

Lander Dhaen (202293982)

- ☒ Front-end Web Development
 - <https://github.com/Web-IV/2324-frontendweb-LanderDhaen>
 - <https://web-iv-2324-frontendweb-landerdhaen.onrender.com>
- ☒ Web Services: GITHUB URL
 - <https://github.com/Web-IV/2324-webservices-LanderDhaen>
 - <https://two324-webservices-landerdhaen.onrender.com>

Logingegevens

Admin

- Gebruikersnaam/e-mailadres: lander.dhaen@gmail.com
- Wachtwoord: 12345678

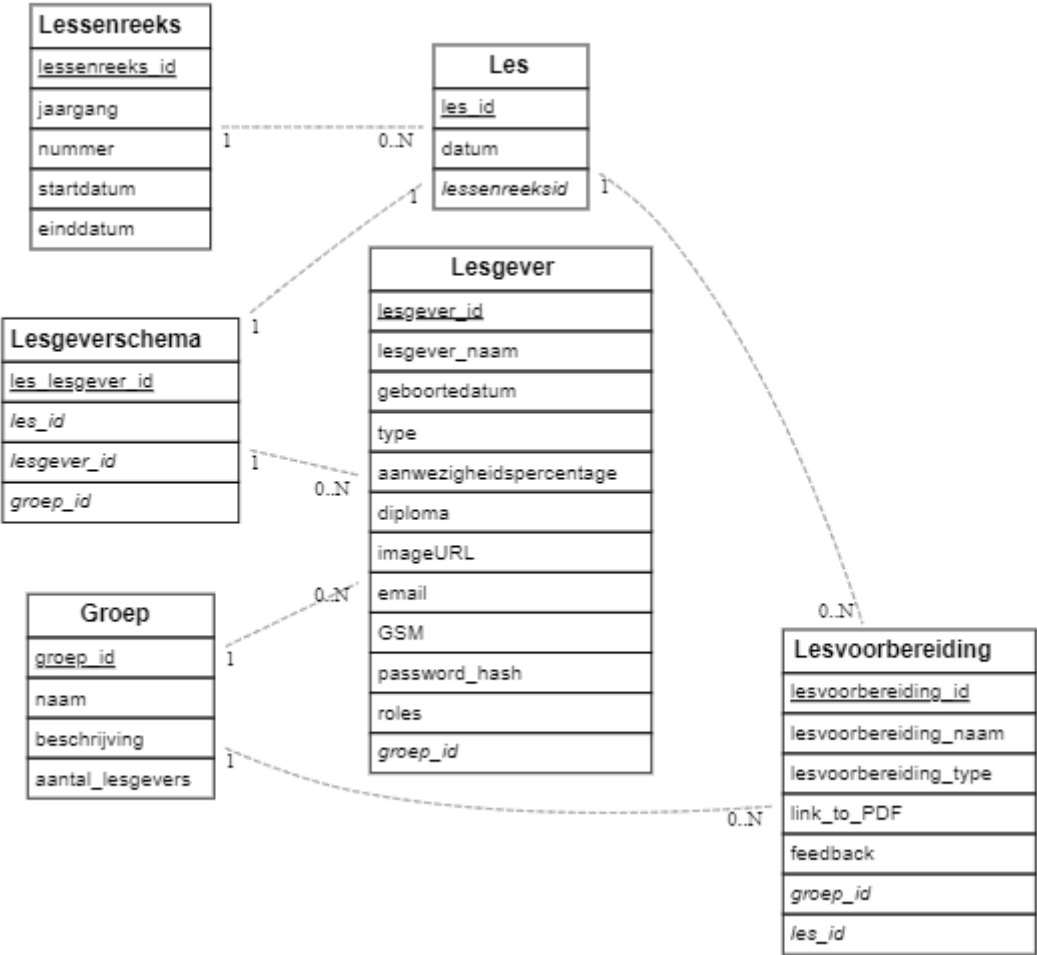
User

- Gebruikersnaam/e-mailadres: evert.walravens@move-united.be
- Wachtwoord: 12345678

