SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU

FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I

INFORMACIJSKIH TEHNOLOGIJA OSIJEK

Sveučilišni diplomski studij Računarstvo

ANALIZA POPULACIJE SVIJETA 1960.-2020.

Projekt

Lazar Volarev

Osijek, 2024. godina

# KV1 - Definiranje projektnog zadatka

## Projektni zadatak

Naziv zadatka: Populacija svijeta 1960.-2020. godine

Opis problema: Analizirati broj stanovnika na svjetskoj razini za odabrano razdoblje

Opis zadatka: Korištenjem tehnika čišćenja I vizualizacije podataka, ovaj projekt će istražiti broj stanovnika I prikazati promjenu tog broja kroz godine

Cilj projekta: Napraviti kartu svijeta koja pokazuje svaku državu i njihov broj stanovnika, te dodatne funkcionalnosti

Poveznica na git repozitorij projekta: https://github.com/lazarbb/World-population-1960.-2020.

## Podatci

* + 1. [*https://data.worldbank.org/indicator/SP.POP.TOTL*](https://data.worldbank.org/indicator/SP.POP.TOTL)Ovdje se nalaze podatci broja stanovnika za navedeno razdoblje.
    2. [*https://github.com/johan/world.geo.json/blob/master/countries.geo.json*](https://github.com/johan/world.geo.json/blob/master/countries.geo.json)JSON datoteka koja se koristi u pravljenu karte svijeta.

## Obrada podataka

* + 1. Podatci koji su bili u CSV formatu su prebačeni u JSON file. Podatci za Bermudu su izbačeni s karte svijeta jer koordinate za tu državu nisu bile točne.

## Relevantne vrste prikaza za korištene podatke

* + 1. Za potrebe projekta važno je koristiti vizualizacije koje će jasno prikazati   
       podatke.

# KV2 - Dizajn vizualizacije podataka.

## Pitanja na koja vizualizacija daje odgovor

* + 1. Kako se kretao broj stanovnika od 1960. do 2020. godine?
    2. Koliko svaka država ima stanovnika u kojoj godini?

## Skica vizualizacije podataka

A map of the world

Description automatically generated

A graph showing the growth of the year

Description automatically generated

A screenshot of a graph

Description automatically generated

## Postojeća rješenja i primjeri

* + 1. *Karta svijeta.* [*https://github.com/johan/world.geo.json/blob/master/countries.geo.json*](https://github.com/johan/world.geo.json/blob/master/countries.geo.json)
    2. Stranica koja sadrži vizualizaciju: https://data.worldbank.org/indicator/SP.POP.TOTL?view=map

## Prilagodba podataka

* + 1. *CSV dokument koji sadrži podatke o broju stanovnika je pretvoren u JSON format.*
    2. *Država Bermuda je izbačena iz prikaza, jer podatci za nju nisu valjani.*

## Boje i podatci

* + 1. Koristi se plava boja kako bi se pokazao broj stanovnika za određenu zemlju. Što je stanovnika više, intenzitet boje je veći i obrnuto.

# KV3 - Izrada prototipne vizualizacije podataka

## Osnovne funkcionalnosti i ponašanja

* + 1. Ključne funkcionalnosti su prikazivanje broja stanovnika na karti svijeta kroz godine od 1960. do 2020.
    2. Korisnik će dobiti pregled karte svijeta za određenu godinu i prelaskom miša preko određene države, dobit će broj stanovnika za tu državu u toj godini.

## Napredne funkcionalnosti i ponašanja:

* + 1. Usporedba dvije države u odabranoj godini bit će vizualizirana stupčastim grafom.
    2. Korisnik će izborom neke države dobiti linijski graf koji pokazuje kretanje broja stanovnika od 1960. do 2020. godine.

