

Web Advanced

Verslag

Daniela Lupo

Nida Ilyas

Marnix Meuwissen

Tim Van Den Bosh

Britt Verschakelen

Concept

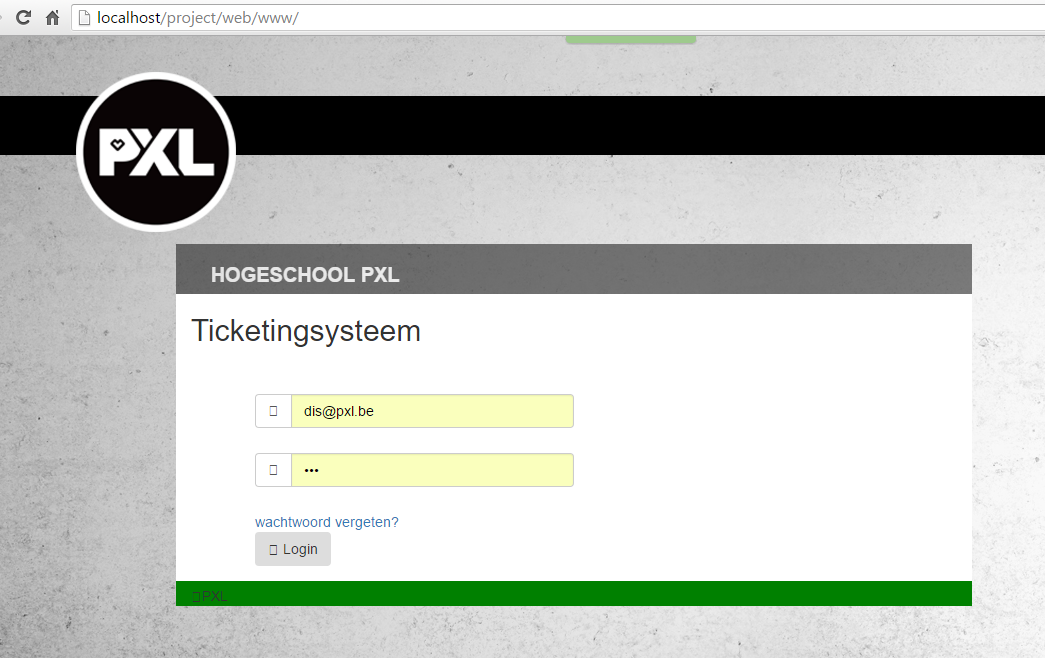
Het concept van dit systeem is om alle problemen, die op te lossen zijn door een werkman, op te slaan. Gebruikers kunnen problemen melden en deze opvolgen. De dispatcher beheert alle problemen en kent deze toe aan een werkman. De admin maakt alle gebruikers aan.

Op deze manier worden er geen problemen vergeten en vlugger opgelost.

# Handleiding website

Je surft naar: [localhost/project/web/www/](http://localhost/project/web/www/)

Je komt op de login pagina uit van het ticketingsysteem.

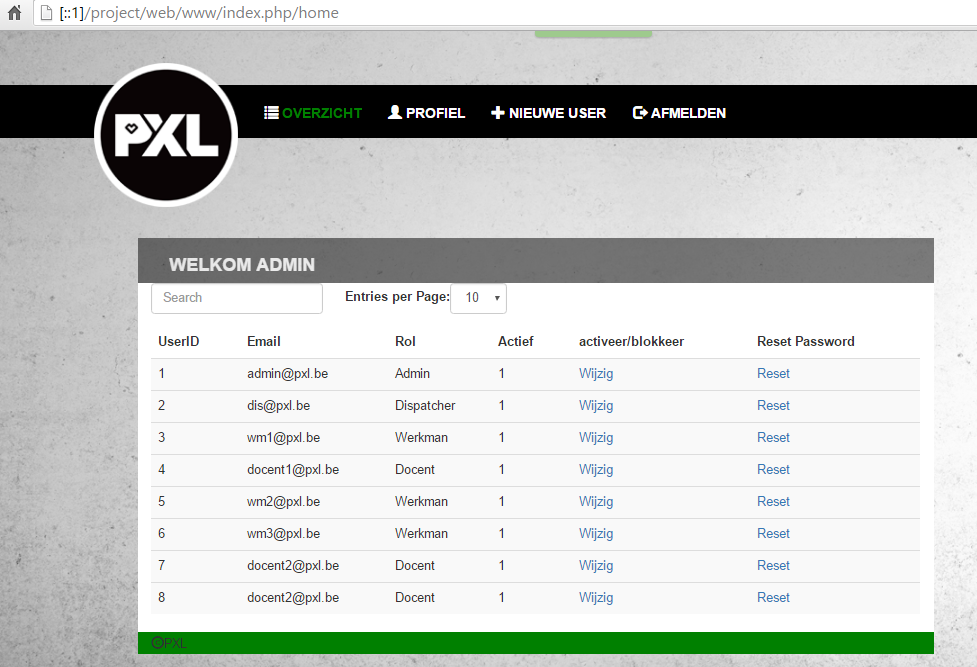


Inloggegevens:

|  |  |
| --- | --- |
| Inlog | Wachtwoord |
| admin@pxl.be | admin |
| docent1@pxl.be | pxl |
| docent2@pxl.be | pxl |
| wm1@pxl.be | pxl |
| wm2@pxl.be | pxl |
| wm3@pxl.be | pxl |
| dis@pxl.be | pxl |

