

## Development project x

### Dropdown



Tijdens het Weldas-project heb ik een uitklapbare navigatiebalk gemaakt. Ik heb deze navigatiebalk vervolgens als basis gebruikt om verder te werken aan de navigatiebalk voor De Albatros.

```
<ul class="nav-menu">
  <div class="navBorderBottom">
    <li class="nav-item"><a href="index.html" class="nav-link"> Home</a></li>
  </div>
  <div class="navBorderBottom">
    <li class="nav-item"><a href="javascript:void(0)" class="nav-link">Aanbod</a></li>
  </div>
  <div class="navBorderBottom">
    <li class="nav-item"><a href="prices.html" class="nav-link">Tarieven</a></li>
  </div>
  <div class="navBorderBottom">
    <li class="nav-item"><a href="about.html" class="nav-link">Over ons</a></li>
  </div>
  <div class="navBorderBottom">
    <li class="nav-item"><a href="contact.html" class="nav-link">Contact</a></li>
  </div>
  <div class="navBorderBottom">
    <li class="nav-item"><a href="register.html" class="nav-link">Aanmelden</a></li>
  </div>
  <div class="navBorderBottom">
    <li class="nav-item" id="socialmediaNavbar"><a href="javascript:void(0)" class="nav-link">Social media</a></li>
  </div>
</ul>
```

Door elk element in een afzonderlijke div-tag te plaatsen, kon ik onder elk individueel vak een streep toevoegen.

#### Javascript-code 1

```
const hamburger = document.querySelector(".hamburger");
const navMenu = document.querySelector(".nav-menu");

hamburger.addEventListener("click", () => {
  hamburger.classList.toggle("active");
  navMenu.classList.toggle("active");
})

document.querySelectorAll(".nav-link").forEach(n => n.
  addEventListener("click", () => {
    hamburger.classList.remove("active");
    navMenu.classList.remove("active");
  }));
```

Allereerst worden twee elementen geselecteerd met behulp van de class 'hamburger' en 'nav-menu'. Deze elementen worden opgeslagen in de variabelen hamburger en navMenu.

Vervolgens wordt er een eventListener toegevoegd aan het hamburger-element, die reageert op het 'click' event. Wanneer er op het hamburgermenu wordt geklikt, wordt de bijbehorende functie uitgevoerd. In de functie wordt de class 'active' toegevoegd aan het hamburger-element en het navMenu-element als deze class nog niet aanwezig is. Als de class al aanwezig is, wordt deze verwijderd. Dit toggle-effect zorgt ervoor dat het hamburgermenu open en dicht kan worden geklikt.

Daarna wordt er een eventListener toegevoegd aan elk element met de class 'nav-link'. Deze eventListener reageert ook op het 'click' event. In de functie wordt de class 'active' verwijderd van zowel het hamburger-element als het navMenu-element. Dit zorgt ervoor dat wanneer er op een navigatielink wordt geklikt, het hamburgermenu wordt gesloten.

#### Javascript-code 2

```
const hamburger = document.querySelector(".hamburger");
const navMenu = document.querySelector(".nav-menu");

hamburger.addEventListener("click", () => {
  hamburger.classList.toggle("active");
  navMenu.classList.toggle("active");
});

document.querySelector(".nav-link[href='javascript:void(0)']").addEventListener("click", (event) => {
  event.preventDefault();
});

document.querySelectorAll(".nav-link:not([href='javascript:void(0)'])").forEach((link) => {
  link.addEventListener("click", () => {
    hamburger.classList.remove("active");
    navMenu.classList.remove("active");
  });
});
```

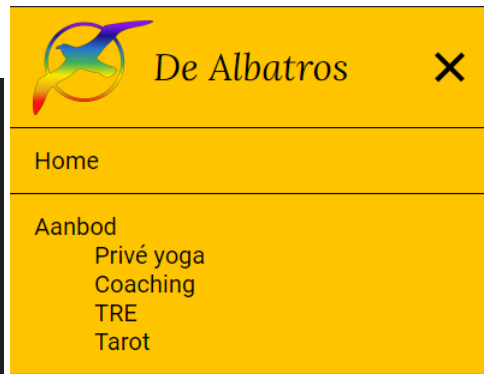
Indien er op aanbod of social media werd geklikt, sloot de navigatiebalk. Ik heb het Javascript aangepast zodat dit probleem werd voorkomen.

'document.querySelector(".nav-link[href='javascript:void(0)']").addEventListener("click", (event) => { ... })' voegt een eventListener toe aan een specifiek element dat de class 'nav-link' heeft en een 'href' met de waarde 'javascript:void(0)'. Wanneer erop wordt geklikt, wordt de bijbehorende functie uitgevoerd. In deze functie wordt het standaardgedrag van de link voorkomen met behulp van event.preventDefault().

'document.querySelectorAll(".nav-link:not([href='javascript:void(0)'])').forEach((link) => { ... })' selecteert alle elementen met de class 'nav-link' in het document, behalve degenen met een 'href' attribuut dat gelijk is aan 'javascript:void(0)'. Vervolgens wordt er voor elk van deze elementen een eventListener toegevoegd voor het 'click' event. Wanneer erop wordt geklikt, wordt de bijbehorende functie uitgevoerd. In deze functie worden de actieve classes van het hamburgericoon en het navigatiemenu verwijderd.

## Submenu

```
<div class="navBorderBottom">
  <li class="nav-item submenu-item">
    <a href="javascript:void(0)" class="nav-link">Aanbod</a>
    <ul class="submenu">
      <li><a href="#">Privé yoga</a></li>
      <li><a href="#">Coaching</a></li>
      <li><a href="#">TRE</a></li>
      <li><a href="#">Tarot</a></li>
    </ul>
  </li>
</div>
```



Ik heb een submenu toegevoegd aan de navigatiebalk met behulp van de ul-tag.

## Javascript-code 3

```
const hamburger = document.querySelector(".hamburger");
const navMenu = document.querySelector(".nav-menu");

hamburger.addEventListener("click", () => {
  hamburger.classList.toggle("active");
  navMenu.classList.toggle("active");
});

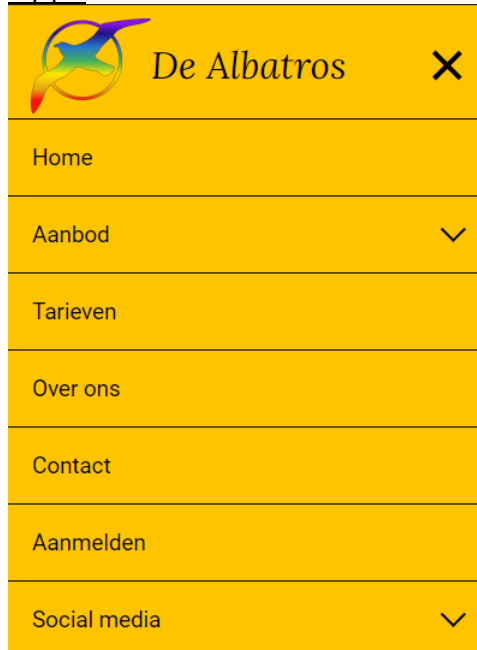
document.querySelectorAll(".nav-link").forEach((link) => {
  link.addEventListener("click", (event) => {
    const submenuItem = link.parentElement;
    const submenu = submenuItem.querySelector(".submenu");
    if (submenu) {
      event.preventDefault();
      submenu.classList.toggle("active");
    } else {
      hamburger.classList.remove("active");
      navMenu.classList.remove("active");
    }
  });
});
```

Om het submenu uit te kunnen klappen, zijn er enkele wijzigingen aangebracht in de JavaScript-code.

In plaats van het toevoegen van een specifieke eventListener aan een element met een bepaalde 'href' waarde, wordt in deze code een eventListener toegevoegd aan alle elementen met de class 'nav-link'.

De functie die wordt uitgevoerd wanneer op deze link wordt geklikt, bevat extra stappen. Ten eerste wordt het ouder-element van de link (submenulitem) gezocht en vervolgens wordt er gecontroleerd of er een element met de class 'submenu' aanwezig is binnen dit submenulitem. Als er een submenu-element is gevonden, wordt het standaardgedrag van de link voorkomen met behulp van event.preventDefault(). Daarna wordt de actieve class van het submenu-element veranderd. Als er geen submenu-element is gevonden, wordt de class active van de elementen hamburger en navMenu verwijderd.

