Project groep 20

Gemaakt door:

Levi Staal  
Joost Bakker  
Mike Wierenga  
Kevin Houkema

Socialmaps

Technisch ontwerp

Table of Contents

[Inleiding 3](#_Toc12970311)

[1.1 Achtergrond 3](#_Toc12970312)

[1.2 Standaarden en richtlijnen 3](#_Toc12970313)

[1.2.2 Hardware 3](#_Toc12970314)

[2. Systeem Architectuur 4](#_Toc12970315)

[2.1 Hardware 4](#_Toc12970316)

[2.2 Software 4](#_Toc12970317)

[2.3 Programmeertalen 4](#_Toc12970318)

[2.4 Gegevensopslag 5](#_Toc12970319)

[3. Logische architectuur 6](#_Toc12970320)

[3.1 Architectuur 6](#_Toc12970321)

[3.2 Usecase 7](#_Toc12970322)

[3.2.1 Login Usecase 7](#_Toc12970323)

[3.2.2 Website/mobiele app Usecase 8](#_Toc12970324)

[4. Schaalbaarheid 9](#_Toc12970325)

[4.1 Toevoeg mogelijkheden 9](#_Toc12970326)

# Inleiding

Dit is het technisch ontwerp voor het project SocialMaps. Dit is een applicatie waarin mensen op een sociale manier berichten voor elkaar achter kunnen laten op een bepaalde locatie. Hierna kunnen andere mensen hier naartoe gaan om het bericht te bekijken. In dit document wordt beschreven welke technieken, hard-/software en ontwerpkeuzes er gemaakt zijn.

## 1.1 Achtergrond

Voor het project van het blok 4 van jaar 2 is er besloten dat projectgroepen zelf met een origineel idee moeten komen en dit vervolgens realiseren. Het idee om een sociaal platform te creëren waarbij mensen actief naar berichten moeten lopen om deze te kunnen bekijken was volgens de docent een zeer creatief idee. Hierdoor is er besloten dit product te realiseren.

## 1.2 Standaarden en richtlijnen

Op basis van gemaakte afspraken zijn er standaarden vastgesteld bij het gebruik van de hard-/software.

1.2.1 software

- Alle code moet op een nette manier geschreven worden.

- Alle zelf gemaakte functies of logica moeten comments bevatten.

- Versiebeheer moet bijgehouden worden

- Al het verkeer naar de database moet via de backend server gerealiseerd worden.

- Er moet gebruik gemaakt worden van GitHub om applicatie delen bij elkaar te voegen.

## 1.2.2 Hardware

- Iedereen maakt gebruik van zijn eigen laptop/computer bij het realiseren van het project

# 2. Systeem Architectuur

In dit hoofdstuk wordt duidelijk gemaakt welke hardware en software gebruikt wordt in dit project. Ook wordt hier de structuur uitgelegd. Dit geeft een goed beeld van hoe de applicatie in elkaar steekt.

## 2.1 Hardware

Voor dit project is de volgende hardware gebruikt:

* Eigen laptops
* Externe NodeJS-server
* Android telefoon

## 2.2 Software

Voor dit project is de volgende software gebruikt:

* Android Studio
* SublimeText (of andere editors)
* Android mobiel v6+
* MongoDB Atlas

## 2.3 Programmeertalen

In dit project zijn de volgende programmeertalen en frameworks gebruikt:

* Angular CLI

Er is gekozen voor dit framework omdat een asynchronisch workflow heeft. Hierdoor is de prestatie van de applicatie hoger ondanks dat het een grote hoeveelheid data moet laden.

* JAVA

Er is gekozen om JAVA te gebruiken voor de mobile applicatie omdat dit de manier is hoe Android Studio applicaties maakt. JAVA is een stabiele programmeertaal met veel mogelijkheden. Ook zijn wij allemaal bekend met JAVA.

* NodeJS

NodeJS is gekozen omdat het de mogelijkheid heeft gemakkelijk data te converteren naar JSON en te verzenden naar de MongoDB database.

