

# Quinten Deconinck (202293145)

- ☒ Front-end Web Development
  - <https://github.com/Web-IV/2324-frontendweb-QuintenDeconinck>
  - <https://2324-frontendweb-quinten-deconinck.vercel.app/>
- ☒ Web Services:
  - <https://github.com/Web-IV/2324-webservices-QuintenDeconinck>
  - <https://frituurdaa-api.onrender.com>

## Logingegevens

```
//User
email: user@email.com
password: password

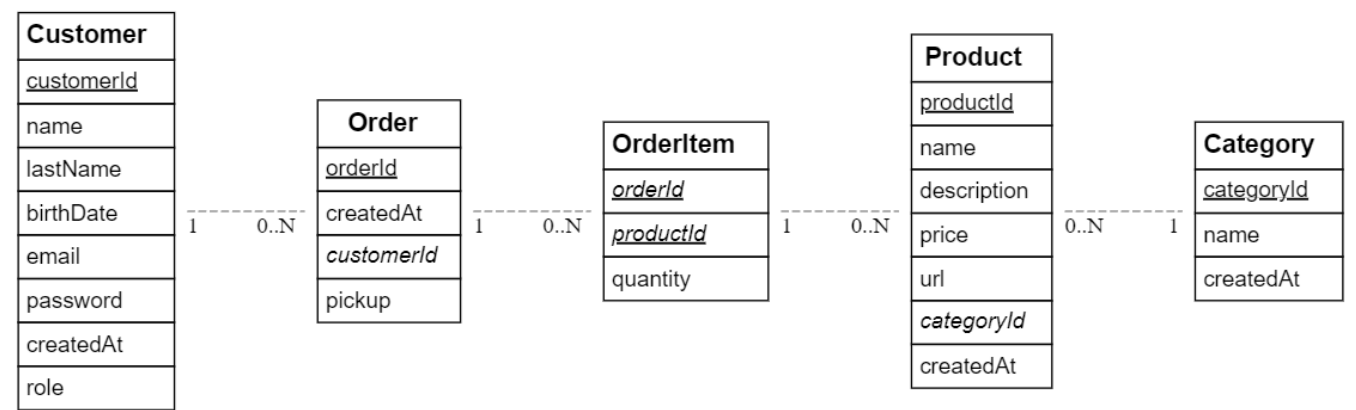
//Admin
email: admin@email.com
password: password

//Admin
email: quinten@email.com
password: StrongPassword
```

## Projectbeschrijving

Het project is een website voor een lokaal frituur (Frituur d'Aa) om klanten de mogelijkheid te geven online te bestellen. Verder is er ook een admin panel zodat de eigenaar producten/categorieën kan toevoegen/aanpassen/verwijderen en bestellingen kan bekijken. De gebruiker kan dus aanmelden en vanaf dan heeft deze de mogelijkheid om bestellingen te plaatsen en zijn eerder geplaatste bestellingen te bekijken.

