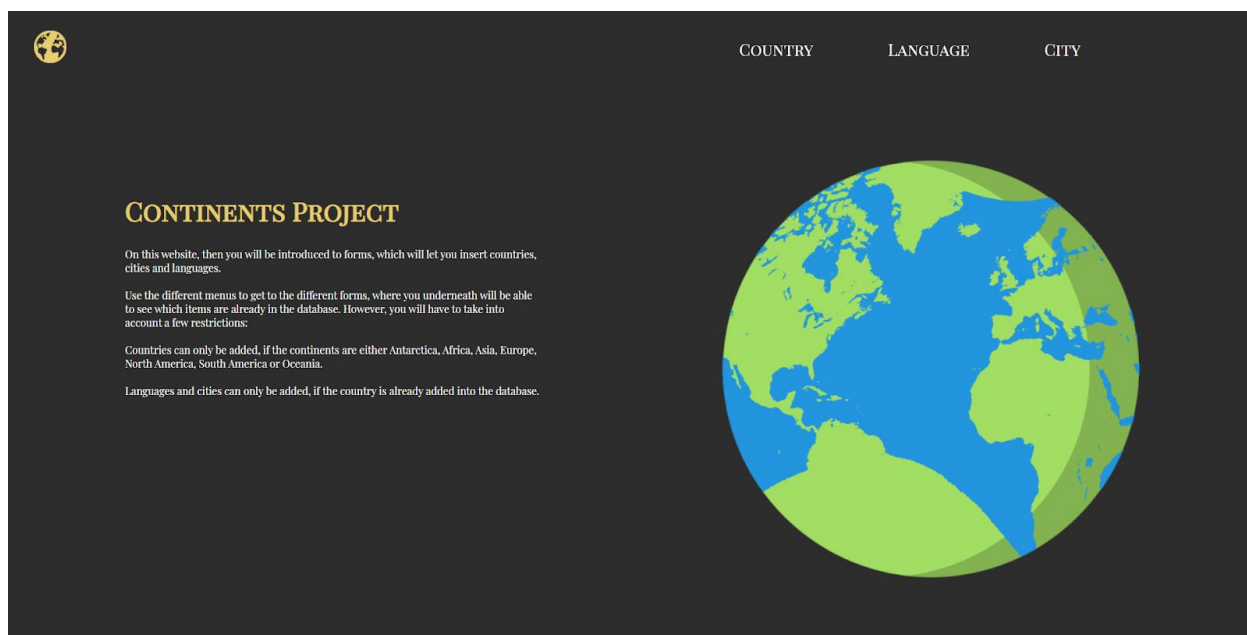


Kontinent Projekt

Uge 8



5,8 sider (14.048 tegn)
á 2400 tegn med mellemrum

Af Christian Kjer Bang, Christian Krag
og Simon Kalmar Riis Mortensen

	1
Indledning	3
Problemformulering	3
Metodeovervejelser	3
Research	4
Dataindsamling	4
Kode Research	5
Analyse	5
Medium Fidelity Prototype	5
Sidens opbygning og design	5
Konstruktion	6
HTML	6
CSS	6
JavaScript	7
Node.js	7
MongoDB	8
Evaluering af proces	9
Kodemæssig evaluering	9
Projektmæssigt evaluering	11
Konklusion	12
Literaturliste	13
Bilag	14

Indledning

I dette projekt fik vi til opgave at lave en hjemmeside baseret på Node.js over forskellige kontinenter og deres lande, byer og sprog. På denne hjemmeside skulle det være muligt at indsætte disse lande, byer og sprog ind i en MongoDB database, dog skulle man også kunne finde hente disse informationer ind på sin hjemmeside.

En del af opgaven indeholdte også at der skulle gøres brug af forskellige restriktioner, når det var at man skulle indsætte disse lande ind i databasen. Disse restriktioner handlede om, at man kun skulle have mulighed for at indsætte et land, hvis et kontinent eksisterede inden for det. Endvidere, så skulle byer og sprog kun have mulighed for at blive indsamlet, hvis landet allerede eksisterede i en collection på databasen. På denne måde, så var alle collections afhængige af kontinenterne på en måde.