## 2.4 Gegevensopslag

Voor dit project is er gebruik gemaakt van de volgende opslagmethodes:

* GitHub

Github geeft de mogelijkheid om in meerdere branches aan een project te werken. Het is een gemakkelijke oplossing om conflicten bij project versies tegen te gaan.

* MongoDB

Er is gekozen voor MongoDB als database omdat het een document based is. Dit betekend dat we berichten kunnen plaatsen ongeacht van hoe groot ze zijn of hoeveel informatie ze bevatten. Ook is dit een simpele oplossing voor het bijhouden van iedereen die een gebruiker volgt, omdat het maar 1 array is die gevuld moet worden.

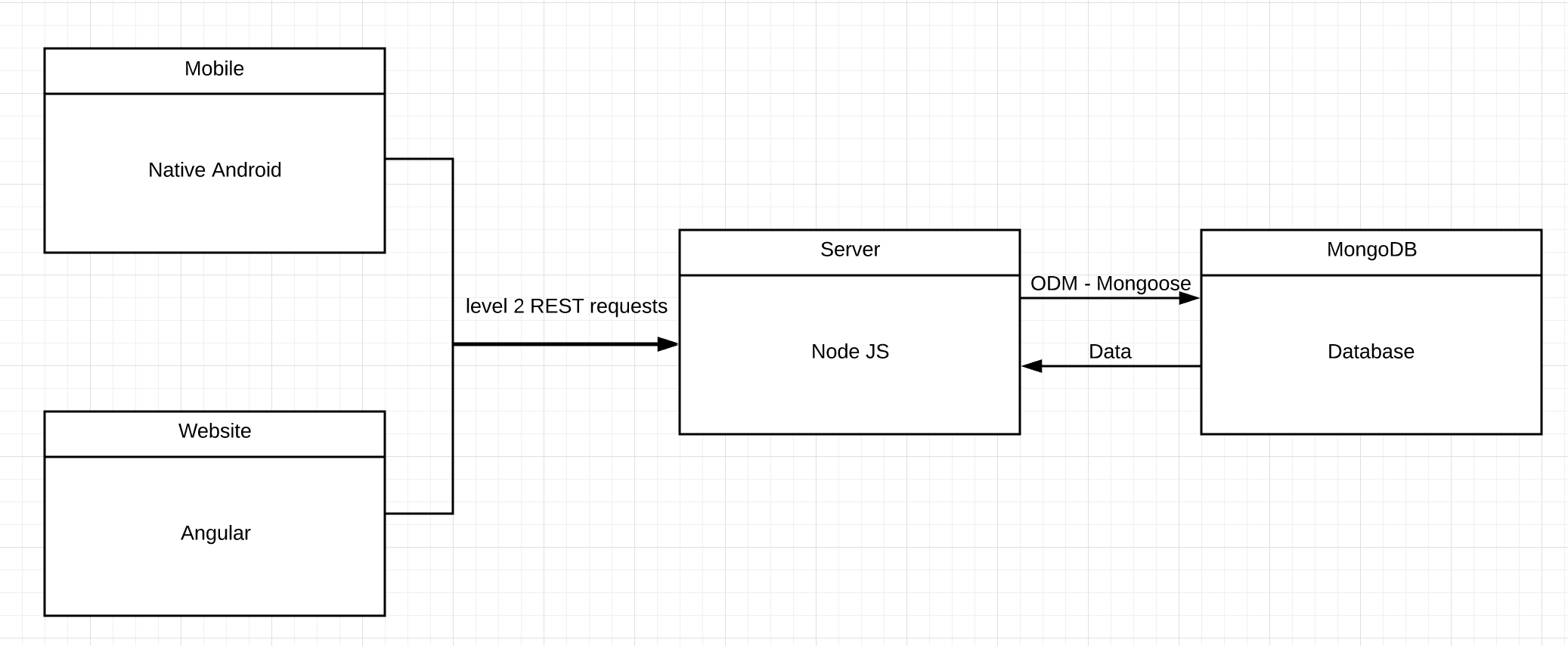
* Openode

Omdat de NodeJS server bereikbaar moest zijn vanaf verschillende apparaten is er gekozen om de server op een online omgeving te plaatsen. Natuurlijk waren hier veel verschillende mogelijkheden voor, maar Openode geeft de mogelijkheid om gemakkelijk in te kunnen zien wat er fout en wat er goed gaat door de online bijgehouden logs. De verbinding met Openode was ook makkelijk om op te zetten.

