SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU

Fakultet elektrotehnike, računarstva i informacijskih tehnologija Osijek

Projektni zadatak iz predmeta

VIZUALIZACIJA PODATAKA

Vizualizacija parametara razvoja za europske zemlje u 2022. godini

Student:[Karlo Kokanović. DRD

Mentor: Barbara Bilonić

U Osijeku, lipanj 2023. godine

SADRŽAJ

[**1. KV1 - Definiranje projektnog zadatka 3**](#_4i6s1smptslt)

[1.1. Projektni zadatak 3](#_wgoiizki7jnh)

[1.2. Podatci 3](#_jjnsdui0w5uv)

[1.3. Obrada podataka 3](#_jfu6ybs7kwrw)

[1.4. Relevantne vrste prikaza za korištene podatke 3](#_1g4rlc1z1va6)

[**2. KV2 - Dizajn vizualizacije podataka. 4**](#_smxrkz6d6wy8)

[2.1. Pitanja na koja vizualizacija daje odgovor 4](#_cqmiylhgneug)

[2.2. Skica vizualizacije podataka 4](#_i61spu8qyxmd)

[2.3. Postojeća rješenja i primjeri 4](#_qmiyijy4jsp8)

[2.4. Prilagodba podataka 4](#_cch1vltulvr)

[2.5. Boje i podatci 4](#_w22k3frzbtb2)

[**3. KV3 - Izrada prototipne vizualizacije podataka 5**](#_k5q372z1w040)

[3.1. Osnovne funkcionalnosti i ponašanja 5](#_yaexlmqqdt0u)

[3.2. Napredne funkcionalnosti i ponašanja: 5](#_y300tbmoybm5)

[3.3. Implementacija osnovnih funkcionalnosti 5](#_g3q3h1rhqzyz)

[3.4. Implementacija osnovnog ponašanja 5](#_itz19pnh67g0)

[**4. KV4 - Izrada konačne vizualizacije podataka 6**](#_sj4kshcbtchw)

[4.1. Implementacija osnovnih funkcionalnosti 6](#_dc19rzmsn9k5)

[4.2. Implementacija osnovnog ponašanja 6](#_exvg8zl4eo3w)

[**5. KV5 - Dovršetak projektnog zadatka i pisanje dokumentacije 7**](#_woao7m26bf75)

[5.1. Izrada dokumenta - projektne dokumentacije 7](#_6zplla712qe7)

[**Literatura 8**](#_ahmk8vkny1f9)

[**Prilog I 9**](#_jefyptb5hazf)

# KV1 - Definiranje projektnog zadatka

## Projektni zadatak

Naziv zadatka: Vizualizacija parametara razvoja za europske zemlje u 2022. godini

Opis problema: Problem kojim se bavi ovaj projekt je vizualizacija razvojnih parametara za europske zemlje u 2022 godini. Konkretni parametar koji se vizualizira je ukupna stopa nezaposlenosti za radno sposobne pojedince u dobi od 25 do 74 godine. Cilj projekta je pružiti interaktivni alat za vizualizaciju podataka koji omogućuje korisnicima istraživanje i usporedbu ukupnih stopa nezaposlenosti u različitim europskim zemljama. Također, korisnici imaju na uvid ukupnu stopu uvoza i izvoza pojedine članice.

Opis zadatka: Zadatak je razviti web aplikaciju za vizualizaciju podataka koja prikazuje kartu europskih zemalja i predstavlja ukupnu stopu nezaposlenosti za svaku zemlju. Aplikacija bi korisnicima trebala omogućiti odabir određenog mjeseca i dinamičko ažuriranje karte kako bi se prikazale ukupne stope nezaposlenosti za odabrani mjesec. Osim toga, aplikacija bi trebala pružiti interaktivne grafikone, uključujući kružni dijagram i linijski grafikon, za daljnju analizu udjela spola u ukupnoj nezaposlenosti i podatke o uvozu/izvozu za odabrane zemlje.

Cilj projekta: Cilj projekta je stvoriti interaktivni i vizualno privlačan alat za vizualizaciju podataka koji korisnicima omogućuje istraživanje i razumijevanje ukupnih stopa nezaposlenosti i drugih razvojnih parametara za europske zemlje 2022. Vizualizacijom podataka na intuitivan i interaktivan način, projekt namjerava poboljšati razumijevanje obrazaca nezaposlenosti i potaknuti dublje istraživanje čimbenika koji doprinose nezaposlenosti u Europi.

Poveznica na git repozitorij projekta:

## Podatci

Izvori podataka: [https://www.econdb.com/dataset/UNE\_RT\_M/?AGE=[Y25-74]&FREQ=[M]&GEO=[HR,AT,BE,BG,HR,CY,CZ,DK,EE,FI,FR,DE,EL,HU,IS,IE,IT,JP,LV,LT,LU,MT,NL,NO,PL,PT,RO,SK,SI,ES,SE,CH,TR,UK,US]&SEX=[F]&S\_ADJ=[SA]&UNIT=[PC\_ACT]&from=2018-01-01&h=TIME&mode=Table&to=2023-01-01&v=Geopolitical+entity+%28reporting%29](https://www.econdb.com/dataset/UNE_RT_M/?AGE=%5bY25-74%5d&FREQ=%5bM%5d&GEO=%5bHR,AT,BE,BG,HR,CY,CZ,DK,EE,FI,FR,DE,EL,HU,IS,IE,IT,JP,LV,LT,LU,MT,NL,NO,PL,PT,RO,SK,SI,ES,SE,CH,TR,UK,US%5d&SEX=%5bF%5d&S_ADJ=%5bSA%5d&UNIT=%5bPC_ACT%5d&from=2018-01-01&h=TIME&mode=Table&to=2023-01-01&v=Geopolitical+entity+%28reporting%29)

* Eurostat podatci sa stopama nezaposlenosti

[https://www.econdb.com/dataset/EI\_ETEA\_M/?FREQ=[M]&GEO=[AT,BE,BG,HR,CY,CZ,DK,EE,EA20,FI,FR,DE,EL,HU,IE,IT,LV,LT,LU,MT,NL,PL,PT,RO,SK,SI,ES,SE]&INDIC=[ET-T,ET-INTER,ET-CONS,ET-CAP]&PARTNER=[]&STK\_FLOW=[EXP,IMP]&UNIT=[MIO-EUR-NSA]&from=2020-08-01&h=TIME&mode=Table&to=2023-02-01&v=Geopolitical+entity+%28reporting%29](https://www.econdb.com/dataset/EI_ETEA_M/?FREQ=%5bM%5d&GEO=%5bAT,BE,BG,HR,CY,CZ,DK,EE,EA20,FI,FR,DE,EL,HU,IE,IT,LV,LT,LU,MT,NL,PL,PT,RO,SK,SI,ES,SE%5d&INDIC=%5bET-T,ET-INTER,ET-CONS,ET-CAP%5d&PARTNER=%5b%5d&STK_FLOW=%5bEXP,IMP%5d&UNIT=%5bMIO-EUR-NSA%5d&from=2020-08-01&h=TIME&mode=Table&to=2023-02-01&v=Geopolitical+entity+%28reporting%29)

* Eurostat podatci o izvozu i uvozu

<https://github.com/leakyMirror/map-of-europe/blob/master/GeoJSON/europe.geojson> - mapa Europe

U projektu će se vizualizirati parametri ukupne nezaposlenosti, te udjela po spolu za ukupnu nezaposlenost. Promatrana dobna granica je od 25 do 74 godina starosti. Ubraja se radno sposobno stanovništvo. Također, dobije se na uvid i trend rasta/pada novčane vrijednosti izvoza kao i uvoza. Podatci su dostupni za svih 27 članica Europske Unije, kao i Švicarsku, Island i Norvešku. Za države koje nisu članice EU nisu javno dostupni, putem Eurostata, podatci o izvozu i uvozu.