Der er blevet gjort brug af forskellige kodnings sprog, som de basale HTML, CSS og JavaScript, men der er også blevet taget brug af Node.js til at skabe en lokal server og MongoDB til at skabe en database for denne server.

Problemformulering

Ved hjælp af MongoDB og Node.JS, hvordan vil vi kunne skabe en hjemmeside, hvor der kan indsættes og printes forskellig information omkring kontinenter, lande og sprog med bestemte restriktioner?

Metodeovervejelser

Tidsplan: For at holde styr på tiden, blev der lavet en tidsplan. Denne tidsplan delegerer hvornår gruppens medlemmer skal mødes, og hvad der skal nås på de forskellige dage. Som eksempel viste tidsplanen at gruppen skulle bruge mandagen på at diskutere skitsering og begynde at kigge på JS filer. Tidsplanen blev brugt vejledende, og efterlod tid til forsinkelser og udskydninger.

Brainstorming: Gruppen satte ned og læse den "Project description", som forklarede hvilke mål og krav projektet havde til formål at løse. Hertil snakkede vi om hvordan hvordan projektet kunne blive løst, og kom med mulige løsninger.

Skitsering: Mens gruppen brainstormede forskellige måder at tackle opgaven på, blev der også tegnet flere skitser til hvordan opgaven kunne løses på.

Wireframing: Gruppen blev relativt hurtigt enig om et design og den måde vi ville løse opgaven på. Da gruppen blev enige om et design, blev der bygget videre på det digitalt.

Low Fidelity Prototype: Denne digitale viderebygning blev brugt som en prototype. Dette gjorde det nemmere at fokusere målet gruppen skulle arbejde imod.

Research

Dataindsamling

Tidligt i projektet, blev der diskuteret hvordan gruppen kunne indsamle dataen, om lande, sprog og byer. Valget faldt på at lave en liste over alle lande inde i et Google Drive dokument. Disse lande var opdelt i kontinenter, og blev farvelagt røde. Når et land med tilsvarende sprog og by var færdigt, ville landet på listen blive farvelagt blå.

Herefter begyndte dataindsamling som foregik på internettet. De to første kontinenter, som gruppen begyndte at indsamle data på var Afrika og Sydamerika. Afrika var valgt pga, at vi havde en forudsætning om, at der kunne opstå nogle udfordringer i forbindelse med at finde informationer om sprog og indbyggertal i byerne. Dette viste sig at være sandt, der var ikke særligt mange artikler eller undersøgelser om sprog ratioen, indbyggertallene og styreformerne.

Ofte var den engelske Wikipedia side, det eneste sted hvor vi var i stand til at finde disse oplysninger. For at standardisere dataen, ville blev den engelske Wikipedia vores primære kilde, selvom større byer og første verdenslande oftest var væsentligt lettere at finde artikler på.

Dataindsamling endte med at tage længere end regnet med. Hvilket rent faktisk kunne tyde på, at der måske er et behov derude for en hjemmeside som tilbyder denne slags service. Gruppen valgte ikke at tilføje nogle lande, sprog eller byer til Antarktis, af den simple årsag, at der ikke er nogen bosætninger, med undtagelse af territorier og baser.

Kode Research

Hver gang der blev stødt på problemer ved selve kodning delen af projektet, så var research en vigtig del af processen for at få mere information omkring forskellige kode mæssige problemer og hvordan man muligvis ville kunne løse dem. Information var taget fra forskellige kilder til at kunne være sikker på at man gjorde ting rigtigt, dog var den største hjælp taget fra undervisningen.

For at få gennem forfrisket information omkring arrays og hvordan forskellige funktioner kunne blive sat op, så kunne dette gøres gennem research delen. Ønskede man bare mere funktionalitet, så blev det fundet ud af ved hjælp af research.

