Τεχνολογίες Εφαρμογών Διαδικτύου
Υποχρεωτική Εργασία – 2019

Μαραγκοζάκης Γεώργιος – 1115201500089
Μαστοράκης Ανδρέας Ευστάθιος – 111520150092

Περιεχόμενα

Τεχνολογίες

Για την εκπόνηση της εργασίας επιλέξαμε να βασιστούμε σε Vue.js για το front end, Express.js για το backend (με sequelize.js για ORM μοντελοποίηση) και MySQL ως βάση δεδομένων.

Η επιλογή της Vue.js έγινε καθώς πρόκειται για framework παρόμοιο της Angular, αλλά λίγο πιο εύκολο για κάποιον που δεν έχει εμπειρία με σχετικά frameworks. Για το backend, επιλέξαμε Express καθώς είναι από τα πιο διαδεδομένα javascript server και υποστήριζε ORM μοντελοποίηση μέσω sequelize. Όσον αφορά τη βάση, επιλέξαμε MySQL καθώς είχαμε και οι δύο εμπειρία με τη συγκεκριμένη τεχνολογία.

Οδηγίες εγκατάστασης και εκτέλεσης

Για την εκτέλεση της εργασίας πρέπει να εκτελεστεί το frontend (Vue.js), backend (Express.js) και να σεταριστεί μία MySQL βάση δεδομένων.

Γενικά απαιτείται η ύπαρξη node/npm και nodemon για auto-restart στο backend

Vue.js:

Μεταφερόμαστε το φάκελο της Vue και εκτελούμε npm install. Στη συνέχεια μπορούμε με npm run serve να εκτελέσουμε το project, το οποίο θα τρέξει στην port 8080. Στο αρχείο store.js ορίζουμε την IP και την port στην οποία βρίσκεται το backend.

Express.js

Ομοίως αρκεί να εκτελέσουμε npm install και npm install nodemon (npm install -g nodemon, αν δεν παίξει το πρώτο) στο directory του backend και στη συνέχεια με nodemon app.js μπορούμε να εκτελέσουμε την εφαρμογή. Η port στην οποία τρέχει το backend μπορεί να οριστεί στο app.js. Η IP, η port και τα credentials για την σύνδεση στη βάση μπορούν να οριστούν στο αρχείο config/database.js.

MySQL:

Για το σετάρισμα της βάσης παρέχουμε τόσο το create script όσο και τα .sql αρχεία με τα data του κάθε table.

FrontEnd

Το FrontEnd, όπως ειπώθηκε παραπάνω, είναι υλοποιημένο με το javascript framework Vue.js. Η λογική πίσω από τη χρήση του συγκεκριμένου, είναι το χτίσιμο μιας σελίδας η οποία αλλάζει δυναμικά, φορτώνοντας τα αντίστοιχα αρχεία-”σελίδες” τα οποία έχουν γίνει import στο router.js, και περιέχονται στο φάκελο views.

Αναλυτικότερα, στην αρχική σελίδα Home, έχουν προσαρμοσθεί οι λειτουργίες για την επισκόπηση των auctions τα οποία είναι ενεργά, η δυνατότητα περιορισμού των εμφανισθέντων auctions μέσω του μηχανισμού της επιλογής categories καθώς και τα πεδία στοχευμένης αναζήτησης, βάσει των απαιτήσεων της εκφώνησης.

Επιπλέον, όταν ο χρήστης εισέρχεται στην εφαρμογή ως guest αποκρύπτονται συγκεκριμένα κουμπιά στο SideBar όπως τα, New Auction, My Auctions, My Messages, τα οποία αφορούν μόνο τους συνδεδεμένους χρήστες. Όσον αφορά τα auctions, επίσης ο χρήστης ως guest μπορεί να πλοηγηθεί σε αυτά, να δει πληροφορίες για το καθένα, αλλά δεν έχει τη δυνατότητα να κάνει προσφορά ή άμεση αγορά.

BackEnd

Το BackEnd είναι υλοποιημένο, όπως ειπώθηκε παραπάνω στο περιβάλλον Node, με τη χρήση των frameworks express.js και sequelize.js, για την υποστήριξη του orm rest api.

