

# Alexander Schatteman (202182201)

- ☒ Front-end Web Development
  - [GitHub repository](#)
  - [Online versie](#)
- ☒ Web Services: GITHUB URL
  - [GitHub repository](#)
  - [Online versie](#)

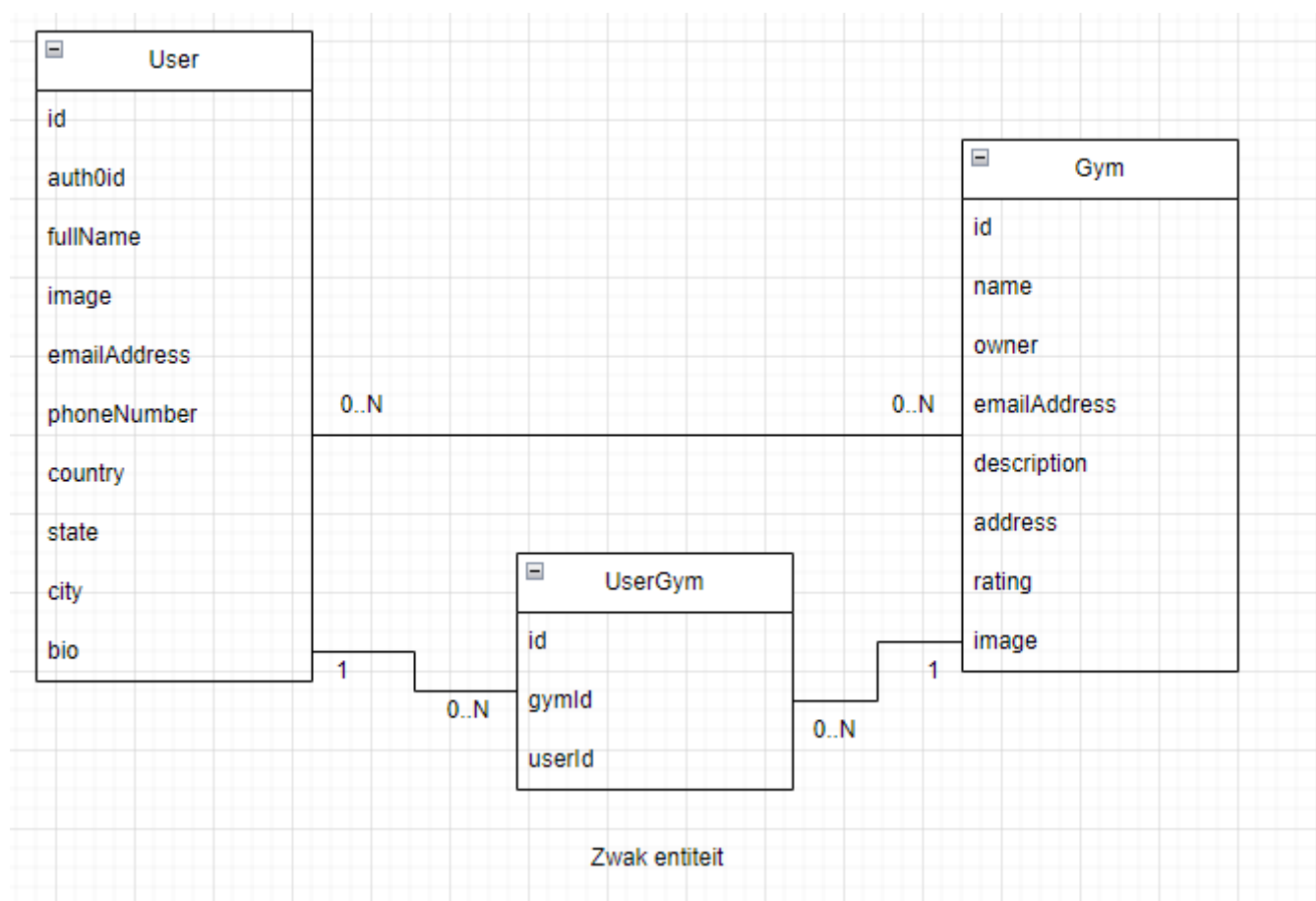
## Logingegevens

- Gebruikersnaam/e-mailadres: e2e-testing@gymderapp.be
- Wachtwoord: TestingUser123456

U kunt ook inloggen met Facebook, discord & google -> developer keys zijn allemaal in orde