Analyse

Medium Fidelity Prototype

Der er blevet udarbejdet et medium fidelity wireframe i adobe XD, for at kunne få et overblik over en generel opbygning på siden, samt ideer til hvordan websitet kunne se ud og fungere. Valget faldt på en simpel mørkegrå baggrund, med hvid eller en lysere grå farve, som fremhæver sidens elementer. Disse elementer består af en navigation bar, samt input felter og derudover er der blevet brugt billeder som illustrerer de forskellige sideres indhold. Alle interaktive elementer, som ikke er i nav baren, er placeret centralt i venstre side af layoutet. Der har været et fokus på at holde designet minimalistisk og let overskueligt.

Link til XD fil:

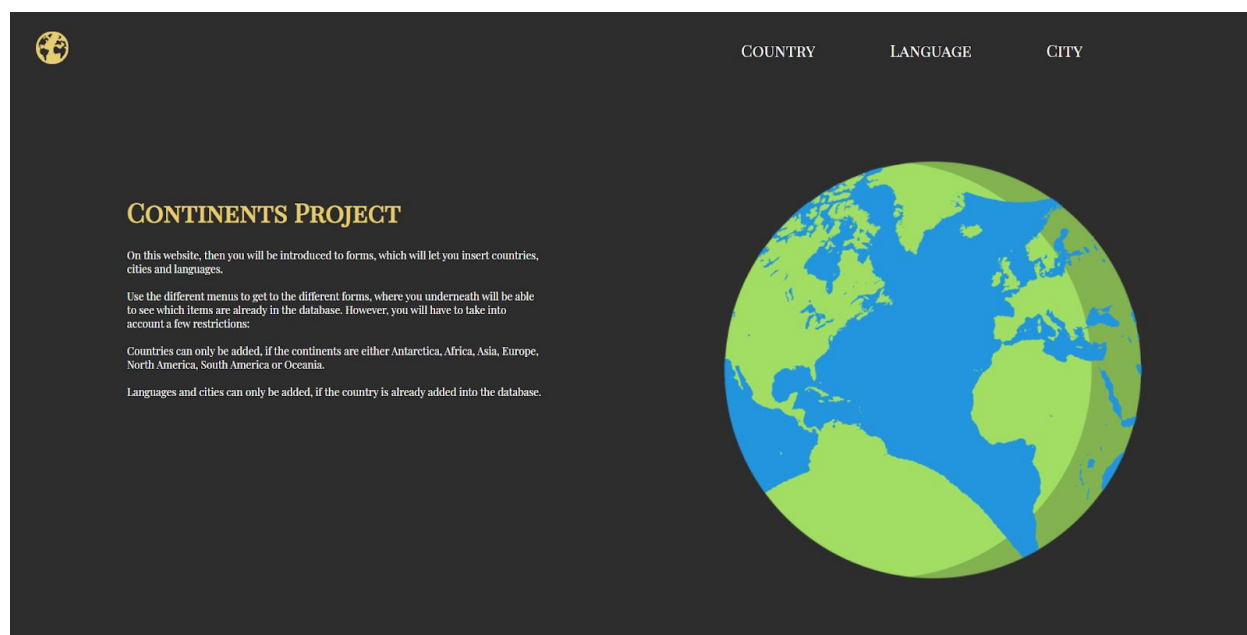
<https://xd.adobe.com/view/9e6f01be-669f-4e3b-7183-cf9b1d13143b-e85e/>

Når det kommer til det færdige produkt er blevet lavet en del ændringer i forhold til designet. Designet er blevet mere farverigt og legende, samt er der kommet flere input-felter, som nu også står mere tydeligt. Derudover er der blevet tilføjet vejledende tekst til siderne.

Sidens opbygning og design

Selve siden design har vi valgt skal være minimalistisk, da det skal være så nem så muligt at navigere rundt på siden som muligt, der er blevet valgt en mørkegrå baggrund farve og en guld farvet tekst som kører igennem alle siderne. Selve teksten på navigation har en hvid farve for at det skiller sig ud fra tekst indholdet på siden, men har derimod en guld farve når man hover over navigations teksten.