## Obrada podataka

## Odabrani period promatranja obuhvaća svih 12 mjeseci 2022. godine u cijelosti. Podatci su direktno preuzimani u .csv formatu. Za bolje formatiranje podataka u izvozu/uvozu korišten je programski jezik Python i skripta dostupna na git repozitoriju. Ona samo izvlači potrebne podatke, u prvi red upisuje sve države odvojene zarezom, pa svaki sljedeći predstavlja vrijednost u milijunima eura kako izvoza tako i uvoza u drugoj datoteci. Dok, prvi .csv je obrađen djelomično ručno zbog jednostavnosti. Formatiran je tako da prvi red sadrži izlistane potrebite parametre, odvojene zarezom. A to su: *„country,month,female\_unemployment,male\_unemployment,total\_unemployment“.* Oni, redom, predstavljaju državu, mjesec u godini koji se promatra, te stope nezaposlenosti, žensku, mušku kao i ukupnu. Svaki sljedeći red je popunjen podatcima koji slijede takav niz.

## Relevantne vrste prikaza za korištene podatke

Kružni dijagram za prikaz udjela spola unutar podatka o ukupnoj stopi nezaposlenosti. Linijski graf za prikaz trenda izvoza i uvoza. Naravno, velika mapa sa svim državama obojanim na osnovi dostupnih podataka.

# KV2 - Dizajn vizualizacije podataka.

## Pitanja na koja vizualizacija daje odgovor

[Navesti i opisati na koja pitanja će vizualizacija pružati odgovor.

*Precizirati pitanja na koja se odgovara vizualizacijom podataka. Potrebno je osigurati da su pitanja jasno formulirana i da se mogu odgovoriti na temelju dostupnih podataka.*]

Koje su varijacije u ukupnim stopama nezaposlenosti u europskim zemljama u 2022.? Na ovo pitanje može se odgovoriti analizom karte koja prikazuje ukupne stope nezaposlenosti za svaku zemlju koristeći nijanse boja.

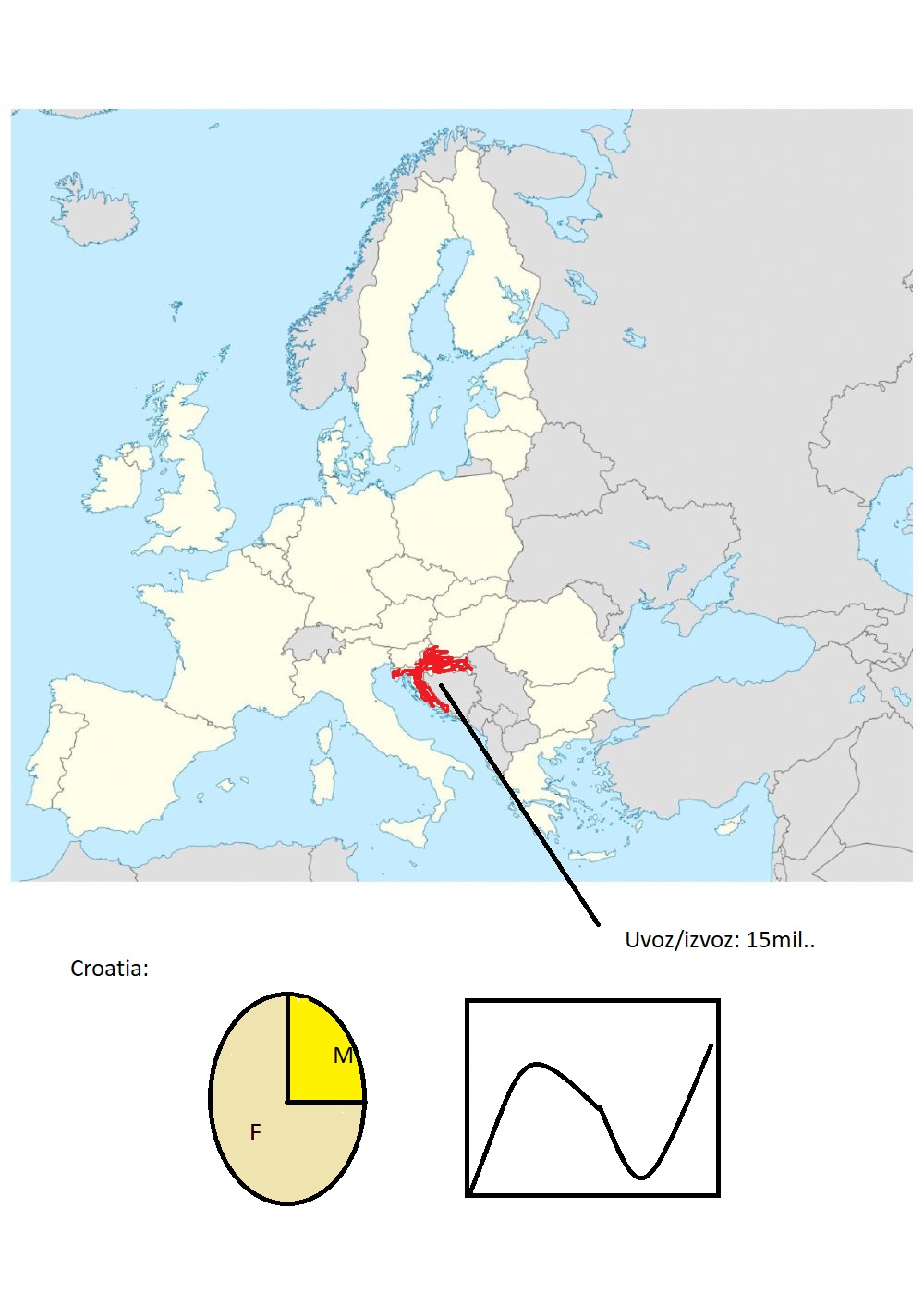
Kako se udio nezaposlenosti razlikuje po spolu za odabranu zemlju i mjesec u 2022.? Kružni dijagram pomaže odgovoriti na ovo pitanje prikazujući postotak nezaposlenosti muškaraca i žena u odabranoj zemlji i mjesecu.

Kako se uvoz i izvoz određene zemlje mijenjaju tijekom vremena u 2022.? Linijski grafikon prikazuje podatke o uvozu i izvozu po mjesecima u godini, omogućujući korisnicima da promatraju trendove i promjene trgovinske bilance.

Koje zemlje imaju najviše i najniže stope nezaposlenosti u Europi 2022.? Proučavajući vizualizaciju karte, korisnici mogu prepoznati zemlje s tamnijim nijansama crvene koje označavaju više stope nezaposlenosti i svjetlijim nijansama koje označavaju niže stope.

Postoje li vidljivi uzorci ili trendovi u trgovinskoj bilanci određene zemlje u 2022.? Linijski grafikon koji prikazuje uvoz i izvoz tijekom vremena može otkriti obrasce kao što su trgovinski deficiti ili suficiti, omogućujući korisnicima analizu trgovinskog učinka zemlje.

