Human Body Interface

An Illustrative Tool

PRA3006 - Programming in the Life Sciences

Shreyas, Nick, Justin, Stijn

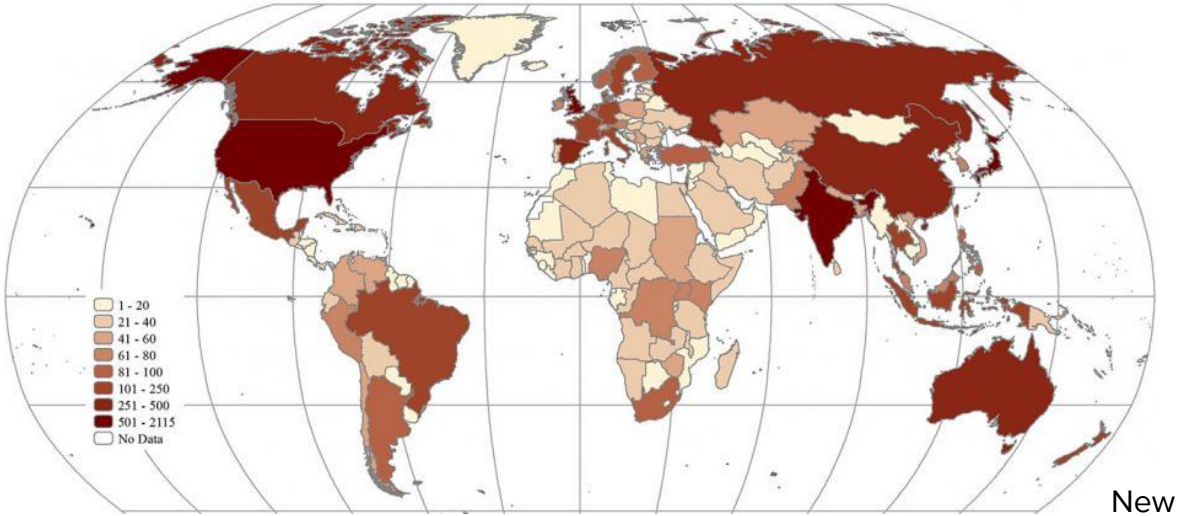
Overview

- Introduction and Research Question
- Website Presentation
- Methods
  - HTML/CSS
  - WikiData and SPARQL Query
  - JS
  - D3.js
- Results & Discussion

Introduction

Problems

- ➔ Lack of an approachable tool for teaching about diseases
- ➔ Interactive tools are better for retaining information
- ➔ How to make this tool clear and concise



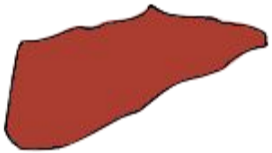
New Diseases since 1980

Research Question

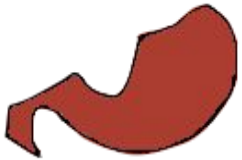
How to build an interface to inform about the most common diseases per body part and how to display this in a clear and concise way?

To demonstrate we used:

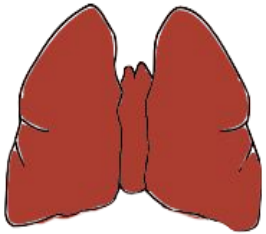
Liver



Stomach



Lungs



Brain

The Human Body Inface

human lung

- pulmonary sarcoidosis
- squamous cell carcinoma of the lung [hide]
- chronic granulomatous disease [hide]

Chronic granulomatous disease (CGD, also known as Bridges–Good syndrome, chronic granulomatous disorder, and Quie syndrome, is a diverse group of hereditary diseases in which certain cells of the immune system have difficulty forming the reactive oxygen nonspecies (most importantly the superoxide radical due to defective phagocyte NADPH oxidase) used to kill certain ingested pathogens.

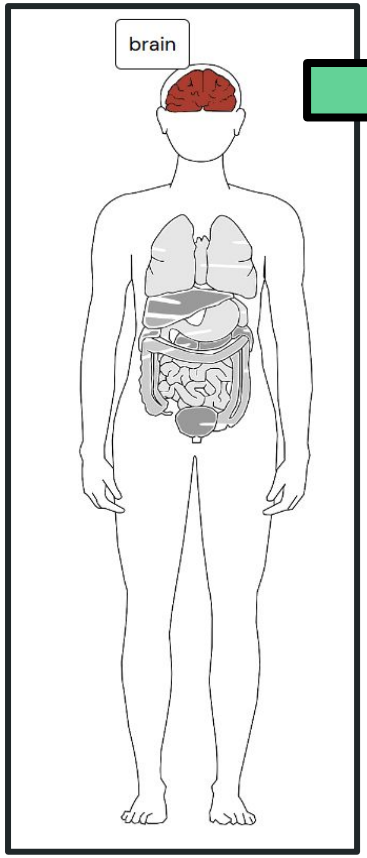
- List of drugs that physically interact with the disease

