Interaktywne wizualizacje w JavaScript

Gdzie jest stan aplikacji?

W PRZEGLĄDARCE

Serwer tylko udostępnia stronę i zasoby. Generowanie wykresów i przetwarzanie danych odbywa się w przeglądarce.

+ Koszt przetwarzania danych i generowania wykresów zostaje przeniesiony na użytkownika
+ Niskie obciążenie serwera, które może być dodatkowo zmniejszone przez serwery CDN
+ Łatwo zintegrować takie rozwiązanie z już istniejącą aplikacją

- przetwarzanie danych w JS jest trudniejsze
- mniejszy zasób bibliotek do wykresów

NA SERWERZE

Każdemu użytkownikowi przydzielany jest proces serwera, który zarządza tym, co użytkownik widzi i reaguje na jego interakcje.

+ szybkie prototypowanie
+ dostęp do wszystkich bibliotek z R/Pythona
+ łatwe przetwarzanie danych

- duże zużycie zasobów

Co jest potrzebne?

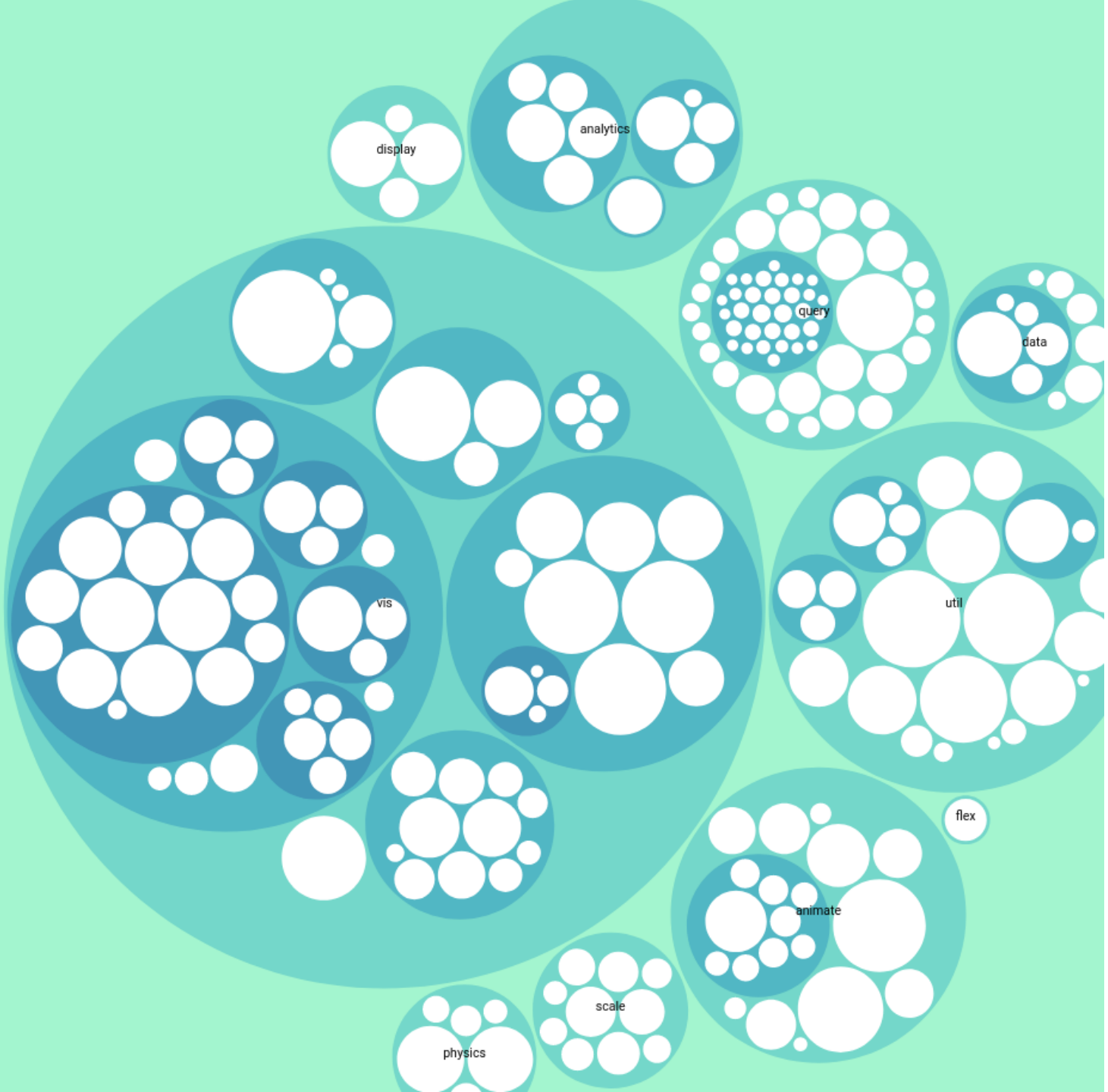
1. Dane
2. Obsługa wejścia od użytkownika (listy wyboru, przyciski itp.)
3. Przetwarzanie danych + wejścia
4. Wizualizacje