Projectbeschrijving

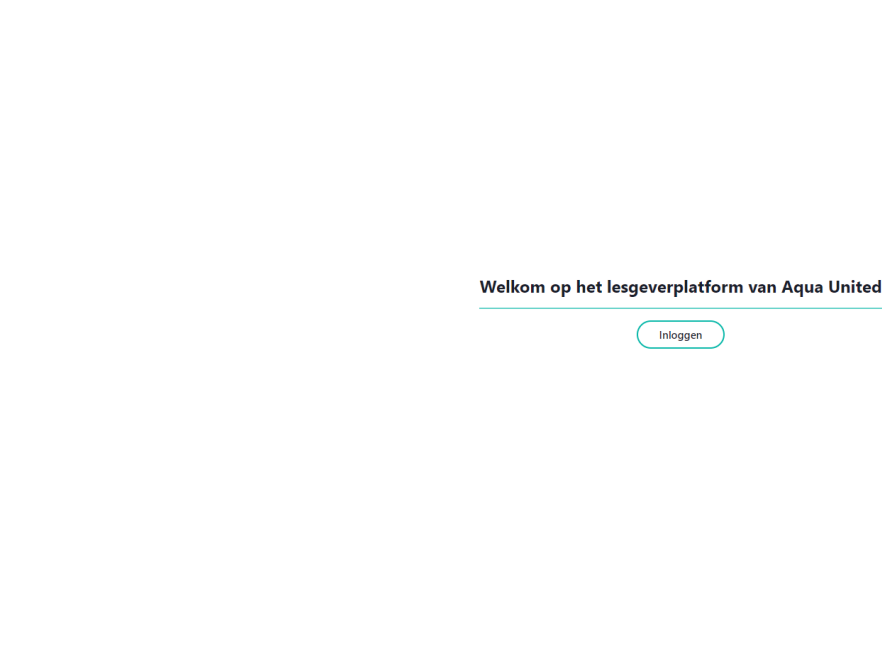
Ik heb een platform ontwikkelt voor Aqua United, een zwemschool binnen Aalst. Op dit platform kunnen leden van het bestuur lesgevers aanmaken, aanpassen en verwijderen. Gegevens van lesgevers, groepen, lesvoorbereidingen en schemas kunnen ze ook raadplegen op verschillende manieren. Het aanmaken en aanpassen van lesgeversschema's kan ook via het platform.

Gewone lesgevers kunnen hun eigen gegevens en de lesvoorbereidingen raadplegen. Daarnaast hebben ze een persoonlijk account waarop hun aanwezigheden per lessenreeks kunnen zien (gebaseerd op de schema's). Ze hebben ook toegang tot een agenda waarop lesdagen ingepland staan.



Screenshots

Landingspagina



Lesgevertabel met sorteropties

MOVEUNITED

Lander Dhaen

Home > Lesgevers > Alle lesgevers

Geef alle lesgevers weer

Voeg een lesgever toe





Zoek een lesgever

☒ Naam

☐ Groep

☐ Type Lesgever

☐ Diploma

NAAM	GROEP	TYPE LESGEVER	DIPLOMA
<div><div></div><div>Lander Dhaen</div><div>Veel aanwezig</div></div>	Losse lesgevers	Lesvrij	Redder
<div><div></div><div>Robbe De Back-End</div><div>Veel aanwezig</div></div>	Dolfijnen	Verantwoordelijke	Initiator
<div><div></div><div>Hannah Van den Steen</div><div>Veel aanwezig</div></div>	Waterschildpadden	Vaste Lesgever	Leerkracht LO
<div><div></div><div>Evert Walravens</div><div></div></div>	Dolfijnen	Verantwoordelijke	Leerkracht LO

Lesgevers opzoeken

Home > Lesgevers > Zoek een lesgever

Geef alle lesgevers weer

Voeg een lesgever toe

Zoek een lesgever

Selecteer een type

☐ Lesvrij

☐ Verantwoordelijke

☐ Vaste Lesgever

☐ Reserve

☐ Redder

Selecteer een diploma

☐ Redder

☐ Initiator

☐ Leerkracht LO

☐ Animator

☐ Geen

Selecteer een groep

☐ Eendjes

☐ Pinguïns

☐ Waterschildpadden

☐ Otters

☐ Walrussen

☐ Orka's


☐ Dolfijnen

☐ Redders

☐ Losse lesgevers

Zoek een lesgever


MOVEUNITED



Lander Dhaen

Geef de details weer!