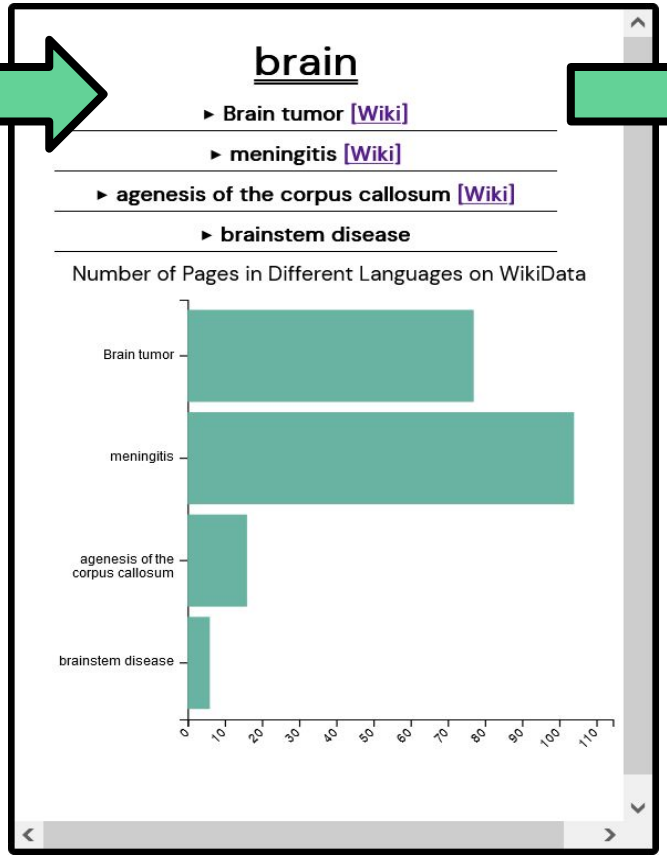
interferon gamma-1b

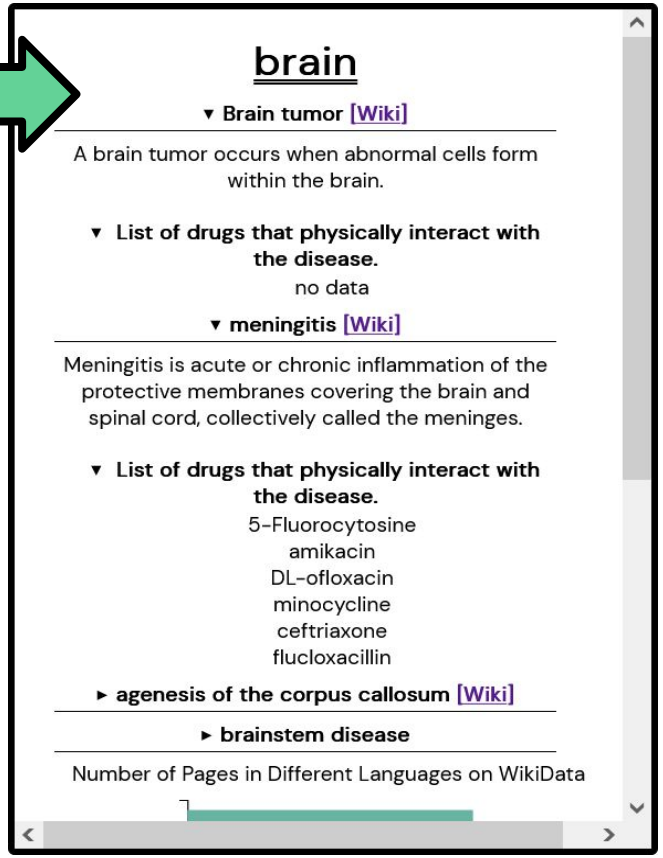
- pulmonary tuberculosis

Number of Pages in Different Languages on WikiData

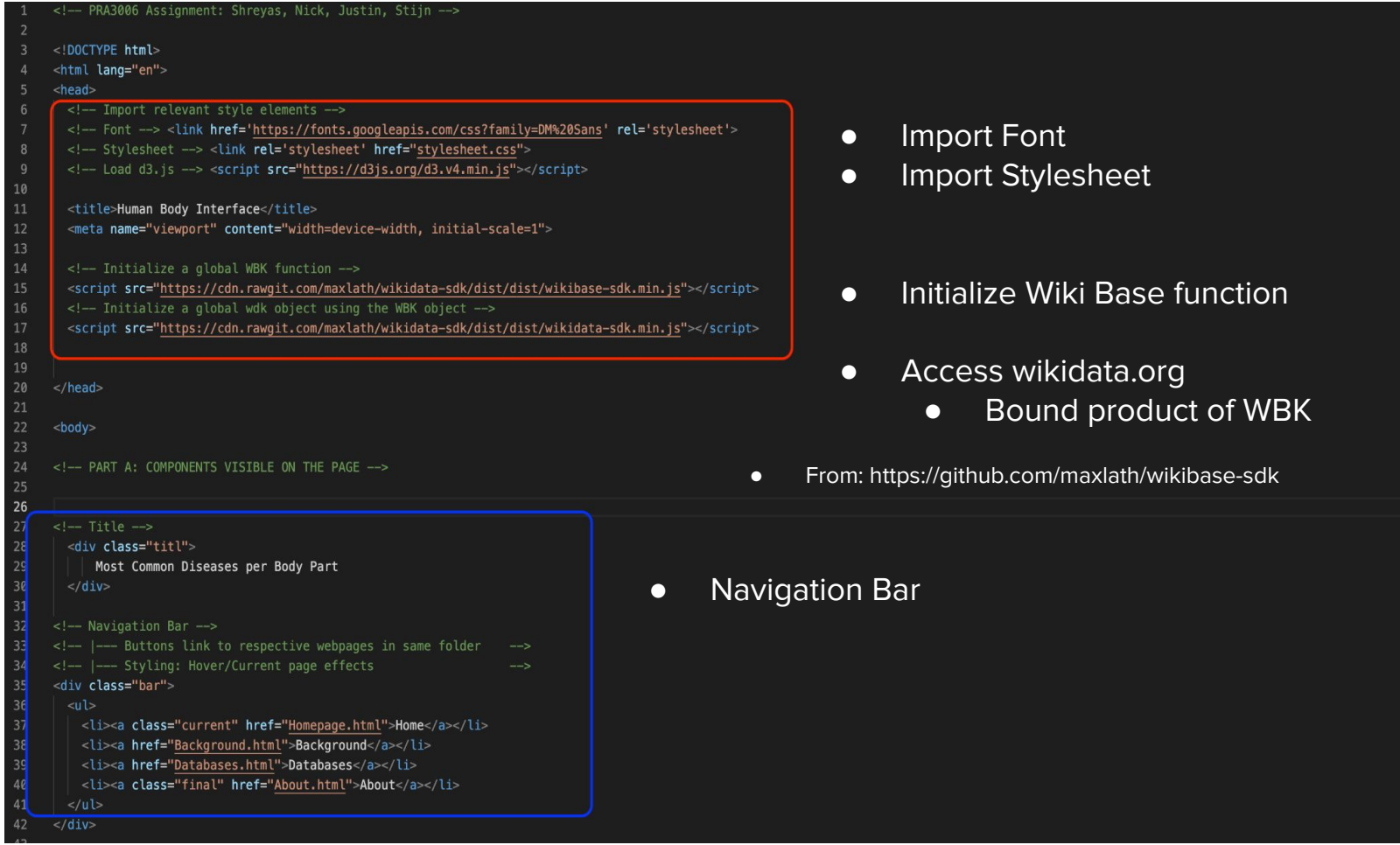
How to get there?







HTML



```
<!-- PRA3006 Assignment: Shreyas, Nick, Justin, Stijn -->

<!DOCTYPE html>
<html lang="en">
<head>
  <!-- Import relevant style elements -->
  <!-- Font --> <link href='https://fonts.googleapis.com/css?family=DM%20Sans' rel='stylesheet'>
  <!-- Stylesheet --> <link rel='stylesheet' href="stylesheet.css">
  <!-- Load d3.js --> <script src="https://d3js.org/d3.v4.min.js"></script>

  <title>Human Body Interface</title>
  <meta name="viewport" content="width=device-width, initial-scale=1">

  <!-- Initialize a global WBK function -->
  <script src="https://cdn.rawgit.com/maxlath/wikidata-sdk/dist/dist/wikibase-sdk.min.js"></script>
  <!-- Initialize a global wdk object using the WBK object -->
  <script src="https://cdn.rawgit.com/maxlath/wikidata-sdk/dist/dist/wikidata-sdk.min.js"></script>


</head>

<body>

<!-- PART A: COMPONENTS VISIBLE ON THE PAGE -->


<!-- Title -->
  <div class="titl">
      Most Common Diseases per Body Part
  </div>

<!-- Navigation Bar -->
<!-- |--- Buttons link to respective webpages in same folder    -->
<!-- |--- Styling: Hover/Current page effects                   -->
<div class="bar">
  <ul>
    <li><a class="current" href="Homepage.html">Home</a></li>
    <li><a href="Background.html">Background</a></li>
    <li><a href="Databases.html">Databases</a></li>
    <li><a class="final" href="About.html">About</a></li>
  </ul>
</div>
```