Vi har valgt farven #2d2d2d og #e7ce71 for at skabe kontrast mellem tekst og baggrund, da den ene er lys og den anden er mørk.



Konstruktion

HTML

HTML delen i denne opgave var meget minimal, da det meste af koden foregik i JavaScript/Node.js, hvilket gjorde involveringen af HTML opbygningen ganske minimal. I selve opbygning af siderne, så blev der gjort brug af dynamiske sider, hvilket blev skabt i JavaScript filer, som så byggede HTML'en op.

Dog i løbet af processen kunne det ses at der skulle have været vægtet en lille smule mere på den basale opbygning, da det bremsede selve projektet op på nogle punkter. Dette var muligvis fordi der ikke havde været blevet planlagt fra starten hvordan side opbygningen skulle have set ud.

CSS

I dette projekt, så blev CSS delen ikke vægtet specielt højt, da denne kodnings process kun blev fokus en dag i løbet af ugen, da JavaScript programmeringen sluttede. Der havde været overvejelser til designet i løbet af ugen, men fokuspunktet havde været på meget andet end

design. Få alt omkring databasen til at virke og kort have fokus på designmæssige overvejelser.

JavaScript

Ud fra javascript har vi arbejdet med forskellige if sætninger til at gøre arbejdet med forskellige restriktioner til forme virke. Arrays har også været en stor del af arbejdet, da informationen der blev sendt var objekter indsat i forskellige collection arrays. Dette er selvfølgelig en simplificeret forklaring for de forskellige ting, som så blev gjort med Node.js og MongoDB, men hovedsageligt, så var det elementer som kom til at blive brugt ved hjælp af det i JavaScript.

```
if (typeof country !== "undefined"
    && country !== null
    && country.length !== null
    && country.length > 0) {
  console.log("true");
}
```

Det at tjekke om et array var tomt eller ej blev noget der endte med at blive en stor hjælp i løbet af projektet, hvilket er en if sætning, som kan teste om et array har noget i sig. For hvis det har det, så ville det kunne bringe os noget.

Node.js

Denne nye måde at kode hjemmesider på var meget god for at nemt kunne komme til at skabe en forbindelse mellem siderne. Man havde nemt ved at kunne give forskellige methods til forskellige actions på siden, hvilket ville gøre forskellige ting ved stadig at have den samme route. For eksempel, så ville man kunne have en hjemmeside med en form, når man kom ind på den, men når man sendte formen ind, så ville node.js hjælpe med at beholde hjemmesiden på den samme route, selvom hele side opbygningen kunne var anderledes.

Denne forbindelse gennem routes ville gøre at man kunne lave forskellige dynamiske sider, når det er at man kun har for eksempel en index.html fil, hvor resten nemt kunne bygges ved bare at åbne en ny route.

```

const page = function(obj) {
  let htmltop = `<!DOCTYPE html>
<html lang="en" dir="ltr">

<head>
<meta charset="utf-8">
<title>Entry Failed</title>
<link rel="stylesheet" href="style.css">
</head>

<body>
<nav>
<a href="/"></a>
<div class="fillerbox"></div>

<a href="/country">Country</a>
<a href="/lang">Language</a>
<a href="/city">City</a>
</nav>



<div class="flex-box" style="margin-top: 250px;">
<div class="box" style="width: 700px;">
  <h2>Entry failed</h2>
  <p style="color: #fff;">Your attempt to enter ${obj.name} failed, as ${obj.country} didn't exist in the database.<br/>
  <br/>
  Please try to enter the country into the database before you try to add ${obj.name} again.
  </p>
</div>
</div>`;

  let htmlbot = `
</body>
</html>`;

  return htmltop + htmlbot;
}

exports.page = page;

```

Via Node.js, så var det også muligt at kunne gøre forskellige ting med den samme route, hvis ens method til siden var anderledes fra hinanden. Dette kunne gøres gennem GET og POST, når det var man åbner en side normalt og når man sender en form ind. Det hjalp med at skabe en mere dynamisk opbygning af siden, hvilket nemt kan give mulighed for at vise ens data.

