WINDOWS KIOSK MODE - LINUX SERVER LOGS NAAR ELK

[ABSTRACT 2](#_Toc133159478)

[INTRODUCTIE 3](#_Toc133159479)

[VERWERKING 4](#_Toc133159480)

[INSTALLATIE MACHINES 4](#_Toc133159481)

[Windows 11 Kiosk Mode 4](#_Toc133159482)

[Installatie Virtuele Machine 4](#_Toc133159483)

[Kiosk Modus Instellen 5](#_Toc133159484)

[Linux Main Machine 5](#_Toc133159485)

[Installatie Virtuele Machine 6](#_Toc133159486)

[Debian Machine Configureren 7](#_Toc133159487)

[Kiosk Applicatie 8](#_Toc133159488)

[Laravel Installatie 8](#_Toc133159489)

[Installatie van PhpMyAdmin 13](#_Toc133159490)

[Creatie van de applicatie 15](#_Toc133159491)

[Aanmaken van een Registratie Model 19](#_Toc133159492)

[Aanmaken van een Seeder 20](#_Toc133159493)

[Aanmaken van een Service & Controller 21](#_Toc133159494)

[Creatie applicatieweergave 23](#_Toc133159495)

[Registratie Implementatie 27](#_Toc133159496)

[Tussenevaluatie 28](#_Toc133159497)

[ELK 29](#_Toc133159498)

[ONDERZOEK 30](#_Toc133159499)

[HULPMIDDELEN 31](#_Toc133159500)

[CONCLUSIE 32](#_Toc133159501)

# ABSTRACT

Samenvatting van paper en de conclusies (op einde)

# INTRODUCTIE

In deze paper wordt beschreven hoe een omgeving tot stand is gekomen waarbij een Linux client fungeert als hoofd pc van een netwerk.

De Linux client is verbonden met één of meerdere Windows Clients die in kiosk mode draaien. Het doel van deze omgeving is om een specifieke applicatie te draaien die vanuit de Linux client verzonden moet worden naar de Windows Clients.

De paper beschrijft de stappen die zijn genomen om dit systeem op te zetten en de uitdagingen die zijn overwonnen om tot een werkende oplossing te komen.

Het resultaat van dit onderzoek is een stabiele en efficiënte omgeving waarin de gewenste applicatie probleemloos gedraaid kan worden op de Windows Clients.

# VERWERKING

## INSTALLATIE MACHINES

### Windows 11 Kiosk Mode

In onze realisatie van dit probleem willen wij gebruik gaan maken van één of meerdere kiosk machines met daarop een registratie applicatie om in een wachtzaal te gebruiken.

Dit zullen wij simuleren met behulp van een **virtuele machine** die draait op Windows 11 met daarop de Kiosk-Modus actief.

Deze modus zal specifieke instellingen verkrijgen om enkel en alleen de nodige functionaliteiten uit te voeren.

#### Installatie Virtuele Machine

1. Installeer een virtualisatiesoftware zoals VMware Workstation (versie 17!) dop de host machine.
2. Download de Windows 11 ISO van de Microsoft-website en installeer het als virtuele machine.
   1. De meest recente versie kan je vinden via volgende link: [Github Howest - Data - ISO File (Windows)](https://github.com/Howest-2021-2024/Data/blob/main/README.md#iso-file)
3. Maak een nieuwe virtuele machine aan en zorg ervoor dat je volgende stappen goed volgt:
   1. Kies voor een **Typical Install**
   2. Kies voor: **I will install the operating system later**
   3. Kies the juiste Microsoft Windows versie (Windows 11 x64)
   4. Encrypteer de machine met een wachtwoord, **KioskGroup12**
   5. Kies voor **Store virtual disk as a single file**
4. Als alles goed verlopen is heeft u een virtuele machine, nu moeten we nog het eerder gedownloade iso-bestand koppelen.

Hiervoor moeten we enkele instellingen aanwezig hebben:

* **Trusted Platform Module**: Present
* **CD/DVD (SATA)**: De locatie van het iso-bestand

1. Sla de instellingen op, start de machine en installeer de windows machine zoals u normaliter zou doen. Zorg er enkel voor dat u kiest voor **Windows 11 Pro**

#### Kiosk Modus Instellen

1. Open ‘**Instellingen**’ > ‘**Accounts**’ > ‘**Andere Gebruikers**’ > ‘**Kiosk**’
2. Kies als naam **Wachtzaal**
3. Kies als app **Microsoft Edge**
4. Klik verder en gebruik de standaard instellingen
5. Type als url **www.howest.be,** later zal dit vervangen worden met het adres van de Linux hoofdmachine die onze applicatie zal draaien.
6. Zet de inactiviteitslimiet op **1** uur

U heeft nu een werkende kiosk-modus machine! Herstart uw machine en als alles goed gaat start deze automatisch de ingestelde website en kunt u geen andere acties uitvoeren.

### Linux Main Machine

De Linux machine zal dienen als hoofd pc van het netwerk en zal fungeren als centrale hub voor de Windows Clients in de kioskmodus. Het zal verantwoordelijk zijn voor het verzenden van de vereiste applicaties naar de Windows Clients, evenals het beheren van de beveiliging en netwerkinstellingen van het netwerk.

Een reden om een Linux-Windows verdeling te doen is omdat beide besturingssystemen hun eigen specifieke voordelen hebben.

Linux is bekend om zijn stabiliteit, veiligheid en flexibiliteit, terwijl Windows wordt geprezen om zijn gebruiksvriendelijkheid en toegankelijkheid voor een breed scala aan gebruikers.

Door beide besturingssystemen samen te voegen, kunnen we de sterke punten van elk systeem benutten en een efficiënt en robuust netwerk opzetten dat aan de vereisten van de gebruiker voldoet.

