ΑΝΑΦΟΡΑ ΕΡΓΑΣΙΑΣ – PROJECT

Μέλη της ομάδας :

* **ΔΕΛΗΜΠΑΛΤΑΔΑΚΗΣ ΓΡΗΓΟΡΙΟΣ**

**Α.Μ. : 1084647**

**Email: up1084647@upnet.gr**

* **ΔΙΑΣΑΚΟΣ ΔΑΜΙΑΝΟΣ**

**A.M. : 1084632**

**Email: up1084632@upnet.gr**

* **ΡΑΙΚΟΣ ΙΑΣΩΝ**

**A.M. : 1084552**

**Email: up1084552@upnet.gr**

Ακολουθούν τα διαγράμματα UML των κλάσεων.

**(Αν κάνετε zoom το Word, θα φανούν πιο καθαρά. Επίσης θα συμπεριλάβουμε ένα πλήρες αναλυτικό UML στο hosting site αρχείων , γιατί δεν γίνεται να «χωρέσει” στο Word το πλήρες UML. )**

**Graphical user interface

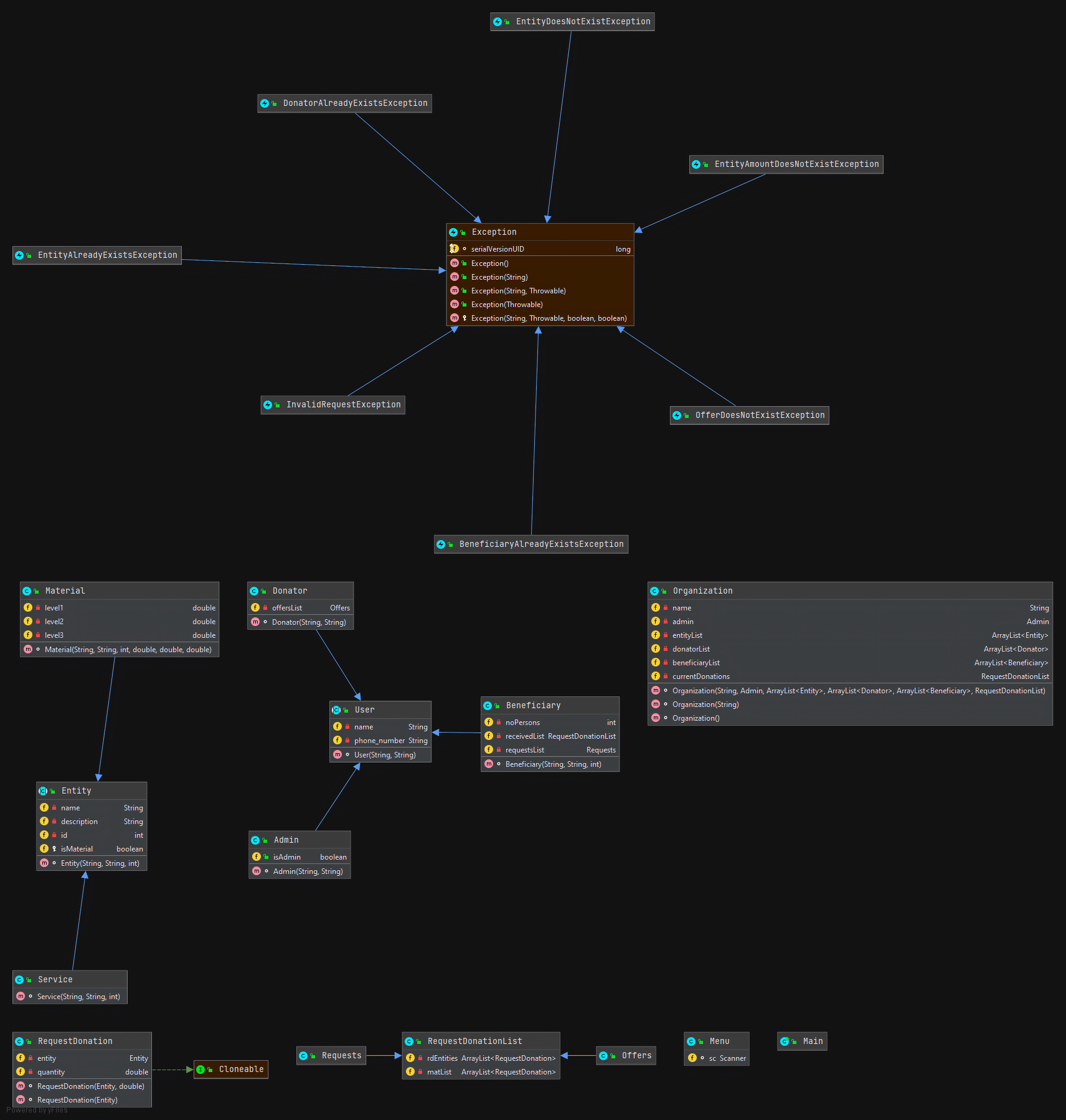
Description automatically generated**

**Graphical user interface

Description automatically generated**

**Graphical user interface

Description automatically generated**



**Σύντομη περιγραφή της υλοποίησης με τεκμηρίωση των σχεδιαστικών αποφάσεων και της υλοποίησης για κάθε κλάση.**

**1.User.java:**

Είναι η κλάση όπου αναφέρεται σε έναν χρήστη και έχει ως **private** πεδία το όνομα και το τηλέφωνο του χρήστη. Μπορούμε να τα αρχικοποιήσουμε μέσω του **constructor** και με τις **getter** μεθόδους υπάρχει η δυνατότητα να επιστραφούν οι τιμές των παραπάνω πεδίων όταν τις καλούμε.

**2.Admin.java:**

Είναι υποκλάση της **User** και αναφέρεται σε έναν διαχειριστή. Η κλάση περιλαμβάνει και ένα **private boolean** πεδίο **isAdmin** που είναι **false** το οποίο γίνεται **true** όταν καλείται ο **constructor**. Υπάρχει και η μέθοδος **reset** που διαγράφει όλα τα είδη που έχει λάβει ένας επωφελούμενος. Έχουμε προσθέσει και μια μέθοδο **getIsAdmin()** που επιστρέφει την τιμή του **isAdmin**.

**3.Beneficiary.java:**

Πρόκειται για την κλάση που κληρονομεί την **User** και χαρακτηρίζει έναν επωφελούμενο. Αποτελείται από τα private πεδία int **noPersons**, **RequestDonationList receivedList** και **Requests requestsList** που αντιπροσωπεύουν τα μέλη της οικογένειάς του, τα είδη και ποσότητες που έχει λάβει και ζητήσει αντίστοιχα. Για την πρόσβαση στα πεδία χρησιμοποιούνται public **getters** μέθοδοι για να μην μπορέσουν να αλλάξουν η τιμές τους. Επιπλέον περιέχει διαφορές wrapper μεθόδους των **receivedList** και **requestsList** , με την κατάλληλη διαχείριση των εξαιρέσεων για την τροποποίηση τους τους, όπως η **public void addToReceivedList(Organization org, RequestDonation rd)** ή η **public void addToRequestsList(Organization org, RequestDonation rd)**. Τέλος υπάρχουν κάποιες μέθοδοι που υπολογίζουν και επιστρέφουν κάποια χρήσιμη-χαρακτηριστική πληροφορία για τον **Beneficiary.** Τέτοιες είναι η **public int getLevel()**, που επιστρέφει το level του και η **public double getQqf(RequestDonation r),** που επιστρέφει την ποσότητα που δικαιούται από ένα **RequestDonation** (quantity qualified for).

**4.Donator.java:**

Είναι υποκλάση της **User** και αναφέρεται σε έναν δωριτή. Έχει ως πεδίο το offersList, τύπου Offers, το οποίο αναπαριστά τη λίστα ειδών που επιθυμεί να προσφέρει. Επίσης υπάρχουν οι μεθόδοι **add()** , **modifyUpdate()** , **removeUpdate()** , **commitUpdate()** , **resetUpdate()** και **monitor()**, που είναι **wrappers** και τροποποιούν την **offerslist**. Τέλος έχουμε προσθέσει και μία **getter** μέθοδο, την **getOffersList()** που επιστρέφει την **offersList.‎‎‎**

**5.Entity.java:**

Είναι η κλάση όπου αντιπροσωπεύει ένα υλικό ή υπηρεσία. Έχει ως **private** πεδία το όνομα του είδους , την περιγραφή του και τον αριθμό για την ταυτοποίηση του (**id**). Περιέχει μεθόδους **getters** και **setters** και τις μεθόδους **getEntityInfo()**, που επιστρέφει τις πληροφορίες του είδους, την **getDetails()** που είναι **abstract** και το σώμα της ορίζεται στις υποκλάσεις και την **toString()** που επιστρέφει τις συνολικές πληροφορίες σε τύπο **String** για ένα είδος. Στην **toString()** αρχικοποιήσαμε δύο μεταβλητές **“s”**, **“s1”** τύπου **String** που θα «αποθηκεύουν» τα **getEntityInfo()** και **getDetails()** και στην συνέχεια μέσω της εντολής **return s.concat(s1)** επιστρέφονται ενωμένα. Επίσης έχουμε ορίσει ως **abstract** τρείς μεθόδους, την **getlevel1()** , την **getlevel2()** και την **getlevel3(),** οι οποίες ορίζονται στις υποκλάσεις.

**6.Material.java:**

Είναι υποκλάση της **Entity** και αντιπροσωπεύει ένα είδος που είναι υλικό. Έχει ως **private** πεδία τα **level1**, **level2** και **level3**, που αντιπροσωπεύουν την ποσότητα που δικαιούται ένας επωφελούμενος ανάλογα με τα μέλη της οικογένειας του. Έχουμε υλοποιήσει τις **getLevel1()** , **getLevel2()** και **getLevel3()** όπου επιστρέφουν την ποσότητα για το κάθε **level**. Επίσης έχουμε ορίσει το σώμα της μεθόδου **getDetails()** , που στην υπερκλάση είχε οριστεί σαν **abstract** επιστρέφοντας ένα **String** που θα περιέχει το όνομα και την ποσότητα του είδους για κάθε **level** και επίσης ότι ανήκει στα υλικά.

**7.Service.java:**

Είναι υποκλάση της **Entity** και αντιπροσωπεύει ένα είδος που είναι υπηρεσία. Έχουμε ορίσει το σώμα της μεθόδου **getDetails()** , που στην υπερκλάση είχε οριστεί σαν **abstract**, όπου επιστρέφει το όνομα της υπηρεσίας και το ότι ανήκει στην κατηγορία των υπηρεσιών. Επίσης έπρεπε να οριστούν οι **abstract** μέθοδοι **getLevel1()** , **getLevel2()** και **getLevel3()** της υπερκλάσης παρόλο που δεν χρειάζονται.

**8.RequestDonationList.java:**

Αναπαριστά μια συλλογή από αντικείμενα **RequestDonation** δηλαδή έχει μια συλλογή **rdEntities** με αντικείμενα **RequestDonation**. Για αυτό έχουμε δημιουργήσει το **private** πεδίο **rdEntities** που είναι **Arraylist** και δέχεται αντικείμενα τύπου **RequestDonation**. Για να έχουμε πρόσβαση σε αυτό το πεδίο έχουμε δημιουργήσει και μία **getter** μέθοδο την **getRdEntities()**. Επιπλέον έχουν υλοποιηθεί οι μέθοδοι **get()** , **add()** , **remove()** , **modify()** , **monitor()** και **reset()**, που επεξεργάζονται την λίστα **rdEntities** μετά από τους απαραίτητους ελέγχους όπου χρειάζεται και έχουν δημιουργηθεί άλλες 4 μέθοδοι για να διευκολυνθούμε όσον αφορά τον διαχωρισμό των υλικών από τις υπηρεσίες και την συγκέντρωση τους σε ξεχωριστά **arraylists**. Συγκεκριμένα οι μέθοδοι **listMaterialRdEntities()** και **listServiceRdEntities()** δημιουργούν μια λίστα **matlist** και **servlist,** και προσθέτουν σε αυτή όλα τα υλικά και τις υπηρεσίες αντίστοιχα. Επίσης εκτυπώνουν αριθμημένα τα στοιχεία της κάθε λίστας και μετά την επιστρέφουν. Οι μεθόδοι **getMaterialRdEntities()** και **getServiceRdEntities()** κάνουν το ίδιο με τις δύο παραπάνω μεθόδους, με μόνη διαφορά ότι δεν εκτυπώνουν τα στοιχεία της κάθε λίστας.

**9.Requests.java:**

Πρόκειται για μια κλάση που κληρονομεί την **RequestDonationList** και αναπαριστά το σύνολο των ειδών (**Material** ή **Services**) που ζητά ο **Beneficiary**. Περιέχει την overloaded μέθοδο **public void add(Organization o,RequestDonation r, Beneficiary b)** και την **public void modify(Organization o,RequestDonation r, Beneficiary b, double quantity),** οι οποίες καλούν τις αντίστοιχες της υπερκλάσης μετά από τους κατάλληλους ελέγχους αξιοποιώντας τις μεθόδους **private boolean validRequestDonation(RequestDonation r, Beneficiary ben)** και **private boolean validRequestDonation(RequestDonation r, Beneficiary ben, double quantity)** αντίστοιχα (είναι **private** γιατί δεν χρησιμοποιούνται σε κανένα άλλο σημείο του προγράμματος). Τέλος η **public RequestDonationList** **commit(Organization o, Beneficiary b)**, η οποία προσθέτει στην **receivedList** του **Benefeciary** τα **requests** του μετά από ελέγχους για άλλη μια φορά (σε περίπτωση που κάποιο **request** ενός beneficiary δεν είναι έγκυρο μετά από παράλληλο commit() κάποιου άλλου) και τοποθετεί σε μία καινούργια **RequestDonationList** με όνομα **requestsNotReceived** τα είδη που δεν μπόρεσαν να προστεθούν. Αυτή η λίστα επιστρέφεται από την εντολή **return** **requestsNotReceived** ώστε να αξιοποιηθεί στην μέθοδο **LoginBen\_Commit ()** της κλάσης **Menu** στην συνέχεια. Παράλληλα διαγράφονται τα περιεχόμενα της **requestsList** του **Beneficiary** με την εντολή **b.getRequestsList().reset()**.

**10.Offers.java:**

Υποκλάση της **RequestDonationList**. Αναπαριστά το σύνολο των ειδών που προτίθεται να συνεισφέρει ο **Donator**. Έχει μια μέθοδο **commit(Organization o)** που ενημερώνει τα **currentDonations** του οργανισμού με τις προσφορές που περιέχονται στη λίστα **rdEntities** δηλαδή στην **offersList** του **Donator**. Αν αυτό ολοκληρωθεί επιτυχώς, διαγράφει τα περιεχόμενα της λίστας **rdEntities**, δηλαδή καταχωρεί την δωρεά.

**11.Menu.java:**

**public void Login(Organization o):**

Συνδέεται ένας χρήστης μέσω του κινητού του και το πρόγραμμα μεταβαίνει στην αντίστοιχη μέθοδο ανάλογα με το αν είναι **Beneficiary**, **Donator** ή **Admin**. Εναλλακτικά αν δεν είναι εγγεγραμμένος καλείται η **SignUp(),** η οποία εξηγείται παρακάτω.

**public void SignUp(Organization o, String phone\_number):**

Ρωτάει τον χρήστη ως τι είδους **User** θέλει να εγγραφεί και ανάλογα με την επιλογή του, ζητά και αποθηκεύει τις απαραίτητες πληροφορίες και μαζί με το κινητό του (που έχει περαστεί ως όρισμα) τον εγγράφει.

**public void Login\_Admin(Organization o):**

Όταν καλείται αυτή η μέθοδος, καλοσορίζουμε τον διαχειριστή εμφανίζοντας του τα στοιχεία του μαζί με τέσσερις επιλογές που μπορεί να εκτελέσει. Μόλις του εμφανιστούν οι τέσσερις επιλογές (1.View , 2. Monitor Organization , 3. Logout , 4.Exit) τον ρωτάμε που θέλει να πλοηγηθεί και ο χρήστης πληκτρολογεί έναν αριθμό. Αν πατήσει το 1 θα κληθεί η **Admin\_Case1()** , αν πατήσει 2 θα κληθεί η **Admin\_Case2()** , αν πατήσει 3 θα κάνει **Logout** και αν πατήσει 4 θα τερματίσει το πρόγραμμα. Αν δώσει έναν **int** εκτός από τα 4 παραπάνω με το switch που έχουμε χρησιμοποιήσει θα πάει στην **default** κατάσταση όπου θα του πεί να ξαναπροσπαθήσει και θα κληθεί η **Login\_Admin()**. Στην περίπτωση που πληκτρολογήσει έναν πολύ μεγάλο αριθμό ή αν πληκτρολογίσει γράμματα, δηλαδή όχι **int**, θα του εμφανιστεί ένα μήνυμα που θα του πει να μην βάζει γράμματα ή μεγάλους αρίθμους και ξανακαλείται η **Login\_Admin()**.