## Implementacija osnovnih funkcionalnosti

1. <div class="chart-container">
2. <h2>Population change from 1960 to 2020</h2>
3. <select id="country-select"></select>
4. <svg id="line-chart" width="800" height="500"></svg>
5. </div>
6. <script>
7. // Postavke za margine i dimenzije grafikona
8. const margin = { top: 20, right: 30, bottom: 50, left: 60 },
9. width = 800 - margin.left - margin.right,
10. height = 500 - margin.top - margin.bottom;
11. const svg = d3.select("#line-chart")
12. .attr("width", width + margin.left + margin.right)
13. .attr("height", height + margin.top + margin.bottom)
14. .append("g")
15. .attr("transform", `translate(${margin.left},${margin.top})`);
16. // Skale za X i Y os
17. const x = d3.scaleLinear()
18. .domain([1960, 2020])
19. .range([0, width]);
20. const y = d3.scaleLinear()
21. .range([height, 0]);
22. // Funkcija za dodavanje X i Y osi
23. const xAxis = svg.append("g")
24. .attr("transform", `translate(0,${height})`)
25. .call(d3.axisBottom(x).tickFormat(d3.format("d")));
26. const yAxis = svg.append("g");
27. // Tooltip
28. const tooltip = d3.select("body").append("div")
29. .attr("class", "tooltip");
30. // Dodaj X i Y oznake
31. svg.append("text")
32. .attr("class", "axis-label")
33. .attr("x", width / 2)
34. .attr("y", height + margin.bottom - 10)
35. .attr("text-anchor", "middle")
36. .text("Year");
37. svg.append("text")
38. .attr("class", "axis-label")
39. .attr("x", -height / 2)
40. .attr("y", -margin.left + 70)
41. .attr("text-anchor", "middle")
42. .attr("transform", "rotate(-90)")
43. .text("Population");
44. // Učitavanje podataka
45. Promise.all([
46. d3.json("countries.geo.json"),
47. d3.json("world\_population\_data.json")
48. ]).then(function(data) {
49. const geoData = data[0]; // Podaci o zemljama
50. const popData = data[1]; // Podaci o populaciji
51. // Popuni padajući izbornik zemljama
52. const countrySelect = d3.select("#country-select");
53. geoData.features.forEach(feature => {
54. countrySelect.append("option")
55. .text(feature.properties.name)
56. .attr("value", feature.id);
57. });
58. // Funkcija za ažuriranje grafikona
59. function updateChart(selectedCountry) {
60. const countryData = popData.find(d => d['Country Code'] === selectedCountry);
61. if (!countryData) {
62. alert("Population data not available for selected country.");
63. return;
64. }
65. const populationData = [];
66. for (let year = 1960; year <= 2020; year++) {
67. if (countryData[year]) {
68. populationData.push({ year: year, population: +countryData[year] });
69. }
70. }
71. // Ažuriranje Y skale
72. y.domain([0, d3.max(populationData, d => d.population)]);
73. yAxis.call(d3.axisLeft(y));
74. // Crtanje linije
75. const line = d3.line()
76. .x(d => x(d.year))
77. .y(d => y(d.population));
78. const path = svg.selectAll(".line")
79. .data([populationData]);
80. path.enter()
81. .append("path")
82. .attr("class", "line")
83. .merge(path)
84. .attr("d", line);
85. path.exit().remove();
86. // Dodaj krugove za svaku točku na liniji i interakciju s tooltip-om
87. const circles = svg.selectAll("circle")
88. .data(populationData);
89. circles.enter()
90. .append("circle")
91. .attr("r", 4)
92. .merge(circles)
93. .attr("cx", d => x(d.year))
94. .attr("cy", d => y(d.population))
95. .attr("fill", "steelblue")
96. .on("mouseover", function(event, d) {
97. tooltip.transition().duration(200).style("opacity", 0.9);
98. tooltip.html(`Year: ${d.year}<br>Population: ${d.population.toLocaleString()}`)
99. .style("left", (event.pageX + 5) + "px")
100. .style("top", (event.pageY - 28) + "px");
101. })
102. .on("mouseout", function() {
103. tooltip.transition().duration(500).style("opacity", 0);
104. });
105. circles.exit().remove();
106. }
107. // Dodaj funkciju za ažuriranje grafikona kad se promijeni zemlja
108. countrySelect.on("change", function() {
109. const selectedCountry = d3.select(this).property("value");
110. updateChart(selectedCountry);
111. });
112. // Inicijalni prikaz za prvu zemlju
113. updateChart(geoData.features[0].id);
114. }).catch(function(error) {