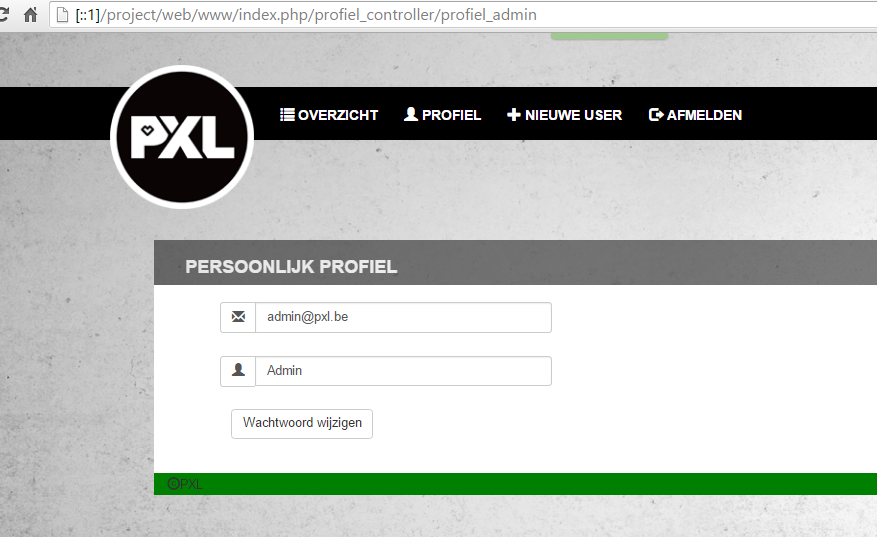
## Admin

Overzicht profielen (welkom pagina admin):

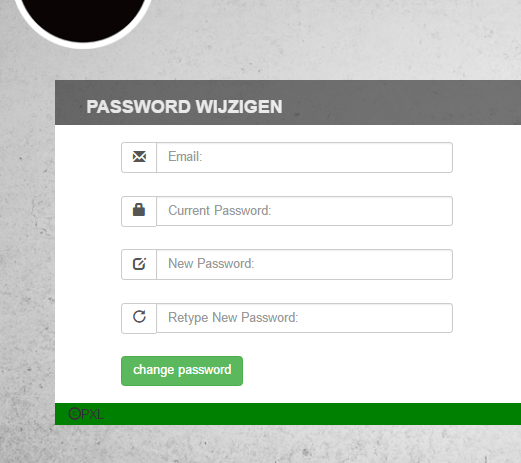


Hier zie je een overzicht van alle profielen. 1 is actief en 0 is non-actief. De admin kan ook profielen bewerken:

Navigatie profiel:

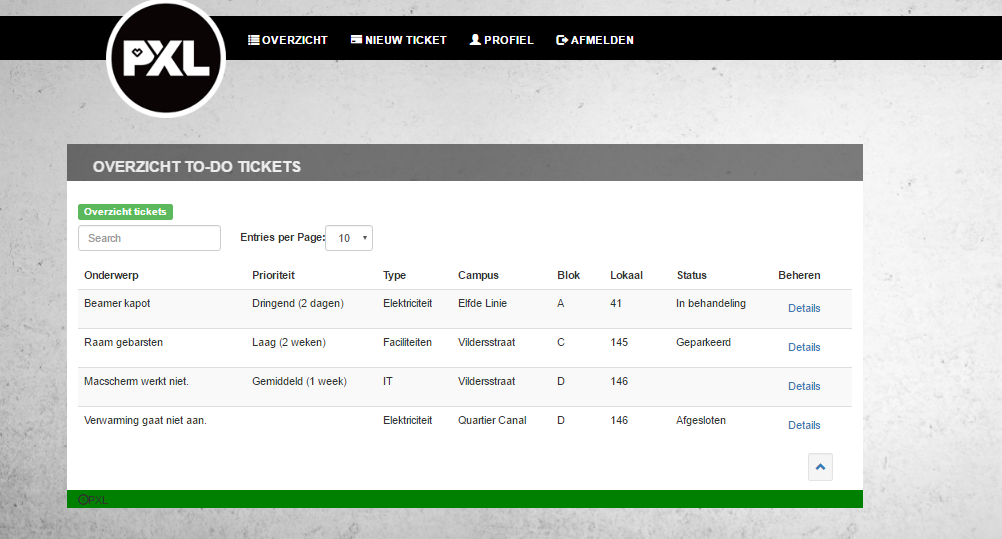


De admin kan zijn eigen wachtwoord hier aanpassen.



