



ΕΛΛΗΝΙΚΗ ΔΗΜΟΚΡΑΤΙΑ  
Εθνικό και Καποδιστριακό  
Πανεπιστήμιο Αθηνών



ΤΜΗΜΑ  
ΠΛΗΡΟΦΟΡΙΚΗΣ &  
ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

## **ΕΘΝΙΚΟ ΚΑΙ ΚΑΠΟΔΙΣΤΡΙΑΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ**

### **ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ**

**Μεταπτυχιακό Πρόγραμμα Σπουδών στην Πληροφορική  
Διαχείριση Δεδομένων, Πληροφορίας και Γνώσης**

**M166 (M115): Τεχνολογία Ηλεκτρονικού Εμπορίου  
Εξάμηνο: Εαρινό '19**

**Ανάπτυξη Εφαρμογής Ηλεκτρονικών Δημοπρασιών  
σε συσκευές με λειτουργικό σύστημα Android**

Κυριάκος Χριστοδούλου - cs2180019  
Δημήτρης Φλουρής - cs2180023

# Πίνακας Περιεχομένων

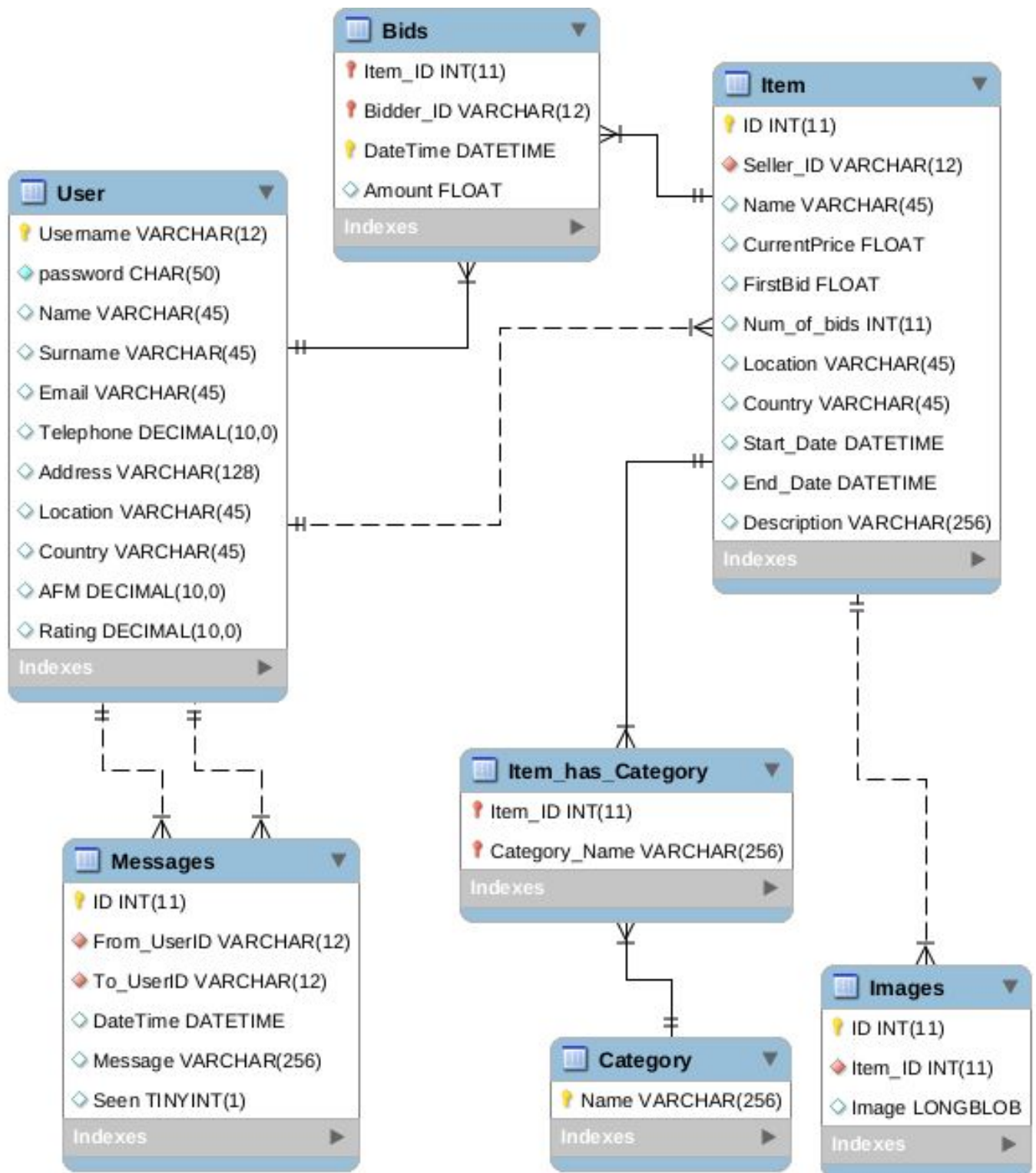
Εισαγωγή	3
Σχεσιακή Βάση Δεδομένων - MySQL	4
Glassfish Server 5.1 & RESTful Web Services	6
Android Application Development	9
Επίλογος	10

