ΟΙΚΟΝΟΜΙΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΚΑΤΑΝΕΜΗΜΕΝΑ ΣΥΣΤΗΜΑΤΑ

Εαρινό Εξάμηνο 2023-2024
Υποχρεωτική εργασία

Τα τελευταία χρόνια έχει παρατηρηθεί μεγάλη αύξηση των εφαρμογών ενοικίασης δωματίων. Αυτά τα συστήματα συνήθως αποτελούνται από μια mobile frontend εφαρμογή για την διαχείριση των καταλυμάτων από τους χρήστες (προβολή, ενοικίαση και προσθήκη καταλυμάτων) και ένα backend σύστημα το οποίο αναλαμβάνει την ανάλυση, επεξεργασία και αποθήκευση των δεδομένων. Γνωστές υπηρεσίες που υπάρχουν διαθέσιμες online είναι π.χ. το Airbnb και το Booking. Στα πλαίσια της εργασίας του μαθήματος καλείστε να δημιουργήσετε ένα απλό τέτοιο σύστημα διαχείρισης καταλυμάτων στο οποίο οι χρήστες θα μπορούν να εκτελούν λειτουργίες manager και ενοικιαστή:

Στην λειτουργία manager θα πρέπει να μπορούν:
- Να προσθέτουν καταλύματα
- Να προσθέτουν τις διαθέσιμες ημερομηνίες προς ενοικίαση για αυτά τα καταλύματα
- Να εμφανίζουν τις κρατήσεις για τα καταλύματα που έχουν στην ιδιοκτησία τους.
- Η διαχείριση μπορεί να γίνεται μέσω console application

Στη λειτουργία ενοικιαστή θα πρέπει να μπορούν:
- Να φιλτράρουν τα καταλύματα ανάλογα
    - με την περιοχή που τους ενδιαφέρει
    - τις κατάλληλες ημερομηνίες που επιθυμούν
    - πλήθος ατόμων
    - τιμή
    - αστέρια
- Να πραγματοποιούν κράτηση για το κατάλυμα που τους ενδιαφέρει
- Να βαθμολογούν το δωμάτιο με αστέρια (1-5)
- Μέσω ενός UI σε Android να μπορεί να πραγματοποιήσει τις παραπάνω ενέργειες
    - Μια συνάρτηση search() θα στέλνει το φίλτρο στον Master ασύγχρονα και θα εμφανίζει στην οθόνη της εφαρμογής τα αποτελέσματα της αναζήτησης.
    - Μέσω μιας συνάρτησης book() θα μπορεί ο χρήστης να πραγματοποιήσει κράτηση για ένα κατάλυμα από τα καταλύματα που έχει επιστρέψει η search().

Προκειμένου το σύστημα να μπορεί να διαχειριστεί τον όγκο των δεδομένων θα πρέπει να μπορεί να τρέξει κατανεμημένα πάνω από ένα σύνολο μηχανημάτων.

Το MapReduce framework είναι ένα προγραμματιστικό μοντέλο που επιτρέπει την παράλληλη επεξεργασία μεγάλων όγκων δεδομένων.

Το MapReduce στηρίζεται στη χρήση δύο συναρτήσεων:

map(key,value) -> [(key2, value2)]

reduce(key2,[value2]) -> [final_value]

- "Map" συνάρτηση: επεξεργάζεται ένα ζεύγος key/value και παράγει ένα ενδιάμεσο ζεύγος key/value. Η είσοδος στη συνάρτηση map μπορεί να είναι γραμμές ενός αρχείου κλπ., και έχουν τη μορφή (κλειδί, τιμή). Η συνάρτηση map μετατρέπει κάθε τέτοιο ζευγάρι σε ένα άλλο ζευγάρι (κλειδί2, τιμή2). Η map συνάρτηση μπορεί να εκτελείται παράλληλα, πάνω σε διαφορετική είσοδο δεδομένων και σε διαφορετικούς κόμβους. Ο βαθμός παραλληλίας εξαρτάται από την εφαρμογή και μπορεί να την ορίσει ο χρήστης.
- "Reduce" συνάρτηση: συγχωνεύει όλα τα ενδιάμεσα values που σχετίζονται με το ίδιο κλειδί και παράγει τα τελικά αποτελέσματα. Για κάθε ξεχωριστό κλειδί δημιουργείται μια λίστα από τις τιμές που αντιστοιχούν σε αυτό το κλειδί. Η συνάρτηση αυτή υπολογίζει μια τελική τιμή για το κλειδί, επεξεργάζοντας τη λίστα των τιμών που αντιστοιχούν σε αυτό το κλειδί. Η επεξεργασία της συνάρτησης reduce γίνεται αφού έχει τελειώσει η επεξεργασία όλων των map συναρτήσεων.