# 3. Logische architectuur

Hier volgt de logische architectuur voor dit project. Dit hoofdstuk legt uit wat de dataflow is en hoe alle onderdelen met elkaar communiceren.

## 3.1 Architectuur



Afbeelding 3.1.1

In de afbeelding hierboven (afbeelding 3.1.1) is te zien hoe de componenten van de website en de mobiele applicatie met de server en database communiceren.

Zowel de website als de mobiele applicatie maken gebruik van de level 2 RESTful API. Een RESTful api maakt gebruik van HTTP requests om data te ophalen, invoegen, veranderen en verwijderen (GET, PUT, POST, DELETE). Wanneer de website of mobiele applicatie gebruik willen maken van de data die in het MongoDB database staat, moet dit altijd via de NodeJS server gaan.

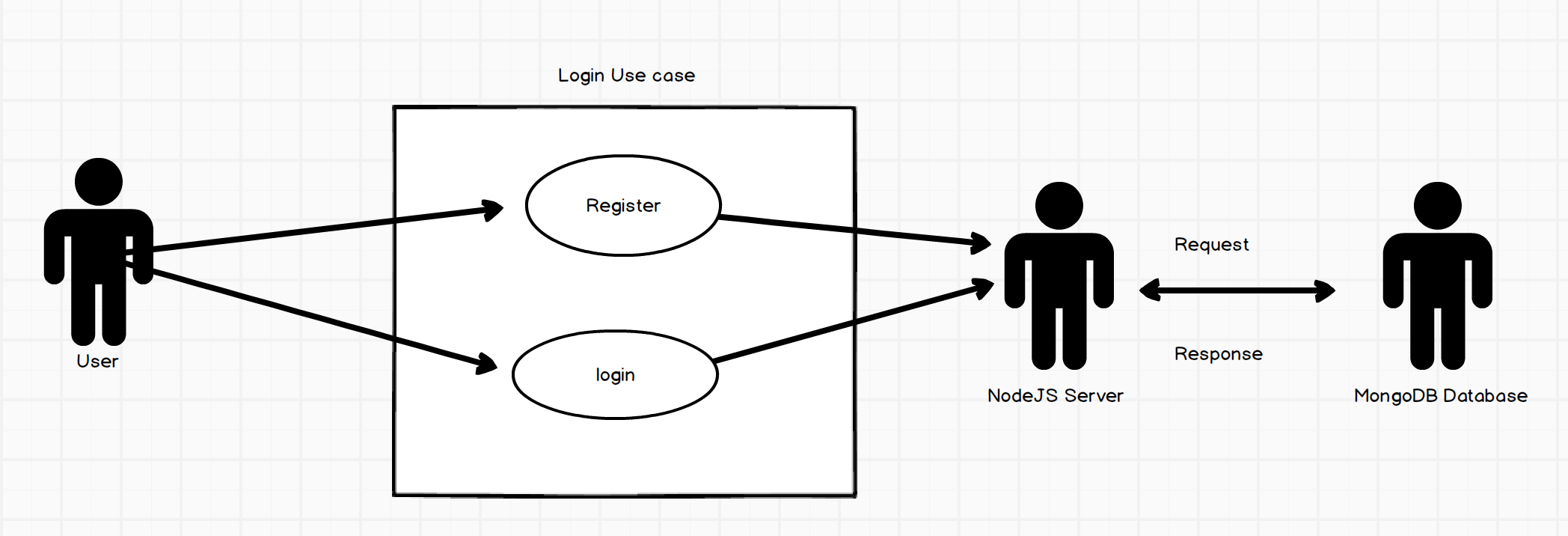
De NodeJS server zal de requests opvangen en deze via het Mongoose ODM (Object Document Mapping) model verwerken en verzenden naar MongoDB. Het Mongoose ODM-model converteert de data die het binnen krijgt en zet het om naar een JSON-formaat die verwerkbaar is in het MongoDB database.

