webbsidor eller webbplatser på internet åt individer och organisationer. Tjänsteleverantören upprätthåller en server där all data som hänger ihop med webbplatsen finns. De sköter också om teknologin som ansluter webbplatsen till internet. (Networksolutions 2020)

När internetanvändare vill nå webbplatsen skriver de in adressen i webbläsaren och deras dator ansluter sig till servern och webbsidorna levereras till deras webbläsare. För att nå servern erbjuder webbhotell olika typer av kontrollpaneler åt kunderna, för att hantera sina servrar och underhållstjänster. Kontrollpanelerna innehåller själva webbservern, filhanterare, databaser, mejlserver, domännamnssystem osv. (Website.com 2020)

Louhi är ett finskt företag som erbjuder hosting tjänster i olika prisklasser. För att hantera tjänsterna som Louhi erbjuder använder de användargränssnittet cPanel. CPanel är ett innehållshanteringssystem (CMS) för webbhotellet eller med andra ord en mjukvara som underlättar hanteringen av server-sidan med hjälp av ett grafiskt gränssnitt. (Louhi 2020)

Louhis huvudprodukter är:

- Domännamn och certifikat
- Webbhotell med olika prissatta paket
- Virtuella servrar
- Molnkapacitet och applikationer

3.4 Webbplatsens uppbyggnad i utgångsläget

Data/översättningar som används för varje kurs i Language Crash Courses är lagrade i Excel datafiler. För att få data överfört till webbplatsen användes Excel makron för konversionen. Makron tog in data från Excel filen och skapade html filer enligt ett botten. Alla html referenser och hela strukturen var definierad i makron. Då makron kördes byggde det en html sida, som sedan uppladdades till webbhotellet. För att ladda upp filer användes cPanels egna filhanterare. För att konfigurera och ändra på webbplatsen användes WordPress egen plattform. Servern och WordPress plattformen var länkade ihop så att WordPress tog html filerna som makron hade skapat och kombinerade det med sidan byggt i WordPress. Louhis tjänster användes för webbhotell. Som innehållshanteringssystem användes cPanel som Louhi använder för att komma åt servern.

4 MÅLSÄTTNING OCH VERKTYGEN FÖR DEN NYA WEBBPLATSEN

Detta kapitel behandlar verktygen och teknologin som används för att skapa nya webbplatsen. Först beskrivs verktygen som kommer att användas i projektet. Till sist i detta kapitel behandlas hur dessa verktyg hänger ihop.

4.1 Jekyll

Jekyll är en statisk webbplatsgenerator skriven med programmeringsspråket Ruby. Jekyll används för att bygga statiska webbsidor utgående från dynamiska komponenter såsom liquid kod (liquid code = open-source template language), mallar (templates) och Markdown filer. Jekyll är känt som en ”simpel, bloggmedveten, statisk webbplatsgenerator” för olika personliga, projekt- eller organisationssidor. (Jekyllbootstrap 2013)

Jekyll installeras som en Ruby gem på din lokala dator. En gem är kod som är inkluderad i Ruby programmeringsspråkets bibliotek. RubyGems är en ”package manager” för Ruby programmeringsspråket som erbjuder ett standardformat för distribution av Ruby-program och bibliotek. Det är ett verktyg som hanterar installation av olika gem och distribuering av dem med hjälp av en server. (Jekyllbootstrap 2013)

Efter installation av Jekyll bygger man upp webbplatsen inom Jekylls katalogstruktur. Man kan antingen använda html eller md filnamnsändelse (file extension) då man bygger upp webbplatsen. När man har en webbplats uppbyggd innanför Jekylls katalogstruktur kan man inom datorns terminal kalla på ”*jekyll serve”*. Detta leder till tolkning av Markdown/textfiler, beräkning av taggar (tags), kategorier, permalänkar (permalinks) och sidor från layoutmallar. Kommandot ”*jekyll serve”* levererar din webbplats lokalt. För att se webbplatsen skall man öppna en webbläsare och sätta in adressen som terminalen ger. (Jekyllbootstrap 2013)

Efter tolkning sparar Jekyll resultatet i en katalog som heter ”*_site”*. Avsikten med detta är att man kan servera (serve) allt innehåll i mappen statiskt från en vanlig statisk webbserver. (Jekyllbootstrap 2013)

Skillnaden mellan Jekyll och en dynamisk blogg är att istället för att tolka (parse) innehållet, taggarna etc. efter varje begäran, tolkar Jekyll i förhand innehållet och cachar (cach) hela webbplatsen i en katalog för att senare servera webbplatsen statiskt. Traditionella dynamiska bloggar som till exempel WordPress, kräver en databas och kod på serversidan. Dynamiska bloggar som har mycket trafik måste använda sig av ett caching lager som slutligen utför samma jobb som Jekyll; att servera statiskt innehåll. (Jekyll-bootstrap 2013)

Webbsideutvecklare tycker om Jekyll eftersom man kan skriva innehåll på samma sätt som man skriver kod. Jekyll anses ha följande fördelar för webbutvecklare:

- Det är möjligt att skriva innehåll i antingen Markdown eller text i sin favorittexteditor.
- Man kan skriva och se i förväg innehållet via localhost.
- Man behöver inte anslutning till internet.
- Det är möjligt att publicera via Git.
- Webbplatsen kan ”hostas” på en statisk webbserver.
- Det är möjligt att ”hosta” gratis på GitHub pages.
- Ingen databas behövs.

4.2 JSON

JSON (JavaScript Object Notation) är ett kompakt och lättviktigt textbaserat format som används vid utbyte av data. Det är ett sätt att lagra information i ett organiserat och lättåtkomligt sätt. JSON är fullständigt språkoberoende och använder bekanta konventioner för programmerare i C-familjen av programmeringsspråk, inklusive C, C++, C#, Java, JavaScript, Perl, Python och många andra. Dessa egenskaper gör JSON till ett bra datautbytespråk. (Json 2020)

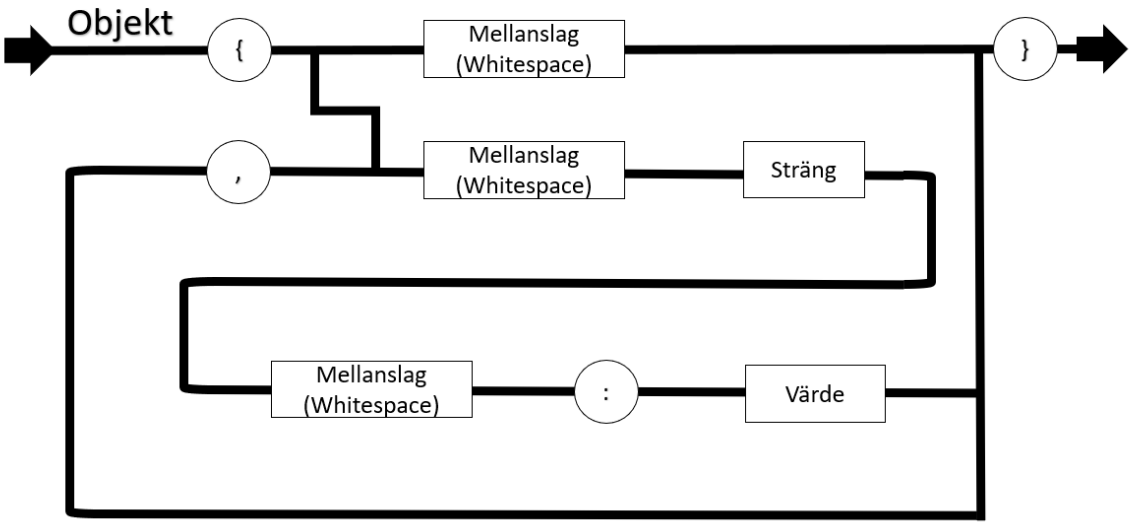
JSON är byggt enligt två strukturer:

1. Namn-värde par (Objekt)
2. En lista av värden (Matris)

Det finns sju olika typer av värden som man kan definiera med JSON: objekt, matris, sträng, nummer, sant, falskt och null. (Javaee 2020)

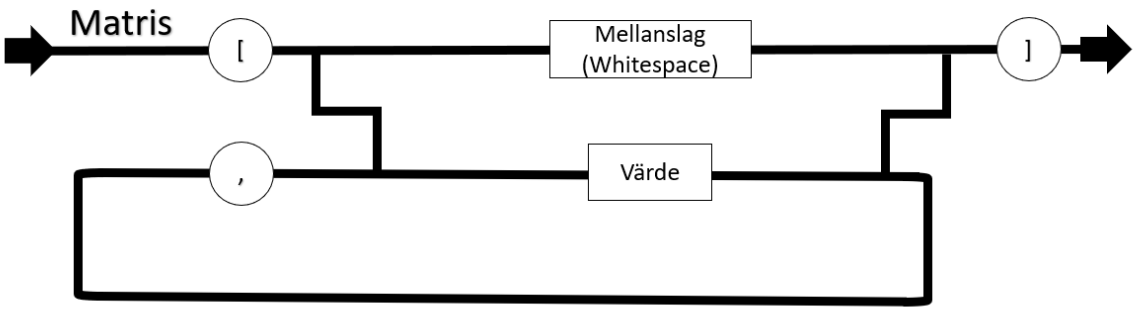
JSON har följande syntax:

- Objekten är inneslutna i klammerparentes ({}), deras namnvärdespar är separerade med ett kommatecken (,), och namnet på värden i ett par separeras med kolon (:). Namnet för objekten är i strängar men värden kan vara någon av de sju värdetyperna, inklusive ett annat objekt eller en matris (se fig. 2). (Javaee 2020)



Figur 2. JSON objektsyntax. Modifierad av skribenten.

