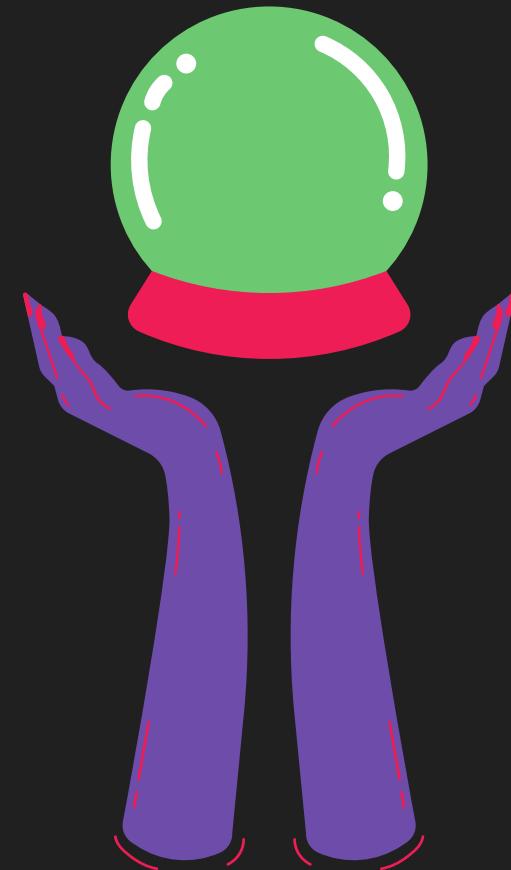
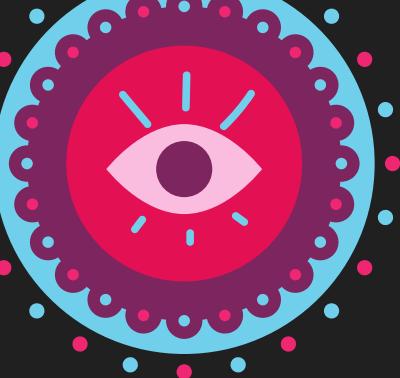


# Interaktywne Infografiki i Wizualizacje Danych na Stronach Internetowych



Sviatoslav Beiar





# Wprowadzenie

Czym są interaktywne infografiki?

Graficzne reprezentacje danych  
umożliwiające interakcję użytkownika.  
Dynamiczne prezentowanie informacji  
na stronach internetowych.

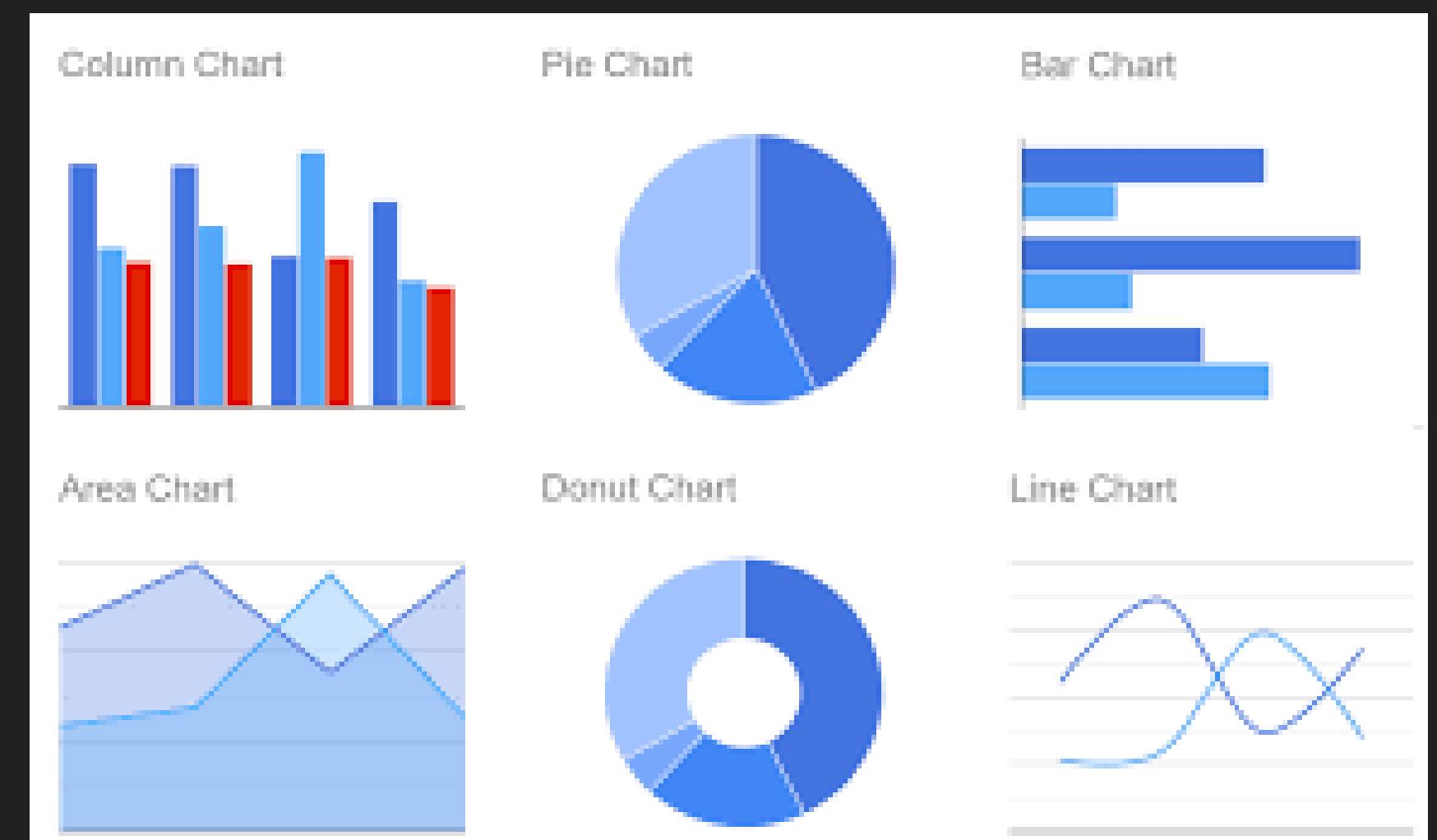
Połączenie estetyki z funkcjonalnością.