## Pijltjes



```
.arrow {  
  width: 10px;  
  height: 10px;  
  border-left: 2px solid black;  
  border-bottom: 2px solid black;  
  transform: rotate(-45deg);  
  display: inline-block;  
  vertical-align: middle;  
  position: fixed;  
  right: 20px;  
}
```

Om een pijltje aan het menu toe te voegen, heb ik een element aangemaakt met een afmeting van 10x10 pixels (smashtheshell, z.d.). Vervolgens heb ik aan de linker- en onderkant van het element een border toegevoegd. Om het pijltje naar beneden te laten wijzen, heb ik het element gedraaid met behulp van 'transform: rotate(-45deg)'.

```
.arrow {  
  width: 10px;  
  height: 10px;  
  border-left: 2px solid black;  
  border-bottom: 2px solid black;  
  transform: rotate(-45deg);  
  display: inline-block;  
  vertical-align: middle;  
  position: fixed;  
  right: 20px;  
}
```

```
.arrow {  
  width: 10px;  
  height: 10px;  
  border-left: 2px solid black;  
  border-bottom: 2px solid black;  
  transform: rotate(-45deg);  
  vertical-align: middle;  
  float: right;  
  margin-right: 15px;  
}
```

Het pijltje in het menu bewoog tijdens het scrollen van de pagina door het gebruik van 'position: fixed'. Om dit probleem op te lossen, heb ik de code aangepast. Ik heb de 'position' eigenschap en het 'display: inline-block' verwijderd. Daarentegen heb ik 'float: right' toegevoegd en de marges aangepast. Hierdoor blijft het element op dezelfde plek in het menu zichtbaar en verandert het niet meer van positie.

## Desktop



Ik heb me gefocust op mobile first. Hierdoor is het op de desktop nog niet werkend.

## Navigatiebalk met video

```
<div class="navBorderBottom">
  <li class="nav-item submenu-item">
    <a href="javascript:void(0)" class="nav-link">Aanbod</a>
    <ul class="submenu">
      <li><a href="#">Privé yoga</a></li>
      <li><a href="#">Coaching</a></li>
      <li><a href="#">TRE</a></li>
      <li><a href="#">Tarot</a></li>
    </ul>
  </li>
</div>
```

```
<nav class="navmenu">
  <ul>
    <li class="nav-item"><a href="index.html" class="nav-link"> Home</a></li>
    <li class="nav-item"><a href="javascript:void(0)" class="nav-link">Aanbod <span class="arrow"></span></a>
      <ul class="submenu">
        <li><a href="#">Privé yoga</a></li>
        <li><a href="#">Coaching</a></li>
        <li><a href="#">TRE</a></li>
        <li><a href="#">Tarot</a></li>
      </ul>
    </li>
    <li class="nav-item"><a href="prices.html" class="nav-link">Tarieven</a></li>
    <li class="nav-item"><a href="about.html" class="nav-link">Over ons</a></li>
    <li class="nav-item"><a href="contact.html" class="nav-link">Contact</a></li>
    <li class="nav-item"><a href="register.html" class="nav-link">Aanmelden</a></li>
    <li class="nav-item"><a href="javascript:void(0)" class="nav-link">Social media <span class="arrow"></span></a>
      <ul class="submenu">
        <li><a href="#">Facebook</a></li>
        <li><a href="#">Instagram</a></li>
        <li><a href="#">WhatsApp</a></li>
      </ul>
    </li>
  </ul>
</nav>
```

Aangezien het mij niet lukte om de navigatiebalk met de gewenste opmaak te krijgen ben ik een YouTube video gaan zoeken om uit te vinden hoe ik het probleem kon oplossen (smashtheshell, 2020).

Ik heb de li-elementen uit de losse divs 'navBorderBottom' gehaald. Dit is nodig om ervoor te zorgen dat het dropdown menu geplaatst kan worden tussen de andere items in. Verder hebben alle li-elementen dezelfde classnaam, namelijk 'nav-item'.

```
header .navmenu ul li ul {
  position: absolute;
  left: 0;
  width: 200px;
  display: none;
}
```

De links in het submenu zijn verborgen door middel van 'display: none'.

```
<input type="checkbox" id="menubar">
```

```
#menubar {
  display: none;
}
```

Ik heb een checkbox toegevoegd aan mijn code, waarmee ik het hamburgermenu kan openen zonder dat ik JavaScript hoeft te gebruiken. Om de checkbox te verbergen, heb ik de eigenschap 'display: none' gebruikt.

```
<label for="menubar">
  <div class="hamburger">
    <span class="bar"></span>
    <span class="bar"></span>
    <span class="bar"></span>
  </div>
</label>
```

```
header label {
  display: block;
  position: relative;
}
```

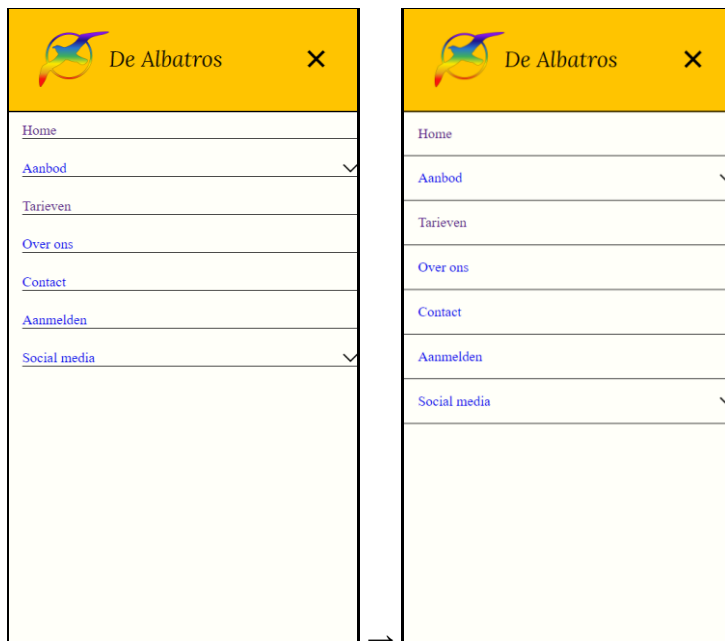
Daarnaast heb ik een label toegevoegd waarin de hamburger-div is geplaatst. Om ervoor te zorgen dat het label altijd zichtbaar blijft, zelfs wanneer het navigatiemenu is ingeklapt, heb ik het buiten het navmenu-element geplaatst. In de mediaquery heb ik de eigenschap van het label aangepast naar 'display: block', zodat het zichtbaar wordt op mobiele apparaten.

```
#menubar:checked ~ .navmenu{
  display: block;
}
```

Bij het klikken op het label wordt de checkbox ingeschakeld of uitgeschakeld. Als de checkbox is ingeschakeld, wordt de eigenschap 'display' van het navmenu gewijzigd naar 'block', waardoor het menu zichtbaar wordt op het scherm.

```
header .navmenu ul li:focus-within > ul,
header .navmenu ul li:hover > ul{
  display: initial;
}
```

Deze code zorgt ervoor dat wanneer de cursor over een ul li element beweegt en er een onderliggende ul aanwezig is (hover), of wanneer erop wordt geklikt (focus-within), de ul die zich daarin bevindt de eigenschap 'display: initial' krijgt. Hierdoor wordt het submenu zichtbaar binnen het menu.



```

.nav-item {
  margin: 12px 16px;
  float: left;
  width: 96%;
}

```

→

```

.nav-item {
  padding: 16px 0 16px 16px;
  width: 96%;
}

```

Ik heb de lijn aangepast zodat deze beter aansluit op de zijkant, Dit heb ik gedaan door de margin te verwijderen en padding toe te voegen. Om te voorkomen dat er witruimte ontstaat naast de gele balk, heb ik aan de rechterkant de padding op 0 gezet. Bovendien heb ik 'float: left' verwijderd, omdat dit geen toegevoegde waarde had.



```

header .navmenu {
  position: absolute;
  top: 119px;
  left: 0;
  right: 0;
  border-top: 1px solid black;
  display: flex;
  justify-content: space-between;
}

header .navmenu ul li {
  flex: 1;
  list-style: none;
  border-bottom: 1px solid black;
}

header .navmenu ul li a {
  text-decoration: none;
  display: block;
  height: 100%;
}

```

→

```


header .navmenu {
  position: absolute;
  top: 119px;
  left: 0;
  right: 0;
  border-top: 1px solid black;
  display: flex;
  padding: 0;
}

header .navmenu ul li {
  flex: 1;
  list-style: none;
  border-bottom: 1px solid black;
  margin: 0;
  padding: 0;
}

header .navmenu ul li a {
  text-decoration: none;
  display: block;
  padding: 16px 0 16px 16px;
}

```

Ik heb het uitklapmenu verbeterd door verdere aanpassingen in de styling door te voeren. Door het verwijderen van 'justify-content' en het toevoegen van padding aan het navigatiemenu, zien de elementen er beter uit. Daarnaast heb ik padding toegepast op 'ul li a', zodat er ruimte tussen de elementen en de lijn mooi vormgegeven is.