## Skica vizualizacije podataka

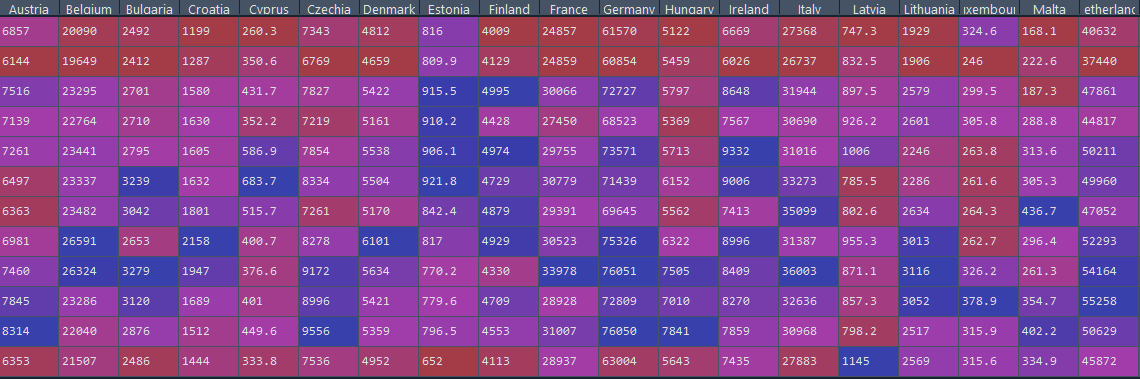


## Postojeća rješenja i primjeri

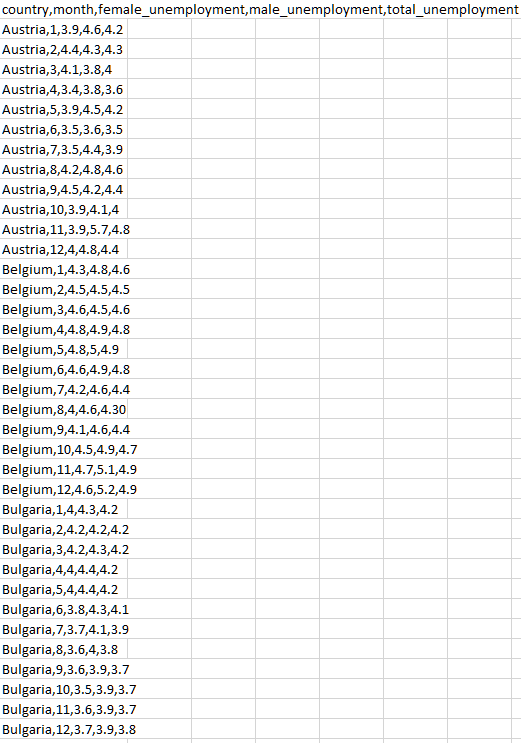
<https://eurostat.github.io/eurostat-map.js/examples/pop-unemploy-bivariate.html> - idejno rješenje slične problematike

<https://data.europa.eu/en/academy/going-beyond-bars-and-lines-effective-non-standard-data-visualisation> - priručnik za osnove vizualizacije podataka

## Prilagodba podataka

Ranije u poglavlju 1.3 je pobliže opisan i opis same prilagodbe podataka. Nakon svega, podatci o uvozu izgledaju ovako: 

Ista stvar se ponavlja i za izvoz, za sve dostupne članice. Podatci o nezaposlenosti, pak, ovako izgledaju:



## Boje i podatci

Sama karta je gradirana u nijansama crvene boje. Odabrao sam crvenu zbog asocijacije za loše/alarmantno stanje, pa stoga što država ima veću stopu ukupne nezaposlenosti, to će biti jačom nijansom obojana. Kružni grafikon ima dvije boje, radi lakšeg razlikovanja, plavu za muški udio i narančastu za ženski. Linijski graf koristi crvenu i plavu kako bi jasno znali što predstavlja uvoz, a što izvoz.

# KV3 - Izrada prototipne vizualizacije podataka

## Osnovne funkcionalnosti i ponašanja

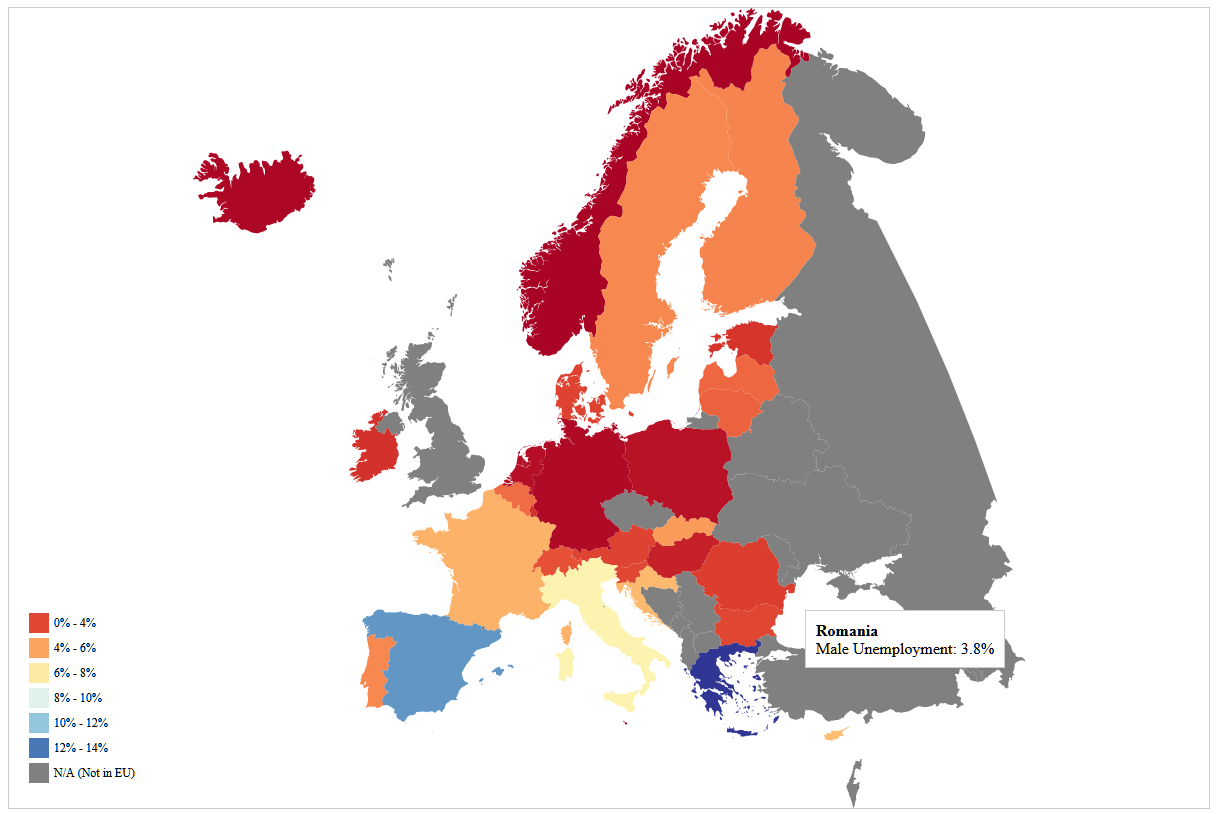
Kada korisnici prijeđu pokazivačem iznad određene zemlje na karti, mogu se prikazati dodatne informacije o toj zemlji, poput stope nezaposlenosti. Korisnici mogu kliknuti zemlju kako bi je odabrali i pogledali detaljnije informacije Pokazuje se onda kružni dijagram koji prikazuje dodatne informacije poput spola (muški ili ženski) i odgovarajućeg postotka nezaposlenosti: Ako korisnici zadrže pokazivač miša iznad podatkovne točke na linijskom grafikonu, mogu se prikazati odgovarajuće vrijednosti uvoza/izvoza. Ova interakcija omogućuje korisnicima da se usredotoče na određene podatkovne točke i usporede različite zemlje i vremenska razdoblja. Opis I legenda se koristi za pružanje dodatnih kontekstualnih informacija kada korisnici zadrže pokazivač iznad određenih elemenata ili stupe u interakciju s vizualizacijom. U padajućem izborniku se bira mjesec za promatranje.