MOVEUNITED



Robbe De Back-End

Geef de details weer!


MOVEUNITED



Hannah Van den Steen

Geef de details weer!

MOVEUNITED



Evert Walravens

Geef de details weer!

3 / 10

Schema aanmaken

Home > Lesgeverschemas > Lesgeverschema aanmaken

Lesgevers zijn gesorteerd op groep

CHECKBOX	NAAM	GROEP	TYPE LESGEVER
<input type="checkbox"/>	Hannah Van den Steen	Waterschildpadden	Vaste Lesgever
<input type="checkbox"/>	Evert Walravens	Dolfijnen	Verantwoordelijke
<input type="checkbox"/>	Robbe De Back-End	Dolfijnen	Verantwoordelijke
<input type="checkbox"/>	Lander Dhaen	Losse lesgevers	Lesvrij

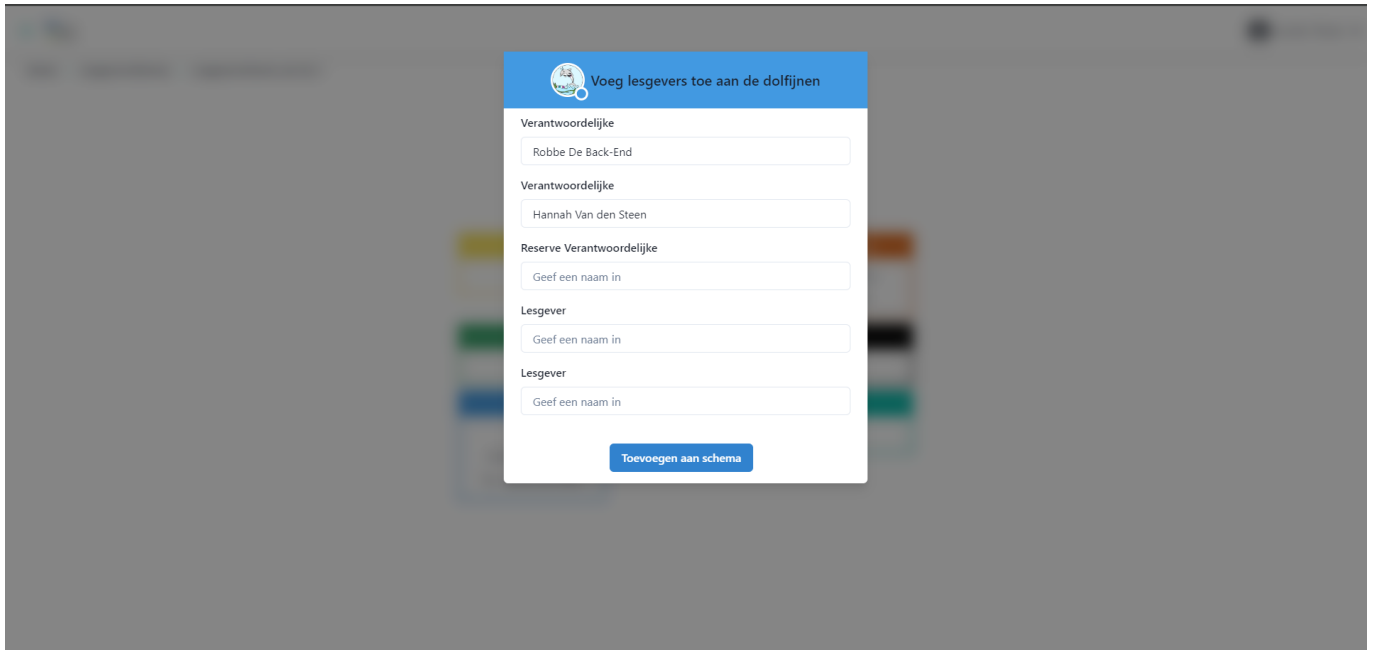
Maak een schema aan!