115. console.error("Error loading data:", error);
116. });
117. </script>
118. <script>
119. // Postavke za margine i dimenzije grafikona
120. const margin = { top: 20, right: 30, bottom: 50, left: 60 },
121. width = 800 - margin.left - margin.right,
122. height = 500 - margin.top - margin.bottom;
123. const svg = d3.select("#line-chart")
124. .attr("width", width + margin.left + margin.right)
125. .attr("height", height + margin.top + margin.bottom)
126. .append("g")
127. .attr("transform", `translate(${margin.left},${margin.top})`);
128. // Skale za X i Y os
129. const x = d3.scaleLinear()
130. .domain([1960, 2020])
131. .range([0, width]);
132. const y = d3.scaleLinear()
133. .range([height, 0]);
134. // Funkcija za dodavanje X i Y osi
135. const xAxis = svg.append("g")
136. .attr("transform", `translate(0,${height})`)
137. .call(d3.axisBottom(x).tickFormat(d3.format("d")));
138. const yAxis = svg.append("g");
139. // Tooltip
140. const tooltip = d3.select("body").append("div")
141. .attr("class", "tooltip");
142. // Dodaj X i Y oznake
143. svg.append("text")
144. .attr("class", "axis-label")
145. .attr("x", width / 2)
146. .attr("y", height + margin.bottom - 10)
147. .attr("text-anchor", "middle")
148. .text("Year");
149. svg.append("text")
150. .attr("class", "axis-label")
151. .attr("x", -height / 2)
152. .attr("y", -margin.left + 70)
153. .attr("text-anchor", "middle")
154. .attr("transform", "rotate(-90)")
155. .text("Population");
156. // Učitavanje podataka
157. Promise.all([
158. d3.json("countries.geo.json"),
159. d3.json("world\_population\_data.json")
160. ]).then(function(data) {
161. const geoData = data[0]; // Podaci o zemljama
162. const popData = data[1]; // Podaci o populaciji
163. // Popuni padajući izbornik zemljama
164. const countrySelect = d3.select("#country-select");
165. geoData.features.forEach(feature => {
166. countrySelect.append("option")
167. .text(feature.properties.name)
168. .attr("value", feature.id);
169. });
170. // Funkcija za ažuriranje grafikona
171. function updateChart(selectedCountry) {
172. const countryData = popData.find(d => d['Country Code'] === selectedCountry);
173. if (!countryData) {
174. alert("Population data not available for selected country.");
175. return;
176. }
177. const populationData = [];
178. for (let year = 1960; year <= 2020; year++) {
179. if (countryData[year]) {
180. populationData.push({ year: year, population: +countryData[year] });
181. }
182. }
183. // Ažuriranje Y skale
184. y.domain([0, d3.max(populationData, d => d.population)]);
185. yAxis.call(d3.axisLeft(y));
186. // Crtanje linije
187. const line = d3.line()
188. .x(d => x(d.year))
189. .y(d => y(d.population));
190. const path = svg.selectAll(".line")
191. .data([populationData]);
192. path.enter()
193. .append("path")
194. .attr("class", "line")
195. .merge(path)
196. .attr("d", line);
197. path.exit().remove();
198. // Dodaj krugove za svaku točku na liniji i interakciju s tooltip-om
199. const circles = svg.selectAll("circle")
200. .data(populationData);
201. circles.enter()
202. .append("circle")
203. .attr("r", 4)
204. .merge(circles)
205. .attr("cx", d => x(d.year))
206. .attr("cy", d => y(d.population))
207. .attr("fill", "steelblue")
208. .on("mouseover", function(event, d) {
209. tooltip.transition().duration(200).style("opacity", 0.9);
210. tooltip.html(`Year: ${d.year}<br>Population: ${d.population.toLocaleString()}`)
211. .style("left", (event.pageX + 5) + "px")
212. .style("top", (event.pageY - 28) + "px");
213. })
214. .on("mouseout", function() {
215. tooltip.transition().duration(500).style("opacity", 0);
216. });
217. circles.exit().remove();
218. }
219. // Dodaj funkciju za ažuriranje grafikona kad se promijeni zemlja
220. countrySelect.on("change", function() {
221. const selectedCountry = d3.select(this).property("value");
222. updateChart(selectedCountry);
223. });
224. // Inicijalni prikaz za prvu zemlju
225. updateChart(geoData.features[0].id);
226. }).catch(function(error) {
227. console.error("Error loading data:", error);
228. });
229. </script>