**public void Admin\_Case1(Organization o):**

Όταν καλείται αυτή η μέθοδος του εμφανίζονται πέντε επιλογές ( 1.Material 2.Services 3. Back 4. Logout 5. Exit) και το μήνυμα να πληκτρολογίσει ένανα αριθμό. Με τη χρήση αμυντικού προγραμματισμού με **do – while** αν δώσει έναν **int** διαφορετικό από τις πάνω πέντε επιλογές θα του ζητήσει να ξαναδώσει έναν ακέραιο αριθμό. Όπως και στην **Login\_Admin()** αν δώσει έναν αριθμό με πολλούς χαρακτήρες ή γράμματα θα κληθεί η **Admin\_Case1()**.

**1.Material**:

Πατώντας 1 ο χρηστής θα μεταβαίνει εδώ. Σκοπός αυτής της επιλογής είναι να δείξει στον admin όλες τις τωρινές προσφορές των ειδών που είναι υλικά. Επίσης ο **admin** μπορεί να δεί και επιπλέον πληροφορίες για το κάθε υλικό που θα επιλέξει. Πάλι με αμυντικό προγραμματισμό με **do – while** αν ο χρήστης πληκτρολογίσει κάτι διαφορέτικο θα του πεί να ξαναπροσπαθήσει και αν, όπως αναφέραμε παραπάνω πλητρολογίσει μεγάλους αριθμούς ή γράμματα θα καλέσει την **Admin\_Case1()**. Αν δεν υπάρχουν τρέχουσες προσφορές για υλικά είδη θα του βγάλει ένα μήνυμα που θα τον ενημερώνει για ακριβώς αυτό.

**2.Services**

Πατώντας 2 ο χρηστής θα μεταβεί εδώ. Σκοπός αυτής της επιλογής είναι να δείξει στον **admin** όλες τις τωρινές προσφορές των ειδών που είναι υπηρεσίες. Με παρόμοιο τρόπο όπως στην επιλογή 1 με αμυντικό προγραμματισμό θα πληκτρολογεί μια τιμή και θα συμβούν τα αντίστοιχα που αναφέρθηκαν στην **1.Material**. Αν δεν υπάρχουν τρέχουσες προσφορές για υπηρεσίες, θα του βγάλει ένα μήνυμα που θα τον ενημερώνει για ακριβώς αυτό.

**3.Back**

Πατώντας 3 ο χρηστής θα μεταβεί εδώ και θα κληθεί η **Login\_Admin()**.

**4.Logout**

Πατώντας 4 ο χρηστής θα μεταβεί εδώ και θα κληθεί η μέθοδος **Login().**

**5.Εxit**

Πατώντας 5 ο χρηστής θα μεταβεί εδώ και θα κληθεί η μέθοδος **Exit().**

**public void Admin\_Case2(Organization o):**

Όταν καλείται αυτή η μέθοδος του εμφανίζονται έξι επιλογές ( 1.List Beneficiaries 2.List Donators 3.Reset Beneficiaries Lists 4. Back 5. Logout 6. Exit)

**1.List Beneficiaries**

Πατώντας 1 ο χρηστής θα μεταβεί εδώ. Τρέχουμε την **BeneficiaryList** με ένα **for** και εκτυπώνονται οι επωφελούμενοι σε μορφή λίστας. Μετά ζητάμε από τον χρήστη να δώσει μια τιμή που :

• Αν είναι 1 θα έχει την επιλογή να διαγράψει όλα τα είδη που έχει λάβει ένας επωφελούμενος της επιλογής του.

• Αν είναι 2 θα έχει την επιλογή να διαγραψεί από τον οργανισμό εντελώς έναν επωφελούμενο της επιλογής του.

• Αν είναι 3 κάνει Back.

• Αν είναι 4 γίνεται logout καλόντας την **Login()**.

• Αν είναι 5 θα τερματίσει το πρόγραμμα καλώντας την **Exit()**.

**2.List Donators**

Πατώντας 2 ο χρηστής θα μεταβεί εδώ. Τρέχουμε την **DonatorsList** με ένα **for** και εκτυπώνονται οι δωρητλες σε μορφή λίστας. Μετά ζητάμε από τον χρήστη να δώσει μια τιμή που:

• Αν είναι 1 θα έχει την επιλογή να του εμφανιστούν όλες οι τρέχουσες παροχές που προτίθεται να παρέχει ο δωρητής της επιλογής του.

• Αν είναι 2 θα έχει την επιλογή να διαγραψεί από τον οργανισμό έναν δωρητή της επιλογής του.

• Αν είναι 3 κάνει **Back**.

• Αν είναι 4 γίνεται logout καλόντας την **Login()**.

• Αν είναι 5 θα τερματίσει το πρόγραμμα καλώντας την **Exit()**.

**3.Reset Beneficiaries List**

Πατώντας 3 ο χρηστής θα μεταβεί εδώ. Με αυτήν την επιλογή ο admin έχει την δυνατότητα να διαγραψει όλα τα είδη που έχουν λάβει όλοι οι επωφελούμενοι.

**4.Back**

Πατώντας 4 ο χρηστής θα μεταβεί εδώ και θα κληθεί η **Login\_Admin()**.

**5.Logout**

Πατώντας 5 ο χρηστής θα μεταβεί εδώ και θα κληθεί η **Login()**.

**6.Exit**

Πατώντας 6 ο χρηστής θα μεταβεί εδώ και θα κληθεί η **Exit()** .

**public void Login\_Donator(Donator donator,Organization o):**

Η Login\_Donator εμφανίζει χαιρετισμό, τα στοιχεία του χρήστη, το όνομα του οργανισμού και το όνομα του **Admin**.Επίσης, μέσω αυτής με την κατάλληλη επιλογή, μπορεί να κληθούν οι μέθοδοι **Add\_Offer()**, **Show\_Offers()**, **Donator\_Commit ()**, **Logout()** και **Exit()**.

**public void Add\_Offer(Donator donator,Organization o):**

Η **Add\_Offer()** (πλοήγηση στις επιλογές για donation) εμφανίζει αριθμημένες τις δύο κατηγορίες (1. Material, 2. Services). Επιλέγοντας μια κατηγορία, εμφανίζεται αριθμημένη λίστα με τα παρεχόμενα υλικά (**Material**) ή υπηρεσίες (**Services**) της κατηγορίας αυτής και για κάθε είδος αναγράφεται η τρέχουσα ποσότητα του παρεχόμενου υλικού ή υπηρεσίας στον οργανισμό. Επιλέγοντας ένα από τα παρεχόμενα είδη (**Material** ή **Services**) εμφανίζονται οι συνολικές πληροφορίες του και ερωτάται ο donator αν θέλει να το προσφέρει (**y/n**). Αν ναι, συμπληρώνεται η προσφερόμενη ποσότητα (**Material**) ή προσφερόμενες ώρες (**Services**) και καλείται η **add().**

**public void Show\_Offers(Donator donator,Organization o):**

Η **Show\_Offers()** καλεί τις αντίστοιχες μεθόδους της Offers για τον συγκεκριμένο **Donator**. Αρχικά εμφανίζεται αριθμημένη λίστα με τα **RequestDonation** που προσφέρει ο **Donator**, διαφορετικά μήνυμα ότι δεν έχει προσφορές αυτή τη στιγμή. Επιλέγοντας μια γραμμή παροχής **RequestDonation** ο χρήστης έχει τη δυνατότητα να διαγράψει την παροχή αυτή ,να τροποποιήσει την παροχή αλλάζοντας την ποσότητά της, να κάνει καθαρισμό όλων των παροχών και να κάνει **commit** (η επιλογή υπάρχει ξανά και στο κύριο μενού του χρήστη).

**public void Donator\_Commit(Donator donator,Organization o):**

Η **Donator\_Commit**(ολοκλήρωση καταχώρησης δωρεάς) καλεί την **commit()** του **Donator** και τυπώνεται κατάλληλο μήνυμα.

**Login\_Beneficiary(Beneficiary b, Organization o):**

Το αρχικό μενού ενός χρήστη άμα συνδεθεί ως **Beneficiary**. Εμφανίζει χαιρετισμό και παρέχει επιλογές πλοήγησης σε υπομενού (1: Add Request, 2: Show Requests, 3: Commit, 4: Back, 5: Logout, 6: Exit), οι οποίες καλούν αντίστοιχες μεθόδους.

**public void AddRequest(Beneficiary b, Organization o):**

Παρέχει την δυνατότητα στον χρήστη να κάνει **requests** για τα παρεχόμενα είδη από δωρητές και φυσικά τυπώνει τα κατάλληλα μνήματα άμα παρουσιαστούν προβλήματα και εγερθούν exceptions από τις μεθόδους της κλάσης **Requests**. Αξίζει να σημειωθεί ότι στα σημεία που επιλέγεται ένα **RequestDonation** από τα current donations του οργανισμού χρησιμοποιείται η μέθοδος **clone()** της κλάσης **RequestDonation** , π.χ. RequestDonation chosenRd = (RequestDonation)matList.get(matChoice-1).clone(), έτσι ώστε να μπορούμε να κάνουμε **chosenRd.setQuantity(quantity)** αργότερα με την ποσότητα πoυ επιθυμεί ο **Beneficiary,** χωρίς να αλλάζει το ίδιο το **RequestDonation** και τελικά να προσθέσουμε το επιλεγμένο **RequestDonation** στην **requestsList** του **beneficiary**.

**public void ShowRequests(Beneficiary b, Organization o):**

Παρέχει την δυνατότητα στον χρήστη να διαχειριστεί τα **requests** του με επιλογές για διαγραφή, τροποποίηση ποσότητας και **commit**. Το τελευταίο εξηγείται παρακάτω και φυσικά στην τροποποίηση της ζητούμενης ποσότητας ενός υλικού γίνεται και πάλι διαχείριση των κατάλληλων εξαιρέσεων.

**public void LoginBen\_Commit( Beneficiary b, Organization o):**

Αυτή η μέθοδος καλεί την **commit()** του **Beneficiary** και τυπώνει μια λίστα με τα **requestsReceived** του. Εάν η λίστα **requestsNotReceived** που επιστρέφεται από την **commit()** δεν είναι άδεια, παρουσιάζονται στον χρήστη τα περιεχόμενα της, δηλαδή μια λίστα με τα είδη που δεν μπορούσε να λάβει .

**public void LoginBen\_Back(Beneficiary b, Organization o):**

Πρόκειται για μια σχεδόν πανομοιότυπη μέθοδο με την **Login\_Beneficiary()** με την διαφορά ότι δεν τυπώνεται χαιρετισμός, μιας και αυτό είναι κάτι που πρέπει να γίνεται μόνο την πρώτη φορά. Kαλείται στο τέλος των παραπάνω μεθόδων και στα σημεία που ο χρήστης κάνει **back**.

Σε όλες τις παραπάνω μεθόδους έχουν εφαρμοστεί τεχνικές αμυντικού προγραμματισμού με την χρήση βρόγχων **do-while**, **if(sc.hasNextInt())** και διαχείριση της εξαίρεσης **InputMismatchException** για να αποφεύγεται η είσοδος **strings** ή μεγάλων αριθμών, που ξεπερνούν το όριο του **integer**, σε σημεία που δεν πρέπει.

**public void Logout(Organization o):**

Η **Logout**() αποσυνδέει τον χρήστη καλώντας την μέθοδο **Login()**.

**public void Exit():**

Η **Exit()** εκτελεί έξοδο από το πρόγραμμα μέσω της εντολής **System.exit(0)**.

**12.Main.java:**

Στη Main αρχικά δημιουργούμε ένα αντικείμενο **Organization** και εισάγουμε 6 **Entities**, 3 **Material** και 3 **Services**. Δημιουργούμε έναν **Admin**, δύο **Beneficiaries** και έναν **Donator**. Επίσης για έναν Beneficiary από τους δύο, εισάγουμε δυο αίτημα (**request**) και αντίστοιχα εισάγουμε δύο παροχές από τον **Donator**. Τέλος, δημιουργούμε ένα αντικείμενο **Menu** και μέσω αυτού εκτελούμε την μέθοδο **Login()**.

**13.Organization.java:**

Αυτή η κλάση αντιστοιχεί στον οργανισμό που υποστηρίζει όλο αυτό το σύστημα με τις δωρεές. Περιλαμβάνει τα **private** πεδία με το όνομα του οργανισμού, τον **admin** τύπου **Admin** , 3 **arraylists** όπου το 1ο είναι μία λίστα με όλα τα είδη, το 2ο είναι μία λίστα με όλους τους δωρητές και το 3ο πεδίο μία λίστα με όλους τους επωφελούμενους του οργανισμού. Επίσης έχουμε και ένα πεδίο **currentDonations** τύπου **RequestDonationList** που αποτελεί μια λίστα με τις διαθέσιμες προσφορές για τα διάφορα είδη. Έχουμε υλοποιήσει **public** **getter** μεθόδους για το όνομα του οργανισμού,τις 3 λίστες που αναφέρθηκαν παραπάνω ,το πεδίο **currentDonations** και τον **admin**. Έχουμε και μία **public** **setter** μέθοδο για να κάνουμε set τον **admin**. Επιπλέον έχουμε προσθέσει τις μεθόδους, **addEntity()** που προσθέτει ένα είδος στην λίστα με τα είδη, **removeEntity()** που αφαιρεί ένα είδος από την λίστα των ειδών , **insertDonator()** , **insertBeneficiary()** για να προσθέτουμε δωριτές και επωφελούμενους στις λίστες **donatorList** και **beneficiaryList** αντίστοιχα. Έχουν προστεθεί και οι μεθόδοι **removeDonator()** και **removeBeneficiary()** όπου αφαιρούν έναν δωριτή και έναν επωφελούμενο από τις λίστες **donatorList** και **beneficiaryList** αντίστοιχα. Επιπροσθέτως, έχουμε ακομά 3 μεθόδους. Η πρώτη είναι η **listEntities()** όπου εκτυπώνει ξεχωριστά τα υλικά και της υπηρεσίες, η 2η η **listBeneficiaries()** εκτυπώνει όλους τους επωφελούμενους και η **listDonators()** εκτυπώνει όλους τους δωρητές.

**14.RequestDonation.java:**

Αυτή η κλάση περιλαμβάνει δύο private πεδία **Entity entity** και **double quantity** οι τιμές των οποίων αρχικοποιούνται μέσω του constructor και προσπελάσονται με **public getters**. Παράλληλα υπάρχει η δυνατότητα να αλλάξουν με την χρήση **public setters**. Φυσικά στο **setQuantity(double quant)** ελέγχεται αν το **quant** είναι μη αρνητικό για να αλλάξει η τιμή του **quantity**. Τέλος υπερκαλύπτεται η μέθοδος **Object** **clone()** της κλάσης **Object** αφού φυσικά κάνουμε implement την διεπαφή **Cloneable** (ώστε να αποφεύγεται το **CloneNotSupportedException**) και τελικά καλείται για να μπορέσουμε να δημιουργήσουμε αντίγραφα των αντικειμένων **RequestDonation**.

**15.** **BeneficiaryAlreadyExistsException.java:**

Εξαίρεση σε περίπτωση που υπάρχει ήδη ο επωφελούμενος.

**16.** **DonatorAlreadyExistsException.java:**

Εξαίρεση σε περίπτωση που υπάρχει ήδη ο δωριτής.

**17.** **EntityAlreadyExistsException.java:**

Εξαίρεση σε περίπτωση που υπάρχει ήδη το/η υλικό/υπηρεσία**.**

**18.** **EntityAmountDoesNotExistException.java:**

Εξαίρεση σε περίπτωση που δεν υπάρχει το ποσό αυτού/ης του/της υλικού/υπηρεσίας.

**19.** **EntityDoesNotExistException.java:**

Εξαίρεση σε περίπτωση που δεν υπάρχει το/η υλικό/υπηρεσία.

**20.** **InvalidRequestException.java:**

Εξαίρεση σε περίπτωση που το αίτημα του επωφελούμενου δεν μπορεί να πραγματοποιηθεί, διότι δεν τηρεί τις κατάληλες προυποθέσεις.

**21.** **OfferDoesNotExistException.java:**

Εξαίρεση σε περίπτωση που δεν υπάρχει η προσφορά.