MongoDB

MongoDB var en stor hjælp, når det var man skulle arbejde med dataindsamling på en hjemmeside, da det giver en database at arbejde med på sin Node.js server. Via update og find funktionerne i koden, så var det muligt at kunne at tilføje og læse forskellige informationer, som enten var fra eller skulle til databasen. Dette var en god hjælp i det at skulle kunne opbevare information på en nem og brugbar måde.


```

findCity(req, res) {
  const mongo = require('mongodb');
  const dbname = "world";
  const constr = `mongodb://localhost:27017`;
  const build = require('../private/showCities.js');

  mongo.connect(
    constr, { useNewUrlParser: true, useUnifiedTopology: true },
    function (error, con) {

      if (error) {
        throw error;
      }

      const db = con.db(dbname); // make dbname the current db
      /* Retrieve,
       * reads cities from the database
       */
      db.collection("city").find().toArray(function (err, city) {
        if (err) {
          throw err;
        }
        res.writeHead(httpStatus.OK, { // yes, write relevant header
          "Content-Type": "text/html; charset=utf-8"
        });
        res.write(build.page(city));
        res.end();
        con.close();
      });
    });
},

```

På billedet her kan man se hvordan en af kode afsnittene i handlers.js fungerer med dens tilkobling til databasen for at læse information fra databasen og derefter bygge siden, hvor det skal vises på. Her handler det bare om at finde dataen først og så fortsætte med at vise siden, som skulle bruges, hvor man derefter lukker forbindelsen og fortsætter med dataen man har hentet næsten som om selve den data hele tiden havde været på siden.

Evaluering af proces

Kodemæssig evaluering

I løbet af selve arbejdet med kodningen af hjemmesiden til visning og indsættelse af information omkring lande, byer og sprog, så var der nogle få problemer, som vi stødte på. Disse problemer handlede meget omkring selve dataindsamlingsprocessen og dens interaktioner med serveren.

I vores test fase af selve funktionerne, så gjorde vi brug af index siden for test af formen og connection med databasen, dog stødte vi på et problem med serveren. Da man på index siden stødte på et problem med at serveren ikke tog dataen med sig, selvom method i <form> var blevet sat til at sende en action via POST, dog sendte det som GET, hvilket kunne løses ved at ændre noget i router.js.

```
} else if (req.url === "/start" && req.method === "GET") {  
    asset = req.url;  
    routedUrl = "views/index.html";  
    type = contentType.html;
```

Et andet problem, som var blevet mødt var under skabelsen af restriktioner på selve indsættelse af data. Vi havde problemer med at få læst information som skulle tjekkes igennem før det var at det kunne opdateres/indsættes ind i databasen. Løsningen til dette problem er ikke helt klart, men det endte med at virke på en eller anden måde.

I forhold til noget andet med restriktioner, så var der i et kort øjeblik glemt at informationen der blev læst var et array, så det gav nogle korte problemer, men blev hurtigt løst igen.

Igen med restriktioner, så var der et problem, når man ikke overholdt de forskellige restriktioner, hvilket hovedsageligt lå i at der ikke var noget information i arrayet man blev sendt, når det var tjekket. Dette blev løst ved at lave en if sætning, som skulle tjekke om arrayet var tomt.

```

mongo.connect(
  constr, { useUrlParser: true, useUnifiedTopology: true },
  function (error, con) {

    if (error) {
      throw error;
    }

    const db = con.db(dbname); // make dbname the current db
    /* Retrieve,
     * reads cities from the database
     */
    db.collection("country").find({name: information.POST.country}).toArray(function (err, country) {
      if (err) {
        throw err;
      }
      if (typeof country !== "undefined"
        && country !== null
        && country.length !== null
        && country.length > 0) {
        console.log("true");

        console.log(country);
        if (information.POST.country === country[0].name) {
          db.collection("city").updateOne(
            { "$set": info }, { upsert: true }, function (err, collection) {
              if (err) {
                throw err;
              }

              res.writeHead(httpStatus.OK, { // yes, write relevant header
                "Content-Type": "text/html; charset=utf-8"
              });
              console.log("City inserted/updated");
              res.write(success.page(info));
              res.end();
              con.close();
            });
        } else {
          res.writeHead(httpStatus.OK, { // yes, write relevant header
            "Content-Type": "text/html; charset=utf-8"
          });
          console.log("entry failed");
          res.write(fail.page(info));
          res.end();
          con.close();
        }
      }
    });
  }
);