Schema aanpassen

Home > Lesgeverschemas > Lesgeverschema van les 1

Eendjes	Pinguïns	Waterschildpadden
Eendjes	Pinguïns	Waterschildpadden Lander Dhaen
Otters	Walrusen	Orkas
Otters	Walrusen	Orkas
Dolfijnen	Losse lesgevers	Redders
Dolfijnen Robbe De Back-End Hannah Van den Steen	Losse lesgevers	Redders

Namen invullen/aanpassen



Voeg lesgevers toe aan de dolfijnen

Verantwoordelijke
Robbe De Back-End

Verantwoordelijke
Hannah Van den Steen

Reserve Verantwoordelijke
Geef een naam in

Lesgever
Geef een naam in

Lesgever
Geef een naam in

Toevoegen aan schema

API calls

Niet elke API call werd gebruikt in de front-end

Lesgevers

- `GET /api/lesgevers`: alle lesgevers ophalen
- `GET /api/lesgevers/:id`: lesgever met een bepaald id ophalen
- `GET /api/lesgevers/:id/lesgeverschemas`: aanwezigheden ophalen van lesgever met bepaald id
- `POST /api/lesgevers`: aanmaken van lesgever
- `POST /api/lesgevers`: inloggen van lesgever
- `PUT /api/lesgevers/:id`: lesgever met bepaald id aanpassen
- `DELETE /api/lesgevers/:id`: lesgever met bepaald id verwijderen

Groepen

- `GET /api/groepen`: alle groepen ophalen
- `GET /api/groepen/:id`: groep met een bepaald id ophalen
- `GET /api/groepen/:id/lesgevers`: lesgevers ophalen van groep met een bepaald id
- `GET /api/groepen/:id/lesvoorbereidingen`: lesvoorbereidingen ophalen van groep met een bepaald id
- `POST /api/groepen`: aanmaken van groep
- `PUT /api/groepen/:id`: groep met bepaald id aanpassen
- `DELETE /api/groepen/:id`: groep met bepaald id verwijderen

Lessen

- `GET /api/lessen`: alle lessen ophalen
- `GET /api/lessen/:id`: les met een bepaald id ophalen
- `GET /api/lessen/:id/lesgeverschemas`: schemas ophalen van les met een bepaald id
- `POST /api/lessen`: aanmaken van les

- `PUT /api/lessen/:id`: les met bepaald id aanpassen
- `DELETE /api/lessen/:id`: les met bepaald id verwijderen

Lesgeverschemas

- `GET /api/lesgeverschemas`: alle lesgeverschemas ophalen
- `POST /api/lesgeverschemas`: aanmaken van lesgeverschema
- `PUT /api/lesgeverschemas/:id`: lesgeverschema met bepaald id aanpassen
- `DELETE /api/lesgeverschemas/:id`: lesgeverschema met bepaald id verwijderen

Lessenreeksen

- `GET /api/lessenreeksen`: alle lessenreeksen ophalen
- `GET /api/lessenreeksen/:id`: lessenreeks met een bepaald id ophalen
- `GET /api/lessenreeksen/:id/lessen`: lessen ophalen van lessenreeks met een bepaald id
- `POST /api/lessenreeksen`: aanmaken van lessenreeks
- `PUT /api/lessenreeksen/:id`: lessenreeks met bepaald id aanpassen
- `DELETE /api/lessenreeksen/:id`: lessenreeks met bepaald id verwijderen

Lesvoorbereidingen

- `GET /api/lesvoorbereidingen`: alle lesvoorbereidingen ophalen
- `GET /api/lesvoorbereidingen/:id`: lesvoorbereiding met een bepaald id ophalen
- `POST /api/lesvoorbereidingen`: aanmaken van lesvoorbereiding
- `PUT /api/lesvoorbereidingen/:id`: lesvoorbereiding met bepaald id aanpassen
- `DELETE /api/lesvoorbereidingen/:id`: lesvoorbereiding met bepaald id verwijderen