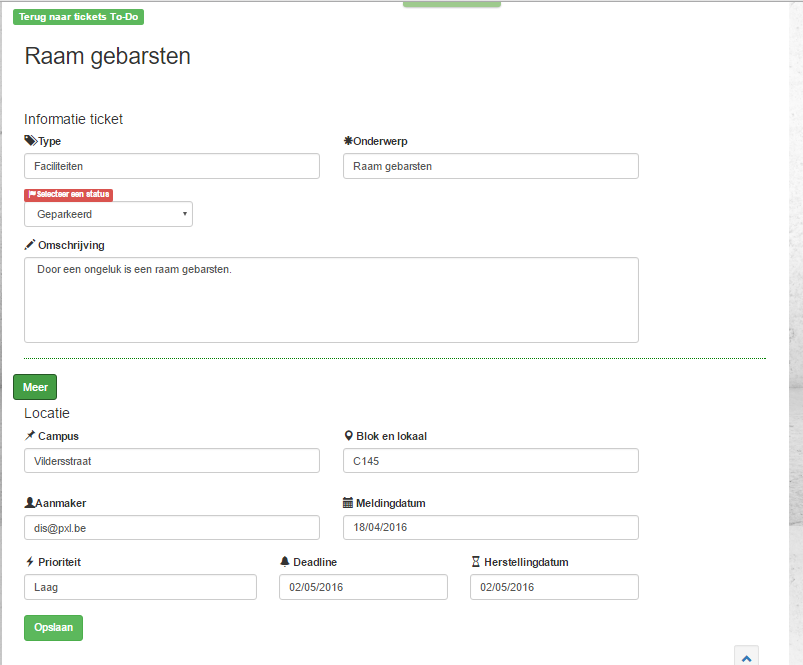
## Werkman

Overzicht to-do tickets (welkom pagina werkman):

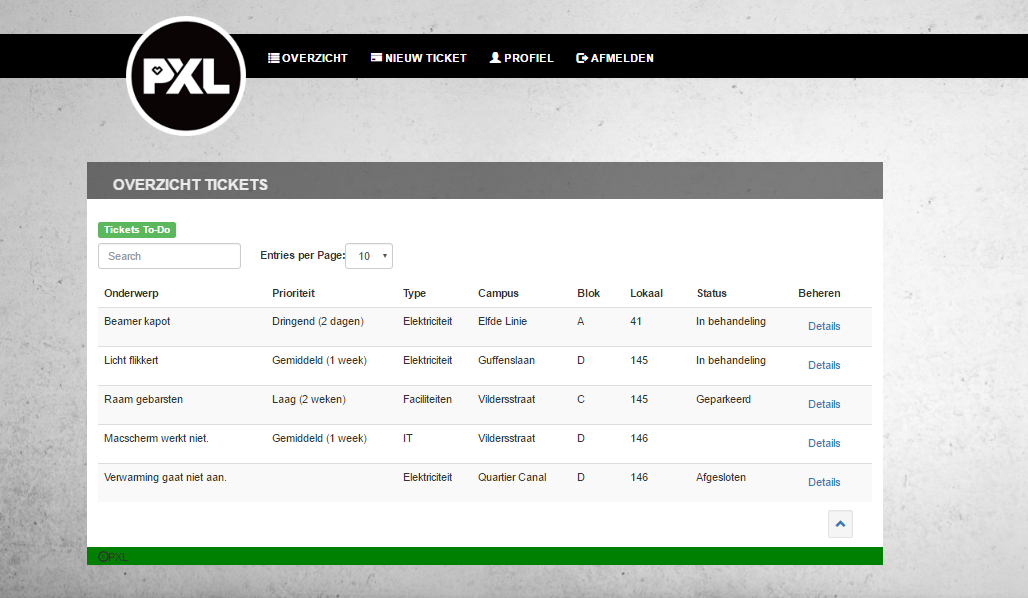


Hier ziet de werkman alle tickets die hij moet oplossen.

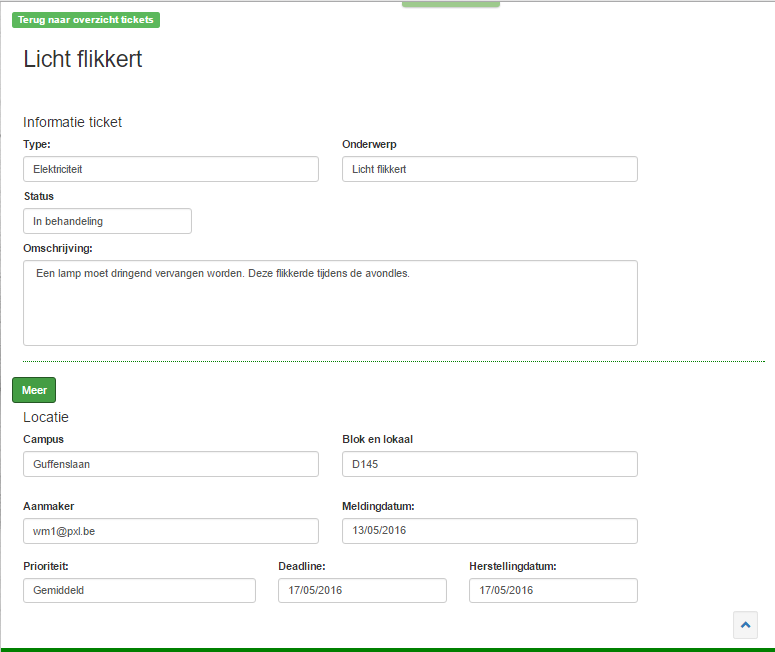
Door op details van een ticket te klikken krijgt hij een formulier te zien met alle informatie over de gekozen ticket. Hij kan de herstellingsdatum raadplegen om te zien wanneer hij dit probleem moet oplossen. Hij kan eveneens de deadline zien.