## Napredne funkcionalnosti i ponašanja:

Kombinacija karte, kružnog dijagrama i linijski grafikona u jedan prikaz. Karta se prikazuje unutar div spremnika karte, dok se kružni i linijski grafikon prikazuju unutar diva. Div spremnik karte sadrži SVG element s ID-om "map", koji predstavlja vizualizaciju karte. Također uključuje opis za prikaz dodatnih informacija i legende za pružanje vizualnog vodiča za podatke karte. Svaki spremnik grafikona ima naslov. Mogućnost dinamičkog ažuriranja vizualizacija na temelju odabranih podataka. Ažuriranje kružnog dijagrama i karte na temelju odabranog mjeseca, što znači da se prikaz transformira kako bi se prikazali relevantni podaci za odabrani mjesec. Interaktivni elementi koji mogu utjecati ili biti pod utjecajem drugih grafikona, konkretno, odabir mjeseca i na temelju tog odabira ažurirate kartu I kružni dijagram. Upotreba tranzicije za ažuriranje mape, kružnog dijagrama I linijskog grafa , pa tako i naslova.

## Implementacija osnovnih funkcionalnosti

1. <!DOCTYPE html>
2. <html>
3. <head>
4. <meta charset="utf-8">
5. <meta name="viewport" content="width=device-width, initial-scale=1">
6. <title>VP Projekt</title>
7. <style>
8. #map-container {
9. position: relative;
10. width: 1200px;
11. height: 800px;
12. border: 1px solid #ccc;
13. }
14. #tooltip {
15. position: absolute;
16. background-color: #fff;
17. padding: 10px;
18. border: 1px solid #ccc;
19. visibility: hidden;
20. }
21. .legend {
22. position: absolute;
23. left: 20px;
24. bottom: 20px;
25. display: flex;
26. flex-direction: column;
27. justify-content: center;
28. align-items: flex-start;
29. font-size: 12px;
30. }
31. .legend-item {
32. display: flex;
33. align-items: center;
34. margin-bottom: 5px;
35. }
36. .legend-color {
37. width: 20px;
38. height: 20px;
39. margin-right: 5px;
40. }
41. </style>
42. </head>
43. <body>
44. <div id="map-container">
45. <svg id="map"></svg>
46. <div id="tooltip"></div>
47. <div class="legend"></div>
48. </div>
49. <script src="https://d3js.org/d3.v6.min.js"></script>
50. <script>
51. d3.csv("unemployment\_data.csv").then(function(data) {
52. const width = 1200;
53. const height = 800;
54. const svg = d3.select("#map")
55. .attr("width", width)
56. .attr("height", height);
57. const maleUnemploymentRates = data.map(d => +d.male\_unemployment);
58. const maxRate = d3.max(maleUnemploymentRates);
59. const minRate = d3.min(maleUnemploymentRates);
60. const colorScale = d3.scaleSequential()
61. .domain([minRate, maxRate])
62. .interpolator(d3.interpolateRdYlBu);
63. d3.json("europe.geojson").then(function(geojson) {
64. const projection = d3.geoMercator()
65. .fitSize([width, height], geojson);
66. const pathGenerator = d3.geoPath()
67. .projection(projection);
68. const handleMouseOver = function(event, d) {
69. const countryData = data.find(country => country.country === d.properties.NAME);
70. const maleUnemploymentRate = countryData ? +countryData.male\_unemployment : null;
71. const tooltip = d3.select("#tooltip");
72. tooltip.style("visibility", "visible")
73. .html(`<strong>${d.properties.NAME}</strong><br>
74. Male Unemployment: ${maleUnemploymentRate !== null ? maleUnemploymentRate + '%' : 'N/A (Not in EU)'}<br>`)
75. .style("left", (event.pageX + 10) + "px")
76. .style("top", (event.pageY + 10) + "px");
77. };
78. const handleMouseOut = function() {
79. d3.select("#tooltip").style("visibility", "hidden");
80. };
81. svg.selectAll("path")
82. .data(geojson.features)
83. .enter()
84. .append("path")
85. .attr("d", pathGenerator)
86. .attr("fill", function(d) {
87. const countryData = data.find(country => country.country === d.properties.NAME);
88. const unemploymentRate = countryData ? +countryData.male\_unemployment : null;
89. return unemploymentRate !== null ? colorScale(unemploymentRate) : "#808080";
90. })
91. .on("mouseover", function(event, d) {
92. handleMouseOver(event, d);
93. })
94. .on("mouseout", function() {
95. handleMouseOut();
96. });
97. const legendData = colorScale.ticks(5).map(d => Math.round(d));
98. const legendColors = legendData.map(d => colorScale(d));
99. const legend = d3.select(".legend")
100. .selectAll(".legend-item")
101. .data(legendData)
102. .enter()
103. .append("div")
104. .attr("class", "legend-item");
105. legend.append("div")
106. .attr("class", "legend-color")
107. .style("background-color", (\_, i) => legendColors[i]);
108. legend.append("span")
109. .text((d, i) => {
110. const lowerBound = i > 0 ? legendData[i - 1] : 0;
111. const upperBound = d;
112. return `${lowerBound}% - ${upperBound}%`;
113. });
114. const grayLegendItem = d3.select(".legend")
115. .append("div")
116. .attr("class", "legend-item");
117. grayLegendItem.append("div")
118. .attr("class", "legend-color")
119. .style("background-color", "#808080");
120. grayLegendItem.append("span")
121. .text("N/A (Not in EU)");
122. });
123. });
124. </script>
125. </body>
126. </html>