- Matriser är inneslutna inom hakparenteser ([]), och deras värden är separerade med ett kommatecken (,). Varje värde i en matris kan vara av olik typ, inklusive en annan matris eller objekt (se fig. 3). (Javaee 2020)



Figur 3. JSON matrissyntax. Modifierad av skribenten

För serialisering och deserialisering av data i applikationer som kommunicerar med varandra över nätet används JSON ofta som format. Applikationer är programmerade med olika programmeringsspråk och de körs i olika applikationsmiljöer. JSON är lämpad för dataöverföring mellan applikationer eftersom det är lätt att läsa och skriva JSON, det är en öppen standard och den är mera kompakt än många andra alternativ. (Javaee 2020)

4.3 GitHub pages

GitHub pages är en statisk hosting tjänst för webbplatser som tar HTML, CSS och JavaScript filer rakt från en förvaringsplats från GitHub, optionellt (optionally) kör filerna genom en byggnadsprocess och publicerar webbplatsen. Man kan ”hosta” sin webbplats med GitHubs domän *github.io* eller en egen domän. (Github 2020)

GitHub pages publicerar alla statiska filer vilka du har i din förvaringsplats (repository). Man kan skapa sina egna statiska filer eller använda en statisk webbplatsgenerator för ett bygga en webbplats. Man kan dessutom skräddarsy sin egen byggprocess lokalt eller på en annan server. GitHub rekommenderar Jekyll som en statisk webbplatsgenerator för att Jekyll har inbyggt stöd för GitHub-sidor och en förenklad byggprocess. GitHub stöder

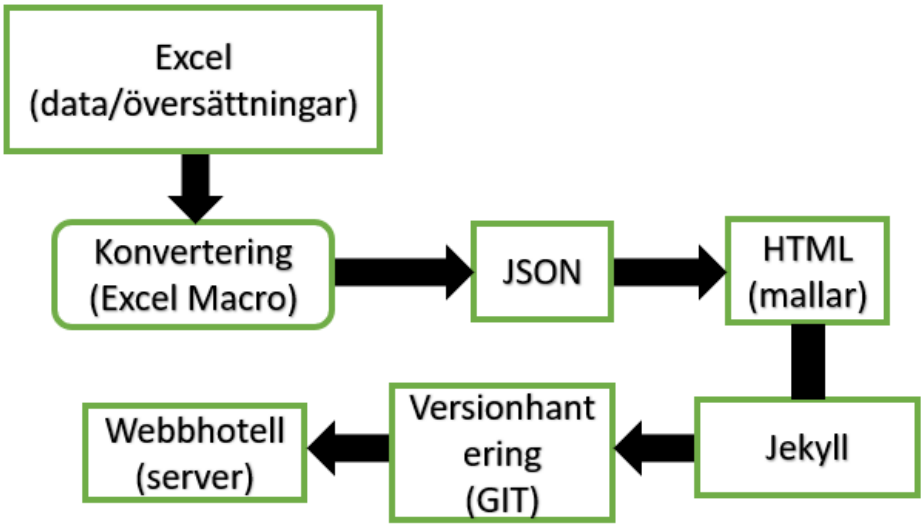
inte serversidigt (server-side) programmeringsspråk som PHP, Ruby eller Python. (Github 2020)

GitHub pages begränsningar:

- Förvaringsplatsen (repository) på GitHub pages har en rekommenderad gräns på 1 GB.
- Publicerade webbplatser på GitHub pages får inte vara större än 1 GB.
- Har en mjuk gräns på 10 byggningar per timme.
- En mjuk bandbreddgräns på 100 GB per månad.

4.4 Uppbyggnad

Målsättningen är att få bytt plattformen från WordPress till Jekyll. Uppdragsgivaren använder Excel för att skapa översättningar till olika kurser. Målet är att få översättningarna insatt i html mall filer. Excel översättningarna blir konverterade till JSON format med hjälp av Excel makron.
För att få insatt JSON data är målet att skapa html mallar som tar in JSON data med hjälp av for-loopar. Sidorna som använder samma mall har likadan struktur men själva innehållet är olikt, eftersom varje JSON fil har olika data. Skapning av nya sidor med html mallarna sker genom att definiera vägen (path) och JSON filnamnet.
Då JSON och html filerna är konfigurerade genererar Jekyll de statiska webbsidorna. Efter att Jekyll har genererat sidorna uppladdas sidorna till servern/webbhotellet med hjälp av Git (se fig. 4).



Figur 4. ”Flödesschema” över uppbyggnad av den nya webbplatsen (Language Crash Courses).

5 KONVERTERINGSPROJEKTET

Ofta då man vill konvertera till exempel webbsidor för användning i en annan plattform eller för att konvertera data från en data typ till en annan, finns det färdiga program som erbjuder konverteringen. I detta kapitel beskriver jag hur jag gjorde konverteringen. Jag kommer inte att djupare behandla installation av program. Många mallar och makron skapades men jag kommer att närmare beskriva endast en mall och ett makro.

5.1 WordPress → Jekyll

Min ursprungliga metod för att genomföra konverteringen var att hitta ett program som skulle ha konverterat alla existerande WordPress sidor till html filer. Det visade sig att av programmet genererade sidor blev fel och dessutom var WordPress sidorna olika sinsemellan. Två sidor som till exempel borde ha samma html struktur var olika. Jag lade även märke till att en stor del av WordPress sidorna var halvfärdiga från utgångsläget. Även om jag lyckats konvertera allting ordentligt, skulle sidorna vara oanvändbara. Därmed bestämde jag att det är lättare att bygga upp hela webbplatsen om på nytt, men samtidigt ta modell av gamla sidorna och använda sig av strukturen som personen som arbetade med webbplatsen innan mig hade byggt.

Nedan beskriver jag kort hur jekyll installationen gick till väga. Enklaste sättet att installera Jekyll är att använda sig av ”RubyInstaller” för Windows. RubyInstaller ät ett fristående Windowsbaserat installationsprogram som innehåller Ruby programmeringsspråket, en omgivning för exekution, viktig dokumentation, med mera. Installeringen gjordes med hjälp av instruktioner från Jekylls hemsida (Jekyllrb 2020) enligt följande:

1. Ladda ner och installera en Ruby+Devkit version från RubyInstaller Downloads nätsidan. Använd standardoptioner för installationen.
2. Kör ridk installtionssteget i sista fasen av installationsguiden. Detta behövs för att installera gems med ursprungliga tillägg (native extensions).

3. Öppna ett nytt kommandotolk fönster från startmenyn så att förändringar i väg-omgivningens (path environment) variabel tas i användning. Installera Jekyll och Bundler med kommandot: *gem install jekyll bundler*.
4. Kontrollera att Jekyll är installerat korrekt med kommandot: jekyll -v.

Installeringen av Jekyll omgivningen enligt instruktionerna gick bra och nu var grunden lagd för vidareutveckling av Language Crash Courses.

Eftersom jag måste bygga webbplatsen från grunden måste jag också fokusera på katalogstrukturen. Katalogstrukturen till steg 1 kapitel 1 för att lära sig holländska är:
dir/english/courses/dutch-english/step-1/chapter-1.
”Dir” är den första katalogen som kommer att innehålla alla html/Markdown filer. I detta kommer man senare att se olika språk. Antalet språk beror på hur många språk man byggt webbplatsen på.
English katalogen innehåller courses katalogen och allmänna sidor som syns, d.v.s. sidorna som syns i menyn på varje sida.
Courses katalogen innehåller kurserna som är tillgängliga för språket. För tillfället innehåller den bara kursen i holländska. I framtiden kommer det att tilläggas till exempel en katalog som heter spanish-english.
Dutch-english katalogen innehåller kataloger för varje steg. Det finns fem steg, därmed finns det kataloger för varje steg. Dessutom innehåller katalogen ”5-steg” html filen som innehåller navigering till stegen.
Sista katalogen step-1 innehåller kapitelsidorna. I första steget kommer det 5 kapitel.
Katalogstrukturen är byggt så att man lätt kan tillägga nya språk och nya kurser till webbplatsen.