Dane

- Statyczny plik (np. na GitHub Pages)
- REST API
- Upload od użytkownika

Wizualizacje - wykresy

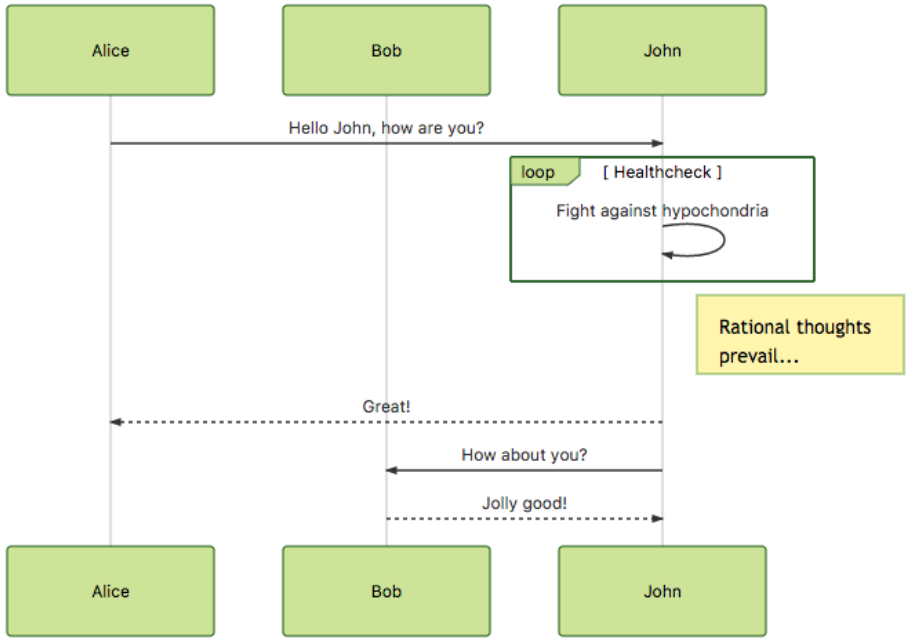

VIZCERAL
Service Traffic Map
Filters
Display
41.44%
0.96%
43.61%
5.18%
###,###
2.88%
49.59%
0.45%



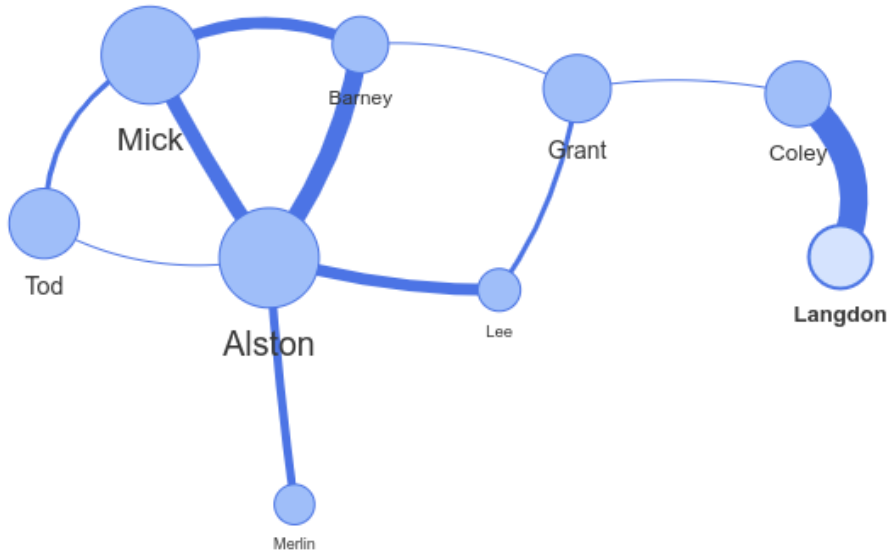
https://observablehq.com/@d3/zoomable-circle-packing

