Technische Documentatie

Vraag- & Antwoorforum, werktitel `WebDBOverflow`.

Requirements

De webapplicatie is ontwikkeld om draaien op een zogenaamde *LAMP*-server. Dat wil zeggen: de code gaat ervan uit dat hij wordt uitgevoerd in een omgeving waarin het volgende beschikbaar is:

- Linux, of een aanverwant besturingssysteem.
  De code is ook getest in een Windows-omgeving. Dit geeft geen problemen met directory-delimiters (\ vs /), of deze zijn door het veranderen van 1 constante in de code oplosbaar, *zolang* de webserver met urls niet anders afhandeld dan Apache doet.
- Apache, de webserver-software.
  De code is getest en werkt correct op een server die Apache versie 2.2.15 draait, en ook op Apache 2.2.21. Voor het correct functioneren van de software, en met name het parsen van URL's, is het essentieel dat de .htaccess file in de root van de code door Apache aangesproken en verwerkt wordt.
- MySQL, de database-software.
  De software maakt intensief gebruik van de database, om alle inhoud in op te slaan en uit te lezen. Ten behoeve van de zoekfunctie wordt MySQL's Full Text Search gebruikt. Hiervoor is het noodzakelijk dat de op zijn minst de database-tabellen Threads en Replies van MySQL's tabel-engine MyISAM gebruik maken.
  Alle overige tabellen, en de ook alle andere queries die de software op de database uitvoert, zouden eventueel eenvoudig naar andere database-software te herschrijven zijn. (Zie de klasse Helpers_Db).
  De software is getest met MySQL versie 5.1.16 en versie 5.5.16
- PHP, de scripting-taal.
  De software is volledig in PHP geschreven, en in grote lijnen gemaakt volgens het model van Object geOrienteerd Programmeren (OOP). De software gebruikt regelmatig functionaliteit uit PHP 5.3 en hoger. Hij is getest op PHP versie 5.3.3.

Database-verbinding

In deze opzet maakt PHP veelvuldig verbinding met de database. Hiertoe is een aantal parameters (naam, wachtwoord) noodzakelijk. Deze gegevens worden opgeslagen in een XML-bestand net boven de code-root. In dit bestand (mysqlconfig.xml) staan nu de gegevens uit de test-omgeving, behorende bij een lokaal draaiende MySQL-database, waarvan een dump (mysqldump.sql) beschikbaar is in dezelfde map als dit document.

Cross-browser compatibiliteit

De software is getest in de meeste recente Firefox- en InternetExplorer-browsers, en functioneert daarin naar behoren.

JavaScript

De software maakt op slechts 1 plek actief gebruik van JavaScript: bij de quick-search. Indien een gebruiker van de site JavaScript heeft uitgeschakeld, werkt deze functionaliteit niet. Dat verstoort verder de overige werking van de site geenszins, en de volledige zoekresultaten worden gewoon weergegeven.

MVC design pattern & design keuzes

Er is bij dit project gekozen voor een *MVC design pattern*. Dit zorgt voor een duidelijke splitsing van de taken binnen de applicatie. Vanwege deze duidelijke taakverdeling staat dit systeem op een sterk modulaire basis. Door deze modulaire basis is het zeer eenvoudig de applicatie uit te breiden met verscheidene extra functionaliteiten. Zo is de applicatie toekomstbestendig en hiermee zeer breed in te zetten.

MVC

MVC staat voor Model View Controller, en het zijn deze drie componenten die de basis vormen van de applicatie. Elk object binnen de applicatie (User, Thread, Reply etc.) is een Model. Elk Model heeft als parent een Base Model. Deze Base zorgt voor de algemene functionaliteiten zoals onder andere *save*, *fetchById* en *count*. Ieder Model dat deze Base extend verkrijgt al deze functionaliteiten wanneer zij een enkele methode implementeren (*declareFields*).