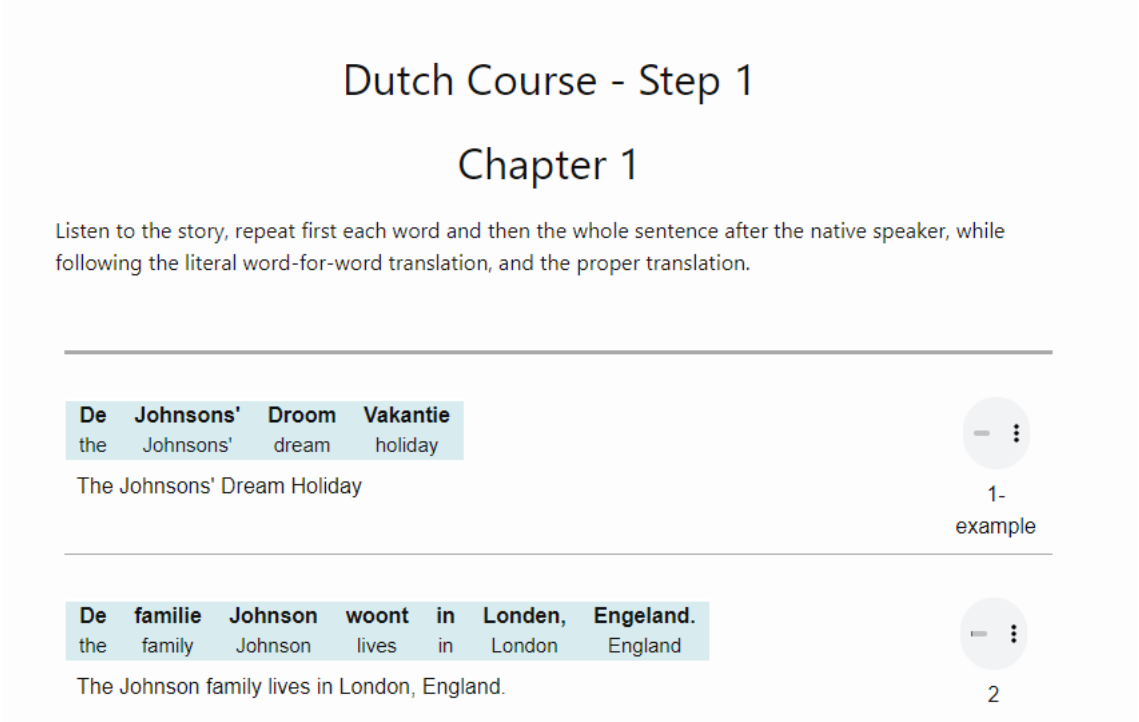
Jekyll har ett omfattande temasystem som man kan utnyttja för att ändra på webbplatsens presentation. Teman definierar layouter, inkluderingar, formatmallar och pluginar. Med hjälp av teman får man en basstruktur lätt implementerat på webbplatsen. Jag valde att använda jekyll minima-temat (Moore 2020) eftersom det verkade minimalistiskt och såg ut att vara enkelt att modifiera. Stilen på stegsidorna definieras i en ”style” tagg i början på sidan. Dessa kan senare flyttas till filen där minima definierar stilen som omfattar hela sidan eller så kan man skapa en Sass fil i data mappen inom jekyll projektet. Eftersom

tyngdpunkten i projektet var att bygga en bas åt webbplatsen fokuserade jag inte mig så mycket på stilen av webbplatsen. Samtidigt modifierade jag dock stilen på stegsidorna ganska mycket.

Målsättningen med projektet var att göra ett botten till webbplatsen som sedan kan utvecklas vidare och modifieras. På grund av detta var fokuset att bygga html mallar för stegsidorna. Uppdragsgivaren hade redan existerande html exempelfiler för varje steg med CSS stil och html struktur. Därmed måste jag utveckla modell filer för varje steg. Till näst kommer jag att beskriva skapandet av steg 1.

Första JSON objektet, i varje JSON fil är tillägnad data som kommer en gång på varje sida. Objektet innehåller titeln på sidan, underkapitel, en instruktionsmening, och vägar (path) för att navigera tillbaka till menyn och till nästa kapitel. För att insätta JSON data för första objektet använde jag en for-loop men i detta fall begränsade jag loopen så att den kördes bara en gång. De tre första värden på objektet syns i figur 5 på sidan 22, men knapparna (länken) för navigering syns inte. Knappen (länken) för att flytta sig tillbaka till menyn är placerad på nedre vänstra hörnet på sidan och knappen (länken) för att komma till nästa kapitel är befinner sig vid slutet av sidan.

Eftersom varje tre meningar i html sidan är delade i egna block (se fig.5) så gällde det att plocka koden från ett block och definiera en for-loop som skapar ett block för varje JSON objekt.



Figur 5. Exempel på steg 1 kapitel 1 (i denna bild ser vi två block)

Till näst behandlar jag koden som definierar varje block. Jag behandlar inte vad varje kodrad gör, utan fokuserar mera på hur värden från JSON sätts in i html filen. Jag kommer att hänvisa till raden, till exempel (rad 12), i koden samtidigt som jag beskriver vad vissa kodsnuttar gör. Radnumret ser man på vänstra sidan av figuren. För att följa med se fig.6.

Första for-loopen (rad 3) definierar att den kör koden innanför loopen en gång för varje JSON objekt. I for-loopen måste man definiera vilken JSON fil man loopar. Eftersom det är en mall html så definieras JSON filen med: include.path och include.file. Dessa variabler blir ifyllda då en sida kallar på mallfilen.

Nästa for-loop (rad 7) definierar att för varje värde i Maintext körs koden innanför loopen. Eftersom värden ligger i en matris måste man definiera positionen på värdet för att få ut data. För att åstadkomma detta har en variabel (var) skapats i början av koden (rad 1) som ökar med 1 efter varje loop (rad 16). Variabeln är insatt inom hakparenteser ([]) där MainText och LiteralTranslation definieras (rad 10, 13). Första loopen tar första ordet

från MainText och första ordet från LiteralTranslation och med nästa loop insätts andra ordet från båda osv. Meningarna som kommer ut från detta syns ovan (se figur 5), de är meningarna som har en blå bakgrund.

Tredje meningen som är under den blåa bakgrunden (se fig. 5) definieras med: object.ProperTranslation (rad 20). Detta behöver inte en egen loop eftersom ProperTranslations värde är en hel mening och meningen är inte delad i skilda ord som MainText tidigare.

Vägen (path) till ljudinspelningarna sätts in med object.AudioPath (rad 30). Denna variabel ger alltså källan till ljudinspelningen. Uppdragsgivaren ville också att ljudinspelningarna är numrerade. För att åstadkomma detta, definierade jag en variabel audioNum i början av koden (rad 2). Denna variabel ökar också med 1 (rad 37) men i detta fall ökar den efter varje JSON objekt.

```
1  {% assign var = 0 %}
2  {% assign audioNum = 1 %}
3  {% for object in {{include.path}}[include.file] offset:1%}
4  <table align="left" width="90%" id="wfw-sentence" border="0" style="max-width:800px !important;">
5      <tr>
6          <td>
7              {% for translations in object.MainText %}
8              <table align="left" id="wfw">
9                  <tr>
10                     <td>  <p class="bodytext"> {{object.MainText[var]}} </p></td>
11                 </tr>
12                 <tr>
13                     <td>  <p class="bodytext"> {{object.LiteralTranslation[var]}} </p></td>
14                 </tr>
15             </table>
16             {% assign var = var | plus:1 %}
17             {% endfor %}
18             <table width="100%" id="pt">
19                 <tr>
20                     <td style="text-align:left"> {{object.ProperTranslation}} </td>
21                 </tr>
22             </table>
23         </td>
24     </tr>
25 </table>
26 <table align="left" width="10%" style="display:inline-block;">
27     <tr>
28         <td height="60">
29             <audio controls style="width:50px">
30                 <source src="{{object.AudioPath}}" type="audio/mpeg">
31                 Your browser does not support the audio tag.
32             </audio> {% if audioNum == 1 %}
33             {{ audioNum }}-example
34             {% else %}
35             {{audioNum}}
36             {%endif%}
37             {% assign audioNum = audioNum | plus:1 %}
38         </td>
39     </tr>
40 </table>
41 {% assign var = 0 %}
42 {% endfor %}
```

Figur 6. Kod för ett block i steg 1

Då man skapar en ny sida inom de fem stegen, kallar man på mallfilen som ligger i *_includes* katalogen inom projektet. Samtidigt som man kallar på mallen så ger man parametrar åt två variabler. Dessa variabler blir insätta i mallen då mall-filen blir kallad.