Znaczenie wizualizacji danych  
Ułatwiają zrozumienie

skomplikowanych informacji.

Zwiększą zaangażowanie  
użytkownika.

Poprawią skuteczność komunikacji.



# Technologie używane do tworzenia interaktywnych wizualizacji

HTML, CSS, JavaScript – Podstawowe technologie webowe.

D3.js – Biblioteka JavaScript do manipulacji danymi i ich dynamicznej wizualizacji.

Chart.js – Proste w użyciu wykresy w JavaScript.

Google Charts – Narzędzie ułatwiające integrację wykresów z usługami Google.

Tableau, Power BI – Narzędzia no-code/low-code do analizy i wizualizacji danych.



**Chart.js**



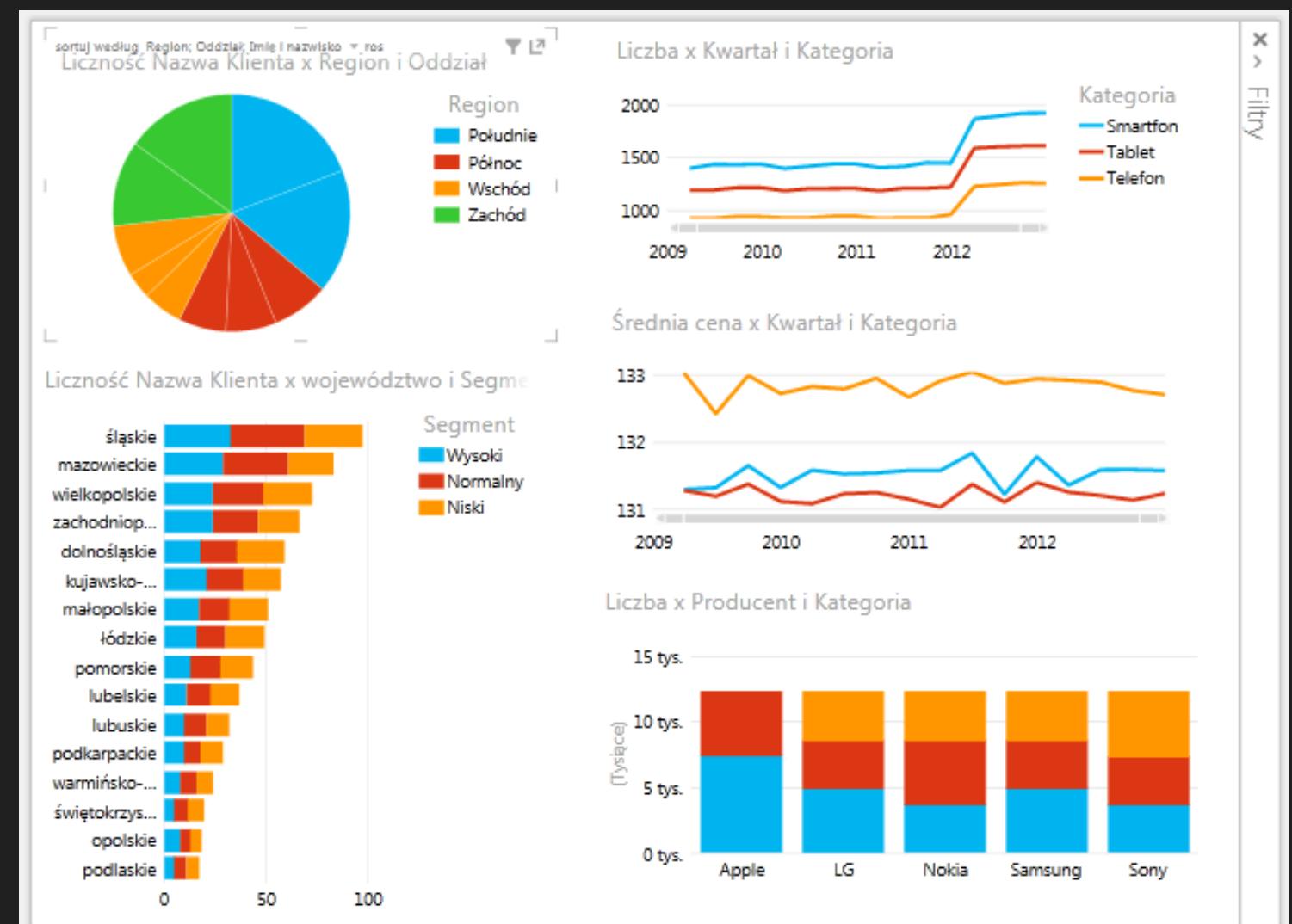
# Przykłady interaktywnych wizualizacji danych

Dashboardy finansowe - dynamiczne raporty prezentujące dane.

Mapy cieplne - analiza ruchu użytkowników na stronach.

Interaktywne wykresy - np. zmieniające się w czasie wykresy giełdowe.

