HELLENIC REPUBLIC
National and Kapodistrian
University of Athens
EST. 1837

"Getaroom"

Εργασία στο μάθημα

Τεχνολογίες Εφαρμογών Διαδικτύου

Καθηγητής: **Ιωάννης Χαμόδρακας**

Κοσμίδη Άννα: 1115202000093
Κυριάκου Λυδία: 111202000107

2023

Περιεχόμενα

1.
Πρόλογος

Η παρούσα εργασία αναπτύχθηκε στα πλαίσια του μαθήματος "Τεχνολογίες Εφαρμογών Διαδικτύου". Το "Get A Room", αποτελεί μία ιστοσελίδα στην οποία καθένας με την είσοδο του έχει την δυνατότητα να περιηγηθεί και να αναζητήσει κατοικίες προσαρμοσμένες στις προτιμήσεις του. Επιπλέον ιστοσελίδα προσφέρει στον εγγεγραμμένο χρήστη τη δυνατότητα ενοικίασης χώρου από εγκεκριμένους οικοδεσπότες, υποβολής κριτικών, συνομιλίας με τον οικοδεσπότη κάθε ενοικιαζόμενου χώρου καθώς και παροχής εξατομικευμένων προτάσεων. Οι εγκεκριμενοι οικοδεσπότες από την άλλη μπορούν να εισάγουν και να τροποποιήσουν τους χώρους προς ενοικίαση καθώς και να συνομιλήσουν με τους ενδιαφερόμενους ενοικιαστές.Τέλος ο διαχειριστής εγκρίνει τους οικοδεσπότες και μπορει να παράξει αρχεία της βάσης.

Η εργασία αποτελείται από δύο μέρη:

a) Το μετωπιαίο άκρο (front-end), το οποίο έχει υλοποιηθεί με Angular.
b) Το νωτιαίο άκρο (back-end), το οποίο έχει υλοποιηθεί με Spring-Boot
Σχετικά με την ασφάλεια την εφαρμογής έχει χρησιμοποιηθεί το Spring Security καθώς και η τεχνολογία JWT και το πρωτόκολλο SSL

2.
Οδηγίες – Εγκατάσταση

Η εγκατάσταση για την λειτουργία της παρούσας εργασία είναι πολύ απλή:

Βάση Δεδομένων:

Απαιτείται να γίνει import του αρχειου της βάσης δεδομένων από το My-Sql-Workbench.

Backend:

1) Λήψη του αρχείου και άνοιγμα του στο intellij
2) Αλλαγή των προσωπικών στοιχείων στο application properties αναφορικά με το url (όνομα βάσης), username, password
3) Εκτέλεση του προγράμματος μέσω Run newAirbnbApplication πάνω δεξιά .

Frontend:

1) Απαιτείται η angular 16 προκειμένου να τρέξει στο vs-code
2) Λήψη του αρχείου και άνοιγμα στο vs-code
3) Χρήση της εντολής ng serve για να εκτελεστεί το πρόγραμμα.

3.
Κύριο Μέρος

3.1 Μετωπιαίο άκρο

Το μετωπιαίο άκρο της ιστοσελίδας έχει υλοποιηθεί με Angular 16. Το app.component.html περιέχει το app header, το rout outlet και το app footer. Σε όλη την εφαρμογή εκτελείται πάντα το header και το footer το οποίο διαφοροποιείται ανάλογα με τον ρόλο του χρήστη. Το Angular Project αποτελείται από το όλα τα απαραίτητα components, τον φάκελο service που περιέχει όλα τα απαραίτητα service.ts αρχεία και τον φάκελο auth που περιέχει όλα τα απαραίτητα services για το authentication του χρηστη. Τέλος στον φάκελο assets εμπεριέχονται κάποιες φωτογραφίες της εφαρμογής και την γραμματοσειρα century-gothic.ttf η οποία τρέχει σε όλες τις σελίδες ως default.

Components

Το Angular Project έχει αρκετά components απαραίτητα για τις λειτουργίες της εφαρμογης. Οι σελίδες που υπάρχουν και είναι διαθέσιμες προς περιήγηση στην εφαρμογή είναι οι εξής:

```
{ path: '', component: HomeComponent },
{ path: 'register', component: RegisterComponent },
{ path: 'log-in', component: LogInComponent },
{ path: 'profile', component: ProfileComponent },
{ path: 'edit-profile', component: EditProfileComponent },
{ path: 'admin', component: AdminComponent },
{ path: 'export-data', component: ExportDataComponent },
{ path: 'user-details/:id', component: UserDetailsComponent },
{ path: 'reservations', component: ReservationsComponent },
{ path: 'rentals', component: RentalsComponent },
{ path: 'rentals/:id', component: RentalDetailsComponent },
{ path: 'edit-rental/:id', component: EditRentalComponent },
{ path: 'send-message/:host-id/:user-id', component: SentMessageComponent },
{ path: 'host', component: HostComponent },
{ path: 'add-rental', component: AddRentalComponent },
{ path: 'make-review/:id', component: MakeReviewComponent },
{ path: 'messages', component: MessagesComponent },
{ path: 'conversation/:id', component: ConversationComponent },
{ path: 'access-denied', component: AccessDeniedComponent },
```