## Implementacija osnovnog ponašanja

* + 1. Osnovno ponašanje je prikazano u gornjem kodu. Prelaskom preko grafa, pojedine države na karti ili linijskog grafa, dobije se točan broj stanovnika za odabranu državu u odabranoj godini.
    2. Slike koje to potvrđuju

A map of the world with blue and white colors

Description automatically generated

A screenshot of a graph

Description automatically generated

# KV4 - Izrada konačne vizualizacije podataka

## Implementacija osnovnih funkcionalnosti

* + 1. Dodane su sve funkcionalnosti i rade kako treba.
    2. Testirati osnovne funkcionalnosti i osigurati da su ispravne, tj. da funkcioniraju na očekivani način. Potrebno potvrditi slikom.
    3. A map of the world

       Description automatically generated

## Implementacija osnovnog ponašanja

* + 1. Dovršiti implementaciju koda za nedostajuće osnovne interakcije.

1. <!DOCTYPE html>
2. <html lang="en">
3. <head>
4. <meta charset="UTF-8">
5. <title>World Population Map</title>
6. <script src="https://d3js.org/d3.v7.min.js"></script>
7. <style>
8. #map {
9. width: 100%;
10. height: 600px;
11. }
12. .tooltip {
13. position: absolute;
14. text-align: center;
15. padding: 4px;
16. background: lightgrey;
17. border: 0px;
18. border-radius: 8px;
19. pointer-events: none;
20. opacity: 0;
21. }
22. .slider-container {
23. position: absolute;
24. top: 20px;
25. left: 20px;
26. }
27. .legend-container {
28. display: flex;
29. justify-content: space-around;
30. margin-top: 20px;
31. width: 600px;
32. margin-left: auto;
33. margin-right: auto;
34. }
35. .legend-item {
36. text-align: center;
37. }
38. .color-box {
39. width: 30px;
40. height: 20px;
41. border: 1px solid #000;
42. }
43. </style>
44. </head>
45. <body>
46. <h1 style="text-align: center;">WORLD POPULATION MAP</h1>
47. <div class="slider-container">
48. <input type="range" id="year-slider" min="1960" max="2020" value="2000" step="1">
49. <span id="year-display">2000</span>
50. </div>
51. <div id="map"></div>
52. <div class="tooltip"></div>
53. <!-- Legend -->
54. <div class="legend-container">
55. <div class="legend-item">
56. <div class="color-box" style="background-color: #f7fbff;"></div>
57. <span>0 - 10M</span>
58. </div>
59. <div class="legend-item">
60. <div class="color-box" style="background-color: #deebf7;"></div>
61. <span>10M - 50M</span>
62. </div>
63. <div class="legend-item">
64. <div class="color-box" style="background-color: #c6dbef;"></div>
65. <span>50M - 100M</span>
66. </div>
67. <div class="legend-item">
68. <div class="color-box" style="background-color: #9ecae1;"></div>
69. <span>100M - 200M</span>
70. </div>
71. <div class="legend-item">
72. <div class="color-box" style="background-color: #6baed6;"></div>
73. <span>200M - 500M</span>
74. </div>
75. <div class="legend-item">
76. <div class="color-box" style="background-color: #4292c6;"></div>
77. <span>500M - 1B</span>
78. </div>
79. <div class="legend-item">
80. <div class="color-box" style="background-color: #2171b5;"></div>
81. <span>1B+</span>
82. </div>
83. </div>
84. <script>
85. const width = 960, height = 600;
86. const projection = d3.geoMercator()
87. .scale(150)
88. .translate([width / 2, height / 1.5]);
89. const path = d3.geoPath().projection(projection);
90. const svg = d3.select("#map").append("svg")
91. .attr("width", width)
92. .attr("height", height);
93. const tooltip = d3.select(".tooltip");
94. Promise.all([
95. d3.json("countries.geo.json"),
96. d3.json("world\_population\_data.json")
97. ]).then(function(data) {
98. const geoData = data[0];
99. const popData = data[1];
100. geoData.features = geoData.features.filter(feature => feature.properties.name !== "Bermuda");
101. const populationByCountry = {};
102. popData.forEach(d => {
103. populationByCountry[d['Country Code']] = d;
104. });
105. const getColor = (countryCode, year) => {
106. if (populationByCountry[countryCode] && populationByCountry[countryCode][year]) {
107. const population = populationByCountry[countryCode][year];
108. return d3.scaleThreshold()
109. .domain([10000000, 50000000, 100000000, 200000000, 500000000, 1000000000])
110. .range(["#f7fbff", "#deebf7", "#c6dbef", "#9ecae1", "#6baed6", "#4292c6", "#2171b5"])
111. (population);
112. }
113. return "#cccccc";
114. };
115. let currentYear = "2000";
116. svg.selectAll("path")
117. .data(geoData.features)
118. .enter()
119. .append("path")
120. .attr("d", path)
121. .attr("fill", d => getColor(d.id, currentYear))
122. .on("mouseover", function(event, d) {
123. tooltip.transition().duration(200).style("opacity", .9);
124. tooltip.html(d.properties.name + "<br/>" + "Population: " +
125. (populationByCountry[d.id][currentYear] || "No data"))
126. .style("left", (event.pageX + 10) + "px")
127. .style("top", (event.pageY - 28) + "px");
128. })
129. .on("mouseout", function() {
130. tooltip.transition().duration(500).style("opacity", 0);
131. });
132. const slider = d3.select("#year-slider")
133. .on("input", function() {
134. currentYear = this.value;
135. d3.select("#year-display").text(currentYear);
136. svg.selectAll("path")
137. .attr("fill", d => getColor(d.id, currentYear));
138. });
139. });
140. </script>
141. </body>
142. </html>