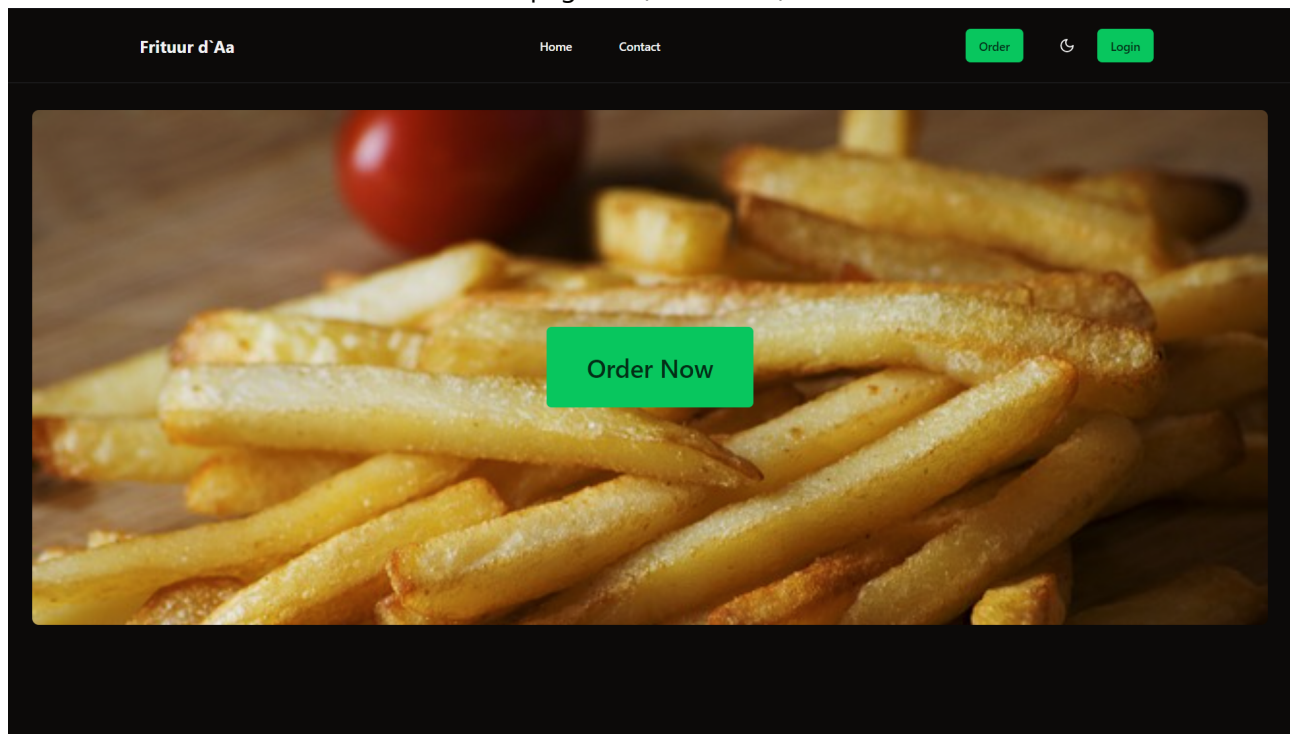
Hieronder een afbeelding van het EERD van de databank:



## Screenshots

Dit is de Home pagina ("/") die bovenaan het volgende toont:

- de pagina met contactgegevens ("/contact") (deze heb ik niet echt geïmplementeerd)
- de bestelpagina ("/order") waar de klant zijn bestelling kan plaatsen
- de dark/light mode toggle
- de login knop ("/login") als de gebruiker niet ingelogd is, anders toont deze een avatar die een submenu toont met links naar de account pagina's ("/account")



Hier de Order page die een lijst toont met alle categorieën die de user kan gebruiken om te filteren. Een lijst met alle producten die de gebruiker aan zijn shopping cart kan toevoegen. Rechts de shopping cart die de totale prijs toont en de gebruiker toelaat zijn order te bekijken en nog aan te passen alvorens de Checkout knop te gebruiken om naar de checkout pagina te gaan ("/checkout") (er is uiteraard geen betalingssysteem geïmplementeerd dus we gaan er vanuit dat de betaling slaagt).

Frituur d'Aa

Home

Contact

Order

Create your order

Categories

Drinks

Fries

Preparations

Cold Sauces

Snacks

Bicky Burgers

Chicken/Turkey Snacks

All Products

Cola

\$2.00

Add to Cart

Cola Zero

\$2.00

Add to Cart

Fanta

\$2.00

Add to Cart

Sprite

\$2.00

Add to Cart

Water

\$2.00

Add to Cart

Your Cart

Cola

Quantity: 1

\$2.00

Fanta

Quantity: 1

\$2.00

Sprite

Quantity: 1

\$2.00

Total: \$6.00

Checkout

Frituur d'Aa

Home

Contact

Order

Thank you for your order!