 De Albatros X	
Home	
Aanbod	▼
Privé yoga	
Coaching	
TRE	
Tarot	
Tarieven	
Over ons	
Contact	
Aanmelden	
Social media	▼

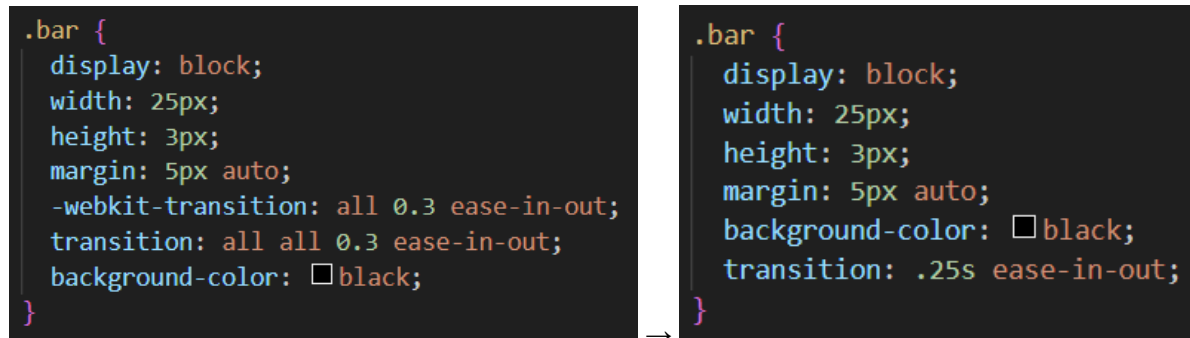
Uiteindelijk heb ik nog verschillende aanpassingen in de CSS doorgevoerd, zoals het gebruik van percentages verminderen. Ook heb ik margin en padding toegevoegd, kleuren aangepast en het lettertype gewijzigd. Dit is het uiteindelijke resultaat van mijn aanpassingen.



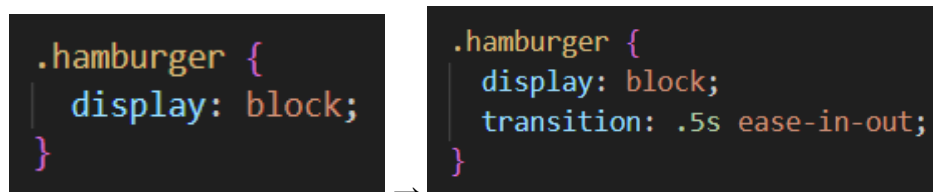
## Animatie hamburgermenu

Op de mobiele website van de Efteling heb ik een vloeiend overgangseffect gezien bij het hamburgermenu (Efteling, z.d.). De drie streepjes veranderden geleidelijk in een kruisje, in plaats van directe verandering. Zo'n effect wilde ik ook toevoegen aan de website.

Om zo'n effect te creëren heb ik gebruik gemaakt van een voorbeeld van Couch (z.d.).



'transition: .25s ease-in-out' definieert een overgangsanimatie met een duur van 0.25 seconden. Het zorgt voor een soepele en vloeiende overgangseffect.



Ik heb 'ease-in-out' toegevoegd aan de hamburgerclass. Hierdoor duurt de overgang van de drie streepjes naar een kruisje in totaal 0.5 seconden.

## Animatie navigatiebalk

```
#menubar:checked ~ .navmenu{  
  display: block;  
}
```

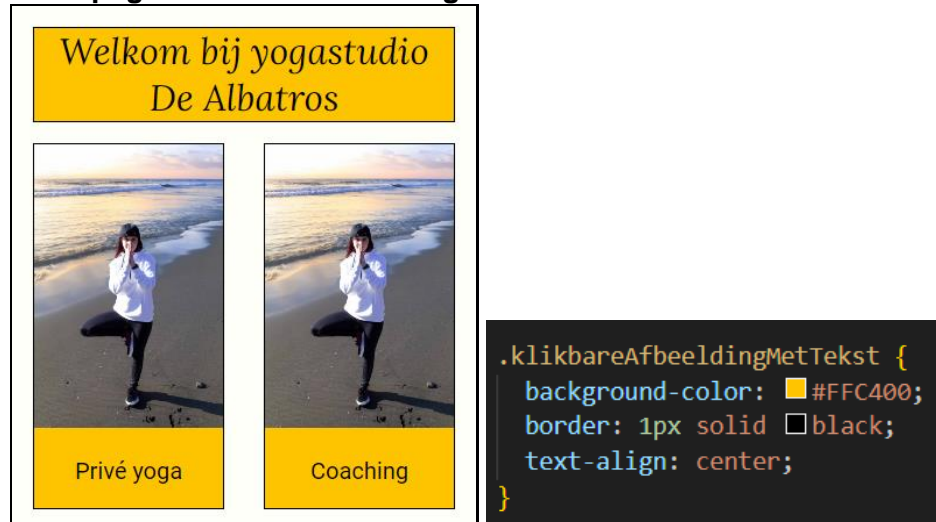
→

```
#menubar:checked ~ .navmenu{  
  display: block;  
  animation: animationNavbar 1s;  
}
```

```
@keyframes animationNavbar {  
  from {  
    opacity: 0;  
  }  
  to {  
    opacity: 1;  
  }  
}
```

Ik heb een keyframe-animatie genaamd 'animationNavbar' gemaakt, waarbij een element geleidelijk van 0 naar 1 opaciteit gaat (volledig zichtbaar wordt). Deze keyframe-animatie heb ik gekoppeld aan de navigatiebalk door middel van 'animation: animationNavbar 1s'. Wanneer de checkbox is aangevinkt, zal de animatie starten en zal de navigatiebalk binnen 1 seconde van volledig onzichtbaar naar volledig zichtbaar overgaan.

## Homepagina klikbare afbeelding



```
<div class="klikbareAfbeeldingen-container">  
  <a href="privateYoga.html">  
    <div class="klikbareAfbeeldingMetTekst">  
        
      <p>Privé yoga</p>  
    </div>  
  </a>  
  
  <a href="coaching.html">  
    <div class="klikbareAfbeeldingMetTekst">  
        
      <p>Coaching</p>  
    </div>  
  </a>  
</div>
```

Ik heb een div-tag gemaakt die de klikbare afbeeldingen bevat. Elke afbeelding heeft een unieke 'href'-tag. Binnen deze 'href'-tag bevindt zich een 'div' met zowel de afbeelding als de tekst. Om duidelijkheid te creëren en de elementen visueel te groeperen, heb ik een zwarte lijn toegevoegd.



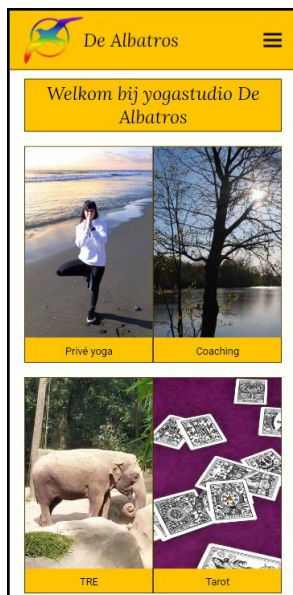
Om de ruimte rondom de tekst te verminderen in de class 'klikbareAfbeeldingMetTekst', heb ik de margin op het p-element aangepast naar 4px. Dit heeft geresulteerd in een smaller gele gebied rondom de tekst.



De afbeeldingen op de website waren niet responsive op verschillende apparaten, waardoor er bij grotere mobiele schermen veel witruimte tussen de afbeeldingen ontstond.

```
.klikbareAfbeeldingen {
  width: 130px;
}
→
.klikbareAfbeeldingen {
  width: 100%;
}
```

Ik heb de grootte van de afbeeldingen aangepast van pixelwaarden naar percentages.



