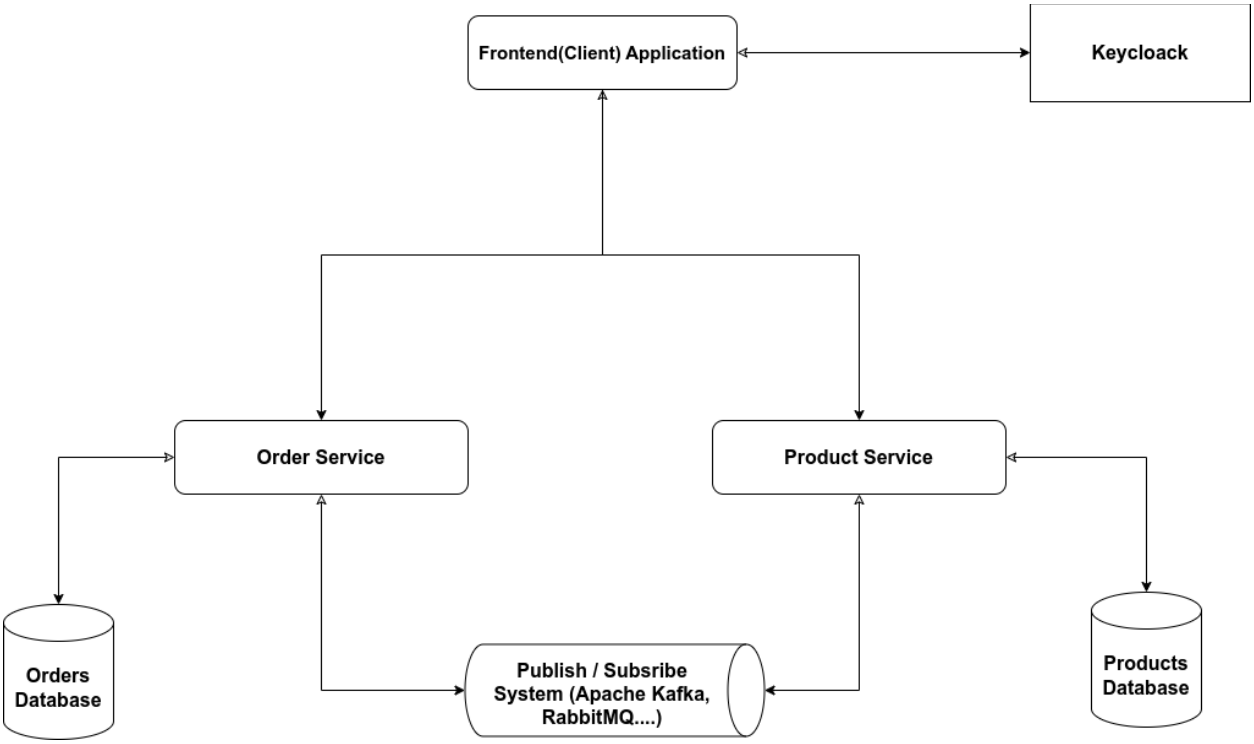
Εργασία για το Μάθημα

Υπολογιστικό Νέφος και Ομίχλη ΠΛΗ513

Παράδοση 10 Ιανουαρίου

Email επικοινωνίας: alazidis@tuc.gr

Για το 2ο μέρος της εργασίας, θα χρησιμοποιήσετε το Apache Kafka και το Keycloak. Το Apache Kafka είναι ένα publish-subscribe σύστημα με το οποίο μπορείτε να στέλνεται μηνλυματα μεταξύ των υπηρεσιών. Αντίθετα το Keycloak είναι ένα σύστημα Authentication και Authorization. Επίσης, όλη η δουλειά σας θα γίνει στηθεί σε ένα Virtual Machine του Google Cloud Platform, όπου και από εκεί θα γίνει η διόρθωση της συνολικής σας δουλειάς. Η αρχιτεκτονική του eshop φαίνεται παρακάτω.