Models zijn het uitgangspunt voor de connectie met de Database. Elke query op de database loopt via de Base Model. Hierdoor is de database connectie gecentraliseerd binnen het systeem en leent deze zich ook voor gemakkelijke koppeling met andere databases. Wegens de genomen ontwerpkeuzes is er op dit moment geen sprake van een PDO implementatie, hoewel deze met enig herschrijven gemakkelijk toe te passen valt op de applicatie.

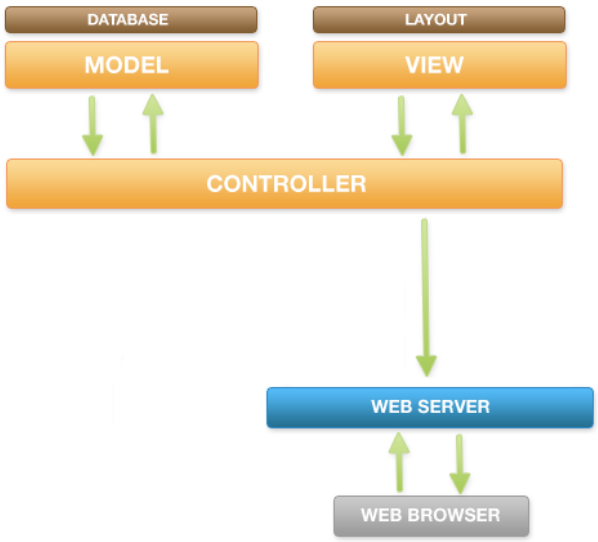


Diagram van de verscheidene componenten binnen de applicatie met een duidelijke verdeling tussen Models, Controllers en Views

De Models zijn het toegangspunt voor de Database, maar de Models zelf worden weer enkel aangeroepen door de Controllers. Waar de taak van de Models het beheren van de datastructuur van de objecten is, is de taak van de Controllers die van het manipuleren van de data in deze datastructuren. Elke manipulatie van data in de Models loopt via een Controller. Deze manipulatie kan het gevolg zijn van een systeemhandeling (cron) of een

gebruikershandeling. Een controller heeft verder de taak om de juiste Views te selecteren nadat een handeling is verricht (soms vindt er ook geen handeling plaats).

Views zijn simpele classes voornamelijk bestaande uit html. Elke View heeft de methode *render* waarbij HTML wordt geoutput naar de buffer. Deze buffer wordt vervolgens ingelezen en geïnjecteerd in een template bestand. De Controller stuurt de View aan en geeft deze de benodigde informatie mee. Wanneer de View de benodigde data heeft bouwt hij deze op in een simpele html structuur.