## Εισαγωγή

Στόχος αυτής της εργασίας ήταν η ανάπτυξη εφαρμογής ηλεκτρονικών δημοπρασιών σε συσκευές με λειτουργικό σύστημα Android. Στην εφαρμογή συνδέονται οι εγγεγραμμένοι χρήστες που έχουν ταυτόχρονα 2 ρόλους: Πωλητή & Προσφοροδότη (bidder), ενώ μπορούν επίσης να εισέρχονται και επισκέπτες για τους οποίους δεν απαιτείται εγγραφή. Κάθε χρήστης που εγγράφεται στην εφαρμογή μπορεί να θέσει με το ρόλο του Πωλητή ένα ή περισσότερα αντικείμενα σε δημοπρασία ενώ με το ρόλο του Προσφοροδότη μπορεί να πλοηγηθεί και να αναζητήσει αντικείμενα που έχουν θέσει σε δημοπρασία άλλοι χρήστες προκειμένου να υποβάλλει προσφορές. Οι επισκέπτες που εισέρχονται στην εφαρμογή μπορούν μόνο να πλοηγηθούν σε αυτή και να αναζητήσουν αντικείμενα που έχουν τεθεί σε δημοπρασία, χωρίς όμως να μπορούν να υποβάλλουν προσφορές. Επίσης, δίνεται η δυνατότητα στους πωλητές να συνομιλήσουν με το νικητή της εκάστοτε δημοπρασίας μέσω μηνυμάτων.

Στο κεφάλαιο 1 θα παρουσιάσουμε την σχεσιακή βάση δεδομένων που δημιουργήσαμε για τους σκοπούς της παρούσας εργασίας, ενώ στο κεφάλαιο 2 θα ασχοληθούμε με τις υπηρεσίες ιστού που αναπτύξαμε και πως αυτές προσφέρονται από τον εξυπηρετητή που έχουμε στήσει. Στο τρίτο και τελευταίο κεφάλαιο θα αναλύσουμε την διεπαφή χρήστη που προσφέρει η Android εφαρμογή μας και πως επιτυγχάνονται οι επιθυμητές λειτουργίες. Καταλήγοντας, θα αναφέρουμε κάποιες από τις δυσκολίες που αντιμετωπίσαμε κατά την υλοποίηση της παρούσας εργασίας και τους τρόπους επίλυσής τους.