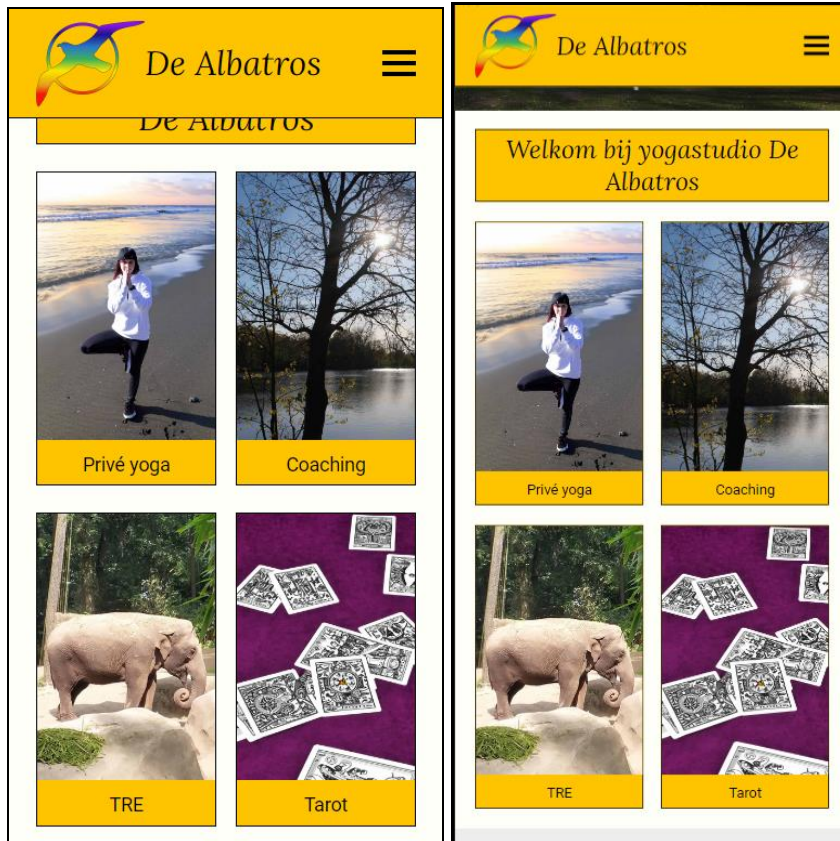
Als gevolg hiervan zijn de afbeeldingen naast elkaar gaan staan zonder tussenruimte.

```
.klikbareAfbeeldingen-container {
  display: flex;
  justify-content: space-between;
  flex-direction: row;
  margin: 20px;
}
```

→

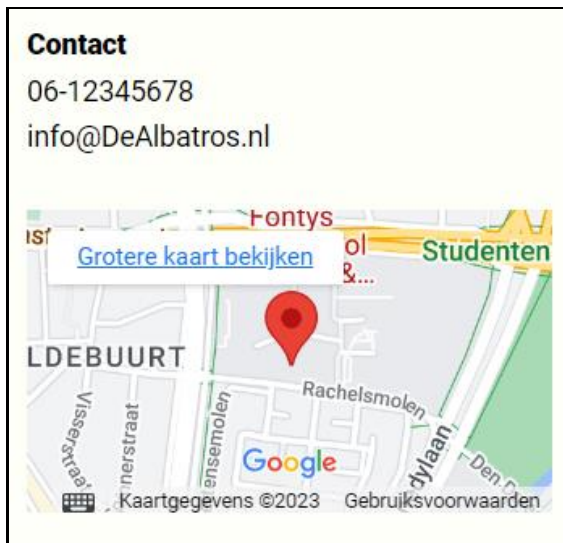
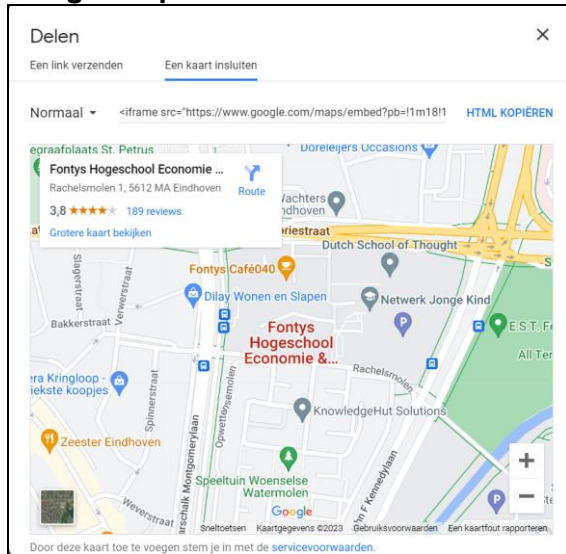
```
.klikbareAfbeeldingen-container {
  display: flex;
  justify-content: space-between;
  flex-direction: row;
  gap: 5%;
  margin: 20px;
}
```

Om dit probleem op te lossen, heb ik een gap toegevoegd tussen de afbeeldingen. Deze tussenruimte is ingesteld op een percentage, zodat de afbeeldingen zich kunnen aanpassen aan verschillende apparaten.



Dankzij de aanpassing worden de afbeeldingen nu responsief weergegeven op verschillende mobiele apparaten.

## Google maps locatie



Via Google Maps kun je een iframe verkrijgen van een specifieke locatie, die je op je website kunt plaatsen (Google, z.d.). De gekopieerde iframe is aangepast qua breedte en hoogte, zodat de kaart met de locatie zichtbaar is op de website en passend wordt uitgelijnd.

## API

Om de kaart toe te voegen met een zoomfunctie, is het volgende script nodig:

```
<script src="https://maps.googleapis.com/maps/api/js?key=YOUR_API_KEY"></script>
```

In de src moet 'YOUR\_API\_KEY' vervangen worden door je eigen Google Maps API-sleutel. Deze is te verkrijgen via je Google Cloud Console (zie verderop).

```
function initMap() {  
  var myLatLng = {lat: 51.436687, lng: 5.473912}; // Coördinaten van Fontys Rachelsmolen  
  
  var map = new google.maps.Map(document.getElementById('map'), {  
    center: myLatLng,  
    zoom: 15  
  });  
  
  var marker = new google.maps.Marker({  
    map: map,  
    position: myLatLng,  
    title: 'Fontys Rachelsmolen'  
  });  
}  
initMap();
```

De functie `initMap() { }` wordt gebruikt om de kaart te initialiseren en alle benodigde instellingen in te stellen.

`var myLatLng = {lat: 51.436687, lng: 5.473912}` creëert een variabele genaamd 'myLatLng' die de coördinaten van een specifieke locatie bevat. In dit geval zijn de coördinaten voor Fontys Rachelsmolen in Eindhoven.

Met `var map = new google.maps.Map(document.getElementById('map'), { })` wordt een nieuw kaartobject gemaakt met behulp van de Google Maps API. De kaart wordt toegevoegd aan het HTML-element met de id 'map'. Het tweede argument bevat enkele opties voor de kaart, zoals het centrum van de kaart (in dit geval de opgegeven coördinaten) en het zoomniveau.



var marker = new google.maps.Marker({ }) maakt een marker op de kaart. De marker wordt geplaatst op de opgegeven coördinaten en krijgt een titel.


initMap() roept de functie 'initMap' aan, zodat de kaart en marker daadwerkelijk worden weergegeven wanneer de webpagina wordt geladen.

### Verify your account to use Maps API or service

Please verify your account to make sure you're not a robot. You'll need to verify your phone number and payment details, but won't be charged until you've used more than \$200/mo usage on Maps Platform and you upgrade to paid account.

[VERIFY ACCOUNT](#)

#### How you pay

 Monthly automatic payments

You pay for this service on a regular monthly basis, via an automatic charge when your payment is due.

#### Payment method ⓘ

Card number

#

Card number is required

MM / YY CVC

Cardholder name

☒ Credit or debit card address is same as above

The personal information you provide here will be added to your payments profile. It will be stored securely and treated in accordance with the [Google Privacy Policy](#).


Om een API aan te maken voor het toevoegen van een kaart met zoomfunctie aan een website, is het nodig om gebruik te maken van de Google Cloud Console. Om toegang te krijgen tot de console, moet er een account worden aangemaakt. Hoewel de eerste 200 dollar per maand gratis zijn, is het vereist om een creditcard te koppelen aan het account om het aan te maken.

Aangezien ik geen creditcard heb, kan ik geen account aanmaken en dus ook geen gebruik maken van een Maps API. Daarom is het voor mij niet mogelijk om een kaart met zoomfunctie toe te voegen aan de website. Ik heb gekeken of er een mogelijkheid is om gratis gebruik te maken van de Google Maps API, echter biedt Google sinds 2018 geen ondersteuning meer voor een gratis Google Maps API (Lanser, 2019).

## FAQ met grid

Om een accordion te maken met behulp van CSS Grid, heb ik de video van Kevin Powell (2023) gekeken.

CSS Grid is een lay-outmodule in CSS waarmee je een raster kunt creëren van rijen en kolommen. Het stelt je in staat om elementen op een flexibele en responsieve manier te positioneren en te ordenen op een webpagina. Met CSS Grid kun je de grootte en positie van cellen in het raster bepalen en ook de ruimte tussen de cellen instellen. Het biedt een eenvoudige syntax en maakt complexe layouts mogelijk zonder afhankelijk te zijn van externe frameworks of JavaScript.