**Κώδικας της αναφοράς.**

**1.User.java:**

public abstract class User

{

 private String name;

 private String phone\_number;

 User(String name,String phone\_number)

 {

     this.name=name;

     this.phone\_number=phone\_number;

 }

 public String getName(){

     return name;

 }

 public String getPhone\_number(){

     return phone\_number;

 }

}

**2.Admin.java:**

public class Admin extends User

{

    public boolean isAdmin=false;

    Admin(String name,String phone\_number)

    {

        super(name,phone\_number);

        isAdmin=true;

    }

    public boolean getIsAdmin(){

        return isAdmin;

    }

    protected void reset(Beneficiary beneficiary)

    {

        beneficiary.getReceivedList().getRdEntities().clear();

        //Ginetai clear olh h lista me auta pou exei labei o beneficiary

    }

}

**3.Beneficiary.java:**

public class Beneficiary extends User

{

    private int noPersons=1;

    private RequestDonationList receivedList = new RequestDonationList(); //είδη και ποσοτητες που έχει ήδη λάβει

    private Requests requestsList = new Requests(); //τρέχουσα λίστα των ειδών και των ποσοτήτων που ζητά να του δοθούν

    Beneficiary(String name, String phone\_number, int noPersons)//, RequestDonationList receivedList, Requests requestsList)

    {

     super(name,phone\_number);

     this.noPersons = noPersons;

    }

    public int getNoPersons(){

        return noPersons;

    }

    public RequestDonationList getReceivedList(){

        return receivedList;

    }

    public Requests getRequestsList(){

        return requestsList;

    }

    //wrapper

    public void addToReceivedList(Organization org, RequestDonation rd){

        try{receivedList.add(org, rd);}

        catch(EntityDoesNotExistException e){e.print();}

        catch(EntityAmountDoesNotExistException e){e.print();}

        catch(InvalidRequestException e){e.print();}

    }

    //wrapper

    public void addToRequestsList(Organization org, RequestDonation rd){

        try{requestsList.add(org, rd, this);}

        catch(EntityDoesNotExistException e){e.print();}

        catch(EntityAmountDoesNotExistException e){e.print();}

        catch(InvalidRequestException e){e.print();}

    }

    //wrapper

    public void removeRequest(RequestDonation rd){

        requestsList.remove(rd);

    }

    //wrapper

    public void modifyRequest(Organization o,RequestDonation r, double quantity){

        try{requestsList.modify(o, r, this, quantity);}

        catch(EntityAmountDoesNotExistException e){e.print();}

        catch(InvalidRequestException e){e.print();}

        catch(EntityDoesNotExistException e){e.print();}

    }

    //wrapper

    public void resetRequests(){

        requestsList.reset();

    }

    //wrapper

    public RequestDonationList commit(Organization o){

        RequestDonationList requestsNotReceived;

        requestsNotReceived = requestsList.commit(o, this);

        return requestsNotReceived;

    }

    //wrapper

    public void monitor(){

        if(receivedList.getRdEntities().isEmpty()){

            System.out.println("Κανένα αίτημα σας δεν ικανοποιήθηκε επιτυχώς");

        }

        else{receivedList.monitor();}

    }

    //tupwnei ola ta requests tou beneficiary kai epistrefei mia lista me auta

    public void listRequests(){

        int counter=0;

        if(this.requestsList.getRdEntities().isEmpty()==false){

            for(RequestDonation rd : this.requestsList.getRdEntities()){

            counter++;

            System.out.print(counter + ". ");

            System.out.println(rd.getEntity().getName());

           }

        }

        else{

            System.out.println("Δεν έχετε κάνει κανένα αίτημα");

        }

    }

    //get quantity qualified for

    public double getQqf(RequestDonation r){

        double qqf = 0;

        switch(getLevel()){

        case 1: qqf = r.getEntity().getLevel1(); break;

        case 2: qqf = r.getEntity().getLevel2(); break;

        case 3: qqf = r.getEntity().getLevel3(); break;

        }

        return qqf;

    }

    //epistrefei to level tou beneficiary

     public int getLevel(){

        if(noPersons==1){return 1;}

        else if(noPersons>=2 && noPersons<=4){return 2;}

        else if(noPersons>=5){return 3;}

        else return 0;

    }

}

**4.Donator.java:**

public class Donator extends User

{

    private Offers offersList = new Offers(); //lista twn eidwn pou thelei na prosferei

                                              //petage NullPointerException an den dimiourgousame antikeimeno Offers edw

    Donator(String name,String phone\_number)

    {

     super(name,phone\_number);

    }

    public Offers getOffersList()

    {

        return offersList;

    }

    public void add(Organization o, RequestDonation r){ //add an offer

        try{

            offersList.add(o,r);

        }

        catch (EntityDoesNotExistException e1) {e1.print();}

        catch (InvalidRequestException e2) {e2.print();}

        catch (EntityAmountDoesNotExistException e3) {e3.print();}

    }

    public void modifyUpdate(RequestDonation rd,double quantity)

    {

        offersList.modify(rd, quantity);

    }

    public void removeUpdate(Organization o, RequestDonation rd)

    {

        offersList.remove(rd);

    }

    public void commitUpdate(Organization o)

    {

        offersList.commit(o);

    }

    //kanei reset tin offersList και vgazei apo tin currentDonations mono osa diagrafikan

    public void resetUpdate()

    {

        offersList.reset();

    }

    //wrapper

    public void monitor(){

        offersList.monitor();

        if(offersList.getRdEntities().isEmpty()){

            System.out.println("\nΟ Συγκεκριμένος δωρητής δεν προτίθεται να παρέχει κανένα είδος αυτήν την στιγμή στον οργανισμό.");

        }

    }

}

**5.Entity.java:**

public abstract class Entity

{

    private String name;

    private String description;

    private int id;

    protected boolean isMaterial=false;

    Entity(String name, String description, int id)

    {

        this.name = name;

        this.description = description;

        this.id = id;

    }

    public String getEntityInfo()

    {

        return "Id: "+id+" ,Περιγραφή: "+description+" ,Όνομα: "+name;

    }

    abstract String getDetails();

    public String toString()

    {

        String s = this.getEntityInfo();

        String s1 = this.getDetails();

        return s.concat(s1);

    }

    public boolean getIsMaterial(){

        return isMaterial;

    }

    public String getName(){

        return name;

    }

    public String getDescription(){

        return description;

    }

    public int getId(){

        return id;

    }

    abstract double getLevel1();

    abstract double getLevel2();

    abstract double getLevel3();

}

**6.Material.java:**

public class Material extends Entity

{

    private double level1; //posotita pou dikaioutai 1 atomo

    private double level2; //posotita pou dikaioutai 2-4 atoma

    private double level3; //posotita pou dikaioutai 5+ atoma

    Material(String name, String description, int id,double level1,double level2,double level3)

    {

        super(name, description, id);

        this.level1=level1;//posotita pou dikaioutai 1 atomo

        this.level2=level2;//posotita pou dikaiountai 2-4 atoma

        this.level3=level3;//posostita pou dikaiountai 5+ atoma

        this.isMaterial = true;

    }

    public String getDetails()

    {

        return "\nΤο " + getName() + " ανήκει στην κατηγορία των υλικών.\n" + "Το 1 άτομο δικαιούται " + level1 + " ποσότητα/ες από το υλικό αυτό.\nΤα 2-4 άτομα δικαιούνται " + level2 + " ποσότητα/ες από το υλικό αυτό.\nΤα 5 άτομα δικαιούνται " + level3 + " ποσότητα/ες από το υλικό αυτό.";

    }

    public double getLevel1(){

        return level1;

    }

    public double getLevel2(){

        return level2;

    }

    public double getLevel3(){

        return level3;

    }

}

**7.Service.java:**

public class Service extends Entity

{

    Service(String name, String description, int id){

        super(name, description, id);

        this.isMaterial = false;

    }

    public String getDetails(){

        return "\n" + getName() + " ανήκει στην κατηγορία των υπηρεσιών.";

    }

    //prepei na ginei ylopoihsh tous epeidi einai orismenenes ws abstract stin Entity

    public double getLevel1(){return 0;}

    public double getLevel2(){return 0;}

    public double getLevel3(){return 0;}

}

**8.RequestDonationList.java:**

import java.util.\*;

public class RequestDonationList

{

    private ArrayList <RequestDonation> rdEntities=new ArrayList<RequestDonation>();

    public ArrayList <RequestDonation> getRdEntities(){

        return rdEntities;

    }

    public RequestDonation get(int id){

        RequestDonation object = null;

        for(int i=0; i<rdEntities.size(); i++)

            if(id==rdEntities.get(i).getId()) {

                object = rdEntities.get(i);

                break;

            }

        return object;

    }

    public void add(Organization o,RequestDonation r)throws EntityDoesNotExistException, EntityAmountDoesNotExistException, InvalidRequestException{

        boolean flag1=false;

        for(Entity ent:o.getEntityList()){

            if(ent.getId()==r.getId()){

                flag1=true;

            }

        }

        //an to entity pou bazw den yparzei sto entityList tou organismou throw exception

        if(flag1==false){

            throw new EntityDoesNotExistException();

        }

        //an to entity pou bazw yparxei sthn entityList tou organismou

        if(flag1==true)

        {

            boolean flag2=false;

            for (RequestDonation rd:rdEntities){ //diatrexw tin rdEntities

                //an to RequestDonation pou prosthetw yparxei stin rdEntities kanw update to quantity

                if(r.getId()==rd.getId()){

                    rd.setQuantity(rd.getQuantity() + r.getQuantity());

                    flag2=true;

                }

            }

            //an to Request Donation pou prosthetw den yparxei stin rdEntities to prosthetw

            if(flag2==false){

                rdEntities.add(r);

            }

        }

    }

    public void remove(RequestDonation rd){

        try{

                for(RequestDonation r:rdEntities){

                if(r==rd){

                    rdEntities.remove(r);

                }

            }

        }

        catch(ConcurrentModificationException e){}

    }

    public void modify(RequestDonation rd,double quantity){

        for(RequestDonation r:rdEntities){

            if(r.getId()==rd.getId())

            {

                r.setQuantity(quantity);

            }

        }

    }

    public void monitor(){

        for(RequestDonation rd:rdEntities){

            System.out.println("Όνομα είδους: " + rd.getEntity().getName());

            System.out.println("Ποσότητα/Ώρες: " + rd.getQuantity());

        }

        System.out.println();

    }

    public void reset(){

        try{rdEntities.clear();}

        catch(ConcurrentModificationException e){}

    }

    public ArrayList<RequestDonation> listMaterialRdEntities(){ //diatrexei thn lista rdentities kai typwnei tisplirofories kai thn posotita twn material

        ArrayList<RequestDonation> matList = new ArrayList<RequestDonation>();

        int counter=0;

        for(int i=0; i<rdEntities.size(); i++){

            if(rdEntities.get(i).getEntity().getIsMaterial()==true){

                counter++;

                matList.add(rdEntities.get(i));

                System.out.print(counter + ": " + rdEntities.get(i).getEntity().getName() + "\n");

            }

        }

        return matList;

    }

    public ArrayList<RequestDonation> listServiceRdEntities(){ //diatrexei thn lista rdentities kai typwnei tis plirofories twn service

        ArrayList<RequestDonation> servList = new ArrayList<RequestDonation>();

        int counter=0;

        for(int i=0; i<rdEntities.size(); i++){

            if(rdEntities.get(i).getEntity().getIsMaterial()==false){

                counter++;

                servList.add(rdEntities.get(i));

                System.out.print(counter + ": " + rdEntities.get(i).getEntity().getName() + "\n");

            }

        }

        return servList;

    }

     public ArrayList<RequestDonation> getMaterialRdEntities() {

        ArrayList<RequestDonation> matList = new ArrayList<RequestDonation>();

        for (int i = 0; i < rdEntities.size(); i++) {

            if (rdEntities.get(i).getEntity().getIsMaterial() == true) {

                matList.add(rdEntities.get(i));

            }

        }

        return matList;

    }

    public ArrayList<RequestDonation> getServiceRdEntities(){

        ArrayList<RequestDonation> servList = new ArrayList<RequestDonation>();

        for (int i = 0; i < rdEntities.size(); i++) {

            if (rdEntities.get(i).getEntity().getIsMaterial() == false) {

                servList.add(rdEntities.get(i));

            }

        }

        return servList;

    }

}

**9.Requests.java:**

public class Requests extends RequestDonationList //synolo twn eidwn poy kanei request o Beneficiary

{

    public void add(Organization o,RequestDonation r, Beneficiary b)throws EntityAmountDoesNotExistException, InvalidRequestException, EntityDoesNotExistException

    {

        boolean validAmount = false;

        boolean valid = false;

        boolean rdFound = false;

        boolean isService = false;

        //diatrexw tin lista me ta prosferomena RequestDonations apo Dwrites an vrw service to kanw add kateuthian

        for(RequestDonation rd : o.getCurrentDonations().getRdEntities()){

            if(rd.getId()==r.getId()){rdFound = true;}

            if(r.getEntity().getIsMaterial()==false){isService = true;}

            if(rdFound && r.getQuantity()<=rd.getQuantity()){validAmount = true;}

            if(rdFound && isService && validAmount){super.add(o, r); break;}

            else if(rdFound && isService){

                if(validAmount==false){throw new EntityAmountDoesNotExistException();}

            }

        }

        rdFound = false;

        validAmount = false;

        //diatrexw tin lista me ta prosferomena RequestDonations apo Dwrites kai kanw add mono otan tyrountai ta A, B tis ekfwnisis

        if(isService==false){

            for(RequestDonation rd : o.getCurrentDonations().getRdEntities()){

                if(rd.getId()==r.getId()){rdFound = true;}

                if(rdFound && r.getQuantity()<=rd.getQuantity()){validAmount = true;}

                if(validRequestDonation(r, b)){valid = true;}

                //an isxoyoun ta A, B apo tin ekfwnisi

                if(rdFound && validAmount && valid){

                    super.add(o, r);

                    break;

                }

                //alliws exceptions

                else if(rdFound){

                    if(validAmount==false){throw new EntityAmountDoesNotExistException();}

                    if(valid==false){throw new InvalidRequestException();}

                }

            }

        }

    }

    private boolean validRequestDonation(RequestDonation r, Beneficiary ben){

        double requestedQuant = r.getQuantity(); //requested quantity

        int benLevel = ben.getLevel(); //to level tou beneficiary

        double qqf = 0; //quantity qualified for. Dhladi posotita pou dikaioutai o beneficiary analoga me to level tou

        if(r.getEntity().getIsMaterial()){

            switch(benLevel){

            case 1: qqf = r.getEntity().getLevel1(); break;

            case 2: qqf = r.getEntity().getLevel2(); break;

            case 3: qqf = r.getEntity().getLevel3(); break;

            }

            boolean flag = false;

            boolean matRequested = false;

            RequestDonation rdReceived = null;

            //diatrexw tin receivedList gia na dw poso exei lavei idi o beneficiary apo to requested material

            for(RequestDonation rd : ben.getReceivedList().getRdEntities()){

                if(rd.getId()==r.getId()){

                    matRequested = true;

                    rdReceived = rd;

                }

            }

            //an den exei lavei tpt o beneficiary epistrefw true

            if(ben.getReceivedList().getRdEntities().isEmpty()){

                if(requestedQuant<=qqf){flag = true;}

            }

            //an exei lavei to requested material kai i posotita pou zitaei + tin posotita po exei parei den kseperna to qqf return true

            else if(matRequested == true){

                if(rdReceived.getQuantity() + requestedQuant <= qqf){

                    flag = true;

                }

                else{flag = false;}

            }

            //an exei lavei kati alla oxi auto pou zitaei twra

            else if(matRequested == false){

                if(requestedQuant <= qqf){flag = true;}

            }

            if(flag == true){return true;}

            else{return false;}

        }

        //an einai service

        else{return true;}

    }

    public void modify(Organization o,RequestDonation r, Beneficiary b, double quantity)throws EntityAmountDoesNotExistException, InvalidRequestException, EntityDoesNotExistException{

        boolean validAmount = false;

        boolean valid = false;

        boolean rdFound = false;

        boolean isService = false;

        //diatrexw tin lista me ta prosferomena RequestDonations apo Dwrites an vrw service to kanw modify kateuthian

        for(RequestDonation rd : o.getCurrentDonations().getRdEntities()){

            if(rd.getId()==r.getId()){rdFound = true;}

            if(r.getEntity().getIsMaterial()==false){isService = true;}

            if(rdFound && quantity<=rd.getQuantity()){validAmount = true;}

            if(rdFound && isService && validAmount){super.modify(r, quantity); break;}

            else if(rdFound && isService){

                if(validAmount==false){throw new EntityAmountDoesNotExistException();}

            }

        }

        rdFound = false;

        validAmount = false;

        //diatrexw tin lista me ta prosferomena RequestDonations apo Dwrites kai kanw modify mono otan tyrountai ta A, B tis ekfwnisis

        if(isService==false){

            for(RequestDonation rd : o.getCurrentDonations().getRdEntities()){

                if(rd.getId()==r.getId()){rdFound = true;}

                if(rdFound && quantity<=rd.getQuantity()){validAmount = true;}

                if(validRequestDonation(r, b, quantity)){valid = true;}

                //an isxoyoun ta A, B apo tin ekfwnisi

                if(rdFound && validAmount && valid){

                    super.modify(r, quantity);

                    break;

                }

                //alliws exceptions

                else if(rdFound && !isService){

                    if(validAmount==false){throw new EntityAmountDoesNotExistException();}

                    if(valid==false){throw new InvalidRequestException();}

                }

            }

        }

    }

    private boolean validRequestDonation(RequestDonation r, Beneficiary ben, double quantity){

        double requestedQuant = quantity;

        int benLevel = ben.getLevel(); //to level tou beneficiary

        double qqf = 0; //quantity qualified for. Dhladi posotita pou dikaioutai o beneficiary analoga me to level tou

        if(r.getEntity().getIsMaterial()){

            switch(benLevel){

            case 1: qqf = r.getEntity().getLevel1(); break;

            case 2: qqf = r.getEntity().getLevel2(); break;

            case 3: qqf = r.getEntity().getLevel3(); break;

            }

            boolean flag = false;

            boolean matRequested = false;

            RequestDonation rdReceived = null;