Ο χρήστης κατά την είσοδό του στην ιστοσελίδα θα μπορεί να πλοηγηθεί στην σελίδα home η οποία περιέχει μια φόρμα αναζήτησης διαθέσιμος ενοικιαζόμενων. Στην ίδια σελίδα αν ο χρήστης είναι συνδεδεμένος με τον ρόλο user θα του εμφανίζει κάτω από την φόρμα τα προτεινόμενα ενοικιαζόμενα για εκείνον. Όταν ο χρήστης κάνει κάποια αναζήτηση η ιστοσελίδα θα μεταβεί στην σελίδα 'rentals' στην οποία θα του εμφανιστούν όλα τα ενοικιαζόμενα τα οποία είναι διαθέσιμα τις ημερομηνίες, στην περιοχή και για τον αριθμό των επισκεπτών που έχει επιλέξει.

Όταν στην εφαρμογή συνδεθεί ένας διαχειριστείς πλοηγείται κατευθείαν την σελίδα 'admin' στην οποία εμφανίζονται όλοι οι εγγεγραμμένοι χρήστες σελιδοποιημένοι ανά 10. Η σελιδοποίηση αυτή των χρηστών έγινε για την ευκολότερη πλοήγηση του διαχειριστή και για την γρηγορότερη εμφάνιση όλων των αποτελεσμάτων, καθώς για την εμφάνιση των χρηστών καλείται η συνάρτηση getSomeUsers(page: number, size: number)
η οποία κάνει αίτηση στο backend για επιστροφή των 10 χρηστων.

Επίσης είναι εφικτή η λήψη όλων των δεδομένων της εφαρμογής σε αρχεία JSON και XML μέσω της σελίδας 'export-data'. Αυτό επιτυγχάνεται καλώντας κάθε φορά την κατάλληλη συνάρτηση απο το Rental ή User Service, η οποία έπειτα από την αποστολη του https request θα μετατρέψει τα τα δεδομένα σε αρχεία μορφής JSON ή XML.

Ο οικοδεσπότης κατά την είσοδο login() του στην εφαρμογή στέλνεται το https request στο backend Αν είναι επιτυχές το αίτημα αυτό, συνδέεται στην εφαρμογή και έπειτα ελέγχεται από τα στοιχεία του αν είναι approved ως οικοδεσπότης από τον διαχειριστή. Στην περίπτωση πού δεν είναι καλείται η συνάρτηση logout() ώστε να διαγραφεί το token που αποθηκεύτηκε στο local storage και στέλνει alert στον χρήστη ότι εκκρεμεί η έγκριση της αίτησης του από τον διαχειριστή. Στην περίπτωση που έχει ήδη εγκριθεί από τον διαχειριστή μπορεί πλέον να πλοηγηθεί στην σελίδα 'host' στην οποία θα εμφανίζονται όλα τα ενοικιαζόμενα που εκείνος έχει βάλει προς ενοικίαση. Επιπλέον θα μπορεί μέσα από την σελίδα 'add-rental' να διαχειριστεί και να τροποποιήσει όλες τις πληροφορίες του κάθε ενοικιαζόμενου. Η διαδικασία αυτή επιτυγχάνεται με την συνάρτηση του rental. service.ts updateRental(id: number, updatedRental: Rental), η οποία στέλνει αίτημα στο backend με τις πληροφορίες του ενοικιαζόμενου που επιθυμεί.

Services

Στον φάκελο services υπάρχουν τρία απαραίτητα services:
user.service.ts
rental.service.ts
messages.service.ts

Το κάθε ένα περιέχει συναρτήσεις οι οποίες στέλνουν ένα http request στο backend και δέχονται στην επιστροφή ένα αποτέλεσμα. Αυτό επιτυγχάνεται με το private http: HttpClient.

Model

Σε όλο το εύρος της εργασίας χρησιμοποιούνται ειδικά φτιαγμένες κλάσης αντικειμένων, όπως για παράδειγμα user, rental. Οι κλάσεις αυτές έχουν οριστεί μέσα στον φάκελο model και γίνονται import σε όποιο component γίνεται η χρήση τους. Η κάθε κλάση περιέχει όλα τα απαραίτητα πεδία του κάθε αντικειμένου, για την ευκολότερη προσπέλαση τους μετέπειτα από τον πρόγραμμα.