#### Installatie Virtuele Machine

1. We zullen nog een nieuwe virtuele machine aanmaken, zorg ervoor dat je volgende stappen goed volgt.
   1. Installeer allereerst de nodige versie van Linux Debian, het iso-bestand kan je hier vinden: [Github Howest - Data - ISO File (Debian)](https://github.com/Howest-2021-2024/Data/blob/main/README.md#debian-iso-file)
   2. Kies voor een **Typical Install**
   3. Kies ervoor een iso file mee te geven en selecteer het bestand die je net hebt gedownload.
   4. Vervolg de installatiewizard met de aanbevolen instellingen.
2. Als alles goed verlopen is heeft u een extra virtuele machine, start deze op.
3. Nu zullen we de Debian Machine volledig instellen
   1. Kies een taal naar voorkeur, **Engels** is aanbevolen.
   2. Kies als country **Belgium** (other/Europe/Belgium)
   3. Kies US als lokale instelling.
   4. Kies een toetsenbord lay-out naar keuze.
   5. Kies een hostname, wij gebruiken doorgaans onze documentatie **LinuxMain**.
   6. Kies ‘**local**’ als domeinnaam
   7. Geef een passwoord in, **KioskGroup12** is aangeraden.
   8. Kies voor ‘**guided-use entire disk**’
   9. Kies **sda** als partitie
   10. Kies voor ‘**all files in one partition**’
   11. Kies voor ‘**finish partitioning and write changes to disk**’
   12. **Bevestig** het schrijven naar de schijf
   13. Scan **niet** verder
   14. Kies **ja** om een netwerk mirror in te stellen:
       1. Kies **Belgium**
       2. Kies **deb.debian.org**
       3. Ga verder zonder proxy op te geven
       4. Neem geen deel aan de survey
       5. Selecteer (door enkel spatie te gebruiken) de **standard system utilities** en **SSH server** (deactiveer alle andere opties)
   15. Kies **ja** om GRUB te installeren
   16. Kies **/dev/sda**
   17. Rond de installatie af

#### Debian Machine Configureren

In de volgende stappen zullen we onze machine gereedmaken om onze toekomstige applicatie te kunnen hosten.

We zullen hiervoor enkele permissies, packages en andere implementaties moeten wijzigen/toevoegen.

1. Log in op uw net gemaakte machine met de gekozen login-informatie
2. Gebruik het commando ‘**ip a**’om het ip-adres van uw machine te vinden.
3. Op uw **hostmachine**, maak een ssh connectie naar de Debian machine:
   1. Voorbeeld: “ssh <HOSTNAAM\_DEBIAN>@<IP\_DEBIAN>”

*(vervang de parameters zonder de bekjes te laten staan)*

1. Gebruik het commando ‘**su –**‘om als root gebruiker te werk te gaan
2. Voer de volgende commando’s uit:
   * apt update
   * apt install sudo
   * usermod -aG <UW\_GEBRUIKERSNAAM>

*(vervang de parameters zonder de bekjes te laten staan)*

1. Ga terug naar uw normale gebruiker (‘**exit**’)/(‘ **su** <UW\_GEBRUIKERSNAAM>’)

*(vervang de parameters zonder de bekjes te laten staan*

1. Voer volgende commando’s uit:

* Git VCS installeren: ‘**apt install git**’
* Apache2 installeren: ‘**sudo apt install apache2**’

1. Maak een map aan in de home directory
   1. ‘**mkdir code**’
   2. Zorg ervoor dat je een **lege repo** hebt waarin u later de applicatie zult gaan maken.

Heeft u dit nog niet? Volg volgende stappen:

[Repository Aanmaken (YouTube)](https://youtu.be/9orjmYFQTWo)

* 1. Maak een kopie van uw repository in uw gemaakte map
     1. ‘**cd code**’
     2. ‘**git clone** <HTTPS\_LINK\_VAN\_UW\_REPO>’
  2. Nu zullen we deze repo beschikbaar maken voor onze apache server.
     1. Voer het volgende commando uit: ‘**ln -s ~/code/NaamVanUwRepo ~/../../var/www**’
     2. Ga vervolgens naar de beschikbare sites map:
        1. ‘**cd /etc/apache2/sites-available**’
     3. Voer er volgende commando’s uit:
        1. ‘**sudo nano 000-default**’
           1. Verander ‘DocumentRoot /var/www/’ naar ‘**DocumentRoot /var/www/NaamVanUwRepo**’
     4. Om deze wijzigingen door te voeren dient u nog enkel de apache service te herstarten.

‘**sudo systemctl restart apache2’**

1. Als alles goed ging is de repo nu toegankelijk op de Debian machine als website. Als u het IP-adres van uw Debian Machine ingeeft in de zoekbalk van een browser.

We zullen dit gebruiken om onze toekomstige applicatie efficiënter te kunnen updaten en uiteindelijk deployen.

### Kiosk Applicatie

Nu is het duidelijk dat er een applicatie zal draaien op de kiosk machines, deze moet alleen nog gemaakt worden. Dit is wat we in dit deel behandelen, hoe we dit tot stand krijgen met de nodige voorbereidingen.

#### Laravel Installatie

Onze applicatie zullen we maken met behulp van het framework Laravel. Deze bevat alle basis-functionaliteiten die we nodig hebben. Denk aan database migraties, seeders, api-requests,…

**PHP Installatie**

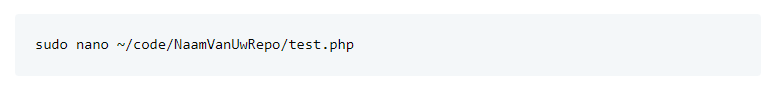
**De volgende stap is het installeren van PHP. Gelukkig zit PHP 7 standaard in de officiële repositories van Ubuntu, wat de installatie heel eenvoudig maakt. U moet de taal zelf en een extra module installeren. Voer hiervoor de volgende opdracht uit:**

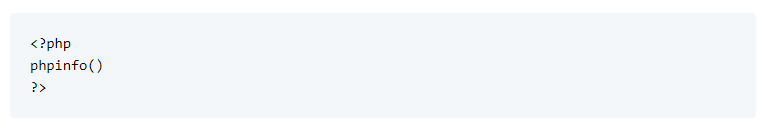
**‘sudo apt install php libapache2-mod-php php-mbstring php-xmlrpc php-soap php-gd php-xml php-cli php-zip php-bcmath php-tokenizer php-json php-pear’**

