**Implementatie Omschrijving Pjotr Wisse 84669 Examen Opdracht opdracht\_2c**

In dit document zal ik documenteren welke aanpassingen ik heb gekozen om te implementeren in de website. Ik zal vertellen waarom ik hiervoor heb gekozen en ik zal ook omschrijven hoe ik de aanpassing heb geïmplementeerd in mijn website. Ik zal erbij afbeeldingen vertonnen en ik zal stukjes code ook mee willen delen.

# Afbeeldingen Toevoegen Voor Reis.

In principe had ik al het idee om afbeeldingen te implementeren in mijn website. Ik heb al in mijn ERD-document ook een database kolom gezet waar staat dat een afbeelding, of tenminste de verwijzing naar een afbeelding, wordt opgeslagen. Maar ik zal wel omschrijven hoe ik dit heb kunnen implementeren in mijn website.

## H.1.1 De database

In de database heb ik ervoor gezorgd dat voor een reis een string van de locatie van een afbeelding wordt opgeslagen.

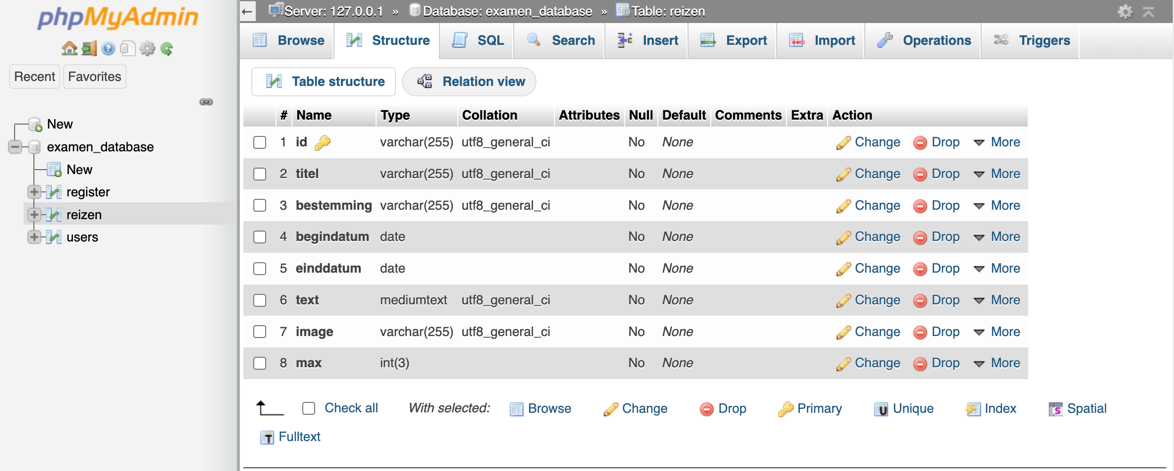
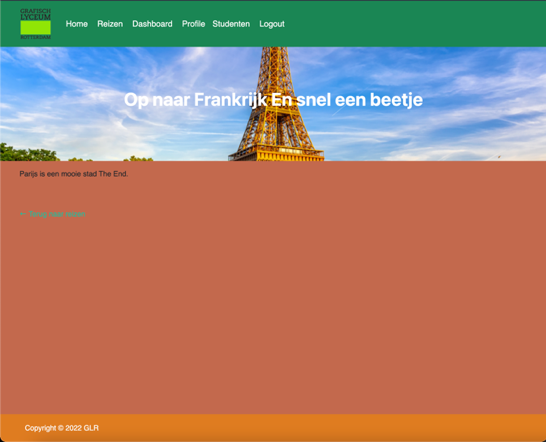


Image is hier de variabel en het is een varchar die 255 karakters kan opslaan. Verder wordt de afbeelding opgeslagen op de server via de move\_uploaded\_file functie. Een wat bredere uitleg over hoe dit precies werkt, kun je vinden in het test verslag.