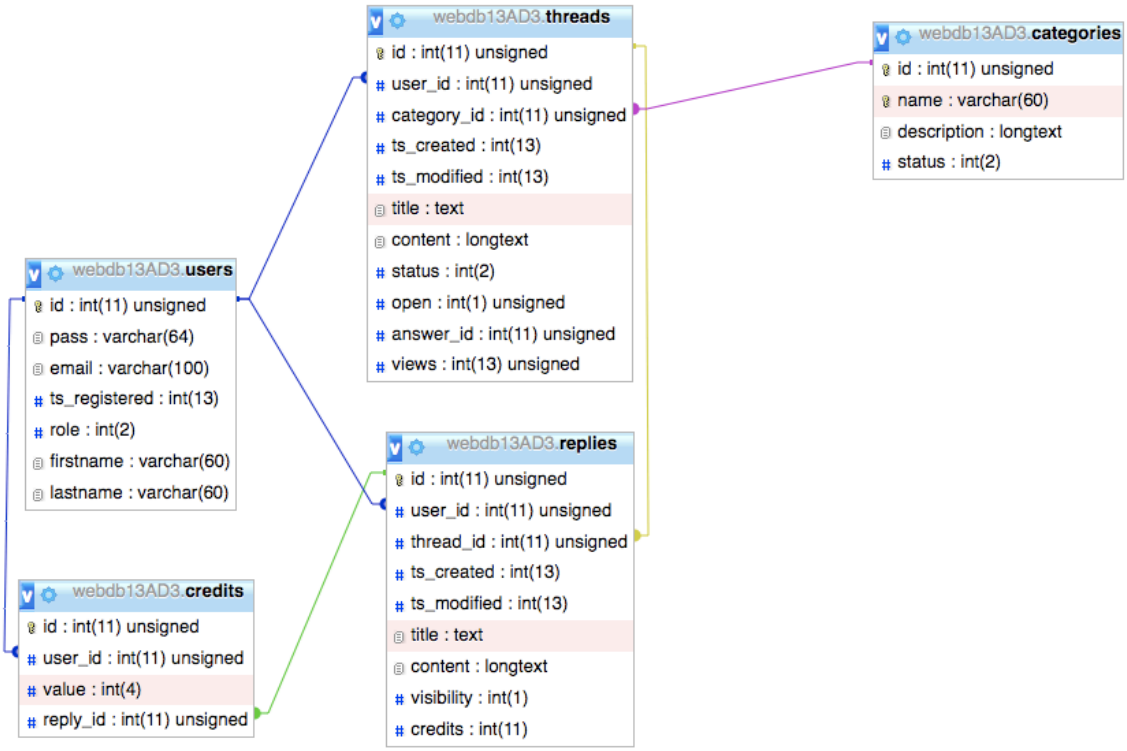
Voordelen

Door de MVC design pattern te gebruiken heeft het systeem een aantal grote voordelen. Door de modulaire aard van de applicatie valt deze, zoals eerder gemeld, zeer eenvoudig uit te breiden. Er kunnen zonder problemen extra classes worden toegevoegd en hun functionaliteiten correct ingedeeld in respectievelijke controllers. Deze duidelijk codestructuur zorgt ervoor dat het onderhoud zeer eenvoudig kan worden gehouden.

Naast het vermogen om gemakkelijke nieuwe extensies te implementeren en deze overzichtelijk te kunnen onderhouden is door de splitsing van Views het ook nog eens mogelijk om het gehele ontwerp van de site aan te passen zonder dat daarvoor de basis van de code van het systeemaangepast hoeft te worden. Deze *templating* draagt er toe bij dat de applicatie in verschillende grafische designs kan worden ontplooid en hiermee verhoogt de applicatie zijn herbruikbaarheid.

Database

Hieronder een grafisch overzicht van het ontwerp van de database. De tabellen corresponderen in hoge mate met de Models uit de MVC-structure.



Bij deze documentatie worden twee MySQL-dumps geleverd.

- Een volledige, met structuur én gegevens van de test-omgeving: documentation/mysqldump-webdb13AD3.sql.gz
  Hierin staan enkele test-users, -categories, -threads en -posts, zodat de werking van de site getest kan worden.
- Een met uitsluitende de structuur: documentation/mysqldump-webdb13AD3-structuur.sql.gz
  Hierin is slechts een record: een admin-user met mail@webdb.science.uva.nl / beoordeling als login-gegevens. Zodoende kunnen nieuwe categorieen worden aangemaakt na bijvoorbeeld een schone installatie.
  Deze login is wel afhankelijk de huidige salt die zich in mysqlconfig.xml (in de root) bevindt.

Bugs en issues

Op het moment van opleveren zitten er nog de volgende onvolkomenheden in de site:

- Gebruikers blokkeren heeft weinig impact
- Gebruikers zijn nog niet te bewerken (naamswijziging bijvoorbeeld)
- Het editen van threads inog niet optimaal
- Credits voor replies zijn in opzet (Model) aanwezig, maar nog niet in controllers en views uitgewerkt.
- Vragen en reacties met aanhalingstekens
  Er zit een bug in het updaten van threads (bij elke view tbv view-counter), zodanig dat het escapen verkeerd gaat.
- PDO-mysql niet geïmplementeerd, aangezien dit de opzet van de Models en Controllers in de basis zou doen veranderen.
- Enige vertraging in de logout.
- Threads-listings-view sorterings-boxje onjuist weergegeven in Safari (display: collapse workaround nodig).

Zie ook de lijst met todo's in de PHP-doc.