3.2 JWT-Security

Στο παρακάτω τμήμα αναπτύσσεται ο τρόπος διεκπεραίωσης του security της εφαρμογής με την χρήση JWT τεχνολογίας και Spring-Security καθώς και το πώς υπάρχει ή ασφαλής ανταλλαγή δεδομένων μεταξύ frontend και backend.

Για την εργασία έχει χρησιμοποιηθεί το JWT. Το JWT (JSON Web Token) είναι ένα πρότυπο ανοικτού κώδικα που χρησιμοποιείται για τη διασύνδεση και την ασφαλή ανταλλαγή δεδομένων μεταξύ του μετωπιαίου και του νωτιαίου άκρου της ιστοσελίδας. Κάθε χρήστης με την εγγραφή του στην εφαρμογή εισάγεται στην βάση δεδομένων. Επομένως κάθε φορά που θα επιχειρεί την είσοδο του στην εφαρμογή(login) θα εκτελείται η διαδικασία για την αυθεντικοποίηση των στοιχείων του.

Στο frontend όταν πραγματοποιείται η διαδικασία του login στέλνεται ένα https request στο backend το οποίο εμπεριέχει το username και password του χρηστη. Σε επόμενο βήμα το backend ελέγχει τα δεδομένα του χρήστη (username, password) μέσω του authentication manager και περνά τον ρόλο του χρήστη για μελλοντικούς ελέγχους. Τέλος παράγει ενά JWT token το οποίο εμπεριέχει το username, user role, user id σε κρυπτογραφημένη μορφή.

Το token αυτό έπειτα από κάθε επιτυχημένη είσοδο (login) του χρήστη στην εφαρμογή αποθηκεύεται από το frontend στο local storage, ώστε να μπορεί να μπορεί να χρησιμοποιηθεί μετέπειτα για τη ταυτοποίηση του χρήστη. Αυτό συμβαίνει σε όλες τις προσωπικές σελίδες.

Κάθε χρήστης με βάση τον ρόλο του έχει πρόσβαση σε συγκεκριμένες σελίδες και ενέργειες. Κάθε φορά που ο χρήστης κάνει κάποια αίτηση για υπηρεσία, το frontend στέλνει ένα https request στο backend, μαζί με το token που έχει αποθηκευτεί στο local storage, το οποίο ελέγχει το παραπάνω με το annotation @Preauthorized(user_role). Σε περίπτωση που πραγματοποιηθεί το authentication, επιτρέπεται η πρόσβαση στην παρούσα υπηρεσία και αποστέλλεται το response του αιτήματος με επιτυχημένο status. Σε περίπτωση αποτυχίας επιστρέφει ότι ο χρήστης δεν έχει δικαίωμα στην συγκεκριμένη υπηρεσία λόγω του ρόλου του.

Τέλος στο frontend σε συγκεκριμένες σελίδες στις οποίες δεν πρέπει να έχουν πρόσβαση όλοι οι τύποι χρηστών, ελέγχεται το token από το local storage. Το token αυτό ή θα είναι null, δηλαδή ο χρήστης έχει τον ρόλο του anonymous user, ή θα εμπεριέχει τον ρόλο του συνδεδεμένου χρήστη. Ανάλογα με το αποτέλεσμα επιτρέπεται η πρόσβαση του χρήστη στην σελίδα ή δεν επιτρέπεται πηγαινει αυτόματα στην σελίδα access denied.

Για παράδειγμα ένας οικοδεσπότης δεν θα μπορεί να έχει πρόσβαση στην σελίδα του διαχειριστή, επειδή κατα την έναρξη της σελίδας του διαχειριστή ελέγχεται ο ρόλος του συνδεδεμένου χρήστη και δεν επιτρέπεται η πρόσβαση σε χρήστες που στο token έχουν ρόλο διαφορετικό από διαχειριστή.

3.1 Νωτιαίο άκρο

Το νωτιαίο άκρο (Backend) της εφαρμογής έχει υλοποιηθεί με SpringBoot. Σε κάθε αίτημα από το μετωπιαίο άκρο (Frontend) το back δέχεται ένα https αίτημα. Τα αιτήματα αυτά εκτελούνται από τον εκάστοτε controller. Αυτός αντιστοιχίζει κατάλληλα το αίτημα και επικοινωνεί με το service layer στο οποίο και το στέλνει. Αυτό με τη σειρά του επικοινωνεί με το αντίστοιχο repository. Τα repository στην πλειοψηφία τους επεκτείνουν το JPA repository το οποίο πραγματοποιεί τα ερωτήματα στη βάση. Υπάρχει και ένα custom repository(αφορά την οντότητα rentals) το οποίο εκτελεί πιο περίπλοκα sql ερωτήματα που αφορούν τον καθορισμό των φίλτρων αναζήτησης ενοικιαζόμενου. Γενικά μέσω του spring-boot υλοποιείται η λογική της πρόσβασης στη βάση δεδομένων που αντιστοιχίζεται με την κλάση μοντέλων μέσω του Java Persistence Library(JPA). Οι οντότητες που υπάρχουν στην βάση έχουν υλοποιηθεί μέσω των models τα οποία εμπεριέχουν τα στοιχεία καθεμιάς τους. Χάρη σε αυτά μπορεί να πραγματοποιηθεί κάθε φορά η σωστή εισαγωγή αντικειμένου σε μία οντότητα.

