

HTML, CSS & JS basis

Les 7:

API, AJAX, JSON



JS

JSON

API

Onderwerpen les:

- JSON/XML
- AJAX
- Wat is API
- CRUD
- Opdracht

Leerdoelen

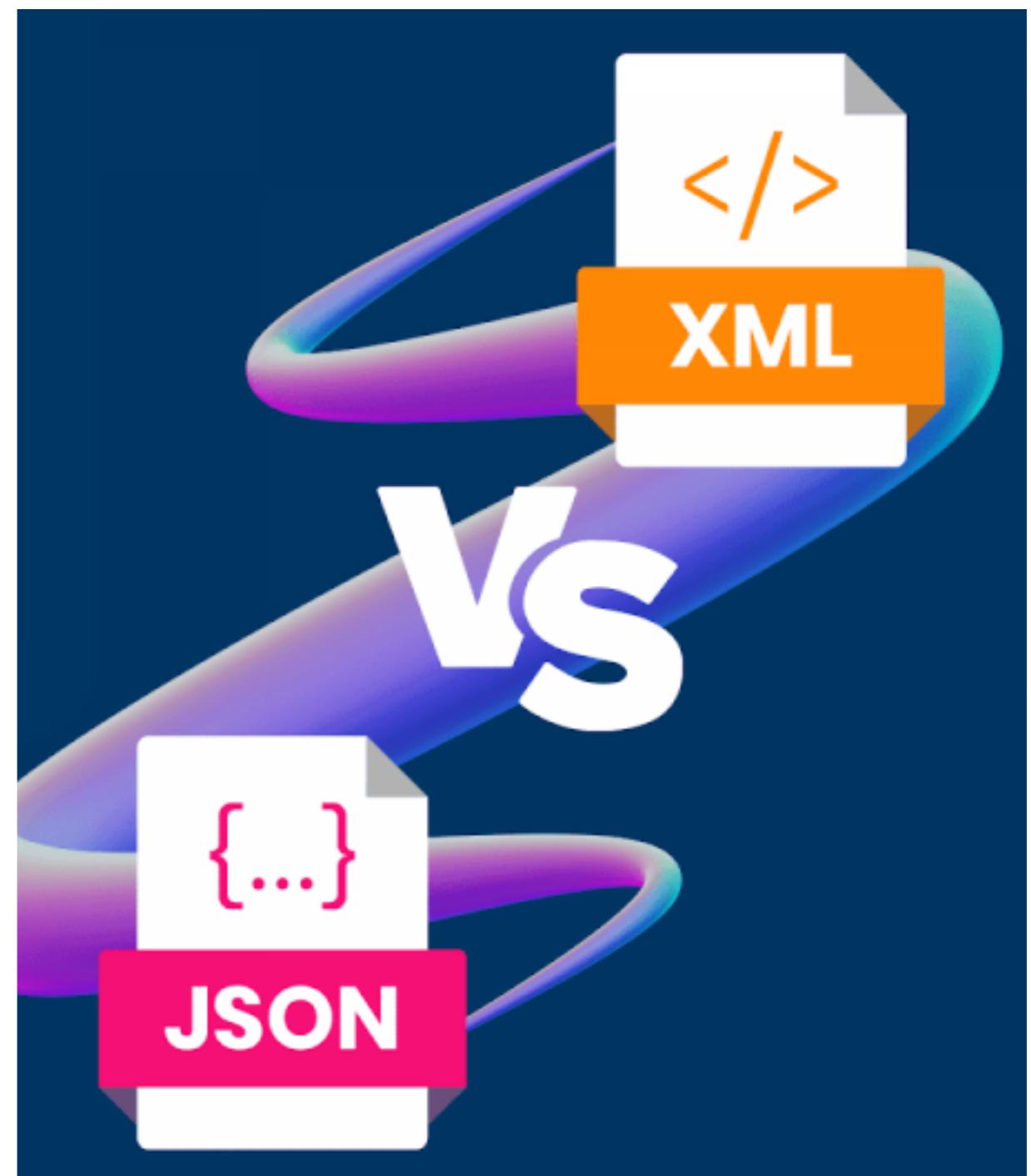
- ✓ Aan het einde van de dag kun je uitleggen wat de DOM is
- ✓ Aan het einde van de dag kan je DOM elementen selecteren en aanpassen
- ✓ Aan het einde van de dag kan je events afhandelen
- ✓ Aan het einde van de dag kun je cookies opslaan en ophalen

JSON / XML

JSON vs XML

Zowel JSON als XML kunnen worden gebruikt om data te ontvangen van een server. JSON is korter en sneller voor AJAX applicaties.