Order Overview:

| Product Name | Quantity | Price  |
|--------------|----------|--------|
| Cola         | 1        | \$2.00 |
| Cola Zero    | 1        | \$2.00 |
| Fanta        | 1        | \$2.00 |
| Sprite       | 1        | \$2.00 |

An email confirmation has been sent to your email address.  
If something went wrong, please call us at:123-456-7890

Go back to the homepage

Dit zijn de verschillende paginas in het admin dashboard ("/admin"):

- de producten pagina ("/admin/products") waar de admin producten kan toevoegen/aanpassen/verwijderen
- de categorieën pagina ("/admin/categories") waar de admin categorieën kan toevoegen/verwijderen
- de bestellingen pagina ("/admin/orders") waar de admin alle bestellingen kan bekijken

Frituur d'Aa Admin Dashboard

Manage your products, categories and orders

Products

Categories

Orders

Add products

Product name

Description

Price


Image Url

Categories


-- Select a categorie --

Add product


Reset



Cola



Cola Zero



Fanta

Frituur d'Aa Admin Dashboard

Manage your products, categories and orders

Products

Categories

Orders

Categories

Drinks

Fries

Preperations

Cold Sauces

Snacks

Bicky Burgers

Chicken/Turkey Snacks

Category name

Add Category

**Frituur d'Aa Admin Dashboard**

Manage your products, categories and orders

Products

Categories

Orders

**Check your orders here**

Total Price: 9.00

Order Date: December 21, 2023 17:10:46

[Show details](#)

Total Price: 37.00

Order Date: December 21, 2023 17:10:46

[Show details](#)

Total Price: 45.50

Order Date: December 21, 2023 17:10:47

[Hide details](#)

Item: Cola

Price: 2

Quantity: 4

Item: Bicky Saus

Price: 4.2

Quantity: 5

## API calls

Swagger: <https://frituurdAA-api.onrender.com/api>

### ### Customers

- `PUT /customers/:id`: een customer met een bepaald id updaten

### ### Categories

- `GET /categories`: alle categorieën ophalen
- `POST /categories`: een nieuwe categorie toevoegen
- `DELETE /categories/:id`: een categorie met een bepaalde id verwijderen

### ### Products

- `GET /products`: alle producten ophalen
- `POST /products`: een nieuw product toevoegen
- `PUT /products/:id`: een product met een bepaalde id updaten
- `DELETE /products/:id`: een product met een bepaalde id verwijderen

### ### OrderItems

- `POST /order-items`: nieuwe order items toevoegen

### ### Orders

- `GET /orders`: alle orders ophalen
- `GET /orders/user/:id`: alle orders voor een bepaalde gebruiker ophalen
- `GET /orders/:id`: een order met een bepaalde id ophalen
- `POST /orders`: een nieuwe order toevoegen

```
### Auth
```

- `POST /login`: inloggen
- `POST /register`: registreren

## 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

- **datalaag**

- ☒ 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

- **repositorylaag**
  - ☒ 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 gebruik gemaakt van de T3 stack dus zij maken een mappenstructuur aan wanneer het project geïnitieerd wordt. De T3 stack bestaat uit NextJs, Typescript, Tailwindcss, Prisma en tRPC. Aangezien we werken met een aparte REST API heb ik de mappen van Prisma en tRPC verwijderd. De mappen zijn als volgt gestructureerd:

- **public** : bevat de images die ik gebruik in dit project
- **src** : bevat alle files en folders van het project behalve de config files
  - **api** : bevat de calls naar de REST API per entiteit
  - **components** : bevat alle componenten zowel eigen gemaakte als van mijn UI-library Shadcn
  - **contexts** : bevat de context voor authenticatie
  - **lib** : utils file die nodig is om shadCN UI-library te gebruiken
  - **pages** : op het moment van initialisatie was er nog geen support voor app router dus gebruik ik de pages router van NextJs
    - **account** : folder die alle /account paths bevat zoals /account/customer-car en /account/orders
    - **admin** : folder die alle /admin paths bevat zoals /admin/products, /admin/categories en /admin/orders

- contact : file voor de /contact pagina
- order : file voor de /order pagina
- layout: layout file voor de applicatie die de navbar bevat
- login : file voor de /login pagina
- register : file voor de /register pagina
- index : file voor de / pagina
- styles : bevat de global styles voor het project, gegenereerd door Shadcn voor darkmode/lightmode
- types.ts : Typescript file die verschillende types bevat die ik hergebruik doorheen het project

## Web Services

Ik heb gebruik gemaakt van NestJs dus heb ik hun mappenstructuur gebruikt: door de CLI te gebruiken kan je voor elke entiteit een map genereren die een service, controller, dto en entity bevat. Verder maakt NestJs ook automatisch test files aan voor elke entiteit. De mappen zijn als volgt gestructureerd:

- prisma : folder die prisma schema file en seed.ts file bevat
- test : initialisatie van jest
- src :
  - auth : bevat alle files die met authenticatie en autorisatie te maken hebben
  - categories
  - config
  - customers
  - order-items
  - orders
  - prisma
  - products
  - main.ts

## Extra technologie

### Front-end Web Development

- **T3 Stack:**
  - Ik heb gebruik gemaakt van de T3 stack en het bijbehorende create-t3-app om mijn project te initialiseren. De T3 stack bestaat uit NextJs, Typescript, Tailwindcss, tRPC en dan de keuze tussen Prisma en Drizzle en eventueel NextAuth. Aangezien we een aparte REST API hebben heb ik Prisma, tRPC en NextAuth niet gebruikt en deze dus verwijderd uit mijn project na initialisatie. <https://create.t3.gg> / <https://www.npmjs.com/package/create-t3-app>

- **NextJs:**

NextJs is een framework om makkelijk web applicaties te maken. Het is gebaseerd op React en komt met een ingebouwde file-based router, Image Optimization, ... en nog veel meer. <https://nextjs.org>

- **Typescript:**



Typescript is een superset van Javascript en zorgt ervoor dat je project strongly typed is. Dit wil zeggen voor alle functies en attributen het type ingeven zodat Typescript aan type checking kan doen. Op deze manier kan je bijvoorbeeld geen props vergeten doorgeven of een return type teruggeven anders dan wat je gespecificeerd hebt. <https://www.npmjs.com/package/typescript>

- **Tailwindcss:**

Tailwindcss maakt gebruik van gewone css styles maar in plaats van alle elementen van de webpagina een class te geven en de styles per class te definiëren kan je de styles gewoon inline direct per component opsommen. <https://www.npmjs.com/package/tailwindcss>

- **Tanstack Query:**

Ik heb gebruik gemaakt van Tanstack Query in plaats van SWR. Tanstack Query of het vroegere react-query maakt gebruik van keys om mutations en queries te cachen en op basis daarvan ook te refetchen. <https://www.npmjs.com/package/@tanstack/react-query>

- **ShadcnUI:**

ShadcnUI is een relatief nieuwe UI collectie, geen library. Dit wil zeggen dat je via de command line enkel de componenten installeert die je echt gaat gebruiken en kan deze dan callen in het project bv. een **Button** component ipv de native **button** component van HTML in plaats van een gewone en hierbij kan je dan verschillende presets kiezen. Dit is geen component library dus er is geen package beschikbaar maar hier is de website: <https://ui.shadcn.com>

## Web Services

- **NestJs:**

NestJs is een alternatief voor NodeJs en ik heb hier vooral voor gekozen omdat het strong support heeft voor Typescript. <https://www.npmjs.com/org/nestjs>

- **Typescript:**

Typescript is een superset van Javascript en verplicht je om alle types in het project te definiëren. <https://www.npmjs.com/package/typescript>

- **Prisma:**

Prisma is een ORM (Object Relational Mapper) om zonder zelf SQL te moeten schrijven, interacties te kunnen hebben met de database. <https://www.npmjs.com/package/prisma>

- **Swagger:**

Swagger is een tool om een API te documenteren. Je kan hier de verschillende endpoints en hun parameters en responses in documenteren. Verder kan je ook een voorbeeld van een request en response toevoegen. In NestJs is het erg makkelijk om deze op te stellen en ik heb deze doorheen het hele project gebruikt om al mijn endpoints te testen in plaats van PostMan. <https://www.npmjs.com/package/@nestjs/swagger>

- Passport:

Passport is een authentication middleware voor NodeJs. Ik heb deze gebruikt om de login en register te implementeren. Voor NestJs is er een aparte package die gebruik maakt van de NestJs standaarden.