Als de opdracht een uitvoer opleverde waarin stond dat sommige pakketten niet werden gevonden, werkt u gewoon uw Ubuntu bij door de volgende opdracht uit te voeren en voert u de vorige opnieuw uit:

**‘apt-get update’**

Nu kunnen we testen of PHP correct werkt. Om dit te doen, moeten we een bestand maken in de hoofdmap van Apache, deze is momenteel uw eerder aangemaakt repository. Laten we het **test.php** noemen. Voer de volgende opdracht uit:

In het texteditor venster die opent, typt u volgende code:



We moeten het opslaan en sluiten. Om op te slaan, drukt u op **CTRL+O** en om af te sluiten, drukt u op **CTRL+X**. Open vervolgens de webbrowser en ga naar **http://JouwApacheServerIP-Adres/test.php**

Als alles goed is verlopen zou u iets als dit moeten zien:

Afbeelding met tekst

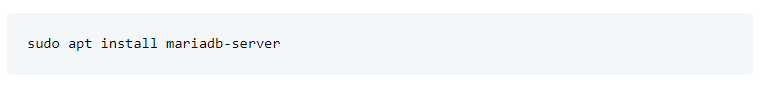
Automatisch gegenereerde beschrijving

**Download en installeer een databasemanager**

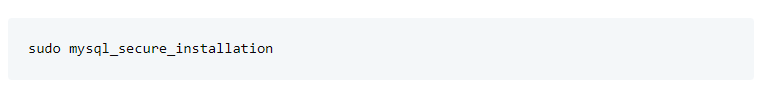
Als we gaan ontwikkelen met Laravel in Ubuntu, is het daarvoor nodig om een databasemanager te installeren.

Laravel ondersteunt PostgreSQL, MySQL, MariaDB, SQLite en SQL-server. We kunnen degene installeren en configureren die we willen. Voor deze case zullen we **MariaDB** installeren.

Voer het volgende commando uit:



Daarna kunt u een wachtwoord instellen voor de root. Om dit te doen, moet u het mysql\_secure\_installation-script gebruiken. Houd er rekening mee dat deze stap optioneel is, maar om veiligheidsredenen wordt aanbevolen.



Nadat we het root-wachtwoord hebben gedefinieerd, worden ons verschillende MariaDB-configuratievragen gesteld. De antwoorden die u moet invoeren, staan naast de coderegels:

Afbeelding met tekst

Automatisch gegenereerde beschrijving

Als alles goed ging, krijgt u dit te zien!

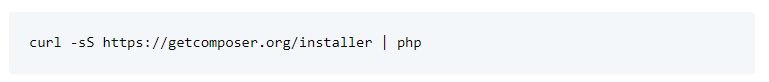
Afbeelding met tekst

Automatisch gegenereerde beschrijving

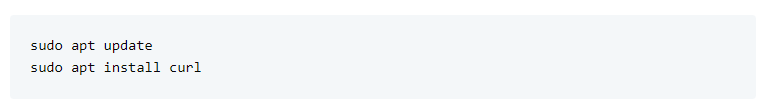
**Composer installeren**

Composer is een PHP-afhankelijkheidsmanager die het downloaden van PHP-bibliotheken in onze projecten vergemakkelijkt. Composer werkt geweldig samen met Laravel en maakt het veel gemakkelijker om Laravel te installeren.

Eerst moeten we Composer downloaden:



Indien het **curl** commando nog niet gekend is, moeten we deze package nog installeren, dit kan je doen door het volgende commando uit te voeren:



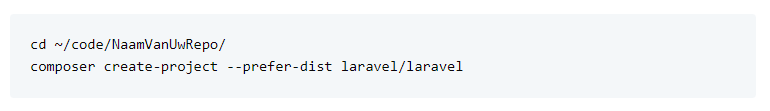
Vervolgens moeten we ervoor zorgen dat Composer globaal kan worden gebruikt en het uitvoerbaar maken. De volgende commando's zullen daarvoor zorgen.

Afbeelding met tekst

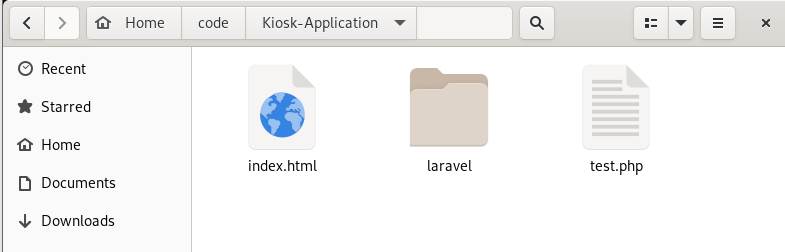
Automatisch gegenereerde beschrijving

**Installeer Laravel op Ubuntu met behulp van Composer**

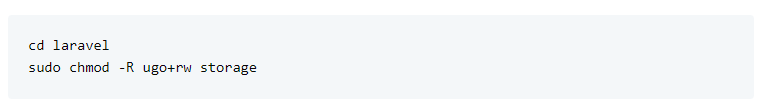
Nu Composer is geïnstalleerd, kunnen we Laravel nu installeren. Voer hiervoor de volgende opdrachten uit:



Uw repo moet er nu als volgt uitzien:



Omdat de het project de file permissies heeft voor een webserver (Apache) moeten we nog een permissie wijzigen om het laravel-project te kunnen bekijken via de browser.

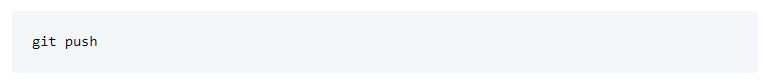


Laten we ervoor zorgen dat deze applicatie in onze repo toegankelijk is.

Afbeelding met tekst

Automatisch gegenereerde beschrijving

Indien je deze wijzigingen wilt versturen.