In dit formulier kan de werkman de status aanpassen en doorsturen nadat hij het probleem heeft opgelost. Door bovenaan op ‘Terug naar tickets To-Do’ te klikken en vervolgens op ‘Overzicht tickets’ krijgt de werkman een overzicht van alle tickets in het systeem.

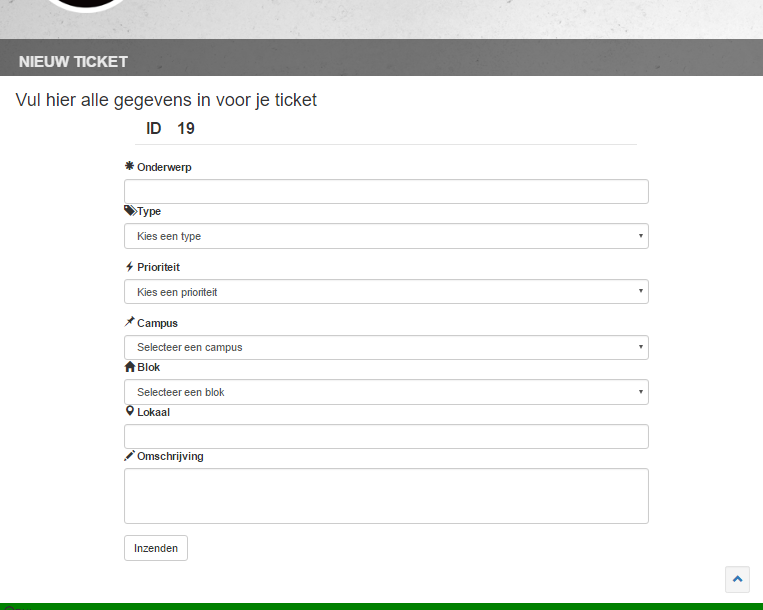


Door hier op de details van een ticket te klikken word er eveneens een formulier getoond met alle informatie.



In dit formulier heeft de werkman wel geen recht om iets aan te passen. Hij kan hier enkel de info raadplegen.

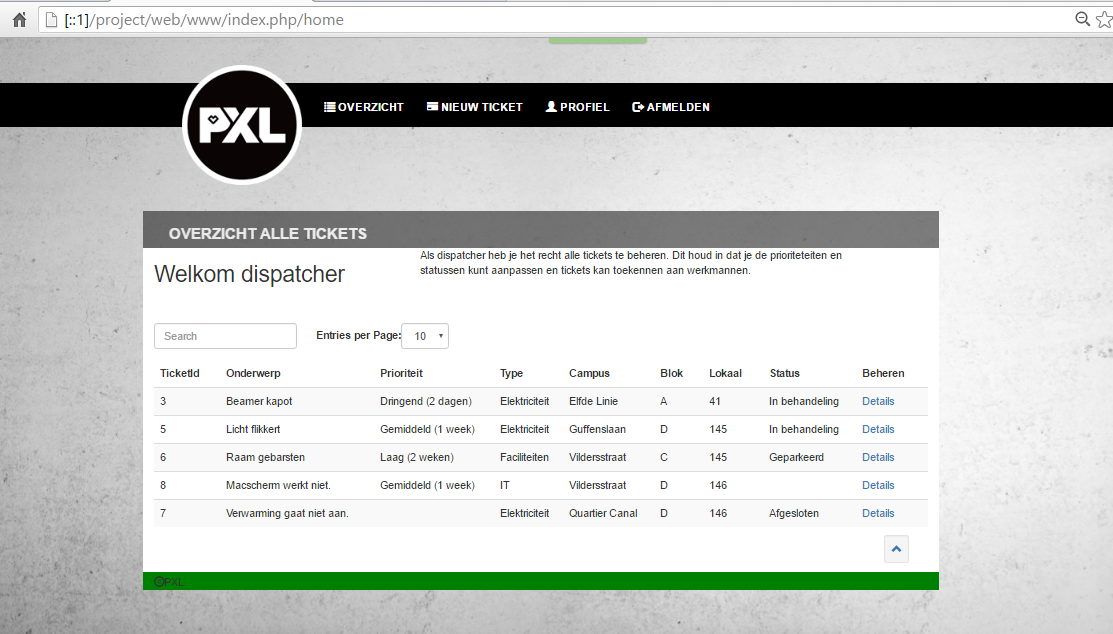
Navigatie nieuw ticket:



Is bij alle gebruikers hetzelfde (Dispatcher, werkman en docent). Hier kan de gebruiker een nieuw ticket aanmaken. Deze formulier is met validaties zodat niets verkeerd kan worden ingegeven.