- Import Font
- Import Stylesheet
- Initialize Wiki Base function
- Access wikidata.org
  - Bound product of WBK
- From: https://github.com/maxlath/wikibase-sdk
- Navigation Bar

CSS

Global Code for all webpages

```css
/* Style elements that apply to all webpages */


.titl {
  width: calc(100%);
  height: 55px;
  background-color: #696969;
  text-align: center;
  font-size: calc(200%);
  font-family: 'DM Sans';
  padding: 10px 0px;
}

body {
  margin: 0;
  font-family: 'DM Sans';
  min-width: 1300px;
}
```

```css
/* Navigation bar */

.bar {
  background-color: #E5DACE;
}

ul {
  list-style-type: none;
  margin: 0;
  padding: 0;
  overflow: hidden;
  font-family: 'DM Sans';
}

li a {
  display: block;
  background-color: #E5DACE;
  color: #000000;
  padding: 14px 16px;
  text-decoration: none;
  float: left;
}

li a.current {
  background-color: #E5DACE;
}

li a.final {
  float: right;
}

li a:hover {
  background-color: #5C5350;
}
```

CSS

Code for the About page

```
/*  Style elements that apply only to About  */


.photos img {
    height: 250px;
    width: 250px;

}

/* Give the photos a corner border */
/* Got this from Arkej, 2017 - */
/* https://stackoverflow.com/questions/42832749/create-corner-border-in-css */

.photoBor {
    position: relative;
    margin: 20px;
    width: 250px;
    height: 250px;

}
```

```
.photoBor:after {
    border-top: 15px solid #696969;
    border-right: 15px solid #696969;
    display: block;
    width: 50px;
    height: 50px;
    position: absolute;
    top: -25px;
    right: 40px;

}

.photoBor:before {
    border-bottom: 15px solid #696969;
    border-left: 15px solid #696969;
    display: block;
    width: 50px;
    height: 50px;
    position: absolute;
    bottom: -25px;
    left: 40px;

}

/* Line and Center the Four Profile Pictures */
.photos {
  display: flex;
}

.photos > div {
  flex: 1; /*grow*/
}
```

CSS

Code for the Elements on the Homepage

```
/*  Style elements that apply only to Homepage  */

.stepOrdinal {
  width: 28px; height: 28px;
  display:inline-block;
  text-align: center;
}

.selection {
  background-color:  #696969;
  text-align: center;
  vertical-align: middle;
  height: 180px;
  line-height: 90px;
  font-size: 2em;
}

.buttonRowHolder {
  text-align: center;
  line-height: 0px;
}
```

```
.clusterButton {
  font-size: 16px;
  font-family: 'DM Sans';
  width: 120px;
  height: 48px;
  background-color: #f2d9f2;
  border: none;
  padding: 5px 10px;
  transition: transform 330ms ease-in-out;
  display: inline-block;
}

.clusterButton:hover {
  transform:scale(1.3,1.3);
}

.legend {
  font-size: 1.3em;
  display:inline-block;
  text-align: left;

}

#legendLabel {
  font-size: 1.3em;
}
```

The SPARQL Query


```
SELECT DISTINCT ?bodypartLabel ?diseaseLabel ?drugLabel ?numLang ?article
WHERE {

    { SELECT ?disease ?bodypart ?drug (count(?lang) as ?numLang) WHERE {
      ?disease wdt:P31 wd:Q12136. #Is an instance of a disease.
      ?disease wdt:P927 ?bodypart.
      OPTIONAL {?disease wdt:P2176 ?drug.}
      ?disease rdfs:label ?label
      filter(!langmatches(lang(?label), 'en')) bind(lang(?label) as ?lang)
    } GROUP BY ?disease ?bodypart ?drug
     }
    FILTER (?numLang > 4).
    OPTIONAL{
      ?article schema:about ?disease .
      ?article schema:inLanguage "en" .
      FILTER (SUBSTR(str(?article), 1, 25) = "https://en.wikipedia.org/")
    }
    SERVICE wikibase:label { bd:serviceParam wikibase:language "[AUTO_LANGUAGE],en" }
    } ORDER BY (?numLang)
```