## H1.2 Op de pagina.

Op de reis pagina komt een hoofd afbeelding, die zou dan in de header van de pagina komen. Het doel is om de string van de database op te halen en die te plaatsen in de website. Dat doen we gewoon met een select query en dan zetten we de afbeelding in de pagina.

  
Met een mooi Resultaat zoals dit.



We doen het heel even makkelijk in de header stijl stoppen zodat het dynamisch blijft. En ik hou hem op important, omdat ik eerst niet de afbeelding te zien kreeg dus heb ik in de CSS veranderd om er altijd voor te zorgen dat de afbeelding wordt weergegeven.

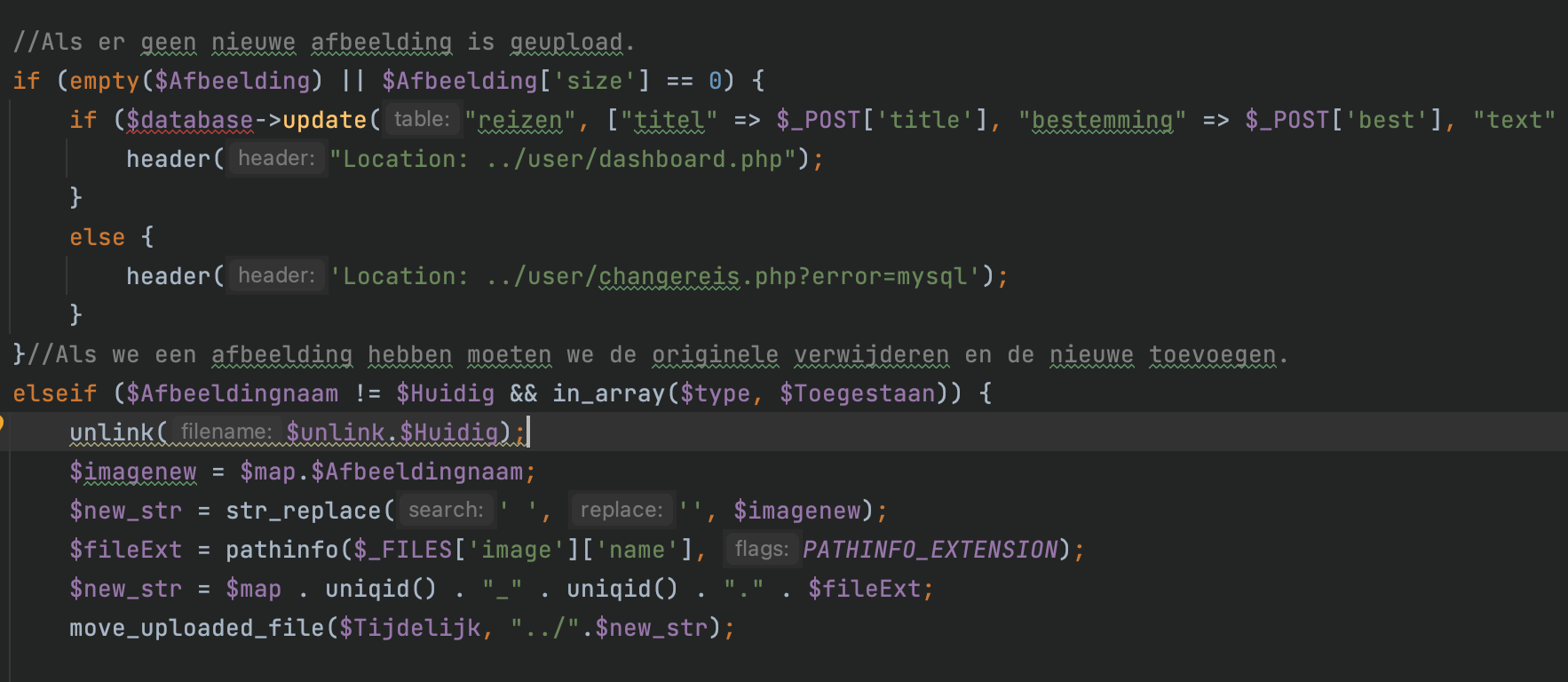
De afbeeldingen worden toegevoegd in de uploads folder via de move\_uploaded\_file functie. Met deze functie moeten we de tijdelijke naam van het bestand gebruiken en daarna zetten we waar dat bestand naar toe moet. Het geüploade bestand krijgt wel een uniek id, zodat als er twee van dezelfde afbeeldingen worden geüploade dat het dan iet de ander overwrite. De nieuw string moet ervoor zorgen dat het geen spaties heeft en bindend is met een uniek id. We moeten ook weten welke file extension we hebben en kijken of het mag worden gebruikt.



