HELLENIC REPUBLIC
National and Kapodistrian
University of Athens
EST. 1837

Εφαρμογή Διαδικτύου Ενοικίασης Δωματίων / Κατοικιών

Ομάδα Χρηστών: 125

Μεϊντάνης Σωτήριος	sdi1900234 (1115201900234)
Κωνσταντίνου Ιωάννης	sdi1700067 (1115201700067)

Εφαρμογή Διαδικτύου Ενοικίασης Δωματίων / Κατοικιών

Περιγραφή

Η εφαρμογή που αναπτύξαμε αποσκοπεί στη δημιουργία ενός χώρου όπου ενοικιαστές στην οποία μπορούν να αναζητήσουν κατοικίες προς ενοικίαση και οικοδεσπότες να προσφέρουν τις κατοικίες τους προς ενοικίαση. Στο παρακάτω PDF αναφέρουμε εν συντομία τις σχεδιαστικές μας αποφάσεις τις οποίες κάναμε, τις παραδοχές της υλοποίησης αλλά και τις δυσκολίες τις οποίες ήρθαμε αντιμέτωποι κατά την δημιουργία της εφαρμογής. Ακόμα παρουσιάζουμε τον τρόπο με τον οποίο θα γίνει η σωστή εγκατάσταση της εφαρμογής έτσι ώστε να τρέξει σε κάποιο μηχάνημα εκτός των δικών μας για να αξιολογηθεί η υλοποίηση της εφαρμογής. Για την υλοποίηση της εργασίας αυτής επιλέξαμε για back end framework το Quarkus, ένα σχετικά νέο framework της Java το οποίο συμπληρώνει την λειτουργικότητα της Java / Jakarta EE χωρίς την ανάγκη ύπαρξης ενός application server. Ταυτόχρονα για το front-end την Angular ενώ για βάση δεδομένων επιλέξαμε την MySQL. Τέλος για το ORM επιλέξαμε το JPA με Hibernate implementation και για το REST implementation επιλέξαμε RestEasy Classic. Για την υλοποίηση του JWT επιλέξαμε SmallRye library και δημιουργήσαμε custom implementation.

Απαιτήσεις

Για να εργαστείτε με αυτό το έργο, βεβαιωθείτε ότι έχετε τις ακόλουθες εκδόσεις λογισμικού:

- **Maven 3.8.2+** (Συνιστώμενη έκδοση: 3.9.3)
- **Java 17** (Open JDK + JRE)
- **Nginx** (v1.24.0)
- **Node.js** v18.18.0 και **npm** v9.8.1 (θα εγκατασταθούν μέσω εντολής Maven αργότερα)
- **MySQL** (Συνιστώμενη έκδοση: 8.0)

Ενότητες

Το project αποτελείται από 3 ενότητες:

- **Parent Module :** diamoni-plus-parent
- **Quarkus REST Backend Module :** diamoni-plus-rest-api
- **Angular UI Module :** diamoni-plus-angular

Κατασκευή του Project

Για να κατασκευάσετε το project και τις ενότητές του ταυτόχρονα, εκτελέστε τις ακόλουθες εντολές στον κατάλογο της γονικής ενότητας (parent module):

```
mvn clean install -pl :diamoni-plus-angular
```

Αυτή η εντολή θα ενεργοποιήσει τη διαδικασία κατασκευής και για την ενότητα Angular.

```
mvn clean package -pl :diamoni-plus-rest-api
```

Αυτή η εντολή θα ενεργοποιήσει τη διαδικασία κατασκευής για την ενότητα Quarkus REST backend.

Ρύθμιση του Project

Για να ρυθμίσετε το project, ακολουθήστε τα παρακάτω βήματα:

- Στον κατάλογο `diamoni-plus-angular/src/main/resources/diamoni-plus-angular/dist`, αντιγράψτε τον φάκελο `diamoni-plus-angular` και μετακινήστε τον στον κατάλογο χρήστη (π.χ., `C:\Users\<username>\diamoni-plus-angular`).