The Results of the Query

liver	liver cirrhosis	boceprevir	110	<https://en.wikipedia.org/wiki/Cirrhosis>
liver	liver cirrhosis	furosemide	110	<https://en.wikipedia.org/wiki/Cirrhosis>
liver	liver cirrhosis	adefovir	110	<https://en.wikipedia.org/wiki/Cirrhosis>
liver	liver cirrhosis	bumetanide	110	<https://en.wikipedia.org/wiki/Cirrhosis>
meninges	meningitis	flucloxacillin	104	<https://en.wikipedia.org/wiki/Meningitis>
meninges	meningitis	ceftriaxone	104	<https://en.wikipedia.org/wiki/Meningitis>
meninges	meningitis	minocycline	104	<https://en.wikipedia.org/wiki/Meningitis>
meninges	meningitis	DL-ofloxacin	104	<https://en.wikipedia.org/wiki/Meningitis>
meninges	meningitis	amikacin	104	<https://en.wikipedia.org/wiki/Meningitis>
meninges	meningitis	5-Fluorocytosine	104	<https://en.wikipedia.org/wiki/Meningitis>
scalp	dandruff		78	<https://en.wikipedia.org/wiki/Dandruff>
brain	Brain tumor		77	<https://en.wikipedia.org/wiki/Brain_tumor>
nervous system	neurological disorder	pregabalin	51	<https://en.wikipedia.org/wiki/Neurological_disorder>
nervous system	neurological disorder	gabapentin	51	<https://en.wikipedia.org/wiki/Neurological_disorder>

Wikipedia API

```
//generates a wikipedia api call to get the first sentence of the associated wikipedia article
async function wikipedia_intro(str1) {

    //The very braindead method of finding the title from the wikipedia url by removing everything before the page name
    let str = String();
    str = str1.replace("https://en.wikipedia.org/wiki/", "")

    var url = "https://en.wikipedia.org/w/api.php";

    var params = {
        action: "query",
        prop: "extracts",
        exsentences: "1",
        explaintext: "1",
        format: "json",
        titles: str
    };

    url = url + "?origin=*";
    Object.keys(params).forEach(function(key){url += "&" + key + "=" + params[key];});

    let res = await fetch(url);
    let wiki =  await res.text();
    wik = wiki.split('"extract":"').pop().split('"}}}}')[0]
    // ^ removes everything but the extract itself. Again, not the best execution, but functional
    return unicodeToChar(wik)

}
```

Asynchronous Calls

Problems:

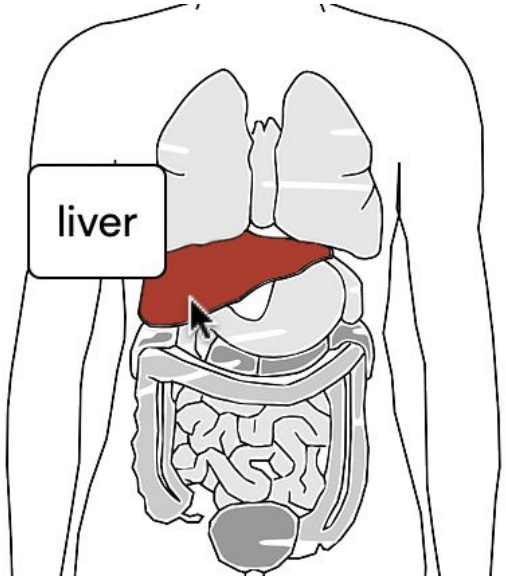
- Fetching data from a database in Javascript creates a promise
- Fetched data is not in a manageable format by default

Solutions:

- All operations with data are wrapped in the async function
- An online-hosted package was used to fetch data from Wikidata and easily convert it to JSON (https://github.com/maxlath/wikibase-sdk)