            //diatrexw tin receivedList gia na dw poso exei lavei idi o beneficiary apo to requested material

            for(RequestDonation rd : ben.getReceivedList().getRdEntities()){

                if(rd.getId()==r.getId()){

                    matRequested = true;

                    rdReceived = rd;

                }

            }

            //an den exei lavei tpt o beneficiary epistrefw true

            if(ben.getReceivedList().getRdEntities().isEmpty()){

                if(requestedQuant<=qqf){flag = true;}

            }

            //an exei lavei to requested material kai i posotita pou zitaei + tin posotita po exei parei den kseperna to qqf return true

            else if(matRequested == true){

                if(rdReceived.getQuantity() + requestedQuant <= qqf){

                    flag = true;

                }

                else{flag = false;}

            }

            //an exei lavei kati alla oxi auto pou zitaei twra

            else if(matRequested == false){

                if(requestedQuant <= qqf){flag = true;}

            }

            if(flag == true){return true;}

            else{return false;}

        }

        //an einai service

        else{return true;}

    }

    public RequestDonationList commit(Organization o, Beneficiary b){//throws EntityAmountDoesNotExistException,InvalidRequestException{

        boolean validAmount = false;

        boolean valid = false;

        boolean rdFound = false;

        boolean nonExistantAmount = false;

        boolean invalidReq = false;

        RequestDonationList requestsNotReceived = new RequestDonationList();

        //gia kathe Request tis requestsList diatrexw tin currentDonations gia na dw an yparxei

        for(RequestDonation r : b.getRequestsList().getRdEntities()){

            if(r.getEntity().getIsMaterial()==false){

                for(RequestDonation rd : o.getCurrentDonations().getRdEntities()){

                    if(r.getId()==rd.getId()){

                        rdFound = true;

                        if(r.getQuantity()<=rd.getQuantity()){validAmount = true;}

                    }

                    if(rdFound && validAmount){

                        double newQuant = rd.getQuantity() - r.getQuantity();

                        o.getCurrentDonations().modify(rd, newQuant); //afairw tin posotita pou phra apo to rd tou currentDonations

                        b.addToReceivedList(o,r); //vazw to service stin received list

                        break;

                    }

                    //alliws exceptions kai prosthiki antigrafou tou r stin lista requestsNotReceived

                    else if(rdFound){

                        if(validAmount==false){

                            try{requestsNotReceived.add(o, (RequestDonation)r.clone());}

                            catch(EntityDoesNotExistException e){e.print();}

                            catch(EntityAmountDoesNotExistException e){e.print();}

                            catch(InvalidRequestException e){e.print();}

                            nonExistantAmount = true;

                        }

                        if(valid==false){

                            try{requestsNotReceived.add(o, (RequestDonation)r.clone());}

                            catch(EntityDoesNotExistException e){e.print();}

                            catch(EntityAmountDoesNotExistException e){e.print();}

                            catch(InvalidRequestException e){e.print();}

                            invalidReq = true;

                        }

                    }

                    validAmount = false;

                    valid = false;

                    rdFound = false;

                }

            }

            validAmount = false;

            valid = false;

            rdFound = false;

            if(r.getEntity().getIsMaterial()){

                for(RequestDonation rd : o.getCurrentDonations().getRdEntities()){

                    if(rd.getId()==r.getId()){

                        rdFound = true;

                        if(r.getQuantity()<=rd.getQuantity()){validAmount = true;}

                        if(validRequestDonation(r, b)){valid = true;}

                    }

                    //an isxoyoun ta A, B apo tin ekfwnisi

                    if(rdFound && validAmount && valid){

                        double newQuant = rd.getQuantity() - r.getQuantity();

                        o.getCurrentDonations().modify(rd, newQuant); //afairw tin posotita pou phra apo to rd tou currentDonations

                        b.addToReceivedList(o,r); //vazw to mat stin received list

                        break;

                    }

                    //alliws exceptions kai prosthiki antigrafou tou r stin lista requestsNotReceived

                    else if(rdFound){

                        if(validAmount==false){

                            try{

                                requestsNotReceived.add(o, (RequestDonation)r.clone());

                            }

                            catch(EntityDoesNotExistException e){e.print();}

                            catch(EntityAmountDoesNotExistException e){e.print();}

                            catch(InvalidRequestException e){e.print();}

                            nonExistantAmount = true;

                        }

                        if(valid==false){

                            try{requestsNotReceived.add(o, (RequestDonation)r.clone());}

                            catch(EntityDoesNotExistException e){e.print();}

                            catch(EntityAmountDoesNotExistException e){e.print();}

                            catch(InvalidRequestException e){e.print();}

                            invalidReq = true;

                        }

                    }

                    rdFound = false;

                    valid = false;

                    validAmount = false;

                }

            }

            validAmount = false;

            valid = false;

            rdFound = false;

        }

        b.getRequestsList().reset(); //ta vgazw ola apo tin requests list

        if(nonExistantAmount){

            try{

                throw new EntityAmountDoesNotExistException();

            }

            catch(EntityAmountDoesNotExistException e){

                e.print();

                return requestsNotReceived;

            }

        }

        else if(invalidReq){

            try{

                throw new InvalidRequestException();

            }

            catch(InvalidRequestException e){

                e.print();

                return requestsNotReceived;

            }

        }

        else{

            return requestsNotReceived;

        }

    }

}

**10.Offers.java:**

public class Offers extends RequestDonationList

{

    public void commit(Organization o)

    {

        //enimerwsi twn currentDonations tou organismou me tis prosfores stin lista rdEntities

        for (int i = 0; i < getRdEntities().size(); i++)

        {

            try{o.getCurrentDonations().add(o,getRdEntities().get(i));}

            catch(EntityDoesNotExistException e){e.print();}

            catch(EntityAmountDoesNotExistException e){e.print();}

            catch(InvalidRequestException e){e.print();}

        }

        //diagrafi tis listas rdEntities

        getRdEntities().clear();

    }

}

**11.Menu.java:**

import java.util.\*;

public class Menu

{

    Scanner sc = new Scanner(System.in);

    public void Login(Organization o)

    {

        do

        {

        String phone\_number;

        System.out.println("Σύνδεση...");

        System.out.print("Δώστε το κινητό σας τηλέφωνο: ");

        phone\_number = sc.nextLine();

        try

        {

            Long.parseLong(phone\_number);

            boolean isSignedUp = false;

            if(phone\_number.equals(o.getAdmin().getPhone\_number())){

           Login\_Admin(o);

           isSignedUp = true;

        }

        for(Donator don : o.getDonatorList()){

            if(phone\_number.equals(don.getPhone\_number())){

                Login\_Donator(don, o);

                isSignedUp = true;

                break;

            }

        }

        for(Beneficiary ben : o.getBeneficiaryList()){

            if(phone\_number.equals(ben.getPhone\_number())){

                Login\_Beneficiary(ben, o);

                isSignedUp = true;

                break;

            }

        }

        if(isSignedUp==false){

            SignUp(o, phone\_number);

        }

        }

        catch(NumberFormatException e)

        {

            System.out.println("\nΔώσατε άκυρους χαρακτήρες. Δώστε το τηλέφωνό σας πάλι.\n");

        }

        }while(true);

    }

    public void SignUp(Organization o, String phone\_number){

        do

        {

            System.out.println("\nΩς τι είδους χρήστη επιθυμείτε να εγγραφείτε;");

            System.out.println("Επιλέξτε παρακάτω:");

            System.out.println("1: Δωριτής");

            System.out.println("2: Επωφελούμενος");

            System.out.println("3: Back");

            String choice1 = null;

            System.out.print("\nΕπιλογή: ");

            try {

                choice1 = sc.nextLine();

                int choice=Integer.parseInt(choice1);

            switch(choice){

                case 1:

                    System.out.println("\n0: Back ");

                    System.out.print("\nΠαρακαλώ εισάγετε το όνομα σας: ");

                    String name = sc.nextLine();

                    if(name.equals("0"))

                    {

                        SignUp(o, phone\_number);

                    }

                    else

                    {

                        Donator donator = new Donator(name, phone\_number);

                        try

                        {

                            o.insertDonator(donator);

                            System.out.println();

                            Login(o);

                        }

                        catch (DonatorAlreadyExistsException e){e.print();}

                    }

                    break;

                case 2:

                    System.out.println("\n0: Back ");

                    System.out.print("\nΠαρακαλώ εισάγετε το όνομα σας: ");

                    String name1 = sc.nextLine();

                    if(name1.equals("0"))

                    {

                        SignUp(o ,phone\_number);

                    }

                    else

                    {

                        int noPersons=0;

                        do

                        {

                            try

                            {

                                System.out.print("\nΠαρακαλώ εισάγετε τον αριθμό των μελών της οικογένεια σας: ");

                                noPersons = sc.nextInt();

                                sc.nextLine();

                                if(noPersons<=0)

                                {

                                    System.out.println("\nΟ αριθμός πρέπει να είναι μεγαλύτερος του 0.");

                                }

                            }

                            catch(InputMismatchException e)

                            {

                                System.out.println("\nΛάθος χαρακτήρες.Ξαναδώστε τον αριθμό.");

                                sc.nextLine();

                            }

                        }while(noPersons<=0);

                        Beneficiary beneficiary = new Beneficiary(name1, phone\_number, noPersons);

                        try

                        {

                            o.insertBeneficiary(beneficiary);

                            System.out.println();

                            Login(o);

                        }

                        catch (BeneficiaryAlreadyExistsException e){e.print();}

                    }

                    break;

                case 3:

                    System.out.println();

                    Login(o);

                    break;

                default:

                    System.out.println("\nΛάθος επιλογή. Παρακαλώ επιλέξτε μία απο τις υπάρχουσες επιλογές");

            }

        }

            catch (NumberFormatException e){

                System.out.println("\nΜΗΝ ΔΙΝΕΤΕ ΑΚΥΡΟΥΣ ΧΑΡΑΚΤΗΡΕΣ\n");

                SignUp(o ,phone\_number);

            }

        }

        while(true);

    }

    public void Login\_Donator(Donator donator,Organization o)

    {

        System.out.println("\nΚαλωσορίσατε δωρητή.\n\nΤα στοιχεία σας.\n--------------\nΌνομα: " + donator.getName() + "\nΤηλέφωνο: " + donator.getPhone\_number() + "\nΟργανισμός: " + o.getName() + "\nΌνομα Admin: " + o.getAdmin().getName());

        do

        {

            String epilogh="";

            System.out.println();

            System.out.println("1: Add offer");

            System.out.println("2: Show offers");

            System.out.println("3: Commit");

            System.out.println("4: Logout");

            System.out.println("5: Exit");

            System.out.print("\nΔώστε τον αντίστοιχο αριθμό για την επιλογή σας: ");

            epilogh=sc.nextLine();

            switch(epilogh)

            {

                case "1":

                Add\_Offer(donator,o);

                break;

            case "2":

            Show\_Offers(donator,o);

            break;

            case "3":

            Donator\_Commit(donator,o);

            break;

            case "4":

            System.out.println();

            Logout(o);

            break;

            case "5":

            Exit();

            break;

            default:

            System.out.println("\nΔώσατε λάθος επιλογή.Ξαναδώστε την επιλογή σας.");

            break;

        }

    }while(true);

    }

    public void Add\_Offer(Donator donator,Organization o)

    {

        String epilogh="";

        System.out.println("\n1: Materials\n2: Services\n3: Back");

        int counter1=0;

        int counter2=0;

        System.out.print("\nΔώστε τον αντίστοιχο αριθμό για την επιλογή σας: ");

        epilogh=sc.nextLine();

        switch(epilogh)

        {

            case "1":

            do

            {

                counter1=0;

                System.out.println("\nΠαρεχόμενα είδη από τον οργανισμό:");

                for(var k: o.getEntityList())

                {

                    if(k.getIsMaterial()==true)

                    {

                        counter1++;

                        System.out.println(counter1 + ".Είδος: " + k.getName() + " Id: " + k.getId());

                    }

                }

                counter1=0;

                System.out.println("\nΤα τωρινά donations για υλικά:");

                for(var l: o.getCurrentDonations().getRdEntities())

                {

                    if(l.getEntity().getIsMaterial()==true)

                    {

                        counter1++;

                        System.out.println(counter1 + ".Είδος: " + l.getEntity().getName() + " Id: " + l.getEntity().getId() + " Ποσότητα: " + l.getQuantity());

                    }

                }

                String epilogh\_material="";

                String epilogh\_material\_donate="";

                System.out.print("\nΔώστε το id του υλικού που θέλετε να δείτε τις πληροφορίες και να κάνετε donate,αλλιώς γράψε back: ");

                epilogh\_material=sc.nextLine();

                if(epilogh\_material.equals("back"))

                {

                    Add\_Offer(donator, o);

                }

                boolean flag1=false;

                boolean flag1\_lathous=false;

                RequestDonation rd1= new RequestDonation(null,0);

                try

                {

                for(var h: o.getEntityList())

                {

                    if(Integer.parseInt(epilogh\_material)==h.getId() && h.getIsMaterial()==true)

                    {

                        flag1=true;

                        rd1.setEntity(h);

                    }

                }

                for(var x:o.getCurrentDonations().getRdEntities())

                {

                    if(Integer.parseInt(epilogh\_material)==x.getEntity().getId() && x.getEntity().getIsMaterial()==true)

                    {

                        flag1=true;

                        rd1.setEntity(x.getEntity());

                        rd1.setQuantity(x.getQuantity());

                    }

                }

                }

                catch(NumberFormatException e)

                {

                    flag1\_lathous=true;

                    System.out.println("\nΔώσατε άκυρους χαρακτήρες.Ξαναδώστε το id.");

                }

                if(flag1==true)

                {

                    System.out.println("\n" + rd1.getEntity().toString());

                    System.out.println("Η τρέχουσα ποσότητα για το υλικό είναι: " + rd1.getQuantity());

                    do

                    {

                        System.out.print("\nΘέλετε να κάνετε donate το υλικό " + rd1.getEntity().getName() + ";(y/n): ");

                        epilogh\_material\_donate=sc.nextLine();

                        if(epilogh\_material\_donate.equals("y"))

                        {

                            String posothta;

                            do

                            {

                            posothta="0";

                            System.out.print("\nΔώστε την ποσότητα που θέλετε να κάνετε donate, αλλιώς γράψτε back: ");

                            posothta=sc.nextLine();

                            try

                            {

                                if(posothta.equals("back"))

                                {

                                    break;

                                }

                                if(Double.parseDouble(posothta)<=0)

                                {

                                    System.out.println("\nΗ ποσότητα που κάνετε donate θα πρέπει να είναι μεγαλύτερη του 0.");

                                    posothta="0";

                                    continue;

                                }

                                rd1.setQuantity(Double.parseDouble(posothta));

                                donator.add(o,rd1);

                                System.out.println("\nΤο donation έγινε με επιτυχία.");

                            }

                            catch(NumberFormatException e)

                            {

                                posothta="0";

                                System.err.println("\nΓράψατε λάθος χαρακτήρες.Ξαναδώστε την επιλογή σας.");

                            }

                            }while(posothta.equals("0"));

                        }

                        else if (epilogh\_material\_donate.equals("y")==false && epilogh\_material\_donate.equals("n")==false)

                        {

                            System.out.println("\nΔώσατε λάθος επιλογή.Ξαναδώστε την επιλογή σας.");

                        }

                    }while(epilogh\_material\_donate.equals("y")==false && epilogh\_material\_donate.equals("n")==false);

                }

                else if(flag1==false && flag1\_lathous==false)

                {

                    try

                    {

                    throw new EntityDoesNotExistException();

                    }

                    catch(EntityDoesNotExistException e)

                    {

                        e.print();

                    }

                }

            }while(true);

            case "2":

            do

            {

                counter2=0;

                System.out.println("\nΠαρεχόμενες υπηρεσίες από τον οργανισμό:");

                for(var p: o.getEntityList())

                {

                    if(p.getIsMaterial()==false)

                    {

                        counter2++;

                        System.out.println(counter2 + ".Υπηρεσία: " + p.getName() + " Id: " + p.getId());

                    }

                }

                counter2=0;

                System.out.println("\nΤα τωρινά donations για υπηρεσίες:");

                for(var v: o.getCurrentDonations().getRdEntities())

                {

                    if(v.getEntity().getIsMaterial()==false)

                    {

                        counter2++;

                        System.out.println(counter2 + ".Υπηρεσία: " + v.getEntity().getName() + " Id: " + v.getEntity().getId() + " Ώρες: " + v.getQuantity());

                    }

                }

                String epilogh\_service="";

                String epilogh\_service\_donate="";

                System.out.print("\nΔώστε το id της υπηρεσίας που θέλετε να δείτε τις πληροφορίες και να κάνετε donate,αλλιώς γράψε back: ");

                epilogh\_service=sc.nextLine();

                if(epilogh\_service.equals("back"))

                {

                    Add\_Offer(donator, o);

                }

                boolean flag2=false;

                boolean flag2\_lathous=false;

                RequestDonation rd2= new RequestDonation(null,0);

                try

                {

                for(var h: o.getEntityList())

                {

                    if(Integer.parseInt(epilogh\_service)==h.getId() && h.getIsMaterial()==false)

                    {

                        flag2=true;

                        rd2.setEntity(h);

                    }

                }

                for(var x:o.getCurrentDonations().getRdEntities())