De MongoDB database zal de aanvragen verwerken en de juiste reactie terugsturen naar de NodeJS server. De server zal dit uiteindelijk terug sturen naar het platform die het request heeft gedaan.

## 3.2 Usecase

Dit hoofdstuk beschrijft de usecases die gebruikt zijn bij het realiseren van dit project.

### 3.2.1 Login Usecase



Afbeelding 3.1

Dit is de usecase voor het inloggen van een gebruiker. Als een gebruiker op de mobiele applicatie of de website komt heeft hij 2 opties: inloggen of registreren. Beide formulieren sturen een post request naar de nodeJS server die vervolgens de data converteert en naar de database stuurt.

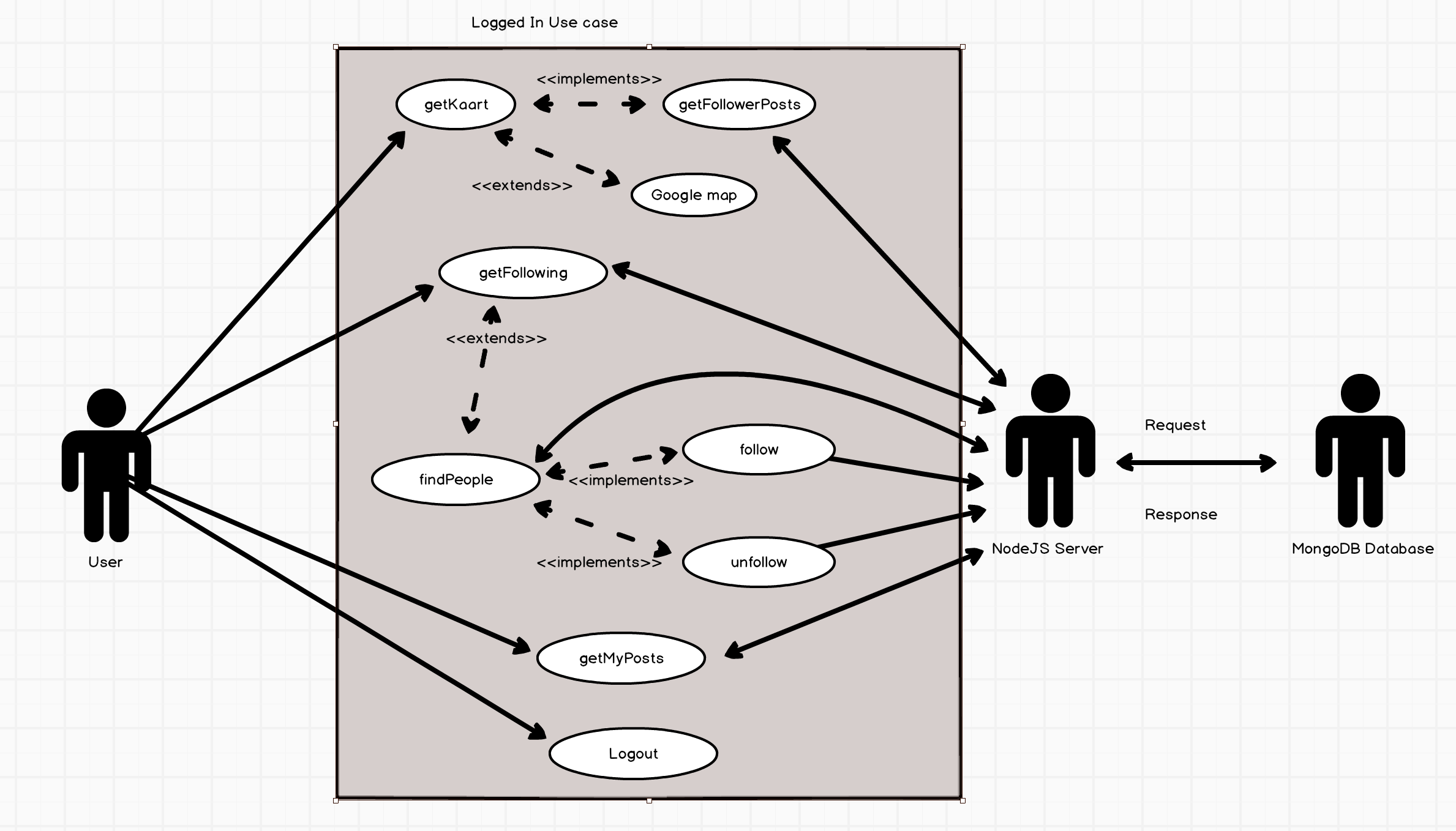
**Register**

Als een user registratie verstuurd wordt, wordt er eerst gekeken of het emailadres al bestaat en vervolgens of de username al bestaat. Als deze beide nog niet in de database bestaan wordt uiteindelijk een 'ok' teruggestuurd.

**Login**

Als een user inlogt met zijn of haar email en wachtwoord zal de database aangeven of er een gebruiker bestaat in de database met deze inloggegevens en deze vervolgens terug sturen. Als er geen gebruiker aanwezig is zal de server hier een foutmelding voor geven.

### 3.2.2 Website/mobiele app Usecase



Afbeelding 3.2

Hierboven (afbeelding 3.2) staat de usecase voor wanneer iemand ingelogd op de website/mobiele applicatie aankomt. Wanneer de gebruiker de website benaderd heeft komen de volgende mogelijkheden beschikbaar: Kaart, volgend, mijn berichten, en loguit.

Als de gebruiker naar 'kaart' navigeert zal er een Google maps kaart geladen worden. Vervolgens zal er een request naar de database gaan die de berichten van de mensen die de gebruiker volgt terugstuurt. Deze berichten zullen hierna te zien zijn links van de kaart als klikbare hyperlinks, en in de kaart als marker.

