

# Gilles Van Cleemput (182542gv)

---

Gilles Van Cleemput 182542gv

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

## Logingegevens


### dit is een admin account

- Gebruikersnaam/e-mailadres: test@gmail.com
- Wachtwoord:Server123\*

### dit is een user account

- Gebruikersnaam/e-mailadres: user@gmail.com
- Wachtwoord:Server123\*

## Projectbeschrijving

Mijn web project gaat over een nieuwe vegan food bar in Gent genaamd SinSin. De food bar bestaat sinds de zomer van 2022 en wordt open gehouden door mijn schoonzus. De foodbar wou zelf een website hebben waar mensen op konden bestellen. Hier door zouden ze geen ubereats of takeaway moeten betalen om hun services te kunnen gebruiken. dit is het erd

## Screenshots

homescreen menu page bestelling page bestelling form user page aanpassen user info pagina user history

## 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

Hoe heb je jouw applicatie gestructureerd (mappen, design patterns, hiërarchie van componenten, state...)? Ik heb mijn applicatie onderverdeeld in componenten voor elke pagina. Ik bouw dan de pagina op met zulke componenten. Het eerste component in een map is het hoofd component waar de rest op wordt gebouwd.

### Web Services

Hoe heb je jouw applicatie gestructureerd (mappen, design patterns...)? Ik heb de rest API gestructureerd zoals in de de les. Ik heb de rest api opgesplitst in het 3 lagen model rest, service en repository.

## Extra technologie

### Front-end Web Development

Voor mijn front-end project heb ik de extra technologie emailjs gebruikt. Het is een javascript library dat help met versturen van emails door alleen client-side technologie te gebruiken. Het heeft geen server nodig je hoeft alleen een supported email service gebruiken zoals gmail en een van hun SDK libraries versturen de email. link: <https://www.npmjs.com/package/emailjs>

### Web Services

De extra technologie dat ik heb gebruikt in het web services project is swagger. Swagger is een open-source software framework. Met Swagger kun je de structuur van je API's beschrijven zodat machines ze kunnen lezen.


link: <https://www.npmjs.com/package/swagger>

## Testresultaten

### Front-end Web Development

Ik heb een test geschreven van het aanpassen van een user account. De test navigeert zich eerst naar de form om de user te weizigen. De form wordt ingevuld en bevestigd. Er wordt dan gekeken of de user effectief is aangepast door naar de user info pagina te navigeren. De tweede test is het plaatsen van een bestelling. De test vult eerst de winkelmand aan om dan verder te gaan de bestelling pagina. Op de bestelling pagina wordt de bestelling bevestigd. Na het bevestigen van de bestelling wordt er genavigeert naar de pagina waar de bestelling history wordt getoont en wordt er gecontroleert of de bestelling is geplaatst.

### Web Services

De eerste test werd geschreven over de het menu. De test probeert de GET, POST, PUT en DELETE van de API voor de menu tabel. De tweede test werd geschreven over de user tabel. De test probeert de checkforuser GETbyAuthId en het aanmaken van een user.  de coverage van de testen

# Gekende bugs

## Front-end Web Development

### Web Services

Swagger heeft enkele problemen. Sommige beschrijvingen in de web voorstelling van swagger willen niet uitvoeren. Dit is omdat de gebruiker niet is geautoriseerd voor die api calls uit te voeren. Ook zijn de fout codes van swagger niet goed gelinked en staat er gewoon string als uitvoer in de plaats van de gepaste melding zoals 404 niet gevonden.

Als de backend testen worden uitgevoerd is er een error dat te maken heeft met swagger dit hindert te testen niet. Alle testen worden nog correct uitgevoerd en geslaagd.