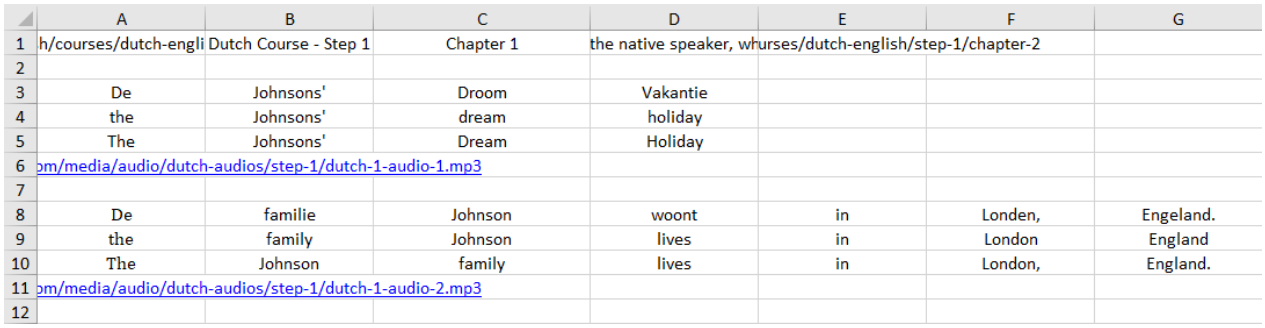
```
{% include steps/step-1.html path= site.data.json.english.dutch-english.step-1
                             file= "chapter-1"
%}
```

Path variabeln skall innehålla vägen till katalogen där JSON filen är lagrad inom projektet. File variabeln skall innehålla namnet på JSON-filen som man vill skapa en sida av. Kodsnutten ovan är alltså det enda man behöver sätta då man skapar en ny sida. Detta förutsätter att man har först skapat en JSON-fil och att strukturen på JSON-filen är rätt.

Detta var ett exempel bara för steg 1. Sammanlagt skapades det 7 olika mallfiler. Alla är lite annorlunda sinsemellan eftersom de har olika funktioner och är byggda på olika sätt.

5.2 Excel → JSON

Kurserna för språkinlärning är alla skapade i Excel. Beroende på vilket steg man skapar är Excelfilerna olika sinsemellan. Från början var strukturen på Excelfilerna byggd på ett sådant sätt att det inte var enkelt att göra makron som tog in data och skapade JSON filer. Första steget var att ändra på strukturen så att man lätt kan programmera makron som går igenom Excel filen. Resultatet blev enligt strukturen nedan (se fig.7).



	A	B	C	D	E	F	G
1	h/courses/dutch-engli	Dutch Course - Step 1	Chapter 1	the native speaker, wh	urses/dutch-english/step-1/chapter-2		
2							
3	De	Johnsons'	Droom	Vakantie			
4	the	Johnsons'	dream	holiday			
5	The	Johnsons'	Dream	Holiday			
6	om/media/audio/dutch-audios/step-1/dutch-1-audio-1.mp3						
7							
8	De	familie	Johnson	woont	in	Londen,	Engeland.
9	the	family	Johnson	lives	in	London	England
10	The	Johnson	family	lives	in	London,	England.
11	om/media/audio/dutch-audios/step-1/dutch-1-audio-2.mp3						
12							

Figur 7. Steg 1 Excelfil

Till näst beskriver jag figur 7 och berättar hur strukturen är uppbyggd och hur data hänger ihop med steg 1. Jag kommer igen att hänvisa till figuren med rader. Radnumret anges på vänstra sidan i figuren.

I första raden definierar man information som inträffar en gång per sida.

- A1 innehåller vägen (path) tillbaka till de fem stegen där det finns menyer där man kan välja steg och kapitel.
- B1 innehåller huvudtiteln på sidan.
- C1 innehåller undertiteln på sidan.
- D1 innehåller instruktionsmeningen.
- E1 innehåller vägen (path) till nästa kapitel.

Sedan kommer resten av data och de är indelade i fyra rader för varje block. Rad 3 innehåller texten som läsaren försöker lära sig. Rad 4 innehåller meningen på språket som läsaren redan vet. Rad 3 och 4 blir insatta i html filen så att varje ord har ett stort mellanrum mellan orden. Detta blir texten som har en blå bakgrund (se fig.5). Rad 5 innehåller hela meningen. Rad 3 och 4 kan man tänka som att de innehåller ordpar medan rad 5 innehåller en hel mening. Rad 6 innehåller vägen (path) till ljudinspelningen.

Målet var att få detta konverterat till ett dugligt JSON format. Eftersom översättningarna redan var i Excel så bestämde jag att använda Excel makron för konverteringen. För att skapa makron använder man Excel VBA. VBA står för ”Visual Basic for Applications”

och det är programmeringsspråket för Excel och andra Office program. För att komma åt Visual Basic måste man först aktivera utvecklare-fliken. Efter detta kan man i utvecklar-fliken välja Visual Basic där man kan skapa VBA projekter, i vilka man kan skapa och modifiera på makron.

För att få Excel data konverterat till JSON måste jag ladda ner och insätta i VBA projektet ett bibliotek (VBA-JSON), som handlar konversionen och tolkning av JSON. Installeringen gjordes enligt bibliotekets dokumentation (Hall, 2019):

1. Ladda ner nyaste versionen av biblioteket från GitHub
2. Importera JsonConverter.bas filen till VBA projektet.
3. Inkludera ordbok referenser/klasser
    - Lägg till referens till ”Microsoft Scripting Runtime”

Med hjälp av biblioteket kan man först konvertera Excel data till en JSON sträng och därefter konvertera det till en JSON fil.

Sedan kommer jag att gå igenom koden för makron som användes för att konvertera steg 1 Excel filen till JSON. Så som tidigare kommer jag att hänvisa till raden i koden samtidigt som jag beskriver vad vissa kodsnuttar gör. Dessutom kommer jag inte att gå igenom vad varje rad gör, utan berättar de viktigaste delarna. Figuren nedan (sidan 29) innehåller koden till makron (se fig. 8).

Första raden som syns på figuren (rad 63) öppnar ett fönster som frågar om man vill köra makron. Om man svarar ja körs makron och om man svarar nej så körs inte makron alls.
Till näst tar man data från första raden i Excel och ger namn åt värden och sparar dem i en ordbok (rad 66–70). Ordboken är en typ av samling som man kan komma åt med en nyckel. Till exempel rad 67 definierar att MainTitle namnet får värde från Cells (1, 2). Första numret i parenteserna skall innehålla radnumret och andra numret skall innehålla kolumnnumret.
Till näst lägger man värden till en JSON samling (Rad 71) och efter det tömmer man JSON ordboken (rad 72). Då man lägger till värden till en samling blir dessa värden ett JSON objekt då data konverteras till JSON senare.

Sedan måste man komma åt resten av data. För att göra det beskriver man en for-loop (rad 74) som börjar på rad 3 och skall köras så länge, tills radnumret kommer till sista raden som innehåller data. Variabeln (i) definierar radnumret.

Härnäst måste man veta hur många kolumner är befolkade (populated). Detta gör man genom att kalla på en funktion som returnerar hur många kolumner innehåller data i raden (i) (rad 76). Mängden kolumner sparas i variabeln colCnt1.

Nästa steg är att få ut orden som är på raden (i). Detta får man genom att kalla på en annan funktion som hämtar alla ord på raden, med hjälp av radnumret (i) och mängden kolumner som är befolkade (populated) (colCnt1). Funktionen returnerar en sträng med alla ord som är på raden i. Orden är separerade med tecknet: (;) och sparas som en sträng i variabeln translations (rad 77).

Nästa steg är att skapa en matris av orden som finns i variabeln translations, detta görs med en splitmetod (rad 78). Sista som man gör är att man sparar matrisen i JSON ordboken med namnet MainText (rad 79).

Rad 81–83 har samma funktion som rad 77–79 men man hämtar orden från följande rad. För att komma till nästa rad så då man kallar funktionen, så ger man variabeln (i) + 1. Med denna rad får man värdet till LiteralTranslation.

Nästa data får man ut med att kalla på en funktion som hämtar varje ord på givna raden men sparar allting i en sträng istället för en matris. Med detta får man värdet till ProperTranslation (rad 86).

Sista data som man skall få ut är vägen för audioinspelningen. Detta får man ut genom att definiera cellen man tar data ut från (rad 88) och spara det till AudioPath.

Sedan skall man spara all data som man har samlat in till en JSON samling och tömma ordboken efter det (rad 89, 90). Detta sparar värden för MainText, LiteralTranslation, ProperTranslation och AudioPath.

För att komma till nästa block som innehåller data, måste man öka på raden (i) med 4 (rad 91).

Eftersom Excels metod som räknar använda rader räknar med tomma rader måste man definiera en if påstående (statement) som hoppar ut ur for-loopen, om nästa två rader är tomma (rad 93–95).