## Implementacija osnovnog ponašanja

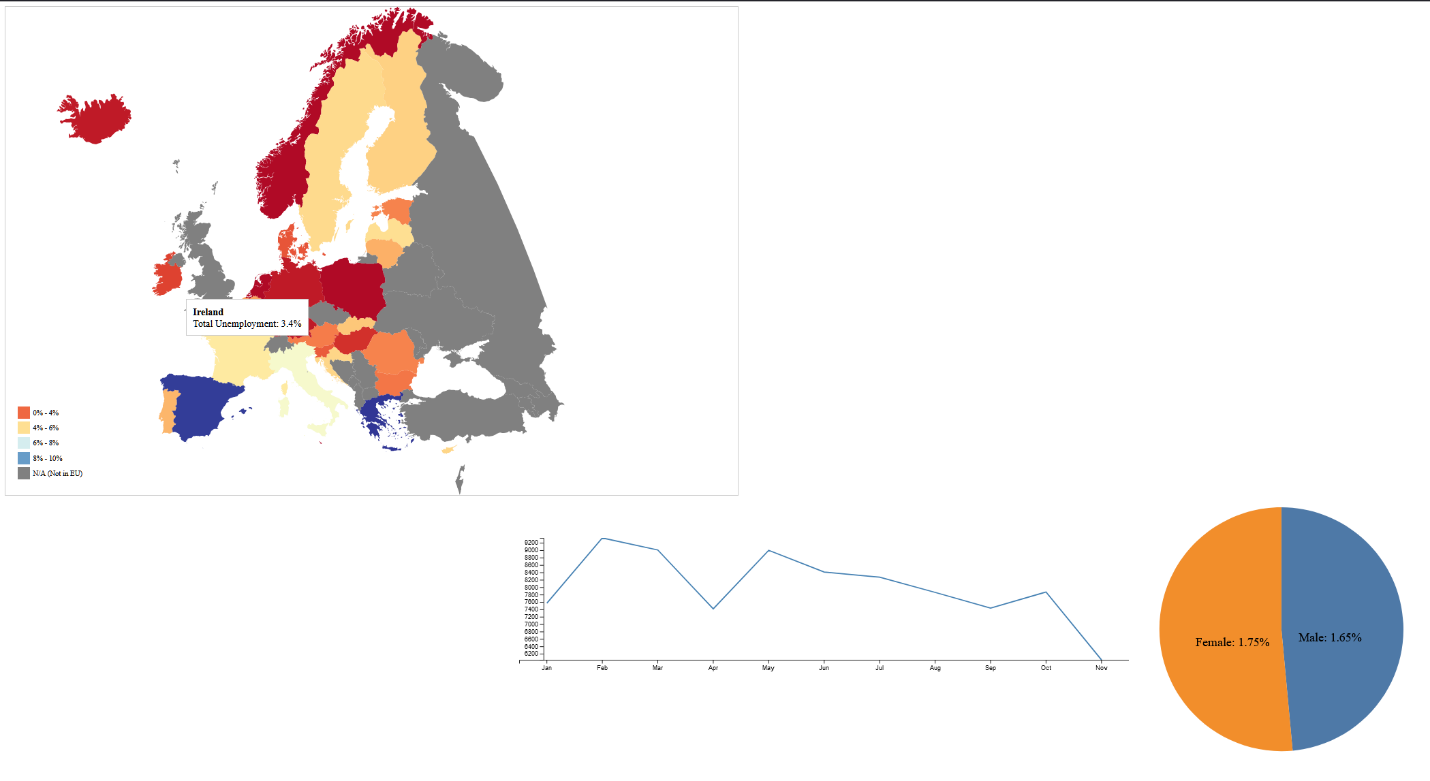
Implementirana je karta Europe, na uvid imamo podatke samo za prvi mjesec godine, vidimo kako su uspješno vizualizirani. Ubačena je legenda da znamo u što gledamo. Zasad nisu implementirani dijagram niti graf.

# KV4 - Izrada konačne vizualizacije podataka

## Implementacija osnovnih funkcionalnosti

1. <!DOCTYPE html>
2. <html>
3. <head>
4. <meta charset="utf-8">
5. <meta name="viewport" content="width=device-width, initial-scale=1">
6. <title>VP Projekt</title>
7. <style>
8. #map-container {
9. position: relative;
10. width: 1200px;
11. height: 800px;
12. border: 1px solid #ccc;
13. }
14. #tooltip {
15. position: absolute;
16. background-color: #fff;
17. padding: 10px;
18. border: 1px solid #ccc;
19. visibility: hidden;
20. }
21. .legend {
22. position: absolute;
23. left: 20px;
24. bottom: 20px;
25. display: flex;
26. flex-direction: column;
27. justify-content: center;
28. align-items: flex-start;
29. font-size: 12px;
30. }
31. .legend-item {
32. display: flex;
33. align-items: center;
34. margin-bottom: 5px;
35. }
36. .legend-color {
37. width: 20px;
38. height: 20px;
39. margin-right: 5px;
40. }
42. #pie-chart-container {
43. margin-top: 20px;
44. display: flex;
45. justify-content: center;
46. }
47. #chart-container {
48. display: flex;
49. }
50. #line-chart-container {
51. flex: 1;
52. display: flex;
53. justify-content: flex-end;
54. align-items: center;
55. }
56. .line {
57. fill: none;
58. stroke: steelblue;
59. stroke-width: 2px;
60. }
61. </style>
62. </head>
63. <body>
64. <div id="map-container">
65. <svg id="map"></svg>
66. <div id="tooltip"></div>
67. <div class="legend"></div>
68. </div>
69. <div id="chart-container">
70. <div id="line-chart-container"></div>
71. <div id="pie-chart-container"></div>
72. </div>
73. <script src="https://d3js.org/d3.v6.min.js"></script>
74. <script>
75. d3.csv("pivot\_table.csv").then(function(data) {
76. const width = 1000;
77. const height = 800;
78. const radius = Math.min(width, height) / 4;
79. const svg = d3.select("#map")
80. .attr("width", width)
81. .attr("height", height);
82. const totalUnemploymentRates = data.map(d => +d.total\_unemployment);
83. const maxRate = d3.max(totalUnemploymentRates);
84. const minRate = d3.min(totalUnemploymentRates);
85. const colorScale = d3.scaleSequential()
86. .domain([minRate, maxRate])
87. .interpolator(d3.interpolateRdYlBu);
88. d3.json("europe.geojson").then(function(geojson) {
89. const projection = d3.geoMercator()
90. .fitSize([width, height], geojson);
91. const pathGenerator = d3.geoPath()
92. .projection(projection);
93. const handleMouseOver = function(event, d) {
94. const countryData = data.find(country => country.country === d.properties.NAME);
95. const totalUnemploymentRate = countryData ? +countryData.total\_unemployment : null;
96. const tooltip = d3.select("#tooltip");
97. tooltip.style("visibility", "visible")
98. .html(`<strong>${d.properties.NAME}</strong><br>
99. Total Unemployment: ${totalUnemploymentRate !== null ? totalUnemploymentRate + '%' : 'N/A (Not in EU)'}<br>`)
100. .style("left", (event.pageX + 10) + "px")
101. .style("top", (event.pageY + 10) + "px");
102. };
103. const handleMouseOut = function() {
104. d3.select("#tooltip").style("visibility", "hidden");
105. };
106. svg.selectAll("path")
107. .data(geojson.features)
108. .enter()
109. .append("path")
110. .attr("d", pathGenerator)
111. .attr("fill", function(d) {
112. const countryData = data.find(country => country.country === d.properties.NAME);
113. const unemploymentRate = countryData ? +countryData.total\_unemployment : null;
114. return unemploymentRate !== null ? colorScale(unemploymentRate) : "#808080";
115. })
116. .on("mouseover", function(event, d) {
117. handleMouseOver(event, d);
118. })
119. .on("mouseout", function() {
120. handleMouseOut();
121. });
122. const pieSvg = d3.select("#pie-chart-container")
123. .append("svg")
124. .attr("width", width)
125. .attr("height", height / 2);
126. const pieData = [
127. { label: "Male", value: 0 },
128. { label: "Female", value: 0 }
129. ];
130. const pie = d3.pie()
131. .value(d => d.value)
132. .sort(null);
133. const arc = d3.arc()
134. .innerRadius(0)
135. .outerRadius(radius);
136. const arcs = pieSvg.selectAll("path")
137. .data(pie(pieData))
138. .enter()
139. .append("path")
140. .attr("transform", `translate(${width / 4},${height / 4})`)
141. .attr("d", arc)
142. .attr("fill", (\_, i) => i === 0 ? "#4e79a7" : "#f28e2b");
143. const arcLabel = d3.arc()
144. .innerRadius(0)
145. .outerRadius(radius \* 0.8);
146. const customText = ["Male", "Female"];
147. const captionLabels = pieSvg.selectAll("text.caption")
148. .data(pie(pieData))
149. .enter()
150. .append("text")
151. .attr("class", "caption")
152. .attr("transform", function(d) {
153. const centroid = arcLabel.centroid(d);
154. const x = centroid[0] + width / 4;
155. const y = centroid[1] + height / 4;
156. return `translate(${x}, ${y})`;
157. })
158. .attr("dy", "1.2em")
159. .attr("text-anchor", "middle")
160. .style("font-size", "20px")
161. .text('');
162. const updatePieChart = function(countryName) {
163. const countryData = data.find(country => country.country === countryName);
164. if (countryData) {
165. pieData[0].value = +countryData.male\_unemployment;
166. pieData[1].value = +countryData.female\_unemployment;
167. arcs.data(pie(pieData))
168. .transition()
169. .duration(500)
170. .attr("d", arc);
171. captionLabels.data(pie(pieData))
172. .transition()
173. .duration(500)
174. .attr("transform", function(d) {
175. const centroid = arcLabel.centroid(d);
176. const x = centroid[0] + width / 4;
177. const y = centroid[1] + height / 4;
178. return `translate(${x}, ${y})`;
179. })
180. .text((d, i) => {
181. if (countryName) {
182. const labelText = customText[i];
183. const valueText = d.data.value !== 0 ? d.data.value : '';
184. return `${labelText}: ${valueText/2}%`;
185. } else {
186. return '';
187. }
188. });
189. pieSvg.style("display", "block");
190. } else{
191. pieSvg.style("display", "none");
192. }
193. };
194. const legendData = colorScale.ticks(5).map(d => Math.round(d));
195. const legendColors = legendData.map(d => colorScale(d));
196. const legend = d3.select(".legend")
197. .selectAll(".legend-item")
198. .data(legendData)
199. .enter()
200. .append("div")
201. .attr("class", "legend-item");
202. legend.append("div")
203. .attr("class", "legend-color")
204. .style("background-color", (\_, i) => legendColors[i]);
205. legend.append("span")
206. .text((d, i) => {
207. const lowerBound = i > 0 ? legendData[i - 1] : 0;
208. const upperBound = d;
209. return `${lowerBound}% - ${upperBound}%`;
210. });
211. const grayLegendItem = d3.select(".legend")
212. .append("div")
213. .attr("class", "legend-item");
214. grayLegendItem.append("div")
215. .attr("class", "legend-color")
216. .style("background-color", "#808080");
217. grayLegendItem.append("span")
218. .text("N/A (Not in EU)");
220. d3.csv("file\_name.csv").then(function(data) {
221. const lineGraphHeight = height / 4;
222. const lineGraphSvg = d3.select("#line-chart-container")
223. .append("svg")
224. .attr("width", width)
225. .attr("height", lineGraphHeight + 80)
226. .style("display", "none");
227. const updateLineGraph = function(countryName) {
228. lineGraphSvg.selectAll("\*").remove();
229. if (countryName in data[0]) {
230. const lineGraphData = data.map(d => Number(d[countryName])).slice(0);
231. const xScale = d3.scaleBand()
232. .domain(d3.range(lineGraphData.length))
233. .range([0, width]);
234. const yMax = d3.max(lineGraphData);
235. const yMin = d3.min(lineGraphData);
236. const yScale = d3.scaleLinear()
237. .domain([yMin, yMax])
238. .range([lineGraphHeight, 0]);
239. const line = d3.line()
240. .x((\_, i) => xScale(i) + xScale.bandwidth() / 2)
241. .y(d => yScale(d));
242. lineGraphSvg.append("path")
243. .datum(lineGraphData)
244. .attr("class", "line")
245. .attr("d", line);
247. lineGraphSvg.append("g")
248. .attr("transform", `translate(0, ${lineGraphHeight})`)
249. .call(d3.axisBottom(xScale).tickFormat((d, i) => {
250. const months = ["Jan", "Feb", "Mar", "Apr", "May", "Jun", "Jul", "Aug", "Sep", "Oct", "Nov"];
251. return months[d];
252. }));
253. lineGraphSvg.append("g")
254. .attr("transform", `translate(40, 0)`)
255. .call(d3.axisLeft(yScale)
256. .ticks(lineGraphData.length)
257. .tickFormat(d3.format(""))
258. );
259. lineGraphSvg.style("display", "block");
260. } else {
261. lineGraphSvg.style("display", "none");
262. }
263. };
264. svg.selectAll("path")
265. .on("click", function(event, d) {
266. updatePieChart(d.properties.NAME);
267. updateLineGraph(d.properties.NAME);
268. });
269. });
270. });
271. });
272. </script>
273. </body>
274. </html>

