Technisch Ontwerp

iConcepts

“EenmaalAndermaal”

|  |  |
| --- | --- |
| *Auteurs* | Groep 14  Mehmet Batal: ID 600720,  Danny Hageman 608227,  Hendrik Otten 619129,  Bram Soutendam 599164,  Elviana Cornelissen 598830 |
| *Cursus* | *I-Project* |
| *Opdrachtgever* | *iConcepts* |
| *Begeleiders* | *Rein Harlé – Product Owner*  *Ben van Hoof – Scrum coach*  *Rody Aldenhoven – Expert Professional Skills* |
| *Onderwijsinstelling* | *Hogeschool van Arnhem en Nijmegen* |
| *Plaats & datum* | *Arnhem, 12 juni 2019* |
| *Versie nummer*  *Status* | *1.0*  *In Bewerking* |



Inhoudsopgave

[1. Inleiding 3](#_Toc9584632)

[2. Front-end-framework 4](#_Toc9584633)

[3. Responsiveness 4](#_Toc9584634)

[4. Netwerkarchitectuur 5](#_Toc9584635)

[5. Database 6](#_Toc9584636)

[5.1 Datatypen & defaults 6](#_Toc9584637)

[5.2 Check-Contraints 6](#_Toc9584638)

[5.3 Conversie 6](#_Toc9584639)

# 1. Inleiding

In dit document beschrijft het team de technische specificaties van de website ‘’EenmaalAndermaal” voor iConcepts. Er wordt beschreven welke functies en variabelen er zijn gebruikt voor het realiseren van de website. Ook worden de gebruikte servers getoond en hoe ze in verbinding staan met de website. In dit document wordt dus vooral verteld ‘hoe’ verschillende componenten werken op/met de site.

# 2. Front-end-framework

Wij hebben gekozen voor ‘Foundation’ als front-end-framework. Hoe zijn wij tot deze keuze gekomen? Er is een onderzoeksrapport gemaakt over verschillende front-end-frameworks. Eerst is er een longlist gemaakt van vijf verschillende front-end-frameworks. Op die 5 frameworks hebben wij criteria toegepast, waardoor er drie overbleven. Deze drie zijn allemaal getest, en de beste hebben wij gekozen. Foundation werkte voor ons het makkelijkst en het best, ook waren er veel voorbeelden te vinden op hun website. Doordat er zo veel voorbeelden te vinden waren is onze keuze uiteindelijk gevallen voor Foundation.



Figuur : Foundation

# 3. Responsiveness

Hoe is onze site responsive gemaakt? Wij hebben gebruik gemaakt van het framework ‘Foundation’. Foundation stelde een grid beschikbaar. Deze grid is opgedeeld in 4 vakken: een header, aside, main en een footer. Alles wat wij gemaakt hebben zit verwerkt in de grid. Hierdoor is alles netjes responsive. Op de mobiel hebben wij zelf een aantal aanpassing gemaakt, zoals het verplaatsen van de zoekbalk en een kleiner logo gebruiken zodat alles netjes blijft.

# 4. Netwerkarchitectuur

# 5. Database

## 5.1 Datatypen & defaults

## 5.2 Check-Contraints

In onze database zitten een aantal check-constraints. Deze check constraints zitten in onze database om vuile data te omzeilen, zodat de database niet vol raakt met onzinnige data. Hieronder staan al onze check-constraints per tabel beschreven.

**Tabel ‘gebruiker’**

* Op de kolom ‘geboortedatum’ zit een check constraint die een datum dan later dan vandaag tegenhoud.
* Op de kolom ‘wachtwoord’ zitten twee check constraints, de eerste is dat een wachtwoord minimaal 7 karakters moet bevatten. De tweede is dat een wachtwoord minimaal één letter en één cijfer moet bevatten.