Περιγράφονται η εφαρμογή και τα services της εικόνας. Όλα τα services μπορούνε να επεκταθούνε με βάση τις προτιμήσεις σας:

1. **Keycloack**: Το Keycloack είναι ένα service που χρησιμοποιείται για Authentication και Authorization. Οπότε, θα το χρησιμοποιεί το Frontend για να κάνει ένας χρήστης Σύνδεση και Εγγραφή. Θα τρέξετε το Keycloack σε ένα Docker Container μαζί με μια βάση δεδομένων (MySQL, PostgreSQL κλπ.) της επιλογής σας. Έπειτα, θα δημιουργήσετε τους εξής ρόλους:
    1. seller (θα είναι οι πωλητές προϊόντων, που θα μπορούν να εισάγουν προϊόντα στο eshop)
    2. customer (θα είναι οι πελάτες που θα μπορούν να δούν και να παραγγείλουν τα προϊόντα)

Εσείς θα είστε ο admin.
Πληροφορίες για το Keycloak θα βρείτε στους παρακάτω συνδέσμους:

- https://www.appsdeveloperblog.com/keycloak-starting-standalone-server/
- https://www.appsdeveloperblog.com/keycloak-creating-a-new-user/
- https://wjw465150.gitbooks.io/keycloak-documentation/content/getting_started/index.html

2. **Frontend**: Αφού προσθέσετε το Keycloak και φτιάξετε σε αυτό τον Client, Realm κλπ. θα πρέπει να ακολουθήσετε τη παρακάτω λογική:

Ένας χρήστης με του που προσπαθήσει να μπει στην εφαρμογή θα τον ανακατευθύνει **στη Login σελίδα του Keycloak** όπου εκεί θα γίνεται η Σύνδεση και η Εγγραφή. (ή αλλιώς θα ανακατευθύνεται στη σελίδα με τα "Προϊόντα" χωρίς όμως να μπορεί να ολοκληρώσει κάποια παραγγελία προτού κάνει Σύνδεση και Εγγραφή).

**** Για τη σύνδεση και την εγγραφή ενός χρήστη στην εφαρμογή μας θα πρέπει να χρησιμοποιήσετε το rest api του Keycloak ή κάποιο framework ****

Ο χρήστης κατά την εγγραφή θα δίνει το email που θα πρέπει να είναι μοναδικό, username, ένα password και θα επιλέξει αν θέλει να είναι seller ή customer. Από το frontend κατά την εγγραφή θα στέλνεται ένα request στο Keycloack για την εγγραφή του χρήστη. Αν η εγγραφή γίνει με επιτυχία η εφαρμογή θα πρέπει να εμφανίζει είτε ένα μήνυμα επιτυχίας, είτε να ανακατευθύνει το χρήστης στη σελίδα Προϊόντα αν είναι customer , όπου ο χρήστης θα έχει κάνει ήδη σύνδεση.

Η σύνδεση του χρήστη θα γίνεται πάλι μέσω του Keycloak και θα δίνονται το username και το password. Με τη σύνδεσή του ο χρήστης θα ανακατευθύνεται στη σελίδα "Προϊόντα" αν είναι customer ή "Τα Προϊόντα Μου" αν είναι seller. Στο πλέον menu θα εμφανίζεται το username του χρήστη και θα του δίνεται και η επιλογή **Αποσύνδεσης**. Αν κάποιος προσπαθήσει να μπει σε κάποια σελίδα χωρίς να κάνει σύνδεση ή που δεν έχει πρόσβαση θα ανακατευθύνεται στη σελίδα Σύνδεσης.

Οι customers θα βλέπουν τις σελίδες "Προϊόντα", "Παραγγελίες", "Καλάθι".