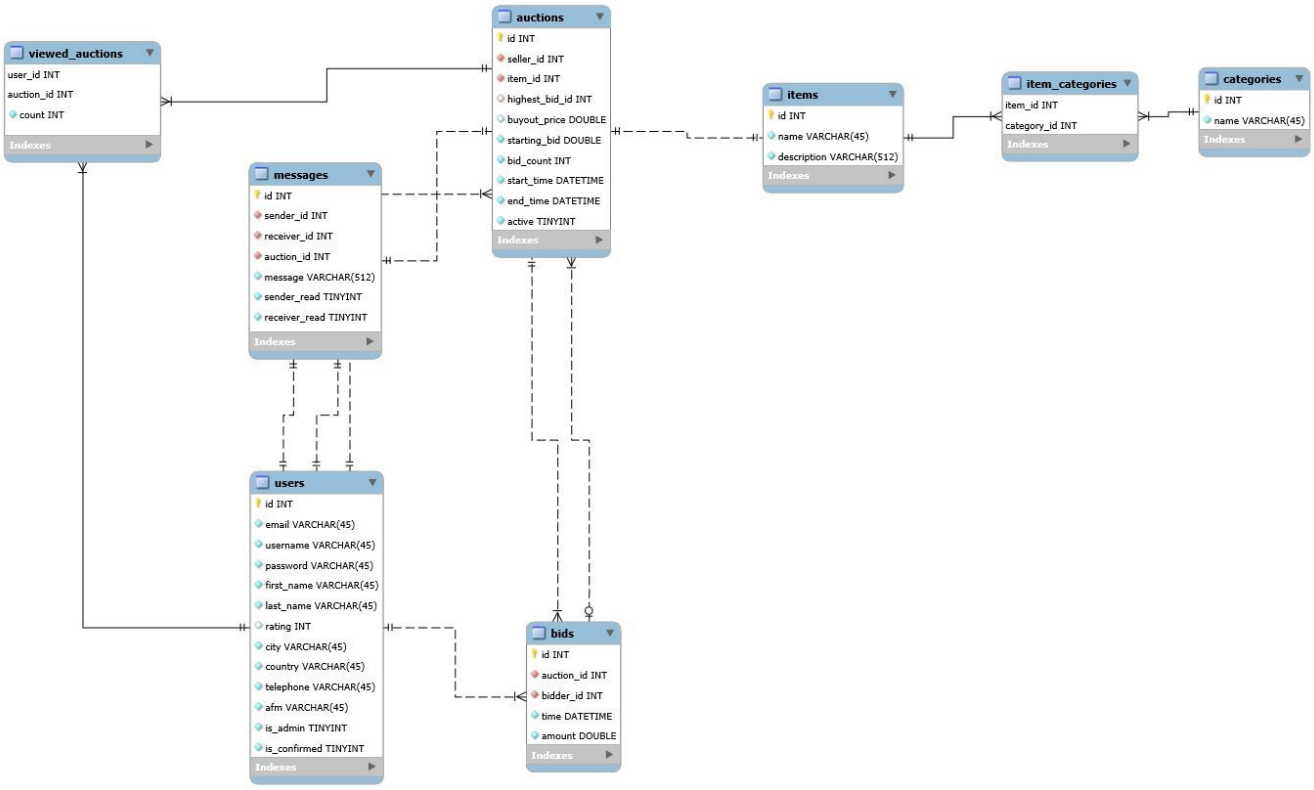
Στο φάκελο routes, υπάρχουν τα αρχεία τα οποία χρησιμεύουν για την ικανοποίηση των αιτημάτων get, put, patch, delete, από το frontEnd. Κάθε αρχείο, ανάλογα το όνομα του ικανοποιεί αιτήματα για συγκεκριμένες οντότητες της βάσης. Επιπλέον, για αιτήματα που απαιτούν πλήθος διαφορετικών δεδομένων, το αποτέλεσμα τους είναι βάσει του αντικειμενοστραφούς μοντέλου.

Σε ότι αφορά τις εικόνες, στην εφαρμογή μας αποθηκεύονται σε ένα φάκελο στο backEnd στη μορφή .json.

Η εφαρμογή χρησιμοποιεί JSON Web Token για το authentication χρηστών. Συγκεκριμένα, όταν ο χρήστης κάνει login παράγεται βάσει των στοιχείων του ένα token το οποίο και επιστρέφεται στο front end, όπου και αποθηκεύεται. Στη συνέχεια, για κάθε λειτουργία που απαιτείται εξακρίβωση χρήστη το backend θα χρησιμοποιήσει το token για να επιβεβαιώσει την ταυτότητα του αιτούντος χρήστη.

