Tristan Burggraeve (202186392)

- ☑ Front-end Web Development
  - 2324-webservices-tristanburggraeve
  - <LINK_ONLINE_VERSIE_HIER>
- ☑ Web Services:
  - 2324-frontendweb-tristanburggraeve
  - <LINK_ONLINE_VERSIE_HIER>

Logingegevens

- Gebruikersnaam/e-mailadres: "a.b@c.com"
- Wachtwoord: "12345678"

Projectbeschrijving

Het doel van mijn project was om bij te houden welke voeding je in huis hebt (in de frigo), en op basis daarvan kom je te weten welke recepten je kan maken. Gebruikers kunnen recepten (recipe) en etenswaren (food) toevoegen, die voor iedereen zichtbaar zijn. Gebruikers kunnen frigo's (fridge) toevoegen waarin ze instanties van voedsel (food_item) kunnen opslaan, dit is enkel voor de gebruiker zichtbaar die ze aangemaakt heeft. Het was de bedoeling dat je op basis van de food_item's die je in je fridges hebt een lijst te zien krijgt van recepten die je kan maken, maar helaas heb ik dit wegens tijdstekort niet kunnen implementeren.

API calls

Gebruikers

- `GET /api/users`: alle gebruikers ophalen
- `GET /api/users/:id`: gebruiker met een bepaald id ophalen

Food

- `GET /api/food` : alle food ophalen
- `GET /api/food/:id` : food met id ophalen
- `POST /api/food` : food createn
- `POST /api/food/:id` : naam en beschrijving van food wijzigen
- `DELETE /api/food/:id` : food met een bepaald id verwijderen

Food_item

- `GET /api/item` : alle food_item's ophalen
- `GET /api/item/:id` : food_item met bepaald id ophalen
- `GET /api/item/frigde/:id` : food_item's die in fridge met een bepaald id zitten ophalen
- `POST /api/item/:id` : waarden van food_item aanpassen
- `POST /api/item` : food_item createn
- `DELETE /api/item/:id` : food_item met bepaalde id verwijderen

Fridge

- `GET /api/fridge` : alle fridges ophalen
- `GET /api/fridge/:id` : fridge met bepaalde id ophalen
- `GET /api/fridge/:id/contents` : inhoud van fridge met een bepaald id ophalen
- `POST /api/fridge` : fridge createn
- `POST /api/fridge/:id` : naam van fridge updaten
- `DELETE /api/fridge/:id` : fridge met bepaalde id verwijderen

Recipe

- `GET /api/recipe` : alle recipes ophalen
- `GET /api/recipe/:id` : recipe met bepaalde id ophalen
- `GET /api/recipe/:id/ingredient` : haalt de ingredienten van een bepaald recept op
- `POST /api/recipe/` : recipe createn
- `POST /api/recipe/:id` : recipe updaten
- `POST /api/recipe/:id/ingredient` : ingredient toevoegen aan recipe
- `DELETE /api/recipe/:id` : recipe met bepaalde id verwijderen

Behaalde minimumvereisten

Front-end Web Development

- **componenten**
  - [x] heeft meerdere componenten - dom & slim (naast login/register)
  - [ ] applicatie is voldoende complex
  - [x] definieert constanten (variabelen, functies en componenten) buiten de component
  - [x] minstens één form met meerdere velden met validatie (naast login/register)
  - [ ] login systeem
- **routing**
  - [x] heeft minstens 2 pagina's (naast login/register)
  - [x] routes worden afgeschermd met authenticatie en ~~autorisatie~~
- **state-management**
  - [x] meerdere API calls (naast login/register)
  - [ ] degelijke foutmeldingen indien API-call faalt
  - [x] gebruikt useState enkel voor lokale state
  - [x] gebruikt gepast state management voor globale state - indien van toepassing
- **hooks**
  - [x] gebruikt de hooks op de juiste manier
- **varia**
  - [ ] een aantal niet-triviale e2e testen
  - [ ] minstens één extra technologie
  - [x] 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

- **datalaag**
  - [x] voldoende complex (meer dan één tabel, 2 een-op-veel of veel-op-veel relaties)
  - [x] één module beheert de connectie + connectie wordt gesloten bij sluiten server
  - [x] heeft migraties - indien van toepassing
  - [x] heeft seeds
- **repositorylaag**
  - [x] definieert één repository per entiteit (niet voor tussentabellen) - indien van toepassing
  - [x] mapt OO-rijke data naar relationele tabellen en vice versa - indien van toepassing
- **servicelaag met een zekere complexiteit**
  - [x] bevat alle domeinlogica
  - [x] bevat geen SQL-queries of databank-gerelateerde code
- **REST-laag**
  - [x] meerdere routes met invoervalidatie
  - [x] degelijke foutboodschappen
  - [x] volgt de conventies van een RESTful API
  - [x] bevat geen domeinlogica
  - [x] geen API calls voor entiteiten die geen zin hebben zonder hun ouder (bvb tussentabellen)
  - [x] degelijke ~~authorisatie~~/authenticatie op alle routes (authenticatie alleen voor data die niet door iedereen gezien mag worden)
- **algemeen**
  - [x] er is een minimum aan logging voorzien
  - [ ] een aantal niet-triviale integratietesten (min. 1 controller >=80% coverage)
  - [x] minstens één extra technologie
  - [x] 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

> Hoe heb je jouw applicatie gestructureerd (mappen, design patterns, hiërarchie van componenten, state...)?

Web Services

src bestaat uit service/rest/repository

Extra technologie

Front-end Web Development

> Wat is de extra technologie? Hoe werkt het? Voeg een link naar het npm package toe!

Web Services

mysql2-migrations voor migrations

Testresultaten

geen testen

Gekende bugs

Front-end Web Development

geen idee

Web Services

geen idee

Wat is er verbeterd/aangepast?

eerste zittijd heb ik niets ingediend