Wrappery do D3

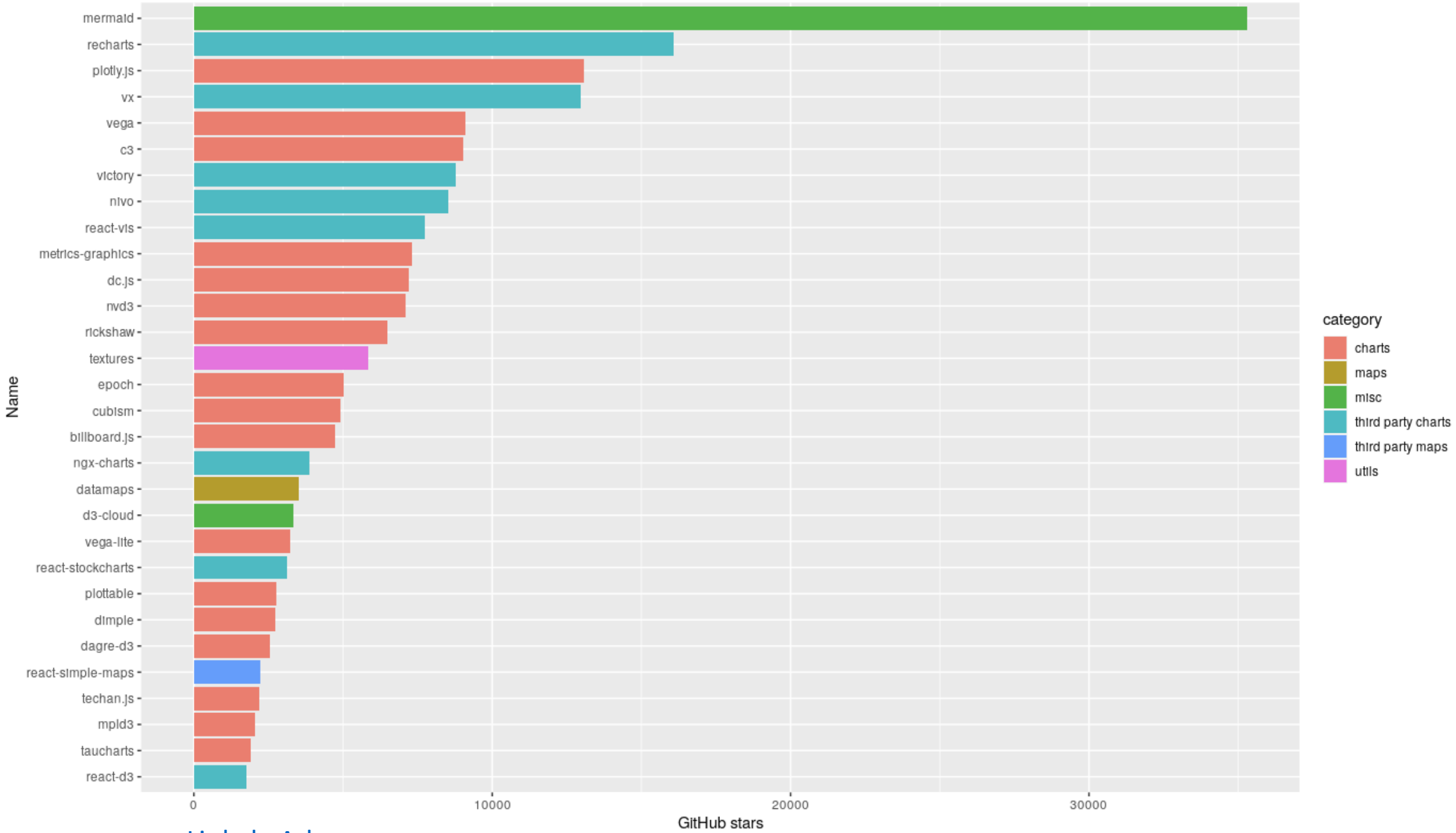
Mermaid



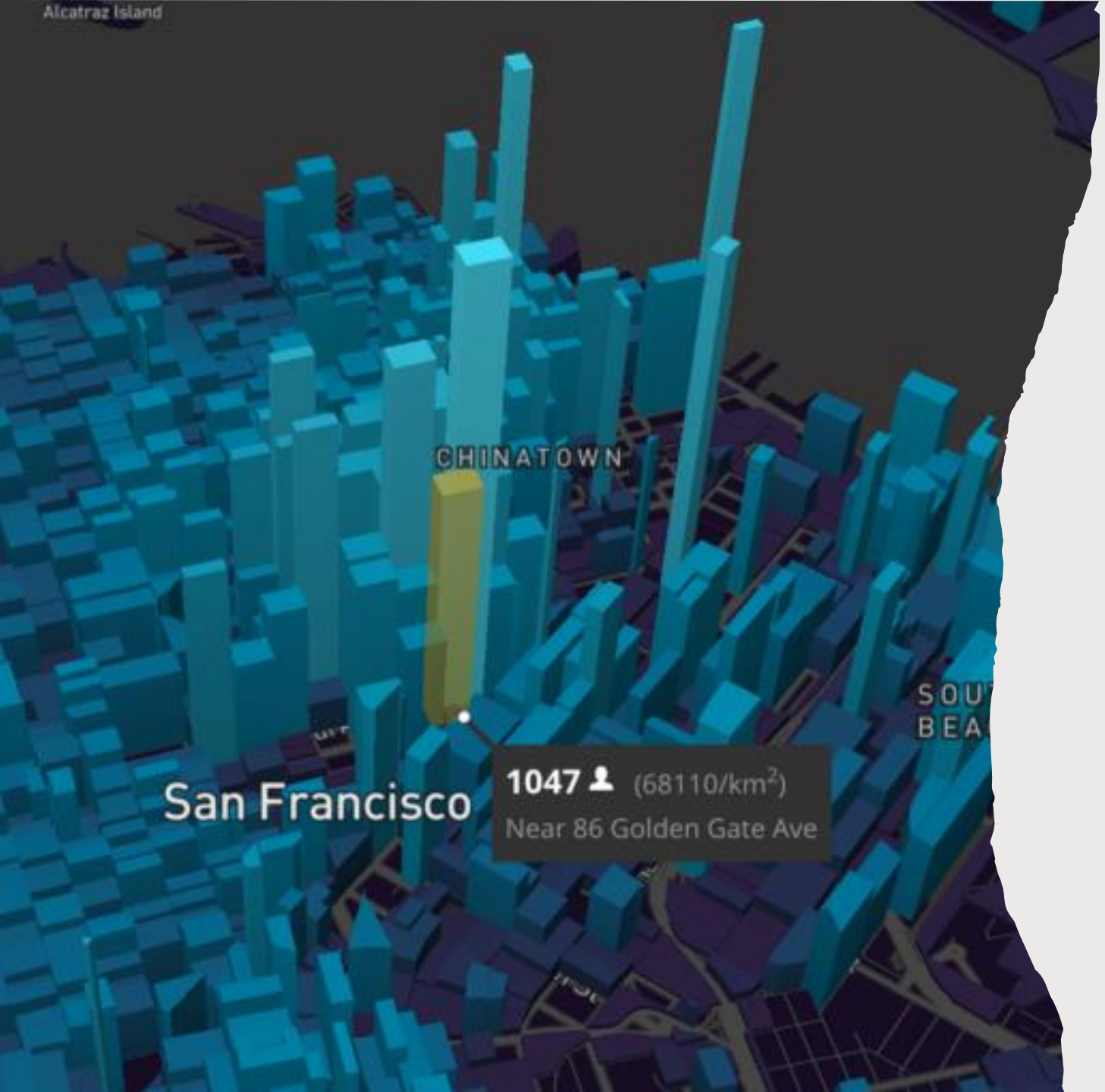
VisNetwork



Inne



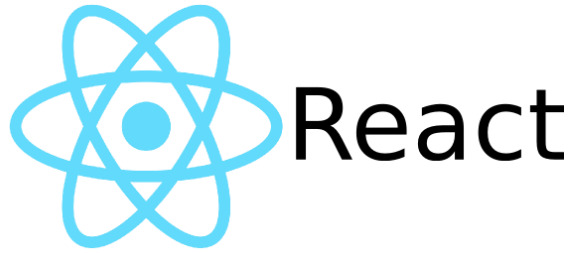
[Link do Arkusza]