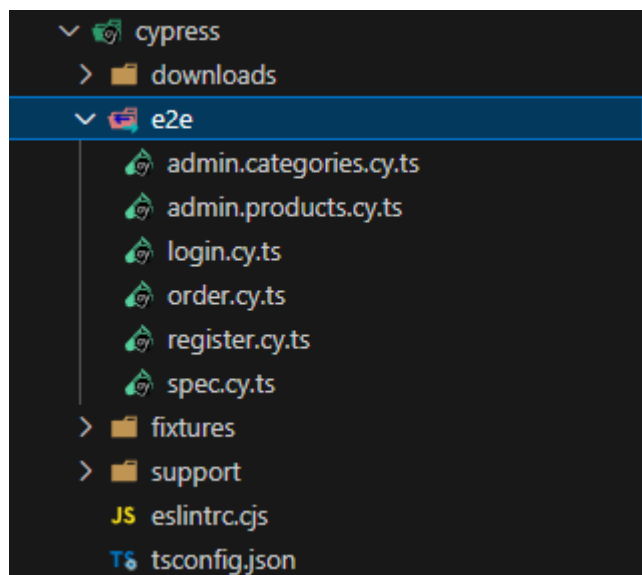
<https://www.npmjs.com/package/@nestjs/passport>

## Testresultaten

### Front-end Web Development

Zoals je op de foto hieronder kan zien heb ik testen geschreven voor de volgende pagina's:

- /login : zowel gefaalde als correcte logins
- /register : zowel gefaalde als correcte registraties
- /order : order maken en alle functionaliteiten van de pagina uitproberen
- /admin/products : producten toevoegen en weer verwijderen
- /admin/categories : categorieën toevoegen en weer verwijderen



### Web Services

Elke file die door NestJs is aangemaakt is getest. Dit wil zeggen elke .controller.ts en .service.ts file heeft een equivalente .controller.spec.ts en .service.spec.ts file die testen definieert. Alle testen slagen, het is alleen spijtig dat dit niet werkt wanneer ik de AuthGuard en RolesGuard toevoeg aan de endpoints. Het is me niet gelukt om deze te mocken binnen NestJs, daarom moet er dus naar de **test** branch geswitched worden om de testen te runnen.

Dit is uiteraard erg spijtig, maar na veel opzoekwerk heb ik besloten om de testen te runnen zonder de protectie van de guards. Ik heb alles van authenticatie en role based protection manueel Swagger getest en alles werkt perfect. Ik heb ook de coverage van de testen hieronder toegevoegd:

```

PASS src/order-items/order-items.service.spec.ts (8.476 s)
PASS src/auth/roles.guard.spec.ts (8.482 s)
PASS src/auth/auth.guard.spec.ts (8.525 s)
PASS src/app.controller.spec.ts (8.87 s)
PASS src/orders/orders.service.spec.ts (9.123 s)
PASS src/customers/customers.service.spec.ts (9.108 s)
PASS src/products/products.service.spec.ts (9.115 s)
PASS src/categories/categories.service.spec.ts (9.136 s)
PASS src/order-items/order-items.controller.spec.ts (9.385 s)
PASS src/orders/orders.controller.spec.ts (9.62 s)
PASS src/categories/categories.controller.spec.ts (9.648 s)
PASS src/products/products.controller.spec.ts (9.734 s)
PASS src/customers/customers.controller.spec.ts (9.761 s)