***Het is mogelijk dat dit niet lukt door het gebrek aan permissies/de auteur niet gekend is****.*

Volg dan deze stappen:

[GitHub Token Video](https://youtu.be/3TybMOu-_E8)

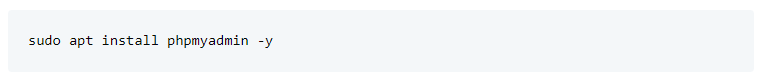
Dit zou het probleem moeten oplossen…

***Nu is onze Laravel-applicatie beschikbaar in onze repo, nu kunnen we deze maken.***

#### Installatie van PhpMyAdmin

We hebben de ondersteuning van MariaDB reeds geïnstalleerd maar voor de applicatie zullen we gebruik maken van een phpmyadmin database. Hiervoor moeten we uiteraard de nodige diensten voor installeren.

U kunt phpMyAdmin installeren voor het beheer van MySQL/MariaDB-databases vanuit het comfort van een webbrowser met behulp van de volgende opdracht:



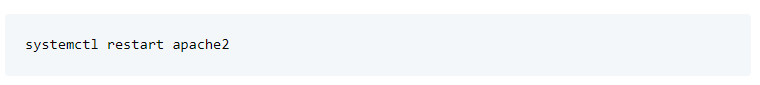
Afbeelding met tekst

Automatisch gegenereerde beschrijving

Tijdens het installatieproces van het pakket wordt u gevraagd om de webserver te kiezen die automatisch moet worden geconfigureerd om phpMyAdmin uit te voeren, u doet dit door **apache** te selecteren door op de **spatiebalk** te drukken en op **Enter** te drukken.

Voer vervolgens het wachtwoord in voor de beheerder (**KioskGroup12**) van MySQL/MariaDB, zodat het installatieprogramma een database voor phpmyadmin kan maken.

Nadat alles is geïnstalleerd, kunt u de apache2-service opnieuw starten om de recente wijzigingen door te voeren.



De root-login zal mislukken bij phpMyAdmin. We moeten een nieuwe gebruiker maken en alle rechten aan die gebruiker toekennen.

Om de volgende aanpassingen te verrichten dienen we als **root** gebruiker te werken:

Afbeelding met Rechthoek

Automatisch gegenereerde beschrijving

Om een nieuwe gebruiker aan te maken, logt u in op MariaDB met de volgende opdracht:

Afbeelding met Rechthoek

Automatisch gegenereerde beschrijving

Als alles goed verloopt heeft u nu toegang tot de MariaDB Shell:

Afbeelding met tekst

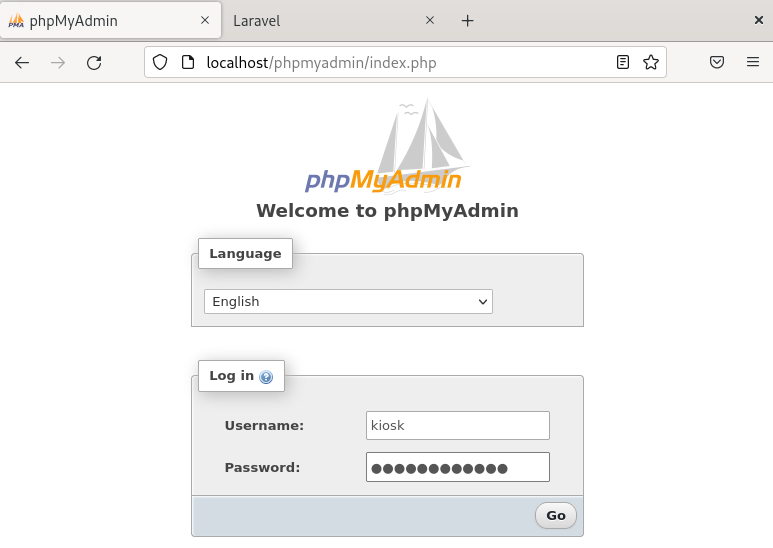
Automatisch gegenereerde beschrijving

Voer de volgende SQL-commando’s uit:

Afbeelding met tekst

Automatisch gegenereerde beschrijving

Probeer vervolgens in te loggen met de aangemaakte gebruiker.

 Afbeelding met tekst

Automatisch gegenereerde beschrijving

#### Creatie van de applicatie

Op dit moment hebben we dus een Linux Machine die een Apache server draaiende heeft met daarop een indexpagina met een standaard reservatie-forum. Deze zullen we later nog aanpassen, maar eerst zullen we ons verdiepen in de Laravel applicatie in diezelfde map.

**Lokaal wijzigen van de applicatie**

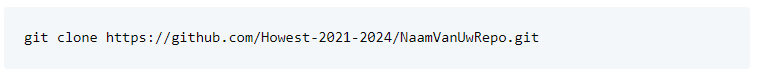
Omdat ons project beschikbaar is op GitHub zullen we deze mogelijkheid benutten om onze applicatie buiten onze VM Machine om te wijzigen. Dit omdat we dan alle tekstbewerkingsprogramma’s en varia kunnen gebruiken waar we gewoon aan zijn.

Er zal telkens duidelijk worden aangegeven wanneer dit **op de VM** of **lokaal** dient te gebeuren.

**Voorbereiden van lokaal project**

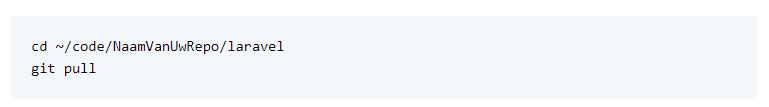
Maak een map op een gewenste plaats op uw lokale computer en zorg ervoor dat **git** beschikbaar is.