JSON format is verwant aan de code voor het maken van JavaScript objecten, om deze reden kan JavaScript makkelijk JSON data omzetten in JavaScript objecten.



JSON / XML

JSON

JSON staat voor JavaScript Object Notation en wordt gebruikt voor het versturen van data tussen computers (front end en back end).

JSON kan door veel codeertalen worden gelezen.

Syntax

De syntax van JSON lijkt veel op JavaScript object notatie:

- Data als een name/value pair.
- Data wordt gescheiden met een komma
- Objecten worden omringd door curly braces
- Square brackets geven arrays aan.
- Verschil: keys moeten strings zijn.

JSON

```
{"name":"John"}
```

JavaScript

```
{name:"John"}
```

JSON / XML

JSON

Values

JSON kent onderstaande data types:

- string (moet met dubbele quotes)
- number
- object
- array
- boolean
- null

JSON string

```
{"name":"John"}
```

JSON object

```
{
  "employee":{"name":"John", "age":30, "city":"New York"}
}
```

JSON / XML

JSON

Values

JSON array

```
{  
  "employees": ["John", "Anna", "Peter"]  
}
```

JSON boolean

```
{"sale": true}
```

JSON null

```
{"middlename": null}
```

JSON / XML

JSON

Values



JSON voorbeeld

```
{  
  "city": "Amsterdam",  
  "special": null,  
  "forecasts": [  
    {  
      "date": "2023-07-01",  
      "description": "sunny",  
      "temp": 22,  
      "danger": false  
    },  
    {  
      "date": "2023-07-02",  
      "description": "storm",  
      "temp": 21,  
      "danger": true  
    }  
  ]  
,  
  "city": "Den Haag",  
  "special": "close to sea",  
  "forecasts": [
```

JSON / XML

JSON

JSON naar JavaScript

`JSON.parse()`

Gebruikt `JSON.parse()` functie om JSON om te zetten naar een JavaScript Object.

```
const obj = JSON.parse('{"name":"John", "age":30, "city":"New York"}');
```

Gebruiken data:

```
document.getElementById("demo").innerHTML = obj.name;
```

JSON / XML

JSON

JSON naar JavaScript

JSON.parse() array

```
const candy = ['Mars', 'Bounty', 'Snicker'];
const myArr = JSON.parse(candy);
```

Gebruiken data:

```
document.getElementById("demo").innerHTML = myArr[0];
```

JSON / XML

JSON

JSON naar JavaScript

parsing dates

Aangezien datum type niet bestaat in JSON zullen we de datum string moeten omzetten.

```
const student = '{"name":"piet", "birth":"1986-12-14", "city":"Amersfoort"}';
const obj = JSON.parse(student);
obj.birth = new Date(obj.birth);
```

JSON / XML

JSON

JavaScript naar JSON

`JSON.stringify()`

Voordat we JavaScript objecten kunnen versturen naar de back end moeten we het omzetten naar JSON.

```
const obj = {name:"Piet", age:30, city:"Amersfoort"};
const myJSON = JSON.stringify(obj);
```

| https://www.w3schools.com/js/js_json_intro.asp
(zie JSON Intro tm JSON Arrays)

JSON / XML

XML

Extensible markup language.