Behaalde minimumvereisten

Front-end Web Development

- **componenten**
 - ☒ heeft meerdere componenten - dom & slim (naast login/register)
 - ☒ applicatie is voldoende complex
 - ☒ definieert constanten (variabelen, functies en componenten) buiten de component
 - ☒ minstens één form met meerdere velden met validatie (naast login/register)
 - ☒ login systeem
- **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 gepast state management voor globale state - indien van toepassing

- **hooks**

- ☒ gebruikt de hooks op de juiste manier

- **varia**

- ☒ een aantal niet-triviale e2e testen
- ☒ minstens één extra technologie
- ☒ maakt gebruik van de laatste ES-features (async/await, object destructuring, spread operator...)
- ☒ duidelijke en volledige README.md
- ☒ volledig en tijdig ingediend dossier en voldoende commits

Web Services

- **data laag**

- ☒ voldoende complex (meer dan één tabel, 2 een-op-veel of veel-op-veel relaties)
- ☒ één module beheert de connectie + connectie wordt gesloten bij sluiten server
- ☒ heeft migraties - indien van toepassing
- ☒ 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 - indien van toepassing

- **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
- ☒ geen API calls voor entiteiten die geen zin hebben zonder hun ouder (bvb tussentabellen)
- ☒ degelijke autorisatie/authenticatie op alle routes

- **algemeen**

- ☒ er is een minimum aan logging voorzien
- ☒ een aantal niet-triviale integratietesten (min. 1 controller $\geq 80\%$ coverage)
- ☒ minstens één extra technologie
- ☒ maakt gebruik van de laatste ES-features (async/await, object destructuring, spread operator...)
- ☒ duidelijke en volledige README.md
- ☒ volledig en tijdig ingediend dossier en voldoende commits

Projectstructuur

Front-end Web Development

Ik heb mijn src onderverdeeld in mappen voor api, assest, componenten, contexts en pages.

Componenten zijn dom, de pages zijn slim. Alles is onderverdeeld per entiteit om het overzicht te bewaren. Zo is alles omtrent groepen in een submap groepen verdeeld. Enkel de componenten die over meerdere entiteiten gebruikt worden zoals de header staan in de shared submap.

Daarnaast staat de code rond testen in de cypress map.

Web Services

Ik heb mijn project onderverdeeld in mappen voor testing, configuratie en src. In de src map is de onderverdeling gebaseerd op de verschillende lagen. Zo bevat de data submap alles omtrent de data (migrations, seeds, mock-data). De repository submap bevat alles over het uitvoeren van queries op de database. De service submap behandelt het doorgeven van de calls uit de rest laag naar de repository laag. Hier worden ook bepaalde errors opgevangen indien de queries onverwachte data teruggeven. De rest submap beheert alles omtrent het ontvangen, valideren en authoriseren van API calls, om deze dan door te geven naar de service laag.

Elke laag is onderverdeeld in submappen per entiteit om het overzicht te bewaren.

Extra technologie

Front-end Web Development

Ik heb gebruik gemaakt van Chakra UI en Formik als extra functionaliteiten.

ChakraUI: <https://chakra-ui.com/>

Componenten library voor React applicaties. Je importeert reeds gemaakte componenten om snel applicaties te bouwen.

Formik: <https://formik.org/>

Open source form library die declaratief en intuïtief formulieren helpt maken.

Web Services

ObjectMapper: <https://www.npmjs.com/package/object-mapper>

Library die a.d.h.v van een opgestelde map een object omvormt naar een ander object.

Testresultaten

Front-end Web Development

```
it("should show the lesgevers")
```


it("should add a new lesgever and return to the table") it("should show the lesgever in the table") it("should remove the lesgever")

Ik heb een test geschreven die alle lesgevers ophaalt na het inloggen en controleert of deze gegevens wel juist zijn. Daarnaast heb ik een test geschreven die een lesgever aanmaakt, controleert of hij rerout naar de tabel. Nadien controleert hij of de lesgever weldegelijk in de tabel toegevoegd is om hem uiteindelijk weer te verwijderen

Web Services

Ik heb voor elke entiteit elke API call getest, zowel het normale verloop als bijna alle edge-cases (controleren of de juiste foutmelding wordt gegooid). Er wordt ook gecontroleerd of de ingelogde lesgever deze API calls mag uitvoeren.