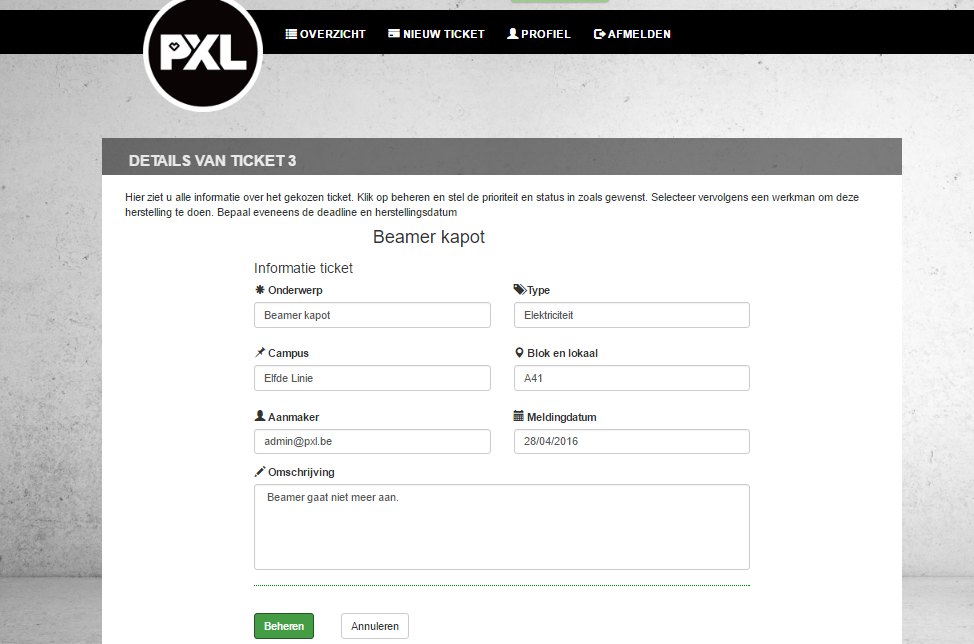
## Dispatcher

Overzicht alle tickets (home pagina dispatcher):

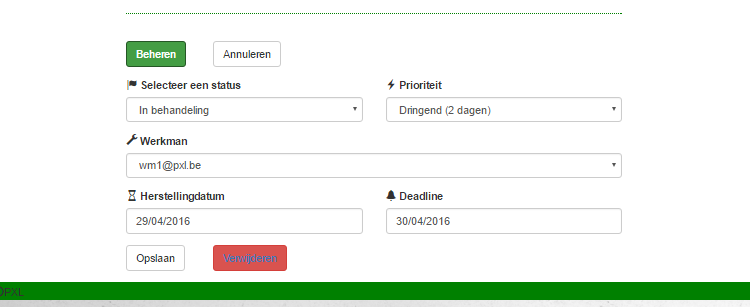


De dispatcher kan alle tickets zien en bewerken door op de detailslink van de ticket te klikken. (laatste kolom van de tickets).

Hier krijg de dispatcher alle informatie over de ticket.



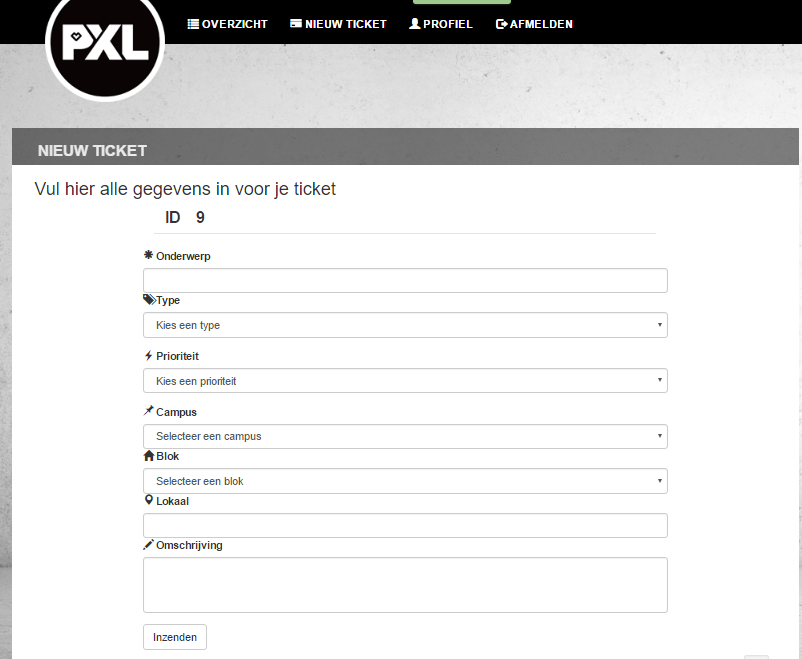
Na het klikken van “Beheren”:



Door de klikken op beheren kan de dispatcher bepaalde velden instellen of aanpassen. Bij het aanpassen zijn er validaties voorzien om verkeerde ingave te voorkomen.