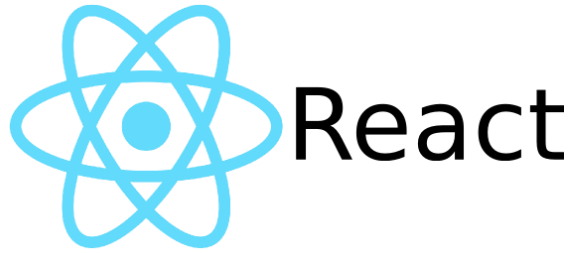
Mapy

- Mapbox
- Google Maps
- Bing Maps

W czym zrobić aplikacje?



W czym zrobić aplikacje?



REAKTYWNE

Reaktywność

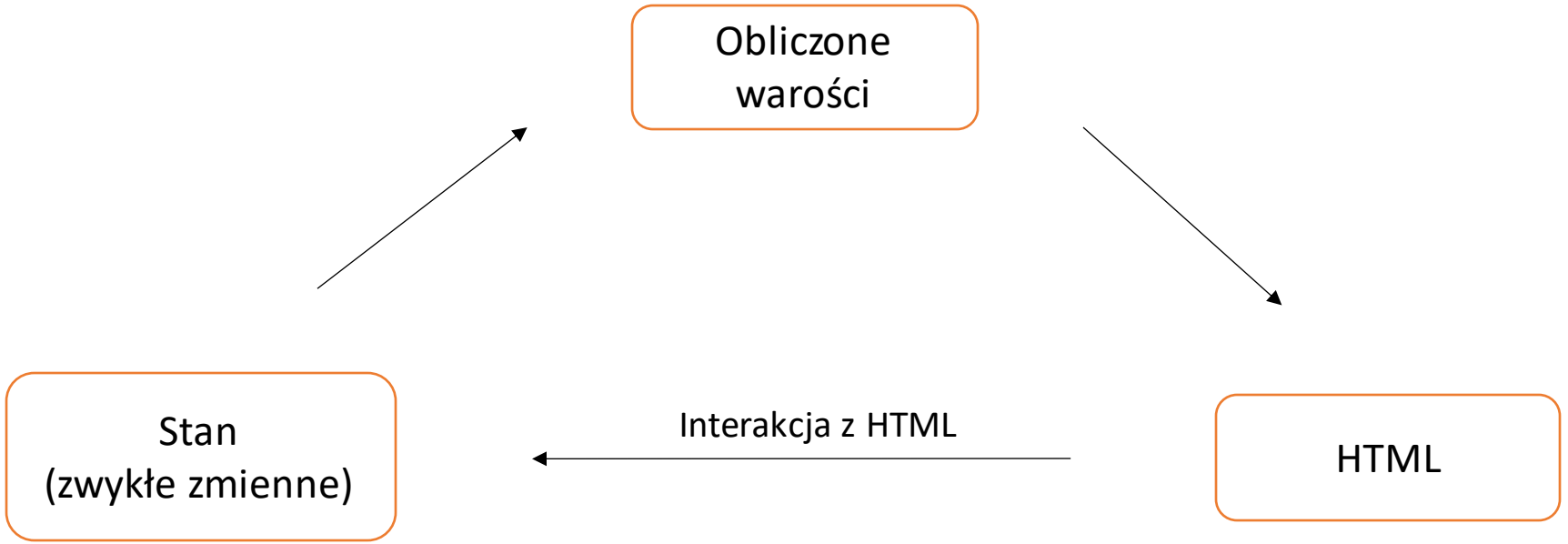
Klasyczne podejście

```
> a = 1
> b = 2
> c := a + b
> print(c)
3
> a = 5
> print(c)
3
```

Reaktywne programowanie

```
> a = 1
> b = 2
> c := a + b
> print(c)
3
> a = 5
> print(c)
7
```

Przepływ danych



Przykład

Zbiór danych pogodowych z kolumnami

- data
- godzina_pomiaru
- temperatura
- predkosc_wiatru
- suma_opadu
- stacja (nazwa miasta)

Jakie wejście ma użytkownik?