Slutligen hämtar man först namnet på Excel filen som makron körs på. Efter det skapar man JSON filen till destinationen som man har definierat. Och slutligen skrivs den konverterade JSON data till filen (rad 98–101).

```
63      Answer = MsgBox("Do you want to run the Step1 macro?", vbYesNo, "Run Macro")
64
65      If Answer = vbYes Then
66          jsonDictionary("StepsPath") = Cells(1, 1)
67          jsonDictionary("MainTitle") = Cells(1, 2)
68          jsonDictionary("SecondTitle") = Cells(1, 3)
69          jsonDictionary("InstructionSentence") = Cells(1, 4)
70          jsonDictionary("NextPath") = Cells(1, 5)
71          jsonItems.Add jsonDictionary
72          Set jsonDictionary = Nothing
73
74          For i = 3 To ActiveSheet.UsedRange.Rows.Count
75
76              colCnt1 = getColCntRow(i)
77              translations = getMainAndLiteralTranslation(i, colCnt1)
78              arrTranslations = Split(translations, ";")
79              jsonDictionary("MainText") = arrTranslations
80
81              translations = getMainAndLiteralTranslation(i + 1, colCnt1)
82              arrTranslations = Split(translations, ";")
83              jsonDictionary("LiteralTranslation") = arrTranslations
84
85              colCnt3 = getColCntRow(i + 2)
86              jsonDictionary("ProperTranslation") = getProperTranslation(i + 2, colCnt3)
87
88              jsonDictionary("AudioPath") = Cells(i + 3, 1)
89              jsonItems.Add jsonDictionary
90              Set jsonDictionary = Nothing
91              i = i + 4
92
93              If (getColCntRow(i)) = 1 And (getColCntRow(i + 1) = 1) Then
94                  Exit For
95              End If
96          Next i
97
98  fileName = fso.GetBaseName(ActiveWorkbook.Name)
99
100 Set jsonFileExport = jsonFileObject.CreateTextFile("C:\Users\Samuel\Desktop\" + fileName + ".json", True)
101 jsonFileExport.WriteLine (JsonConverter.ConvertToJson(jsonItems, Whitespace:=3))
102 End If
103 End Sub
```

Figur 8. Steg 1 Makro

Resultatet av detta är en JSON-fil som innehåller all data som finns i steg 1 Excel filen. Första JSON objektet innehåller StepsPath (vägen till menyn), MainTitle (huvudtiteln), SecondTitle (undertiteln), InstructionSentence (instruktionsmeningen) och NextPath (vägen till nästa kapitel) (se fig.9). Resten av JSON objekten som förekommer, innehåller data som har samma struktur genom hela sidan (blocken). JSON strukturen är samma men innehållet ändras (se fig.9 rad 10-).


```
[
    {
        "StepsPath": "/dir/english/courses/dutch-english/5-steps",
        "MainTitle": "Dutch Course - Step 1",
        "SecondTitle": "Chapter 1",
        "InstructionSentence": "Listen to the story, repeat first each word and then the whole sentence after t
        "NextPath": "/dir/english/courses/dutch-english/step-1/chapter-2"
    },
    {
        "MainText": [
            "De",
            "Johnsons'",
            "Droom",
            "Vakantie"
        ],
        "LiteralTranslation": [
            "the",
            "Johnsons'",
            "dream",
            "holiday"
        ],
        "ProperTranslation": "The Johnsons' Dream Holiday",
        "AudioPath": "http://www.languagecrashcourses.com/media/audio/dutch-audios/step-1/dutch-1-audio-1.mp3"
    },
    {
        "MainText": [
            "De",
            "familie",
            "Johnson",
            "woont",
            "in",
            "Londen,",
            "Engeland."
        ],
        "LiteralTranslation": [
            "the",
            "family",
            "Johnson",
            "lives",
            "in",
            "London",
            "England"
        ],
        "ProperTranslation": "The Johnson family lives in London, England.",
        "AudioPath": "http://www.languagecrashcourses.com/media/audio/dutch-audios/step-1/dutch-1-audio-2.mp3"
    },
```


Figur 9. Steg 1 JSON

Detta var ett exempel på hur konvertering av steg 1 görs. Sammanlagt skapades 6 makron och dessa används för att skapa JSON filer för kurserna. Varje kurs innehåller 29 kapitel.

5.3 Hosting

Min första tanke angående hosting var att ha den hostad med GitHub pages. Då jag började med projektet rekommenderade många artiklar och bloggar GitHub. Med GitHub pages kan man hosta webbplatsen gratis med GitHubs egen domän github.io eller med en skräddarsydd domän. GitHub pages är dessutom driven med jekyll bakom kulisserna så det är ett bra alternativ för hosting då man bygger upp en webbplats med jekyll. webbplatsen blir automatiskt genererad då man laddar upp (push) källfilerna till GitHub. Så man behöver inte bygga sidan innan man laddar upp (push) filerna till GitHub. Detta kräver dock att man har konfigurerat förvaret (repository) så att GitHub serverar webbplatsen. I början använde jag också GitHub-pages för att hosta webbplatsen, eftersom den var så användarvänlig och GitHub skötte om att webbplatsen byggdes och serverades automatiskt.

Men eftersom GitHub har restriktioner på hur mycket utrymme ett förvar (repository) kan ha, så bestämde jag att vi fortsätter och använda samma värdtjänst (hosting service) som WordPress sidan hade före konversionen (Louhi). Language Crash Courses domännamnet och e-posten är dessutom köpta via Louhi, så jag resonerade att det är bättre at fortsätta med Louhi. Louhis tjänst hanteras genom användargränssnittet cPanel. Med hjälp av cPanel kan man publicera webbplatser, organisera webbfiler, skapa E-postkonton och hantera domäner. Det finns många olika sätt att ladda upp filer till cPanel: genom cPanels egen filhanterare, FTP (File Transfer Protocol) eller med Git. Eftersom jag har mest erfarenhet med Git och man ser historien av varje uppladdning så bestämde jag att använda Git för versionshantering och uppladdning.

Skapandet av Git förvaret inom cPanel gjorde jag med hjälp av cPanels egna guide (cPanel 2018). Att skapa ett Git förvar (repository) i cPanel var inte särskilt komplicerat. Först måste man navigera till Git version control fliken, klicka på skapa knappen, efter det måste man välja om man vill klona ett förvar (repository) eller skapa en ny, ge namnet åt mappen vart förvaret sparas, och till sist ge ett namn åt förvaret. Jag valde att skapa ett nytt förvar och sedan ladda upp projektet som jag hade sparat lokalt på datorn. Efter att jag skapat ett förvar i cPanel, gällde det att öppna kommandotolken och navigera till

projektmappen med kommandot ’cd’. Näst skall man konfigurera lokala förvaret att den uppladdar filerna till cPanel med kommandot:

git remote add origin ssh://användarnamn@languagecrashcourses.com/home/användarnamn/förvarnamn

Sedan måste man ladda upp projektet till cPanel med kommandot:

Git push -u origin master

Nu är lokala projektmappen ansluten till cPanel. För att i fortsättningen för att ladda upp ändringar i projektet skall man använda kommandon:

1. *Git add .*
2. *Git commit -m ”kommentarer kommer här”*
3. *Git push*

Git Add lägger till alla ändringar som man gjort lokalt på alla filer och gör den färdig för att committas. Den berättar alltså vilka filer man vill att skall uppdateras. Det finns många olika kommandon som man kan lägga efter Git add. Till exempel, man kan specificera enstaka filer som man vill att uppdateras.
Git commit kan man tänka att det är en ”snapshot” som fångar projektets tillstånd vid den tidpunkten Git commit körs. Inom citattecken berättar man vilka ändringar man har gjort på projektet.
Git push uppladdar ändringarna från lokala förvaret till fjärrförvaret. I detta fall uppdaterar den Jekyll projektet som ligger i cPanel.

Då man vill uppdatera webbplatsen som finns på cPanel måste man först generera webbplatsen med ett Jekyll kommando. Kommandot är *jekyll build* eller *jekyll b.* Kommandot bygger upp webbplatsen och lägget det i *_site* mappen. Efter att man har laddat upp projektet med Git till cPanel så gäller det att antingen flytta eller kopiera innehållet från *_site* mappen till mappen som cPanel publicerar till nätet. I detta fall heter mappen ”public_html”.

6 SLUTRESULTAT AV PLATFORMUTBYTET

Som ett resultat av projektet har man nu en basstruktur för att vidareutveckla webbplatsen. De verktyg och funktioner som nu skapades bildar en helhet som är en bra språngbräda för att skapa en komplett webbplats.

Data som befinner sig i Excel i dagens läge och i framtiden kan konverteras med de makron som skapades. Sammanlagt skapades 6 makron som omfattar alla fem steg som förekommer på webbplatsen. Makron tar data från Excel filer och konverterar dem till JSON filer. Uppdragsgivarens sätt att skapa nya kurser ändrades inte, utan ett verktyg byggdes för att få data konverterad och insatt till webbplatsen.

