# KV4 - Izrada konačne vizualizacije podataka

## Implementacija naprednih funkcionalnosti

* + 1. Napredne funkcionalnosti su uvođenje opisnoga prozora koji ispisuje dodatne informacije kada se nacilja na određeni element. Druga napredna funkionalnost je mogućnost odabira podataka koji će se koristiti prilikom vizualizacije odnosno izrade ili crtanja grafova.
    2. Opisni prozor koji ispisuje dodatne informacije prilikom postavljanja kursora na neki element radi tako što se otkriva ili sakriva određeni tooltip prozor te prilikom kretanja po elementu prati kursor I piše podatke vezane za taj element. Ova funkcionalnost je realizirana pomoću sljedećeg koda

pie\_charts.js

|  |
| --- |
| .on("mouseover", function(event, d) {  d3.select("#tooltip")  .html(`<b>OS: </b>Windows<br><b>Supports: </b>${d.data.label}<br><b>Count:</b> ${d.data.value}`)  .transition()  .duration(350)  .style("opacity", 1);  })  .on("mousemove", function(event) {  d3.select("#tooltip")  .style("left", (event.pageX + 5) + "px")  .style("top", (event.pageY - 28) + "px");  })  .on("mouseout", function() {  d3.select("#tooltip")  .transition()  .duration(350)  .style("opacity", 0);  })  ...  .on("mouseover", function(event, d) {  d3.select("#tooltip")  .html(`<b>OS: </b>Linux<br><b>Supports: </b>${d.data.label}<br><b>Count:</b> ${d.data.value}`)  .transition()  .duration(350)  .style("opacity", 1);  })  .on("mousemove", function(event) {  d3.select("#tooltip")  .style("left", (event.pageX + 5) + "px")  .style("top", (event.pageY - 28) + "px");  })  .on("mouseout", function() {  d3.select("#tooltip")  .transition()  .duration(350)  .style("opacity", 0);  })  ...  .on("mouseover", function(event, d) {  d3.select("#tooltip")  .html(`<b>OS: </b>MacOS<br><b>Supports: </b>${d.data.label}<br><b>Count:</b> ${d.data.value}`)  .transition()  .duration(350)  .style("opacity", 1);  })  .on("mousemove", function(event) {  d3.select("#tooltip")  .style("left", (event.pageX + 5) + "px")  .style("top", (event.pageY - 28) + "px");  })  .on("mouseout", function() {  d3.select("#tooltip")  .transition()  .duration(350)  .style("opacity", 0);  }) |

Kod 4.1.1: Prikaz pie\_charts.js skripte za prikaz, skrivanje, ispis opisnoga prozora kod kružnih grafova

Histogram.js

|  |
| --- |
| ...  .on("mouseover", function(event, d) {  d3.select("#tooltip")  .html(`<b>Year:</b> ${d.x}<br><b>Released Games:</b> ${d.y}`)  .transition()  .duration(350)  .style("opacity", 1);  })  .on("mousemove", function(event) {  d3.select("#tooltip")  .style("left", (event.pageX + 5) + "px")  .style("top", (event.pageY - 28) + "px");  })  .on("mouseout", function() {  d3.select("#tooltip")  .transition()  .duration(350)  .style("opacity", 0);  })  ... |

Kod 4.1.2: Prikaz Histogram.js skripte za prikaz, skrivanje, ispis opisnoga prozora kod histograma

Bar\_chart.js

|  |
| --- |
| ...  .on("mouseover", function(event, d) {  d3.select("#tooltip")  .html(`<b>Category:</b> ${d.y}<br><b>Number of Games:</b> ${d.x}`)  .transition()  .duration(350)  .style("opacity", 1);  })  .on("mousemove", function(event) {  d3.select("#tooltip")  .style("left", (event.pageX + 5) + "px")  .style("top", (event.pageY - 28) + "px");  })  .on("mouseout", function() {  d3.select("#tooltip")  .transition()  .duration(350)  .style("opacity", 0);  })  ... |

Kod 4.1.3: Prikaz Bar\_chart.js skripte za prikaz, skrivanje, ispis opisnoga prozora kod stupčastoga grafa