JSON

```
{"employees": [  
    { "firstName": "John", "lastName": "Doe" },  
    { "firstName": "Anna", "lastName": "Smith" },  
    { "firstName": "Peter", "lastName": "Jones" }  
]
```

XML

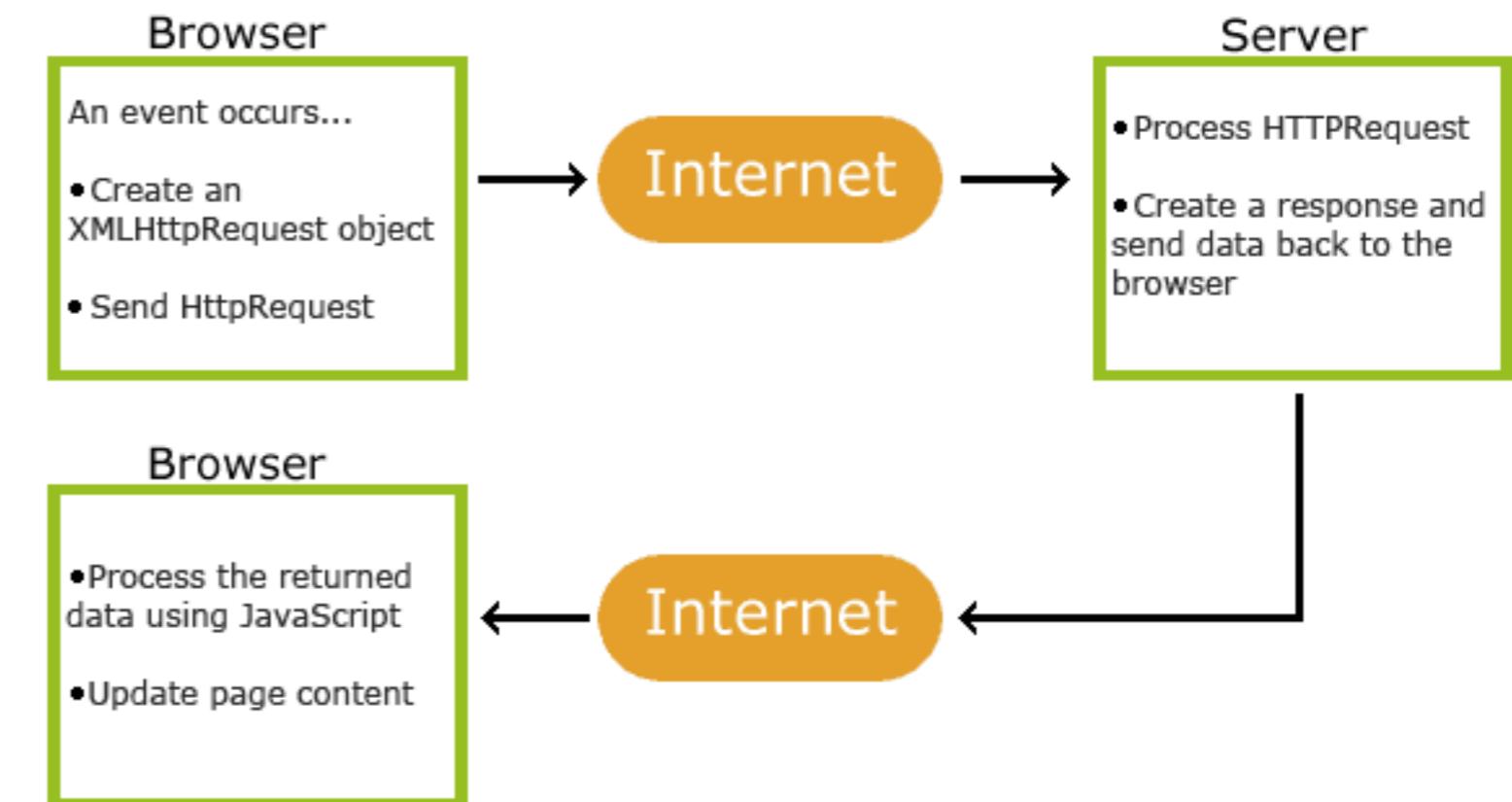
```
<employees>  
  <employee>  
    <firstName>John</firstName> <lastName>Doe</lastName>  
  </employee>  
  <employee>  
    <firstName>Anna</firstName> <lastName>Smith</lastName>  
  </employee>  
  <employee>  
    <firstName>Peter</firstName> <lastName>Jones</lastName>  
  </employee>  
</employees>
```

AJAX

Wat is AJAX

AJAX staat voor Asynchronous JavaScript And XML. AJAX is geen codeertaal het gebruikt de XMLHttpRequest object die is ingebouwd in browser waar wij in JavaScript gebruik van kunnen maken om data van een web server op te halen. En het gebruikt de HTML DOM om de data te tonen op de pagina.