**De Albatros** 

**Veelgestelde vragen**

**Wat is yoga en wat zijn de voordelen ervan?**

Yoga combineert fysieke houdingen, ademhalingsoefeningen en meditatie voor balans en harmonie in lichaam en geest. Voordelen zijn onder andere verbeterde flexibiliteit, kracht, balans, stressvermindering, ontspanning en algeheel welzijn. Het helpt ook bij ademhaling, spierversterking en het verminderen van spanning in het lichaam.

**Welke soorten sessies bieden jullie aan?**

**Wat moet ik meenemen naar een yogales?**

**Kan ik een les inhalen als ik een keer niet kan komen?**

```
<div class="accordion-container">  
  <div class="accordion-panel">  
    <button class="accordionButton" aria-expanded="false" >  
      Wat is yoga en wat zijn de voordelen ervan?  
    </button>  
    <div class="accordion-content" aria-hidden="true" >  
      <div>  
        <p>Yoga combineert fysieke houdingen, ademhalingsoefeningen en meditatie voor balans en harmonie in lichaam en geest. Voordelen zijn onder andere verbeterde flexibiliteit, kracht, balans, stressvermindering, ontspanning en algeheel welzijn. Het helpt ook bij ademhaling, spierversterking en het verminderen van spanning in het lichaam.</p>  
      </div>  
    </div>  
  </div>  
  
  <div class="accordion-panel">  
    <button class="accordionButton" aria-expanded="false" >  
      Welke soorten sessies bieden jullie aan?  
    </button>  
  </div>  
</div>
```

Voor het ontwikkelen van een accordion in HTML heb ik één div-tag aangemaakt, met de naam 'accordion-container'. Deze container bevat alle elementen die bij de accordion horen. Elk vraag-antwoordpaar is vervolgens in een aparte div-tag geplaatst, genaamd 'accordion-panel'. Binnen elke 'accordion-panel' heb ik een button met de naam 'accordionButton' gemaakt, waarin de vraag wordt weergegeven. Daarnaast is er een div-tag met de naam 'accordion-content' toegevoegd, waarin het antwoord is geplaatst.

In de bovenstaande code wordt gebruikgemaakt van de attributen 'aria-expanded' en 'aria-hidden'. Het attribuut 'aria-expanded=false' wordt toegepast op de button met de class 'accordionButton'. Hiermee wordt aangegeven dat het element initieel niet is uitgeklaapt. Daarnaast wordt het attribuut 'aria-hidden=true' gebruikt voor de div met de class 'accordion-content'. Dit attribuut geeft aan dat het element initieel verborgen is.

```
.accordion-content {  
  display: grid;  
  grid-template-rows: 0fr;  
  transition: grid-template-rows 500ms;  
}
```

Met 'display: grid' wordt aangegeven dat het element een CSS Grid-container is. Dit maakt het mogelijk om de inhoud en positie van elementen binnen het element te organiseren met behulp van een grid.



De 'grid-template-rows: 0fr' bepaalt de hoogte van de rijen in het grid. Hier wordt slechts één rij gedefinieerd en de hoogte wordt ingesteld op 0fr, wat betekent dat de rij geen zichtbare hoogte heeft. Door een rij met een hoogte van 0fr toe te voegen aan het CSS-grid van de accordion en deze te verbergen wanneer de accordion is ingeklapt, zorg ik ervoor dat er geen ruimte wordt ingenomen door de verborgen inhoud.

Met 'transition: grid-template-rows 500ms' wordt een overgangseffect gedefinieerd voor de CSS-eigenschap 'grid-template-rows'. De duur van de overgang is ingesteld op 500 milliseconden, waardoor aanpassingen in de hoogte van de rijen vloeiend en geanimeerd worden.

```
.accordion-content[aria-hidden="false"] {  
  grid-template-rows: 1fr;  
  background-color: #EDED;   
  padding: 0 10px;  
}
```

Wanneer 'aria-hidden=false' van toepassing is op een element met de class 'accordion-content', zal het CSS-grid van dat element rijen hebben met een hoogte van 1fr. De inhoud van de geopende sectie in de accordion neemt de beschikbare ruimte in en past zich automatisch aan aan de grootte van het grid.

```
.accordion-content > div {  
  overflow: hidden;  
}
```

De CSS-selector `.accordion-content > div` wordt gebruikt om specifieke stijlen toe te passen op directe div-elementen die zich binnen elementen bevinden met de class 'accordion-content'.

De 'overflow: hidden' geeft aan dat eventuele overlopende inhoud binnen de geselecteerde div-elementen verborgen moet worden. Dit betekent dat de inhoud die groter is dan de beschikbare ruimte binnen het element niet zichtbaar zal zijn.

```
const accordion = document.querySelector(".accordion-container");  
  
accordion.addEventListener("click", (e) => {  
  const activePanel = e.target.closest(".accordion-panel");  
  if (!activePanel) return;  
  toggleAccordion(activePanel);  
});
```

`const accordion = document.querySelector(".accordion-container")` maakt een variabele aan genaamd 'accordion' en wijst deze toe aan het eerste element dat overeenkomt met '.accordion-container'.

`accordion.addEventListener("click", (e) => { ... })` voegt een eventListener toe aan het accordion-element voor het 'click' event. Dit betekent dat wanneer er ergens in het accordion-element wordt geklikt, de bijbehorende functie wordt uitgevoerd.

`(e) => { ... }` is een arrow functie die als callback fungeert voor het click event. De parameter 'e' bevat informatie over het event, zoals de bron van de klik.

`const activePanel = e.target.closest(".accordion-panel")` maakt een variabele genaamd 'activePanel' en wijst deze toe aan het dichtstbijzijnde (closest) element dat overeenkomt met '.accordion-panel'. Dit element bevindt zich binnen het element waarop geklikt is, zoals gedefinieerd door `e.target`.

Als er geen overeenkomstig paneel wordt gevonden, wordt de functie vroegtijdig verlaten met behulp van 'if (!activePanel)' return. Dit betekent dat er geen verdere actie wordt ondernomen.

toggleAccordion(activePanel) roept de functie toggleAccordion aan en het gevonden paneel (activePanel) wordt doorgegeven als argument.

```
function toggleAccordion(panelToActivate) {
  const activeButton = panelToActivate.querySelector("button");
  const activePanel = panelToActivate.querySelector(".accordion-content");
  const activePanelIsOpened = activeButton.getAttribute("aria-expanded");

  if (activePanelIsOpened === "true") {
    panelToActivate
      .querySelector("button")
      .setAttribute("aria-expanded", false);

    panelToActivate
      .querySelector(".accordion-content")
      .setAttribute("aria-hidden", true);
  } else {
    panelToActivate.querySelector("button").setAttribute("aria-expanded", true);

    panelToActivate
      .querySelector(".accordion-content")
      .setAttribute("aria-hidden", false);
  }
}
```

De 'toggleAccordion' functie wordt gebruikt om een accordionpaneel te openen of te sluiten. Het eerste deel van de code selecteert het knopelement en het paneel binnen het paneel dat geactiveerd moet worden. Vervolgens wordt de waarde van het 'aria-expanded' attribuut van het knopelement opgehaald en toegewezen aan de variabele 'activePanellsOpened'.

Als 'activePanellsOpened' gelijk is aan 'true', wordt het paneel als geopend beschouwd. In dit geval wordt het 'aria-expanded' attribuut van het knopelement ingesteld op 'false' en het 'aria-hidden' attribuut van het paneel ingesteld op 'true'. Hierdoor wordt het paneel gesloten.

Als 'activePanellsOpened' niet gelijk is aan 'true', betekent dit dat het paneel gesloten is. In dit geval wordt het 'aria-expanded' attribuut van het knopelement ingesteld op 'true' en het 'aria-hidden' attribuut van het paneel ingesteld op 'false'. Hierdoor wordt het paneel geopend.

```

let activePanel = null;

accordion.addEventListener("click", (e) => {
  const clickedPanel = e.target.closest(".accordion-panel");
  if (!clickedPanel) return;

  if (activePanel === clickedPanel) {
    toggleAccordion(clickedPanel);
    activePanel = null;
  } else {
    if (activePanel) {
      toggleAccordion(activePanel);
    }
    toggleAccordion(clickedPanel);
    activePanel = clickedPanel;
  }
});

```

Deze code heb ik toegevoegd om ervoor te zorgen dat een actief accordionpaneel wordt gesloten wanneer er op een nieuw paneel wordt geklikt.

let activePanel = null zet de variabele 'activePanel' als null, wat betekent dat er geen actief paneel is.

accordion.addEventListener("click", (e) => { ... }) voegt een eventListener toe aan het 'accordion'-element voor het 'click' event. Dus wanneer er ergens in het 'accordion'-element wordt geklikt, wordt de bijbehorende functie uitgevoerd.