We moeten ook weten welke file extension we hebben en kijken of het mag worden gebruikt. Hieronder is de toegestaande files die mogen worden verwerkt, we vergelijken ook of de extensie in de array zit. Zodra die check is gedaan mogen we het bestand uploaden, naar de server.



Deze functie geld niet alleen voor reizen, maar ook voor profielen (later daar meer over). Als we een afbeelding willen veranderen moeten we eerst het originele bestand verwijderen en daarna de andere foto toevoegen. Als we geen afbeelding veranderen, dan hoeven niks te verwijderen of filenaam veranderen.



Als we een afbeelding willen verwijderen, dan unlinken we het bestand via de unlink functie, die zorgt ervoor dat het bestand van de server wordt gehaald.

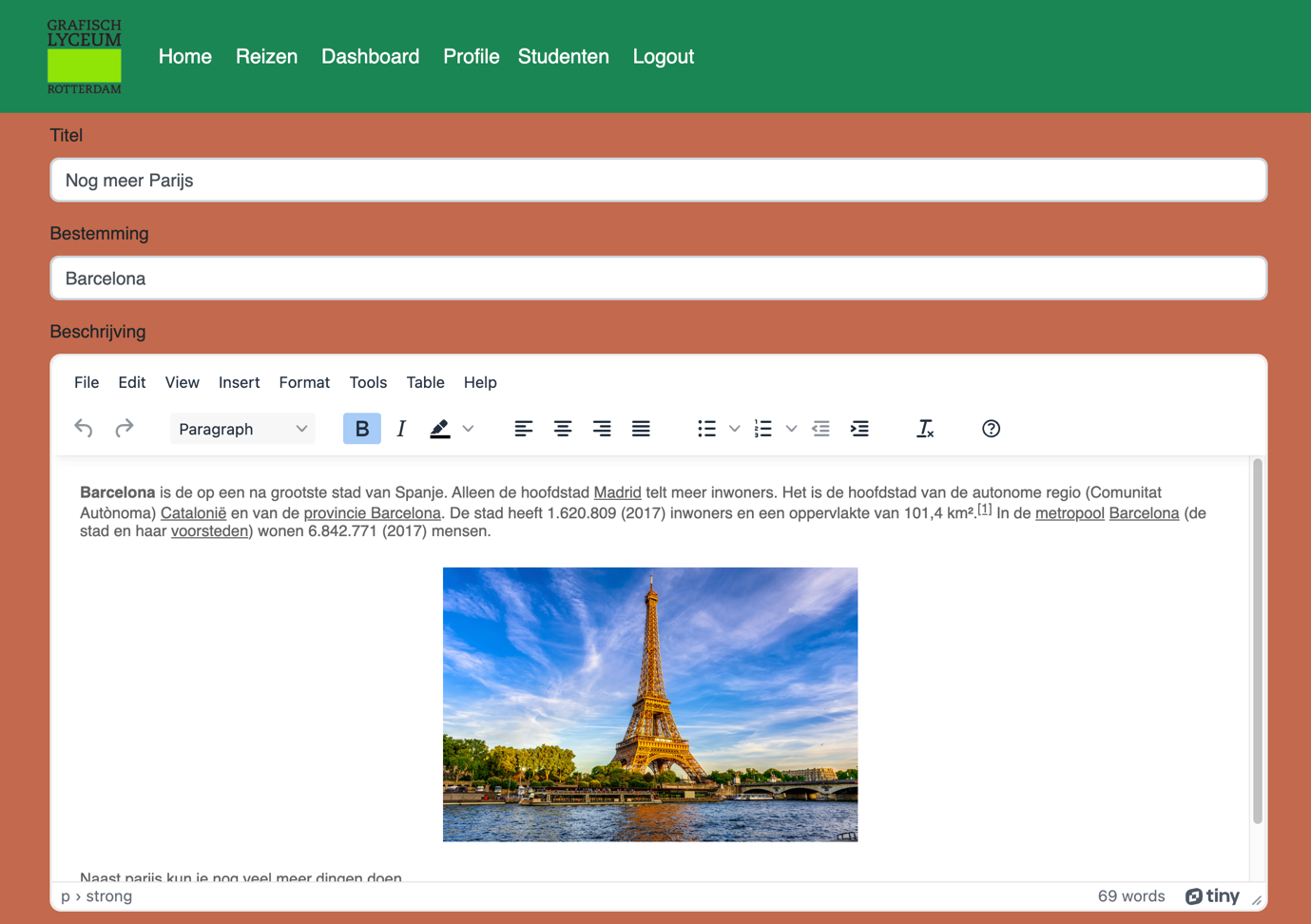


Dat was een beetje de back-end van hoe alles

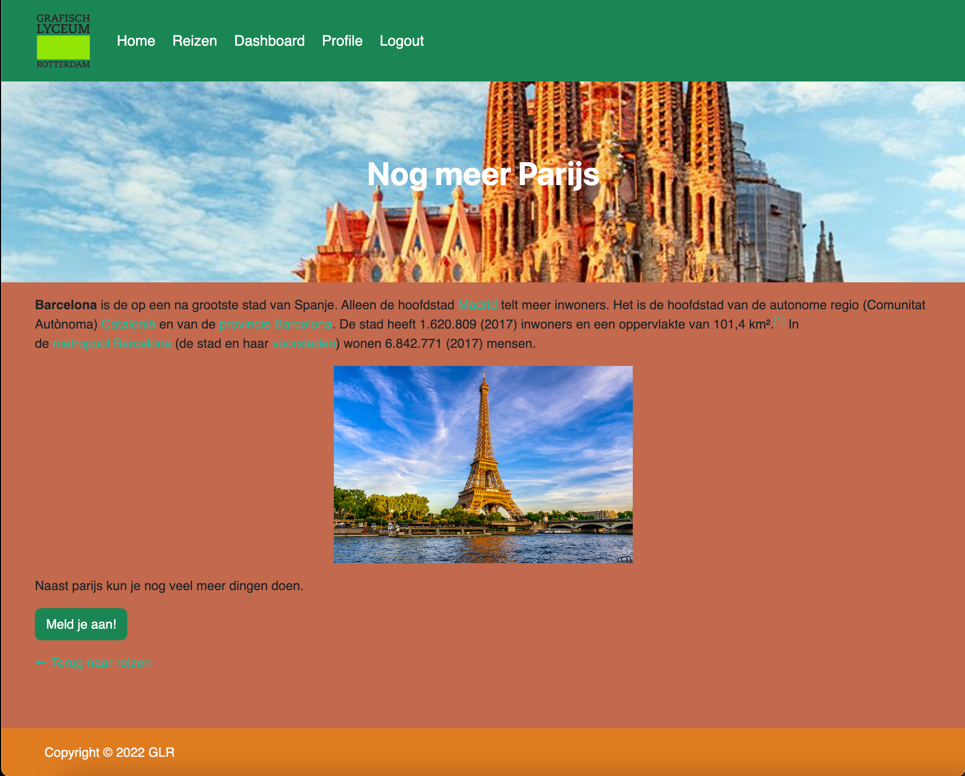
## H1.3 TinyMCE.