## Implementacija osnovnog ponašanja

Za svaku državu prolazimo mišem i dobijamo na uvid podatak o ukupnoj nezaposlenosti. Klikom na pojedinu dobijemo kružni dijagram sa udjelom po spolovima, te linijski graf o ukupnom izvozu te države.

# KV5 - Dovršetak projektnog zadatka i pisanje dokumentacije

## Izrada dokumenta - projektne dokumentacije

Hijerarhija projekta:

* index.html
* style.css
* d3.v6.min.js
* unemployment.csv
* europe.geojson
* imports.csv
* exports.csv

Popis korištenih tehnologija, bez opisa:

* HTML
* CSS
* JavaScript
* D3.js (version 6)
* Python

Upute za postavljanje:

Preuzmite sve datoteke izlistane u hijerarhiji projekta. Smjestite preuzete datoteke u direktorij na vašem lokalnom računalu.

Upute za korištenje:

Otvorite datoteku "index.html" u web pregledniku. Stranica će prikazati podatkovnu vizualizaciju parametara razvoja za europske zemlje u 2022. godini. Prikazat će se karta ukupne nezaposlenosti po zemlji, označena nijansom crvene boje na temelju stope nezaposlenosti. Pomoću padajućeg izbornika odaberite mjesec i pogledajte podatke o nezaposlenosti za taj mjesec. Odabirom zemlje na karti pokazivačem vidite ukupnu stopu nezaposlenosti za tu zemlju. Legenda dolje lijevo na karti prikazuje ljestvicu boja za stope nezaposlenosti. Ispod karte prikazan je kružni dijagram koji prikazuje spolni udio ukupne nezaposlenosti za odabranu zemlju i mjesec. Odaberite državu na karti da biste ažurirali graf s odgovarajućim podacima. Linijski graf prikazan s desne strane prikazuje podatke o uvozu i izvozu za odabranu zemlju. Kliknite zemlju na karti da ažurirate linijski grafikon s podacima o uvozu i izvozu za tu zemlju. Linijski grafikon prikazuje mjesečne vrijednosti uvoza i izvoza kroz cijelu godinu, on se ne ažurira klikom. Možete postaviti pokazivač miša iznad linijskog grafikona da biste vidjeli određene vrijednosti uvoza i izvoza za svaki mjesec.