```

| File                      | % Stmts | % Branch | % Funcs | % Lines | Uncovered Line #s         |
|---------------------------|---------|----------|---------|---------|---------------------------|
| All files                 | 92.94   | 71.26    | 100     | 92.94   |                           |
| src                       | 100     | 100      | 100     | 100     |                           |
| app.controller.ts         | 100     | 100      | 100     | 100     |                           |
| app.service.ts            | 100     | 100      | 100     | 100     |                           |
| src/auth                  | 96.92   | 69.23    | 100     | 96.92   |                           |
| auth.guard.ts             | 94.28   | 42.85    | 100     | 94.28   | 18-19                     |
| role.enum.ts              | 100     | 100      | 100     | 100     |                           |
| roles.decorator.ts        | 100     | 100      | 100     | 100     |                           |
| roles.guard.ts            | 100     | 100      | 100     | 100     |                           |
| src/categories            | 87.8    | 70.58    | 100     | 87.8    |                           |
| categories.controller.ts  | 100     | 100      | 100     | 100     |                           |
| categories.service.ts     | 78.57   | 54.54    | 100     | 78.57   | ...7-29,39-41,56-58,66-68 |
| src/categories/dto        | 100     | 100      | 100     | 100     |                           |
| create-category.dto.ts    | 100     | 100      | 100     | 100     |                           |
| update-category.dto.ts    | 100     | 100      | 100     | 100     |                           |
| src/categories/entities   | 100     | 100      | 100     | 100     |                           |
| category.entity.ts        | 100     | 100      | 100     | 100     |                           |
| src/customers             | 89.37   | 66.66    | 100     | 89.37   |                           |
| customers.controller.ts   | 100     | 100      | 100     | 100     |                           |
| customers.service.ts      | 79.26   | 50       | 100     | 79.26   | ...8-40,50-52,68-70,78-80 |
| src/customers/dto         | 100     | 100      | 100     | 100     |                           |
| create-customer.dto.ts    | 100     | 100      | 100     | 100     |                           |
| update-customer.dto.ts    | 100     | 100      | 100     | 100     |                           |
| src/customers/entities    | 100     | 100      | 100     | 100     |                           |
| customer.entity.ts        | 100     | 100      | 100     | 100     |                           |
| src/order-items           | 92.85   | 80       | 100     | 92.85   |                           |
| order-items.controller.ts | 100     | 100      | 100     | 100     |                           |
| order-items.service.ts    | 86.36   | 66.66    | 100     | 86.36   | 18-20                     |
| src/order-items/dto       | 100     | 100      | 100     | 100     |                           |
| create-order-item.dto.ts  | 100     | 100      | 100     | 100     |                           |
| src/order-items/entities  | 100     | 100      | 100     | 100     |                           |
| order-item.entity.ts      | 100     | 100      | 100     | 100     |                           |
| src/orders                | 89.65   | 71.42    | 100     | 89.65   |                           |
| orders.controller.ts      | 100     | 100      | 100     | 100     |                           |
| orders.service.ts         | 82.35   | 55.55    | 100     | 82.35   | 17-19,35-37,54-56,64-66   |
| src/orders/dto            | 100     | 100      | 100     | 100     |                           |
| create-order.dto.ts       | 100     | 100      | 100     | 100     |                           |
| src/orders/entities       | 100     | 100      | 100     | 100     |                           |
| order.entity.ts           | 100     | 100      | 100     | 100     |                           |
| src/products              | 88.28   | 70.58    | 100     | 88.28   |                           |
| products.controller.ts    | 100     | 100      | 100     | 100     |                           |
| products.service.ts       | 78.26   | 54.54    | 100     | 78.26   | ...7-29,40-42,55-57,65-67 |
| src/products/dto          | 100     | 100      | 100     | 100     |                           |
| create-product.dto.ts     | 100     | 100      | 100     | 100     |                           |
| update-product.dto.ts     | 100     | 100      | 100     | 100     |                           |
| src/products/entities     | 100     | 100      | 100     | 100     |                           |
| product.entity.ts         | 100     | 100      | 100     | 100     |                           |

```

Test Suites: 13 passed, 13 total
Tests:       53 passed, 53 total
Snapshots:   0 total
Time:        10.628 s

```

```

Run all test suites

```

## Gekende bugs

### Front-end Web Development

Voor zover ik weet zijn er geen gekende bugs.

## Web Services

- Het lukt niet om de api te testen wanneer de Authenticatie Guards worden toegepast. Daarom is er een "test" branch die gewoon de useGuards(AuthGuard en RolesGuard) weglaat en de api dus test zonder de authentication header en zonder te checken of de gebruiker admin is of gewoon user. Ik heb manueel op Swagger al mijn endpoints getest met de useGuards(AuthGuard, RolesGuard) en alles werkt perfect, het probleem ligt dus bij het mocken in de test scenarios. Ik heb erg lang liggen proberen om de authenticatie te kunnen mocken maar dit bleek niet mogelijk.
- Voor de testen wordt gebruik gemaakt van dezelfde databank als de online applicatie draait. In Prisma is het blijkbaar momenteel niet mogelijk om een aparte databank te gebruiken voor testen. Ik heb dit opgelost door de testen te laten runnen op een aparte branch die de databank seed, de testen runt en dan de databank opnieuw seed. Dit is natuurlijk niet ideaal maar ik heb geen andere oplossing gevonden en ik heb er dan ook voor gezorgd dat de seed file voldoende data bevat zodat het verschil eigenlijk niet zo groot is.