```

Projektmæssigt evaluering

Når det er vi kigger på hvordan sammenholdet og selve gruppearbejdet har fungeret, så er et godt sted at kigge på vil være tidsplanen og hvordan den er blevet fulgt.

Fredag og weekend: Tidsplan, problemformulering og skitsering, opsættelse af server og git repo.

Mandag	Tirsdag	Onsdag	Torsdag	Fredag	Lørdag	Søndag
Basal opbygning af HTML / Diskussion omkring skitsering / Begyndelse på JS filer.	Research omkring lande / Fortsat JS	CSS Begyndelse / Fortsat JS	Småt begyndelse på rapport / Færdiggørelse af CSS / Fortsat JS	Rapport / Færdiggørelse af JS	Rapport	Aflevering / powerpoint

I forhold til denne tidsplan, så fulgte vi planen godt, dog blev JavaScript skubbet ned på tre dage i stedet for de fem dage, som det havde været givet. I forhold til rapportskrivning, så blev denne process skubbet ned til bare at foregå på torsdag og fredag, men ellers var alting stort set overholdt.

Gruppearbejdet foregik roligt nok og med minimale problemer mellem gruppemedlemmerne. Fordelingen af arbejdsopgaver var måske ikke helt korrekt, men udover det, så lod det til at alt fungerede med det. Det eneste, som muligvis kunne have været en idé at bringe ind i det hele fra starten ville have været en kontrakt mellem medlemmerne. Denne kontrakt kunne have været med til at skabe bedre idéer om hvad man bragte ind i projektet med hinanden, men da projektet gik godt nok til sidst, så er det bare noget som kan tænkes at gøre brug af en anden gang i fremtiden.

Konklusion

I løbet af dette projektet omkring verden og dens kontinenter, så blev der sat et fokus på kodning af funktionaliteten på selve hjemmesiden, hvor designet blev sat lidt på sidelinjen. Dog var projektet også skabt for at give mulighed for netop denne tilgang til opgaven. Det handlede om at arbejde med Node.js og MongoDB og den sammenhæng, som de har med hinanden.

Samarbejdet i projektet gik fremragende og tidsplanen var fulgt. Selve opdelingen af opgaver havde muligvis nogle problemer, men der kunne nemt blive talt med hinanden om de forskellige ting i projektet, som skulle tales om. Den kreative del var skabt gennem brainstorming mellem gruppemedlemmer gennem ugen og til sidst sat sammen til en hjemmeside, som gruppen er tilfreds med.