Οι sellers θα βλέπουν τη σελίδα τα "Τα Προϊόντα μου"

3. **Product Service**: Αν ένας χρήστης προσπαθήσει να στείλει ένα request στο API θα ελέγχεται αρχικά αν έχει πρόσβαση σε αυτό και αν είναι συνδεδεμένος με ένα Token και στη συνέχεια θα του επιτρέπεται η ενέργεια. Αλλιώς θα επιστρέφεται ένα μήνυμα ότι ο χρήστης δεν έχει πρόσβαση στο συγκεκριμένο Service ή (προαιρετικά) αν το Token έχει λήξει θα τον ανακατευθύνει στη σελίδα της σύνδεσης. Επίσης, προαιρετικά αν θέλετε μπορείτε να κάνετε έναν έλεγχο από το Product Service με το Keycloak, το Token του χρήστη είναι valid.

Η βάση δεδομένων που θα επιλέξετε θα αποθηκεύει τα εξής στοιχεία:

1. **Id** (Ένας αριθμός που θα είναι μοναδικός για κάθε προϊόν)
2. **Title** (Όνομα του προϊόντος)

3. **Img** (Εικόνα του προϊόντος)
4. **Price** (Τιμή του προϊόντος)
5. **Quantity** (Ο αριθμός των προϊόντων που είναι διαθέσιμος)
6. **Username** (Το username του seller που δημιούργησε το προϊόν)

Είναι στην επιλογή σας να ονομάσετε όπως θέλετε τα παραπάνω στοιχεία και να προσθέσετε περισσότερα στοιχεία για κάθε προϊόν στη βάση (Δεν θα έχει σημασία στο βαθμό).

Θα πρέπει πλέον να προσαρμόσετε το API σας για τα Products, ώστε να μπορούν μόνο οι sellers να δημιουργούνε, να επεξεργάζονται και να διαγράφουν προϊόντα π.χ.

METHOD	Routes	Description
GET	/products/:id και /products/:title /products/:username	Το **id** είναι ο μοναδικός αριθμός κάθε προϊόντος. Επιστρέφει ένα ή περισσότερα προϊόντα με το συγκεκριμένο id. Για **title** επιστρέφονται όλα τα προϊόντα με το συγκεκριμένο όνομα. Για **username**, επιστρέφονται όλα τα προϊόντα του χρήστη με αυτό το username.

Order Service: Αν ένας χρήστης προσπαθήσει να στείλει ένα request στο API θα ελέγχεται αρχικά αν έχει πρόσβαση σε αυτό και αν είναι συνδεδεμένος με ένα Token και στη συνέχεια θα του επιτρέπεται η ενέργεια. Αλλιώς θα επιστρέφεται ένα μήνυμα ότι ο χρήστης δεν έχει πρόσβαση στο συγκεκριμένο Service ή (προαιρετικά) αν το Token έχει λήξει θα τον ανακατευθύνει στη σελίδα της σύνδεσης. Επίσης, προαιρετικά αν θέλετε μπορείτε να κάνετε έναν έλεγχο από το Order Service με το Keycloak, ότι το Token του χρήστη είναι valid.

Η βάση δεδομένων που θα επιλέξετε θα αποθηκεύει τα εξής στοιχεία:

1. **Id** (Ένας αριθμός που θα είναι μοναδικός για κάθε παραγγελία)
2. **Products**: [
   {
   “title_1”, (Όνομα προϊόντος)
   “amount”, (Πόσες φορές πρόσθεσε ο χρήστης το προϊόν με τίτλο title_1 στο καλάθι του)
   “product_id_1” (id προϊόντος)
   },{
   “title_2”, (Όνομα προϊόντος)
   “amount”, (Πόσες φορές πρόσθεσε ο χρήστης το προϊόν με τίτλο title_1 στο καλάθι του)
   “product_id_2” (id προϊόντος)

},…]
(Σε ένα array από objects θα αποθηκεύσετε όλα τα ονόματα και το id των προϊόντων που αγόρασε ο χρήστης)

3. **Total_price** (Πόσο κοστίζει συνολικά η αγορά του χρήστη)
4. **Status** (Το status θα έχει τιμές είτε *Pending*, είτε *Success*, είτε *Reject*. Η αρχική τιμή του status θα είναι *Pending*)
5. **username** (Το όνομα του customer που έκανε την αγορά)

Ένα παράδειγμα του API των orders περιγράφεται παρακάτω (Το API είναι ενδεικτικό και δεν είναι απαραίτητα αυτό που χρειάζεστε).