JSON filerna blir insatta i de html mallarna som skapades som också omfattar alla fem steg. Sammanlagt skapades 7 olika html mallfiler. Med hjälp av mallfilerna kan man skapa nya sidor med att definiera vilken JSON fil man vill skapa en sida av.

Webbplatsen byggdes upp från början och plattformen byttes från WordPress till Jekyll. Katalogstrukturen blev ombyggd för att lätt tillägga nya sidor och kurser till webbplatsen. Basen för webbplatsen byggdes på engelska. För exempel på sidor se bilaga 2. Webbplatsen innehåller följande sidor:

- Hemsida.
- En sida som berättar om Language Crash Courses.
- Donationssida.
- En ”bli involverad” sida (om man vill bli delaktig i utvecklande av webbplatsen).
- En sida där man kan välja vilken steg och kapitel man vill navigera till med en drop-down meny.
- En kontaktsida där man kan skicka email till Language Crash Courses.
- Sidorna för första kapitlet för steg 1–4 och alla nio kapitelsidor för steg 5.

Ett Git förvar skapades och konfigurerades i webbservern för version kontroll av projektet. Projektet innehåller hela jekyll projektet och webbplatsen som blir genererad av

jekyll. Eftersom Jekyll projektet är belägen på servern och konfigurerad med Git kan många utvecklare samtidigt arbeta med projektet och uppdatera den vid behov.

Webbhotelltjänsten (web hosting service) ändrades inte. Louhis tjänster fortsattes att användas och för att komma åt servern används användargränssnittet cPanel. Servern städades med att radera allting relaterat till WordPress och Git konfigurerades på servern.

7 DISKUSSION OCH SLUTSATSER

Sammanfattningsvis kan man konstatera att projektet var framgångsrikt. Konverteringen och uppbyggnaden av den nya webbplatsen lyckades och uppdragsgivaren var mycket nöjd med resultatet (för uppdragsgivarens feedback se bilaga 1).

Målet för projektet var att få en bas till webbplatsen färdig och funktionell på den nya plattformen samt att flytta nya webbplatsen på ett webbhotell. Efter detta kan uppdragsgivaren lägga till mera funktionaliteter på webbplatsen samt fokusera mera på att lägga till nya språk till sidan och attrahera nya användare.

För ändamålet som webbplatsen uppfyller var Jekyll ett bra alternativ till WordPress. Eftersom Language Crash Courses erbjuder språkinlärning gratis åt besökare och istället för att betala för innehållet tar sidan emot donationer, behöver sidan inte vara dynamisk. Om uppdragsgivaren till exempel ville begränsa innehåll åt gratis användare, då skulle WordPress ha varit ett bättre alternativ. I detta fall skulle man ha implementerat konton åt användare och med hjälp av konton visat innehåll åt användare som har betalat för tjänsten. Ifall uppdragsgivaren vill ha mera inkomst från webbplatsen, kan man implementera reklamer på webbplatsen med Google AdSense.

Man kan lägga till nya funktioner till webbplatsen relativt enkelt. Fast webbplatsen är statisk kan man implementera till exempel: google analytics, google AdSense, SEO (sökmotoroptimering), webbplatsen flerspråkig, donation knapp. Dessutom kan man skapa egna pluginar med vilka man kan utöka Jekylls beteende för att anpassa behovet. Man kan alltså implementera många funktioner till webbplatsen, men man kan inte byta innehållet på sidan dynamiskt.

För att få Excel data insatt till html mallfilerna var JSON ett bra format eftersom det är lätt att läsa och insätta i html mallfilerna. Makrona var ett bra alternativ till konverteringen av data eftersom det är inbyggt i Excel. Makrona som skapades omfattar alla stegen inom webbplatsen och kan användas i framtiden då nya kurser behöver konvertering. Ända bristen med koden i Excel makron är att de konverterar en Excel fil åt gången. Detta borde man ha programmerat så att makron konverterar alla Excel filer inom en katalog.

Louhi som webbhotell var också ett bra val eftersom uppdragsgivaren redan köpt tjänster från denna leverantör. Eftersom valet av webbhotell inte ändrades behövde man inte heller splittra hosting tjänsterna. Nu befinner webbplatsen på Louhi där domännamnet är köpt från. Bristet inom hosting var att jag inte automatiserade uppdateringen av webbplatsen. Nu när man uppladdar filen med Git måste man manuellt kopiera innehållet från _site katalogen till katalogen som publiceras i nätet. Någon form av automatisering borde ha tillämpats så att när man laddar upp projektet till cPanel skulle det automatiskt uppdatera katalogen som publiceras till nätet med en ny version.

Mest svårigheter hade jag med att få Excel konverterat till JSON som sedan sätts in i html mall filer. Det tog länge på att hitta på rätta for-loopen för att få JSON data att skrivas rätt till html mall filen. Dessutom måste man samtidigt hitta på en bra JSON struktur som passar ihop med mall filen. Det tog länge att komma till slutresultatet med att pröva på implementationer som ofta visade data på ett fel sätt. Efter att jag fick JSON strukturen och fungera med mallen började jag sedan och programmera makron. Första makron tog länge att programmera men makron efter det var inte väldigt svåra. Detta var arbetssättet som jag använde för alla html mall filer och makron. Först tog jag kodsnutten från färdiga html filen som definierar ett block, skapa for-loopen till html mallen runt blocket, sedan skapade jag JSON strukturen och pröva att data från html filen syns rätt och efter det skapa ett makro som konverterar Excel filen till JSON formatet som fungerar.

Webbplatsen har varit publicerad på nätet under hela utvecklingen av webbplatsen. Orsaken till detta är att jag kunde under utvecklingen av webbplatsen publicera uppdateringar av webbplatsen och fråga feedback av uppdragsgivaren. Med feedbacken kunde jag sedan göra ändringar enligt önskan.

Syftet med detta arbete är att beskriva ett plattformombyte av en halvfärdig webbplats. Arbetet behandlar hur plattformombytet gjordes, de verktyg som användes samt hur basen för nya webbplatsen skapades. Målet med arbetet var att visa hur plattformbyte av en webbplats kan gå till väga och hur man bygger en bas åt en webbplats så att den kan vidareutvecklas. Detta innehåller själva plattformutbytet, konvertering av data för nya plattformen samt val och konfigurering av webbhotell.

Jag anser att både syftet och målsättningen för detta arbete har uppfyllts. Arbetet beskriver de använda verktygen och programvarorna på ett omfattande sätt. Konverteringen från Excel till JSON beskrivs och implementeringen av den nya webbplatsen med hjälp av Jekyll. Ytterligare evalueras hosting alternativen och resultatet redogörs i detta arbete.

Alla ovannämnda faser i projektet gick bra och jag har strävat till att på ett logiskt och strukturerat sätt redogöra för utgångsläget, använda verktyg och programvaror, själva implementationsprojektet och dess resultat.

8 FORTSATT UTVECKLING AV LANGUAGE CRASH COURSES

Eftersom jag fick html mallarna gjorda för webbplatsen så gäller det till näst att skapa mera sidor med mallarna. Nu finns det bara ett kapitel för varje steg förutom steg 5 där alla kapitel är färdiga. Därmed är det relevant att generera sidor för resten av stegen och efter det skapa nya kurser.

Dessutom rekommenderas det att göra mera mallar åt exempel till 5 steps sidan som innehåller bilder med rullgardinsmenyn (drop-down menu). Varje kurs har alltså en egen rullgardinsmeny, i vilken man väljer steget och kapitlen som man vill läsa.

Front-enden på sidan är för tillfället ganska simpel och har inte en unik design. Så man borde också utveckla på designen på sidan och göra den mera unik.

Uppdragsgivaren har också en önskan att det i framtiden skulle finnas en donations möjlighet. Detta skulle säkert implementeras med att göra ett paypal konto med företagets namn, efter det skulle man lägga en knapp på webbplatsen som för en vidare till paypals sida för att göra en donation åt Language Crash Courses. För tillfället finns det en donationssida med text men en knapp för donationer fattas.

En av den viktigaste framtidsplanen är att göra sidan flerspråkig, så att besökaren skulle få webbplatsen att visa texten på sitt eget modersmål. Då skulle innehållet och kurserna ändras enligt språket.

Någon typ av analytik om besökare borde implementeras också. Detta skulle visa olik info om besökare, till exempel hur många besökare webbplatsen har per dag, från vilket land besökarna kommer ifrån, vilket innehåll är mest populärt osv. Med data från analytiken kan man sedan optimera webbplatsen så att den blir mera användarvänlig.

För tillfället måste man manuellt kopiera _site mappen till public_html mappen då man vill uppdatera webbplatsen på servern. Någon slags funktion som skulle automatisera uppdateringen av webbplatsen då man laddar upp projektet skulle vara bra att implementera.

Slutligen kommer jag att tänka på är att optimera sidan med SEO (sökmotoroptimering).