Hij kan eveneens een ticket verwijderen.

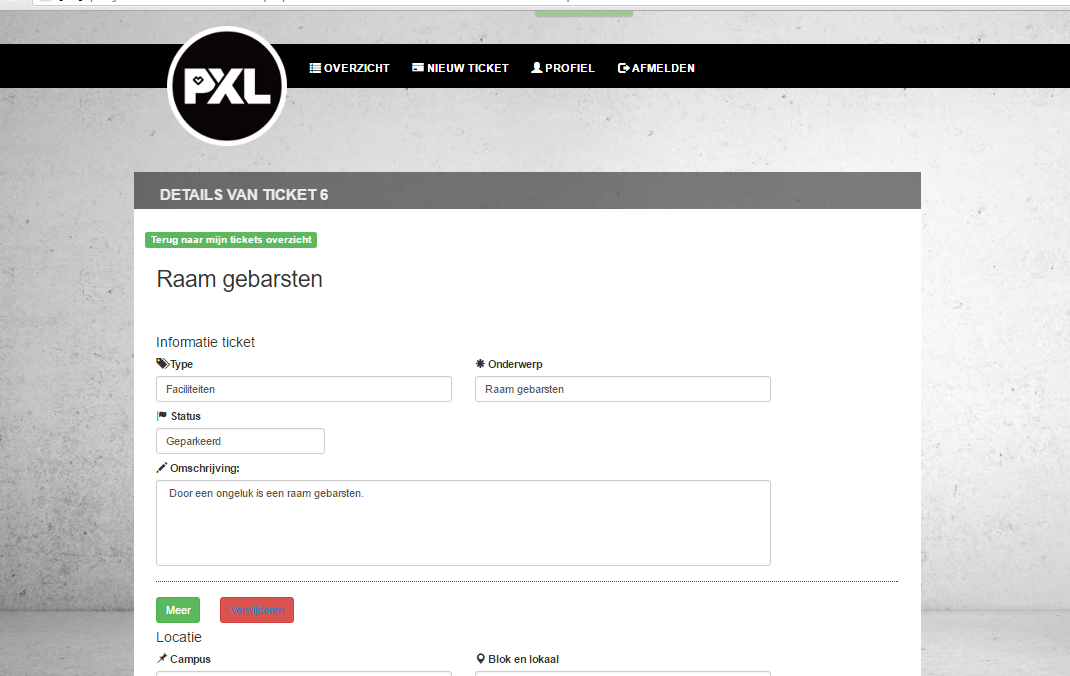
De dispatcher kan ook tickets aanmaken en zijn profiel bekijken.



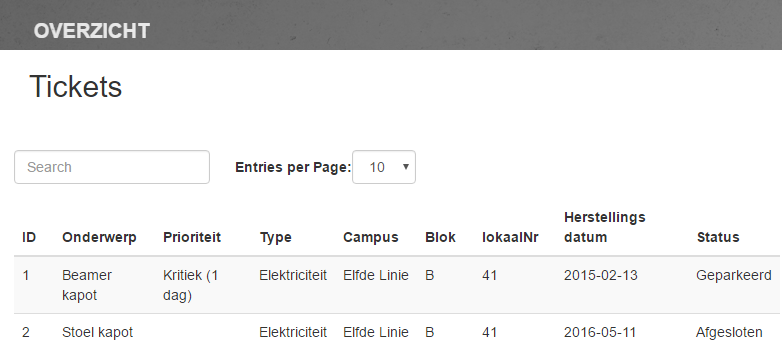
## 

Het eigen profiel overzicht is identiek voor dispatcher, werkman en docent. Hier kan de user een nieuw wachtwoord aanvragen. Maar ook de details van hun eigen tickets raadplegen.

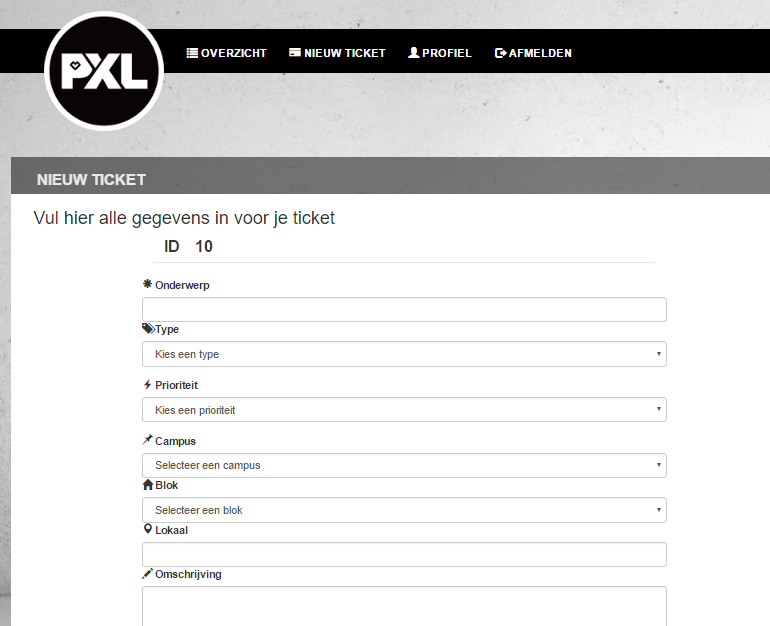
Bij het raadplegen van de zelf toegevoegde tickets kan deze ticket eveneens verwijdert worden.