                {

                    if(Integer.parseInt(epilogh\_service)==x.getEntity().getId() && x.getEntity().getIsMaterial()==false)

                    {

                        flag2=true;

                        rd2.setEntity(x.getEntity());

                        rd2.setQuantity(x.getQuantity());

                    }

                }

                }

                catch(NumberFormatException e)

                {

                    flag2\_lathous=true;

                    System.out.println("\nΔώσατε άκυρους χαρακτήρες.Ξαναδώστε το id.");

                }

                if(flag2==true)

                {

                    System.out.println("\n" + rd2.getEntity().toString());

                    System.out.println("Οι τρέχουσες ώρες για την υπηρεσία είναι: " + rd2.getQuantity());

                    do

                    {

                        System.out.print("\nΘέλετε να κάνετε donate την υπηρεσία " + rd2.getEntity().getName() + ";(y/n): ");

                        epilogh\_service\_donate=sc.nextLine();

                        if(epilogh\_service\_donate.equals("y"))

                        {

                            String hours;

                            do

                            {

                            hours="0";

                            System.out.print("\nΔώστε τις ώρες που θέλετε να κάνετε donate, αλλιώς γράψτε back: ");

                            hours=sc.nextLine();

                            try

                            {

                                if(hours.equals("back"))

                                {

                                    break;

                                }

                                if(Double.parseDouble(hours)<=0)

                                {

                                    System.out.println("\nΤο πλήθος των ωρών που κάνετε donate θα πρέπει να είναι μεγαλύτερο του 0.");

                                    hours="0";

                                    continue;

                                }

                                rd2.setQuantity(Double.parseDouble(hours));

                                donator.add(o,rd2);

                                System.out.println("\nΤο donation έγινε με επιτυχία.");

                            }

                            catch(NumberFormatException e)

                            {

                                hours="0";

                                System.err.println("\nΓράψατε λάθος χαρακτήρες.Ξαναδώστε την επιλογή σας.");

                            }

                            }while(hours.equals("0"));

                        }

                        else if (epilogh\_service\_donate.equals("y")==false && epilogh\_service\_donate.equals("n")==false)

                        {

                            System.out.println("\nΔώσατε λάθος επιλογή.Ξαναδώστε την επιλογή σας.");

                        }

                    }while(epilogh\_service\_donate.equals("y")==false && epilogh\_service\_donate.equals("n")==false);

                }

                else if(flag2==false && flag2\_lathous==false)

                {

                    try

                    {

                    throw new EntityDoesNotExistException();

                    }

                    catch(EntityDoesNotExistException e)

                    {

                        e.print();

                    }

                }

            }while(true);

            case "3":

                Login\_Donator(donator,o);

                break;

            default:

                System.out.println("\nΔώσατε λάθος επιλογή.Ξαναδώστε τον αριθμό για την επιλογή σας.");

                Add\_Offer(donator, o);

                break;

        }

    }

    public void Show\_Offers(Donator donator,Organization o)

    {

        do

        {

            if(donator.getOffersList().getRdEntities().isEmpty())

            {

                System.out.println("\nΔεν προσφέρετε κανένα υλικό ή υπηρεσία αυτή την στιγμή.");

                Login\_Donator(donator, o);

            }

            else

            {

                System.out.println("\nΤα υλικά και υπηρεσίες που παρέχετε:");

                for(int i=0; i<donator.getOffersList().getRdEntities().size(); i++)

                {

                    System.out.println(i + "." + donator.getOffersList().getRdEntities().get(i).getEntity().toString());

                    System.out.println("Η/Οι ποσότητα/ώρες του/της υλικού/υπηρεσίας είναι: " + donator.getOffersList().getRdEntities().get(i).getQuantity() + "\n");

                }

                System.out.println("1. Διαγραφή συγκεκριμένης παροχής");

                System.out.println("2. Τροποποίηση ποσότητας παροχής");

                System.out.println("3. Καθαρισμός όλων των παροχών");

                System.out.println("4. Commit");

                System.out.println("5. Back");

                String epilogh="";

                try

                {

                    System.out.print("\nΔώστε την επιλογή σας: ");

                    epilogh=sc.nextLine();

                }

                catch(InputMismatchException e){}

                switch(epilogh)

                {

                    case "1":

                        boolean exists\_diagrafhs=false;

                        do

                        {

                            String epilogh\_diagrafhs="";

                            System.out.print("\nΔώστε το id του συγκεκριμένου υλικού ή υπηρεσίας για διαγραφή της παροχής αυτής, αλλιώς γράψτε back: ");

                            epilogh\_diagrafhs=sc.nextLine();

                            if(epilogh\_diagrafhs.equals("back"))

                            {

                                Show\_Offers(donator, o);

                            }

                            try

                            {

                                for(var x: donator.getOffersList().getRdEntities())

                                {

                                    if(Integer.parseInt(epilogh\_diagrafhs)==x.getEntity().getId())

                                    {

                                        exists\_diagrafhs=true;

                                        donator.removeUpdate(o,x);

                                        System.out.println("\nΗ διαγραφή της παροχής έγινε με επιτυχία.");

                                        break;

                                    }

                                }

                                if(exists\_diagrafhs==false)

                                {

                                    throw new OfferDoesNotExistException();

                                }

                            }

                            catch(NumberFormatException e)

                            {

                                System.err.println("\nΔώσατε άκυρους χαρακτήρες.Ξαναδώστε το id.");

                            }

                            catch(OfferDoesNotExistException e)

                            {

                                System.err.println();

                                e.print();

                                System.err.println("Ξαναδώστε το id.");

                            }

                        }while(exists\_diagrafhs==false);

                        break;

                    case "2":

                        boolean exists\_tropopoihshs=false;

                        do

                        {

                            String epilogh\_tropopoihshs="";

                            System.out.print("\nΔώστε το id του συγκεκριμένου υλικού ή υπηρεσίας που θέλετε να τροποποιηθεί, αλλιώς γράψτε back: ");

                            epilogh\_tropopoihshs=sc.nextLine();

                            if(epilogh\_tropopoihshs.equals("back"))

                            {

                                Show\_Offers(donator, o);

                            }

                            try

                            {

                                for(var y: donator.getOffersList().getRdEntities())

                                {

                                    if(Integer.parseInt(epilogh\_tropopoihshs)==y.getEntity().getId())

                                    {

                                        exists\_tropopoihshs=true;

                                    }

                                }

                                String posothta\_tropopoihshs="";

                                if(exists\_tropopoihshs==true)

                                {

                                    do

                                    {

                                        posothta\_tropopoihshs="";

                                        System.out.print("\nΔώστε την καινούργια ποσότητα της παροχής, αλλιώς γράψτε back: ");

                                        posothta\_tropopoihshs=sc.nextLine();

                                        try

                                        {

                                            if(posothta\_tropopoihshs.equals("back"))

                                            {

                                                exists\_tropopoihshs=false;

                                                break;

                                            }

                                            if(Double.parseDouble(posothta\_tropopoihshs)<=0)

                                            {

                                                System.out.println("\nΗ ποσότητα που τροποποιείτε θα πρέπει να είναι μεγαλύτερη του 0.");

                                                continue;

                                            }

                                            else

                                            {

                                                for(var y: donator.getOffersList().getRdEntities())

                                                {

                                                    if(Integer.parseInt(epilogh\_tropopoihshs)==y.getEntity().getId())

                                                    {

                                                        donator.modifyUpdate(y, Double.parseDouble(posothta\_tropopoihshs));

                                                        break;

                                                    }

                                                }

                                                System.out.println("\nΗ τροποποίηση της παροχής έγινε με επιτυχία.");

                                                break;

                                            }

                                        }

                                        catch(NumberFormatException e)

                                        {

                                            System.err.println("\nΔώσατε λάθος χαρακτήρες.Ξαναδώστε την επιλογή σας.");

                                        }

                                    }while(true);

                                }

                                if(exists\_tropopoihshs==false && posothta\_tropopoihshs.equals("back")==false)

                                {

                                    throw new OfferDoesNotExistException();

                                }

                            }

                            catch(NumberFormatException e)

                            {

                                System.err.println("\nΔώσατε λάθος χαρακτήρες.Ξαναδώστε το id.");

                            }

                            catch(OfferDoesNotExistException e)

                            {

                                System.err.println();

                                e.print();

                                System.err.println("Ξαναδώστε το id.");

                            }

                        }while(exists\_tropopoihshs==false);

                        break;

                    case "3":

                        donator.resetUpdate();

                        System.out.println("\nΌλες οι παροχές σας διαγράφτηκαν με επιτυχία.");

                        Login\_Donator(donator, o);

                        break;

                    case "4":

                        donator.commitUpdate(o);

                        System.out.println("\nΤο commit έγινε με επιτυχία.");

                        Login\_Donator(donator, o);

                        break;

                    case "5":

                        Login\_Donator(donator, o);

                        break;

                    default:

                        System.out.println("\nΔώσατε λάθος αριθμό ή χαρακτήρα.Ξαναδώστε την επιλογή σας.");

                        break;

                }

            }

        }while(true);

    }

    public void Donator\_Commit(Donator donator,Organization o)

    {

        if(donator.getOffersList().getRdEntities().isEmpty())

        {

            System.out.println("\nΔεν προσφέρετε κανένα είδος για να γίνει commit.");

        }

        else

        {

            donator.commitUpdate(o);

            System.out.println("\nΤο commit έγινε με επιτυχία.");

        }

    }

    public void Login\_Admin(Organization o)

    {

            System.out.println("Καλωσορίσατε διαχειριστή.\n\nΤα στοιχεία σας.\n--------------\nΌνομα: " + o.getAdmin().getName() + "\nΤηλέφωνο: " + o.getAdmin().getPhone\_number() + "\n");

            System.out.println("");

            System.out.println("1. View");

            System.out.println("2. Monitor Organization");

            System.out.println("3. Logout");

            System.out.println("4. Exit");

            System.out.print("Που θες να πλοηγηθείς; Δώσε τον αριθμό που αντιστοιχεί στην κάθε επιλογή: ");

        String epilogi1\_a = null;

        try {

            epilogi1\_a = sc.nextLine();

            switch (Integer.parseInt(epilogi1\_a)) {

                case 1:

                    Admin\_Case1(o);

                    break;

                case 2:

                    Admin\_Case2(o);

                    break;

                case 3:

                    Login(o);

                    break;

                case 4:

                    Exit();

                    break;

                default:

                    System.out.println("\nΕΔΩΣΕΣ ΛΑΘΟΣ ΕΠΙΛΟΓΗ!! ΞΑΝΑΠΡΟΣΠΑΘΗΣΕ");

                    System.out.println();

                    Login\_Admin(o);

            }

        }

        catch (NumberFormatException e){

            System.out.println("\nΜΗΝ ΔΙΝΕΤΕ ΑΚΥΡΟΥΣ ΧΑΡΑΚΤΗΡΕΣ\n");

            System.out.println();

            Login\_Admin(o);

        }

    }

    public void Admin\_Case1(Organization o)

    {

        System.out.println();

        System.out.println("------------------");

        System.out.println("|      View      |");

        System.out.println("------------------");

        System.out.println("1. Material");

        System.out.println("2. Services");

        System.out.println("3. Back");

        System.out.println("4. Logout");

        System.out.println("5. Exit");

        System.out.print("Δώσε την τιμή που αντιστοιχει στην πράξη σου: ");

        String epilogi1\_1\_a = null;

        //amyntikos programmatismos

        do{

            try {

                epilogi1\_1\_a = sc.nextLine();

        if(Integer.parseInt(epilogi1\_1\_a)==1) {

            System.out.println();

            if (o.getCurrentDonations().getMaterialRdEntities().isEmpty()) {

                System.out.println("Δεν υπάρχουν τωρινές προσφορές!\n");

                Admin\_Case1(o);

            } else {

                for (int i = 0; i < o.getCurrentDonations().getMaterialRdEntities().size(); i++) {

                    System.out.print((i + 1) + ". " + o.getCurrentDonations().getMaterialRdEntities().get(i).getEntity().getName());

                    System.out.println("(" + o.getCurrentDonations().getMaterialRdEntities().get(i).getQuantity() + ")");

                }

                System.out.println(o.getCurrentDonations().getMaterialRdEntities().size()+1+". Back");

                System.out.println();

                System.out.print("Δωσε τον αντιστοίχο αριθμό για το Material που θες να δείς ή κανε back: ");

                String epilogi\_Material = null;

                do{

                try {

                    epilogi\_Material = sc.nextLine();

                    for (int i = 0; i < o.getCurrentDonations().getMaterialRdEntities().size(); i++) {

                        if (Integer.parseInt(epilogi\_Material) == (i + 1)) {

                            o.getCurrentDonations().getMaterialRdEntities().get(i).getEntity().toString();

                            System.out.println(o.getCurrentDonations().getMaterialRdEntities().get(i).getEntity().getDetails());

                            System.out.println("Η τρέχουσα ποσότητα στον οργανισμό: " + o.getCurrentDonations().getMaterialRdEntities().get(i).getQuantity());

                            System.out.println();

                            break;

                        }

                    }

                    if (Integer.parseInt(epilogi\_Material) > o.getCurrentDonations().getMaterialRdEntities().size() + 1 || Integer.parseInt(epilogi\_Material) < 0) {

                        System.out.println("ΔΩΣΑΤΕ ΛΑΘΟΣ ΕΠΙΛΟΓΗ");

                        System.out.print("Δώστε ξανά την επιλογή σας: ");

                    }

                    if (Integer.parseInt(epilogi\_Material) == o.getCurrentDonations().getMaterialRdEntities().size() + 1)

                        Admin\_Case1(o);

                }

                catch (NumberFormatException e){

                    System.out.println("\nΜΗΝ ΔΙΝΕΤΕ ΑΚΥΡΟΥΣ ΧΑΡΑΚΤΗΡΕΣ\n");

                    System.out.println();

                    Admin\_Case1(o);

                }

                }while(Integer.parseInt(epilogi\_Material)>o.getCurrentDonations().getMaterialRdEntities().size()+1||Integer.parseInt(epilogi\_Material)<0);

                    System.out.println("1. Back");

                    System.out.println("2. Logout");

                    System.out.println("3. Exit");

                String x6 = null;

                do {

                    System.out.print("Δώσε την τιμή που θές: ");

                    try {

                        x6 = sc.nextLine();

                        if (Integer.parseInt(x6) == 1)

                            Admin\_Case1(o);

                        if (Integer.parseInt(x6) == 2)

                            Login(o);

                        if (Integer.parseInt(x6) == 3)

                            Exit();

                        if (Integer.parseInt(x6) != 1 && Integer.parseInt(x6) != 2 && Integer.parseInt(x6) != 3) {

                            System.out.println("ΔΩΣΑΤΕ ΛΑΘΟΣ ΕΠΙΛΟΓΗ - ΘΑ ΠΡΕΠΕΙ ΝΑ ΔΩΣΕΤΕ ΞΑΝΑ ΜΙΑ ΤΙΜΗ");

                        }

                    }

                    catch (NumberFormatException e){

                        System.out.println("\nΜΗΝ ΔΙΝΕΤΕ ΑΚΥΡΟΥΣ ΧΑΡΑΚΤΗΡΕΣ\n");

                        System.out.println();

                        Admin\_Case1(o);

                    }

                }while (Integer.parseInt(x6)!=1&& Integer.parseInt(x6)!=2 && Integer.parseInt(x6)!=3);

            }

        }

            if (Integer.parseInt(epilogi1\_1\_a)== 2) {

                System.out.println();

                if (o.getCurrentDonations().getServiceRdEntities().isEmpty()) {

                    System.out.println("Δεν υπάρχουν τωρινές προσφορές!\n");

                    Admin\_Case1(o);

                } else {

                    for (int i = 0; i < o.getCurrentDonations().getServiceRdEntities().size(); i++) {

                        System.out.print((i + 1) + ". " + o.getCurrentDonations().getServiceRdEntities().get(i).getEntity().getName());

                        System.out.println("(" + o.getCurrentDonations().getServiceRdEntities().get(i).getQuantity() + ")");

                    }

                    System.out.println(o.getCurrentDonations().getServiceRdEntities().size()+1+". Back");

                    System.out.println();

                    System.out.print("Δωσε τον αντιστοίχο αριθμό για το Service που θες να δείς: ");

                    String epilogi\_Service = null;

                    do{

                    try {

                        epilogi\_Service = sc.nextLine();

                        for (int i = 0; i < o.getCurrentDonations().getServiceRdEntities().size(); i++) {

                            if (Integer.parseInt(epilogi\_Service) == (i + 1)) {

                                System.out.println();

                                System.out.print(o.getCurrentDonations().getServiceRdEntities().get(i).getEntity().getEntityInfo());

                                System.out.println(o.getCurrentDonations().getServiceRdEntities().get(i).getEntity().getDetails());

                                System.out.println("Η τρέχουσα ποσότητα στον οργανισμό: " + o.getCurrentDonations().getServiceRdEntities().get(i).getQuantity());

                                System.out.println();

                                break;

                            }

                        }

                        if (Integer.parseInt(epilogi\_Service) > o.getCurrentDonations().getServiceRdEntities().size() + 1 || Integer.parseInt(epilogi\_Service) < 0) {