Animowane infografiki - storytelling z wykorzystaniem danych.



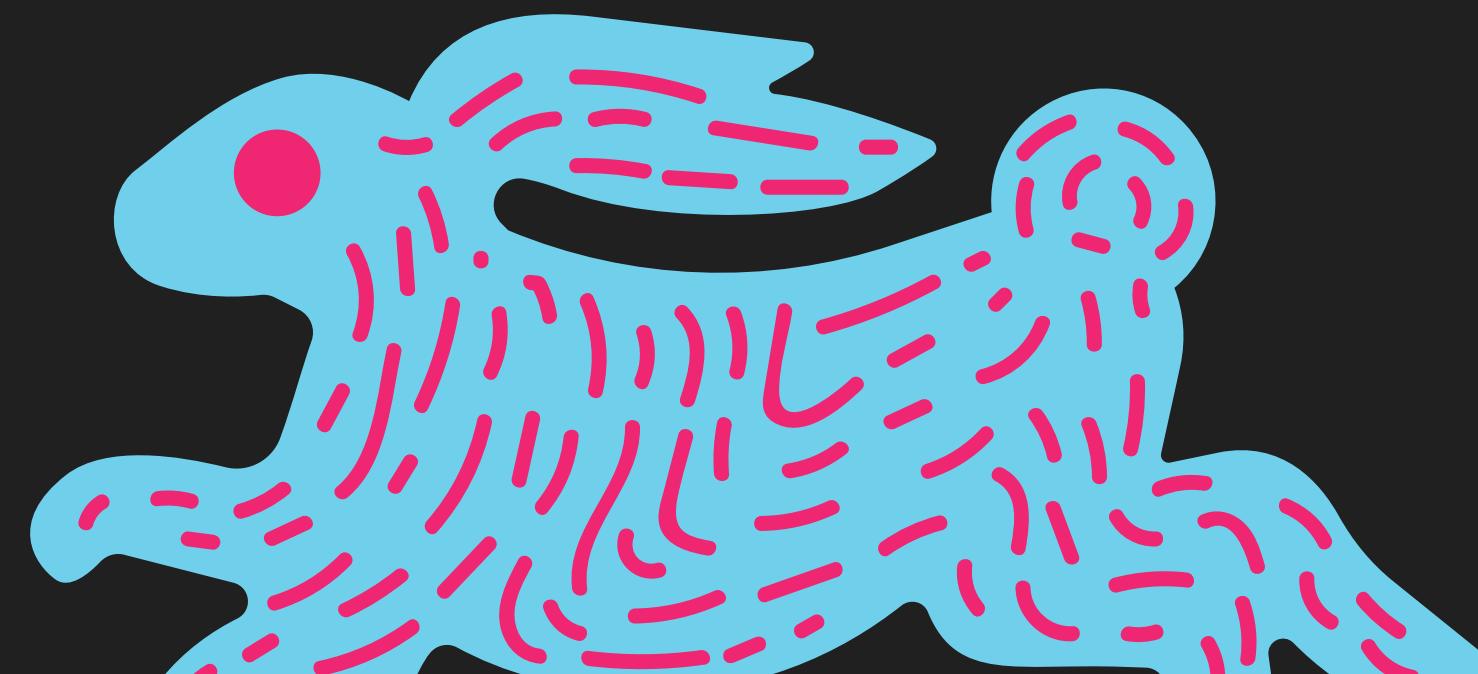
# D3.js – Biblioteka JavaScript do manipulacji danymi i ich dynamicznej wizualizacji

D3.js (Data-Driven Documents) to potężna biblioteka JavaScript do tworzenia interaktywnych wizualizacji danych w przeglądarce. Umożliwia manipulowanie dokumentami HTML i SVG na podstawie danych, dzięki czemu pozwala na dynamiczne generowanie wykresów, map, diagramów i innych form prezentacji informacji.



# Kluczowe cechy D3.js:

- ✓ Elastyczność - D3.js nie narzuca gotowych wykresów, tylko pozwala budować je od podstaw, dostosowując do potrzeb.
- ✓ Manipulacja DOM - Możliwość dynamicznej zmiany struktury HTML i SVG w oparciu o dane.
- ✓ Wsparcie dla skal i osi - Ułatwia przekształcanie danych liczbowych na wizualizacje.
- ✓ Obsługa animacji i interakcji - Można dodawać efekty przejść i reakcje na akcje użytkownika.
- ✓ Praca z danymi - Wspiera formaty takie jak CSV, JSON i API.



# Przykładowy kod: Tworzenie prostego wykresu słupkowego

```
// Dane wejściowe  
const dane = [10, 25, 40, 30, 20, 50];  
  
// Tworzenie kontenera SVG  
const svg = d3.select("body")  
    .append("svg")  
    .attr("width", 500)  
    .attr("height", 300);  
  
// Tworzenie słupków wykresu  
svg.selectAll("rect")  
    .data(dane)  
    .enter()  
    .append("rect")  
    .attr("x", (d, i) => i * 50)  
    .attr("y", d => 300 - d * 5)  
    .attr("width", 40)  
    .attr("height", d => d * 5)  
    .attr("fill", "blue");
```



Jak to działa?

Tworzymy obiekt `svg` w `body`. Pobieramy dane i dla każdego punktu tworzymy prostokąt (`rect`). Ustawiamy szerokość, wysokość i pozycję słupków na podstawie danych.