SSL protocol

Όπως προαναφέρθηκε παραπάνω το frontend στέλνει https αιτήματα. Αυτό συμβαίνει καθώς χρησιμοποιείται το πρωτόκολλο ssl(Secure Sockets Layer), το οποίο επιτυγχάνει την ασφαλή σύνδεση στο Διαδίκτυο με κρυπτογράφηση των δεδομένων που αποστέλλονται μεταξύ της ιστοσελίδας και του server.

Security

Επίσης έχουν αξιοποιηθεί οι υπηρεσίες του spring security οι οποίες έχουν αναλυθεί παραπάνω συνδυαστικά με το frontend. Επιπλέον έχει χρησιμοποιηθεί το password encoding το οποίο χρησιμοποιείται κατά την εγγραφή ενός χρήστη στη εφαρμογή, κρυπτογραφώντας το password του ώστε να είναι ασφαλές σαν δεδομένο στη βάση.

Σύστημα συστάσεων

Τέλος σημαντικό κομμάτι της εργασίας αποτελεί το σύστημα συστάσεων, το οποίο παρέχει εξατομικευμένες προτάσεις σε κάθε χρήστη με βάση κριτικές που έχει γράψει, κρατήσεις που έχει κάνει και καταλύματα τα οποία έχει επιλέξει να δει. Για την υλοποίηση του συστήματος αυτού έχει χρησιμοποιηθεί ο αλγόριθμος matrix factorization. Δημιουργείται αρχικά ένας δισδιάστατος πίνακας του οποίου οι στήλες αντιπροσωπεύουν τους χρήστες και οι γραμμές τους χώρους προς ενοικίαση. Έτσι κάθε κελί αποτυπώνει κατα πόσο ταιριαστό είναι με βάση τα κριτήρια που αναφέρθηκαν παραπάνω. Σε κάθε κριτήριο προστίθεται ένα βάρος το οποίο αποτυπώνει και την σημαντικότητά του, αθροίζεται με τα υπόλοιπα και έτσι προκύπτουν τα νούμερα στα κελιά. Όσο πιο μεγάλος ο αριθμός στο κελί τοσο πιο "ταιριαστό" θα είναι για τον χρήστη το κατάλυμα. Έπειτα καλείται ο αλγόριθμος και γεμίζει τα κενά κελιά που έχουν απομείνει. Όταν ένας χρήστης θα μεταβεί στο home page θα του εμφανιστουν οι προτάσεις του οι οποίες προέκυψαν τοποθετώντας τα δεδομένα της γραμμής του, που προέκυψαν από την παραπάνω διαδικασία, σε φθίνουσα σειρά(μεγαλύτερος αριθμός = μεγαλύτερη συνάφεια). Έχει γίνει η παραδοχή να εμφανίζονται τα 10 με τον μεγαλύτερο βαθμό, για λόγους συνάφειας.

4.
Επίλογος

Η εργασία "Getaroom" αποτελεί ένα full-stack project, καθώς όπως προαναφέρθηκε απαρτίζεται από δύο μέρη: το μετωπιαίο και το νωτιαίο άκρο. Αποτελεί την παραδοτέα εργασία στο μάθημα Τεχνολογίες Εφαρμογών Διαδικτύου. Η εργασία σε συνδυασμό με το μάθημα αποτέλεσε ευκαιρία απόκτησης-οικείωσης πλήθους γνώσεων ώστε να παραχθεί μία ολοκληρωμένη εφαρμογή. Προφανώς καθώς αποτελεί την πρώτη μας απόπειρα εφαρμογής, υπήρξαν διάφορες δυσκολίες σε αρκετά στάδια της δημιουργίας της. Παρόλα αυτά ξεπεράστηκαν και έτσι προέκυψε το τελικό αποτέλεσμα.

HELLENIC REPUBLIC
National and Kapodistrian
University of Athens
EST. 1837

"Getaroom"

Εργασία στο μάθημα

Τεχνολογίες Εφαρμογών Διαδικτύου

Καθηγητής: **Ιωάννης Χαμόδρακας**

Κοσμίδη Άννα: 1115202000093
Κυριάκου Λυδία: 111202000107

2023