Όσον αφορά την μοντελοποίηση ORM, με χρήση sequelize-auto παράγουμε αυτομάτως μοντέλα για τους πίνακες της βάσης και με κάποιες μικρές αλλαγές τα χρησιμοποιούμε στην sequelize για να παίρνουμε ή να ενημερώνουμε δεδομένα από και προς τη βάση.

Βάση



Στην παραπάνω απεικόνιση είναι η βάση σε mysql, η οποία κατασκευάστηκε για τις ανάγκες της εργασίας. Οι συσχετίσεις μεταξύ των οντοτήτων είναι οι ενδεδειγμένες βάσει της εκφώνησης. Στην βάση δεν περιέχονται εικόνες, αλλά είναι αποθηκευμένες σε ένα φάκελο στο backend, από το οποίο μέσω του server στέλνονται στο μπροστά μέρος της διεπαφής.

Σημείωση: Δε φαίνεται καλά στην παραπάνω οθόνη, αλλά ο πίνακας users συνδέεται με σχέση 1-n με τον πίνακα auctions, δηλαδή κάθε auction έχει δημιουργηθεί από ένα χρήστη και κάθε χρήστης μπορεί να δημιουργεί πολλά auctions.

Σχόλια

1. Όταν ένας χρήστης είναι συνδεδεμένος στην εφαρμογή, και επιλέξει να πλοηγηθεί στη σελίδα My Messages, τότε έχει τις εξής δυνατότητες. Αρχικά, αν είναι seller μπορεί να στείλει πρώτος μήνυμα στον αγοραστή(buyout guy, ή highest bidder) μετά την ολοκλήρωση του auction, ενώ ο bidder δεν μπορεί μέχρι να λάβει μήνυμα από τον πρώτο. Αυτή είναι μια παραδοχή στην εφαρμογής μας, για να αποφευχθεί το spamming των bidders προς τους sellers.
2. Όταν σε ένα auction, έχει περάσει το ending_time, τότε το auction, γίνεται ανενεργό και επιλέγεται ο highest_bidder μέχρι εκείνη τη στιγμή ώστε να έρθει σε επικοινωνία μαζί του ο seller. Αντίστοιχα για το buy Now.
3. Όταν δημιουργείται ένα νέο auction, υπάρχει ο περιορισμός ότι η εικόνα που αφορά το item δεν μπορεί να είναι πάνω από 50kb περίπου.
4. Στην εφαρμογή, δεν έχουν υλοποιηθεί:
    1. μηχανισμός, export to xml, εξαγωγής δεδομένων των χρηστών από τον διαχειριστή
    2. rating χρηστών
    3. χάρτης OpenStreetMap
    4. διαγραφή auction

Bonus

Για το bonus αρχικά κάναμε μία επεξεργασία στο παρεχόμενο dataset, ώστε να απομονώσουμε τις χρήσιμες για εμάς πληροφορίες (τα σχετικά script που χρησιμοποιήθηκαν παρέχονται). Συγκεκριμένα κρατάμε για κάθε χρήστη τον αριθμό των bids που έχει κάνει για κάθε κατηγορία. Στη συνέχεια χρησιμοποιούμε αυτή τη γνώση για να του παρέχουμε προτάσεις για νέες αγγελίες.

Συγκεκριμένα για κάθε χρήστη που χρειάζεται recommendation βρίσκουμε τους χρήστες με τους οποίους έχει παρόμοιες προτιμήσεις στις κατηγορίες (με χρήση KNN και μίας συνάρτησης απόστασης που περιγράφεται παρακάτω). Αφότου βρεθούν οι πιο κοντινοί του γείτονες, θα γίνουν στο χρήστη-στόχο προτάσεις για τις κατηγορίες που είναι πιο δημοφιλείς ανάμεσα στους γείτονες του.

Συνάρτηση Απόφασης:

Δεδομένου ενός χρήστη για τον οποίο θέλουμε να παράξουμε recommendation, θα εκτελεστεί η συνάρτηση απόφασης που θα αναθέτει μία τιμή σε κάθε άλλο χρήστη. Η τιμή αυτή εξαρτάται από το πόσα κοινά categories έχουν οι δύο χρήστες και από το πόσο δημοφιλή είναι για τον καθένα τα κοινά αυτά categories. Έτσι για κάθε χρήστη έχουμε μία τιμή απόστασης οπότε μπορούμε να λάβουμε υπόψη τους k (k=3) κοντινότερους γείτονες και να προτείνουμε στη χρήστη τις δημοφιλέστερες για αυτούς κατηγορίες.