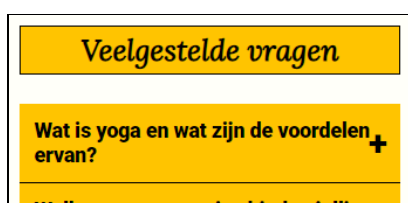
(e) => { ... } is een arrow functie die als callback fungeert voor het click event. De parameter e bevat informatie over het event, zoals de bron van de klik.

const clickedPanel = e.target.closest(".accordion-panel") maakt de variabele 'clickedPanel' en wordt toegewezen aan het dichtstbijzijnde (closest) element dat overeenkomt met '.accordion-panel'. Dit element is het specifieke paneel waarop is geklikt.

Als clickedPanel niet waar is, wordt de functie vroegtijdig verlaten (return) zonder verdere actie te ondernemen.

Als activePanel gelijk is aan clickedPanel, betekent dit dat hetzelfde paneel opnieuw is geklikt. In dit geval wordt de functie aangeroepen om het paneel te openen of te sluiten. Vervolgens wordt de variabele 'activePanel' weer op 'null' gezet om aan te geven dat er geen actief paneel meer is.

Als het huidige actieve paneel niet gelijk is aan het paneel waarop is geklikt, betekent dit dat er op een ander paneel is geklikt. In dit geval wordt eerst gecontroleerd of er al een actief paneel is. Als dit het geval is, wordt de functie opgeroepen om het actieve paneel te sluiten. Vervolgens wordt de functie opgeroepen voor het nieuw geklikte paneel om het te openen. Ten slotte wordt de variabele 'activePanel' bijgewerkt met het nieuw geklikte paneel.



```

<button class="accordionButton" aria-expanded="false" >
  <span class="accordionButton-label">
    Wat is yoga en wat zijn de voordelen ervan?
  </span>
  <span class="accordionButton-icon"></span>
</button>

```

`<span class="accordionButton-label"></span>` creëert een span-element met de class 'accordionButton-label' dat de vraag bevat. Dit span-element wordt gebruikt om de tekst van de accordeonknop weer te geven.

`<span class="accordionButton-icon"></span>` creëert een leeg span-element met de class 'accordionButton-icon'. Dit span-element wordt gebruikt om een plus- of minteken naast de tekst van de knop weer te geven.

```
.accordionButton[aria-expanded="false"] .accordionButton-icon::after {
  content: "+";
}

.accordionButton[aria-expanded="true"] .accordionButton-icon::after {
  content: "-";
}
```

`.accordionButton[aria-expanded="false"] .accordionButton-icon::after` selecteert het pseudo-element '::after' van het element met de class 'accordionButton' waarvan het attribuut 'aria-expanded' de waarde 'false' heeft. Dit betekent dat dit van toepassing is op de knoppen van gesloten accordeonpanelen. 'content: "+"' stelt de inhoud van het pseudo-element '::after' in op het plusteken.

`.accordionButton[aria-expanded="true"] .accordionButton-icon::after` selecteert het pseudo-element '::after' van het element met de class 'accordionButton' waarvan het attribuut 'aria-expanded' de waarde 'true' heeft. Dit betekent dat dit van toepassing is op de knoppen van geopende accordeonpanelen. 'content: "-"' stelt de inhoud van het pseudo-element '::after' in op het minteken.

### Veelgestelde vragen

Wat is yoga en wat zijn de voordelen ervan?	+
Welke soorten sessies bieden jullie aan?	+
Wat moet ik meenemen naar een yogales?	+
Kan ik een les inhalen als ik een keer niet kan komen?	+
Moet ik flexibel zijn om yoga te beoefenen?	+
Kan ik iemand meenemen?	+

```
.accordionButton-label {
  display: inline-block;
  padding-right: 20px;
}
```

De padding is vergroot om ervoor te zorgen dat het gemakkelijk is om op elke mobiele telefoon het accordion aan te klikken.

## Tarotkaart trekken

```
[
  {
    "id": 1,
    "picture": "deZegewagen.png",
    "title": "De Zegewagen",
    "kernwoorden": "Wilskracht + Volharding + Controle + Succes",
    "kaart": "De Zegewagen is een kaart die staat voor wilskracht en actie, zoals weergegeven op de afbeelding.",
    "betekenis": "Als je een boost voor je zelfvertrouwen nodig hebt, komt de Zegewagen je te hulp."
  },
  {
    "id": 2,
    "picture": "bekers2.png",
    "title": "Twee van Bekers",
    "kernwoorden": "Verbinding + Partnerschap + Harmonie + Samenwerking + Gelijkaardigheid",
    "kaart": "Een man en vrouw staan tegenover elkaar en bieden elkaar hun beker aan.",
    "betekenis": "De Twee van Bekers staat voor een wederzijds liefdevol en gunstig partnerschap."
  },
  {
    "id": 3,
    "picture": "koningVanPentakels.png",
    "title": "Koning van Pentakels",
    "kernwoorden": "Materieel succes + Ambitie + Vaardigheden + Zaken + Overvloed",
    "kaart": "De Koning van Pentakels is een tevreden man die zijn plek op de troon heeft gevonden.",
    "betekenis": "Wanneer de Koning van Pentakels verschijnt, zit je waarschijnlijk in een positieve situatie."
  },
  {
    "id": 4
  }
]
```

De tarotkaarten bestaan uit een totaal van 78 kaarten, waarbij elke kaart zowel rechtop als omgekeerd kan worden getrokken. Mijn focus ligt op de functie van het trekken van een kaart. Hierdoor heb ik een JSON-bestand aangemaakt met 5 kaarten. De kaarten heb ik ingescand en de achtergrond verwijderd.



```
<button id="trekKaartButton">Klik om een kaart te trekken</button>
<div id="gekozenKaart"></div>
```

Ik heb een button met de naam 'trekKaartButton' gemaakt. Wanneer erop wordt geklikt, moet een random kaart worden weergegeven in de lege div 'gekozenKaart'.

```
const kaarten = document.getElementById('gekozenKaart');
fetch('tarotcard.json')
  .then(response => response.json())
  .then(data => trekEenKaart(data));

function trekEenKaart(data){
  console.log(data)
}
```

```
Uncaught TypeError: Cannot read properties of null (reading 'addEventListener')
at script.js:4:11
```

Ik heb de code toegevoegd aan het script waarin ook de andere Javascript-functies staan. Omdat de functie `accordion.addEventListener` eerder in het script wordt opgeroepen, maar niet kan worden uitgevoerd op deze pagina, krijg ik een foutmelding te zien en werkt deze code niet.

In deze code wordt een fetch-verzoek gedaan naar een JSON-bestand genaamd 'tarotcard.json'. Zodra de respons binnenkomt, wordt de ontvangen data omgezet naar JSON-formaat en vervolgens doorgegeven aan de functie `trekEenKaart()`. Deze functie wordt opgeroepen met de verkregen data als argument. In de `trekEenKaart()`-functie wordt de ontvangen data weergegeven in de console met behulp van `console.log(data)`.