D3.js Library

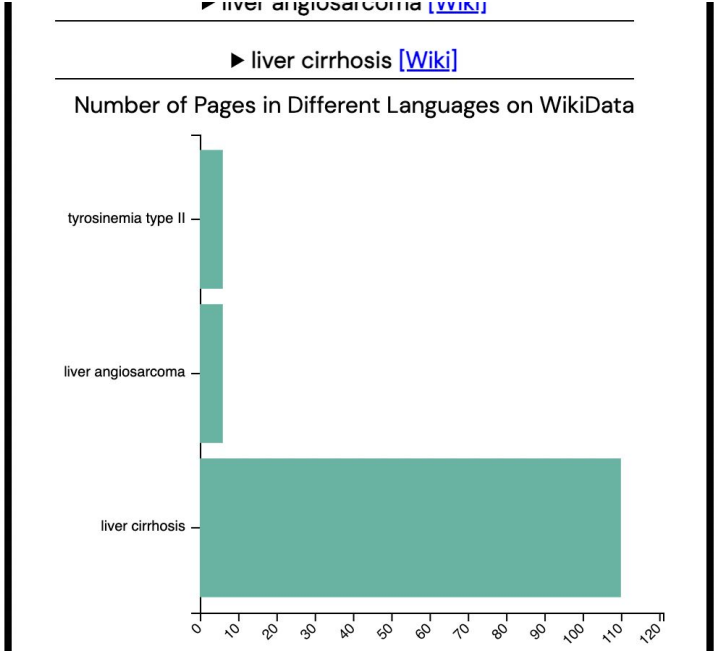
D3.js library was used to implement interactive buttons, mouse tracking, and tooltips and pop-ups with information




```
// visualize tooltips and popups
for (let i = 0; i < bodyPartArray.length; i++) {
    d3.select("#circleCustomTooltip" + i)
        .on("click", function() {
            closePopups();
            d3.select("#highlightedImage"  + i).attr("visibility", "visible");
            popups[i].style("visibility", "visible");
        })
        .on("mouseover", function() {
            if (popups[i].style("visibility") === "hidden")
                tooltips[i].style("visibility", "visible");
            d3.select("#highlightedImage"  + i).attr("visibility", "visible");
        })
        .on("mousemove", function() {
            tooltips[i].style("top", (event.pageY- 50)+"px").style("left",(event.pageX-60)+"px");
        })
        .on("mouseout", function() {
            if (popups[i].style("visibility") === "visible") {
                tooltips[i].style("visibility", "hidden");
            } else {
                tooltips[i].style("visibility", "hidden");
                d3.select("#highlightedImage"  + i).attr("visibility", "hidden");
            }
            letClose = true;
        });
}
```

D3.js Library

D3.js was also used to create the visual representation of data in form of the bar chart with the number of languages for a wikidata page for each disease




```
//Bars
svg.selectAll("myRect")
    .data(Object.keys(disease))
    .enter()
    .append("rect")
    .attr("x", x(0))
    .attr("y", function (d) {
        return y(d);
    })
    .attr("width", function (d) {
        return x(disease[d]);
    })
    .attr("height", y.bandwidth())
    .attr("fill", "#69b3a2")
```

Goal

Automating the processing and visualization of the query results

How?

Store all the required information in a list and dictionary

In such form [BodyPart, [disease, WikipediaLink, numberLang, drugList]]

1. Find all the body parts in **temporary sets**.
2. Find all the **associated diseases**.
3. Find the relevant information for each disease.
4. Store the lists on the right format.

Reuse the formatted data in the pop-up text, graphs,...

Problems

Requires manual input for the location on the body image

Several organs or sub-organs at the same location

Advantages

Easy to manipulate - changing the list of body parts impacts the whole visualisation (pop up, text, graph,...)

Disadvantages

Computer-intensive

Loads everything at once

Loads unused information

Important variables
bodyPartLst - list of wanted organs
results - information from query
dataLst, dataDic - formatted output data

for loop to get the list of disease and the associated information (wikipedia, languages)

for loop for all the drugs related to each disease

Rearrange the temporary lists in the right order

Store the body part and the related information

```
138    for (let el in bodyPartLst){
139        // find all the diseases and associated data (wiki and number of Language) for each body part
140        for (let i = 0; i < results.length; i++){
141            if (results[i].bodypartLabel === bodyPartLst[el]){
142                if (!testSet.has(results[i].diseaseLabel)){
143                    diseaseSet.add([results[i].diseaseLabel, results[i].article, results[i].numLang]);
144                }
145                testSet.add(results[i].diseaseLabel);
146            }
147        }
148        // Create a list of out it (easier to work with because of indexes)
149        let diseaseLst = Array.from(diseaseSet);
150        // Resets sets for next bodypart
151        diseaseSet.clear();
152        testSet.clear();
153
154        // Go through all the disease and find all the associated drugs
155        let myDiseaseLst = [];
156        for (let el2 in diseaseLst){
157            for (i=0; i<results.length; i++){
158                // diseaseLst[el2][0] only selects the disease out of the disease information
159                if (results[i].diseaseLabel === diseaseLst[el2][0]){
160                    drugSet.add(results[i].drugLabel);
161                }}
162            let drugLst = [];
163            drugSet.forEach(function(value){
164                if (value == undefined){
165                    drugLst.push("no data");
166                } else{drugLst.push(value);}
167            })
168            myDiseaseLst.push([diseaseLst[el2][0], diseaseLst[el2][1], diseaseLst[el2][2], drugLst]);
169            drugSet.clear();
170        }
171        // Add all the disease and related information next to the bodypart
172        dataLst.push([bodyPartLst[el], myDiseaseLst]);
173        dataDic[bodyPartLst[el]] = myDiseaseLst;
174    }
```