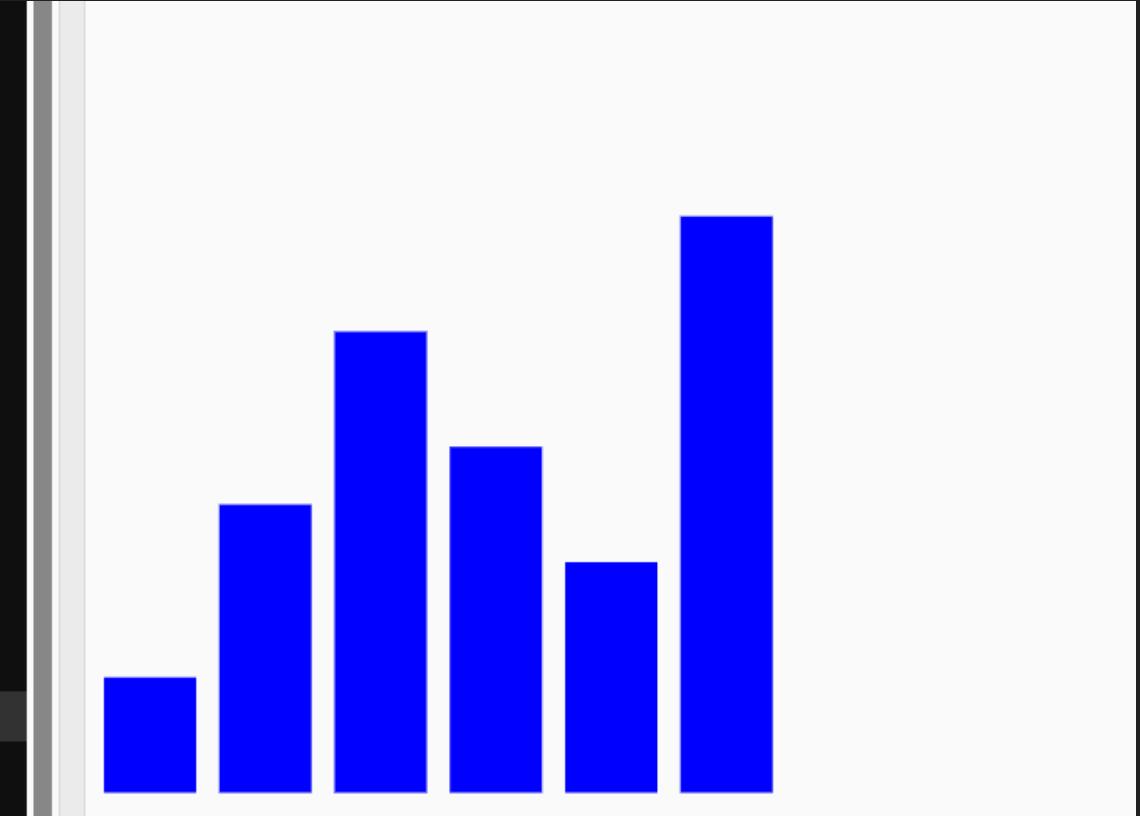
Oto jak będzie wyglądał wykres wygenerowany z podanego kodu.

Mögęcie sami wejść na stronę i pobawić się danymi

```
.append( "svg" )
    .attr("width", 500)
    .attr("height", 300);

// Tworzenie słupków wykresu
svg.selectAll("rect")
    .data(dane)
    .enter()
    .append("rect")
    .attr("x", (d, i) => i * 50)
    .attr("y", d => 300 - d * 5)
    .attr("width", 40)
    .attr("height", d => d * 5)
    .attr("fill", "blue");

</script>
</body>
```



[https://www.tutorialspoint.com/online\\_d3js\\_editor.php](https://www.tutorialspoint.com/online_d3js_editor.php)

# Zastosowania D3.js

 Dashboardy biznesowe – Interaktywne raporty, analizy finansowe.

 Mapy interaktywne – Wizualizacja danych geograficznych.

 Wykresy dynamiczne – Linie, kołowe, histogramy i wiele innych.

 Edukacja i nauka – Tworzenie animowanych diagramów i schematów.

D3.js jest potężnym narzędziem, ale wymaga dobrej znajomości JavaScript oraz manipulacji DOM. Jednak dzięki swojej elastyczności pozwala na tworzenie niezwykle rozbudowanych i interaktywnych wizualizacji. 

# Chart.js – Proste wykresy w JavaScript

Chart.js to popularna i łatwa w użyciu biblioteka JavaScript do tworzenia interaktywnych wykresów na stronach internetowych. Jest lekka, intuicyjna i oferuje szeroki wybór gotowych typów wykresów, co sprawia, że jest świetnym wyborem dla początkujących oraz zaawansowanych użytkowników.



# Kluczowe cechy Chart.js

- ✓ Łatwa integracja - Wystarczy dodać skrypt i używać prostej składni.
- ✓ Wiele typów wykresów - Obsługuje m.in. liniowe, słupkowe, kołowe, radarowe i inne.
- ✓ Responsywność - Automatyczne dostosowanie wykresów do różnych ekranów.
- ✓ Animacje i interaktywność - Możliwość podświetlania punktów i dynamicznych efektów.
- ✓ Dane dynamiczne - Można aktualizować wykresy na podstawie API i innych źródeł danych.

# Przykładowy kod: Wykres liniowy

```
<canvas id="myChart"></canvas>

<script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
<script>
const ctx = document.getElementById('myChart').getContext('2d');
const myChart = new Chart(ctx, {
    type: 'line',
    data: {
        labels: ['Styczeń', 'Luty', 'Marzec', 'Kwiecień'],
        datasets: [
            {
                label: 'Sprzedaż (w tys.)',
                data: [30, 50, 40, 60],
                borderColor: 'blue',
                borderWidth: 2,
                fill: false
            }
        ],
        options: {
            responsive: true,
            plugins: {
                legend: { display: true }
            }
        }
    );
</script>
```



Jak to działa?