# Literatura

<https://data.europa.eu/en/academy/going-beyond-bars-and-lines-effective-non-standard-data-visualisation> - "Making data visualisations like a pro using D3.js" webinar

# Prilog I

Poveznica na git repozitorij projekta:

Programski kod:

<!DOCTYPE html>

<html>

<head>

  <meta charset="utf-8">

  <meta name="viewport" content="width=device-width, initial-scale=1">

  <title>VP Projekt Kokanović Karlo</title>

  <style>

    #map-container {

      position: relative;

      width: 1200px;

      height: 800px;

      border: 1px solid #ccc;

      background-color: lightblue;

    }

    #tooltip {

      position: absolute;

      background-color: #fff;

      padding: 10px;

      border: 1px solid #ccc;

      visibility: hidden;

    }

    .legend {

      position: absolute;

      left: 20px;

      bottom: 20px;

      display: flex;

      flex-direction: column;

      justify-content: center;

      align-items: flex-start;

      font-size: 12px;

    }

    .legend-item {

      display: flex;

      align-items: center;

      margin-bottom: 5px;

    }

    .legend-color {

      width: 20px;

      height: 20px;

      margin-right: 5px;

    }

    #chart-container {

      display: flex;

      justify-content: flex-start;

    }

    #pie-chart-container {

      margin-top: 20px;

      margin-right: 20px;

      display: flex;

      justify-content: center;

    }

    #line-chart-container {

      margin-top: 90px;

      display: flex;

      justify-content: center;

      align-items: center;

    }

    .line {

      fill: none;

      stroke: steelblue;

      stroke-width: 2px;

    }

    #map path {

      stroke: #000;

      stroke-width: 0.5px;

    }

    h1 {

      text-align: center;

    }

    footer{

      margin-top: 30px;

    }

    label {

      font-size: larger;

      margin-bottom: 10px;

    }

    select {

      padding: 5px;

      font-size: 16px;

      border: 1px solid #ccc;

      border-radius: 4px;

      margin-bottom: 10px;

    }

  </style>

</head>

<body>

  <h1>Data Visualization of development parameters for European countries\* in 2022</h1>

  <h2>Map of total unemployment (age 25-74) by country:</h2>

  <label for="cars">Choose a month:</label>

  <select name = "month-select" id="month-select">

    <option value="1">January</option>

    <option value="2">February</option>

    <option value="3">March</option>

    <option value="4">April</option>

    <option value="5">May</option>

    <option value="6">June</option>

    <option value="7">July</option>

    <option value="8">August</option>

    <option value="9">September</option>

    <option value="10">October</option>

    <option value="11">November</option>

    <option value="12">December</option>

  </select>

  <div id="map-container">

    <svg id="map"></svg>

    <div id="tooltip"></div>

    <div class="legend"></div>

  </div>

  <div id="chart-container">

    <div class="chart-container">

      <h2 class="chart-title" id="pie-chart-title"></h2>

      <div id="pie-chart-container"></div>

    </div>

    <div class="chart-container">

      <h2 class="chart-title" id="line-chart-title"></h2>

      <div id="line-chart-container"></div>

    </div>

  </div>

  <script src="https://d3js.org/d3.v6.min.js"></script>

  <script>

    d3.csv("unemployment.csv").then(function(data) {

      const width = 1000;

      const height = 800;

      const radius = Math.min(width, height) / 4;

      const svg = d3.select("#map")

        .attr("width", width)

        .attr("height", height);

      const maxRate = d3.max(data, d => +d.total\_unemployment);

      const minRate = d3.min(data, d => +d.total\_unemployment);

      const colorScale = d3.scaleSequential()

        .domain([minRate, maxRate])

        .interpolator(d3.interpolateReds);

      d3.json("europe.geojson").then(function(geojson) {

        const projection = d3.geoMercator()

          .fitSize([width, height], geojson);

        const pathGenerator = d3.geoPath()

          .projection(projection);

        const updateMap = function(selectedMonth) {

          const filteredData = data.filter(d => parseInt(d.month) === selectedMonth);

          const handleMouseOver = function(event, d) {

            const countryData = filteredData.find(country => country.country === d.properties.NAME);

            const totalUnemploymentRate = countryData ? +countryData.total\_unemployment : null;

            const tooltip = d3.select("#tooltip");

            tooltip.style("visibility", "visible")

              .html(`<strong>${d.properties.NAME}</strong><br>

                    Total Unemployment: ${totalUnemploymentRate !== null ? totalUnemploymentRate + '%' : 'N/A (Not in EU)'}<br>`)

              .style("left", (event.pageX + 10) + "px")

              .style("top", (event.pageY - 80) + "px");

            };

          const handleMouseOut = function() {

            d3.select("#tooltip").style("visibility", "hidden");

            };

            svg.selectAll("path")

              .data(geojson.features)

              .join("path")

              .attr("d", pathGenerator)

              .transition()

              .duration(500)

              .attr("fill", function(d) {

                const countryData = filteredData.find(country => country.country === d.properties.NAME);

                const unemploymentRate = countryData ? +countryData.total\_unemployment : null;

                return unemploymentRate !== null ? colorScale(unemploymentRate) : "#808080";

              });

            svg.selectAll("path")

              .on("mouseover", function(event, d) {

                handleMouseOver(event, d);

              })

              .on("mouseout", function() {

                handleMouseOut();

              });

            };

        const legendData = colorScale.ticks(5).map(d => Math.round(d));

          const legendColors = legendData.map(d => colorScale(d));

          const legend = d3.select(".legend")

            .selectAll(".legend-item")

            .data(legendData)

            .enter()

            .append("div")

            .attr("class", "legend-item");

          legend.append("div")

            .attr("class", "legend-color")

            .style("background-color", (\_, i) => legendColors[i]);

          legend.append("span")

            .text((d, i) => {

              const lowerBound = i > 0 ? legendData[i - 1] : 0;

              const upperBound = d;

              return `${lowerBound}% - ${upperBound}%`;

            });

          const grayLegendItem = d3.select(".legend")

            .append("div")

            .attr("class", "legend-item");

          grayLegendItem.append("div")

            .attr("class", "legend-color")

            .style("background-color", "#808080");

          grayLegendItem.append("span")

            .text("N/A (Not in EU)");

        let selectedCountry = "";

        const monthSelect = document.getElementById("month-select");

        monthSelect.addEventListener("change", function(event) {

          const selectedMonth = parseInt(monthSelect.value);

          updateMap(selectedMonth);

          updatePieChart(selectedCountry, selectedMonth);

        });

        updateMap(parseInt(monthSelect.value));

        const pieSvg = d3.select("#pie-chart-container")

          .append("svg")

          .attr("width", width)

          .attr("height", height / 2);

        const pieData = [

          { label: "Male", value: 0 },

          { label: "Female", value: 0 }

        ];

        const pie = d3.pie()

          .value(d => d.value)

          .sort(null);

        const arc = d3.arc()

          .innerRadius(0)

          .outerRadius(radius);

        const arcs = pieSvg.selectAll("path")

          .data(pie(pieData))

          .enter()

          .append("path")

          .attr("transform", `translate(${width / 4},${height / 4})`)

          .attr("d", arc)

          .attr("fill", (\_, i) => i === 0 ? "#4e79a7" : "#f28e2b");

        const arcLabel = d3.arc()

          .innerRadius(0)

          .outerRadius(radius \* 0.8);

        const customText = ["Male", "Female"];

        const captionLabels = pieSvg.selectAll("text.caption")

          .data(pie(pieData))

          .enter()

          .append("text")

          .attr("class", "caption")

          .attr("transform", function(d) {

            const centroid = arcLabel.centroid(d);

            const x = centroid[0] + width / 4;

            const y = centroid[1] + height / 4;

            return `translate(${x}, ${y})`;

          })

          .attr("dy", "1.2em")

          .attr("text-anchor", "middle")

          .style("font-size", "20px")