Als je trouwens afvraagt of we ook in de content onderaan ook afbeeldingen kunnen toevoegen, dan heb ik goed nieuws. In mijn website heb ik een rich text editor toegevoegd, TinyMCE. Zoals ik al heb omschreven in mijn test document kun je met TinyMCE, makkelijk implementeren in je bootstrap websites en je kan veel als je weet hoe je de script config moet bewerken.

Je kan er makkelijk tekst bij toevoegen en afbeeldingen toevoegen. Die worden, net zoals beschreven in het test formulier, opgeslagen in blobs (binary large objects). Hierdoor verminderen we de ruimte die afbeeldingen meestal nemen op de server nemen. Hier is een voorbeeld van hoe het eruit ziet:



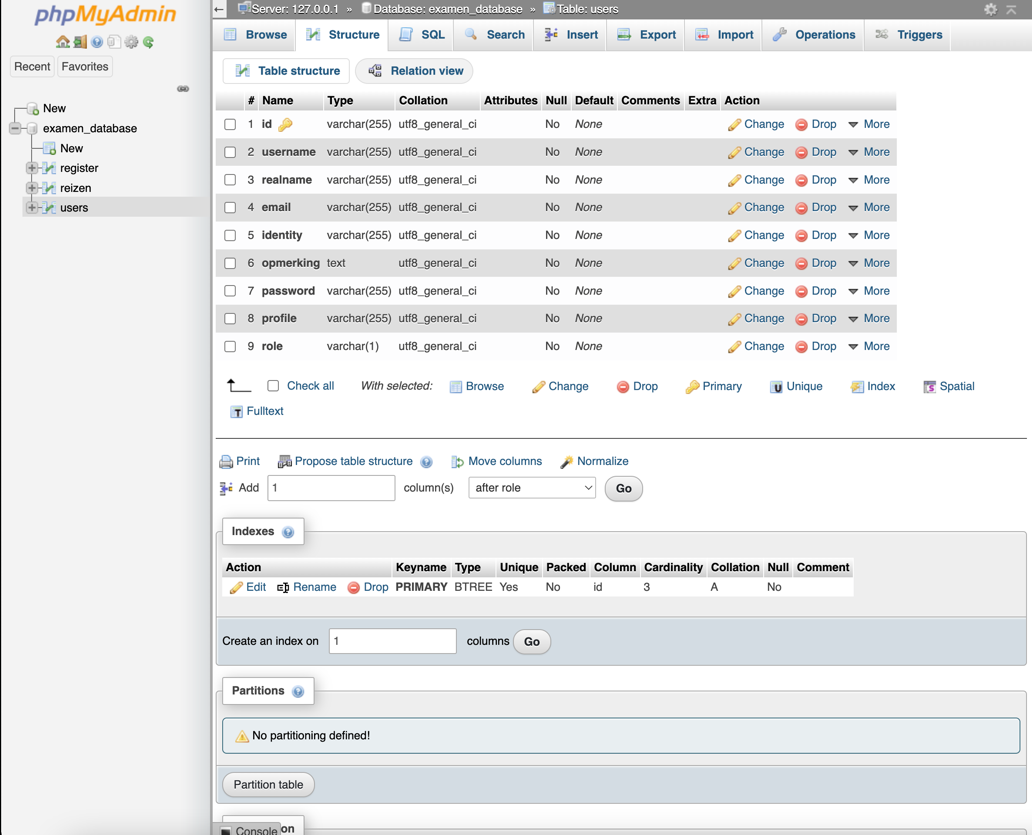
Hier is het aanmaken van de reis.