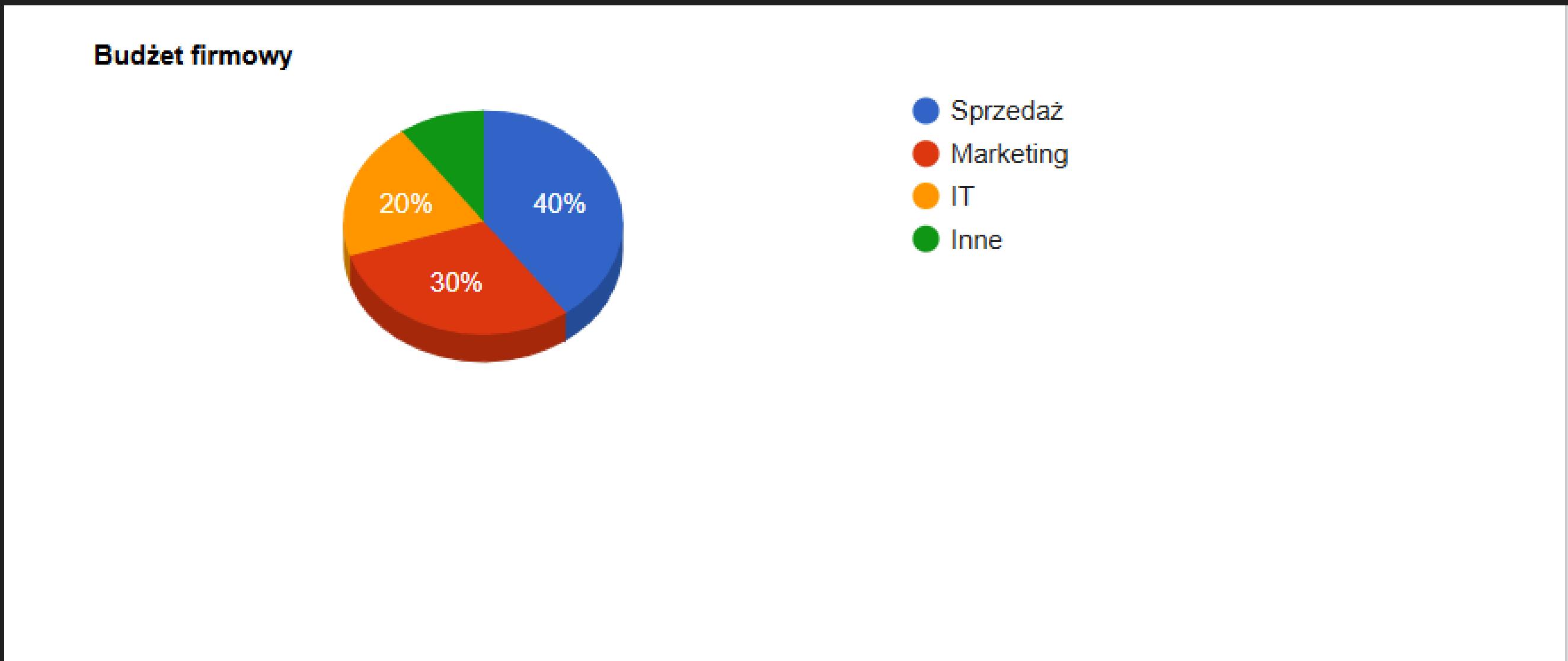
Tworzymy element `<canvas>`, który będzie miejscem dla wykresu.

Dołączamy bibliotekę `Chart.js`. Tworzymy wykres, określając typ (`line`), etykiety i dane.

Mozemy dostosować kolory, szerokość linii i inne opcje.

Oto jak będzie wyglądał wykres wygenerowany z podanego kodu.

[https://www.tutorialspoint.com/online\\_d3js\\_editor.php](https://www.tutorialspoint.com/online_d3js_editor.php)



# Zastosowania Chart.js

- 📊 Dashboardy biznesowe – Wizualizacja danych sprzedażowych i finansowych.
  - 📈 Analizy trendów – Śledzenie zmian w czasie (np. ruch na stronie, pogoda).
  - ⚡ Monitoring systemów – Prezentacja statystyk wydajności serwera czy aplikacji.
  - 🛒 E-commerce – Analiza popularności produktów i zachowań użytkowników.
- Chart.js to świetne narzędzie dla osób, które chcą szybko stworzyć estetyczne i funkcjonalne wykresy bez konieczności pisania skomplikowanego kodu. 🎯