Haal vervolgens uw zelfgemaakte repo binnen en open de map in een editor als **Visual Studio Code**:



**Database configuratie**

Zorg ervoor dat u de laatste versie heeft:

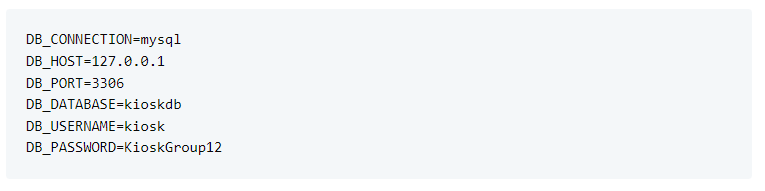


In de root directory van uw project gebruikt u het volgende commando:

Afbeelding met Rechthoek

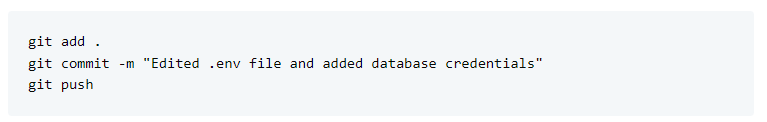
Automatisch gegenereerde beschrijving

Vervolgens wijzigen we de volgende gegevens:



Sla de wijzigingen op met **CTRL+O** en sluit het met **CTRL+X**.

We slaan deze wijziging op.



**Database aanmaken**

Surf naar uw **phpmyadmin** en log in.

Kies vervolgens voor **New** en vul **dezelfde** naam in als de waarde die u DB\_DATABASE gaf in het .env bestand. (De stap hiervoor)

Afbeelding met tekst

Automatisch gegenereerde beschrijving

**Aanmaken van een migratie**

Om tabellen aan de maken in de database zullen we gebruik maken van **migraties**.

Voer de volgende commando’s uit om een migratie aan te maken:

Afbeelding met tekst

Automatisch gegenereerde beschrijving

Als alles goed verloopt krijgt u het volgende bericht te zien;

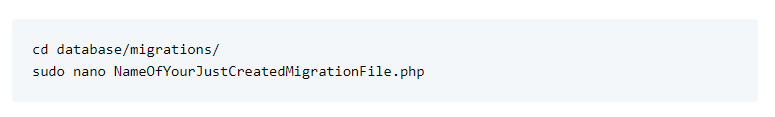


Nu moeten we er nog voor zorgen dat deze migratie ook iets doet.

Kopieer hiervoor de volgende code die voor u beschikbaar werd gemaakt:

[GitHub Howest - Data - Migration File Code](https://github.com/Howest-2021-2024/Data/blob/main/code/laravel/Database/Database-Registrations-Table-Migration.php)

Plaats **deze** **code** **in** het **migratiebestand** die net is aangemaakt:



Sla de wijzigingen op met **CTRL+O** en sluit het met **CTRL+X**.

Afbeelding met tekst

Automatisch gegenereerde beschrijving

Nu voeren we de migratie uit:

Afbeelding met Rechthoek

Automatisch gegenereerde beschrijving

En opnieuw, indien dit succesvol is krijgt u het volgende te zien:

Afbeelding met tekst

Automatisch gegenereerde beschrijving

Surf opnieuw naar uw **phpmyadmin**, log in, u ziet daar uw nieuwe tabel.

Afbeelding met tekst

Automatisch gegenereerde beschrijving

#### Aanmaken van een Registratie Model

Om een beter overzicht te creëren in onze applicatie en om toekomstige CRUD-operaties efficiënt te laten verlopen zullen we gebruikmaken van een model.

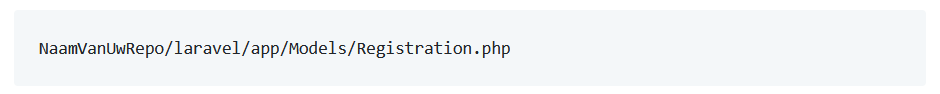
Door het gebruik van Models, worden de database tabellen in de applicatie geabstraheerd, waardoor de code beter georganiseerd en gemakkelijker te onderhouden is. In plaats van SQL-query's te schrijven om gegevens op te halen of te bewerken, kan de ontwikkelaar eenvoudige methoden aanroepen op de Model-klassen om dezelfde taken uit te voeren.

We maken een model aan:

Afbeelding met Rechthoek

Automatisch gegenereerde beschrijving

Als alles goed verloopt kunt u deze terugvinden op de volgende locatie:



We hebben ook deze keer de code al voorzien:

[GitHub Howest - Data - Registration Model Code](https://github.com/Howest-2021-2024/Data/blob/main/code/laravel/Application/Application-Registration-Model-Code.php)

*Kleine toelichting bij deze code:*

De "fillable" -array geeft aan welke velden van het model kunnen worden opgeslagen in de database. In dit geval zijn dat de velden 'first\_name', 'last\_name' en 'email'.

Elk van de velden in de array "fillable" komt overeen met een kolom in de database tabel die bij dit model hoort. Dit betekent dat de code alleen waarden voor deze velden kan opslaan in de database, alle andere velden zullen worden genegeerd.

Nu zullen we deze gebruiken bij het maken van een seeder.

#### Aanmaken van een Seeder

Om onze databank op een juiste manier te voeden met data en ervoor te zorgen dat deze data niet verloren gaat tijdens het onderzoek, maken we gebruik van één of meerdere seeders (afhankelijk van de scope).

Allereerst heeft u de nodige startdata nodig, deze kunt u voorzien door de volgende stappen uit te voeren:

Wij hebben al de code van de seeder voorzien, kopieer deze:

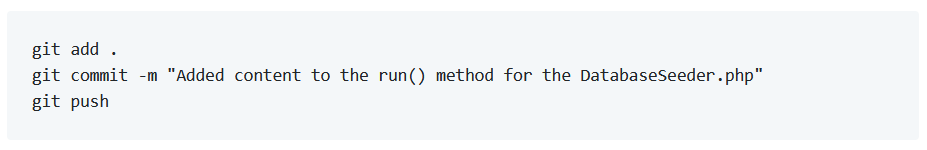
[GitHub Howest - Data - Seeder File Code](https://github.com/Howest-2021-2024/Data/blob/main/code/laravel/Database/Database-Registrations-Seeder-Code.php)

Voer de volgende commando’s uit en plak de code vervolgens in de editor:

Afbeelding met tekst

Automatisch gegenereerde beschrijving

Sla de wijzigingen op met **CTRL+O** en sluit het met **CTRL+X**.



We **testen** de seeder door het volgende uit te voeren:

Afbeelding met tekst

Automatisch gegenereerde beschrijving

Dit zal de migraties ‘vers’ en opnieuw uitvoeren gepaard met het vullen van de tabel(len) met de gegeven data in de seeder.