Αρχικά όταν ένας ιδιοκτήτης επιθυμεί να προσθέσει ένα δωμάτιο στην πλατφόρμα, θα πρέπει να δώσει την περιγραφή του δωματίου με μορφή αρχείου json, όπως επισυνάπτεται, όπως επίσης και μια φωτογραφία. Μπορείτε να έχετε ένα φάκελο που να περιέχει το json και τη φωτογραφία. Το path της φωτογραφίας θα περιέχεται μέσα στο json. Το json επίσης θα περιέχει ένα μετρητή από review και τη βαθμολογία του καταλύματος (1-5). Επίσης θα πρέπει να παρέχεται τρόπος ώστε να μπορούν να δηλωθούν οι διαθέσιμες ημερομηνίες προς ενοικίαση του καταλύματος.

Η αρχική επικοινωνία του console app του manager θα πρέπει να γίνεται με τον Master μέσω της οποίας θα αποστέλλονται όλα τα στοιχεία του καταλύματος. Ο Master αφού λάβει τα στοιχεία, μέσω μιας hash συνάρτησης H(roomName) θα πρέπει να επιλέξει τον worker node στον οποίο θα αποθηκευτεί το κατάλυμα. Π.χ. NodeId = H(roomName) mod NumberOfNodes. Αφού επιλέξει τον κόμβο αποστέλλει τις πληροφορίες σε αυτόν. **Για τα πλαίσια της εργασίας ο Worker θα αποθηκεύει τις πληροφορίες στις κατάλληλες δομές δεδομένων στην μνήμη του. *Δεν επιτρέπεται η αποθήκευση στον δίσκο*** (εκτός από τις φωτογραφίες αν επιθυμείτε).

Όταν ένας χρήστης επιθυμεί να νοικιάσει ένα κατάλυμα αρχικά μέσω της εφαρμογής θα πρέπει να επιλέξει τα κατάλληλα φίλτρα για το κατάλυμα που επιθυμεί. Τότε αποστέλλεται ένα request προς τον Master που περιέχει τα φίλτρα. **Ο Master θα πρέπει με διαδικασία MapReduce να επιστρέψει στον χρήστη τα καταλύματα που ικανοποιούν τα κριτήρια.**

Αφού προβληθούν στον χρήστη, εκείνος θα μπορεί να δει αναλυτικά τα στοιχεία και τη φωτογραφία του δωματίου και να προβεί σε κράτηση εφόσον επιθυμεί.

Όταν ο χρήστης πραγματοποιήσει κράτηση, στέλνει ένα request κράτησης στον Master για το συγκεκριμένο κατάλυμα και ο Master με τη σειρά του πρέπει να ενημερώσει κατάλληλα τον Worker που διαχειρίζεται το κατάλυμα για την κράτηση. **Προσοχή θα πρέπει να ληφθούν μέτρα ώστε να μην μπορεί να συμβεί ταυτόχρονη κράτηση στο ίδιο κατάλυμα για τις ίδιες ημερομηνίες.**

```
{
   "roomName": "roomName",
   "noOfPersons": 5,
   "area": "Area1",
   "stars": 3,
   "noOfReviews": 15,
   "roomImage": "/usr/bin/images/roomName.png"
}
```


Τέλος στην λειτουργία manager, θα πρέπει να μπορούν να εμφανιστούν οι συνολικές κρατήσεις ανα περιοχή για ένα συγκεκριμένο χρονικό διάστημα που θα δίνει ο χρήστης.

Π.χ με input : 01/02/2024 - 01/04/2024
output: area1: 100
area2: 50
…

Απαιτήσεις υλοποίησης Backend:

- Ο **Master** πρέπει να υλοποιηθεί σε **Java** και να υλοποιεί **TCP Server**. **Δεν επιτρέπεται** η χρήση έτοιμων libraries πέρα των default ServerSocket της Java ή HTTP πρωτοκόλλου με τη χρήση έτοιμου server, όπως της Java ή Apache.