De naam is misschien misleidend we hoeven niet perse XML te gebruiken voor het ophalen en versturen van data. In veel gevallen wordt juist JSON gebruikt ipv. XML.



AJAX

Wat is AJAX

Ophalen data van een webserver nadat de pagina is geladen.

Updaten content webpagina zonder de pagina te herladen.

Toesturen data naar een webserver.

AJAX

XMLHttpRequest Object

Met het XMLHttpRequest kun je data uitwisselen met een webserver.

Aanmaken XMLHttpRequest

```
const xhttp = new XMLHttpRequest();
```

Maak een Callback function

Hier komt de code die wordt uitgevoerd zodra de request klaar is.

```
xhttp.onreadystatechange = function() {  
    // code uitvoeren  
}
```

AJAX

XMLHttpRequest Object

Controleren status request

De readyState van een request gaat door 5 fases (0=unsent, 1=opened, 2=headers received, 3=loading, 4= done). Vervolgens kan het request een status hebben (100-199=informational responses, 200-299=successful responses, 300-399=redirection messages, 400-499=client error responses, 500-599=server error responses).

AJAX

XMLHttpRequest Object

Versturen request

Voor het versturen van de request open je de url van het bestand of endpoint in je backend en vervolgens stuurt je het request.

```
xhttp.open("GET", "ajax_info.txt");
xhttp.send();
```

Asynchroon true or false

Indien het een server request bevat kan je het asynchroon laten verlopen zodat andere scripts uitgevoerd kunnen worden tijdens het wachten.

```
xhttp.open("GET", "ajax_info.txt", true);
xhttp.send();
```

AJAX

XMLHttpRequest Object

Response

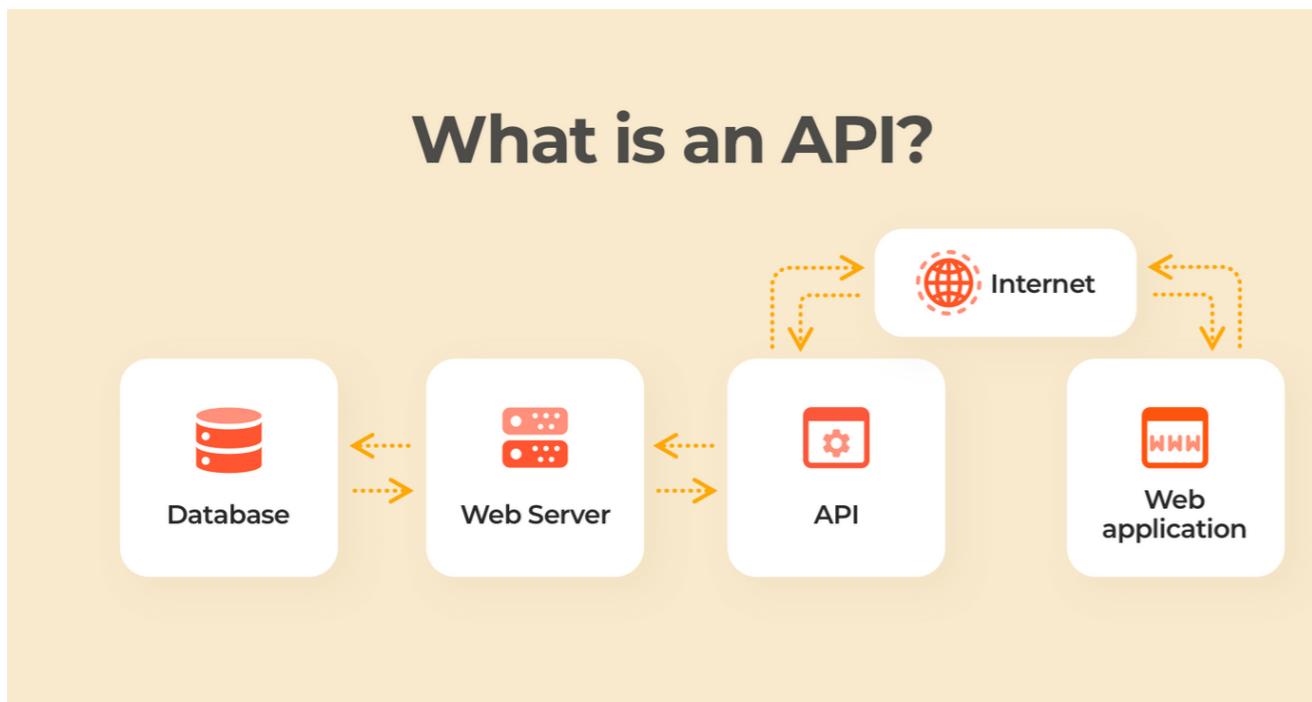
Het aangemaakte request krijgt een response van de server, deze kunnen wij vervolgens gebruiken in JavaScript. Met .responseText haal je de data op als een string.

```
xhttp.onreadystatechange = function() {
  if (xhttp.readyState === XMLHttpRequest.DONE) {
    if (xhttp.status === 'OK' || (xhttp.status >= 200 && xhttp.status < 400)) {
      let responseText = xhttp.responseText;
      // of
      let inhoudDB = JSON.parse(xhttp.responseText);
    }
  }
}
```