                            System.out.println("ΔΩΣΑΤΕ ΛΑΘΟΣ ΕΠΙΛΟΓΗ");

                            System.out.print("Δώστε ξανά την επιλογή σας: ");

                        }

                        if (Integer.parseInt(epilogi\_Service) == o.getCurrentDonations().getServiceRdEntities().size() + 1)

                            Admin\_Case1(o);

                    }

                    catch (NumberFormatException e){

                        System.out.println("\nΜΗΝ ΔΙΝΕΤΕ ΑΚΥΡΟΥΣ ΧΑΡΑΚΤΗΡΕΣ\n");

                        System.out.println();

                        Admin\_Case1(o);

                    }

                }while (Integer.parseInt(epilogi\_Service)>o.getCurrentDonations().getServiceRdEntities().size()+1||Integer.parseInt(epilogi\_Service)<0);

                }

                System.out.println("1. Back");

                System.out.println("2. Logout");

                System.out.println("3. Exit");

                String  x7 = null;

                do {

                    System.out.print("Δώσε την τιμή που θές: ");

                    try {

                        x7 = sc.nextLine();

                    if (Integer.parseInt(x7) == 1)

                        Admin\_Case1(o);

                    if (Integer.parseInt(x7) == 2)

                        Login(o);

                    if (Integer.parseInt(x7) == 3)

                        Exit();

                    if (Integer.parseInt(x7) != 1 && Integer.parseInt(x7) != 2 && Integer.parseInt(x7) != 3) {

                        System.out.println("ΔΩΣΑΤΕ ΛΑΘΟΣ ΕΠΙΛΟΓΗ - ΘΑ ΠΡΕΠΕΙ ΝΑ ΞΑΝΑΔΩΣΕΤΕ ΜΙΑ ΤΙΜΗ");

                    }

                }

                    catch (NumberFormatException e){

                        System.out.println("\nΜΗΝ ΔΙΝΕΤΕ ΑΚΥΡΟΥΣ ΧΑΡΑΚΤΗΡΕΣ\n");

                        System.out.println();

                        Admin\_Case1(o);

                    }

                }while (Integer.parseInt(x7)!=1&& Integer.parseInt(x7)!=2 && Integer.parseInt(x7)!=3);

            }

            if (Integer.parseInt(epilogi1\_1\_a) == 3)

                Login\_Admin(o);

            if (Integer.parseInt(epilogi1\_1\_a) == 4)

                Login(o);

            if (Integer.parseInt(epilogi1\_1\_a) == 5)

                Exit();

                if (Integer.parseInt(epilogi1\_1\_a)!=1&&Integer.parseInt(epilogi1\_1\_a)!=2&&Integer.parseInt(epilogi1\_1\_a)!=3&&Integer.parseInt(epilogi1\_1\_a)!=4&&Integer.parseInt(epilogi1\_1\_a)!=5) {

                    System.out.println("\nΕΔΩΣΕΣ ΛΑΘΟΣ ΕΠΙΛΟΓΗ!! ΞΑΝΑΠΡΟΣΠΑΘΗΣΕ");

                    System.out.print("Δώστε ξανά την επιλογή σας: ");

                }

            }

            catch (NumberFormatException e)

            {

                System.out.println("\nΜΗΝ ΔΙΝΕΤΕ ΑΚΥΡΟΥΣ ΧΑΡΑΚΤΗΡΕΣ\n");

                Admin\_Case1(o);

            }

        }while(Integer.parseInt(epilogi1\_1\_a)!=1&&Integer.parseInt(epilogi1\_1\_a)!=2&&Integer.parseInt(epilogi1\_1\_a)!=3&&Integer.parseInt(epilogi1\_1\_a)!=4&&Integer.parseInt(epilogi1\_1\_a)!=5);

    }

    public void Admin\_Case2(Organization o)

    {

        System.out.println();

        System.out.println("------------------------");

        System.out.println("| Monitor Organization |");

        System.out.println("------------------------");

        System.out.println("1. List Beneficiaries");

        System.out.println("2. List Donators");

        System.out.println("3. Reset Beneficiaries Lists");

        System.out.println("4. Back");

        System.out.println("5. Logout");

        System.out.println("6. Exit");

        System.out.print("Επίλεξε ποια λίστα θες να επεξεργαστείς: ");//επίλεξε οχι επέλεξε

        String epilogi2\_a =null;

        do {

            try {

                epilogi2\_a = sc.nextLine();

                if (Integer.parseInt(epilogi2\_a) == 1) {

                    if (o.getBeneficiaryList().isEmpty()) {

                        System.out.println("\nΔΕΝ ΥΠΑΡΧΟΥΝ Επωφελούμενοι!");

                        Admin\_Case2(o);

                    } else {

                        System.out.println();

                        for (int i = 0; i < o.getBeneficiaryList().size(); i++)

                            System.out.println((i + 1) + ". " + o.getBeneficiaryList().get(i).getName());

                        System.out.println();

                        System.out.println("1. Δυνατότητα να γίνει καθαρισμός όλως των ειδών που έχει λάβει ο Επωφελούμενος της επιλογής σου.");

                        System.out.println("2. Δίνεται η δυνατότητα να διαγραφεί ο Επωφελούμενος της επιλογής σου. ");

                        System.out.println("3. Back");

                        System.out.println("4. Logout");

                        System.out.println("5. Exit");

                        String x = null;

                        do {

                            System.out.print("\nΔώσε την τιμή που θές: ");

                            try {

                                x = sc.nextLine();

                                if (Integer.parseInt(x) == 1) {

                                    System.out.println();

                                    for (int i = 0; i < o.getBeneficiaryList().size(); i++)

                                        System.out.println((i + 1) + ". " + o.getBeneficiaryList().get(i).getName());

                                    System.out.println(o.getBeneficiaryList().size() + 1 + ". Back");

                                    System.out.println();

                                    System.out.print("Διάλεξε σε ποιον θες να κάνεις καθαρισμό ή κάνε back: ");

                                    String x1 = null;

                                    do {

                                        try {

                                            x1 = sc.nextLine();

                                            for (int i = 0; i < o.getBeneficiaryList().size(); i++) {

                                                if (Integer.parseInt(x1) == (i + 1)) {

                                                    if (o.getBeneficiaryList().get(i).getReceivedList().getRdEntities().isEmpty())

                                                        System.out.println("\nΔεν έχει λάβει κανένα είδος!\n");

                                                    else if (!o.getBeneficiaryList().get(i).getReceivedList().getRdEntities().isEmpty()) {

                                                        System.out.println("Είχε: \n");

                                                        for (int j = 0; j < o.getBeneficiaryList().get(i).getReceivedList().getRdEntities().size(); j++) {

                                                            System.out.print((j + 1) + "." + o.getBeneficiaryList().get(i).getReceivedList().getRdEntities().get(j).getEntity().getName());

                                                            System.out.println("(" + o.getBeneficiaryList().get(i).getReceivedList().getRdEntities().get(j).getQuantity() + ")");

                                                        }

                                                        System.out.println();

                                                        o.getAdmin().reset(o.getBeneficiaryList().get(i));

                                                        System.out.println("Εγινε διαγραφή των ειδών που έχει λάβει ο Επωφελούμενος: " + o.getBeneficiaryList().get(i).getName() + " ✓");

                                                        break;

                                                    }

                                                }

                                            }

                                            if (Integer.parseInt(x1) == o.getBeneficiaryList().size() + 1)

                                                Admin\_Case2(o);

                                            if (Integer.parseInt(x1) > o.getBeneficiaryList().size() + 1 || Integer.parseInt(x1) < 0) {

                                                System.out.println("ΔΩΣΑΤΕ ΛΑΘΟΣ ΕΠΙΛΟΓΗ");

                                                System.out.print("Δώστε ξανά την επιλογή σας: ");

                                            }

                                        }

                                        catch (NumberFormatException e)

                                        {

                                            System.out.println("\nΜΗΝ ΔΙΝΕΤΕ ΑΚΥΡΟΥΣ ΧΑΡΑΚΤΗΡΕΣ\n");

                                            System.out.println();

                                            Admin\_Case2(o);

                                        }

                                    } while (Integer.parseInt(x1) > o.getBeneficiaryList().size() + 1 || Integer.parseInt(x1) < 0);

                                    System.out.println();

                                    System.out.println("1. Back");

                                    System.out.println("2. Logout");

                                    System.out.println("3. Exit");

                                    String x8 = null;

                                    do {

                                        System.out.print("Δώσε την τιμή που θές: ");

                                        try {

                                            x8 = sc.nextLine();

                                            if (Integer.parseInt(x8) == 1)

                                                Admin\_Case2(o);

                                            if (Integer.parseInt(x8) == 2)

                                                Login(o);

                                            if (Integer.parseInt(x8) == 3)

                                                Exit();

                                            if (Integer.parseInt(x8) != 1 && Integer.parseInt(x8) != 2 && Integer.parseInt(x8) != 3) {

                                                System.out.println("ΔΩΣΑΤΕ ΛΑΘΟΣ ΕΠΙΛΟΓΗ - Θα πρέπει να ξαναδώσετ μια τιμή");

                                            }

                                        }

                                        catch (NumberFormatException e)

                                        {

                                            System.out.println("\nΜΗΝ ΔΙΝΕΤΕ ΑΚΥΡΟΥΣ ΧΑΡΑΚΤΗΡΕΣ\n");

                                            System.out.println();

                                            Admin\_Case2(o);

                                        }

                                    } while (Integer.parseInt(x8) != 1 && Integer.parseInt(x8) != 2 && Integer.parseInt(x8) != 3);

                                }

                                if (Integer.parseInt(x) == 2) {

                                    System.out.println();

                                    for (int i = 0; i < o.getBeneficiaryList().size(); i++)

                                        System.out.println((i + 1) + ". " + o.getBeneficiaryList().get(i).getName());

                                    System.out.println(o.getBeneficiaryList().size() + 1 + ". Back");

                                    System.out.print("Δώσε ποιον Επωφελούμενο θες να διαγράψεις ή κάνε back: ");

                                    String x2 = null;

                                    do {

                                        try {

                                            x2 = sc.nextLine();

                                            for (int i = 0; i < o.getBeneficiaryList().size(); i++) {

                                                if (Integer.parseInt(x2) == (i + 1)) {

                                                    System.out.println("\nO Επωφελούμενος: " + o.getBeneficiaryList().get(i).getName() + " διαγραφτηκε! ✓");

                                                    o.removeBeneficiary(o.getBeneficiaryList().get(i));

                                                    Admin\_Case2(o);

                                                }

                                            }

                                            if (Integer.parseInt(x2) == o.getBeneficiaryList().size() + 1)

                                                Admin\_Case2(o);

                                            if (Integer.parseInt(x2) < 0 || Integer.parseInt(x2) > o.getBeneficiaryList().size() + 1) {

                                                System.out.println("ΔΩΣΑΤΕ ΛΑΘΟΣ ΕΠΙΛΟΓΗ");

                                                System.out.print("Δώστε ξανά την επιλογή σας: ");

                                            }

                                        }

                                        catch (NumberFormatException e)

                                        {

                                            System.out.println("\nΜΗΝ ΔΙΝΕΤΕ ΑΚΥΡΟΥΣ ΧΑΡΑΚΤΗΡΕΣ\n");

                                            System.out.println();

                                            Admin\_Case2(o);

                                        }

                                    } while (Integer.parseInt(x2) < 0 || Integer.parseInt(x2) > o.getBeneficiaryList().size() + 1);

                                }

                                if (Integer.parseInt(x) == 3)

                                    Admin\_Case2(o);

                                if (Integer.parseInt(x) == 4)

                                    Login(o);

                                if (Integer.parseInt(x) == 5)

                                    Exit();

                                if (Integer.parseInt(x) != 1 && Integer.parseInt(x) != 2 && Integer.parseInt(x) != 3 && Integer.parseInt(x) != 4 && Integer.parseInt(x) != 5) {

                                    System.out.print("ΔΩΣΑΤΕ ΛΑΘΟΣ ΕΠΙΛΟΓΗ - ΘΑ ΠΡΕΠΕΙ ΝΑ ΞΑΝΑΔΩΣΕΤΕ ΜΙΑ ΤΙΜΗ");

                                }

                            }

                            catch (NumberFormatException e)

                            {

                                System.out.println("\nΜΗΝ ΔΙΝΕΤΕ ΑΚΥΡΟΥΣ ΧΑΡΑΚΤΗΡΕΣ\n");

                                System.out.println();

                                Admin\_Case2(o);

                            }

                        } while (Integer.parseInt(x) != 1 && Integer.parseInt(x) != 2 && Integer.parseInt(x) != 3 && Integer.parseInt(x) != 4 && Integer.parseInt(x) != 5);

                    }

                }

                if (Integer.parseInt(epilogi2\_a) == 2) {

                    if (o.getDonatorList().isEmpty()) {

                        System.out.println("\nΔΕΝ ΥΠΑΡΧΟΥΝ Δωρητές!");

                        Admin\_Case2(o);

                    } else {

                        System.out.println();

                        for (int i = 0; i < o.getDonatorList().size(); i++)

                            System.out.println((i + 1) + "." + o.getDonatorList().get(i).getName());

                        System.out.println();

                        System.out.println("1. Δυνατότητα να εμφανιστούν όλες οι τρέχουσες παροχές που πρροτίθεται να παρέχει ο Δωριτής της επιλογής σου.");

                        System.out.println("2. Δίνεται η δυνατότητα να διαγραφεί ο Δωριτής της επιλογής σου. ");

                        System.out.println("3. Back");

                        System.out.println("4. Logout");

                        System.out.println("5. Exit");

                        System.out.print("Δώσε την τιμή που θές: ");

                        String x3 = null;

                        do {

                            try {

                                x3 = sc.nextLine();

                                if (Integer.parseInt(x3) == 1) {

                                    System.out.println();

                                    for (int i = 0; i < o.getDonatorList().size(); i++)

                                        System.out.println((i + 1) + ". " + o.getDonatorList().get(i).getName());

                                    System.out.println(o.getDonatorList().size() + 1 + ". Back");

                                    System.out.print("Δωσε ποιου Δωριτή τις παροχές θες να δεις ή κάνε back: ");

                                    String x4 = null;

                                    do {

                                        try {

                                            x4 = sc.nextLine();

                                            for (int i = 0; i < o.getDonatorList().size(); i++) {

                                                if (Integer.parseInt(x4) == (i + 1)) {

                                                    System.out.println("\nO Δωριτής: " + o.getDonatorList().get(i).getName() + " πρροτίθεται να παρέχει: ");

                                                    o.getDonatorList().get(i).monitor();

                                                    break;

                                                }

                                            }

                                            if (Integer.parseInt(x4) == o.getDonatorList().size() + 1)

                                                Admin\_Case2(o);

                                            if (Integer.parseInt(x4) < 0 || Integer.parseInt(x4) > o.getDonatorList().size() + 1) {

                                                System.out.println("ΔΩΣΑΤΕ ΛΑΘΟΣ ΕΠΙΛΟΓΗ");

                                                System.out.print("Δώστε ξανά την επιλογή σας: ");

                                            }

                                        }

                                        catch (NumberFormatException e)

                                        {

                                            System.out.println("\nΜΗΝ ΔΙΝΕΤΕ ΑΚΥΡΟΥΣ ΧΑΡΑΚΤΗΡΕΣ\n");

                                            System.out.println();

                                            Admin\_Case2(o);

                                        }

                                    } while (Integer.parseInt(x4) < 0 || Integer.parseInt(x4) > o.getDonatorList().size() + 1);

                                    System.out.println();

                                    System.out.println("1. Back");

                                    System.out.println("2. Logout");

                                    System.out.println("3. Exit");

                                    System.out.print("Δώσε την τιμή που θές: ");

                                    String x10 = null;

                                    do {

                                        try {

                                            x10 = sc.nextLine();

                                            if (Integer.parseInt(x10) == 1)

                                                Admin\_Case2(o);

                                            if (Integer.parseInt(x10) == 2)

                                                Login(o);

                                            if (Integer.parseInt(x10) == 3)

                                                Exit();

                                            if (Integer.parseInt(x10) != 1 && Integer.parseInt(x10) != 2 && Integer.parseInt(x10) != 3) {

                                                System.out.println("ΔΩΣΑΤΕ ΛΑΘΟΣ ΕΠΙΛΟΓΗ");

                                                System.out.print("Δώστε ξανά την επιλογή σας: ");

                                            }

                                        }

                                        catch (NumberFormatException e)

                                        {

                                            System.out.println("\nΜΗΝ ΔΙΝΕΤΕ ΑΚΥΡΟΥΣ ΧΑΡΑΚΤΗΡΕΣ\n");

                                            System.out.println();

                                            Admin\_Case2(o);

                                        }

                                    } while (Integer.parseInt(x10) != 1 && Integer.parseInt(x10) != 2 && Integer.parseInt(x10) != 3);

                                }

                                if (Integer.parseInt(x3) == 2) {

                                    for (int i = 0; i < o.getDonatorList().size(); i++)

                                        System.out.println((i + 1) + ". " + o.getDonatorList().get(i).getName());

                                    System.out.println(o.getDonatorList().size() + 1 + ". Back");

                                    System.out.print("Δωσε ποιον Δωριτή θες να διαγράψεις ή κάνε back: ");