## Implementacija naprednih funkcionalnosti

1. <div class="chart-container">
2. <h2>Population change from 1960 to 2020</h2>
3. <select id="country-select"></select>
4. <svg id="line-chart" width="800" height="500"></svg>
5. </div>
6. <script>
7. // Postavke za margine i dimenzije grafikona
8. const margin = { top: 20, right: 30, bottom: 50, left: 60 },
9. width = 800 - margin.left - margin.right,
10. height = 500 - margin.top - margin.bottom;
11. const svg = d3.select("#line-chart")
12. .attr("width", width + margin.left + margin.right)
13. .attr("height", height + margin.top + margin.bottom)
14. .append("g")
15. .attr("transform", `translate(${margin.left},${margin.top})`);
16. // Skale za X i Y os
17. const x = d3.scaleLinear()
18. .domain([1960, 2020])
19. .range([0, width]);
20. const y = d3.scaleLinear()
21. .range([height, 0]);
22. // Funkcija za dodavanje X i Y osi
23. const xAxis = svg.append("g")
24. .attr("transform", `translate(0,${height})`)
25. .call(d3.axisBottom(x).tickFormat(d3.format("d")));
26. const yAxis = svg.append("g");
27. // Tooltip
28. const tooltip = d3.select("body").append("div")
29. .attr("class", "tooltip");
30. // Dodaj X i Y oznake
31. svg.append("text")
32. .attr("class", "axis-label")
33. .attr("x", width / 2)
34. .attr("y", height + margin.bottom - 10)
35. .attr("text-anchor", "middle")
36. .text("Year");
37. svg.append("text")
38. .attr("class", "axis-label")
39. .attr("x", -height / 2)
40. .attr("y", -margin.left + 70)
41. .attr("text-anchor", "middle")
42. .attr("transform", "rotate(-90)")
43. .text("Population");
44. // Učitavanje podataka
45. Promise.all([
46. d3.json("countries.geo.json"),
47. d3.json("world\_population\_data.json")
48. ]).then(function(data) {
49. const geoData = data[0]; // Podaci o zemljama
50. const popData = data[1]; // Podaci o populaciji
51. // Popuni padajući izbornik zemljama
52. const countrySelect = d3.select("#country-select");
53. geoData.features.forEach(feature => {
54. countrySelect.append("option")
55. .text(feature.properties.name)
56. .attr("value", feature.id);
57. });
58. // Funkcija za ažuriranje grafikona
59. function updateChart(selectedCountry) {
60. const countryData = popData.find(d => d['Country Code'] === selectedCountry);
61. if (!countryData) {
62. alert("Population data not available for selected country.");
63. return;
64. }
65. const populationData = [];
66. for (let year = 1960; year <= 2020; year++) {
67. if (countryData[year]) {
68. populationData.push({ year: year, population: +countryData[year] });
69. }
70. }
71. // Ažuriranje Y skale
72. y.domain([0, d3.max(populationData, d => d.population)]);
73. yAxis.call(d3.axisLeft(y));
74. // Crtanje linije
75. const line = d3.line()
76. .x(d => x(d.year))
77. .y(d => y(d.population));
78. const path = svg.selectAll(".line")
79. .data([populationData]);
80. path.enter()
81. .append("path")
82. .attr("class", "line")
83. .merge(path)
84. .attr("d", line);
85. path.exit().remove();
86. // Dodaj krugove za svaku točku na liniji i interakciju s tooltip-om
87. const circles = svg.selectAll("circle")
88. .data(populationData);