## Projectbeschrijving

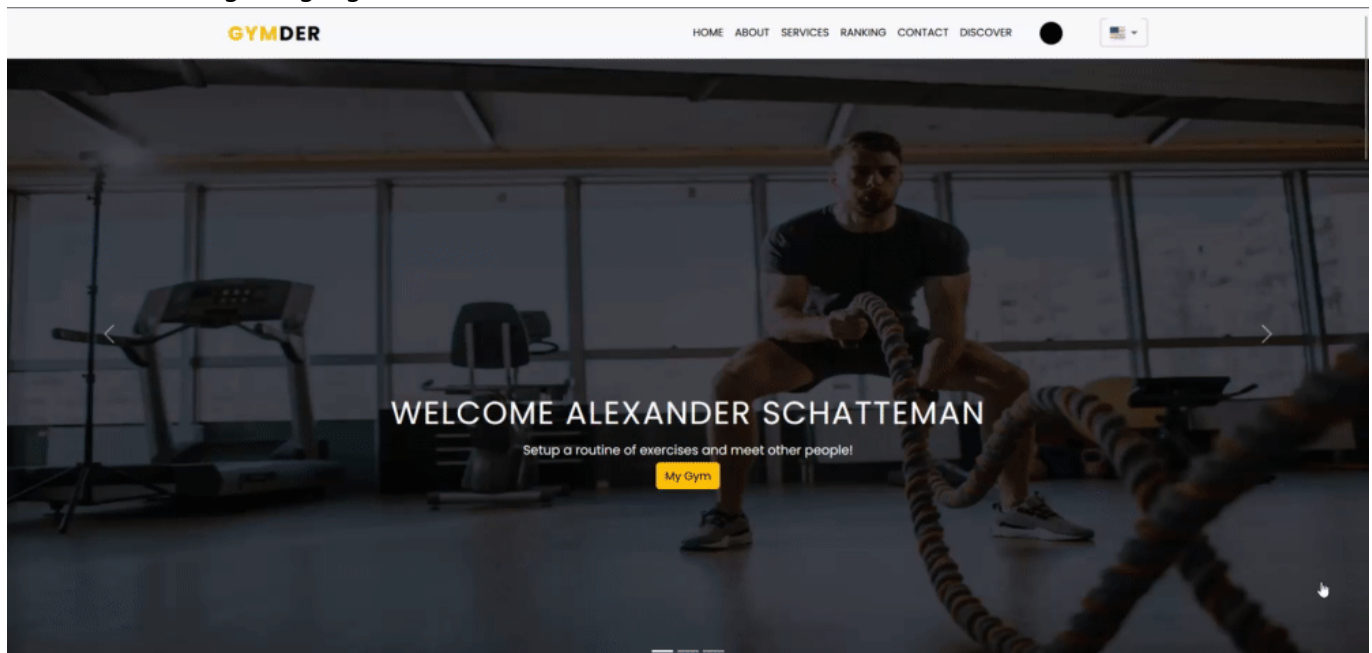
Het idee bij het project was een applicatie waar mensen die veel fitnessen gyms op kunnen toevoegen. Deze mensen zouden ook uiteindelijk de optie hebben om een gym toe te treden. Je zou ook gemakkelijk andere mensen kunnen vinden die een account hebben gemaakt op de site en deze dus vinden. Hieronder vindt u een EERD van het project.



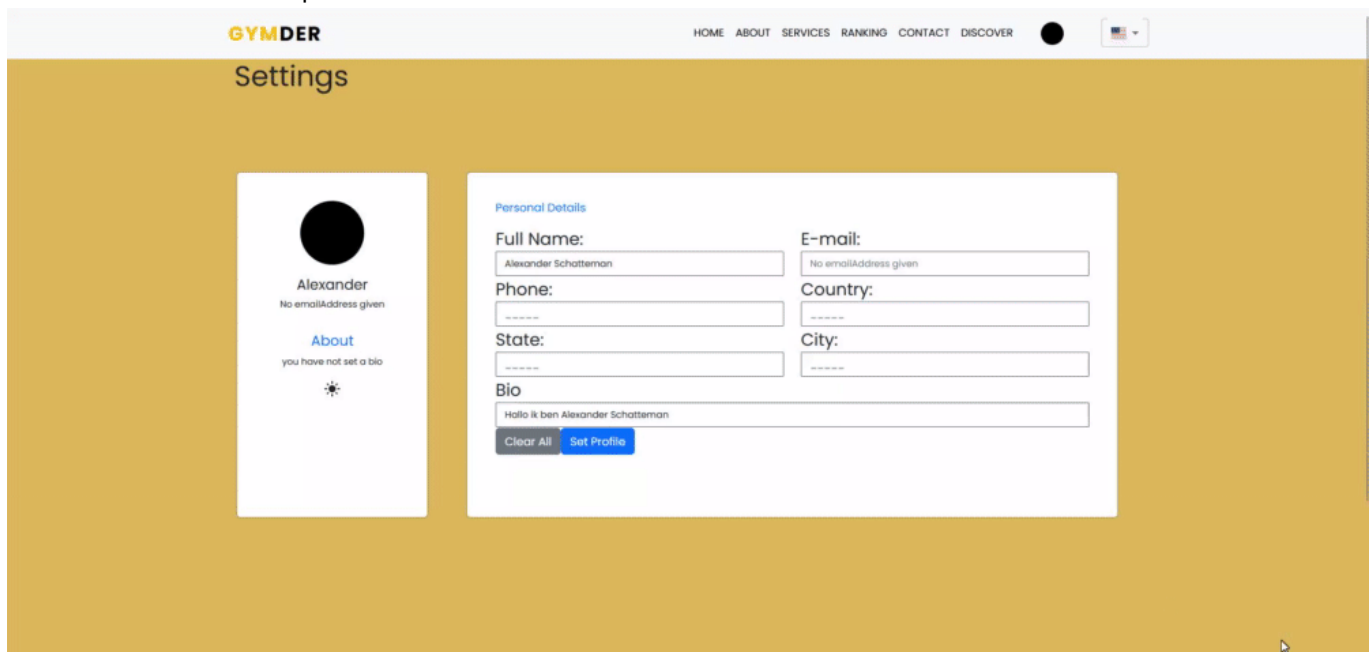
## Screenshots

Bekijk hier de repo markdown om de gifs te laten werken!