## Σχεσιακή Βάση Δεδομένων - MySQL



Για την παρούσα εργασία χρειάστηκε να σχεδιάσουμε και να αναπτύξουμε μία σχεσιακή βάση δεδομένων, το οποίο κάναμε με τη χρήση MySQL. Αρχικά, έχουμε τις οντότητες User και Item οι οποίες συνδέονται με μία σχέση Ένα-Προς-Πολλά, καθώς κάθε χρήστης μπορεί να δημιουργήσει πολλές δημοπρασίες. Ακόμη, συσχετίζονται με μία σχέση Πολλά-Προς-Πολλά, αφού κάθε χρήστης μπορεί να δώσει προσφορά (Bids) σε πολλά αντικείμενα, όμως κάθε Αντικείμενο μπορεί να λάβει προσφορές από πολλούς χρήστες. Επίσης, οι χρήστες μπορούν να ανταλλάξουν μηνύματα μεταξύ τους, το οποίο αναπαριστούμε με μία σχέση Πολλά-Προς-Πολλά (Messages) ενώ τα αντικείμενα προς δημοπρασία σχετίζονται με μία σχέση Πολλά-Προς-Πολλά (Item\_has\_Category) με την οντότητα Category καθώς το καθένα μπορεί να ανήκει σε πολλές κατηγορίες όπως και η κάθε κατηγορία μπορεί να περιέχει πολλά αντικείμενα. Τέλος, κάθε αντικείμενο προαιρετικά συνδέεται με τα Images με μία Ένα-προς-Πολλά συσχέτιση καθώς μπορεί να έχει πολλές φωτογραφίες.

Όσον αφορά τα πρωτεύοντα κλειδιά της κάθε οντότητας, οι Users μπορούν να χαρακτηριστούν μοναδικά από το Username τους, τα Items από το ID τους, τα Bids από το ID του αντικειμένου και του χρήστη που έκανε την προσφορά καθώς και από την ημερομηνία και ώρα της προσφοράς, αφού κάθε χρήστης μπορεί να κάνει περισσότερες από μία προσφορές. Τα μηνύματα από το ID των χρηστών (παραλήπτη και αποστολέα) καθώς και το ID του μηνύματος ενώ οι κατηγορίες μόνο από το όνομά τους και οι φωτογραφίες από το ID τους. Τέλος, ο πίνακας Item\_has\_Category περιέχει μόνο το ID του αντικειμένου και το όνομα της κατηγορίας προσδιορίζοντας έτσι μοναδικά την κάθε εγγραφή.

## Glassfish Server 5.1 & RESTful Web Services

Για την υλοποίηση της εφαρμογής χρειάστηκε αρχικά να υλοποιήσουμε τις κατάλληλες υπηρεσίες ιστού RESTful που θα επιτρέπουν στην εφαρμογή μας να έχει πρόσβαση στα δεδομένα της βάσης. Οι υπηρεσίες αυτές αναπτύχθηκαν με τη βοήθεια της γλώσσας προγραμματισμού Java, η οποία μας επέτρεψε την μοντελοποίηση των οντοτήτων σε αντικείμενα. Οι συγκεκριμένες υπηρεσίες προσφέρονται από τον τοπικό Glassfish Server 5.1, ενώ όλες οι συναλλαγές μεταξύ της κινητής εφαρμογής και του εξυπηρετητή ιστού προσφέρονται ως υπηρεσίες και μόνο.

Ακολουθεί μια λίστα με τις οντότητες που δημιουργήσαμε βασισμένοι στη ΒΔ που είχαμε φτιάξει και ποιες αλλαγές χρειάστηκε να κάνουμε σε αυτές και στις υπηρεσίες που έχουν πρόσβαση σε αυτές και δημιουργήθηκαν από το RESTful καθώς και ποιες επιπλέον υπηρεσίες χρειάστηκε να φτιάξουμε από την αρχή.