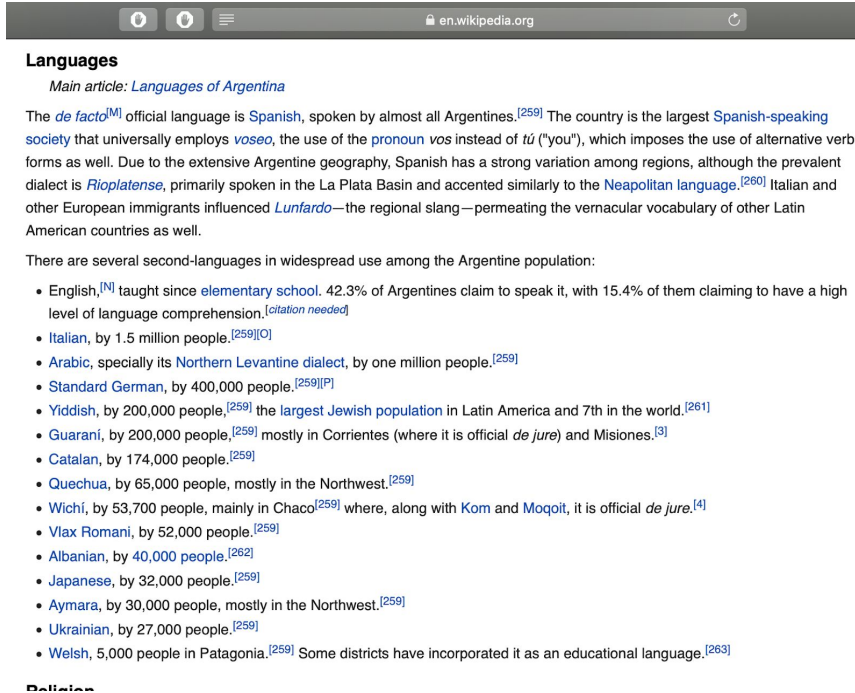
I forhold til opgavens krav om restriktioner til indsættelse af data, så blev der fundet nogle problemer på hjemmesiden. Dette indeholdte nogle problemer med at opfange information og bruge det som et krav, dog blev det løst og der blev også tilføjet en if sætning som gjorde det muligt at teste om arrayet fra databasen var tomt eller ej, for ellers ville serveren crash.

Grundlæggende, så gik projektet godt og ligetil, selvom der var et par enkelte problemer på hjemmesiden i forhold til kodning, men det var noget, som nemt kunne blive fikset ved hjælp af research. Research var også brugt til forskellig dataindsamling til databasen, så der var information på selve hjemmesiden.

Så alt i alt, der var ingen store problemer med projektet og samarbejdet var godt, hvilket gjorde selve oplevelsen af at lave projektet til en god ting. Selve produktet er velfungerende og virker på den måde, som det var udtænkt med de forskellige restriktioner.

Literaturliste

Title: Wikipedia, URL: https://en.wikipedia.org/wiki/Main_Page, last visited: 20/02/2020



The screenshot shows the Wikipedia Main Page. At the top, there's a navigation bar with the Wikipedia logo and a search bar. Below the navigation bar, the page title "Languages" is displayed. The main content area discusses the official language of Argentina, Spanish, and mentions various regional dialects and second languages spoken in the country. A list of second languages is provided, including English, Italian, Arabic, Standard German, Yiddish, Guarani, Catalan, Quechua, Wichi, Vlax Romani, Albanian, Japanese, Aymara, Ukrainian, and Welsh.

Languages

Main article: *Languages of Argentina*

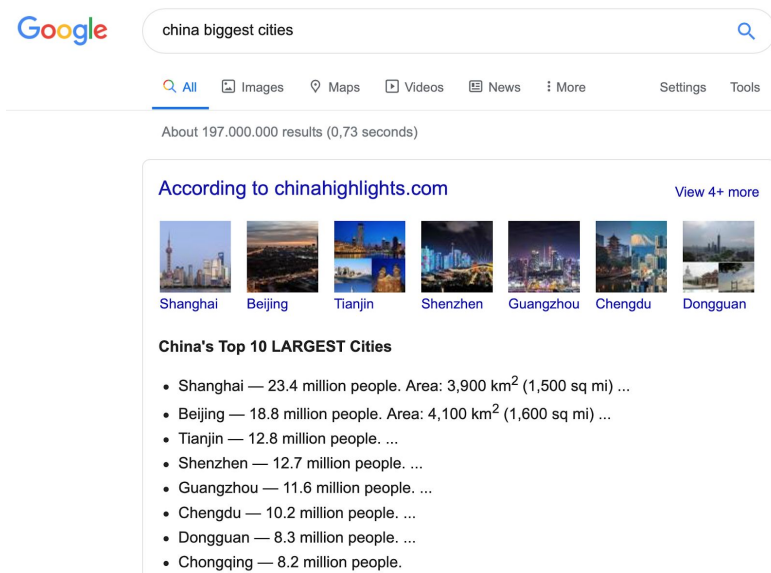
The *de facto*^[M] official language is **Spanish**, spoken by almost all Argentines.^[259] The country is the largest **Spanish-speaking society** that universally employs *voseo*, the use of the pronoun *vos* instead of *tú* ("you"), which imposes the use of alternative verb forms as well. Due to the extensive Argentine geography, Spanish has a strong variation among regions, although the prevalent dialect is *Rioplatense*, primarily spoken in the La Plata Basin and accented similarly to the *Neapolitan language*.^[260] Italian and other European immigrants influenced *Lunfardo*—the regional slang—permeating the vernacular vocabulary of other Latin American countries as well.

