# KV4 - Izrada konačne vizualizacije podataka

## Implementacija osnovnih funkcionalnosti

Dovršena implementacija osnovnih funkcionalnosti

Sav kod je dostupan na github-u: <https://github.com/IamGrimbo/Visualization>

## Implementacija osnovnog ponašanja

Dovršena implementacija osnovnog ponašanja

Sav kod je dostupan na github-u: <https://github.com/IamGrimbo/Visualization>

## Implementacija naprednih funkcionalnosti

Dovršena implementacija naprednihfunkcionalnosti

Sav kod je dostupan na github-u: <https://github.com/IamGrimbo/Visualization>

## Implementacija naprednog ponašanja

Dovršena implementacija naprednog ponašanja

Sav kod je dostupan na github-u: <https://github.com/IamGrimbo/Visualization>

Primjer za barchart/piechart:

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>Visualizations</title>

<link rel="stylesheet" href="../CSS/visualizations.css">

<link rel="stylesheet" href="../CSS/topbar.css">

<script src="https://cdn.jsdelivr.net/npm/chart.js"></script>

<script src="../JS/visualizations.js" defer></script>

<style>

body {

font-family: Arial, sans-serif;

margin: 0;

padding: 0;

background-color: #000000;

height: 100vh;

display: flex;

flex-direction: column;

}

.container {

display: flex;

flex-direction: column;

flex-grow: 1;

box-sizing: border-box;

overflow: hidden;

}

.chart-container {

flex-grow: 1;

position: relative;

width: 100%;

height: calc(100% - 200px);

}

#chartCanvas {

width: 100%;

height: 100%;

}

.controls {

display: flex;

flex-direction: column;

align-items: center;

}

.controls-row {

display: flex;

justify-content: center;

align-items: center;

gap: 20px;

width: 100%;

}

</style>

</head>

<body>

<!-- Import Topbar -->

<div class="topbar-container">

<!-- Topbar will be imported here -->

</div>

<div class="container">

<h1>Game Visualizations</h1>

<div class="controls">

<select id="visualizationType">

<option value="genres">Genres</option>

<option value="tags">Tags</option>

<option value="categories">Categories</option>

<option value="os">Operating Systems</option>

<option value="release\_dates">Release Dates</option>

<option value="user\_reviews\_positive\_ratio">Reviews/Ratio</option>

<option value="user\_reviews\_price">Reviews/Price</option>

</select>

<div id="chartTypeContainer">

<select id="chartType">

<option value="bar">Bar Chart</option>

<option value="pie">Pie Chart</option>

</select>

</div>

</div>

<div class="chart-container">

<canvas id="chartCanvas"></canvas>

</div>

</div>

<script>

document.addEventListener('DOMContentLoaded', () => {

let chartInstance = null;

document.getElementById('generateChart').addEventListener('click', generateChart);

document.getElementById('visualizationType').addEventListener('change', handleVisualizationTypeChange);

async function fetchData() {

const response = await fetch('../Dataset/games.json');

const data = await response.json();

return data;

}

function countOccurrences(arr) {

return arr.reduce((acc, val) => {

acc[val] = (acc[val] || 0) + 1;

return acc;

}, {});

}

function processData(data, type) {

let items = [];

data.forEach(game => {

if (type === 'genres') {

items = items.concat(game.genres);

} else if (type === 'tags') {

items = items.concat(game.tags);

} else if (type === 'categories') {

items = items.concat(game.categories);

} else if (type === 'os') {

if (game.win === "True") items.push("Windows");

if (game.mac === "True") items.push("Mac");

if (game.linux === "True") items.push("Linux");

if (game.steam\_deck === "True") items.push("Steam Deck");

}

});

let counts = countOccurrences(items);

const threshold = type === 'genres' ? 100 : type === 'tags' ? 1500 : 140;

let filteredCounts = {};

let othersCount = 0;

let othersItems = [];

for (let key in counts) {

if (counts[key] >= threshold) {

filteredCounts[key] = counts[key];

} else {

othersCount += counts[key];

othersItems.push(key);

}

}

let sortedCounts = Object.entries(filteredCounts).sort((a, b) => b[1] - a[1]);

if (othersCount > 0) {

sortedCounts.push(['Others', othersCount]);

}

let labels = sortedCounts.map(item => item[0]);

let values = sortedCounts.map(item => item[1]);

return { labels, values, othersItems };

}

function generateColors(numColors) {

const colors = [];

for (let i = 0; i < numColors; i++) {

const color = `hsl(${Math.random() \* 360}, 100%, 75%)`;

colors.push(color);

}

return colors;

}

function handleVisualizationTypeChange() {

const visualizationType = document.getElementById('visualizationType').value;

const chartTypeContainer = document.getElementById('chartTypeContainer');

chartTypeContainer.style.display = 'block';

}

function generateChart() {

const visualizationType = document.getElementById('visualizationType').value;

const chartType = document.getElementById('chartType').value;

if (visualizationType === 'release\_dates' && !year && !month) {

alert('Please select at least one of year or month.');

return;

}

fetchData().then(data => {

if (chartInstance) {

chartInstance.destroy();

}

const { labels, values, othersItems } = processData(data, visualizationType);

const ctx = document.getElementById('chartCanvas').getContext('2d');

const chartData = {

labels: labels,

datasets: [{

label: visualizationType,

data: values,

backgroundColor: generateColors(labels.length),

borderColor: 'rgba(54, 162, 235, 1)',

borderWidth: 1

}]

};

const options = {

responsive: true,

color: 'red',

maintainAspectRatio: false,

onClick: (event, elements) => {

if (elements.length > 0) {

const index = elements[0].index;

displayGameDetails(labels[index], data, visualizationType, othersItems);

}

},

scales: {

y: {

beginAtZero: true,

ticks: {

stepSize: 1,

color: 'red',

callback: function (value) { return value; }

},

title: {

display: true,

color: 'red',

text: 'Count'

}

},

x: {

title: {

display: true,

color: 'red',

text: visualizationType.charAt(0).toUpperCase() + visualizationType.slice(1)

},

ticks: {

color: 'red',

}

}

}

};

if (chartType === 'pie') {

delete options.scales;

}

chartInstance = new Chart(ctx, {

type: chartType,

data: chartData,

options: options

});

});

}

});

</script>

</body>

</html>