#### Aanmaken van een Service & Controller

Om de applicatie op te splitsen in kleinere, beheersbare stukken code, maken we gebruik van services en controllers. Dit is geen vereiste maar een ‘good-practice’ om toekomstige complexiteit te vermijden.

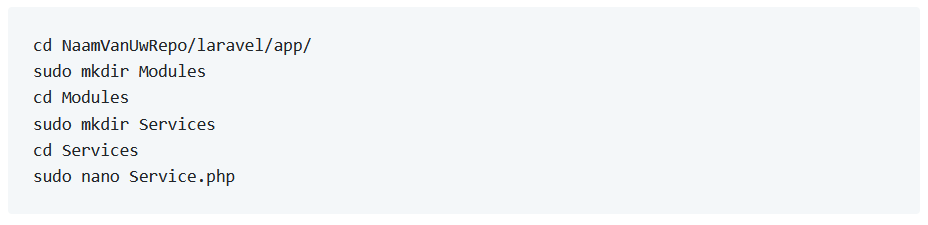
**Service**

Om efficiënt CORS-requests te valideren en uit te voeren zullen we een **basisklasse** maken waar onze **RegistrationService** van af zal stammen. Zo zullen we makkelijk een verzameling validatieregels kunnen meegeven met een **constructor** samen met het **model** die gevalideerd dient te worden (in dit geval **Registration**).

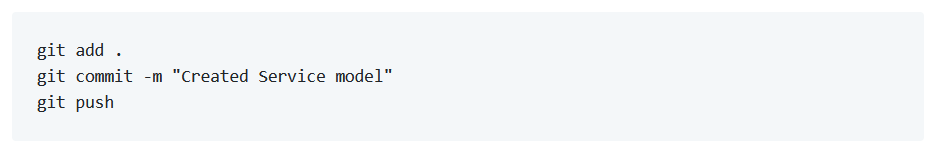
Deze code is al voorzien:

[GitHub Howest - Data - Service File Code](https://github.com/Howest-2021-2024/Data/blob/main/code/laravel/Application/Application-Service-Code.php)

Volg de volgende stappen en plaats de code in het bestand:



Sla de wijzigingen op met **CTRL+O** en sluit het met **CTRL+X**.



**RegistrationService**

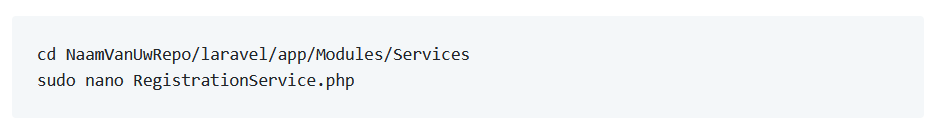
Net lieten we al merken dat er ook nood zal zijn aan een taken-specifieke service, de RegistrationService.

Hier zal de specifieke code komen te staan die de data in de database zal gaan manipuleren/opvragen en terugsturen naar de **controller** die we zo dadelijk zullen aanmaken.

Ook deze code is al voorzien:

[GitHub Howest - Data - RegistrationService File Code](https://github.com/Howest-2021-2024/Data/blob/main/code/laravel/Application/Application-RegistrationService-Code.php)

Volg de volgende stappen en plaats de code in het bestand:

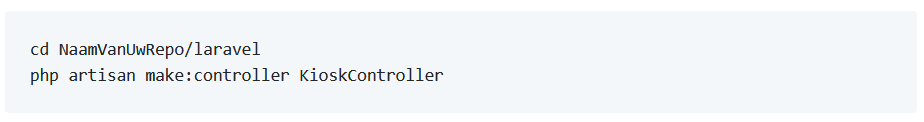


Sla de wijzigingen op met **CTRL+O** en sluit het met **CTRL+X**.

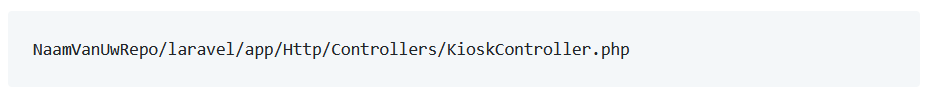
Afbeelding met tekst

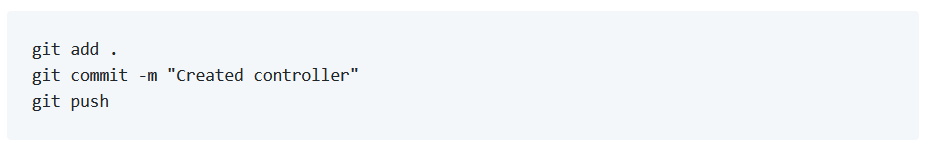
Automatisch gegenereerde beschrijving

**Controller**



Als alles goed ging ziet u deze nu staan in:





Ok, goed! Nu kunnen we beginnen met de uitwerking van de nodige functionaliteiten.

#### Creatie applicatieweergave

Om de kiosk te voorzien van een nuttige functionaliteit, is het belangrijk om een visueel aspect te creëren dat gebruikers kunnen gebruiken. Een applicatieweergave is daarom essentieel. Het biedt de mogelijkheid om informatie te verstrekken over het doel van de kiosk en de functionaliteiten die beschikbaar zijn voor de gebruikers.

Het gebruik van de build-in Laravel views heeft verschillende voordelen. Allereerst zijn de Laravel views zeer flexibel en aanpasbaar. Er zijn verschillende sjablonen beschikbaar die kunnen worden gebruikt om een ​​aantrekkelijke en gebruiksvriendelijke gebruikersinterface te maken. Bovendien zijn de Laravel views zeer goed geoptimaliseerd voor de Laravel-applicatie, wat betekent dat ze naadloos integreren met de Laravel-backend.

**View aanmaken met nodige data**

Navigeer naar uw controller en vervang de code met de volgende:

[GitHub Howest - Data - KioskController File Code](https://github.com/Howest-2021-2024/Data/blob/main/code/laravel/Application/Application-Controller-View.php)

Afbeelding met tekst

Automatisch gegenereerde beschrijving

Sla de wijzigingen op met **CTRL+O** en sluit het met **CTRL+X**.