89. circles.enter()
90. .append("circle")
91. .attr("r", 4)
92. .merge(circles)
93. .attr("cx", d => x(d.year))
94. .attr("cy", d => y(d.population))
95. .attr("fill", "steelblue")
96. .on("mouseover", function(event, d) {
97. tooltip.transition().duration(200).style("opacity", 0.9);
98. tooltip.html(`Year: ${d.year}<br>Population: ${d.population.toLocaleString()}`)
99. .style("left", (event.pageX + 5) + "px")
100. .style("top", (event.pageY - 28) + "px");
101. })
102. .on("mouseout", function() {
103. tooltip.transition().duration(500).style("opacity", 0);
104. });
105. circles.exit().remove();
106. }
107. // Dodaj funkciju za ažuriranje grafikona kad se promijeni zemlja
108. countrySelect.on("change", function() {
109. const selectedCountry = d3.select(this).property("value");
110. updateChart(selectedCountry);
111. });
112. // Inicijalni prikaz za prvu zemlju
113. updateChart(geoData.features[0].id);
114. }).catch(function(error) {
115. console.error("Error loading data:", error);
116. });
117. </script>
118. <script>
119. // Postavke za margine i dimenzije grafikona
120. const margin = { top: 20, right: 30, bottom: 50, left: 60 },
121. width = 800 - margin.left - margin.right,
122. height = 500 - margin.top - margin.bottom;
123. const svg = d3.select("#line-chart")
124. .attr("width", width + margin.left + margin.right)
125. .attr("height", height + margin.top + margin.bottom)
126. .append("g")
127. .attr("transform", `translate(${margin.left},${margin.top})`);
128. // Skale za X i Y os
129. const x = d3.scaleLinear()
130. .domain([1960, 2020])
131. .range([0, width]);
132. const y = d3.scaleLinear()
133. .range([height, 0]);
134. // Funkcija za dodavanje X i Y osi
135. const xAxis = svg.append("g")
136. .attr("transform", `translate(0,${height})`)
137. .call(d3.axisBottom(x).tickFormat(d3.format("d")));
138. const yAxis = svg.append("g");
139. // Tooltip
140. const tooltip = d3.select("body").append("div")
141. .attr("class", "tooltip");
142. // Dodaj X i Y oznake
143. svg.append("text")
144. .attr("class", "axis-label")
145. .attr("x", width / 2)
146. .attr("y", height + margin.bottom - 10)
147. .attr("text-anchor", "middle")
148. .text("Year");
149. svg.append("text")
150. .attr("class", "axis-label")
151. .attr("x", -height / 2)
152. .attr("y", -margin.left + 70)
153. .attr("text-anchor", "middle")
154. .attr("transform", "rotate(-90)")
155. .text("Population");
156. // Učitavanje podataka
157. Promise.all([
158. d3.json("countries.geo.json"),
159. d3.json("world\_population\_data.json")
160. ]).then(function(data) {
161. const geoData = data[0]; // Podaci o zemljama
162. const popData = data[1]; // Podaci o populaciji
163. // Popuni padajući izbornik zemljama
164. const countrySelect = d3.select("#country-select");
165. geoData.features.forEach(feature => {
166. countrySelect.append("option")
167. .text(feature.properties.name)
168. .attr("value", feature.id);
169. });
170. // Funkcija za ažuriranje grafikona
171. function updateChart(selectedCountry) {
172. const countryData = popData.find(d => d['Country Code'] === selectedCountry);