**Tabel ‘voorwerp’**

* Op de kolom ‘betalingswijze’ zit een check constraint die kijkt of de betalingswijze in een van de vier opgegeven opties zit.
* Op de kolom ‘startprijs’ zit een check constraint die kijkt of de opgegeven startprijs groter is dan nul.
* Op de kolom ‘veilinggesloten’ zit een check constraint die kijkt of de data ‘wel’ of ‘niet’ is. Hiermee wordt gekeken of de veiling is gesloten.
* Met de check-constraint ‘ck\_voorwerp\_verkoopprijs\_negatief’ wordt gekeken of de verkooprijs gelijk of groter is dan de startprijs.
* Met de check-constraint ‘ck\_voorwerp\_verkoper\_geen\_koper’ wordt gekeken of de koper van een product niet ook de verkoper was van dat product.
* Met de check-constraint ‘ck\_looptijd’ wordt gekeken of de data ‘1-3-5-7-10’ gebruikt wordt, dit is de looptijd in dagen. Er kan dus bijvoorbeeld geen 12 dagen gekozen worden want dit wordt tegen gehouden door de check-constraints.

**Tabel ‘bod’**

* Met de check constraint ‘ck\_NietBiedenEigenProduct’ wordt gekeken of de koper niet op zijn eigen product biedt.
* Met de check constraint ‘CHK\_VeilingOpen’ wordt gekeken of een veiling nog open staat of al gesloten is.

## 5.3 Conversie

# 6 Javascript, AJAX en jQuery

## 6.1 Waarom javascript?

In het product is gebruik gemaakt van javascript. Hierin wordt behandelt de meest belangrijke onderdelen die gebruik hebben gemaakt van javascript. De timer, het navigatie menu, rubrieken kiezen op laagste niveau en het toevoegen van meerdere bestanden.

Waarom is er gebruik gemaakt van javascript? De grootste reden waarom hiervoor gekozen is, is dat het mogelijk is om met javascript een pagina te updaten zonder dat de gebruiker de pagina zelf herlaadt. Met PHP is dat wat lastiger bijvoorbeeld om een timer te gebruiken die afloopt zonder dat de gebruiker zelf de pagina herlaadt. Omdat je niet live ziet wanneer het precies afloopt op seconden. Kan het zijn dat de gebruiker te laat is om te bieden zonder dat hij het zelf merkt. Dus hierdoor voorkom je dat hij zelf de pagina moet updaten.

Er zijn verschillende manieren gebruikt in javascript om een deel werkend te krijgen. Hieronder zal er uitgelegd worden welke delen in het product javascript gebruikt heeft. Ook wordt er vermeldt hoe het gedaan is.

## 6.2 Timer

Op de productpagina staat een timer die afloopt en geeft aan of een veiling sluit of niet. De timer voert een functie uit, vanaf de datum wanneer het product is aangemaakt tot de einddatum die berekend is in de database met behulp van de looptijd. Dit wordt gedaan met PHP maar het gedeelte dat het live wordt weergegeven wordt gedaan met javascript. De timer functie laat zien dat het goed aftelt en hij berekent hoeveel tijd er nog over is totdat de veiling sluit.

## 6.3 Rubrieken

Bij het navigatie menu en het kiezen van de laagste rubriek is er beide gebruik gemaakt van AJAX. AJAX is een manier om informatie op te vragen via de server en ook informatie te verkrijgen van de server nadat de pagina geladen is. De informatie wordt automatisch geüpdatet zodat de relevante informatie wordt laten zien.

Links op elke pagina staan alle rubrieken weergegeven dit wordt het navigatie menu genoemd. Het navigatie menu werkt als volgt, er wordt op een rubriek geklikt en deze klapt uit waar de sub-rubrieken van die hoofdrubriek wordt weergegeven. En zo verder totdat het laagste niveau is bereikt. De reden dat hiervoor javascript gebruikt is komt doordat er performance issues ontstond. De pagina laadt te lang doordat het door een heel script moet waar het controleert of elk rubriek en sub rubriek heeft en zo verder. Met 3000+ rubrieken is dat wel te begrijpen waarom het zo lang duurt. Daardoor is er gebruik gemaakt van jQuery en Ajax. jQuery is een javascript library en maakt het gebruiken van javascript makkelijker.