- User: Χρειάστηκε να τροποποιήσουμε την create έτσι ώστε να γίνεται έλεγχος για ύπαρξη χρήστη με το ίδιο username/ email/ ΑΦΜ και να επιστρέφεται ανάλογος κωδικός λάθους. Επίσης ο έλεγχος για υπάρχον username γίνεται και μέσα από την συνάρτηση checkUsername. Τέλος, προσθέσαμε στις συναρτήσεις edit και remove έλεγχο εγκυρότητας του jwt για να αποφύγουμε τυχόν κακόβουλες επιθέσεις.
- Item: Προσθέσαμε την συνάρτηση addBid η οποία καλείται για να τροποποιήσει τις μεταβλητές numOfBids, currentPrice και να προσθέσει στη λίστα με τις προσφορές του αντικειμένου την

καινούργια προσφορά. Για να επιστραφεί η λίστα με τις προσφορές χρειάστηκε πρώτα να την ταξινομήσουμε με φθίνουσα σειρά ως προς την τιμή της κάθε προσφοράς. Κατά την δημιουργία μιας καινούργιας δημοπρασίας, δημιουργούμε τις κατηγορίες στις οποίες ανήκει, εφόσον δεν υπάρχουν ήδη και ακολούθως εισάγουμε το αντικείμενο στη λίστα τους. Για να το πετύχουμε αυτό, καλούμε την συνάρτηση `merge` του `Entity Manager` για τη δημιουργία της κατηγορίας για να χειριστούμε το ενδεχόμενο ύπαρξής της ήδη. Για την διαγραφή μιας δημοπρασίας ελέγχουμε πρώτα το `numOfBids` να ισούται με 0. Τροποποιήσαμε επίσης την `findAll` να επιστρέφει τα αντικείμενα χωρίς τις φωτογραφίες τους καθώς αυτό απαιτεί αρκετό χρόνο και αντ'αυτού δημιουργήσαμε την `fetchImage` η οποία επιστρέφει τις φωτογραφίες ενός συγκεκριμένου αντικειμένου και καλείται μόνο όταν αυτό είναι απαραίτητο. Επίσης, φτιάξαμε την `findBySeller` για να βρίσκουμε τις δημοπρασίες που δεν έχουν λήξει ή δεν έχουν ξεκινήσει ακόμη κάποιου συγκεκριμένου χρήστη, καθώς και την `findByDesc` η οποία ψάχνει για δημοπρασίες που περιέχουν τους όρους αναζήτησης στο όνομα ή την περιγραφή τους και είναι ενεργές. Τόσο η `create` όσο και η `edit` και `remove` απαιτούν στις παραμέτρους τους `jwt` για να γίνει η επιβεβαίωση του χρήστη.

- **Category:** Εδώ χρειάστηκε μόνο η δημιουργία της `find_itCategory` η οποία επιστρέφει τις δημοπρασίες που ανήκουν στην συγκεκριμένη κατηγορία και είναι ενεργές.
- **Bids:** Αρχικά υλοποιήσαμε την `compareTo` έτσι ώστε να μπορούμε να συγκρίνουμε τις προσφορές βάση τιμής ευκολότερα. Η `create`, αφού επιβεβαιώσει την ταυτότητα του χρήστη, ελέγχει την τιμή της

καινούργιας προσφοράς αν είναι μεγαλύτερη από την τρέχουσα τιμή και τότε αφού προστεθεί η προσφορά στη λίστα του αντικειμένου, καλούμε την merge στο ίδιο το αντικείμενο για να προκαλέσει τις αλλαγές που θέλουμε στη ΒΔ.