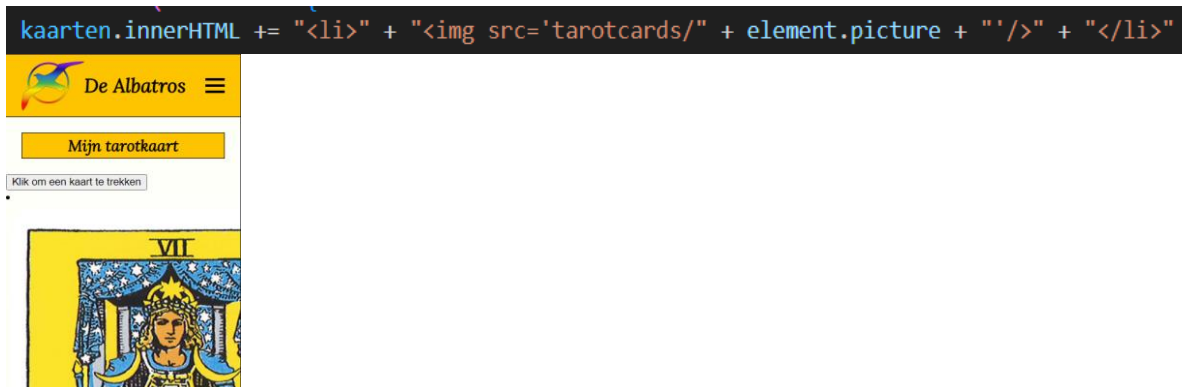
```
▼ Array(5)
  ▶ 0: {id: 1, picture: 'deZegewagen.png', title: 'De Zegewagen', kernwoorden: 'Wilskracht + Volharding + Controle + Succes', kaart: 'De Zegewagen is een kaart die staat voor wilskracht en actie, zoals weergegeven door de wagenmenner die sterk en trots op zijn wagen staat en zijn orders met vastberadenheid uitdeelt. Hij is gekroond met succes en versiert met symbolen van hemelse verbinding.', betekenis: 'Als je een boost voor je zelfvertrouwen nodig hebt, komt de Zegewagen je dat brengen. Je pad is juist en leidt naar succes. Je beschikt over alle vaardigheden en de vastberadenheid om je doel te behalen. Blijf je eigen koers volgen, wat er ook op je afkomt. Verhoog je focus en onderneem actie in de richting van wat je wilt - het is nu niet de tijd om achterover te leunen. De Zegewagen kan ook staan voor reizen, dus mogelijk ligt er een reisje in het verschiet.'}
  ▶ 1: {id: 2, picture: 'bokers2.png', title: 'Twee van Bokers', kernwoorden: 'De Zegewagen', kaart: 'De Zegewagen is een kaart die staat voor wilskracht en actie, zoals weergegeven door de wagenmenner die sterk en trots op zijn wagen staat en zijn orders met vastberadenheid uitdeelt. Hij is gekroond met succes en versiert met symbolen van hemelse verbinding.', betekenis: 'Als je een boost voor je zelfvertrouwen nodig hebt, komt de Zegewagen je dat brengen. Je pad is juist en leidt naar succes. Je beschikt over alle vaardigheden en de vastberadenheid om je doel te behalen. Blijf je eigen koers volgen, wat er ook op je afkomt. Verhoog je focus en onderneem actie in de richting van wat je wilt - het is nu niet de tijd om achterover te leunen. De Zegewagen kan ook staan voor reizen, dus mogelijk ligt er een reisje in het verschiet.'}
  ▶ 2: {id: 3, picture: 'koningVanPentakels.png', title: 'Koning van Pentakels', kernwoorden: 'De Zegewagen', kaart: 'De Zegewagen is een kaart die staat voor wilskracht en actie, zoals weergegeven door de wagenmenner die sterk en trots op zijn wagen staat en zijn orders met vastberadenheid uitdeelt. Hij is gekroond met succes en versiert met symbolen van hemelse verbinding.', betekenis: 'Als je een boost voor je zelfvertrouwen nodig hebt, komt de Zegewagen je dat brengen. Je pad is juist en leidt naar succes. Je beschikt over alle vaardigheden en de vastberadenheid om je doel te behalen. Blijf je eigen koers volgen, wat er ook op je afkomt. Verhoog je focus en onderneem actie in de richting van wat je wilt - het is nu niet de tijd om achterover te leunen. De Zegewagen kan ook staan voor reizen, dus mogelijk ligt er een reisje in het verschiet.'}
  ▶ 3: {id: 4, picture: 'stavenVier.png', title: 'Vier van Staven', kernwoorden: 'De Zegewagen', kaart: 'De Zegewagen is een kaart die staat voor wilskracht en actie, zoals weergegeven door de wagenmenner die sterk en trots op zijn wagen staat en zijn orders met vastberadenheid uitdeelt. Hij is gekroond met succes en versiert met symbolen van hemelse verbinding.', betekenis: 'Als je een boost voor je zelfvertrouwen nodig hebt, komt de Zegewagen je dat brengen. Je pad is juist en leidt naar succes. Je beschikt over alle vaardigheden en de vastberadenheid om je doel te behalen. Blijf je eigen koers volgen, wat er ook op je afkomt. Verhoog je focus en onderneem actie in de richting van wat je wilt - het is nu niet de tijd om achterover te leunen. De Zegewagen kan ook staan voor reizen, dus mogelijk ligt er een reisje in het verschiet.'}
  ▶ 4: {id: 5, picture: 'zwaardenDrie.png', title: 'Drie van Zwaarden', kernwoorden: 'De Zegewagen', kaart: 'De Zegewagen is een kaart die staat voor wilskracht en actie, zoals weergegeven door de wagenmenner die sterk en trots op zijn wagen staat en zijn orders met vastberadenheid uitdeelt. Hij is gekroond met succes en versiert met symbolen van hemelse verbinding.', betekenis: 'Als je een boost voor je zelfvertrouwen nodig hebt, komt de Zegewagen je dat brengen. Je pad is juist en leidt naar succes. Je beschikt over alle vaardigheden en de vastberadenheid om je doel te behalen. Blijf je eigen koers volgen, wat er ook op je afkomt. Verhoog je focus en onderneem actie in de richting van wat je wilt - het is nu niet de tijd om achterover te leunen. De Zegewagen kan ook staan voor reizen, dus mogelijk ligt er een reisje in het verschiet.'}
  length: 5
  [[Prototype]]: Array(0)
```

Ik heb de Javascript-code van de accordion verplaatst naar een apart script-bestand, waardoor de code het nu wel doet.



```
function trekEenKaart(data){
  data.forEach(element => {
    kaarten.innerHTML += "<li>" + element.picture + "</li>"
    kaarten.innerHTML += "<li>" + element.title + "</li>"
    kaarten.innerHTML += "<li>" + element.kernwoorden + "</li>"
    kaarten.innerHTML += "<li>" + element.kaart + "</li>"
    kaarten.innerHTML += "<li>" + element.betekenis + "</li>"
  });
}
```

Ik heb de functie trekEenKaart aangepast om de gegevens op te halen en toe te voegen aan de HTML. Voor elk element in de data-array worden de afbeelding, titel, kernwoorden, kaart en betekenis opgehaald en toegevoegd aan de HTML op de locatie van de div met het id 'gekozenKaart'. Hierdoor wordt alle tekst weergegeven op de pagina.



Om ervoor te zorgen dat de afbeeldingen zichtbaar zijn, heb ik het pad naar de map toegevoegd. Hierdoor worden de afbeeldingen weergegeven vanuit de map.

```
function trekEenKaart(data){
  data.forEach(element => {
    const cardInfo = document.createElement('div');
    cardInfo.className = 'card-info';

    cardInfo.innerHTML = `
      
      <h2>${element.title}</h2>
      <p>${element.kernwoorden}</p>
      <p>${element.kaart}</p>
      <p>${element.betekenis}</p>
    `;

    kaarten.appendChild(cardInfo);
  });
}
```



Om CSS toe te kunnen passen heb ik de code aangepast. Als gevolg hiervan is de afbeelding nu niet meer zichtbaar. Verder heb ik een div-element met de class 'card-info' toegevoegd. Hierdoor kan ik specifieke CSS-stijlen toepassen op dit element.

```
const afbeeldingTarotkaart = 'tarotcards/';
const afbeeldingURL = afbeeldingTarotkaart + element.picture;

cardInfo.innerHTML = `
  

```



Om dit probleem op te lossen, heb ik enkele aanpassingen gemaakt in de code. Allereerst heb ik een constante genaamd 'afbeeldingTarotkaart' gecreëerd, die het pad naar de map met de afbeeldingen bevat. Vervolgens heb ik een andere variabele genaamd 'afbeeldingURL' gemaakt door 'afbeeldingTarotkaart' te combineren met 'element.picture'. Hierdoor krijgt 'afbeeldingURL' de waarde van het samengevoegde pad, inclusief de naam van de afbeelding. Daarna heb ik in de 'innerHTML' een nieuw img-element aangemaakt en de 'src'-eigenschap ingesteld op de samengestelde 'afbeeldingURL'. Hierdoor wordt de afbeelding weergegeven op de pagina.

```
<p><strong>De kaart: </strong>${element.kaart}</p>
<p><strong>Betekenis: </strong>${element.betekenis}</p>
```

**De kaart:** De Zegewagen is een kaart die staat voor wilskracht en actie, zoals weergegeven door de wagenmenner die sterk en trots op zijn wagen staat en zijn orders met vastberadenheid uitdeelt. Hij is gekroond met succes en versiert met symbolen van hemelse verbinding.