Line\_diagram.js

|  |
| --- |
| ...  .on("mouseover", function(event, d) {  d3.select("#tooltip")  .html(`<b>Year:</b> ${d.time}<br><b>Average Price:</b> ${d.price}`) //This will need to be updated  .transition()  .duration(350)  .style("opacity", 1);  })  .on("mousemove", function(event) {  d3.select("#tooltip")  .style("left", (event.pageX + 5) + "px")  .style("top", (event.pageY - 28) + "px");  })  .on("mouseout", function() {  d3.select("#tooltip")  .transition()  .duration(350)  .style("opacity", 0);  })  ... |

Kod 4.1.4: Prikaz line\_diagram.js skripte za prikaz, skrivanje, ispis opisnoga prozora kod linijskoga grafa

Druga ujedno i zadnja napredna funkcionalnost je mogućnost odabira podataka koji će se koristiti za vizualizaciju. Ova funkcionalnost radi tako što su svi elementi „clickable“ te dodaju svoju informaciju kao filter za buduće dohvaćanje podataka. Funkcionalnost je implementirana na sljedeći način:

pie\_charts.js

|  |
| --- |
| ...  .on('click',(e,d) =>{  const object = {win: d.data.label === "Yes"? "True" : "False"};  const filter\_objects = filters.filter(filter => typeof filter === 'object')  const final = filter\_objects.findIndex(filter => JSON.stringify(filter) === JSON.stringify(object))  if (final !== -1) {  alert("Filter is already added");  } else {  filters.push(object);  AddToFilterBar();  }  });  ...  .on('click',(e,d) =>{  const object = {linux: d.data.label === "Yes"? "True" : "False"};  const filter\_objects = filters.filter(filter => typeof filter === 'object')  const final = filter\_objects.findIndex(filter => JSON.stringify(filter) === JSON.stringify(object))  if (final !== -1) {  alert("Filter is already added");  } else {  filters.push(object);  AddToFilterBar();  }  });  ...  .on('click',(e,d) =>{  const object = {mac: d.data.label === "Yes"? "True" : "False"};  const filter\_objects = filters.filter(filter => typeof filter === 'object')  const final = filter\_objects.findIndex(filter => JSON.stringify(filter) === JSON.stringify(object))  if (final !== -1) {  alert("Filter is already added");  } else {  filters.push(object);  AddToFilterBar();  }  }); |

Kod 4.1.5: Prikaz pie\_charts.js skripte za dodavanje filtera operacijski sustav

histogram.js

|  |
| --- |
| ...  .on('click', e => {  const year = e.srcElement.\_\_data\_\_.x;  if (filters.includes(year)) {  alert("Filter is already added");  } else {  filters.push(year);  AddToFilterBar();  }  }); |

Kod 4.1.6: Prikaz histogram.js skripte za dodavanje filtera godina izlaska

bar\_char.js

|  |
| --- |
| ...  .on("click", e => {  if (filters.includes(e.srcElement.\_\_data\_\_.y)){  alert("Filter is already added");  }else{  filters.push(e.srcElement.\_\_data\_\_.y);  AddToFilterBar();  }  })  ... |

Kod 4.1.7: Prikaz bar\_chart.js skripte za dodavanje filtera rating razine

line\_diagram

|  |
| --- |
| …  .on('click',e =>{  if (filters.includes(e.srcElement.\_\_data\_\_.time)){  alert("Filter is already added");  }  else{  filters.push(e.srcElement.\_\_data\_\_.time);  AddToFilterBar();  }  })  … |