- Zakres dat
- Jednostka temperatury
- Miasto
- Które krzywe wyświetlać

```
<template>
  <div id="app">
  </div>
</template>
<script>
export default {
  name: 'App',
  data () {
    return {
    }
  },
  computed: {
  },
  methods: {
  },
  created () {
  }
}
</script>
<style>
</style>
```

```
<template>
  <div id="app">
  </div>
</template>
<script>
export default {
  name: 'App',
  data () {
    return {
      rawData: []
    }
  },
  computed: {
  },
  methods: {
  },
  created () {
    const url = 'https://gist.githubusercontent.com/piotrpiatyszek/dd032f41a46675a8f9de231f56f2fefc/raw/94a4893c0914e89daf5ec3139390cbd4406b15bc/meteo.json'
    this.$http.get(url).then(response => {
      this.rawData = response.body
    })
  }
}
</script>
<style>
</style>
```

```
<template>
  <div id="app">
    <select @change="selectedCity = $event.target.value">
      <option v-for="city in cities" :key="city">{{ city }}</option>
    </select>
  </div>
</template>
<script>
export default {
  name: 'App',
  data () {
    return {
      rawData: [],
      selectedCity: ''
    }
  },
  computed: {
    cities () {
      const citiesSet = new Set(this.rawData.map(d => d.stacja))
      return [...citiesSet]
    }
  },
  methods: {
  },
  created () {
    const url = 'https://gist.githubusercontent.com/piotrpiatyszek/dd032f41a46675a8f9de231f56f2fefc/raw/94a4893c0914e89daf5ec3139390cbd4406b15bc/meteo.json'
    this.$http.get(url).then(response => {
      this.rawData = response.body
    })
  }
}
</script>
<style>
</style>
```