- Επίσης, εντός του φακέλου `diamoni-plus-rest-api/target/`, αντιγράψτε τον φάκελο `quarkus-app` και μετακινήστε τον στον κατάλογο χρήστη (π.χ., `C:\Users\<username>\quarkus-app`).
- Κατεβάστε το Nginx (για Windows, ο σύνδεσμος είναι εδώ: Nginx Download), αποσυμπιέστε τον συμπιεσμένο φάκελο και μετακινήστε τον στον κατάλογο χρήστη. Το μονοπάτι για το εκτελέσιμο του Nginx πρέπει να φαίνεται όπως εξής: `C:\Users\<username>\nginx-1.24.0\nginx-1.24.0`.
- Εντός του φακέλου Nginx, στον φάκελο `conf`, αντικαταστήστε το αρχείο `nginx.conf` με το αντίστοιχο από το έργο (στη διαδρομή `diamoni-plus-rest-api/src/main/resources/nginx.conf`) και αντικαταστήστε το `username` με το πραγματικό όνομα χρήστη (σε περίπτωση Windows). Για το Linux ή άλλα περιβάλλοντα, μπορείτε να χρησιμοποιήσετε προσαρμοσμένες διαδρομές και να τις διαμορφώσετε χειροκίνητα.
- Επίσης, εντός του φακέλου `diamoni-plus-rest-api/src/main/resources`, αντιγράψτε τα αρχεία `private-key.key` και `certificate.crt` και μετακινήστε τα στον φάκελο `C:/Users/<username>/nginx-1.24.0/nginx-1.24.0/ssl/` ή σε οποιαδήποτε άλλη διαδρομή επιθυμείτε, αλλά βεβαιωθείτε ότι ενημερώνετε το `nginx.conf` αναλόγως.

Δημιουργία του Διαχειριστή Χρήστη για το Quarkus για τη Διαχείριση της Βάσης Δεδομένων

Συνδεθείτε στην MySQL με τον ριζικό σας λογαριασμό και εκτελέστε τον κώδικα που περιέχεται για τη δημιουργία του διαχειριστή:
`diamoni-plus-rest-api/src/main/resources/db_admin.sql`

Μπορείτε να αλλάξετε το όνομα χρήστη και τον κωδικό όπως επιθυμείτε, αλλά πρέπει να τα αλλάξετε και στο αρχείο
`diamoni-plus-rest-api/src/main/resources/application.properties` στις παρακάτω ιδιότητες:

- `quarkus.datasource.username`
- `quarkus.datasource.password`

Δημιουργία της Βάσης Δεδομένων

Συνδεθείτε στην MySQL (με οποιονδήποτε λογαριασμό διαχειριστή) και εκτελέστε το σενάριο: `diamoni-plus-rest-api/src/main/resources/create_database.sql`

Εκτέλεση της Εφαρμογής Quarkus

Μεταβείτε στον φάκελο `quarkus-app` με το Command Prompt ή το τερματικό (αφού έχετε προηγουμένως διαμορφώσει το περιβάλλον σας, π.χ., `C:\Users\<username>\quarkus-app`) και εκτελέστε την ακόλουθη εντολή:

```
java -jar quarkus-run.jar
```

Αυτό θα ξεκινήσει όχι μόνο την εφαρμογή αλλά και αρχικά θα δημιουργήσει τους πίνακες της βάσης δεδομένων για την εφαρμογή.

Δημιουργία του Διαχειριστή Χρήστη για την Εφαρμογή

Αφού εκτελέσετε αρχικά την εφαρμογή Quarkus, συνδεθείτε στην MySQL (με οποιονδήποτε λογαριασμό διαχειριστή) και εκτελέστε το σενάριο: `diamoni-plus-rest-api/src/main/resources/application_admin.sql`

Αυτό θα δημιουργήσει τον διαχειριστή για την εφαρμογή με τα εξής στοιχεία:
username: admin
password: password1234 .

Εκτέλεση της Εφαρμογής Angular

Διότι η εφαρμογή Quarkus εκτελείται, μεταβείτε ξανά στον φάκελο του Nginx (μετά την διαμόρφωση που πραγματοποιήσατε στα προηγούμενα βήματα) και μέσα στο `C:\Users\<username>\nginx-1.24.0\nginx-1.24.0` στο Command Prompt ή το τερματικό, εκτελέστε την εντολή:

```
start nginx
```

Για να σταματήσετε τον διακομιστή, χρησιμοποιήστε:

```
nginx -s stop
```

Εγκατάσταση του Πιστοποιητικού στο Chrome (Για Ανάπτυξη)

1. Ανοίξτε τις Ρυθμίσεις του Chrome:

Πηγαίνετε στο `chrome://settings/`.

2. Πλοηγηθείτε στην Ενότητα "Απόρρητο και Ασφάλεια":

Κάτω από την επιλογή "Απόρρητο και Ασφάλεια," κλικ στην "Ασφάλεια."

3. Διαχείριση των Πιστοποιητικών:

Κύλιση προς τα κάτω στο τμήμα "Προχωρημένες ρυθμίσεις" και κλικ στο "Διαχείριση πιστοποιητικών."

4. Εισαγωγή του Πιστοποιητικού:

Μεταβείτε στην καρτέλα "Αρχές πιστοποίησης εμπιστοσύνης" και κάντε κλικ στο "Εισαγωγή."