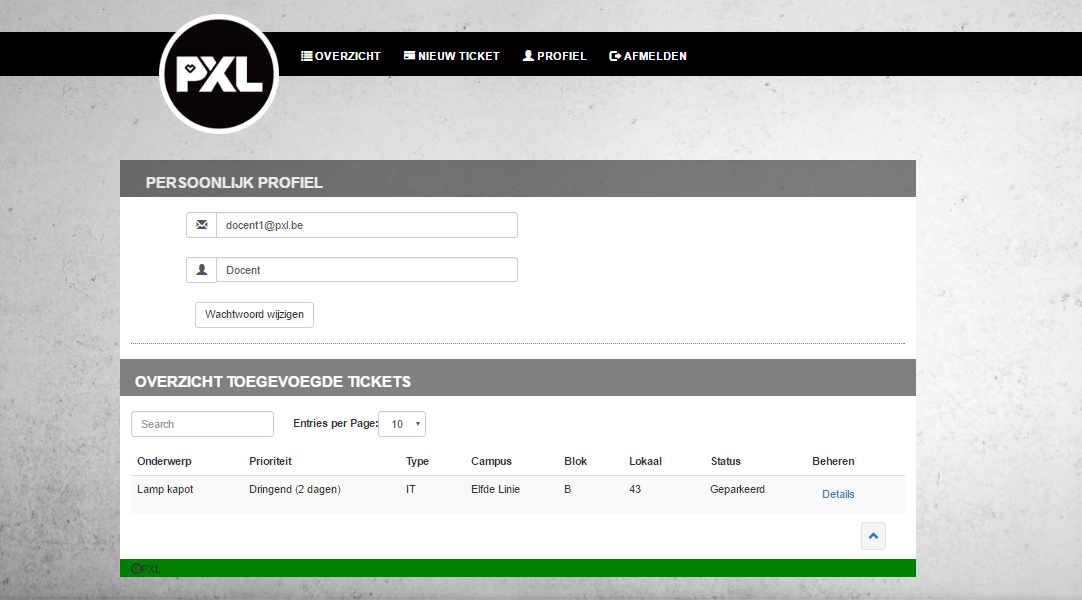


## Docent (user)

Overzicht eigen tickets (home pagina docent): 

Bij het inloggen ziet de gebruiker een overzicht van zijn zelf gemelde tickets. De docent kan ook tickets aanmaken en zijn profiel bekijken.





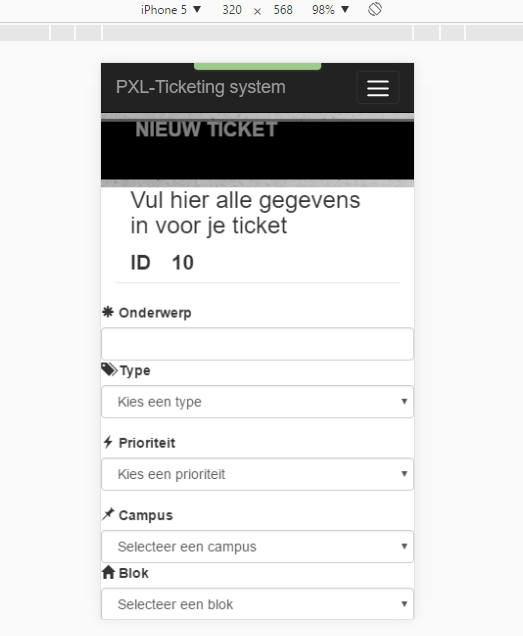
# Korte uitleg technologieën

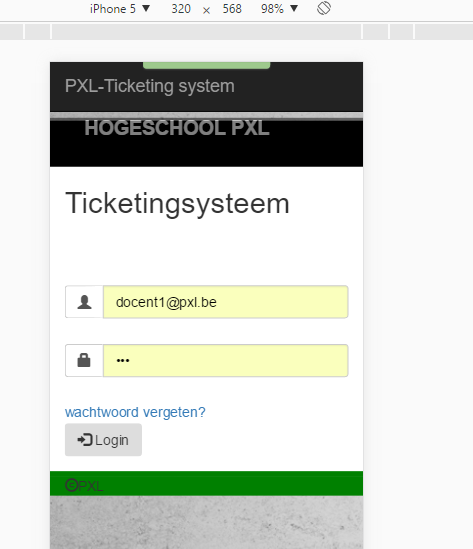
Het systeem in opgebouwd met codeigniter. We hebben gebruikt gemaakt van PHP,HTML, CSS, JS,….