Kod 4.1.8: Prikaz line\_diagram.js skripte za dodavanje filtera godine izlaska

filter\_and\_update

|  |
| --- |
| function ApplyFilters(){  rating\_levels = ['Overwhelmingly Positive','Very Positive','Positive','Mostly Positive','Mixed','Mostly Negative','Negative','Very Negative','Overwhelmingly Negative']  years = [2012, 2013, 2014, 2015, 2016, 2017, 2018, 2019, 2020, 2021, 2022, 2023];  let yearFilters = filters.filter(item => typeof item === 'number');  let ratingFilters = filters.filter(item => typeof item === 'string');  let osFilters = filters.filter(item => typeof item === 'object');  yearFilters.sort((a,b) => a-b);  if(yearFilters.length === 0){  yearFilters = years;  }  if(ratingFilters.length === 0){  ratingFilters = rating\_levels;  }  filtered\_data = all\_data.filter(item =>  yearFilters.includes(item.year) &&  ratingFilters.includes(item.rating) &&  (  (  osFilters.findIndex(filter => JSON.stringify(filter) === JSON.stringify({win: item.win})) !== -1 ||  osFilters.findIndex(filter => JSON.stringify(filter) === JSON.stringify({linux: item.linux})) !== -1 ||  osFilters.findIndex(filter => JSON.stringify(filter) === JSON.stringify({mac: item.mac})) !== -1  ) || osFilters.length === 0  )  );    // Line Diagram  data\_line = [];  yearFilters.forEach(year => {  temp\_data = filtered\_data.filter(entry => entry.year === year);  let sum = 0;  temp\_data.forEach( datum =>{  sum += parseFloat(datum.price\_final)  })  data\_line.push({  time: parseInt(year),  price: (sum / temp\_data.length).toFixed(2)  });  });  // Bar-chart  data\_ratings= [];  ratingFilters.forEach(rating =>{  data\_ratings.push({  y: rating,  x: filtered\_data.filter(entry => entry.rating === rating).length  })  });  // Histogram  data\_histogram = [];  yearFilters.forEach(year => {  temp\_data = filtered\_data.filter(entry => entry.year === year);  data\_histogram.push({  x: parseInt(year),  y: temp\_data.length  });  });  data\_histogram.push({ // Have to add this otherwise last column would go outside of svg  x: parseInt(yearFilters.slice(-1))+1,  y: 0  });  // Pie Chart  win[0].value = filtered\_data.filter( entry => entry.win === 'True').length  win[1].value = filtered\_data.filter(entry => entry.win === 'False' ).length  linux[0].value = filtered\_data.filter( entry => entry.linux === 'True').length  linux[1].value = filtered\_data.filter(entry => entry.linux === 'False' ).length  mac[0].value = filtered\_data.filter( entry => entry.mac === 'True').length  mac[1].value = filtered\_data.filter(entry => entry.mac === 'False' ).length  update\_line\_diagram();  updateBarChart();  updateHistogram();  updatePies();  } |

Kod 4.1.9: Prikaz filter\_and\_update.js skripte za obradu svih dodanih filtera i dohvaćanje podataka

* + 1. Testiranjem je dokazano da su sve gore navedene funkcionalnosti uspješno implementirane a dokaz tome su i ove slike:

Slika na kojoj se prikazuje tekst, snimka zaslona, dijagram, Font

Opis je automatski generiran

Slika 4.1.1: Prikaz histograma sa opisnim prozorom



Slika 4.1.2: Prikaz liste aktivnih filtera

## Implementacija naprednog ponašanja

* + 1. Napredna ponašanja koja će se implementirati su: Transformacija prikaza točnije rečeno vizualizacija promjena prilikom odabira podataka. Uvođenje tranzicija kod skoro svih vizualizacijskih prijelaza osim kod kružnih dijagrama zbog tehničkih ograničenja.
    2. Transformacija prikaza i uvođenje tranzicija su međusobno povezani te su zbog toga implementirani zajedno. Transformacija prikaza će se izvesti nakon što se obrade filter te podaci sa tim filterima te su se njihovi pozivi mogli vidjeti u ranijem kodu (Kod 4.1.9). Napredna ponašanja su implementirana na sljedeći način:

histogram.js

|  |
| --- |
| function updateHistogram(){  x\_hist.domain(d3.extent(data\_histogram, d => d.x));  y\_hist.domain([0, d3.max(data\_histogram, d => d.y)]).nice();  xAxis\_hist.transition().duration(750).call(d3.axisBottom(x\_hist).ticks(data\_histogram.lenght));  yAxis\_hist.transition().duration(750).call(d3.axisLeft(y\_hist).tickSize(0).tickPadding(10));  const bars = svg\_hist.selectAll('rect').data(data\_histogram);  bars.enter().append('rect')  .attr('x', d => x\_hist(d.x))  .attr('width', 40)  .attr('y', d => y\_hist(d.y))  .attr('height', d => (height\_hist - margin\_hist.bottom) - y\_hist(d.y))  .attr('fill', '#01D1FF')  .style('cursor','pointer')  .on("mouseover", function(event, d) { //This must be added in case all filters are dropped and there are no more old rect  d3.select("#tooltip")  .html(`<b>Year:</b> ${d.x}<br><b>Released Games:</b> ${d.y}`)  .transition()  .duration(350)  .style("opacity", 1);  })  .on("mousemove", function(event) {  d3.select("#tooltip")  .style("left", (event.pageX + 5) + "px")  .style("top", (event.pageY - 28) + "px");  })  .on("mouseout", function() {  d3.select("#tooltip")  .transition()  .duration(350)  .style("opacity", 0);  })  .on('click', e => {  const year = e.srcElement.\_\_data\_\_.x;  if (filters.includes(year)) {  alert("Filter is already added");  } else {  filters.push(year);  AddToFilterBar();  }  })  .merge(bars)  .transition().duration(750)  .attr('x', d => x\_hist(d.x))  .attr('width', 40)  .attr('y', d => y\_hist(d.y))  .attr('height', d => (height\_hist - margin\_hist.bottom) - y\_hist(d.y));  bars.exit().transition().duration(750).style("opacity", 0).remove();  } |

Kod 4.2.1: Prikaz histogram.js skripte za transformaciju prikaza (ažuriranje dijagrama) histograma

bar\_chart.js

|  |
| --- |
| function updateBarChart() {  x\_bar.domain([0, d3.max(data\_ratings, d => d.x)]).nice();  y\_bar.domain(data\_ratings.map(d => d.y));  const bars = g\_bar.selectAll("rect").data(data\_ratings);  bars.enter().append("rect")  .attr("x", 0)  .attr("y", d => y\_bar(d.y))  .attr("width", d => x\_bar(d.x))  .attr("height", y\_bar.bandwidth())  .attr('fill', '#4E8CF6')  .style('cursor','pointer')  .on("mouseover", function(event, d) { //This must be added in case all filters are dropped and there are no more old elements  d3.select("#tooltip")  .html(`<b>Category:</b> ${d.y}<br><b>Number of Games:</b> ${d.x}`)  .transition()  .duration(350)  .style("opacity", 1);  })  .on("mousemove", function(event) {  d3.select("#tooltip")  .style("left", (event.pageX + 5) + "px")  .style("top", (event.pageY - 28) + "px");  })  .on("mouseout", function() {  d3.select("#tooltip")  .transition()  .duration(350)  .style("opacity", 0);  })  .on("click", e => {  if (filters.includes(e.srcElement.\_\_data\_\_.y)){  alert("Filter is already added");  }else{  filters.push(e.srcElement.\_\_data\_\_.y);  AddToFilterBar();  }  })  .merge(bars)  .transition().duration(750)  .attr("x", 0)  .attr("y", d => y\_bar(d.y))  .attr("width", d => x\_bar(d.x))  .attr("height", y\_bar.bandwidth());  yAxis\_bar.remove() // This is because text won't go over rect if it is not created after rect  yAxis\_bar = g\_bar.append("g")  .attr("class", "y-axis Apply-white")  .style('font-size','15px');  xAxis\_bar.transition().duration(750).call(d3.axisBottom(x\_bar));  yAxis\_bar.transition().duration(750).call(d3.axisLeft(y\_bar).tickSize(0).tickPadding(10));  yAxis\_bar.selectAll("text")  .style("text-anchor", "start");  bars.exit().transition().duration(750).style("opacity", 0).remove();  } |