Ακολουθήστε τον οδηγό για να εισάγετε το αρχείο `certificate.crt` που δημιουργήσατε νωρίτερα.

5. Επανεκκίνηση του Chrome:

Κλείστε όλα τα παράθυρα του Chrome και επαναλάβετε τον Chrome. Το αυτο-υπογεγραμμένο σας πιστοποιητικό θα πρέπει τώρα να είναι αξιόπιστο από το Chrome.

Παραδοχές

Για την εφαρμογή μας μια παροδοχή που έχουμε κάνει είναι ότι θα υπάρχει ένας ρόλος για κάθε χρήστη, δηλαδή κάθε χρήστης θα ανήκει σε μία από τις τέσσερις κατηγορίες είτε πάροχος του χώρου ενοικίασης, είτε ενοικιαστής είτε διαχειριστής της εφαρμογής είτε τέλος απλός επισκέπτης, ο οποίος δεν έχει συνδεθεί ακόμα στην εφαρμογή.

Στην προσπάθεια μας να κρατήσουμε όσο πιο αποδοτική γίνεται την εφαρμογή μας σκεφτήκαμε να χωρίσουμε τις πληροφορίες τις οποίες έχουμε με ανάλογη μορφή pagination. Με αυτόν τον τρόπο μειώνεται το απαιτούμενο εύρος ζώνης της https κλήσης καθώς και η πληθυκότητα των δεδομένων που καλείται η βάση.

Μια ακόμη παραδοχή που κάναμε είναι η ικανότητα του διαχειριστή να μπορεί να φιλτράρει τους χρήστες σύμφωνα με τα στοιχεία του προφίλ τους. Επιπλέον προσθέσαμε την ικανότητα του ενοικιαστή να μπορεί να ακυρώσει μια κράτηση, καθώς την δυνατότητα της διαγραφής εικόνων απο τον πάροχο κατα την τροποποίηση στοιχείων ενος χώρου ενοικίασης.

Τέλος μια αξιοσημείωτη παραδοχή που κάναμε είναι εκτός απο την χρήση του openstreet map για τον εντοπισμό της τοποθεσίας του χώρου προς ενοικίαση είναι και η χρήση του για τον προσδιορισμό των συντεταγμένων για ορισμένες τοποθεσίες που προσδίδουν οδηγίες στον ενοικιαστή για να μετακινηθεί στον χώρο ενοικίασης.

Υλοποιήσεις / Δυσκολίες

Επιλέξαμε για back end framework το Quarkus, ένα σχετικά νέο framework της Java το οποίο συμπληρώνει την λειτουργικότητα της Java / Jakarta EE χωρίς την ανάγκη ύπαρξης ενός application server γεγονός που επιταχύνει τον ρυθμό ανάπτυξης της εφαρμογής και επιπλέον προσδίδει αρκετά features όπως γρήγορο start up time και hot reloading, είναι φιλικό προς την παραγωγή τοπικού εκτελέσιμου σε σχέση με το περιβάλλον του λειτουργικού συστήματος καθώς επίσης προσαρμόζεται εύκολα στο cloud (πράγμα που μας ενδιαφέρει στα μελλοντικά μας σχέδια με την εφαρμογή).

Για το ORM επιλέξαμε το JPA με Hibernate implementation και για το REST implementation επιλέξαμε RestEasy Classic. Για την υλοποίηση του JWT επιλέξαμε SmallRye library και δημιουργήσαμε custom implementation (δεν χρησιμοποιήσαμε Keycloak ή κάποιο αντίστοιχο πάροχο).

Τέλος για βάση δεδομένων επιλέξαμε την MySQL ενώ για το front-end την Angular.

Αξίζει να σημειωθεί ότι οι πίνακες στην βάση δεδομένων παράγονται αυτόματα απο τα entities.

Αποφασίσαμε ότι θα υποστηρίζουμε την αποθήκευση των κωδικών των χρηστών κρυπτογραφιμένους μέσω Bcrypt καθώς και η αποθήκευση των εικόνων να γίνεται στην βάση (και όχι στο file system) με κωδικοποίηση και αποδικοποίηση base64. Είναι σημαντικό να τονίσουμε πως μονάχα μια εικόνα μπορεί να φορτώνεται μέσω του REST για ευνόητους λόγους (https bandwidth), συνεπώς είναι ευθύνη του πελάτη (Angular) να φορτώσει με πολλαπλές κλήσεις το σύνολο των εικόνων.