**Betekenis:** Als je een boost voor je zelfvertrouwen nodig hebt, komt de

Om de paragraaf te laten beginnen met een dikgedrukte tekst, namelijk 'De kaart:' en 'Betekenis:', heb ik strong-tags gebruikt. Hierdoor wordt de tekst dikgedrukt weergegeven. Vervolgens worden de waarden van de elementen toegevoegd.

```
function trekEenKaart(data){
  const randomIndex = Math.floor(Math.random() * data.length);
  const selectedCard = data[randomIndex];

  const cardInfo = document.createElement('div');
  cardInfo.className = 'card-info';

  const afbeeldingTarotkaart = 'tarotcards/';
  const afbeeldingURL = afbeeldingTarotkaart + selectedCard.picture;

  cardInfo.innerHTML = `
    
    <h2>${selectedCard.title}</h2>
    <p>${selectedCard.kernwoorden}</p>
    <p><strong>De kaart: </strong>${selectedCard.kaart}</p>
    <p><strong>Betekenis: </strong>${selectedCard.betekenis}</p>
  `;

  kaarten.appendChild(cardInfo);
}
```

Om ervoor te zorgen dat elke keer als de pagina wordt geladen een willekeurige kaart wordt weergegeven, heb ik 'forEach' verwijderd. In plaats daarvan wordt er een willekeurig getal gegenereerd tussen 0 en 1 met behulp van Math.random(). Dit getal wordt vermenigvuldigd met de lengte van de 'data'-array. Door Math.floor() toe te passen, wordt dit getal afgerond naar beneden, wat resulteert in een willekeurige index genaamd randomIndex. Met behulp van deze randomIndex wordt het element in de 'data'-array dat overeenkomt met deze index, toegewezen aan de variabele 'selectedCard'. Door 'selectedCard' te gebruiken in plaats van het oorspronkelijke 'element' in de rest van de code, worden alleen de gegevens teruggegeven die bij de geselecteerde kaart horen.

```
const kaarten = document.getElementById('gekozenKaart');
const trekKaartButton = document.getElementById('trekKaartButton');

fetch('tarotcard.json')
  .then(response => response.json())
  .then(data => {
    trekKaartButton.addEventListener('click', () => trekEenKaart(data));
  });
```

Ik heb de code aangepast zodat deze wordt uitgevoerd wanneer er op de knop met de id 'trekKaartButton' wordt geklikt. Dit wordt gedaan door het toevoegen van een 'click'-eventListener aan de knop. Telkens wanneer de knop wordt geklikt, wordt er een nieuwe kaart weergegeven. Het enige probleem is dat elke keer dat er op de knop wordt geklikt, er een extra kaart wordt toegevoegd. Met andere woorden, uiteindelijk staan er meerdere kaarten onder elkaar.

```
kaarten.innerHTML = '';
```

Om te voorkomen dat de kaarten zich opstapelen, wordt ervoor gezorgd dat de container eerst leeg gemaakt is voordat de nieuwe gegevens opgehaald worden. Op deze manier is er altijd maar één kaart zichtbaar.

```
<div class="teksten">
  <button id="trekKaartButton"></button>
  <div id="gekozenKaart"></div>
</div>
```



```
button#trekKaartButton {
  border: none;
  background: none;
  padding: 0;
  cursor: pointer;
}
```

Om de afbeelding als een knop te kunnen gebruiken, heb ik de afbeelding binnen de button-tag geplaatst. Door middel van CSS heb ik ervoor gezorgd dat er geen rand rond de afbeelding wordt weergegeven. De CSS-style 'cursor: pointer' wordt toegepast om de cursor aan te passen, zodat het duidelijk is dat de afbeelding als knop fungeert en erop geklikt kan worden.

Kaart openen in andere pagina

```
<button id="trekKaartButton"></button>
```

In tarot.html heb ik de button toegevoegd.

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

In drawnCard.html heb ik de div met id 'gekozenKaart' toegevoegd.

```
const trekKaartButton = document.getElementById('trekKaartButton');

fetch('tarotcard.json')
  .then(response => response.json())
  .then(data => {
    trekKaartButton.addEventListener('click', () => trekEenKaart(data));
  });

function trekEenKaart(data) {
  const randomIndex = Math.floor(Math.random() * data.length);
  const selectedCard = data[randomIndex];

  sessionStorage.setItem('selectedCard', JSON.stringify(selectedCard));
  window.location.href = 'drawnCard.html';
}
```

Door middel van 'sessionStorage.setItem('selectedCard', JSON.stringify(selectedCard))' wordt de waarde van selectedCard opgeslagen in de session storage. Hierbij wordt de selectedCard omgezet naar een JSON-string met behulp van JSON.stringify() voordat deze wordt opgeslagen.

Door middel van 'window.location.href = 'drawnCard.html'' zorg ik ervoor dat de drawnCard pagina wordt geopend.

```

window.addEventListener('DOMContentLoaded', () => {
  const kaarten = document.getElementById('gekozenKaart');
  const selectedCard = JSON.parse(sessionStorage.getItem('selectedCard'));

  const cardInfo = document.createElement('div');
  cardInfo.className = 'card-info';

  const afbeeldingTarotkaart = 'tarotcards/';
  const afbeeldingURL = afbeeldingTarotkaart + selectedCard.picture;

  cardInfo.innerHTML = `
    
    <h2>${selectedCard.title}</h2>
    <p>${selectedCard.kernwoorden}</p>
    <p><strong>De kaart: </strong>${selectedCard.kaart}</p>
    <p><strong>Betekenis: </strong>${selectedCard.betekenis}</p>
  `;

  kaarten.appendChild(cardInfo);
});

```

Door middel van 'window.addEventListener('DOMContentLoaded', () => {' wordt de bijbehorende functie uitgevoerd als het HTML-document volledig is geladen.

Ik heb 'const kaarten = document.getElementById('gekozenKaart')' in de functie geplaatst in plaats van erbuiten. Hierdoor wordt deze namelijk pas ingesteld op het moment dat de HTML-pagina is geladen.

Verder haal ik de eerder opgeslagen 'selectedCard' op uit de session storage en sla ik deze op in de variabele 'selectedCard' met behulp van const selectedCard = JSON.parse(sessionStorage.getItem('selectedCard')).

## Bronnen

- Couch, J. (z.d.). *Menu “Hamburger” Icon Animations*. codepen.io. Geraadpleegd op 9 juni 2023, van <https://codepen.io/designcouch/pen/ExvwPY>
- Efteling. (z.d.). *Wereld vol Wonderen*. Geraadpleegd op 9 juni 2023, van [https://www.efteling.com/nl?gclid=Cj0KCQjwj\\_ajBhCqARIsAA37s0wGe4bYytly\\_6yo\\_8bh\\_xSmTiSgNto9rRIGwKwIIWNIMjOj2HUzBgaAv2EEALw\\_wcB](https://www.efteling.com/nl?gclid=Cj0KCQjwj_ajBhCqARIsAA37s0wGe4bYytly_6yo_8bh_xSmTiSgNto9rRIGwKwIIWNIMjOj2HUzBgaAv2EEALw_wcB)
- Google. (z.d.). *Fontys Hogeschool Economie & Communicatie eindhoven*. Geraadpleegd op 9 juni 2023, van <https://www.google.nl/maps/place/Fontys+Hogeschool+Economie+%26+Communicatie+eindhoven/@51.4517645,5.476525,17z/data=!4m10!1m2!2m1!1sfontys+rachelsmolen!3m6!1s0x47c6d92199730073:0x1c98f3a0d8ee087b!8m2!3d51.4512481!4d5.4796619!15sChNmb250eXMgcmlFjaGVsc21vbGVulGwOIAQGSaQdjb2xsZWdl4AEA!16s%2Fg%2F1z264qdk?hl=nl&entry=tu>
- Kevin Powell. (2023, 2 mei). *The simple trick to transition from height 0 to auto with CSS* [Video]. YouTube. Geraadpleegd op 12 juni 2023, van [https://www.youtube.com/watch?v=B\\_n4YONte5A](https://www.youtube.com/watch?v=B_n4YONte5A)
- Lanser, A. (2019, 2 april). *Je eigen Google Maps API key aanvragen*. Tijdvooreensite. Geraadpleegd op 9 juni 2023, van <https://www.tijdvooreensite.nl/blog/een-eigen-google-maps-api-key-aanvragen/>
- smashtheshell. (z.d.). *Untitled*. codepen.io. Geraadpleegd op 8 juni 2023, van <https://codepen.io/smashtheshell/pen/zYBWgPW>
- smashtheshell. (2020, 29 oktober). *Responsive Multi Level Dropdown Menu with CSS | CSS3 Animated Hamburger Menu* [Video]. YouTube. Geraadpleegd op 8 juni 2023, van <https://www.youtube.com/watch?v=pro3ceh2YZ8>