- Images: Για την είσοδο μιας φωτογραφίας στη ΒΔ τροποποιήσαμε την create έτσι ώστε να δέχεται δεδομένα τύπου MultiPart Form Data αντί JSON και σαν Header Parameter και πάλι το jwt. Αφού το ελέγξει μετατρέπει την εισερχόμενη εικόνα σε bytes array και δίνει τιμές στο αντικείμενο τύπου Image που θα δημιουργήσει.
- Messages: Για την αποστολή μηνύματος καλείται η create, στην οποία αφού πιστοποιηθεί η ταυτότητα του αποστολέα, τότε γίνεται ένας σύνθετος έλεγχος για να διαφανεί αν μπορούν οι 2 χρήστες να ανταλλάξουν μηνύματα. Αυτό γίνεται μέσω ενός ερωτήματος που γίνεται στη ΒΔ το οποίο ελέγχει αν κάποιος από τους δύο χρήστες έχει κερδίσει τουλάχιστον μια δημοπρασία που έχει δημιουργήσει ο άλλος. Αν αυτό ισχύει τότε καλείται κλασσικά η super.create. Πιστοποίηση γίνεται επίσης στις edit/remove καθώς και στις υπόλοιπες συναρτήσεις όπως είναι η find (ένα συγκεκριμένο μήνυμα), η findFrom (όλα τα μηνύματα από κάποιον χρήστη), η findTo (όλα τα μηνύματα προς κάποιον συγκεκριμένο χρήστη) καθώς και στην countREST η οποία μετράει τα αδιάβαστα μηνύματα ενός χρήστη.



# Android Application Development

Αφού υλοποιήσαμε τις υπηρεσίες ιστού και ήμασταν σίγουροι για την σωστή και εύρυθμη λειτουργία τους με τη βοήθεια του Postman, ξεκινήσαμε να υλοποιούμε την Android εφαρμογή. Πιο συγκεκριμένα σχεδιάσαμε στο χαρτί της διεπαφής χρήστη, και ακολούθως δημιουργήσαμε τα xml αρχεία τους. Πιο κάτω εμφανίζονται στιγμιότυπα της εφαρμογής:

Αρχικά ο χρήστης ανοίγοντας την εφαρμογή βλέπει την οθόνη όπου μπορεί να βάλει τα στοιχεία του για να μπει στο λογαριασμό του, να κάνει εγγραφή στην εφαρμογή ή να συνεχίσει ως επισκέπτης.

[illegible]

Βάζοντας τα στοιχεία του και πατώντας το κουμπί SIGN IN εμφανίζεται η οθόνη με τις δικές του δημοπρασίες. Από εκεί μπορεί να επιλέξει:

- Να δημιουργήσει μια δημοπρασία επιλέγοντας το κουμπί (+) και συμπληρώνοντας τα απαραίτητα στοιχεία.

eCommerce- Welcome Demetris

MY AUCTIONS

SEARCH

INBOX

OUTBOX

Toyota Yaris

FROM: 10000.0 EUR Not yet started

Samsung Galaxy S10

FROM: 350.0 EUR Not yet started

Samsung TV

FROM: 850.0 EUR Not yet started

Bosch Fridge

FROM: 380.0 EUR Not yet started

Macbook Pro 13

FROM: 850.0 EUR Not yet started

FIFA 20

FROM: 45.0 EUR Not yet started

Backpack

FROM: 30.0 EUR Not yet started

+

← Create Auction

Name of Item

Starting Price

Country

Location

Description of Item

Categories (seperated by comma)

Uplaod Image

CREATE AUCTION

- Να ξεκινήσει αλλα και να διαγράψει καποια προϋπάρχουσα δημοπρασία επιλέγοντας την και πατώντας το ανάλογο κουμπί πάνω δεξιά της οθόνης. Ξεκινώντας μια δημοπρασία ο δημιουργός συμπληρώνει την επιθυμητή ώρα που θέλει να ξεκινήσει και ποια ώρα να λήξει η συγκεκριμένη δημοπρασία.