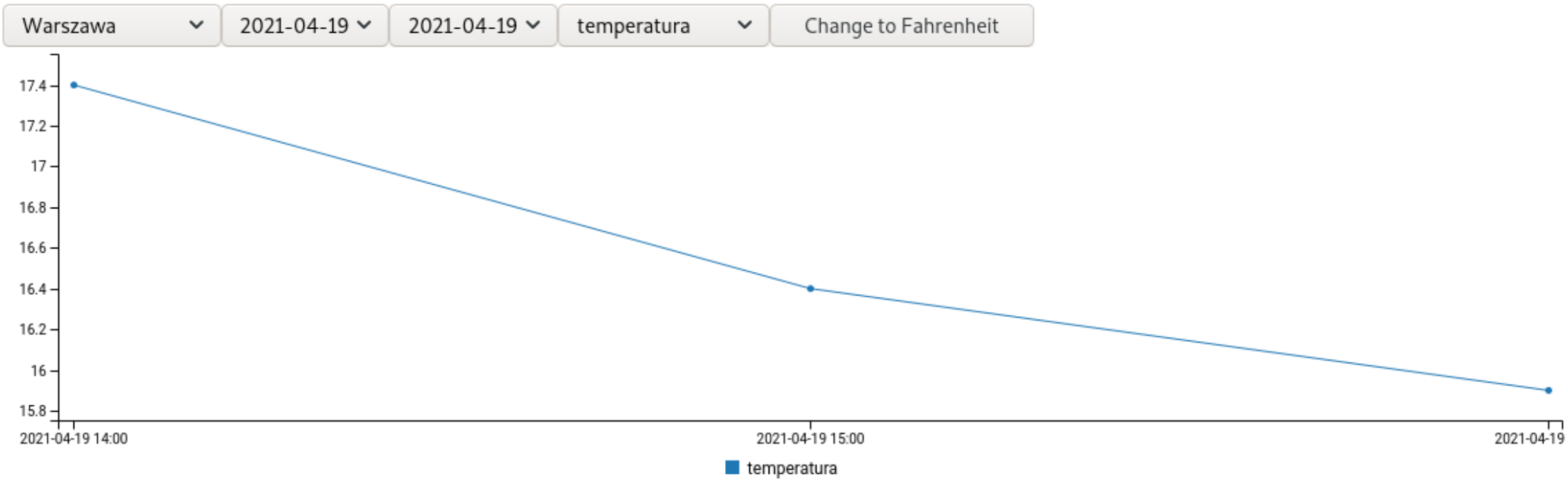
```
<template>
  <div id="app">
    <select @change="selectedCity = $event.target.value">
      <option></option>
      <option v-for="city in cities" :key="city">{{ city }}</option>
    </select>
    <select @change="selectedStartDate = $event.target.value">
      <option></option>
      <option v-for="date in dates" :key="date">{{ date }}</option>
    </select>
    <select @change="selectedEndDate = $event.target.value">
      <option></option>
      <option v-for="date in dates" :key="date">{{ date }}</option>
    </select>
    <select @change="selectedCurve = $event.target.value">
      <option></option>
      <option v-for="curve in availableCurves" :key="curve">{{ curve }}</option>
    </select>
  </div>
</template>
<script>
export default {
  name: 'App',
  data () {
    return {
      rawData: [],
      selectedCity: '',
      selectedStartDate: '',
      selectedEndDate: '',
      selectedCurve: '',
      availableCurves: ['temperatura', 'predkosc_wiatru', 'suma_opadu']
    }
  },
  computed: {
    cities () {
      const citiesSet = new Set(this.rawData.map(d => d.stacja))
      return [...citiesSet]
    },
    dates () {
      const datesSet = new Set(this.rawData.map(d => d.data_pomiaru))
      return [...datesSet]
    }
  },
```

```
    <select @change="selectedCurve = $event.target.value">
      <option></option>
      <option v-for="curve in availableCurves" :key="curve">{{ curve }}</option>
    </select>
    <button @click="temperatureInCelcius = !temperatureInCelcius">Change to {{ temperatureInCelcius ? 'Fahrenheit' : 'Celcius' }}</button>
  </div>
</template>
<script>
export default {
  name: 'App',
  data () {
    return {
      rawData: [],
      selectedCity: '',
      selectedStartDate: '',
      selectedEndDate: '',
      selectedCurve: '',
      temperatureInCelcius: true,
      availableCurves: ['temperatura', 'predkosc_wiatru', 'suma_opadu']
    }
  },
```

```
computed: {
  filteredData () {
    return this.rawData
      .filter(x => x.stacja === this.selectedCity)
      .filter(x => x.data_pomiaru >= this.selectedStartDate)
      .filter(x => x.data_pomiaru <= this.selectedEndDate)
      .filter(x => x[this.selectedCurve] !== undefined && x[this.selectedCurve] !== null)
  },
  unitsTransformed () {
    return this.filteredData.map(x => {
      return { ...x, temperatura: this.temperatureInCelcius ? x.temperatura : ((x.temperatura * (9 / 5)) + 32) }
    })
  },
  chartData () {
    return {
      x: this.unitsTransformed.map(x => x.data_pomiaru),
      y: this.unitsTransformed.map(x => x[this.selectedCurve])
    }
  },
```

```vue
<template>
  <div class="chart">
    <div id="chart"></div>
  </div>
</template>
<script>
import c3 from 'c3'
import 'c3/c3.css'
export default {
  name: 'Chart',
  props: {
    x: Array,
    y: Array
  },
  watch: {
    x: 'generateChart',
    y: 'generateChart'
  },
  methods: {
    generateChart () {
      c3.generate({
        data: {
          x: 'x',
          xFormat: '%Y-%m-%d %H',
          columns: [['x', ...this.x], ['y', ...this.y]]
        },
        axis: {
          x: { type: 'timeseries', tick: { format: '%Y-%m-%d %H:%M' } }
        }
      })
    }
  },
  mounted () {
    this.generateChart()
  }
}
</script>
```

```
    <button @click="temperatureInCelcius = !temperatureInCelcius">Change to {{ temperatureInCelcius ? 'Fahrenheit' : 'Celcius' }}</button>
    <Chart :x="chartData.x" :y="chartData.y" />
  </div>
</template>
<script>
import Chart from '@/components/Chart.vue'
export default {
  name: 'App',
  components: {
    Chart
  },
```


Warszawa
2021-04-19
2021-04-19
temperatura
Change to Fahrenheit
17.4
17.2
17
16.8
16.6
16.4
16.2
16
15.8
2021-04-19 14:00
2021-04-19 15:00
2021-04-19
temperatura