          .text('');

        let pieChartTitle;

        let lineGraphTitle;

        const updatePieChart = function(countryName, selectedMonth) {

          const filteredData = data.filter(d => parseInt(d.month) === selectedMonth);

          const countryData = filteredData.find(country => country.country === countryName);

          if (countryData) {

            pieData[0].value = +countryData.male\_unemployment;

            pieData[1].value = +countryData.female\_unemployment;

            arcs.data(pie(pieData))

              .transition()

              .duration(500)

              .attr("d", arc);

            captionLabels.data(pie(pieData))

              .transition()

              .duration(500)

              .attr("transform", function(d) {

                const centroid = arcLabel.centroid(d);

                const x = centroid[0] + width / 4;

                const y = centroid[1] + height / 4;

                return `translate(${x}, ${y})`;

              })

              .text((d, i) => {

                if (countryName) {

                  const labelText = customText[i];

                  const valueText = d.data.value !== 0 ? d.data.value : '';

                  return `${labelText}: ${valueText / 2}%`;

                } else {

                  return '';

                }

              });

            pieSvg.style("display", "block");

            pieChartTitle = `${countryName} - Gender distribution of total unemployment:`;

          } else {

            pieSvg.style("display", "none");

            pieChartTitle = null;

          }

        };

        const importDataPromise = d3.csv("imports.csv");

        const exportDataPromise = d3.csv("exports.csv");

        Promise.all([importDataPromise, exportDataPromise]).then(function(data) {

          const importData = data[0];

          const exportData = data[1];

          const lineGraphHeight = height / 4;

          let lineGraphTitle;

          const lineGraphSvg = d3.select("#line-chart-container")

              .append("svg")

              .attr("width", width)

              .attr("height", lineGraphHeight + 80)

              .style("display", "none");

              const updateLineGraph = function(countryName) {

              lineGraphSvg.selectAll("\*").remove();

              if (countryName in importData[0]) {

                const lineGraphDataImports = importData.map(d => Number(d[countryName])).slice(0);

                const lineGraphDataExports = exportData.map(d => Number(d[countryName])).slice(0);

                let scaler, smaller;

                if (d3.max(lineGraphDataImports) > d3.max(lineGraphDataExports)) {

                  scaler = lineGraphDataImports;

                } else {

                  scaler = lineGraphDataExports;

                }

                if (d3.min(lineGraphDataImports) > d3.min(lineGraphDataExports)) {

                  smaller = lineGraphDataExports;

                } else {

                  smaller = lineGraphDataImports;

                }

                const xScale = d3.scaleBand()

                  .domain(d3.range(scaler.length))

                  .range([0, width]);

                const yMax = d3.max(scaler);

                const yMin = d3.min(smaller);

                const yScale = d3.scaleLinear()

                  .domain([yMin \* 0.95, yMax \* 1.05]) // To fully render y axis values

                  .range([lineGraphHeight, 0]);

                const line = d3.line()

                  .x((\_, i) => xScale(i) + xScale.bandwidth() / 2)

                  .y(d => yScale(d));

                lineGraphSvg.append("path")

                  .datum(lineGraphDataImports)

                  .attr("class", "line")

                  .attr("d", line)

                  .style("opacity", 0)

                  .transition()

                  .delay(500)

                  .duration(1000)

                  .style("opacity", 1)

                  .attr("d", line);

                lineGraphSvg.append("path")

                  .datum(lineGraphDataExports)

                  .attr("class", "line")

                  .style("stroke", "red")

                  .attr("d", line)

                  .style("opacity", 0)

                  .transition()

                  .delay(500)

                  .duration(1000)

                  .style("opacity", 1)

                  .attr("d", line);

                lineGraphSvg.append("g")

                  .attr("transform", `translate(0, ${lineGraphHeight})`)

                  .call(d3.axisBottom(xScale).tickFormat((d, i) => {

                    const months = ["Jan", "Feb", "Mar", "Apr", "May", "Jun", "Jul", "Aug", "Sep", "Oct", "Nov", "Dec"];

                    return months[d];

                  }));

                lineGraphSvg.append("g")

                  .attr("transform", `translate(40, 0)`)

                  .call(d3.axisLeft(yScale)

                    .ticks(scaler.length)

                    .tickFormat(d3.format(""))

                  );

                lineGraphSvg.style("display", "block");

                lineGraphTitle = `${countryName} - Total imports and exports for each month (in mil. €):`;

                const handleMouseOverLine = function(event, d) {

                  const tooltip = d3.select("#tooltip");

                  tooltip.style("visibility", "visible")

                    .html(`${d}&nbspmillion&nbsp€`) //Adding spacing, because space button is a new row

                    .style("left", (event.pageX + 10) + "px")

                    .style("top", (event.pageY - 150) + "px")

                };

                const handleMouseOutLine = function() {

                  d3.select("#tooltip").style("visibility", "hidden");

                };

                lineGraphSvg.selectAll("circle")

                  .data(scaler)

                  .enter()

                  .append("circle")

                  .attr("cx", (\_, i) => xScale(i) + xScale.bandwidth() / 2)

                  .attr("cy", d => yScale(d))

                  .attr("r", 5)

                  .on("mouseover", handleMouseOverLine)

                  .on("mouseout", handleMouseOutLine)

                  .style("opacity", 0)

                  .transition()

                  .delay(1000)

                  .duration(1000)

                  .style("opacity", 1);

                lineGraphSvg.selectAll("circle2")

                  .data(smaller)

                  .enter()

                  .append("circle")

                  .attr("cx", (\_, i) => xScale(i) + xScale.bandwidth() / 2)

                  .attr("cy", d => yScale(d))

                  .attr("r", 5)

                  .on("mouseover", handleMouseOverLine)

                  .on("mouseout", handleMouseOutLine)

                  .style("opacity", 0)

                  .transition()

                  .delay(1000)

                  .duration(1000)

                  .style("opacity", 1);

              } else {

                lineGraphSvg.style("display", "none");

                lineGraphTitle = null;

              }

              const lineGraphLegend = lineGraphSvg.append("g")

                .attr("class", "legend")

                .attr("transform", `translate(${width - 70}, 0)`);

              lineGraphLegend.append("rect")

                .attr("width", 20)

                .attr("height", 20)

                .style("fill", "steelblue");

              lineGraphLegend.append("text")

                .attr("x", 30)

                .attr("y", 15)

                .text("Imports");

              lineGraphLegend.append("rect")

                .attr("width", 20)

                .attr("height", 20)

                .attr("y", 25)

                .style("fill", "red");

              lineGraphLegend.append("text")

                .attr("x", 30)

                .attr("y", 40)

                .text("Exports");

            };

            const updateChartTitles = function() {

              const pieChartTitleElement = document.getElementById("pie-chart-title");

              const lineGraphTitleElement = document.getElementById("line-chart-title");

              if (pieChartTitle) {

                pieChartTitleElement.textContent = pieChartTitle;

                pieChartTitleElement.style.display = "block";

              } else {

                pieChartTitleElement.style.display = "none";

              }

              if (lineGraphTitle) {

                lineGraphTitleElement.textContent = lineGraphTitle;

                lineGraphTitleElement.style.display = "block";

              } else {

                lineGraphTitleElement.style.display = "none";

              }

            };

          svg.selectAll("path")

            .on("click", function(event, d) {

              const selectedMonth = parseInt(monthSelect.value);

              selectedCountry = d.properties.NAME;

              updatePieChart(d.properties.NAME, selectedMonth);

              updateLineGraph(d.properties.NAME);

              updateChartTitles();

            });

        });

    });

});

  </script>

</body>

<footer>\* - 27 EU members + Switzerland, Iceland and Norway</footer>

</html>