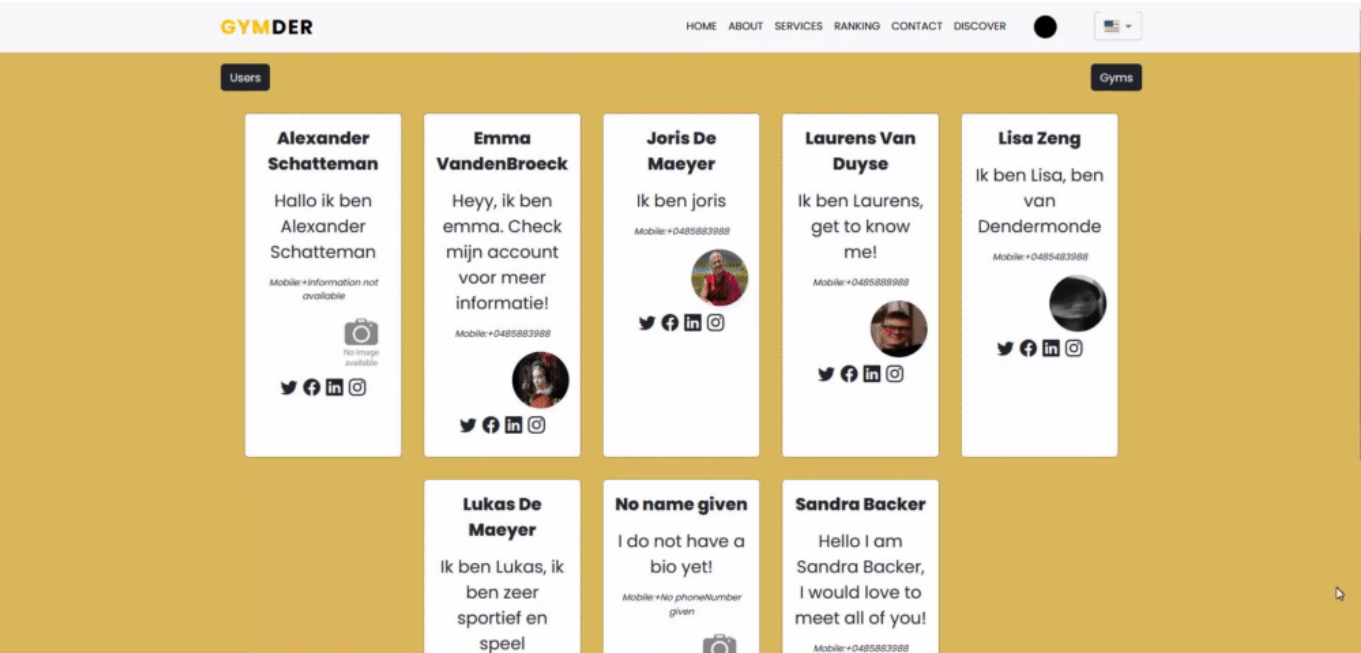
How we can change languages:



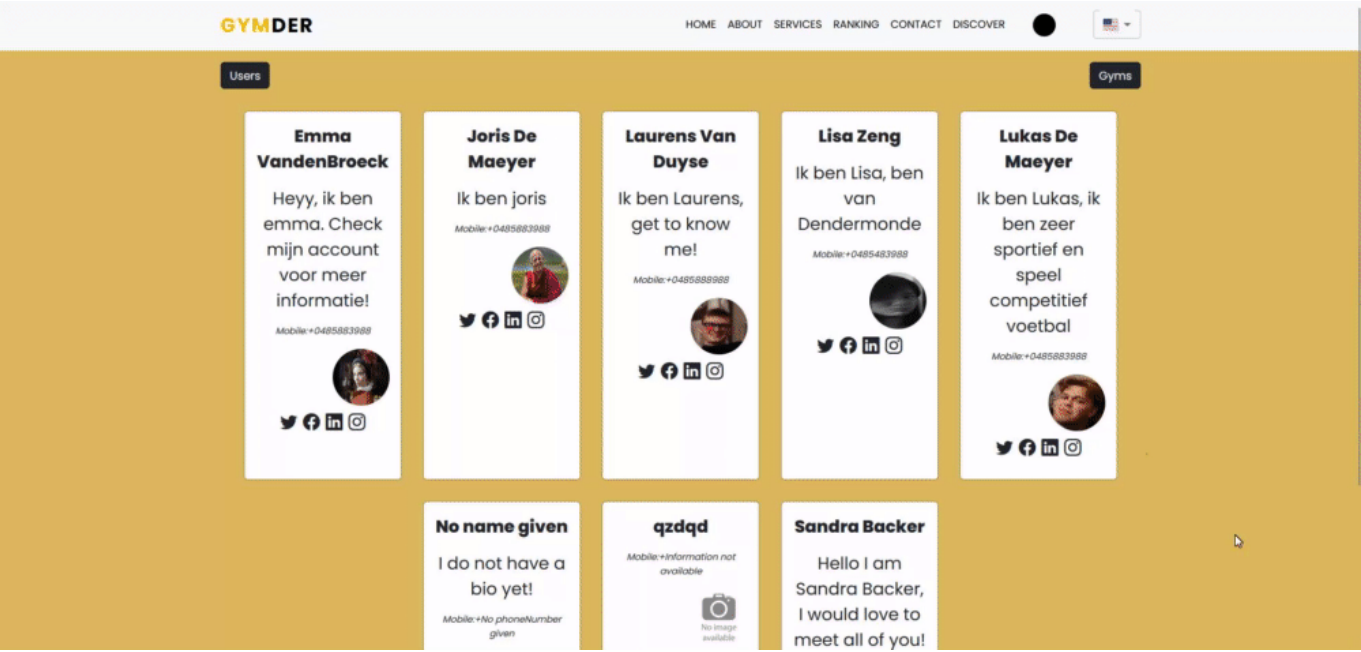
How we can edit a user profile:



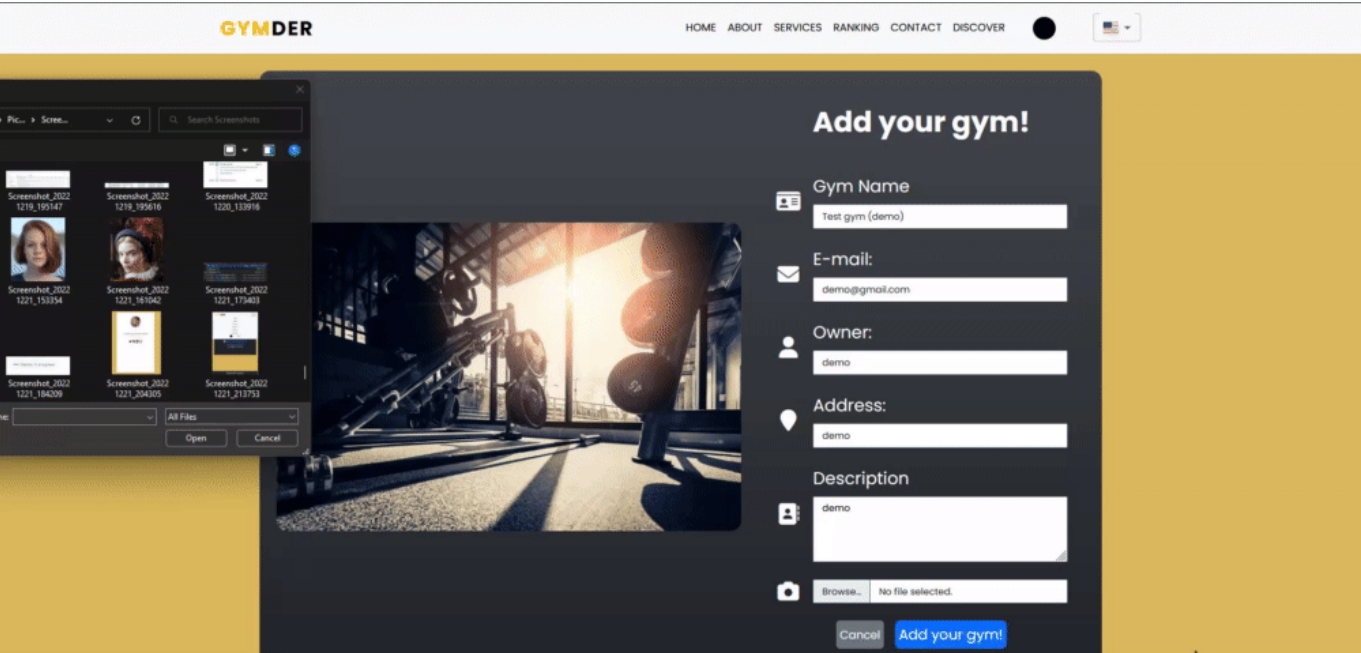
Dark mode / userProfile:



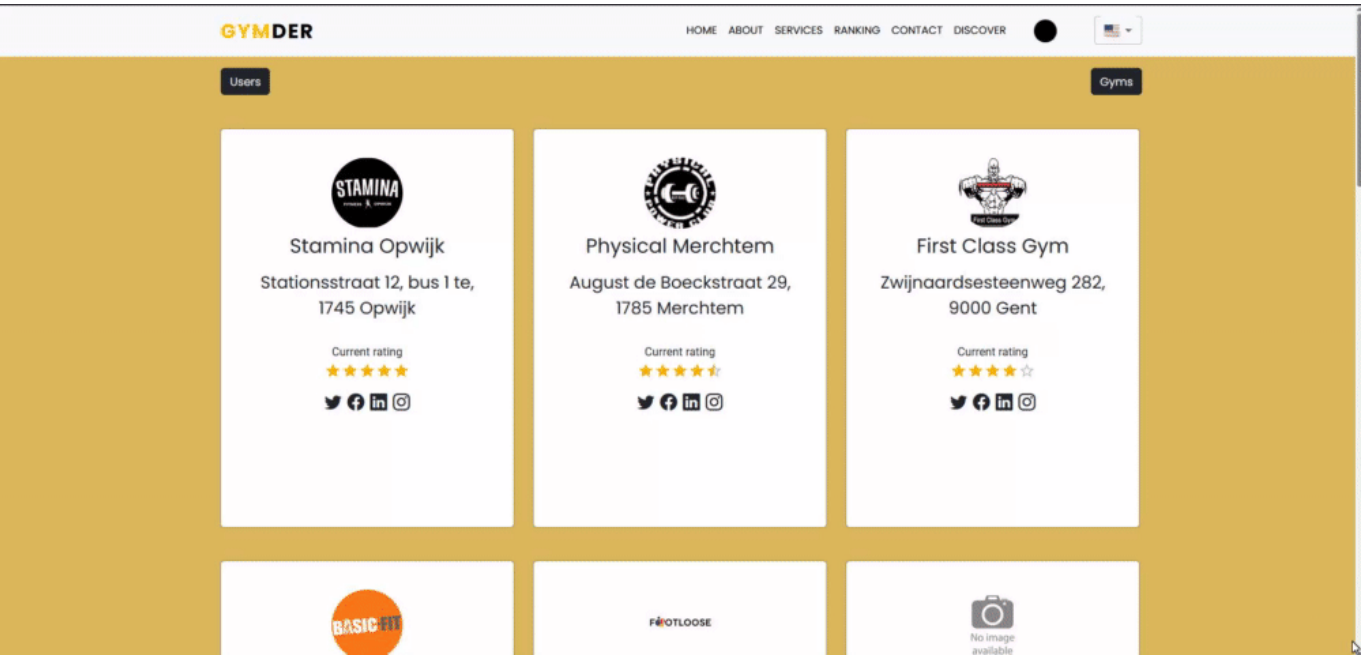
How we can remove a gym and add one from a user's profile (with snackbars):



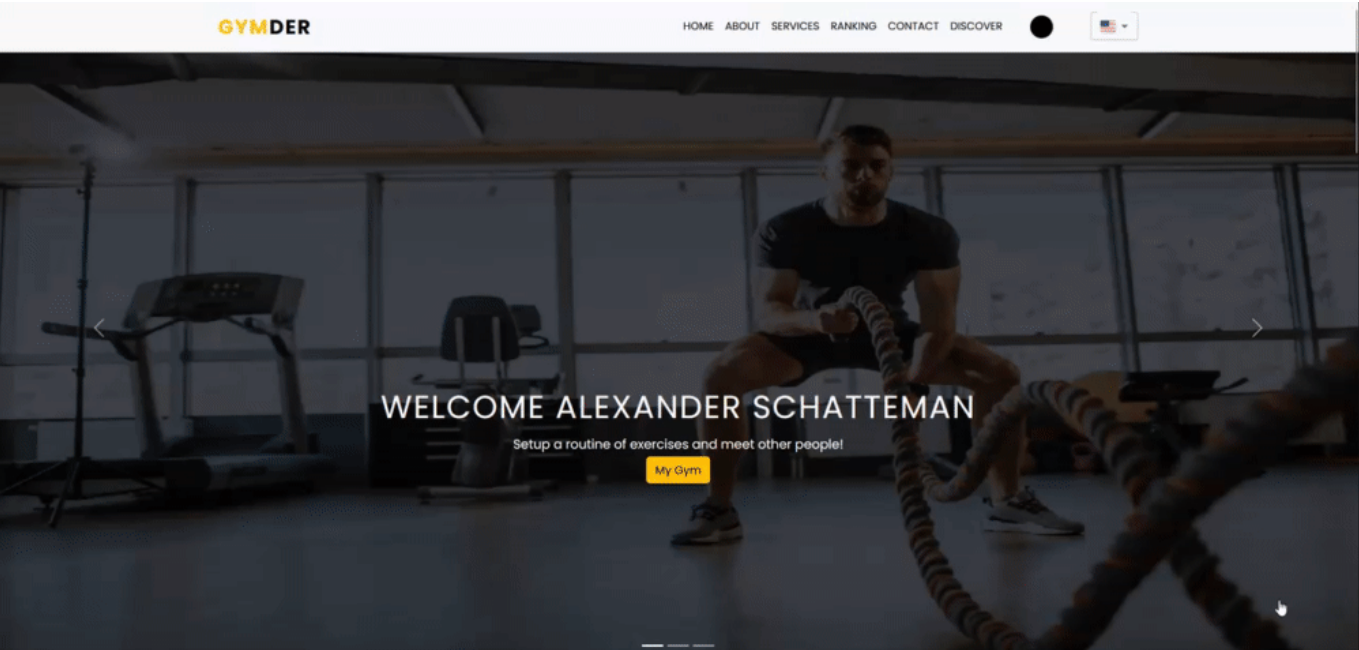
How we can add a gym with images to website:



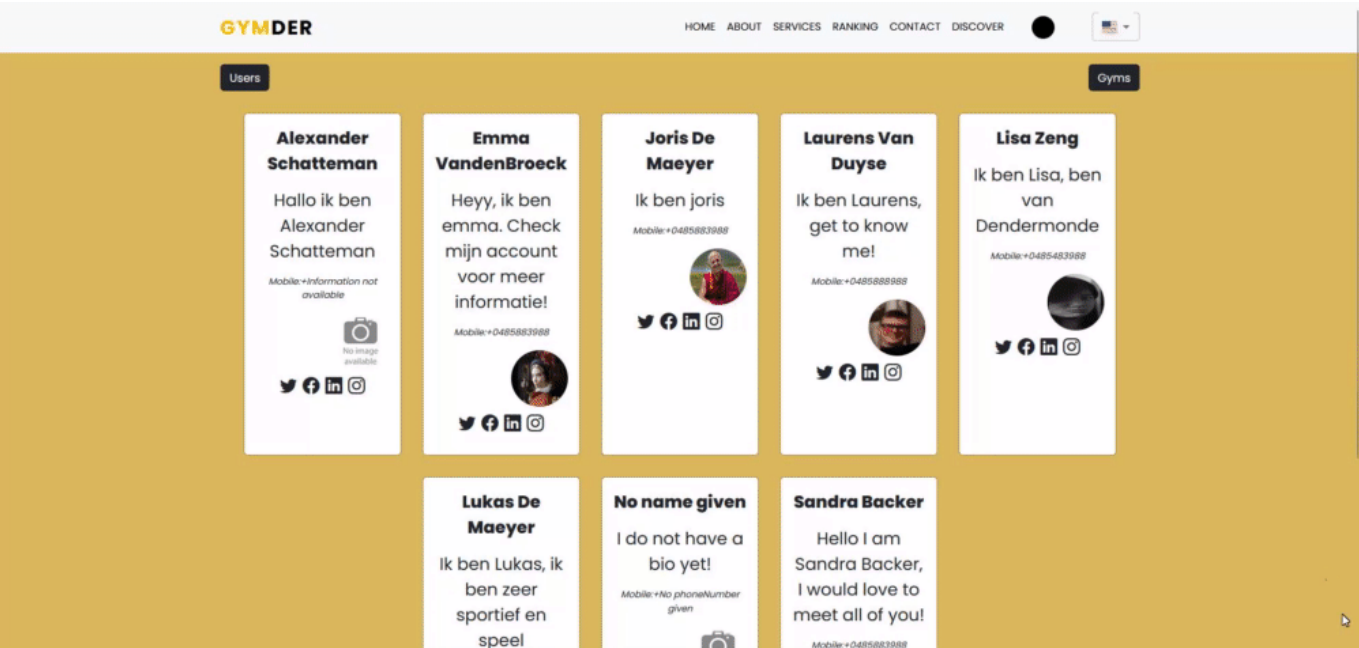
Add the created gym to user:



How we can check users and details of a user:

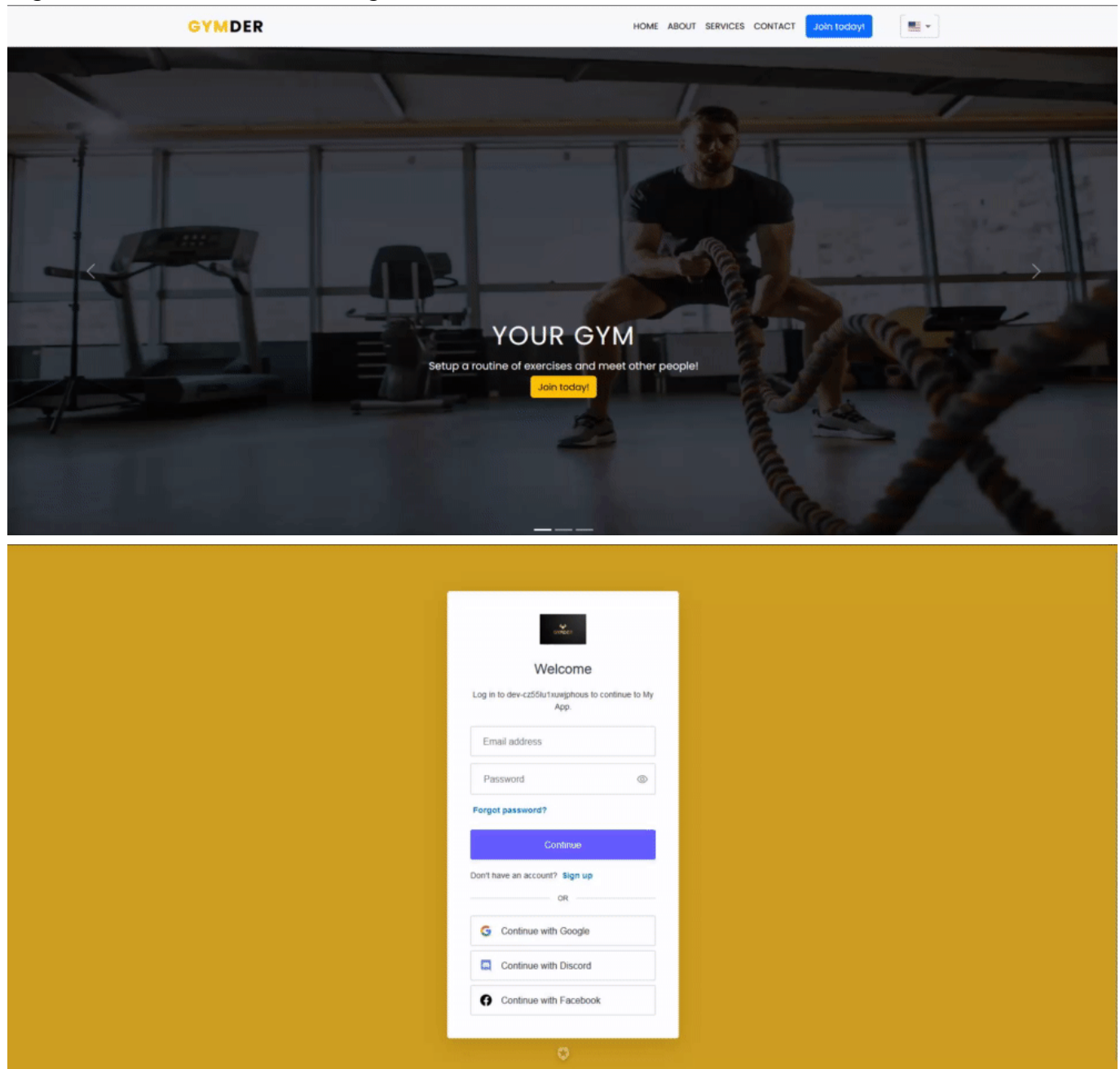


How we can check the gym details:





Login & is not authenticated warning



## Behaalde minimumvereisten

### Front-end Web Development

- **componenten**

- ☒ heeft meerdere componenten - dom & slim (naast login/register)
- ☒ definieert constanten (variabelen, functies en componenten) buiten de component
- ☒ minstens één form met validatie (naast login/register)
- ☒ login systeem (eigen of extern zoals bv. Auth0)

- **routing**

- ☒ heeft minstens 2 pagina's (naast login/register)
- ☒ routes worden afgeschermd met authenticatie en autorisatie

- **state-management**

- ☒ meerdere API calls (naast login/register)
- ☒ degelijke foutmeldingen indien API call faalt
- ☒ gebruikt useState enkel voor lokale state
- ☒ gebruikt Context, useReducer, Redux... voor globale state

- **hooks**

- ☒ kent het verschil tussen de hooks (useCallback, useEffect...)
- ☒ gebruikt de hooks op de juiste manier

- **varia**

- ☒ een aantal niet-triviale testen (unit en/of e2e en/of ui)
- ☒ minstens één extra technologie
- ☒ duidelijke en volledige README.md
- ☒ volledig en tijdig ingediend dossier

## Web Services

- **data laag**

- ☒ voldoende complex (meer dan één tabel)
- ☒ één module beheert de connectie + connectie wordt gesloten bij sluiten server
- ☒ heeft migraties
- ☒ heeft seeds

- **repository laag**

- ☒ definieert één repository per entiteit (niet voor tussentabellen) - indien van toepassing
- ☒ mapt OO-rijke data naar relationele tabellen en vice versa

- **servicelaag met een zekere complexiteit**


- ☒ bevat alle domeinlogica
- ☒ bevat geen SQL-queries of databank-gerelateerde code

- **REST-laag**

- ☒ meerdere routes met invoervalidatie
- ☒ degelijke foutboodschappen
- ☒ volgt de conventies van een RESTful API
- ☒ bevat geen domeinlogica
- ☒ degelijke autorisatie/authenticatie op alle routes

- **varia**

- ☒ een aantal niet-triviale testen (min. 1 controller >=80% coverage)
- ☒ minstens één extra technologie
- ☒ duidelijke en volledige **README.md**
- ☒ maakt gebruik van de laatste ES6-features (object destructuring, spread operator...)

-  volledig en tijdig ingediend dossier

## Projectstructuur

### Front-end Web Development

Ik heb een hoofdmap "src" waarin ik verschillende mappen heb georganiseerd voor verschillende doeleinden. Zo heb ik bijvoorbeeld "components" map waarbinnen ik visuele componenten bewaar zoals forms, cards, knoppen... Ik heb ook een pages folder waarbinnen ik de logische componenten van mijn applicatie bewaar, zoals de componenten die de state beheren en de componenten die de gegevens ophalen van de externe API.

Voor het ontwerp van mijn componenten heb ik gebruik gemaakt van het "presentational and container components" patroon, wat betekent dat ik visuele componenten heb die alleen verantwoordelijk zijn voor het renderen van de interface en logische componenten die verantwoordelijk zijn voor het beheren van de state en het ophalen van gegevens. Dit helpt om de complexiteit van mijn applicatie te verminderen en de componenten beter te laten samenwerken.

### Web Services

In de src map vindt u /core, /data, /repository, /rest & /service. De core bevat hulpbestanden die errors loggen zoals logging. De data map is verantwoordelijk voor de connectie met de database & voor het doorvoeren van seeds en aanmaken van tables. Repository zorgt voor de communicatie tussen backend en database. Rest is de communicatie met front-end en vangt requests op en stuurt gevraagde data indien nodig door adhv posts/gets. Service bevat de domein logica.