KÄLLOR

Corporate finance institute. 2020, *Excel definition.* Tillgänglig: https://corporatefinance-institute.com/resources/excel/study/excel-definition-overview/
Hämtad: 2.10.2019

cPanel. 2018, *Guide to Git – Host Git repositories on a cPanel account*. Tillgänglig: https://documentation.cpanel.net/display/CKB/Guide+to+Git+-+Host+Git+Repositories+on+a+cPanel+Account#6708b05bf99d4c25aef6ef270d0bc5eb
Hämtad 18.11.2019.

French, Ted. 2019, *What is Microsoft Excel*. Tillgänglig: https://www.lifewire.com/what-is-microsoft-excel-3573533
Hämtad: 2.10.2019.

Github. 2020, *About GitHub pages.* Tillgänglig: https://help.github.com/en/articles/about-github-pages
Hämtad 15.10.2019.

Hall, Tim. 2019, *VBA-JSON*. [GitHub repository] Tillgänglig: https://github.com/VBA-tools/VBA-JSON
Hämtad: 17.11.2019.

Jackson, Brian. 2020, *WordPress.com vs WordPress.org.* Tillgänglig: https://kinsta.com/blog/wordpress-com-vs-wordpress-org/
Hämtad: 8.10.2019.

Javaee. 2020, *Introduction to JSON*. Tillgänglig: https://javaee.github.io/tutorial/jsonp001.html
Hämtad: 20.9.2019.

Jekyllbootstrap. 2013, *Jekyll intrduction*. Tillgänglig: http://jekyllbootstrap.com/lessons/jekyll-introduction.html
Hämtad: 6.11.2019.

Jekyllrb. 2020, *Jekyll on windows*. Tillgänglig: https://jekyllrb.com/docs/installation/windows/
Hämtad 5.10.2019.

Json. 2020, *Introducing JSON*. Tillgänglig: https://www.json.org
Hämtad: 18.9.2019.

Kinsta. 2020, *What is WordPress?* Tillgänglig: https://kinsta.com/knowledgebase/what-is-wordpress/

Hämtad: 8.10.2019.

Language Crash Courses, 2020. Tillgänglig: https://www.languagecrashcourses.com/
Hämtad 17.11.2019.

Louhi. 2020. Tillgänglig: https://www.louhi.fi/en/
Hämtad: 12.10.2019

Moore, Parker. 2020, *Minima*. [GitHub repository] Tillgänglig: https://github.com/jekyll/minima
Hämtad: 17.11.2019.

Networksolutions. 2020, *What is website hosting?* Tillgänglig: http://www.networksolutions.com/education/what-is-web-site-hosting/
Hämtad 14.10.2019.

Shopify. 2020, *Liquid*. Tillgänglig: https://shopify.github.io/liquid/
Hämtad 12.11.2019.

Website.com. 2020, *What is web hosting?* Tillgänglig: https://www.website.com/beginnerguide/webhosting/6/1/what-is-web-hosting?.ws
Hämtad: 13.10.2019.

BILAGOR

Bilaga 1. *Feedback av uppdragsgivaren*.

- Your overall thoughts on changing from WordPress to Jekyll?

The change from WordPress to Jekyll has been a major breakthrough, without which the project would probably not have succeeded at all. A number of software engineers had noted that WP just wasn't going to do it for me. The amount of data and the need for heavy data processing into good working html for my language project made it virtually impossible to stick with WordPress. There just wasn't a way to create courses and then later modify the html layout without having to redo all the existing pages for each course. As far as I understand the data of the html was stored in the Excel macros. So, if the layout of the html had to be changed, then also the macros had to be changed and all the course materials be recreated all over again. Since I am planning to create hundreds of different courses, it would have led to a dead end. WP also had the disadvantage that we faced constant problems with WP updates and plugin updates that were often not matching, causing lots of technical problems.

- Do you think Jekyll a better alternative to WordPress?

Jekyll is definitely a better alternative to WordPress in that the data of the many language texts, pictures and audio file paths are stored in json files. They function in a way as a data base. The html templates can be modified at any time without having to change the json files that work with them. This gives us the freedom to improve and further develop the design of the site with ease.

- What you think about the Excel Macros?

At first Samuel tried to find suitable software to convert the Excel files to json, but nothing seemed to work well, until he discovered the option to insert some code into Excel macros which would do the job. I'm very happy that he was able to come up with this option and made it to work well for my project. He taught me how to do the Excel to json conversion myself and helped me solve some problems due to the fact that the conversion doesn't work on my Mac but only in Windows. We've had to deal with a lot of technical issues since the structures of the language pages and the different languages and language sites are quite complex. Whatever problems we have faced, Samuel has always been able to come up with solutions. I very much admire that in him. He never gives up and makes everything to work no matter how hard it is.

- Do you think this is a good base for the website which you can continue building on?

The software Samuel has developed is a very good base to continue building on. He has streamlined the process to such an extent that I can basically continue creating courses and further develop the site on my own. I am not experienced with computers and don't understand much at all about coding and web development. Samuel has also taught me a lot of the basics of how these web things work, both front end and back end.

He also recreated the existing html templates and made them to work with the json files through Jekyll. Her did quite a lot of the modification work to the overall design and outlook of the pages as well. They have improved a lot.

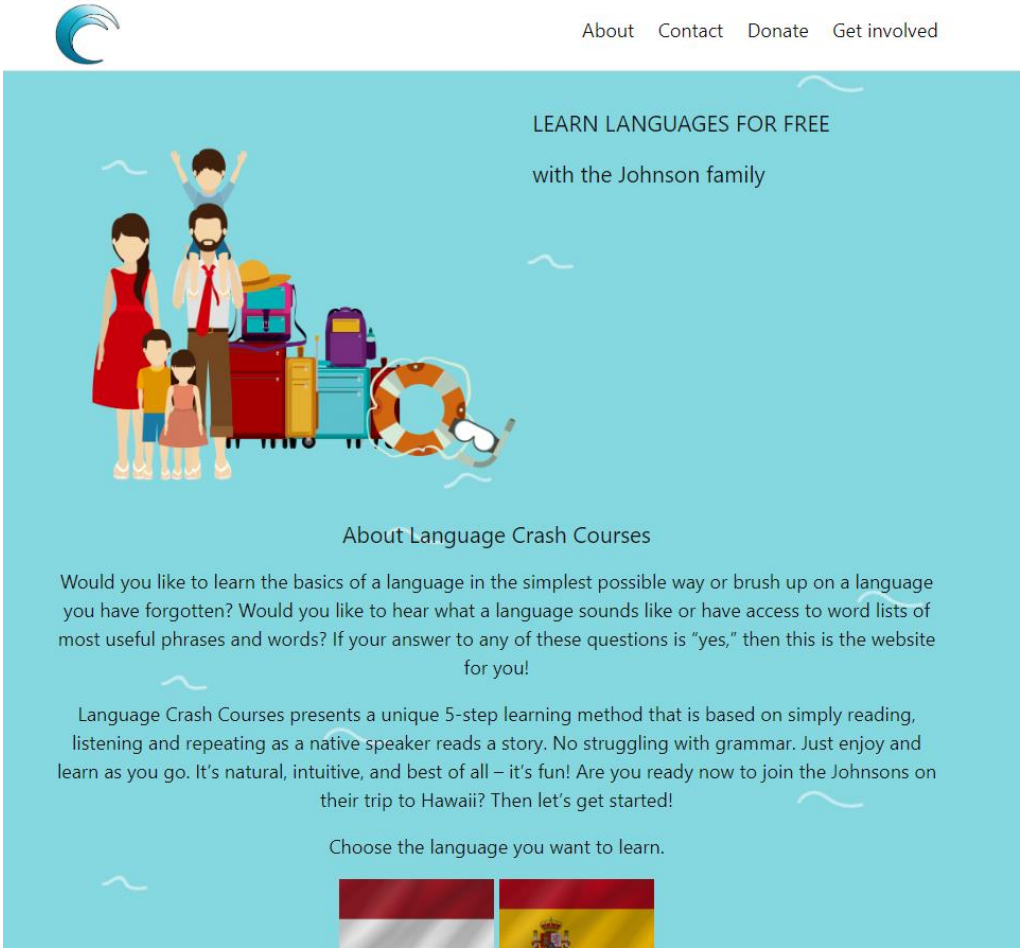
He configured Git so that it will be easy to upload any pages and implement any changes to the server. Samuel rearranged the folder structure on the server and gave it a much more logical and clear structure. He reduced the disk space usage on the server from 88% to only 1.5% which will give me plenty of space to add many more pages, pictures and audio files. He did this by deleting many unnecessary files and folders, including the WP related data files.

- Is it easy to create new pages for the courses?

Samuel will still run me through the whole process of creating the courses from beginning to end, once the json files for all the html templates have been created. He has helped me to install and use the right kind of software to do this on my own (Jekyll, Terminal, Git, etc.). He already explained to me the process and it seems to me quite easy to learn. Once everything is in place, it will be just a routine to go through that I can also teach others to do, when the workload of creating dozens of courses with many language combinations increases over time.