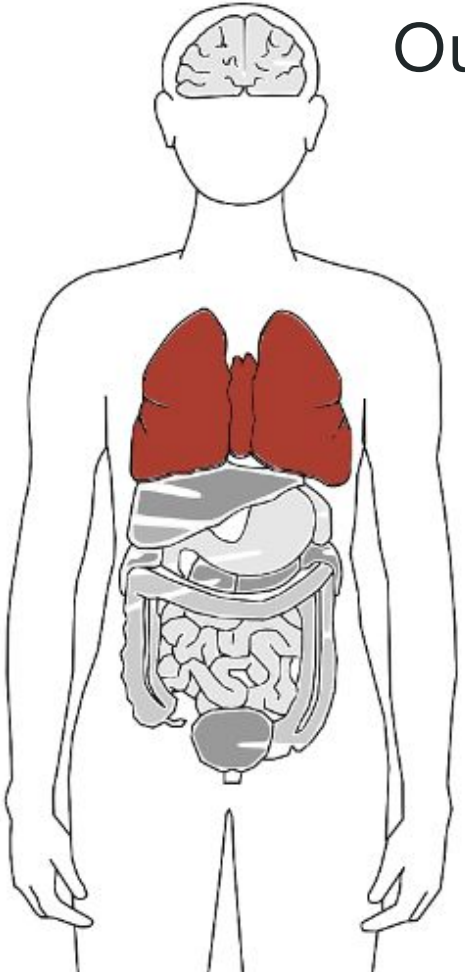
Add HTML tags around the information from the data list to display on the wanted format

If applicable, extract the Wikipedia

List and drop-down menus

Store for each body part the title separated from the text

```
220     // Go through the data list and makes a text with html tags out of it
221     for (el in dataLst){
222         myText = "";
223         bodyPart = dataLst[el][0]; // Name of the bodyPart
224         theDiseaseLst = dataLst[el][1]; // All the information about the bodyPart
225         myTitle = "<h1>" + bodyPart + "</h1> \n";
226         // Go through all the associated information
227         for (el2 in theDiseaseLst){
228             disease = theDiseaseLst[el2][0]; // Name of the disease
229             diseaseWiki = theDiseaseLst[el2][1]; // WikiLink
230             theDrugLst = theDiseaseLst[el2][3]; // List of drugs associated with the disease
231             myText += "<details>";
232             if (diseaseWiki !== undefined){
233                 // This is where we get the Wikipedia extract for every disease
234                 wik = await wikipedia_intro(diseaseWiki);
235                 myText += "<summary class='summary'>" + disease + " <a href = '"
236                 myText += diseaseWiki + "' target = '_blank'>[Wiki]</a></summary>" + wik;
237             }else{
238                 myText += "<summary class='summary'>" + disease + "</summary>";
239             }
240
241             myText += "<ul><details><summary class='subTitle'>"
242             myText += "List of drugs that physically interact with the disease.</summary>";
243             for (el3 in theDrugLst){
244                 myText += "<li>" + theDrugLst[el3] + "</li>";
245             }
246             myText += "</ul></details></details>";
247         }
248         // Create a dictionnary with the title separated from the text
249         // Allows to only display the title of only one bodyPart
250         textDict[bodyPart] = [myTitle, myText];
251     }
```