API

Wat is API

API staat voor Application Programming Interface, het is een manier voor twee of meer computer programma's om met elkaar te communiceren. De techniek die wij gebruiken voor het communiceren tussen de front end (website) en de back end (endpoints) is REST API (RESTful). Wanneer een request wordt gemaakt via een REST API wordt een representatie van de state van een bron (bijvoorbeeld een java of JavaScript object) verstuurd naar de aanvrager (front end) of een endpoint (backend).



CRUD

Create, read, update en delete. Via een API kun je data opvragen (read), data aanmaken (create), data veranderen (update) en data verwijderen (delete).

Crud benaming	API benaming
Create	Post
Read	Get
Update	Put & Patch (Put updates hele entiteit, patch alleen de velden die je meegeeft)
Delete	Delete

CRUD

Create / Post

Met een formulier kan je data verzamelen om te versturen naar de backend laag via REST API. Het formulier kun je een action meegeven die een JavaScript functie aanroept om een Request uit te voeren.

HTML formulier

```
<form action="javascript:postemployee()>
  <input type="text" id="name" name="name">
  <input type="number" id="salary" name="salary">
  <input type="submit" value="add employee">
</form>
```

CRUD

Create / Post

setRequestHeader

Hier maken we request headers aan. Wij geven aan ons request mee wat voor soort content wij versturen. Ook kunnen wij hier een key mee sturen voor authenticatie.

JavaScript function

```
function postemployee() {
    let name = document.getElementById('name').value;
    let salary = document.getElementById('salary').value;
    let employee = {};
    employee.name = name;
    employee.salary = salary;

    const xhttp = new XMLHttpRequest();

    xhttp.onreadystatechange = function() {
        if (xhttp.readyState === XMLHttpRequest.DONE) {
            if (xhttp.status === 'OK' || (xhttp.status >= 200 && xhttp.status < 400)) {
                console.log("POST employee success!");
            } else {
                console.log("POST employee failed!");
            }
        }
    }

    let json = JSON.stringify(employee);

    xhttp.open("POST", "http://localhost:8082/api/employee/new", true);
    xhttp.setRequestHeader("Content-type", "application/json");
    xhttp.send(json);
}
```

CRUD

Read / Get

Misschien wil je 1 of juist een lijst aan data ophalen (bijvoorbeeld werknemers), dan gebruik je Get.