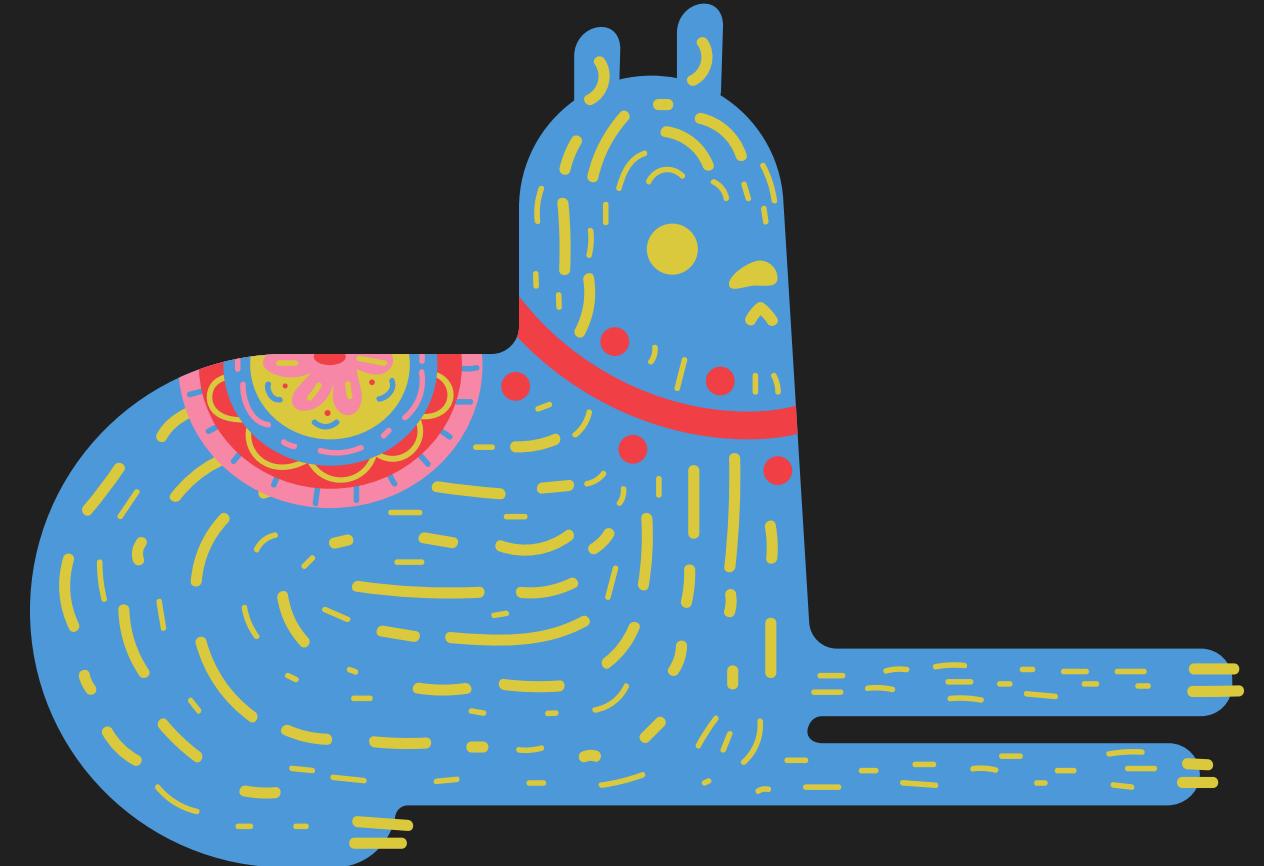
# Google Charts – łatwa integracja z usługami Google

Google Charts to darmowa biblioteka JavaScript od Google, umożliwiająca tworzenie interaktywnych wykresów na stronach internetowych. Jest łatwa w obsłudze, dobrze zoptymalizowana i świetnie integruje się z innymi usługami Google, takimi jak Google Sheets.



# Kluczowe cechy Google Charts

- ✓ Łatwa integracja – Wystarczy dodać skrypt i kilka linii kodu.
- ✓ Wiele typów wykresów – Obsługuje wykresy słupkowe, liniowe, kołowe, mapy, diagramy i wiele innych.
- ✓ Interaktywność – Obsługuje dynamiczne podświetlanie i filtry.
- ✓ Połączenie z Google Sheets – Można pobierać dane bezpośrednio z arkuszy Google.
- ✓ Optymalizacja – Wykresy są lekkie i działają płynnie nawet na dużych zbiorach danych.



# Przykładowy kod: Wykres kołowy

```
<div id="piechart"></div>

<script type="text/javascript"
src="https://www.gstatic.com/charts/loader.js"></script>
<script type="text/javascript">
google.charts.load('current', {'packages':['corechart']});
google.charts.setOnLoadCallback(drawChart);

function drawChart() {
var data = google.visualization.arrayToDataTable([
  ['Kategoria', 'Liczba'],
  ['Sprzedaż', 400],
  ['Marketing', 300],
  ['IT', 200],
  ['Inne', 100]
]);
var options = {
  title: 'Budżet firmowy',
  is3D: true
};
var chart = new
google.visualization.PieChart(document.getElementById('piechart'));
chart.draw(data, options);
}
</script>
```

 Jak to działa?

Dodajemy skrypt  
google.charts.loader.js.  
Ładujemy pakiet wykresów  
corechart.

Tworzymy funkcję  
drawChart(), która pobiera  
dane i rysuje wykres.

Definiujemy opcje wyglądu  
wykresu (np. is3D: true).

Oto jak będzie wyglądał wykres wygenerowany z podanego kodu.

[https://www.w3schools.com/js/tryit.asp?  
filename=trygoogle\\_chart\\_pie](https://www.w3schools.com/js/tryit.asp?filename=trygoogle_chart_pie)