Kod 4.2.2: Prikaz bar\_chart.js skripte za transformaciju prikaza (ažuriranje) stupčastoga dijagrama

line\_diagram.js

|  |
| --- |
| function update\_line\_diagram() {  // Update scales  x\_line.domain(d3.extent(data\_line, d => d.time));  y\_line.domain([0, Math.max(...data\_line.map(o => o.price))]);  const svg = d3.select("#line\_diagram");  // Update axes  xAxis\_line.transition().duration(750).call(d3.axisBottom(x\_line));  yAxis\_line.transition().duration(750).call(d3.axisLeft(y\_line).tickSize(0).tickPadding(10));  // Define the line  const line = d3.line()  .defined(d => !isNaN(d.price))  .x(d => x\_line(d.time))  .y(d => y\_line(d.price));  // Update line path  svg.select(".line")  .datum(data\_line)  .transition().duration(750)  .attr("d", line);  // Update circles  const circles = svg.selectAll(".dot")  .data(data\_line);  circles.enter().append("circle")  .attr("class", "dot")  .attr("r", 5)  .attr("cx", d => x\_line(d.time))  .attr("cy", d => y\_line(d.price))  .attr('fill','#E5C852')  .style('cursor','pointer')  .on("mouseover", function(event, d) { //This must be added in case all filters are dropped and there are no more old elements  d3.select("#tooltip")  .html(`<b>Year:</b> ${d.time}<br><b>Average Price:</b> ${d.price}`)  .transition()  .duration(350)  .style("opacity", 1);  })  .on("mousemove", function(event) {  d3.select("#tooltip")  .style("left", (event.pageX + 5) + "px")  .style("top", (event.pageY - 28) + "px");  })  .on("mouseout", function() {  d3.select("#tooltip")  .transition()  .duration(350)  .style("opacity", 0);  })  .on('click',e =>{  if (filters.includes(e.srcElement.\_\_data\_\_.time)){  alert("Filter is already added");  }  else{  filters.push(e.srcElement.\_\_data\_\_.time);  AddToFilterBar();  }  })  .merge(circles)  .transition().duration(750)  .attr("cx", d => x\_line(d.time))  .attr("cy", d => y\_line(d.price));  circles.exit().transition().duration(750).style("opacity", 0).remove();  } |

Kod 4.2.3: Prikaz line\_diagram.js skripte za transformaciju prikaza (ažuriranje) linijskoga dijagrama