                                    String x5 = null;

                                    do {

                                        try {

                                            x5 = sc.nextLine();

                                            for (int i = 0; i < o.getDonatorList().size(); i++) {

                                                if (Integer.parseInt(x5) == i + 1) {

                                                    System.out.println("\nΟ Δωριτής: " + o.getDonatorList().get(i).getName() + " διαγραφτηκε! ✓");

                                                    o.removeDonator(o.getDonatorList().get(i));

                                                    Admin\_Case2(o);

                                                }

                                                if (Integer.parseInt(x5) == o.getDonatorList().size() + 1)

                                                    Admin\_Case2(o);

                                                if (Integer.parseInt(x5) < 0 || Integer.parseInt(x5) > o.getDonatorList().size() + 1) {

                                                    System.out.println("ΔΩΣΑΤΕ ΛΑΘΟΣ ΕΠΙΛΟΓΗ");

                                                    System.out.print("Δώστε ξανά την επιλογή σας: ");

                                                }

                                            }

                                        }

                                        catch (NumberFormatException e)

                                        {

                                            System.out.println("\nΜΗΝ ΔΙΝΕΤΕ ΑΚΥΡΟΥΣ ΧΑΡΑΚΤΗΡΕΣ\n");

                                            System.out.println();

                                            Admin\_Case2(o);

                                        }

                                    } while (Integer.parseInt(x5) < 0 || Integer.parseInt(x5) > o.getDonatorList().size() + 1);

                                }

                                if (Integer.parseInt(x3) == 3)

                                    Admin\_Case2(o);

                                if (Integer.parseInt(x3) == 4)

                                    Login(o);

                                if (Integer.parseInt(x3) == 5)

                                    Exit();

                                if (Integer.parseInt(x3) != 1 && Integer.parseInt(x3) != 2 && Integer.parseInt(x3) != 3 && Integer.parseInt(x3) != 4 && Integer.parseInt(x3) != 5) {

                                    System.out.println("ΔΩΣΑΤΕ ΛΑΘΟΣ ΕΠΙΛΟΓΗ");

                                    System.out.print("Δώστε ξανά την επιλογή σας: ");

                                }

                            }

                            catch (NumberFormatException e)

                            {

                                System.out.println("\nΜΗΝ ΔΙΝΕΤΕ ΑΚΥΡΟΥΣ ΧΑΡΑΚΤΗΡΕΣ\n");

                                Admin\_Case2(o);

                            }

                        } while (Integer.parseInt(x3) != 1 && Integer.parseInt(x3) != 2 && Integer.parseInt(x3) != 3 && Integer.parseInt(x3) != 4 && Integer.parseInt(x3) != 5);

                    }

                }

                if (Integer.parseInt(epilogi2\_a) == 3) {

                    if (o.getBeneficiaryList().isEmpty()) {

                        System.out.println("\nΔΕΝ ΥΠΑΡΧΟΥΝ επωφελούμενοι!");

                        Admin\_Case2(o);

                    } else {

                        for (int i = 0; i < o.getBeneficiaryList().size(); i++) {

                            o.getAdmin().reset(o.getBeneficiaryList().get(i));

                        }

                        System.out.println("\nΈγινε διαγραφή των ειδών που έχουν λάβει όλοι οι Επωφελούμενοι. ✓\n");

                        System.out.println("1. Back");

                        System.out.println("2. Logout");

                        System.out.println("3. Exit");

                        System.out.print("Δώσε την τιμή που θές: ");

                        String x12 = null;

                        do {

                            try {

                                x12 = sc.nextLine();

                                if (Integer.parseInt(x12) == 1)

                                    Admin\_Case2(o);

                                if (Integer.parseInt(x12) == 2)

                                    Login(o);

                                if (Integer.parseInt(x12) == 3)

                                    Exit();

                                if (Integer.parseInt(x12) != 1 && Integer.parseInt(x12) != 2 && Integer.parseInt(x12) != 3) {

                                    System.out.println("ΔΩΣΑΤΕ ΛΑΘΟΣ ΕΠΙΛΟΓΗ");

                                    System.out.print("Δώστε ξανά την επιλογή σας: ");

                                }

                            }

                            catch (NumberFormatException e)

                            {

                                System.out.println("\nΜΗΝ ΔΙΝΕΤΕ ΑΚΥΡΟΥΣ ΧΑΡΑΚΤΗΡΕΣ\n");

                                Admin\_Case2(o);

                            }

                        } while (Integer.parseInt(x12) != 1 && Integer.parseInt(x12) != 2 && Integer.parseInt(x12) != 3);

                    }

                }

                if (Integer.parseInt(epilogi2\_a) == 4)

                    Login\_Admin(o);

                if (Integer.parseInt(epilogi2\_a) == 5)

                    Login(o);

                if (Integer.parseInt(epilogi2\_a) == 6)

                    Exit();

                if (Integer.parseInt(epilogi2\_a) != 1 && Integer.parseInt(epilogi2\_a) != 2 && Integer.parseInt(epilogi2\_a) != 3 && Integer.parseInt(epilogi2\_a) != 4 && Integer.parseInt(epilogi2\_a) != 5 && Integer.parseInt(epilogi2\_a) != 6) {

                    System.out.println("\nΕΔΩΣΕΣ ΛΑΘΟΣ ΕΠΙΛΟΓΗ!! ΞΑΝΑΠΡΟΣΠΑΘΗΣΕ");

                    System.out.print("Δώστε ξανά την επιλογή σας: ");

                }

            }

            catch (NumberFormatException e){

                System.out.println("\nΜΗΝ ΔΙΝΕΤΕ ΑΚΥΡΟΥΣ ΧΑΡΑΚΤΗΡΕΣ\n");

                System.out.println();

                Admin\_Case2(o);

            }

        }while (Integer.parseInt(epilogi2\_a) != 1 && Integer.parseInt(epilogi2\_a) != 2 && Integer.parseInt(epilogi2\_a) != 3 && Integer.parseInt(epilogi2\_a) != 4 && Integer.parseInt(epilogi2\_a) != 5 && Integer.parseInt(epilogi2\_a) != 6);

    }

    public void Login\_Beneficiary(Beneficiary b, Organization o){

        System.out.println();

        System.out.println("Οργανισμός: " + o.getName());

        System.out.println("Καλωσορίσατε επωφελούμενε\n\nΤα στοιχεία σας:\n----------------\nΌνομα: " + b.getName() + "\nΤηλέφωνο: " + b.getPhone\_number());

        System.out.println("Αριθμός μελών οικογένειας: " + b.getNoPersons() +"\n");

        System.out.println("1: Add Request");

        System.out.println("2: Show Requests");

        System.out.println("3: Commit");

        System.out.println("4: Back");

        System.out.println("5: Logout");

        System.out.println("6: Exit");

        try

        {

        String sChoice;

        int choice = -1;

        do{

            System.out.println();

            System.out.print("Δώστε την επιλογή σας: ");

            //amintikos programmatismos gia tin antimetopisi number format exception

            sChoice = sc.nextLine();

            choice = Integer.parseInt(sChoice);

            switch(choice){

                case 1: AddRequest(b, o); break;

                case 2: ShowRequests(b, o); break;

                case 3: LoginBen\_Commit(b, o); break;

                case 4: Logout(o); break;

                case 5: Logout(o); break;

                case 6: Exit(); break;

                default: System.out.println("Λάθος επιλογή...");

                }

        }

        while(choice<1 || choice>6);

        }

        //apotrepei ton xristi apo to na dinei grammata

        catch(NumberFormatException e){

            System.out.println("\nΜΗΝ ΔΙΝΕΤΕ ΑΚΥΡΟΥΣ ΧΑΡΑΚΤΗΡΕΣ\n");

            LoginBen\_Back(b, o);

        }

    }

    public void AddRequest(Beneficiary b, Organization o){

        if(o.getCurrentDonations().getRdEntities().isEmpty()){

            System.out.println("Δεν προσφέρεται κανένα είδος αυτή την στιγμή");

            LoginBen\_Back(b, o);

        }

        else{

            //diaxeirisi NumberFormatException gia na apotrepw ton xrisrti na plhktrologei strings i megalous aritmous ekei pou den prepei

            try

            {

                System.out.print("1. Material");

                int matCounter=0;

                //μετράει τα υλικά απο την λίστα με τα προσφερώμενα entities toy organismou

                for(RequestDonation rd : o.getCurrentDonations().getRdEntities()){

                    if(rd.getEntity().getIsMaterial()){matCounter++;}

                }

                System.out.print(" (ο οργανισμός παρέχει " + matCounter + " υλικά)\n");

                System.out.print("2. Services");

                int servCounter=0;

                //μετράει τις υπηρεσίες απο την λιστα με τα προσφερώμενα entities του organismou

                for(RequestDonation rd : o.getCurrentDonations().getRdEntities()){

                    if(rd.getEntity().getIsMaterial()==false){servCounter++;}

                }

                System.out.print(" (ο οργανισμός παρέχει " + servCounter + " υπηρεσίες)\n");

                int choice = -1;

                do{

                    System.out.println("3: Back");

                    System.out.print("\nΠαρακαλώ επιλέξτε μία κατηγορία: ");

                    String sChoice = sc.nextLine();

                    choice = Integer.parseInt(sChoice);

                    if(choice==1){

                        ArrayList<RequestDonation> matList;

                        matList = o.getCurrentDonations().listMaterialRdEntities();//epistrefei lista me ola ta material kai parallila ta typwnei gia na ta dei o xristis

                        if(matList.isEmpty()){

                            System.out.println("\nΔεν παρέχεται κανένα υλικό αυτή την στιγμή.\n");

                            AddRequest(b, o);

                        }

                        int matChoice = -1;

                        do{

                            System.out.println("0: Back");

                            System.out.print("\nΕπιλέξτε ποιο υλικό επιθυμείτε: ");

                            String sMatChoice = sc.nextLine();

                            matChoice = Integer.parseInt(sMatChoice);

                            if(matChoice==0){AddRequest(b, o);}

                            else if(matChoice>0 && matChoice<=matList.size()) {

                                RequestDonation chosenRd = (RequestDonation)matList.get(matChoice-1).clone();// dimiourgw clone tou RequestDonation pou dialekse

                                RequestDonation requestedRd = null;

                                //diatrexv thn requests list gia na dw an exei zitisei to yliko idi

                                for(RequestDonation rd : b.getRequestsList().getRdEntities()){

                                    if(rd.getId()==chosenRd.getId()){

                                        requestedRd = rd;

                                    }

                                }

                                //diatrexw tin current donations gia na dw poso exei lavei idi apo to iliko

                                RequestDonation receivedRd = null;

                                for(RequestDonation rd : b.getReceivedList().getRdEntities()){

                                    if(rd.getId()==chosenRd.getId()){

                                        receivedRd = rd;

                                    }

                                }

                                System.out.println(matChoice + ". " + chosenRd.getEntity().getName() + ": ");

                                System.out.println(chosenRd.getEntity().toString());

                                if(requestedRd != null){

                                    System.out.println("Ποσότητα που έχετε ζητήσει: " + requestedRd.getQuantity());

                                }

                                if(receivedRd != null){

                                    System.out.println("Ποσότητα που έχετε λάβει: " + receivedRd.getQuantity());

                                }

                                System.out.println("Διαθέσιμη ποσότητα στον οργανισμό: " + chosenRd.getQuantity());

                                System.out.println("Ποσοτητα που δικαιούστε: " + b.getQqf(chosenRd));

                                String sQuantity;

                                double quantity = -1;

                                do{

                                    System.out.println("0: Back\n");

                                    System.out.print("Δώστε την ποσότητα που επιθυμείτε ή 0 για να πάτε πίσω: ");

                                    sQuantity = sc.nextLine();

                                    quantity = Double.parseDouble(sQuantity);

                                    if(quantity==0){AddRequest(b, o);}

                                    else if(requestedRd != null && receivedRd == null){

                                        chosenRd.setQuantity(requestedRd.getQuantity() + quantity);

                                    }

                                    else if(receivedRd != null && requestedRd == null){

                                        chosenRd.setQuantity(receivedRd.getQuantity() + quantity);

                                    }

                                    else if(requestedRd != null && receivedRd != null){

                                        chosenRd.setQuantity(requestedRd.getQuantity() + receivedRd.getQuantity() + quantity);

                                    }

                                    else{chosenRd.setQuantity(quantity);}

                                }

                                while(quantity<0);

                                String YorN;

                                do{

                                    System.out.println("\nΕίστε σίγουροι οτι θέλετε να κάνετε αίτημα για το συγκεκριμένο υλικό;");

                                    System.out.print("Αν είστε πατήστε 'y' αν δεν είστε πατήστε 'n': ");

                                    YorN = sc.nextLine();

                                    switch(YorN){

                                        case "y":

                                            b.addToRequestsList(o, chosenRd); //prosthiki stin lista twn requests tou beneficiary

                                        break;

                                        case "n": AddRequest(b, o); break;

                                        default: System.out.println("Λάθος επιλογή...");

                                    }

                                }

                                while(!YorN.equals("y") && !YorN.equals("n"));

                                break;

                            }

                            else{System.out.println("Λάθος επιλογή...");}

                        }

                        while(matChoice<0 || matChoice>matList.size());

                        break;

                    }

                    else if(choice==2){

                        ArrayList<RequestDonation> servList;

                        servList = o.getCurrentDonations().listServiceRdEntities();

                        if(servList.isEmpty()){

                            System.out.println("\nΔεν παρέχεται καμία υπηρεσία αυτή την στιγμή.\n");

                            AddRequest(b, o);

                        }

                        int servChoice = -1;

                        do{

                            System.out.println("0: Back");

                            System.out.print("\nΕπιλέξτε ποια υπηρεσία επιθυμείτε: ");

                            String sServChoice = sc.nextLine();

                            servChoice = Integer.parseInt(sServChoice);

                            if(servChoice==0){AddRequest(b, o);}

                            else if(servChoice>0 && servChoice<=servList.size()){

                                RequestDonation chosenRd = (RequestDonation)servList.get(servChoice-1).clone();//anitigrafo tou RequestDonation pou dialekse

                                //diatrexv tin requests list gia na dw an exei zitisei to service idi

                                RequestDonation requestedRd = null;

                                for(RequestDonation rd : b.getRequestsList().getRdEntities()){

                                    if(rd.getId()==chosenRd.getId()){

                                        requestedRd = rd;

                                    }

                                }

                                //diatrexw tin current donations gia na dw poso exei lavei idi apo to service

                                RequestDonation receivedRd = null;

                                for(RequestDonation rd : b.getReceivedList().getRdEntities()){

                                    if(rd.getId()==chosenRd.getId()){

                                        receivedRd = rd;

                                    }

                                }

                                System.out.println(servChoice + ". " + chosenRd.getEntity().getName() + ": ");

                                System.out.println(chosenRd.getEntity().toString());

                                double quantity = -1;

                                if(requestedRd != null){

                                    System.out.println("Ώρες που έχετε ζητήσει: " + requestedRd.getQuantity());

                                }

                                if(receivedRd != null){

                                    System.out.println("Ώρες που έχετε λάβει: " + receivedRd.getQuantity());

                                }

                                System.out.println("Διαθέσιμες ώρες στον οργανισμό: " + chosenRd.getQuantity());

                                chosenRd.setQuantity(0);

                                do{

                                    System.out.println("0: Back\n");

                                    System.out.print("Δώστε τις ώρες που επιθυμείτε ή 0 για να πάτε πίσω: ");

                                    String sQuantity = sc.nextLine();

                                    quantity = Double.parseDouble(sQuantity);

                                    if(quantity==0){AddRequest(b, o);}

                                    else if(requestedRd != null && receivedRd == null){

                                        chosenRd.setQuantity(requestedRd.getQuantity() + quantity);

                                    }

                                    else if(receivedRd != null && requestedRd == null){

                                        chosenRd.setQuantity(receivedRd.getQuantity() + quantity);

                                    }

                                    else if(requestedRd != null && receivedRd != null){

                                        chosenRd.setQuantity(requestedRd.getQuantity() + receivedRd.getQuantity() + quantity);

                                    }

                                    else{chosenRd.setQuantity(quantity);}

                                }

                                while(quantity<0);

                                String YorN;

                                do{

                                    System.out.println("\nΕίστε σίγουροι οτι θέλετε να κάνετε αίτημα για την συγκεκριμένη υπηρεσία;");

                                    System.out.print("Πατήστε 'y' ή 'n' αν είστε ή όχι: ");

                                    YorN = sc.nextLine();

                                    switch(YorN){

                                        case "y":

                                            b.addToRequestsList(o, chosenRd); //prosthiki stin lista twn requests tou beneficiary

                                            break;

                                        case "n": AddRequest(b, o); break;

                                        default: System.out.println("Λάθος επιλογή...");

                                    }

                                }

                                while(!YorN.equals("y") && !YorN.equals("n"));

                            }

                            else{System.out.println("Λάθος επιλογή...");}

                        }

                        while(servChoice<1 || servChoice>servList.size());

                        break;

                    }

                    else if(choice==3){

                        LoginBen\_Back(b, o);

                        break;

                    }

                    else{System.out.println("Λάθος επιλογή...");}

                }

                while(choice!=1 && choice!=2 && choice!=3);

