# Casebeschrijving Tweakers.net – Sander Koch S27

Zoals de titel van deze document al aangaf, heb ik tijdens DBS2 er voor gekozen om mijn database te baseren op de website [www.Tweakers.net](http://www.Tweakers.net).

Ik heb van tweakers.net een gedeelte van de website gebruikt voor mijn DBS2 opdracht. Hierbij heb ik er voor gekozen om de volgende delen van de website wel te implementeren:

* Nieuwsoverzicht & nieuwsberichten(artikelen)
* Reacties op artikelen/reviews en andere reacties.
* Pricewatch/producten
* Reviews van producten
* Profiel van gebruikers

Hierbij vallen dus de aspecten zoals het forum en blogs buiten de applicatie zelf. De staf van tweakers.net zijn als enigste bevoegd op artikels & producten te plaatsen en aan te passen. Andere gebruikers kunnen reageren op artikelen en reviews, ze kunnen natuurlijk zelf ook reviews plaatsen. Daarnaast is er ook nog een Abonnee gebruiker, dit is een speciale gebruiker die geen reclame krijgt te zien op de website. De staf van tweakers.net heeft natuurlijk ook de zelfde rechten als normale gebruikers. De Pricewatch is ook een veel gebruikt onderdeel van tweakers.net. Hierbij kan je opzoek gaan naar bepaalde producten, prijzen vergelijken van verschillende winkels. De eigenschappen van het product inzien, en eventuele reviews van het product zijn ook zichtbaar om naar te navigeren. Het zoeken naar producten kan volgens zoektermen, maar ook het gebruiken van verschillende filters, of navigeren door de categorieën heen. Daarnaast mogen gebruikers ook de profielen van andere gebruikers inzien.

Aan de hand van de bovenstaande DBS2 implementatie en interpretatie komen we tot de volgende klassen:

* DatabaseManager
* LoginSystem
* ProductManager
* UserManager
* ReviewManager
* ArticleManager

Models:

* Product
* User
* Review
* Article
* PricewatchCategory(enum)
* ArticleCategory(enum

Waarbij voor producten, reviews, en users het van belang is dat ze ook nog bestanden kunnen uploaden(bijv. foto’s). Alle database requests zal door de DatabaseManager lopen, daarnaast heeft elke aspect van het systeem zijn eigen manager/logic klasse. Dit om er voor te zorgen dat niet alle logica in één onoverzichtelijke klasse komt te staan.

## Planning

|  |  |  |
| --- | --- | --- |
| Onderdelen | Einddatum | Afwijking |
| Analyse & onderzoek document | 13/05/2016 | 24-05-2016 uit eindelijke behaalde datum, dit kwam doordat er telkens andere dingen tussen kwamen. Waardoor ik het niet op tijd heb afgemaakt. |
| Productie omgeving ESX  (onderdeel van onderzoek document) | 13/05/2016 | Het zelfde als hierboven. |
| Aanmaken git repository en start implementatie(aanmaken van klassen aan de hand van klassendiagram) | 17/05/2016 | 06-06-2016 uit eindelijk behaalde datum, ik was al de week daarvoor begonnen met coderen maar was vergeten gelijk een repo aan te maken. |
| DatabaseManager & inlogsysteem + bijhorende webpages | 23/05/2016 | 31-05-2016 uit eindelijk behaalde datum.  Vanwege vertraging in mijn analyse/onderzoek fase. |
| Pricewatch/product & profielen | 27/05/2016 | 08-06-2016 uit eindelijk behaalde datum |
| Artikel en review systemen | 03/06/2016 | 10-06-2016  Uit eindelijke behaalde datum vanwege eerdere vertraging. |
| Debuggen van applicatie | 08/06/2016 | 12-06-2016  Uit eindelijke behaalde datum vanwege eerdere vertraging. |
| Oplevering applicatie | 10/06/2016 | 12-06-2016  Uit eindelijke behaalde datum vanwege eerdere vertraging. |

Uit eindelijk is mijn planning met ongeveer een week verschoven. Mijn ontwerpdocument is op 10-06-2016 nog aangepast.