Οι απαρραίτητες https κλήσεις προς το REST γίνονται μέσω JWT και όπου χρειάζεται υπάρχει περιορισμός μέσω των ρόλων. Να αναφερθεί επίσης πως μέσα στο JWT υπάρχουν κρυπτογραφιμένες οι πληροφορίες του χρήστη και συγκεκριμένα το username του και ο ρόλος του. Είχαμε το εξής δίλημμα για το αν θα αποθηκεύουμε το JWT στον client μέσω cookie ή εαν θα το αποθηκεύουμε στο local storage του browser. Εν τέλη αποφασίσαμε να το αποθηκεύσουμε στο local storage του browser σκεπτόμενοι ότι δεν διαθέτουμε CSRF token αλλα και για ευνόητους λόγους.

Αξιοσημείωτο είναι επίσης ότι τα φίλτρα για τους χώρους ενοικίασης αλλα και για τους χρήστες είναι reactive που σημαίνει ότι με την παραμικρή προποποίηση γίνεται κλήση στην βάση με ανανεωμένα αποτελέσματα χωρίς να απαιτείται υποβολή της φόρμας των φίλτρων μέσω κουμπιού.

Όσον αφορά τις συνομιλίες μεταξύ παρόχων και ενοικιαστών αποφασίσαμε να χρησιμοποιήσουμε real time συστήματα και συγκεκριμένα web sockets σε συνδιασμό με https κλήσεις. Ένα ακόμα δίλημμα που αντιμετωπίσαμε είναι η αποθήκευση του νέου μηνύματος στην βάση δεδομένων και την αποστολή του. Οι τεχνικές που σκεφτήκαμε είναι τρείς. Η πρώτη είναι να είναι ευθύνη του πελάτη (Angular) κατα την αποστολή του μηνύματος να στέλνει το μήνυμα μέσω https προς αποθήκευση και ταυτοχρονα να αποστέλνει το μήνυμα μέσω web socket στους συνδεδεμένους χρήστες. Η δεύτερη τεχνική είναι καθώς το μήνυμα στέλνεται μέσω web socket κατα την διαχείρηση τους στο back-end να ανοίγει καινούργιο thread μέσω executor service το οποίο θα διαχειρίζεται την αποθήκευση του μηνύματος. Το προβλημά μας με αυτή την υλοποίηση ήταν οτι δεν πετύχενε μεγάλη απόδοση καθώς για πολλαπλά μηνύματα δημιουργούνταν και πολλά threads. Για αυτό καταλήξαμε πως η καλύτερη λύση (η τρίτη) είναι η αποθήκευση του μηνύματος στην βάση με ασύγχρονο τρόπο, με την χρήση Kafka ή ActiveMQ.

Προκειμένου να ελαττώσουμε τις απαιτήσεις της εφαρμογής για τον σκοπό της εργασίας υλοποιήσαμε την πρώτη τεχνική.

Σχετικά με την απεικόνιση των προγεννέστερων μηνυμάτων μας προβλημάτισε αρκετά πως είναι εφικτό να φορτώνονται όλα τα μηνύματα απο την βάση με μια κλήση. Για τον λόγο αυτό εμπνευσμένοι απο την στρατηγική της META (εφαρμογή Messenger) αποφασίσαμε να φορτώνουμε τα μηνύματα paginated, φορτώνοντας τα προηγούμενα “χ” κάθε φορά όταν το scroll bar φτάνει στο ανώτατο σημείο.

Για το front-end αποφασίσαμε επίσης να χρησιμοποιήσουμε reverse proxy και συγκεκριμένα τον Nginx προκειμένου να μην είναι πρόβλημα για τον πελάτη (Angular) να διαχειριστεί τα URLs για τις κλήσεις με το back-end προσδίδοντας ένα πιο configurable περιβάλλον το οποίο ευνοεί επίσης και στην υλοποίηση της κρυπτογράφησης των κλήσεων μέσω SSL/TLS.

Σχετικά με την κρυπτογράφηση μέσω SSL/TLS να αναφέρουμε ότι υποστηρίζεται τοσο για τον πελάτη (Angular) όσο και για τις κλήσεις στο REST και στο web socket.

Να αναφέρουμε πως το certificate που απαιτείται δεν είναι έμπιστο απο τον φυλλομετρητή (trusted) καθώς το δημιουργήσαμε εμείς οι ίδιοι.

Τέλος δύο μικρά προβλήματα που αντιμετωπίσαμε και δεν μπορέσαμε να επιλύσουμε είναι η απεικόνιση της πινέζας στο openstreet map και η απεικόνιση της ισχυρότητας του κωδικού όταν συμπληρώνεται στην φόρμα κατα την εγγραφή ή την σύνδεση στην εφαρμογή.