Dan zullen we nu een pagina voorzien gebruikmakend van bootstrap die 2 delen zal bevatten:

* Weergave van registraties **vandaag**
* Een **registratiegedeelte** waar personen zich kunnen registreren.

Navigeer naar de juiste map, maak het bestand aan en plaats de volgende code er in:

[GitHub Howest - Data - Index Blade File Code](https://github.com/Howest-2021-2024/Data/blob/main/code/laravel/Application/Application-View.blade.php)

Afbeelding met tekst

Automatisch gegenereerde beschrijving

Sla de wijzigingen op met **CTRL+O** en sluit het met **CTRL+X**.

In het bestand staat volgende code:

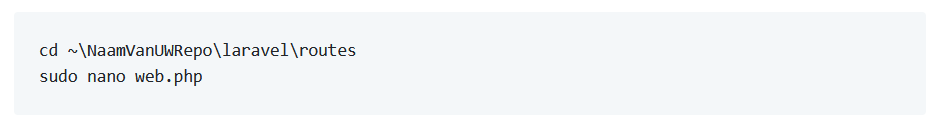
Afbeelding met tekst

Automatisch gegenereerde beschrijving

Deze data is alleen nog niet beschikbaar voor de pagina, hiervoor moeten we nog de juiste verwijzing aan onze laravel applicatie aangeven en de nodige data meegeven.

Volg de volgende stappen en vervang de inhoud van het bestand met de volgende:

[GitHub Howest - Data - Web Route File Code](https://github.com/Howest-2021-2024/Data/blob/main/code/laravel/Application/Application-Routes.php)



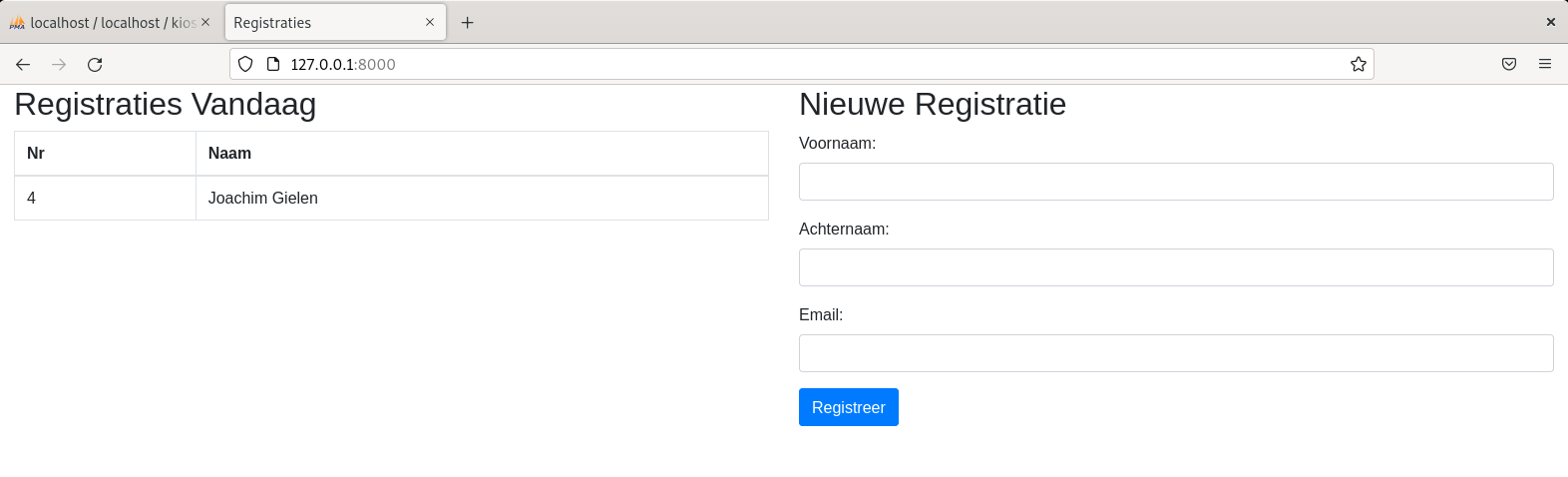
Sla de wijzigingen op met **CTRL+O** en sluit het met **CTRL+X**.

Om de vorige wijzigingen te testen doen we het volgende:

Afbeelding met Rechthoek

Automatisch gegenereerde beschrijving

… en als alles goed is gegaan krijgen we het volgende te zien:



Alleen zal de “registraties vandaag” leeg zijn indien u dit doet, u kunt dit uittesten door via phpmyadmin een nieuwe record toe te voegen.

U voelt het waarschijnlijk al aankomen, dit moet eigenlijk via het “nieuwe registratie” forum, laten we dit uitwerken.

**Registratie functionaliteit**

Allereerst moeten we onze view even aanpassen zodat de juiste actie wordt ondernomen bij het aanklikken van de registratieknop.

Wijzig **enkel** het **form-element** met de nieuwe gegeven code:

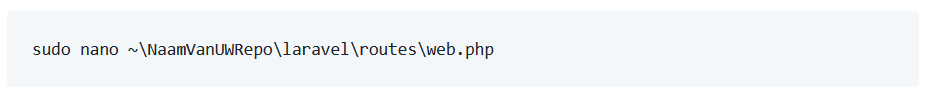
[GitHub Howest - Data - Index Blade File Code v2](https://github.com/Howest-2021-2024/Data/blob/main/code/laravel/Application/Application-View.blade-v2.php)

Afbeelding met tekst

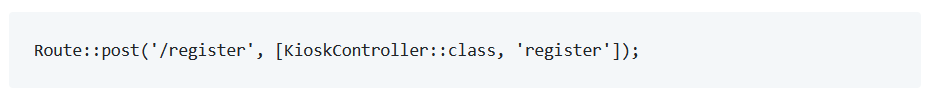
Automatisch gegenereerde beschrijving

Sla de wijzigingen op met **CTRL+O** en sluit het met **CTRL+X**.