← Toyota Yaris ▶ ✕

Starting Price: EUR 10000.0

Current Price: -

Started on: -

Ending on: -

Categories: Cars

Description: Toyota Yaris RED

Country: Greece

Location: Athens

Seller: Demetris Rating: null/10

Bids: 0

Provide dates in pattern dd/MM/yyyy HH:mm

Start Date

End Date

START

CANCEL

Ένας χρήστης μπορεί να αναζητήσει ενεργές δημοπρασίες άλλων χρηστών απο τη πιο κάτω οθόνη, επιλέγοντας τη κατηγορία που θέλει από το drop down μενού ή/και πληκτρολογώντας το όνομα της δημοπρασίας.

eCommerce- Welcome Demetris

MY AUCTIONS

SEARCH

INBOX

OUTBOX

Q Search Here

RESET

None

Backpack

Cars

Fridges

Gaming

Laptops

Smartphones

TV

eCommerce- Welcome Kyriakos

MY AUCTIONS

SEARCH


INBOX<sup>1</sup>

OUTBOX

Q Search Here

RESET

Cars



No image available

Toyota Yaris

FROM: 10000.0 EUR UNTIL: Mon Sep 23 05:25:00 GMT+03:00 2019

Για να κάνει κάποιος χρήστης bid σε μια δημοπρασία θα πρετπει αφού την αναζητήσει να πατήσει πάνω της και στη συνέχεια θα εμφανιστεί μια οθόνη για το ποσό που θέλει να πληρώσει και στη συνέχεια μια οθόνη επιβεβαίωσης.

← Toyota Yaris



No image available

Starting Price: EUR 10000.0

Current Price: -

Started on: Mon Sep 23 05:23:00 GMT+03:00 2019

Ending on: Mon Sep 23 05:25:00 GMT+03:00 2019

Categories: Cars

Description: Toyota Yaris RED

Country: Greece

Location: Athens

Seller: Demetris Rating: null/10

Bid Now:  EUR

BID NOW

Bids: 0

## Submit your bid

Are you sure you want to bid for this item?

CANCEL

BID

Αφού λήξει μια δημοπρασία ο νικητής/πωλητής μπορεί να επικοινωνήσει μέσω μηνύματος με τον νικητή/πωλητή. Στην οθόνη outbox πατώντας το κουμπί (+) μπορεί να συντάξει το μήνυμα που επιθυμεί και να το στείλει στο χρήστη που θέλει.

### eCommerce- Welcome Kyriakos

MY  
AUCTIONS

SEARCH

INBOX

OUTBOX

TO: Demetris

Message: Do you need extra fee for shipping?

### ← Create Message

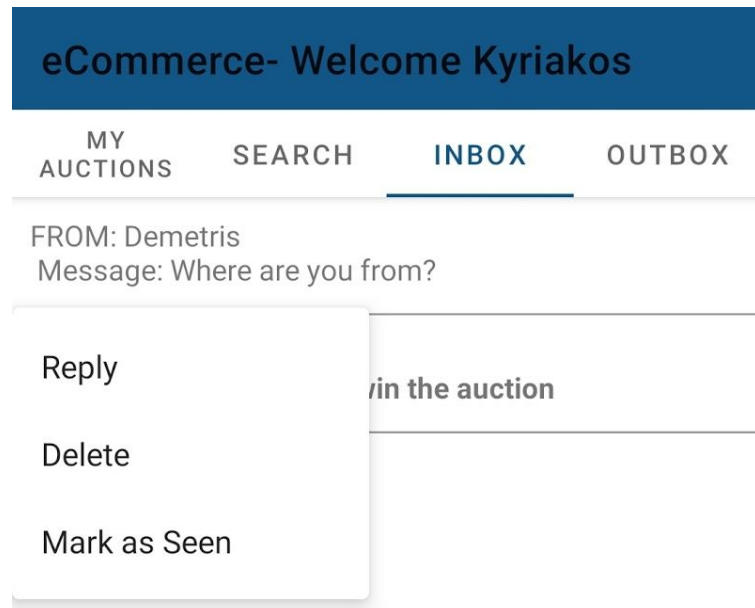
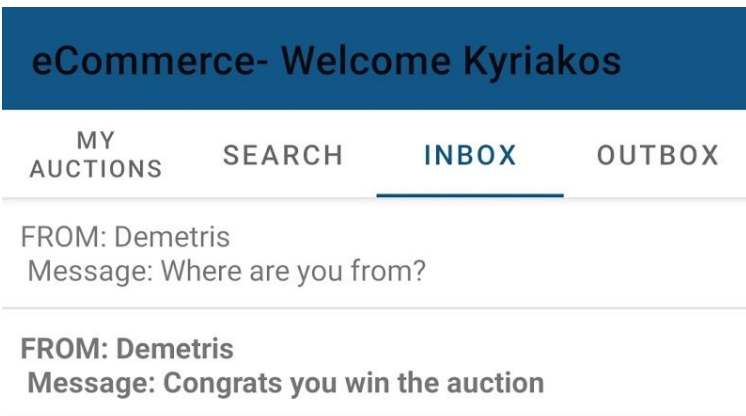
To:

Text message



SEND

Στη συνέχεια ο παραλήπτης του μηνύματος θα μπορεί να δει το εισερχόμενο μήνυμα στην οθόνη inbox. Από εκεί πατώντας πάνω στο μήνυμα εμφανίζεται ένα drop down μενού όπου μπορεί να απαντήσει στον αποστολέα να διαγράψει το μήνυμα ή να το μαρκάρει σαν αναγνωσμένο όπου και γίνεται unbold.



## Επίλογος

Η συγκεκριμένη εργασία για μας ήταν μια πρώτη επαφή με την ανάπτυξη εφαρμογών Android. Δουλέψαμε μεθοδικά σχεδιάζοντας πρώτα προσεκτικά την Βάση Δεδομένων ούτως ώστε τυχόν αλλαγές στην πορεία να μην μας επηρεάσουν δραματικά. Ακολούθως, υλοποιήσαμε τις υπηρεσίες ιστού RESTful στον Glassfish 5.1 δοκιμάζοντάς τις με το Postman. Αφού καλύψαμε όλες τις πιθανές ανάγκες που σκεφτήκαμε ότι μπορεί να έχουμε από την πλευρά του Server, ξεκινήσαμε την σχεδίαση της διεπαφής χρήστη. Αυτό που μας κέντρισε το ενδιαφέρον κατά την διάρκεια της εργασίας, είναι η δυνατότητα δημιουργίας των υπηρεσιών ιστού χρησιμοποιώντας το σχήμα της Βάσης Δεδομένων με τόσο μεγάλη ευκολία. Αφού γίνει αυτό

σωστά, τότε μπορεί κανείς να εξυπηρετήσει τόσο εφαρμογές Android, όσο και ιστοσελίδες χρησιμοποιώντας την ίδια λογική.

Με την χρήση του GitHub μπορέσαμε να δουλέψουμε εξ' αποστάσεως το οποίο ήταν ιδιαίτερα σημαντικό κατά τη διάρκεια του καλοκαιριού.

Ακόμη, το φροντιστήριο φάνηκε ιδιαίτερα βοηθητικό, ειδικά για εμάς που δεν είχαμε ασχοληθεί ποτέ ξανά με Android development.

Ένα σημείο στο οποίο δυσκολευτήκαμε αρκετά ήταν το ανέβασμα φωτογραφίας και η αποθήκευσή της στη ΒΔ στο οποίο αναγκαστήκαμε να κάνουμε και μία “έκπτωση” στα ζητούμενα της εκφώνησης, καθώς μόνο μία εικόνα επιτρέπουμε στον χρήστη να ανεβάσει για το κάθε αντικείμενο προς δημοπρασία. Η δυσκολία προέκυψε από το γεγονός ότι έπρεπε να δημιουργηθεί πρώτα το αντικείμενο και ακολούθως να ανέβει η φωτογραφία, αφού πρώτα κωδικοποιηθεί, χρησιμοποιώντας το ID που θα δώσει η ΒΔ σαν αναγνωριστικό στο αντικείμενο.