JavaScript function

```
function getemployee() {
    const xhttp = new XMLHttpRequest();

    xhttp.onreadystatechange = function() {
        if (xhttp.readyState === XMLHttpRequest.DONE) {
            if (xhttp.status === 'OK' || (xhttp.status >= 200 && xhttp.status < 400)) {
                let inhoudDB = JSON.parse(xhttp.responseText);
                let employeeString = "";
                for (x=0; x<inhoudDB.length; x++) {
                    employeeString +=

                        '

### Name = ${inhoudDB[x].name}</h3> <p>Salary = ${inhoudDB[x].salary}</p> '; } document.getElementById("employees-div").innerHTML = employeeString; } else { console.log("GET employee failed!"); } } } xhttp.open("GET", "http://localhost:8082/api/employee/all/", true); xhttp.send(json); }


```

CRUD

Update / Put

Wil je iets veranderen aan de data (bijvoorbeeld de naam van een medewerker), dan gebruik je Put. De data kun je net als bij Post ophalen uit een formulier.

JavaScript function

```
function putemployee() {
    let emp-id = document.getElementById('emp-id').value;
    let name = document.getElementById('name').value;
    let salary = document.getElementById('salary').value;
    let employee = {};
    employee.name = name;
    employee.name = name;
    employee.salary = salary;

    const xhttp = new XMLHttpRequest();

    xhttp.onreadystatechange = function() {
        if (xhttp.readyState === XMLHttpRequest.DONE) {
            if (xhttp.status === 'OK' || (xhttp.status >= 200 && xhttp.status < 400)) {
                console.log("POST employee success!");
            } else {
                console.log("POST employee failed!");
            }
        }
    }

    let json = JSON.stringify(employee);

    xhttp.open("PUT", "http://localhost:8082/api/employee/id/" + emp-id, true);
    xhttp.setRequestHeader("Content-type", "application/json");
    xhttp.send(json);
}
```

CRUD

Delete

Wil je iets verwijderen uit de database (bijvoorbeeld een medewerker), dan gebruik je Delete.

JavaScript function

```
function deleteemployee() {
    const xhttp = new XMLHttpRequest();

    xhttp.onreadystatechange = function() {
        if (xhttp.readyState === XMLHttpRequest.DONE) {
            if (xhttp.status === 'OK' || (xhttp.status >= 200 && xhttp.status < 400)) {
                console.log("DELETE employee success!");
            } else {
                console.log("DELETE employee failed!");
            }
        }
    }

    xhttp.open("DELETE", "http://localhost:8082/api/employee/id/" + emp-id, true);
    xhttp.send();
}
```

Opdracht

Er zijn online veel gratis API's te vinden, waar je toegang kunt krijgen tot een database via endpoints. Zo heb je API's voor aandelen, crypto, recepten, winkel items en ook het weer.

| <https://openweathermap.org/>

Via openweather kun je het huidige weer, voorspellingen en meer ophalen. Maak hier een gratis account aan. Neem het **gratis plan met Current Weather and forecast**.

Wat ga je maken

Een pagina waar je het weer kan zien van een bepaalde locatie:

- dropdown met meerdere locaties (bv: hoofdsteden van de wereld)
- het huidige weer
- voorspelling aankomende 3 dagen

Opdracht

Design

Je bent geheel vrij om te bepalen hoe de pagina eruit komt te zien. Gebruik eventueel icoontjes voor het weer. If weather is rainy... gebruik dan een regen icoontje etc.

Pas eventueel de look aan als het weer koud is of juist warm. Plaats een foto van de gekozen stad op de pagina et cetera. Maak het een leuke spannende pagina. Zie voorbeeld:

The image shows two side-by-side weather forecast cards from a website titled "Weather all over the World".

Left Card (Amsterdam):

- City:** Amsterdam
- Current Temperature:** 4°
- Wind:** 12km/h
- Rain:** 4mm
- Feels Like:** -2°
- Forecast:** Vrijdag 02-12: -2°, Zaterdag 02-12: 4°, Zondag 02-12: 3°

Right Card (Madrid):

- City:** Madrid
- Current Temperature:** 32°
- Wind:** 12km/h
- Rain:** 4mm
- Feels Like:** -2°
- Forecast:** Vrijdag 02-12: 28°, Zaterdag 02-12: 30°, Zondag 02-12: 34°

Both cards include a small photo of the city's skyline at the top and are powered by OpenWeather.