Υπολογιστικό Νέφος και Ομίχλη ΠΛΗ513

Εργασία για το Μάθημα

Αναφορά

Καράλης Αστερινός 2020030107

Frontend:

Το frontend κομμάτι της εργασίας μου αποτελείται απο το **server.js** που τρέχει το eshop στον localhost καθώς και απο το **directory Public** που περιέχει τα html αρχεία για την εφαμορφή (**index.html, cart.html, my-products.html, order.html**), το **style.css** και το **script.js** που δίνει το functionality στις σελίδες.

Για το κομμάτι αυτό βασίστηκα στα tutorials απο το discord για να σχεδιάσω την σελίδα αλλά και για να επιτρέψω την εποικωνία με το API για τα προιόντα (μέσω του script.js). Επίσης βασίστηκα σε αρκετά παραδείγματα στο Github.

Τεχνολογίες: **JavaScript**, **CSS, HTML**, **Express** (framework), **PG** (postgreSQL client), **CORS** (security feature), **Axios** (library for http requests) , **Postman** (testing)

Backend:

Το backend κομμάτι της εργασίας μου αποτελείται από τα API που επικοινωνούν με τις βάσεις δεδομένων. Συγκεκριμένα το **api.js** είναι το **Products API** που επικοινωνεί με την βάση για τα προιόντα **eshop_db** ενώ το **api2.js** είναι το **Orders API** που επικοινωνεί με την βάση για τις παραγγελείες **orders_db**.

Για το κομμάτι αυτό βασίστηκα κυρίως στα tutorials στο discord για να φτιάξω τα API αλλά και για να καταφέρω να δημιουργήσω τα containers για το κομμάτι του Docker. Επίσης βασίστηκα ξανά σε διάφορα ανεβασμένα tutorials στο Github καθώς είχα θέματα στο κομμάτι των URL όταν πλέον έβαλα το project σε containers.

Τεχνολογίες: **JavaScript**, **PG**, **CORS**, **PostgreSQL**, **Docker Desktop**, **Postman** (testing)

Keycloak:

Στο κομμάτι του **authentication/authorization** χρησιμοποίησα το **Keycloak** μαζί με μία **Postgres (v17)** βάση δεδομένων.

Τρέχοντας τα containers τους, έκανα access την σελίδα του admin στην οποία δημιούργησα το **eshop realm** σύμφωνα με την εκφώνηση (client, ρόλοι κλπ) το οποίο **έκανα export** στο **directory realms** του root directory. Η **σελίδα του admin** είναι προσβάσιμη στο **localhost:8080** με credentials: admin, adminpassword. Για το κομμάτι του **login/register** χρησιμοποίησα το **default theme του keycloak** όπου όρισα σαν **default ρόλο** κάθε νέου χρήστη που δημιουργήται να είναι '**customer**'. Θα πρέπει να κάνουμε login στο localhost:8080 ως **admin** και να **αλλάξουμε manually** τον **ρόλο** κάποιου χρήστη.

Στο realm υπάρχουν **2 έτοιμοι χρήστες** που έφτιαξα: ο **seller1** (seller1 user/password) και ο **customer1** (customer1 user/password) οι οποίοι βλέπουν τα αντίστοιχα pages όταν κάνουν login (**Υπάρχει** και ο **ρόλος admin** που μπορεί να οριστεί σε κάποιο χρήστη ώστε να μπορεί να δει όλες τις σελίδες, **η λογική για αυτό βρίσκεται στο script.js**). Το **login** είναι **απαίτητο** ώστε να μπορεί κάποιος να δει οποιαδήποτε σελίδα του eshop (products, cart κλπ.). Μπορούμε να κάνουμε κανονικά login/register καθώς **logout**.

Kafka:

Το κομμάτι αυτό δεν κατάφερα να το υλοποιήσω καθώς δεν μου έφτανε ο χρόνος. Έκανα μια προσπάθεια προσθέτοντας το **kafka/zookeeper** στο docker-compose και κατάφερα να βρω στο internet κάποια παραδείγματα χρήσης του σε js ωστόσο είναι πρακτικά ανύπαρτκο στο project ενώ το container του kafka μου παρουσίαζε errors που δεν πρόλαβα να κάνω debug. Επίσης τα προιόντα εμφανίζονται χωρίς να υπάρχει έξτρα μεταβλητή στην βάση που να υποδικνύει σε ποιόν seller ανήκουν.

Δομή τελικού project:

```
e-shop-project/
├── api/
│   ├── api.js (products api that communicates with eshop_db, port 5000)
│   ├── Dockerfile
├── api2/
│   ├── api2.js (orders api that communicates with orders_db, port 5001)
│   ├── Dockerfile
├── public/
│   ├── cart.html
│   ├── index.html
│   ├── my-products.html
│   ├── orders.html
│   ├── script.js (holds the logic for the frontend part)
│   ├── style.css
│   ├── keycloak.min.js
│   ├── keycloak.json
├── realms/
│   ├── eshop-realm.json (το export του τελικού realm)
├── init_eshop_db.sql    (database initialization)
├── init_orders_db.sql            -//-
├── server.js (frontend server, port 3000)
├── Dockerfile
├── docker-compose.yml
```

Τωρινή κατάσταση της τελικής εργασίας:

Κατεβάζοντας και κάνοντας extract το zip αρχείο, αν τρέξουμε στο τερματικό **docker compose up --build** μέσα στο root folder, το eshop θα δημιουργηθεί χωρίς κανένα θέμα και θα τρέχει στην διεύθυνση **http://localhost:3000** ενώ τα databases θα είναι αρχικά άδεια.

Στο συγκεκριμένο build ο χρήστης μπορεί να δει τα προιόντα, να κάνει αναζητήση προιόντος, να προσθέσει στο καλάθι και έπειτα στο καλάθι να δημιουργήσει μια παραγγελεία ή να καθαρίσει το καλάθι. Μπορεί να προσθέσει, να διαγράψει ή να τροποποιήσει τις πληφορορίες κάποιου προιόντος ενώ μπορεί να δει και τις παραγγελείες που έχει πραγματοποιήσει.

Ο χρήστης θα πρέπει να κάνει login ή register για να κάνει access οποιαδήποτε σελίδα του eshop. Κάνοντας εγγραφή κάθε χρήστης είναι αυτόματα customer ενώ πρέπει ο admin να του αλλάξει role σε seller.

Μπορούμε να κάνουμε access το admin page του Keycloak στο http://localhost:8080 με credentials: user-admin, password-adminpassword.