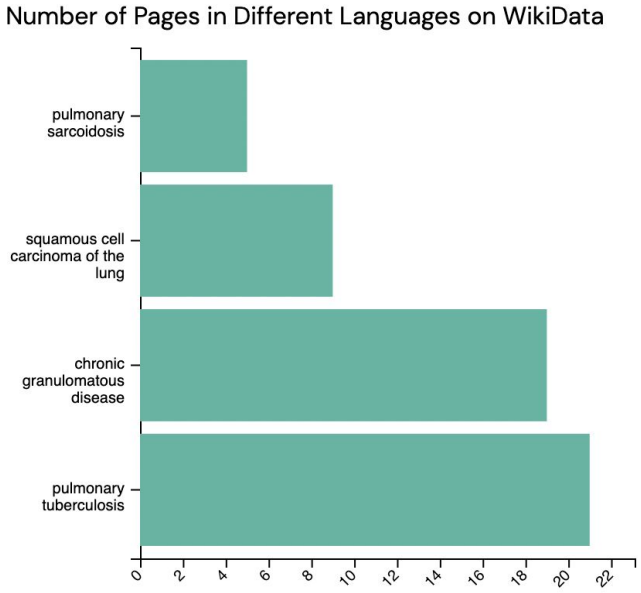
Our Visualization

- Inherently intuitive
- Visually appealing
- Simple implementation

Limitations

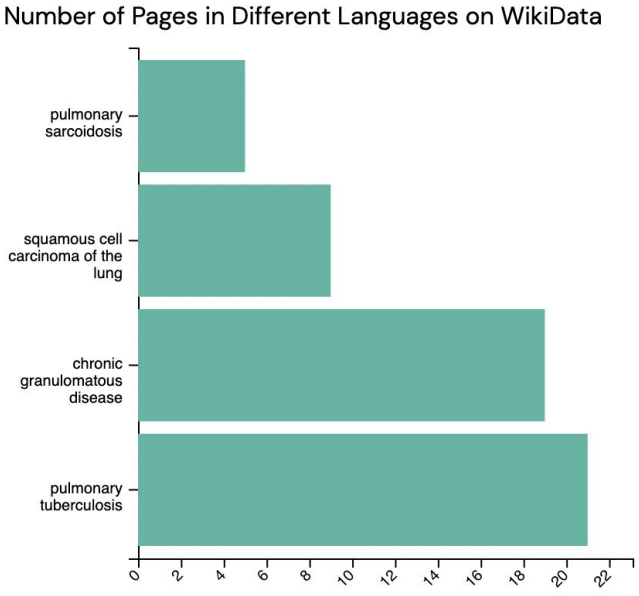
- The organs available for visualization are picked by us and not automated (may be possible to display every organ WikiData has information on)
- 3D visualization of the body would provide more space and interaction than 2D.
- WikiData was used as a reliable and broad database, but other more specialized databases may have more detailed information about diseases.
- Prevalence of diseases is estimated based on the number of languages WikiData page for the disease is available in.
- Information about disease linked to a certain body part but no generalized disease

Number of Languages Estimation vs Real Incidence



- Pulmonary sarcoidosis - **1-140** (varies per region) **per 100,000** (Arkema & Cozier, 2020)
- Squamous cell carcinoma - **14 per 100,000** (high mortality) (Molina, et al., 2008)
- Granulomatous disease - **1 per 200,000** (high mortality) (Rider, et al., 2018)
- Tuberculosis - **130 per 100,000** (high mortality) (WHO, 2022)

Number of Languages Estimation vs Real Incidence



- Pneumonia, asthma, chronic obstructive pulmonary disease, **lung cancer**, and **tuberculosis** are the five most important lung diseases worldwide from a prevalence standpoint (Soriano, et al., 2020).
- While not exactly accurate, such method of estimation produces relatively adequate results.

Does the Result Answer the Research Question?

- We were able to build a clear and intuitive interface
- Information about the diseases is presented concisely and interactively
- Some sections for several diseases are empty due to the limitations of WikiData
- Estimation of most prevalent diseases is very rough although effective
- Accuracy of the estimation can vary between organs
- Prevalence of diseases often depends significantly on location

Future Outlook

- Improved body model
  - 3D model or coordinates mapping
- Reduce loading time
  - Only load the data once
  - Filter out unnecessary information
- Additional organs

References

Rider, N. L., Jameson, M. B., & Creech, C. B. (2018). Chronic Granulomatous Disease: Epidemiology, Pathophysiology, and Genetic Basis of Disease. *Journal of the Pediatric Infectious Diseases Society*, *7*(suppl_1), S2–S5. https://doi.org/10.1093/jpids/piy008

Arkema, E. V., & Cozier, Y. C. (2020). Sarcoidosis epidemiology: recent estimates of incidence, prevalence and risk factors. *Current Opinion in Pulmonary Medicine*, *26*(5), 527–534. https://doi.org/10.1097/mcp.0000000000000715

Tuberculosis (TB). (2022, October 27). https://www.who.int/news-room/fact-sheets/detail/tuberculosis

Molina, J. R., Yang, P., Cassivi, S. D., Schild, S. E., & Adjei, A. A. (2008). Non-Small Cell Lung Cancer: Epidemiology, Risk Factors, Treatment, and Survivorship. *Mayo Clinic Proceedings*, *83*(5), 584–594. https://doi.org/10.4065/83.5.584

Soriano, J. B., Kendrick, P. J., Paulson, K. R., Gupta, V., Abrams, E. M., Adedoyin, R. A., Adhikari, T. B., Advani, S. M., Agrawal, A., Ahmadian, E., Alahdab, F., Aljunid, S. M., Altirkawi, K. A., Alvis-Guzman, N., Anber, N. H., Andrei, C. L., Anjomshoa, M., Ansari, F., Antó, J. M., . . . Vos, T. (2020). Prevalence and attributable health burden of chronic respiratory diseases, 1990–2017: a systematic analysis for the Global Burden of Disease Study 2017. *The Lancet Respiratory Medicine*, *8*(6), 585–596. https://doi.org/10.1016/s2213-2600(20)30105-3