Het is voor elke entiteit gelijkaardig, dus bij deze een oplijsting van lesgever:

GET /api/lesgevers

- test("should 200 and return all lesgevers")
- test("should 400 when given an argument")
- test("should 403 when not admin")

GET /api/lesgevers/:id

- test("should 200 and return the correct lesgever")
- test("should 404 when requesting not existing lesgever")
- test("should 400 with invalid lesgever_id")
- test("should 403 when not admin and requesting different user")

GET /api/lesgevers/:id/lesgeverschemas

- test("should 200 and return the correct schemas for the requested lesgever")
- test("should 404 when requesting not existing lesgever")
- test("should 400 with invalid lesgever_id")
- test("should 403 when not admin and requesting different user")

PUT /api/lesgevers/:id

- test("should 200 and return the updated lesgever")
- test("should 404 with not existing lesgever")
- test("should 400 with invalid lesgever_id")
- test("should 404 when groep does not exist")
- test("should 400 when missing (voor elke paramater een test)")
- test("should 403 when not admin")

POST /api/lesgevers

- test("should be 201 and return the created lesgever")
- test("should 400 when groep does not exist")
- test("should 400 when duplicate email")
- test("should 400 when duplicate gsm")

- test("should 400 when missing (voor elke paramater een test)")
- test("should 403 when not admin")

DELETE /api/lesgevers/:id

- test("should be 204 and return nothing")
- test("should 400 with invalid lesgever_id")
- test("should 404 with not existing lesgever")
- test("should 403 when not admin")

De AuthHeader werd ook op elke route getest.

Coverage:

File	% Stmts	% Branch	% Funcs	% Lines	Uncovered Line #s
All files	94.15	83.82	100	94.15	
repository	92.05	75.67	100	92.05	
groep.js	90.19	77.77	100	90.19	76-80,89-93
les.js	88.63	72.72	100	88.63	89-93,105-109,118-122
lesgever.js	96.95	84.61	100	96.95	173-175,184-186
lesgeverschema.js	91.85	71.42	100	91.85	38-45,192-194,208-210,225-227,245-246,257-258
lessenreeks.js	86.36	66.66	100	86.36	63-67,84-88,97-101
lesvoorbereiding.js	94.15	81.81	100	94.15	142-146,157-161
rest	100	100	100	100	
groep.js	100	100	100	100	
health.js	100	100	100	100	
index.js	100	100	100	100	
les.js	100	100	100	100	
lesgever.js	100	100	100	100	
lesgeverschema.js	100	100	100	100	
lessenreeks.js	100	100	100	100	
lesvoorbereiding.js	100	100	100	100	
service	90.08	78.68	100	90.08	
_handleDBError.js	49.5	38.46	100	49.5	11-13,18,23,30,43-45,54-56,59-60,63-99
groep.js	95.12	81.81	100	95.12	54-55,72-73
health.js	100	100	100	100	
les.js	94.69	82.35	100	94.69	66-67,84-85,102-103
lesgever.js	93.25	83.33	100	93.25	134-135,152-153,205-207,238-243,248-252
lesgeverschema.js	95.52	85	100	95.52	75-76,99-100,122-123
lessenreeks.js	93.54	72.72	100	93.54	46-47,65-66,83-84
lesvoorbereiding.js	97.27	88.88	100	97.27	113-114,136-137
Test Suites: 7 passed, 7 total					
Tests: 257 passed, 257 total					
Snapshots: 0 total					
Time: 23.464 s					
Ran all test suites.					
Done in 28.27s.					

Gekende bugs

Front-end Web Development

Het ophalen van de lesgever na inloggen duurt lang omdat de eerste call altijd faalt.

Web Services

/