## Extra technologie

### Front-end Web Development

Ik heb als extra technologieën [i18n](#) & [MUI](#) gebruikt om mijn website te translaten en beter te stylen. Ik heb een multi-language site gecreëerd met talen zoals Frans, Engels en Nederlands. Met MUI heb ik een aantal coole features kunnen toevoegen zoals snackbars, dropdown menus etc.

### Web Services

Ik heb [MULTER](#) gebruikt als extra technologie zodat users een logo zouden kunnen toevoegen bij een gym dat ze toevoegen. Het is de bedoeling dat een user een file kan uploaden in de frontend, deze file wordt bijgehouden in de backend zodat deze gemakkelijk op te halen is. De naam van de file wordt opgeslagen

## Testresultaten

### Front-end Web Development

#### LoginTest

Logintest gaat na of user ingelogd is nadat ze de website inladen en bepaalde knoppen indrukken.

#### MakeGymTest



MakeGymtest kijkt of het lukt om een gym succesfully toe te voegen.

TranslationTest

Translationtest kijkt of de pagina daadwerkelijk vertaald wordt nadat er op een andere taal geklikt wordt.

Web Services

Ik heb userGym met al zijn functionaliteiten getest, dit zijn delete

Coverage:

All files

64.35% Statements 3840/5958 65% Branches 52/80 43.61% Functions 43/74 64.35% Lines 3840/5958

Press n or j to go to the next uncovered block, b, p or k for the previous block.

Filter:

File	Statements	Branches	Functions	Lines
__tests__	100%	58/58	100%	58/58
config	100%	42/42	100%	42/42
src	59.11%	94/159	46.66%	7/15
src/core	63.94%	133/208	66.66%	8/12
src/data	71.51%	113/158	44.44%	4/9
src/repository	34.76%	89/256	75%	6/8
src/rest	74.94%	332/443	70%	14/20
src/service	61.3%	179/292	75%	9/12

PASS

\_\_tests\_\_/rest/userGym.spec.js

userGyms

GET /api/userGym

✓ it should 200 and return all userGyms (308 ms)

POST /api/userGym

✓ it should 201 and create a new userGym (21 ms)

DELETE /api/userGym/:id

✓ it should 200 and delete the requested userGym (10 ms)

File	% Stmts	% Branch	% Funcs	% Lines	Uncovered Line #s
All files	64.35	65	43.61	64.35	
__tests__	100	100	100	100	
helpers.js	100	100	100	100	
config	100	100	100	100	
custom-environment-variables.js	100	100	100	100	
test.js	100	100	100	100	
src	59.11	46.66	83.33	59.11	
createServer.js	59.11	46.66	83.33	59.11	38-39,61,69-74,82-87,89-131,142-148
src/core	63.94	66.66	42.1	63.94	
auth.js	49.53	60	75	49.53	20-24,39-41,44-67,71-92
logging.js	100	71.42	83.33	100	14,22
serviceError.js	55.31	100	0	55.31	8-12,15-16,19-20,23-24,27-28,31-32,35-36,39-40,43-44
src/data	71.51	44.44	80	71.51	
index.js	71.51	44.44	80	71.51	21-33,54-57,83-85,93-96,100-110,114-121,142-143
src/repository	34.76	75	30	34.76	
gym.js	22.58	100	0	22.58	5-6,8-15,18-44,47-64,67-78,80-81,83-85
user.js	29.47	100	14.28	29.47	5-6,9-11,14-15,22-46,49-72,75-85
userGym.js	58.82	71.42	83.33	58.82	13-18,22-37,47-52
src/rest	74.94	70	39.13	74.94	
_gym.js	70.47	100	14.28	70.47	10,13-14,18-23,35-36,40-57,70-71
_health.js	87.09	100	33.33	87.09	8-9,13-14
_user.js	60.52	100	16.66	60.52	9-11,15-16,25-39,44-65,69-71
_userGym.js	81.81	66.66	100	81.81	14-19,32-37
_validation.js	80	60	50	80	12-22,47,64,79,86-90
index.js	100	100	100	100	
src/service	61.3	75	36.84	61.3	
gym.js	44.18	100	0	44.18	5-7,10-14,17-19,22-29,32-36
health.js	85.71	100	0	85.71	13-15
user.js	47.29	75	28.57	47.29	17-40,46-53,64-74,81-84,99-120,131-139
userGym.js	90	75	100	90	48-51,69-72

Test Suites: 1 passed, 1 total

Tests: 3 passed, 3 total

Snapshots: 0 total

Gekende bugs

Front-end Web Development

Zijn er gekende bugs?

User profile wordt niet meteen geupdate, page refresh is required.

## Web Services

Zijn er gekende bugs?

images worden in het ram geheugen opgeslagen op render. Hierdoor blijven images niet voor altijd.

## Front-end Web Development

- Ik was ook van plan give rating toe te voegen maar heb dit uiteindelijk niet gedaan om meer tijd te hebben voor andere examens. U kunt bij gym details een give rating component vinden, deze werkt dus niet in de backend. Enkel visueel voor de user.