Met elke klik op een rubriek zal er een verzoek verstuurd worden die controleert of elk rubriek een sub-rubriek heeft. Hieronder zie je een deel van de code hoe er een verzoek verstuurd wordt.

var ajax = new XMLHttpRequest();

ajax.open("GET", "get-rubrieken.php?rubrieknummer=" + rubrieknummer, true);

ajax.send();

*Figuur @@: Verzenden verzoek deel code*

Wanneer er antwoordt wordt gegeven zal het een HTML-code aanmaken die de sub-rubrieken laat zien. De nieuwe HTML-code wordt dan geplaatst in het element die de *id* heeft van “sub-rubriek”.

ajax.onreadystatechange = function() {

if(this.readyState == 4 && this.status == 200){

var data = JSON.parse(this.responseText);

if(data.length == 0){

var html = "";

}else{

var html = "<option disabled selected> Kies een subrubriek </option>";

}

for(var a = 0; a < data.length; a++){

html += "<option value='" + data[a].rubrieknummer + "'>"+ data[a].rubrieknaam +"</option>";

}

document.getElementById("sub-rubriek").innerHTML = html;

}

};

*Figuur @@: Het uitlezen van gevraagde data deel code*

Als er nog een keer op dezelfde rubriek gedrukt wordt zal het gedeelte met sub-rubrieken inklappen en de HTML-code verwijderen zodat het voorkomt dat de pagina eigenlijk te veel html heeft en te lang moet laden.

Op het toevoegen van een voorwerp pagina kan je kiezen in een select box welke rubriek je voorwerp moet worden weergegeven. Dit kan alleen op het laagste rubriek. Als er een optie gekozen is zal er automatisch zoeken naar sub rubriek en die weergeven in de volgende select box. Totdat het op het laagste niveau is. Dit wordt verstuurd met het form naar de database. Dit werkt hetzelfde met het navigatie menu als er een rubriek gekozen is zal hij vragen naar de sub rubrieken.   
  
  
6.4 Bestanden toevoegen

Op de voorwerp toevoegen pagina is het ook mogelijk om afbeeldingen te plaatsen. Dit wordt met PHP afgehandeld. Dus voor het plaatsen van meerdere bestanden worden de bestand toevoegen knoppen toegevoegd. Dit maakt het overzichtelijker en makkelijker om afbeeldingen te kiezen.   
Voor het toevoegen van knoppen is er gebruik gemaakt van jQuery. Als er op de knop gedrukt wordt om meerdere bestanden toe te voegen dan zal de bestand toevoegen knop zichtbaar worden. Hoe dit werkt gaat als volgt. Als op een knop wordt gedrukt zal deze een functie uitvoeren. De knop heeft een *id* die “add” heet. Met de javascript code die daarbij hoort staat hieronder weergegeven.

var bestand = 0;

var max = 3;

$("#add").click(function(){

bestand++;

if(bestand <= max){

$("tbody").append('<tr class="add\_row"><td>#</td><td><input class="file" name="upfile[]" type="file" required></td><td class="text-center"><button type="button" class="btn button btn-sm" id="delete" title="Verwijder bestand">Verwijder bestand</button></td><tr>');

}

if(bestand == max){

$("#add").css("visibility", "hidden");

}

});

*Figuur @@: Bestanden toevoegen deel code*

Dus met de “add” *id* als daarop geklikt is zal hij de functie uitvoeren. Binnen de tbody tag zal append() functie ervoor zorgen dat er extra HTML-code wordt toegevoegd. Er is ook een check voor de knop, als er meer dan 3 bestanden zijn toegevoegd dan zal de knop verdwijnen.