Samuel's contribution to the project has been of major importance and it is my hope that within a few months we will be able to launch a good looking and well working site with two or three language courses to start with. The success of the site will very much be determined by the many hours Samuel has put in to create the structures and processes that allow the site to grow fast without too many complications.

Bilaga 2. *Bilder av webbplatsen (Hemsida, Meny för att välja steg och kapitel, Steg 1 kapitel 1, Steg 2 kapitel 1, Steg 3 kapitel 1, Steg 4 kapitel 1 och Steg 5 Personal Pronouns)*



You can use our 5 steps to learn any way you like, but here is the order we recommend:

Listen to the story and repeat.

Chapter 1
Chapter 2
Chapter 3
Chapter 4
Chapter 5

Listen to the story while reading along.

Dutch Course - Step 1

Chapter 1

Listen to the story, repeat first each word and then the whole sentence after the native speaker, while following the literal word-for-word translation, and the proper translation.

De	**Johnsons'**	**Droom**	**Vakantie**
the	Johnsons'	dream	holiday

The Johnsons' Dream Holiday

1-example

De	**familie**	**Johnson**	**woont**	**in**	**Londen,**	**Engeland.**
the	family	Johnson	lives	in	London	England

The Johnson family lives in London, England.

2

Dutch Course - Step 2

Chapter 1

Listen to the story while reading along. You can choose slow or normal reading speeds.

Slow speed

0:00 / 4:14

Normal speed

0:00 / 2:21

De	Johnsons'	Droom	Vakantie
the	Johnsons'	dream	holiday

De	familie	Johnson	woont	in	Londen,	Engeland.	Peter	Johnson	en
the	family	Johnson	lives	in	London	England	Peter	Johnson	and

zijn	vrouw	Helen	hebben	drie	kinderen.	Hun	dochter	Mary	is
his	wife	Helen	have	three	children	their	daughter	Mary	is

Dutch Course - Step 3

Chapter 1

Further improve you knowledge with this exercise. Translate each sentence of the story into English before looking at the correct translation by hovering the cursor over the text line. You can also listen to the audio recordings if you like.

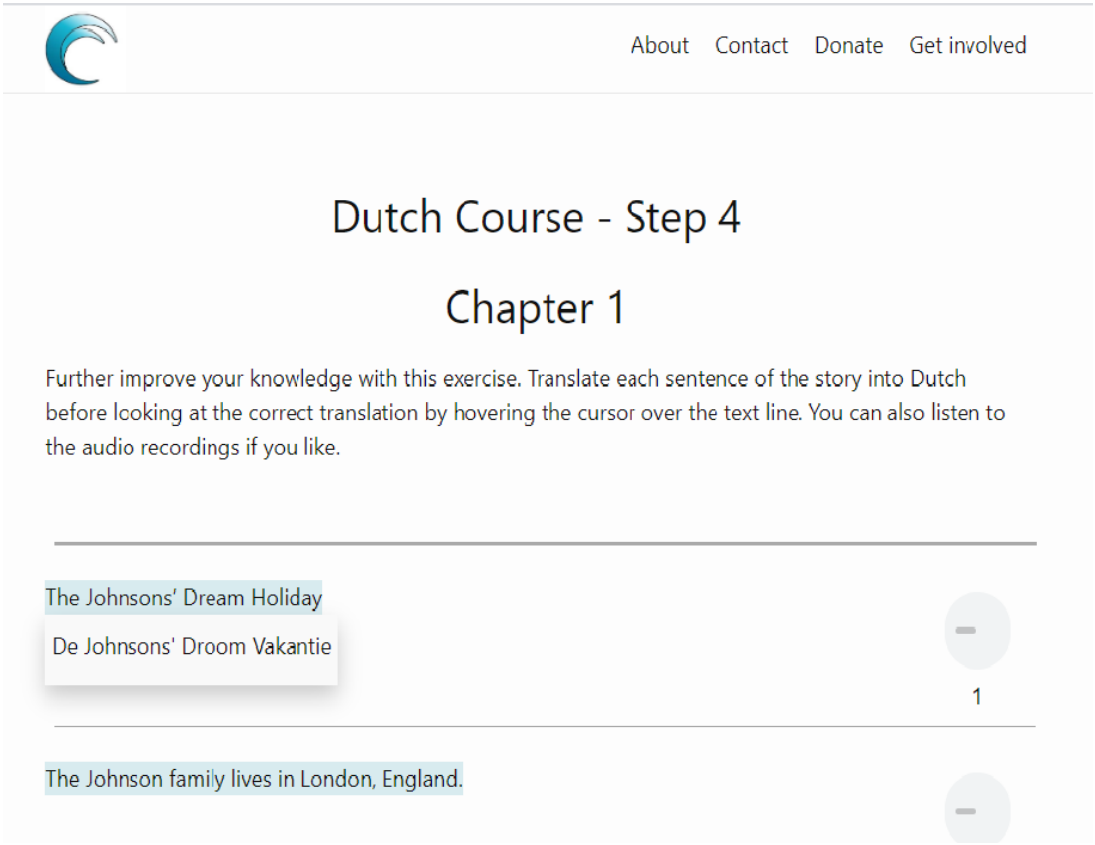
De Johnsons' Droom Vakantie

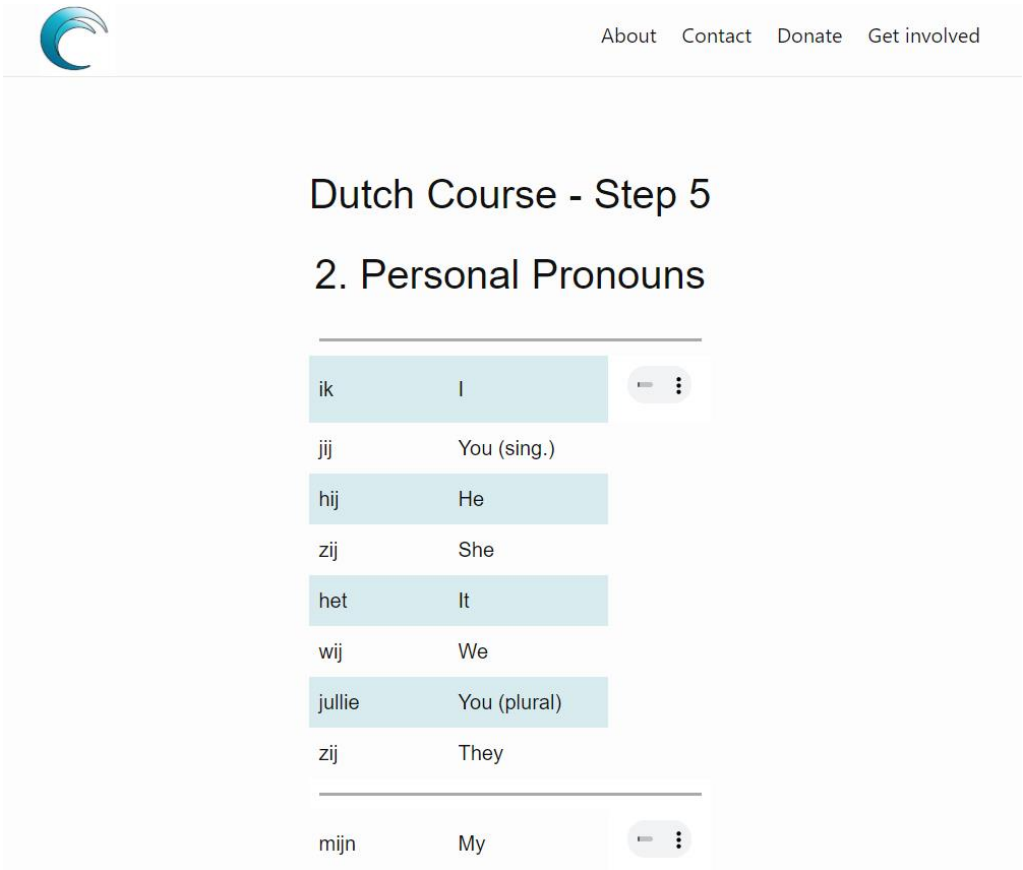
The Johnsons' Dream Holiday

1

De familie Johnson woont in Londen, Engeland.

2


About
Contact
Donate
Get involved
Dutch Course - Step 4
Chapter 1
Further improve your knowledge with this exercise. Translate each sentence of the story into Dutch before looking at the correct translation by hovering the cursor over the text line. You can also listen to the audio recordings if you like.
The Johnsons' Dream Holiday
De Johnsons' Droom Vakantie
1
The Johnson family lives in London, England.


About
Contact
Donate
Get involved
Dutch Course - Step 5
2. Personal Pronouns
ik I
jij You (sing.)
hij He
zij She
het It
wij We
jullie You (plural)
zij They
mijn My