There are several second-languages in widespread use among the Argentine population:

- English,^[N] taught since **elementary school**. 42.3% of Argentines claim to speak it, with 15.4% of them claiming to have a high level of language comprehension.^[citation needed]
- Italian, by 1.5 million people.^{[259][O]}
- Arabic, specially its **Northern Levantine dialect**, by one million people.^[259]
- Standard German, by 400,000 people.^{[259][P]}
- Yiddish, by 200,000 people,^[259] the **largest Jewish population** in Latin America and 7th in the world.^[261]
- Guarani, by 200,000 people,^[259] mostly in Corrientes (where it is official *de jure*) and Misiones.^[3]
- Catalan, by 174,000 people.^[259]
- Quechua, by 65,000 people, mostly in the Northwest.^[259]
- Wichi, by 53,700 people, mainly in Chaco^[259] where, along with *Kom* and *Moqoit*, it is official *de jure*.^[4]
- Vlax Romani, by 52,000 people.^[259]
- Albanian, by 40,000 people.^[262]
- Japanese, by 32,000 people.^[259]
- Aymara, by 30,000 people, mostly in the Northwest.^[259]
- Ukrainian, by 27,000 people.^[259]
- Welsh, 5,000 people in Patagonia.^[259] Some districts have incorporated it as an educational language.^[263]

Definition

Title: Google, URL: <https://www.google.com>, last visited: 20/02/2020



The screenshot shows a Google search for "china biggest cities". The search bar is at the top, and the results are displayed below. The first result is from "chinahighlights.com" and is titled "China's Top 10 LARGEST Cities". It includes a list of cities with their populations and areas.

Google

china biggest cities

Q All Images Maps Videos News More Settings Tools

About 197,000,000 results (0,73 seconds)

According to chinahighlights.com [View 4+ more](#)

Shanghai Beijing Tianjin Shenzhen Guangzhou Chengdu Dongguan

China's Top 10 LARGEST Cities

- Shanghai — 23.4 million people. Area: 3,900 km² (1,500 sq mi) ...
- Beijing — 18.8 million people. Area: 4,100 km² (1,600 sq mi) ...
- Tianjin — 12.8 million people. ...
- Shenzhen — 12.7 million people. ...
- Guangzhou — 11.6 million people. ...
- Chengdu — 10.2 million people. ...
- Dongguan — 8.3 million people. ...
- Chongqing — 8.2 million people.

Title: MongoDB: db.collection.findOne() method, URL: <https://www.w3resource.com/mongodb/shell-methods/collection/db-collection-findOne.php>,

last visited: 19/02/2020

Title: Check if an array is empty or not in JavaScript, URL:

<https://www.geeksforgeeks.org/check-if-an-array-is-empty-or-not-in-javascript/>, last visited: 20/02/2020

Bilag

Link til Adobe XD wireframe.

<https://xd.adobe.com/view/9e6f01be-669f-4e3b-7183-cf9b1d13143b-e85e/>






Country Language City

Country






CountryLanguageCity

Language

תודה
 Dankie Gracias
 شكر
 Спаси́бо
 Köszönjük
 Grazie
 Dziękujemy
 Ďakujeme
 Kiitos
 Tāname teid
 谢谢
 Tak
 Terima kasih
 Dékojame
 Vielen Dank
 Paldies
 Tak
 Teşekkür ederiz
 Obrigado
 감사합니다
 Σας Ευχαριστούμ
 ขอบคุณ
 Bedankt
 Děkuje vám
 ありがとうございます
 Tack



CountryLanguageCity

City