We dienen nog enkel een nieuwe route aan te maken zodat de **actie** van het forum **/register** ook wordt opgenomen.



**Voeg de volgende route toe:**



Sla de wijzigingen op met **CTRL+O** en sluit het met **CTRL+X**.

Om de functionaliteit te testen, maken we in de **KioskController** een extra methode aan:



Afbeelding met tekst

Automatisch gegenereerde beschrijving

Hiermee kunnen we de inhoud van de request analyseren.

Sla de wijzigingen op met **CTRL+O** en sluit het met **CTRL+X**.

We testen dit:

Afbeelding met Rechthoek

Automatisch gegenereerde beschrijving

Vul het forum in met correcte data en klik op de knop.

Als alles goed ging, ziet u iets als het volgende, maar dan met variabele data:

Afbeelding met Rechthoek

Automatisch gegenereerde beschrijving

**We slaan alle wijzigingen op!**

Afbeelding met tekst

Automatisch gegenereerde beschrijving

#### Registratie Implementatie

Eerst en vooral gaan we aan de slag met de data uit de vorige stap.

Laten we hiervoor de KioskController inhoud vervangen met de volgende:

[GitHub Howest - Data - Final KioskController File Code](https://github.com/Howest-2021-2024/Data/blob/main/code/laravel/Application/Application-KioskController-Final-Code.php)

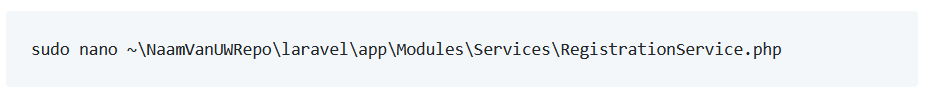


Sla de wijzigingen op met **CTRL+O** en sluit het met **CTRL+X**.

Om de systeemlagen te respecteren zullen we gebruikmaken van onze eerder gecreëerde **RegistrationService**.

Vervang de inhoud met de volgende code:

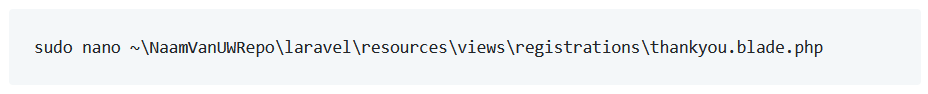
[GitHub Howest - Data - Final RegistrationService File Code](https://github.com/Howest-2021-2024/Data/blob/main/code/laravel/Application/Application-RegistrationService-Final-Code.php)



Sla de wijzigingen op met **CTRL+O** en sluit het met **CTRL+X**.

Voor gebruikersvriendelijkheid is er ook een registratiebevestiging view toegevoegd, deze kun je hier vinden:

[GitHub Howest - Data - ThankYou Blade File Code](https://github.com/Howest-2021-2024/Data/blob/main/code/laravel/Application/Application-ThankYou-View.php)



Sla de wijzigingen op met **CTRL+O** en sluit het met **CTRL+X**.

**Finale Test**

Bekijk de volgende video en voer dezelfde stappen uit, als alles goed is gegaan bekomt u hetzelfde resultaat.

[Youtube - Demonstratie Applicatie Test](https://youtu.be/L_qBDonuPZ4)

#### Tussenevaluatie

Onze Laravel applicatie is klaar!

Gebruikers kunnen zich zonder problemen registreren en de nodige activiteiten worden op de achtergrond uitgevoerd.

Nu is het tijd om deze applicatie ook beschikbaar te maken voor onze virtuele machine die kioskmodus draait!

Vervolgens zullen we ervoor zorgen dat er een ELK-systeem gekoppeld wordt aan de kioskactiviteiten.

### ELK

Alle activiteiten die plaatsvinden op de Windows Clients in de kioskmodus worden geregistreerd en gelogd in de ELK-machine.

Dit omvat logbestanden van verschillende bronnen, zoals het gebruik van applicaties, systeemfouten en gebruikersinteracties.

De ELK-stack maakt het mogelijk om deze gegevens te analyseren en te visualiseren, waardoor we snel problemen en fouten kunnen identificeren en oplossen.

Door het opnemen van deze logging functionaliteit in de opzet van het netwerk, kunnen we een betrouwbare en beveiligde omgeving creëren voor de gebruikers.

## ONDERZOEK

Veel mensen zouden denken om gewoon de laravel “artisan sail up” functie te gebruiken ook om het in productie te brengen, nu zal er ingegaan worden op de standpunten waarom dit geen goed idee is.

De nummer één reden waarom is schaalbaarheid en prestaties. Wanneer u weet dat er veel mensen uw website gaan bezoeken dan moet u ook denken aan de performantie van de servers. Apache heeft op dit vlak een stuk meer interessante features zoals load balancing, caching en meer in tegenstelling tot laravel hosting

De configuratiemogelijkheden van een Apache server zijn een goede reden om geen laravel hosting te gebruiken. In vergelijking met laravel geeft Apache alle controle over aan de gebruiker. Het is handig voor configuraties zoals SSL-certificaten, toegangscontrole, enz.

Apache heeft een goede integratie met andere technologieën, waardoor het tijd bespaard. Door de manier hoe Apache naadloos kan geïntegreerd worden met andere tools en technologieën die mogelijk gebruikt kunnen worden in de applicatiestack. Zo kunnen complexere architecturen vereenvoudigd worden.

Daarnaast is Apache ook een industriestandaard, interessant voor grote hoeveelheid aan documentatie. Voor elke tool of configuratie bestaat, hoogstwaarschijnlijk, een online stappenplan.

Al deze voordelen ter zijde, de redenen waarom we nog steeds artisan gebruiken om de applicatie uit te voeren zijn als volgt.

* Onze app heeft weinig load dus er is geen performantie probleem.
* Het gaat niet zoveel meer groeien sinds het enkel op het lokaal netwerk word gebruikt.
* Ook is en gaat er niet veel nood zijn aan features waardoor we het simpel kunnen houden voor dingen zoals database sinds dat we gewoon de built in laravel database kunnen gebruiken.

## HULPMIDDELEN

# CONCLUSIE