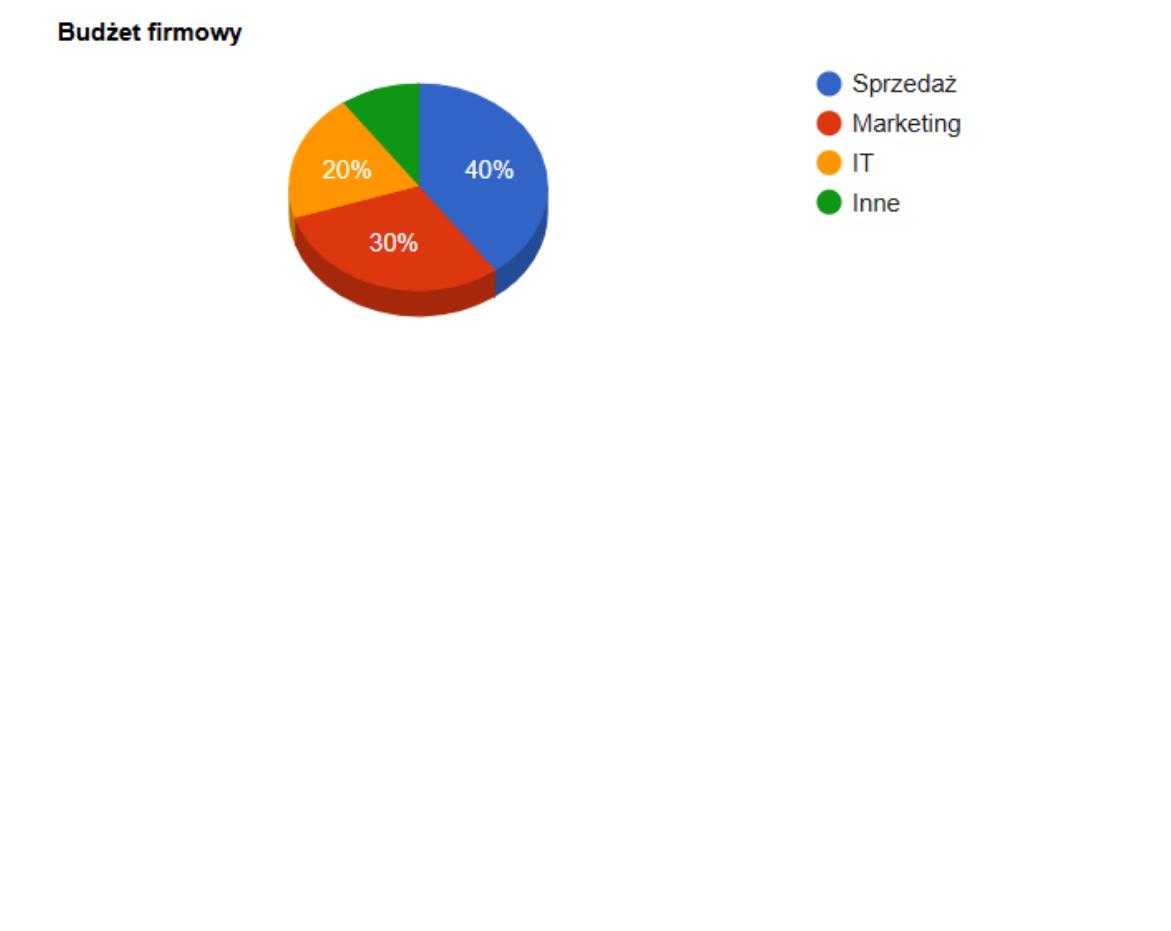
```
<div id="piechart"></div>

<script type="text/javascript" src="https://www.gstatic.com/charts/loader.js"></script>
<script type="text/javascript">
  google.charts.load('current', {'packages':['corechart']});
  google.charts.setOnLoadCallback(drawChart);

  function drawChart() {
    var data = google.visualization.arrayToDataTable([
      ['Kategoria', 'Liczba'],
      ['Sprzedaż', 400],
      ['Marketing', 300],
      ['IT', 200],
      ['Inne', 100]
    ]);

    var options = {
      title: 'Budżet firmowy',
      is3D: true
    };

    var chart = new google.visualization.PieChart(document.getElementById('piechart'));
    chart.draw(data, options);
  }
</script>
```



# Zastosowania Google Charts

 Raporty biznesowe – Wizualizacja sprzedaży, przychodów i kosztów.

 Dashboardy analityczne – Prezentacja statystyk w aplikacjach webowych.

 Mapy danych – Analiza geograficzna (np. mapa klientów).

 Prezentacje i edukacja – Szybkie tworzenie atrakcyjnych wykresów.

Google Charts to świetne rozwiązanie dla tych, którzy szukają prostego, ale funkcjonalnego narzędzia do wizualizacji danych z możliwością integracji z ekosystemem Google. 

# D3.js vs. Chart.js vs. Google Charts vs. No-Code Tools – Która technologia do wizualizacji danych jest najlepsza?

Na rynku dostępnych jest wiele narzędzi do wizualizacji danych – od zaawansowanych bibliotek JavaScript po intuicyjne rozwiązania no-code. Wybór odpowiedniego narzędzia zależy od potrzeb projektu, poziomu umiejętności użytkownika oraz wymagań dotyczących personalizacji i interaktywności.

# D3.js – Pełna kontrola i zaawansowane wizualizacje

📌 D3.js to potężna biblioteka JavaScript, dająca pełną swobodę w manipulacji danymi i ich prezentacji. Działa na bazie SVG i WebGL, co pozwala tworzyć interaktywne wykresy, mapy cieplne, diagramy i niestandardowe wizualizacje.



 Największe zalety:

Nieograniczone możliwości personalizacji - pełna kontrola nad wyglądem i działaniem wykresów.

Wysoka wydajność - dobrze radzi sobie z dużymi zbiorami danych.

Interaktywność i animacje - możliwość dodawania efektów dynamicznych i interakcji z użytkownikiem.

 Główne wyzwania:

Wysoka krzywa uczenia się - wymaga znajomości JavaScript, SVG i DOM.

Dłuższy czas implementacji - każda wizualizacja wymaga ręcznego kodowania.