METHOD	Routes	Description
POST	/orders	Δημιουργία παραγγελίας από έναν customer.
GET	/orders	Επιστροφή όλων των παραγγελιών του χρήστη.
GET	/orders/:username	Αυτό αφορά μόνο τη περίπτωση χρήσης συστήματος ταυτοποίησης. Επιστροφή όλων των παραγγελιών ενός customer.

4. **Publish-Subscribe**: Το Publish-Subscribe σύστημα κάνει την επικοινωνία μεταξύ του *Product Service* και του *Order Service*. Σαν publish-subscribe συστημα χρησιμοποιείστε είτε το Kafka είτε το RabbitMQ. Το σενάριο είναι το εξής: Ας πούμε ότι ο customer με username John κάνει κάνει μια παραγγελία για ένα προϊόν με τίτλο «Nike Airmax». Μόλις ο John πατήσει να γίνει η παραγγελία για τα «Nike Airmax», αυτή θα σταλεί στο Order Service. Το Order Service θα ελέγξει αν ο John είναι customer και αν ναι τότε η παραγγελία του θα αποθηκευτεί στη βάση δεδομένων. Η παραγγελία του John έχει Status=“Pending”. Αφού η παραγγελία αποθηκευτεί στη βάση δεδομένων, στη συνέχεια στέλνεται στο Publish-Subscribe σύστημα ένα object με τα εξής στοιχεία της παραγγελίας:

```
{
  Id (Το id της παραγγελίας),
  Products: [{
    product_id (id προϊόντος),
    amount (ποσότητα προϊόντος)
  }]
}
```

Το Product Service με τη σειρά του θα διαβάσει από το Publish-Subscribe σύστημα τη παραπάνω παραγγελία και θα βρει το προϊόν στη βάση δεδομένων του. Θα δεί τον αριθμό του Quantity (Ποσότητα προϊόντος) και θα τον μειώσει με βάση τον αριθμό

amount, του προϊόντος. Στη περίπτωσή μας θα ενημερώσει τη ποσότητα για το προϊόν «Nike Airmax». Οπότε αν το Quantity για το Nike Airmax είναι 10 και το amount 2 τότε το νέο Quantity για το «Nike Airmax» θα είναι 8. Η βάση δεδομένων που συνδέεται με τα products θα ενημερώσει το Quantity για το συγκεκριμένο προϊόν και θα στείλει στο Publish-Subscribe σύστημα ένα μήνυμα (object) “Success” με το id της παραγγελίας. Αν το Quantity μετά την αφαίρεση με το amount είναι αρνητικός αριθμός τότε το Product Service δεν θα ενημερώσει τη βάση του, αλλά αντίθετα θα στείλει στο Publish-Subscribe σύστημα ένα μήνυμα “Reject” με το id της παραγγελίας.
Π.χ. {
Id (Id παραγγελίας),
Status: “Success” ή “Reject”
}

Το Order Service με τη σειρά του θα διαβάσει το μήνυμα από το publish-subscribe σύστημα και στη συνέχεια θα ενημερώσει το Status της παραγγελίας με το συγκεκριμένο id.

Παραδοτέα

Ένα zip αρχείο με όλο το κώδικα και μια αναφορά με το τι κάνατε και τι όχι και τι χρεάζεται να κάνω για να τρέξω την εφαρμογή. Όλη η εφαρμογή θα πρέπει να τρέχει μέσα από το **docker-compose.yml** αρχείο, όπου εκεί θα υπάρχουν όλες οι υπηρεσίες σε docker containers. Επίσης η δουλειά σας θα πρέπει να στηθεί σε ένα Virtual Machine του Google Cloud Platform. Στην αναφορά θα βάλετε και τις στατικές IP του Virtual Machine για το Frontend και το Keycloak. Πριν την διόρθωση της άσκησης θα ζητηθεί να ξεκινήσετε τα Virtual Machines.