- Ο **Master** πρέπει να είναι **πολυνηματικός** και να μπορεί να εξυπηρετεί πολλούς χρήστες ταυτόχρονα και να επικοινωνεί ταυτόχρονα με τους workers.

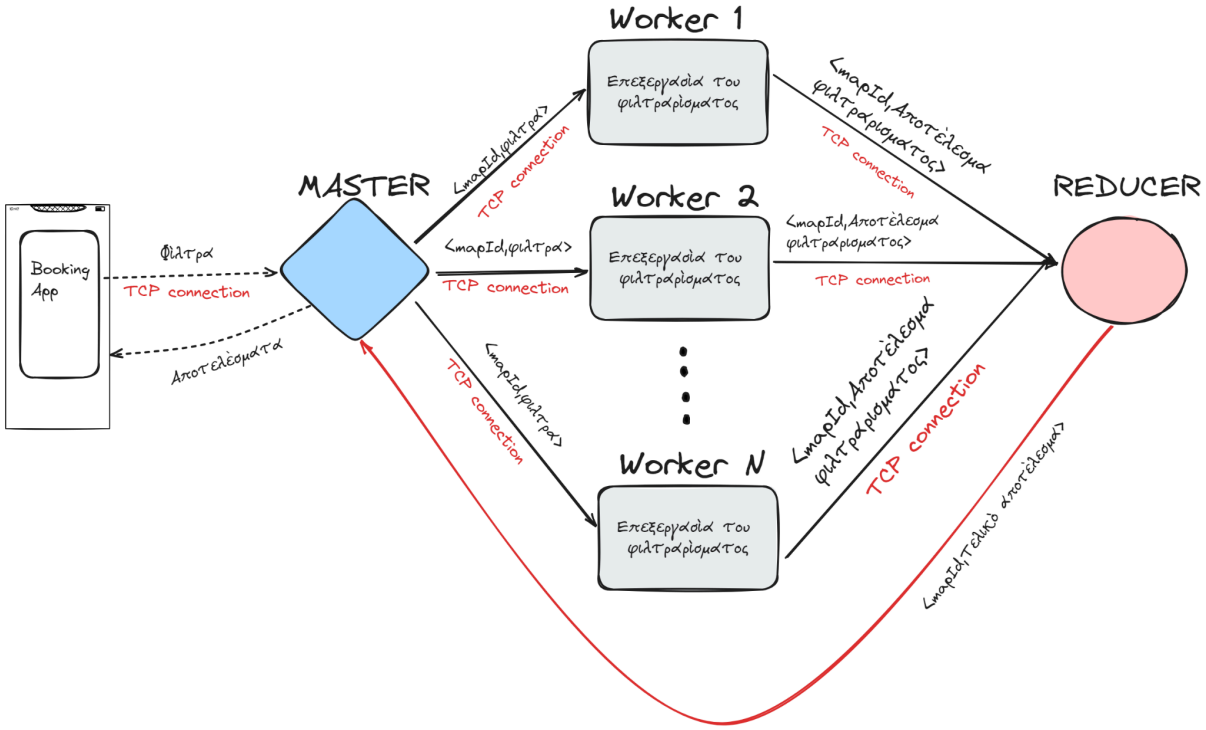
- Οι **Workers** πρέπει να υλοποιηθούν σε **Java** και να είναι **πολυνηματικοί** για να εκτελούν παράλληλα πολλά requests από τον Master.

- Οι **Workers** θα πρέπει να ορίζονται δυναμικά κατά το initialization του Master (από τα arguments ή config file) και ο αριθμός τους θα μπορεί να είναι αυθαίρετος.

- Η επικοινωνία Master / Worker να υλοποιείται και αυτή **αποκλειστικά μέσω TCP sockets**.