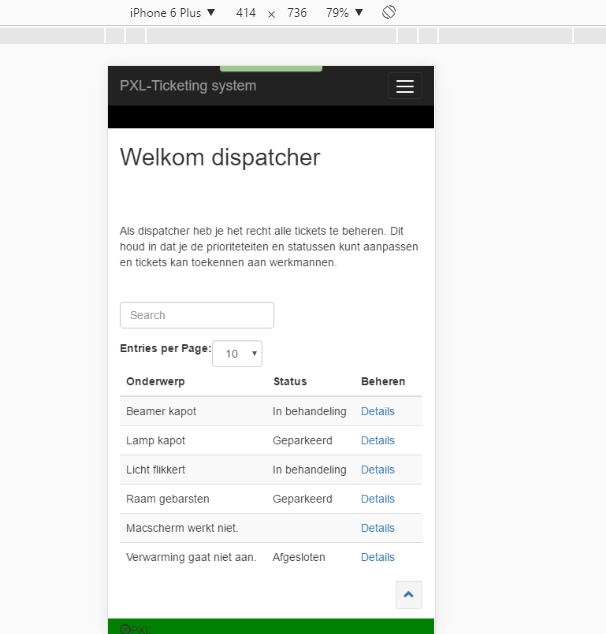
Alle tabellen zijn sorteerbaar door JS.

Het gehele systeem is ook responsive. Dit komt door het gebruikt van bootstrap. Alle vensters passen zich aan aan het scherm waarop het geopend is. De items in de header verdwijnen in een menu.

Enkele voorbeelden :







# Lijst van geleende code-fragmenten

Controllers: verifylogin.php, home.php, login.php

**class** Login **extends** CI\_Controller  
{  
 **function** \_\_construct()  
 {  
 **parent**::*\_\_construct*();  
 }  
  
 **function** index()  
 {  
 $this -> **load** -> view(**'login\_view'**);  
 }  
}

$this -> **session** -> unset\_userdata(**'logged\_in'**);  
session\_destroy();  
redirect(**'login'**, **'refresh'**);

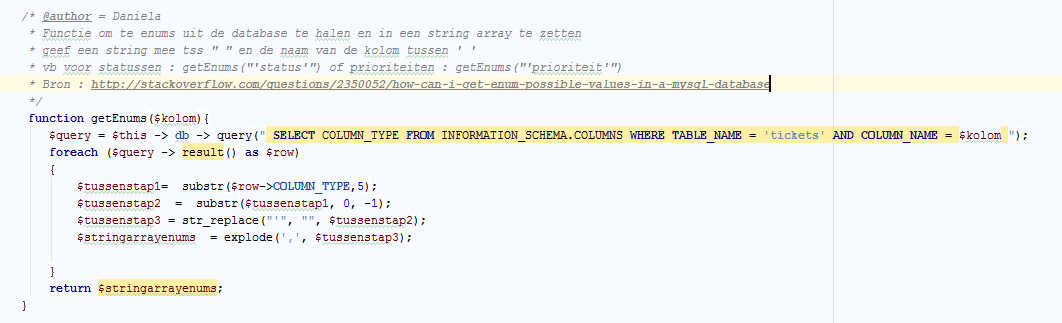
**function** logout()  
{  
 $session\_data = $this -> **session** -> userdata(**'logged\_in'**);  
 $userID = $session\_data[**'userID'**];  
 $rol = $this -> **user\_model** -> neem\_rol($userID);  
 $enquete = $this -> **user\_model** -> check\_enquete($userID);

Models: user\_model

function login :

$this -> **db** -> limit(1);  
  
$query = $this -> **db** -> get();  
  
*// Als de gebruiker gevonden wordt in de db***if**($query -> num\_rows() == 1)  
{  
 **return** $query -> result();  
}  
**else**{  
 **return false**;  
}

is gebaseerd op: <http://www.iluv2code.com/login-with-codeigniter-php.html>



Gebasseerd op: <http://stackoverflow.com/questions/2350052/how-can-i-get-enum-possible-values-in-a-mysql-database>

# Codeconventions

## Bekendmaking in commentaar

/\* @author = Naam

\* Date = dd/mm/yyyy

\* Bron = link fso

\* Uitleg functie

\*/

## Edit van code onder aanmaker zetten

/\* @edit = Naam

\* Date = dd/mm/yyyy

\* Bron = bron

\* Uitleg functie

\*/

## Algemene regels

* Inspringen met tab
* Keywords grote letters (FALSE, TRUE, ...)
* Variabele declaratie: camelcasing, \_, begint met kleine letter ditIsEenVoorbeeld1
* Klasses beginnen met een hoofdletter
* Methodes beginnen met een kleine letter

## Spaties

* No: $a=$b+$c; $this->load->view('');
* Yes: $a = $b + $c; $this -> load -> view('');
* geen spatie bij parameters: $a = getFoo($b);
* geen spatie bij geen paramater: $c = getBar();
* spaties bij commentaar: // dit: is een goed voorbeeld //dit is geen goed voorbeeld (geen spatie bij begin)
* functies en lussen: function test() { hier komt code; }

## Controllers

* Geen echo/print
* Geen SQL

## Models

* geen html

## vieuws

* geen SQL