                LoginBen\_Back(b, o);

            }

            catch(NumberFormatException e){

                System.out.println("\nΜΗΝ ΔΙΝΕΤΕ ΑΚΥΡΟΥΣ ΧΑΡΑΚΤΗΡΕΣ\n");

                AddRequest(b, o);

            }

        }

    }

    public void ShowRequests(Beneficiary b, Organization o){

        ArrayList <RequestDonation> requestsList = b.getRequestsList().getRdEntities();

        System.out.println("\nΤα είδη που έχετε ζητήσει: ");

        System.out.println();

        b.listRequests();

        if(requestsList.isEmpty()){LoginBen\_Back(b, o);}

        else{

            //diaxeirisi NumberFormatException gia na apotrepw ton xrisrti na plhktrologei strings i megalous aritmous ekei pou den prepei

            try

            {

                System.out.println();

                System.out.println("1: Επιλογή ενός αιτήματος για διαγραφή ή τροποποίηση");

                System.out.println("2: Καθαρισμός όλων των αιτημάτων");

                System.out.println("3: Commit");

                System.out.println("4: Back");

                int choice = -1;

                do{

                    System.out.print("\nΕπιλέξτε ένα απο τα παραπάνω: ");

                    String sChoice = sc.nextLine();

                    choice = Integer.parseInt(sChoice);

                    if(choice==1){

                        int requestChoice = -1;

                        RequestDonation chosenRequest = null;

                        RequestDonation chosenOffer = null;

                        do{

                            System.out.print("Παρακαλώ επιλέξτε ένα αίτημα: ");

                            String sRequestChoice = sc.nextLine();

                            requestChoice = Integer.parseInt(sRequestChoice);

                            if(requestChoice>=1 && requestChoice<=requestsList.size()){

                                chosenRequest = requestsList.get(requestChoice-1);

                                //vriskw to antistoixw offer stin currentDonations tou Organismou gia na dw posi posotita exei minei

                                for(RequestDonation rd: o.getCurrentDonations().getRdEntities()){

                                    if(rd.getId()==chosenRequest.getId()){

                                        chosenOffer = (RequestDonation)rd.clone();

                                    }

                                }

                                System.out.println(requestChoice + ". " + chosenRequest.getEntity().getName() + ":");

                                System.out.println(chosenRequest.getEntity().toString());

                                if(chosenRequest.getEntity().getIsMaterial()){

                                    System.out.println("Ποσότητα που έχετε ζητήσει: " + chosenRequest.getQuantity());

                                    System.out.println("Διαθέσιμη ποσότητα στον οργανισμό: " +chosenOffer.getQuantity());

                                    System.out.println("Ποσοτητα που δικαιούστε: " + b.getQqf(chosenRequest));

                                }

                                else{

                                    System.out.println("Ώρες που έχετε ζητήσει: " + chosenRequest.getQuantity());

                                    System.out.println("Διαθέσιμες ώρες στον οργανισμό: " + chosenOffer.getQuantity());

                                }

                            }

                            else{System.out.println("Λάθος επιλογή...");}

                        }

                        while(requestChoice<1 || requestChoice>b.getRequestsList().getRdEntities().size());

                        System.out.println("\n0: Back");

                        System.out.println("1: Διαγραφή του επιλεγμένου αιτήματος");

                        System.out.println("2: Αλλαγή της ποσότητας του επιλεγμένου αιτήματος");

                        int choice1 = -1;

                        do{

                            System.out.print("Επιλέξτε ένα απο τα παραπάνω: ");

                            String sChoice1 = sc.nextLine();

                            choice1 = Integer.parseInt(sChoice1);

                            if(choice1==1){

                                System.out.println("Γίνεται διαγραφή του αιτήματος: " + requestChoice);

                                b.removeRequest(chosenRequest);

                                break;

                            }

                            else if(choice1==2){

                                double newQuant = -1;

                                do{

                                    System.out.print("Γράψτε την νέα ποσότητα/ώρα που επιθυμείται: ");

                                    String sNewQuant = sc.nextLine();

                                    newQuant = Double.parseDouble(sNewQuant);

                                }

                                while(newQuant<0);

                                b.modifyRequest(o, chosenRequest, newQuant);

                                break;

                            }

                            else if(choice1==0){ShowRequests(b, o); break;}

                            else{System.out.println("Λάθος επιλογή...");}

                        }

                        while(choice1!=1 && choice1!=2 && choice1!=0);

                        break;

                    }

                    else if(choice==2){

                        System.out.println("Είστε σίγουρος/η οτι θέλετε να διαγράψεται όλα τα αιτήματα σας;");

                        String yn;

                        do{

                            System.out.print("Πατήστε 'y' αν είστε και 'n' αν δεν είστε: ");

                            yn = sc.nextLine();

                            if(yn.equals("y")){

                                b.resetRequests();

                                break;

                            }

                            else if(yn.equals("n")){

                                ShowRequests(b, o);

                                break;

                            }

                            else{System.out.println("Λάθος επιλογή...");}

                        }

                        while(!yn.equals("y") && !yn.equals("n"));

                        break;

                    }

                    else if(choice==3){

                        LoginBen\_Commit(b, o);

                        break;

                    }

                    else if(choice==4){

                        LoginBen\_Back(b, o);

                        break;

                    }

                    else{System.out.println("Λάθος επιλογή...");}

                }

                while(choice<1 || choice>4);

                LoginBen\_Back(b, o);

            }

            catch(NumberFormatException e){

                System.out.println("\nΜΗΝ ΔΙΝΕΤΕ ΑΚΥΡΟΥΣ ΧΑΡΑΚΤΗΡΕΣ\n");

                ShowRequests(b, o);

            }

        }

    }

    public void LoginBen\_Commit( Beneficiary b, Organization o){

        System.out.println();

        System.out.println("\nΟλοκλήρωση καταχώρησης αιτημάτων...\n");

        RequestDonationList requestsNotReceived;

        requestsNotReceived = b.commit(o);

        System.out.println("Αιτήματα που ικανοποιήθηκαν επιτυχώς: ");

        b.monitor();

        if(!requestsNotReceived.getRdEntities().isEmpty()){

            System.out.println("Αιτήματα που δεν μπορούν να ικανοποιηθουν:");

            requestsNotReceived.monitor();

        }

        LoginBen\_Back(b, o);

    }

    public void LoginBen\_Back(Beneficiary b, Organization o){

        System.out.println();

        System.out.println("1: Add Request");

        System.out.println("2: Show Requests");

        System.out.println("3: Commit");

        System.out.println("4: Back");

        System.out.println("5: Logout");

        System.out.println("6: Exit");

        try

        {

            String sChoice;

            int choice = -1;

            do{

                System.out.println();

                System.out.print("Δώστε την επιλογή σας: ");

                //amintikos programmatismos gia tin antimetopisi number format exception

                sChoice = sc.nextLine();

                choice = Integer.parseInt(sChoice);

                switch(choice){

                    case 1: AddRequest(b, o); break;

                    case 2: ShowRequests(b, o); break;

                    case 3: LoginBen\_Commit(b, o); break;

                    case 4: Logout(o);

                    case 5: Logout(o);

                    case 6: Exit(); break;

                    default: System.out.println("Λάθος επιλογή...");

                    }

            }

            while(choice<1 || choice>6);

        }

        //apotrepei ton xristi apo to na dinei grammata

        catch(NumberFormatException e){

            System.out.println("\nΜΗΝ ΔΙΝΕΤΕ ΑΚΥΡΟΥΣ ΧΑΡΑΚΤΗΡΕΣ\n");

            LoginBen\_Back(b, o);

        }

    }

    public void Logout(Organization o)

    {

        Login(o);

    }

    public void Exit()

    {

        sc.close();

        System.exit(0);

    }

}

**12.Main.java:**

public class Main

{

    public static void main(String[] args){

        Organization org = new Organization("Τυχαίος Οργανισμός");

        Material gala = new Material("Γάλα", "Πλήρες", 001, 5, 10, 15);

        Material zaxari = new Material("Ζάχαρη", "Κρυσταλική", 002, 2, 5, 7);

        Material rizi = new Material("Ρύζι", "Μπασμάτι", 003, 1, 15, 20);

        Service medicalSupport = new Service("Ιατρική Περίθαλψη", "", 004);

        Service nurserySupport = new Service("Nοσηλευτική Περίθαλψη", "", 005);

        Service babySitting = new Service("Μπέιμπι Σιτινγκ", "", 006);

        RequestDonation rd1 = new RequestDonation(gala, 6);

        RequestDonation rd2 = new RequestDonation(rizi, 8);

        RequestDonation rd3 = new RequestDonation(medicalSupport, 48);

        RequestDonation rd4 = new RequestDonation(zaxari, 5);

        try{

            org.addEntity(gala);

            org.addEntity(rizi);

            org.addEntity(zaxari);

            org.addEntity(medicalSupport);

            org.addEntity(nurserySupport);

            org.addEntity(babySitting);

        }

        catch(EntityAlreadyExistsException e){e.print();}

        Admin admin = new Admin("Γρηγόρης", "123456");

        org.setAdmin(admin);

        Beneficiary ben1 = new Beneficiary("Γιάννης", "555", 3);

        Beneficiary ben2 = new Beneficiary("Γιώργος", "222", 5);

        try{

            org.insertBeneficiary(ben1);

            org.insertBeneficiary(ben2);

        }

        catch(BeneficiaryAlreadyExistsException e){e.print();}

        Donator don1 = new Donator("Τάσος", "333");

        try{org.insertDonator(don1);}

        catch(DonatorAlreadyExistsException e){e.print();}

        try{

            //vazw duo requests sta currentdonations tou org

            org.getCurrentDonations().add(org, rd1);

            org.getCurrentDonations().add(org, rd2);

            //vazw duo requests gia ton ben1

            RequestDonation rd1\_1 = (RequestDonation)rd1.clone();

            RequestDonation rd2\_1 = (RequestDonation)rd2.clone();

            ben1.addToRequestsList(org, rd1\_1);

            ben1.addToRequestsList(org, rd2\_1);

            //vazw duo offers sto offerslist tou don1

            don1.add(org, rd3);

            don1.add(org, rd4);

        }

        catch(EntityDoesNotExistException e){e.print();}

        catch(EntityAmountDoesNotExistException e){e.print();}

        catch(InvalidRequestException e){e.print();}

        Menu menu = new Menu();

        menu.Login(org);

    }

}

**13.Organization.java:**

import java.util.ArrayList;

public class Organization {

    private String name;

    private Admin admin;

    private ArrayList<Entity> entityList = new ArrayList<Entity>(); //lista apo entities pou mporoun na dianemithoun se beneficiaries

    private ArrayList<Donator> donatorList = new ArrayList<Donator>(); //lista apo dorites

    private ArrayList<Beneficiary> beneficiaryList = new ArrayList<Beneficiary>(); //lista apo beneficiaries

    private RequestDonationList currentDonations = new RequestDonationList(); //diathesimes prosfores gia entities kai posotites tous

    Organization(String name, Admin admin, ArrayList<Entity> entityList, ArrayList<Donator> donatorList, ArrayList<Beneficiary> beneficiaryList, RequestDonationList currentDonations){

        this.name = name;

        this.admin = admin;

        this.entityList = entityList;

        this.donatorList = donatorList;

        this.beneficiaryList = beneficiaryList;

        this.currentDonations = currentDonations;

    }

    Organization(String name){

        this.name = name;

    }

    Organization(){}

    public String getName(){

        return name;

    }

    public ArrayList<Entity> getEntityList(){

        return entityList;

    }

    public ArrayList<Donator> getDonatorList(){

        return donatorList;

    }

    public ArrayList<Beneficiary> getBeneficiaryList(){

        return beneficiaryList;

    }

    public RequestDonationList getCurrentDonations(){

        return currentDonations;

    }

    public Admin getAdmin() {

        return admin;

    }

    public void setAdmin(Admin admin) {

        this.admin = admin;

    }

    public void addEntity(Entity entity)throws EntityAlreadyExistsException{

        boolean entityExists = false;

        for(int i=0; i<entityList.size(); i++){

            if(entity.equals(entityList.get(i))){

                entityExists = true;

                throw new EntityAlreadyExistsException();

            }

        }

        if(entityExists==false){entityList.add(entity);}

    }

    public void removeEntity(Entity entity){

        entityList.remove(entity);

    }

    public void insertDonator(Donator donator)throws DonatorAlreadyExistsException {

        if(donatorList.isEmpty()){donatorList.add(donator);}

        else{

            boolean donatorExists = false;

            for(int i=0; i<donatorList.size(); i++){

                if(donator==donatorList.get(i)){

                    donatorExists = true;

                    throw new DonatorAlreadyExistsException();

                }

            }

            if(donatorExists==false){donatorList.add(donator);}

        }

    }

    public void removeDonator(Donator donator){

        donatorList.remove(donator);

    }

    public void insertBeneficiary(Beneficiary beneficiary)throws BeneficiaryAlreadyExistsException{

        if(beneficiaryList.isEmpty()){beneficiaryList.add(beneficiary);}

        else{

            boolean benExists = false;

            for(int i=0; i<beneficiaryList.size(); i++){

                if(beneficiary==beneficiaryList.get(i)){

                    benExists = true;

                    throw new BeneficiaryAlreadyExistsException();

                }

            }

            if(benExists==false){beneficiaryList.add(beneficiary);}

        }

    }

    public void removeBeneficiary(Beneficiary beneficiary){

        beneficiaryList.remove(beneficiary);

    }

    public void listEntities(){

        System.out.println("    Υλικά");

        System.out.println("---------------");

        for(int i=0; i<entityList.size(); i++){

            if(entityList.get(i).getIsMaterial()==true){

                System.out.println(entityList.get(i).toString());

            }

        }

        System.out.println();

        System.out.println("   Services");

        System.out.println("---------------");

        for(int i=0; i<entityList.size(); i++){

            if(entityList.get(i).getIsMaterial()==false){

                System.out.println(entityList.get(i).toString());

            }

        }

    }

    public void listBeneficiaries(){

        for(int i=0; i<beneficiaryList.size(); i++){

            System.out.println(beneficiaryList.get(i).getName() + ":");

            beneficiaryList.get(i).getReceivedList().listMaterialRdEntities();

            beneficiaryList.get(i).getReceivedList().listServiceRdEntities();

        }

    }

    public void listDonators(){

        for (Donator don:donatorList)

            System.out.println(don.getName()+"\n");

    }

    //proairetikes wrappers methodoi gia tin current donations

}

**14.RequestDonation.java:**

public class RequestDonation implements Cloneable //gia na apofugoume CloneNotSupportedException

{

    private Entity entity; //to id(entity.getId()) tou entity einai "kleidi" na ginei xrisi getter gia access

    private double quantity;

    RequestDonation(Entity entity, double quantity){

        this.entity = entity;

        this.quantity = quantity;

    }

    public int getId(){

        return entity.getId();

    }

    public Entity getEntity(){

        return entity;

    }

    public double getQuantity(){

        return quantity;

    }

    public void setEntity(Entity ent){

        this.entity = ent;

    }

    public void setQuantity(double quant){

        if(quant>=0){

            this.quantity = quant;

        }

    }

    public Object clone(){

        Object obj = null;

        try{obj = super.clone();}

        catch(CloneNotSupportedException e){}

        return obj;

    }

}

**15.** **BeneficiaryAlreadyExistsException.java:**

public class BeneficiaryAlreadyExistsException extends Exception

{

    public void print(){

        System.out.println("Aυτός ο επωφελούμενος υπάρχει ήδη στον οργανισμό");

    }

}

**16.** **DonatorAlreadyExistsException.java:**

public class DonatorAlreadyExistsException extends Exception

{

    public void print(){

        System.out.println("Αυτος ο δωρητής υπάρχει ήδη στον οργανισμό");

    }

}

**17.** **EntityAlreadyExistsException.java:**

public class EntityAlreadyExistsException extends Exception

{

    public void print(){

        System.out.println("Αυτο/η το υλικό/υπηρεσία παρέχεται ήδη απο τον οργανισμό");

    }

}

**18.** **EntityAmountDoesNotExistException.java:**

public class EntityAmountDoesNotExistException extends Exception

{

    public void print(){

        System.out.println("To αίτημα σας δεν μπορεί να πραγματοποιηθεί, επειδή η ποσότητα του/της υλικού/υπηρεσίας δεν είναι διαθέσιμη");

    }

}

**19.** **EntityDoesNotExistException.java:**

public class EntityDoesNotExistException extends Exception

{

    public void print(){

        System.out.println("\nΣυγνώμη αλλά αυτό/ή το/η υλικό/υπηρεσία δεν παρέχεται απο τον οργανισμό.");

    }

}

**20.** **InvalidRequestException.java:**

public class InvalidRequestException extends Exception

{

    public void print(){

        System.out.println("To αίτημα σας δεν μπορεί να πραγματοποιηθεί, επειδή δεν τηρείται τις κατάληλες προυποθέσεις");

    }

}

**21.** **OfferDoesNotExistException.java:**

public class OfferDoesNotExistException extends Exception{

    public void print(){

        System.out.print("Δεν υπάρχει η συγκεκριμένη παροχή στην λίστα.");

    }

}