- Πρέπει να υπάρχει συγχρονισμός στα σημεία που κρίνετε απαραίτητο. Ο συγχρονισμός πρέπει να γίνει **αποκλειστικά** χρησιμοποιώντας τεχνικές **synchronized, wait - notify** και **όχι με τη χρήση έτοιμων εργαλείων της βιβλιοθήκης *java.util.concurrent* ή άλλων έτοιμων εργαλείων.**

- Οι δομές δεδομένων πρέπει να είναι αποθηκευμένες στην μνήμη. **Απαγορεύεται η χρήση βάσης δεδομένων.**



Απαιτήσεις υλοποίησης Frontend:

Θα αναπτύξετε μια εφαρμογή που θα εκτελείται σε συσκευές με λειτουργικό Android και θα αποτελεί interface για το σύστημα. Μέσω αυτής ο χρήστης:

- Η επικοινωνία του Application με τον Master θα πρέπει να γίνεται αποκλειστικά με τη χρήση **TCP Sockets**. Το Application θα πρέπει **να συνδέεται με TCP Socket στον Master**. Μέσω αυτού του Socket γίνεται η αποστολή των φίλτρων και της αίτησης για κράτηση. Ο Master αποστέλλει πίσω τα αποτελέσματα της επεξεργασίας **μέσω του ίδιου Socket** το οποίο παραμένει ανοιχτό έως ότου αυτά ληφθούν. **Αυτή η διαδικασία θα πρέπει να υλοποιηθεί με τη χρήση Threads, ώστε η εφαρμογή να παραμένει διαδραστική μέχρι να ληφθούν τα αποτελέσματα**, **δηλαδή η αποστολή γίνεται ασύγχρονα**.

Bonus

Μια γνωστή τεχνική προκειμένου η πλατφόρμα να είναι ανθεκτική σε σφάλματα είναι το active replication. Σε αυτή την τεχνική τα δεδομένα των καταλυμάτων θα πρέπει να βρίσκονται ταυτόχρονα σε πολλαπλούς κόμβους και να παραμένουν απολύτως συγχρονισμένα. Σε περίπτωση που κάποιος κόμβος worker πέσει δεν θα πρέπει να χαθεί η πρόσβαση στα δεδομένα που διαχειρίζεται (θα πρέπει να γίνει δρομολόγηση του request στο

replica του). Επίσης τα back up δεδομένα θα πρέπει να χρησιμοποιούνται μόνο όταν υπάρχει σφάλμα στον κανονικό κόμβο.

Το Bonus μετράει +20% στον βαθμό της εργασίας με μέγιστο δυνατό βαθμό εργασίας 10 και είναι υποχρεωτικό για ομάδες των 4 ατόμων.

Παραδοτέα εργασίας

Το project θα παραδοθεί σε **δύο** φάσεις:

Παραδοτέο Α: (Ημερομηνία παράδοσης: 7/04/2024)

Στο παραδοτέο αυτό, θα πρέπει να έχετε ολοκληρώσει εντελώς το backend σύστημα και το manager console app, όπως ακριβώς σας έχει ζητηθεί, έτσι ώστε να μπορεί να χρησιμοποιηθεί στην επόμενη φάση της εργασίας του μαθήματος (προσθήκη δωματίου και διαθέσιμων ημερομηνιών για κρατήσεις, εμφάνιση κρατήσεων). Αντί για android application θα έχετε μια dummy εφαρμογή όπου θα στέλνει το τα φίλτρα για τα δωμάτια στον Master και θα λαμβάνει τα αντίστοιχα αποτελέσματα. Επίσης θα πρέπει ο χρήστης να μπορεί να κάνει κράτηση από την dummy εφαρμογή.

Παραδοτέο Β: (Ημερομηνία παράδοσης: 26/05/2024)

Το παραδοτέο αυτό αποτελεί το Android application, που περιγράφηκε παραπάνω. Στη φάση αυτή το σύστημα θα πρέπει να είναι πλήρως λειτουργικό και ολοκληρωμένο, με όλα τα components του να λειτουργούν σωστά. Σε αυτή τη φάση το manager console app θα πρέπει να μπορεί να εκτελέσει και το aggregation query για τις **συνολικές κρατήσεις ανα περιοχή για ένα συγκεκριμένο χρονικό διάστημα.**

Ομάδες: Όλοι οι φοιτητές θα πρέπει να σχηματίσουν ομάδες των τριών (3) ατόμων προκειμένου να εκπονήσουν την προγραμματιστική τους εργασία. Γλώσσα προγραμματισμού θα είναι η Java, στην οποία και θα παρέχεται υποστήριξη από τους βοηθούς του μαθήματος.

Αναφορές – Χρήσιμοι Σύνδεσμοι:

[1] https://docs.oracle.com/javase/8/docs/api/java/util/stream/package-summary.html
[2] Android. URL : http://code.google.com/android/
[3] Android SDK: http://developer.android.com/sdk/index.html
[4] Android Studio http://developer.android.com/sdk/index.html