173. if (!countryData) {
174. alert("Population data not available for selected country.");
175. return;
176. }
177. const populationData = [];
178. for (let year = 1960; year <= 2020; year++) {
179. if (countryData[year]) {
180. populationData.push({ year: year, population: +countryData[year] });
181. }
182. }
183. // Ažuriranje Y skale
184. y.domain([0, d3.max(populationData, d => d.population)]);
185. yAxis.call(d3.axisLeft(y));
186. // Crtanje linije
187. const line = d3.line()
188. .x(d => x(d.year))
189. .y(d => y(d.population));
190. const path = svg.selectAll(".line")
191. .data([populationData]);
192. path.enter()
193. .append("path")
194. .attr("class", "line")
195. .merge(path)
196. .attr("d", line);
197. path.exit().remove();
198. // Dodaj krugove za svaku točku na liniji i interakciju s tooltip-om
199. const circles = svg.selectAll("circle")
200. .data(populationData);
201. circles.enter()
202. .append("circle")
203. .attr("r", 4)
204. .merge(circles)
205. .attr("cx", d => x(d.year))
206. .attr("cy", d => y(d.population))
207. .attr("fill", "steelblue")
208. .on("mouseover", function(event, d) {
209. tooltip.transition().duration(200).style("opacity", 0.9);
210. tooltip.html(`Year: ${d.year}<br>Population: ${d.population.toLocaleString()}`)
211. .style("left", (event.pageX + 5) + "px")
212. .style("top", (event.pageY - 28) + "px");
213. })
214. .on("mouseout", function() {
215. tooltip.transition().duration(500).style("opacity", 0);
216. });
217. circles.exit().remove();
218. }
219. // Dodaj funkciju za ažuriranje grafikona kad se promijeni zemlja
220. countrySelect.on("change", function() {
221. const selectedCountry = d3.select(this).property("value");
222. updateChart(selectedCountry);
223. });
224. // Inicijalni prikaz za prvu zemlju
225. updateChart(geoData.features[0].id);
226. }).catch(function(error) {
227. console.error("Error loading data:", error);
228. });
229. </script>
     * 1. A graph showing the growth of the year

          Description automatically generated

## Implementacija naprednog ponašanja

A graph showing the growth of the population

Description automatically generated

# KV5 - Dovršetak projektnog zadatka i pisanje dokumentacije

## Eventualne preinake i dorade rješenja - u dogovoru s nastavnikom

Nije bilo nikakvih promjena.

## Izrada dokumenta - projektne dokumentacije

* + 1. Hijerarhija projekta.

Projekt se sastoji od karte svijeta, stupčastog grafa i linijskog grafa.

* + 1. Popis korištenih tehnologija, bez opisa.

Korišten je HTML, točnije D3 biblioteka.

* + 1. Upute za postavljanje.

Potrebno je pokrenuti server, jer se html datoteka ne može samo tako otvoriti zbog CORS ograničenja.

* + 1. Upute za korištenje.

Na početnom ekranu se otvara karta svijeta. Korisnik pomoću kliznika bira godinu za koju želi prikaz popisa broja stanovnika. Prelaskom u /comparison korisnik može odabrati godinu u dvije države za usporedbu, tada mu se ta razlika vizualizira pomoću stupčastog grafa. Prelaskom u /country-chart i odabirom države korisnik dobije linijski graf s prikazom broja stanovnika u razdoblju od 1960. do 2020. godine.