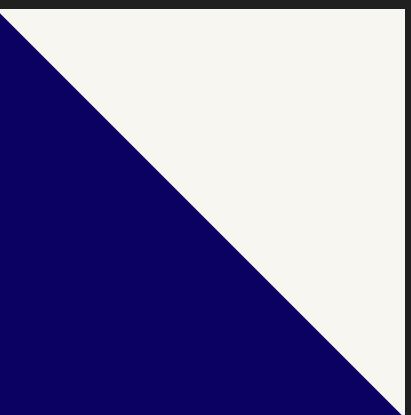
## 📌 Dla kogo?

Idealne rozwiązanie dla zaawansowanych programistów, którzy potrzebują niestandardowych wizualizacji np. do Big Data, dashboardów analitycznych czy raportów finansowych.



# Chart.js – Proste wykresy dla każdego

📌 Chart.js to lekka i intuicyjna biblioteka JavaScript do tworzenia wykresów na bazie HTML5 Canvas. Jest idealnym rozwiązaniem dla tych, którzy chcą szybko dodać estetyczne wykresy do stron internetowych.





Największe zalety:

Łatwość użycia - minimalna ilość kodu do stworzenia wykresu.

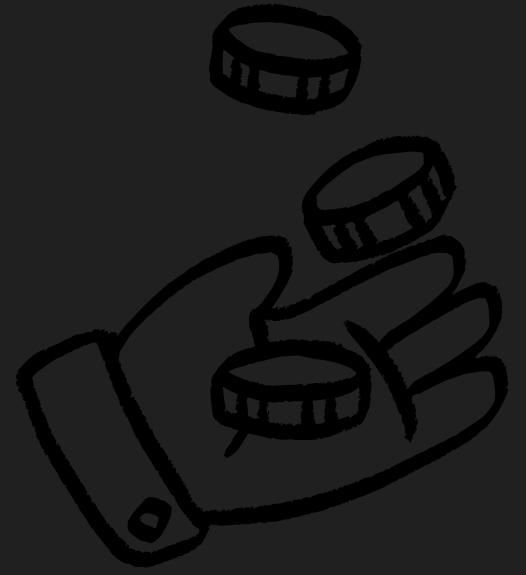
Obsługuje najczęściej używane typy wykresów - słupkowe, liniowe, kołowe, radarowe itp.

Wbudowane animacje - wykresy są dynamiczne i interaktywne bez potrzeby zaawansowanego kodowania.

✗ Główne ograniczenia:

Ograniczona personalizacja - nie można tworzyć skomplikowanych, niestandardowych wykresów.

Mniej elastyczne niż D3.js - ogranicza się do predefiniowanych typów wizualizacji



## 📌 Dla kogo?

Idealne dla deweloperów webowych, którzy potrzebują szybkiego i estetycznego rozwiązania do wizualizacji danych np. w dashboardach, raportach czy aplikacjach analitycznych.

# Google Charts – Gotowe wykresy, łatwa integracja

📌 Google Charts to jedno z najprostszych narzędzi do wizualizacji danych. Nie wymaga zaawansowanej znajomości programowania i pozwala szybko generować wykresy z danych np. z Google Sheets.



✓ Największe zalety:

Szybka implementacja – łatwe do wdrożenia nawet dla osób nieznających kodowania.

Bogaty zestaw gotowych wykresów – od podstawowych po bardziej zaawansowane, np. mapy czy diagramy organizacyjne.

Doskonała integracja z ekosystemem Google – świetne dla użytkowników korzystających z Google Sheets i Google Data Studio.

✗ Główne ograniczenia:

Ograniczona personalizacja – nie można tworzyć niestandardowych, skomplikowanych wizualizacji.

Zależność od Google – jeśli Google Charts zostanie wycofane, trzeba będzie przenieść dane do innego narzędzia.



## 📌 Dla kogo?

Najlepsze dla biznesu, edukacji i osób nietechnicznych, które potrzebują szybkich i przejrzystych wykresów np. do raportów i analiz.

# No-Code Tools – Wizualizacja danych bez kodowania

📌 Narzędzia no-code/low-code, takie jak Tableau, Microsoft Power BI, Datawrapper czy Google Data Studio, umożliwiają tworzenie wykresów i analiz danych bez potrzeby programowania. Są świetnym rozwiążaniem dla firm, analityków i osób, które chcą skupić się na interpretacji danych, a nie na ich kodowaniu.



 Największe zalety:

Brak konieczności kodowania – wystarczy kilka kliknięć, aby stworzyć raport lub wykres.

Zaawansowane możliwości analityczne – idealne do przetwarzania dużych zbiorów danych.

Integracja z wieloma źródłami danych – np. pliki CSV, bazy danych, API, usługi chmurowe.

 Główne ograniczenia:

Mniejsza elastyczność niż D3.js – można korzystać tylko z gotowych komponentów.

Niektóre narzędzia są płatne – np. Tableau wymaga subskrypcji.





## 📌 Dla kogo?

Doskonałe dla analityków danych, menedżerów, zespołów biznesowych – wszędzie tam, gdzie liczy się szybka analiza i wizualizacja danych bez znajomości programowania.

## Podsumowanie – Któż technologię wybrać?

- ◆ D3.js – dla zaawansowanych użytkowników, którzy chcą tworzyć niestandardowe, interaktywne wizualizacje.
- ◆ Chart.js – dla programistów szukających prostego i szybkiego rozwiązania do tworzenia wykresów.
- ◆ Google Charts – dla osób, które chcą łatwej integracji z Google i gotowych wykresów.
- ◆ No-Code Tools (Tableau, Power BI) – dla firm i analityków, którzy chcą wizualizować dane bez kodowania.

Każda z tych technologii ma swoje zastosowanie – wybierz tę, która najlepiej pasuje do Twojego projektu! 🚀



Dziękuję za uwagę!