Als de gebruiker naar 'volgend' navigeert zal er een lijst komen met alle mensen die hij op dat moment volgt. Hier krijgt hij de optie mensen uit zijn lijst te halen, of nieuwe mensen te zoeken om te volgen.

Als de gebruiker naar 'mijn berichten' navigeert zal hij een lijst krijgen met alle berichten die hij heeft geplaatst.

Als de gebruiker op de 'loguit' knop drukt, zal de sessie afsluiten en zal de gebruiker uitloggen.

De mobiele versie werkt hetzelfde als de website, maar heeft een extra functionaliteit. Het plaatsen van berichten.

# 4. Schaalbaarheid

Een van de vereisten van dit project was dat er makkelijke mogelijkheden moest zijn voor uitbereidingen binnen de applicatie. Omdat de applicatie in moderne frameworks en met moderne technieken zijn gerealiseerd, zullen uitbereidingen makkelijk toe te voegen zijn.

## 4.1 Toevoeg mogelijkheden

De volgende ideeën zouden als toevoeging gezien kunnen worden voor de applicatie SocialMaps:

* Foto's

Het is momenteel alleen mogelijk om tekst berichten te sturen. Dit komt door het gebrek aan resources en opslagmogelijkheden. Als een betere online omgeving die meer dataopslag ondersteund gebruikt zal worden zou dit gemakkelijk toegevoegd kunnen worden. Voor dit project is hier niet voor gekozen omdat dat maandelijks geld zou gaan kosten.

* Videofragmenten

Dit is hetzelfde idee als bij de foto's. Ook hier is de opslagruimte niet beschikbaar voor momenteel.

* Groepen

Een leuke toevoeging zou de mogelijkheid zijn om groepen gebruikers in één keer te volgen. Op deze manier zal er een gemakkelijke mogelijkheid zijn om voor het eerst de app te betreden en te gebruiken. Dit zou ook de mogelijkheid geven om evenementen te organiseren op basis van locatie.