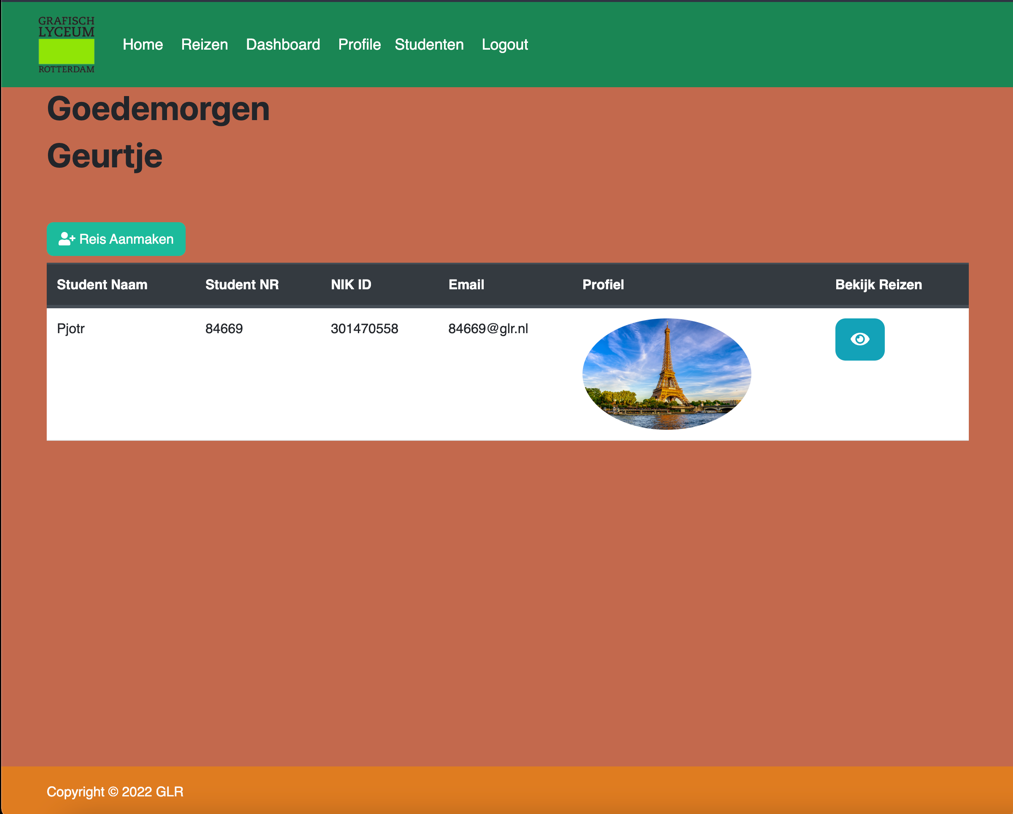
En hier is het uiteindelijke resultaat. We zien dus dat met TinyMCE we veel mogelijkheden hebben aangezien we makkelijk allemaal content in een textarea kunnen toevoegen. Gelukkig hoeven we in de backend niet heel veel te doen, we hoeven alleen maar de post mee te brengen en TinyMCE en Medoo doen de rest.

## H1.4 Profiel foto’s.

Als een klein extraatje heb ik besloten om ook ervoor te zorgen dat studenten en admins een profiel foto moeten uploaden. Ik gebruik hetzelfde type dataopslag als in de reis tabel. Maar nu doe ik het voor users, de string van de foto wordt opgeslagen in profile.



Een goed voorbeeld om te laten zien is in de studenten dashboard van de admins. Daar staat informatie over de student, met natuurlijk hun profiel foto ernaast.



Hier is een werkend voorbeeld van de studenten dashboard. Hierin wordt de afbeelding opgehaald en in de src gezet van het image element. En daarmee kunnen we deze implementatie afsluiten, aangezien.