pie\_charts.js

|  |
| --- |
| function updatePies(){ // For some reason can't update but have to redraw  // First Pie  svg1.selectAll('\*').remove()  const arc1 = d3.arc().innerRadius(0).outerRadius(radius\_pie);  arcs1 = svg1.selectAll('.arc')  .data(pie1(win))  .enter()  .append('g')  .attr('class', 'arc')  .attr('transform',`translate(${width\_pie / 2}, ${height\_pie / 2})`);  arcs1.append('path')  .attr('d', arc1)  .attr('fill', d => color\_pie(d.data.label))  .style('cursor','pointer')  .on("mouseover", function(event, d) {  d3.select("#tooltip")  .html(`<b>OS: </b>Windows<br><b>Supports: </b>${d.data.label}<br><b>Count:</b> ${d.data.value}`)  .transition()  .duration(350)  .style("opacity", 1);  })  .on("mousemove", function(event) {  d3.select("#tooltip")  .style("left", (event.pageX + 5) + "px")  .style("top", (event.pageY - 28) + "px");  })  .on("mouseout", function() {  d3.select("#tooltip")  .transition()  .duration(350)  .style("opacity", 0);  })  .on('click',(e,d) =>{  const object = {win: d.data.label === "Yes"? "True" : "False"};  const filter\_objects = filters.filter(filter => typeof filter === 'object')  const final = filter\_objects.findIndex(filter => JSON.stringify(filter) === JSON.stringify(object))  if (final !== -1) {  alert("Filter is already added");  } else {  filters.push(object);  AddToFilterBar();  }  });  arcs1.exit().transition().duration(750).style("opacity", 0).remove();  arcs1.append('text')  .attr('transform', d => `translate(${arc1.centroid(d)})`)  .attr('text-anchor', 'middle')  .text(d => d.data.label);  // Second pie  svg2.selectAll('\*').remove()  const arc2 = d3.arc().innerRadius(0).outerRadius(radius\_pie);  arcs2 = svg2.selectAll('arc')  .data(pie2(linux))  .enter()  .append('g')  .attr('class', 'arc')  .attr('transform',`translate(${width\_pie / 2}, ${height\_pie / 2})`);  arcs2.append('path')  .attr('d', arc2)  .attr('fill', d => color\_pie(d.data.label))  .style('cursor','pointer')  .on("mouseover", function(event, d) {  d3.select("#tooltip")  .html(`<b>OS: </b>Linux<br><b>Supports: </b>${d.data.label}<br><b>Count:</b> ${d.data.value}`)  .transition()  .duration(350)  .style("opacity", 1);  })  .on("mousemove", function(event) {  d3.select("#tooltip")  .style("left", (event.pageX + 5) + "px")  .style("top", (event.pageY - 28) + "px");  })  .on("mouseout", function() {  d3.select("#tooltip")  .transition()  .duration(350)  .style("opacity", 0);  })  .on('click',(e,d) =>{  const object = {linux: d.data.label === "Yes"? "True" : "False"};  const filter\_objects = filters.filter(filter => typeof filter === 'object')  const final = filter\_objects.findIndex(filter => JSON.stringify(filter) === JSON.stringify(object))  if (final !== -1) {  alert("Filter is already added");  } else {  filters.push(object);  AddToFilterBar();  }  });  arcs2.append('text')  .attr('transform', d => `translate(${arc2.centroid(d)})`)  .attr('text-anchor', 'middle')  .text(d => d.data.label);  arcs2.exit().transition().duration(750).style("opacity", 0).remove();  //Third pie  svg3.selectAll('\*').remove()  const arc3 = d3.arc().innerRadius(0).outerRadius(radius\_pie);  arcs3 = svg3.selectAll('arc')  .data(pie3(mac))  .enter()  .append('g')  .attr('class', 'arc')  .attr('transform',`translate(${width\_pie / 2}, ${height\_pie / 2})`);  arcs3.append('path')  .attr('d', arc3)  .attr('fill', d => color\_pie(d.data.label))  .style('cursor','pointer')  .on("mouseover", function(event, d) {  d3.select("#tooltip")  .html(`<b>OS: </b>MacOS<br><b>Supports: </b>${d.data.label}<br><b>Count:</b> ${d.data.value}`)  .transition()  .duration(350)  .style("opacity", 1);  })  .on("mousemove", function(event) {  d3.select("#tooltip")  .style("left", (event.pageX + 5) + "px")  .style("top", (event.pageY - 28) + "px");  })  .on("mouseout", function() {  d3.select("#tooltip")  .transition()  .duration(350)  .style("opacity", 0);  })  .on('click',(e,d) =>{  const object = {mac: d.data.label === "Yes"? "True" : "False"};  const filter\_objects = filters.filter(filter => typeof filter === 'object')  const final = filter\_objects.findIndex(filter => JSON.stringify(filter) === JSON.stringify(object))  if (final !== -1) {  alert("Filter is already added");  } else {  filters.push(object);  AddToFilterBar();  }  });    // Add labels  arcs3.append('text')  .attr('transform', d => `translate(${arc2.centroid(d)})`)  .attr('text-anchor', 'middle')  .text(d => d.data.label);  arcs3.exit().transition().duration(750).style("opacity", 0).remove();  } |

Kod 10: Prikaz pie\_charts.js skripte za transformaciju prikaza (ažuriranje) kružnih grafova

* + 1. Nakon izvođenja testiranja dokazano je da sva navedena napredna ponašanja rade na način koji je ranije naveden. Dokaz za transformaciju prikaza je sljedeća slika a za tranzicije ne postoji slika jer se oni ne vide na slikama:

Slika na kojoj se prikazuje tekst, snimka zaslona, dijagram, Multimedijski softver

Opis je automatski generiran

Slika 4.2.1: Prikaz Dashoboard-a nakon transformacije prikaza korištenjem raznih filtera