# Maximum studenten.

Ik wilde proberen om nog maximaal aantal studenten toe te voegen bij de applicatie. Ik heb in de database, boven weergegeven, heb ik een max aantal toegevoegd. Als dat max aantal wordt bereikt dan zou het systeem niet meer verder moeten tellen en geen extra aanmeldingen toevoegen.

Met medoo kan ik wellicht heel simpel via een count query opzoeken hoeveel er al zijn aangemeld en het vergelijken met een andere query waar ik het maximum getal vandaan haal. Maar voor dat we dat gaan doen moeten we er eerst voor zorgen dat dezelfde student niet twee keer kan inschrijven voor een reis. Dat doen we via een simpele select query waar we vragen of het studenten id al eens een keer voor kwam in het register tabel bij een specifieke reis.

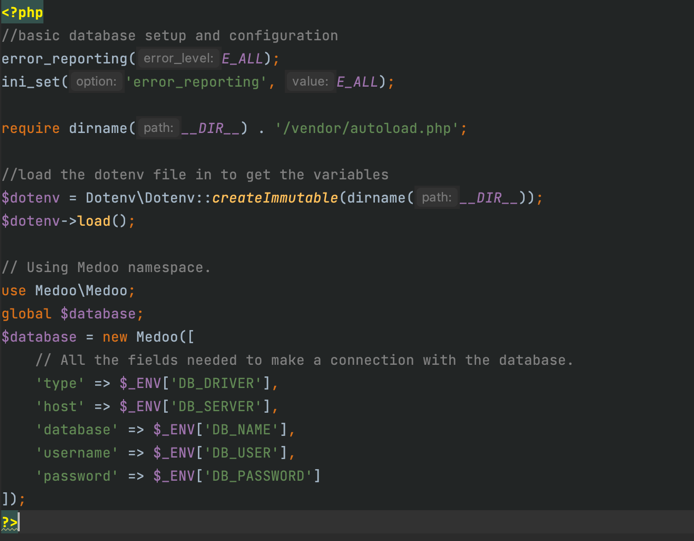


Nu naar de maximum. Op de reis pagina kun je alleen de knop aanklikken als de max nog niet is bereikt. Als het max nog niet is bereikt dan mag de knop getoond worden.



# Extra: Medoo

Medoo is een PHP library waar je makkelijk SQL data kan versturen. Het automatiseert het gehele SQL proces. Alle variabelen worden bekeken voor speciale karakters. Het stuurt ze op via prepared statements en zet de data om in PDO (PHP Data Objects). Hierdoor voorkom je SQL injection, bij PDO maakt het dan niet uit welke type database je gebruikt, en het minimaliseert bandbreedte op de server en computer. Ik had eerder nog niet Medoo gebruikt in een web project en het leek heel makkelijk om te gebruiken. Dus toen had ik besloten om het te leren en ging aan de slag met het maken van een werkend Medoo Model. Medoo is zeer makkelijk om toe te voegen tot je website, hier is hoe verbindt met de database:



Hier wordt via PHPENV, een applicatie waarmee je omgevingsvariabelen kan gebruiken, ervoor gezorgd dat de database een nieuwe Medoo verbinding wordt. Ik ben heel erg positief gebleven over Medoo en hoop het in de toekomst nog meer te gebruiken aangezien het wel mijn PHP development heeft versneld in de eerste twee dagen.

# Afsluiting.

In dit document hebben we besproken welke implementaties ik heb toegevoegd tot de website. Ik had in ieder geval gekozen om afbeeldingen toe te voegen aangezien ik daar zelf al voor had gepland, maar ik wilde ook iets nieuws uit proberen. Daarom heb ik nog maximum studenten gekozen. Om ervoor te zorgen dat studenten, weten wanneer ze wel en niet kunnen aanmelden. Zodra het vol is, is het vol. Ik hoop dat met dit document iets meer duidelijkheid heb gegeven aan de veranderingen binnen in mijn applicatie.