ΕΙΔΙΚΑ ΘΕΜΑΤΑ ΤΕΧΝΟΛΟΓΙΑΣ ΛΟΓΙΣΜΙΚΟΥ

Υλοποίηση έργου

TextSecure

Ομάδα: javac

Μέλη:

Αντμίρ Ντεμίραϊ 8100003

Δανάη Τουσιούδη 8090125

Αρχικά για την υλοποίηση της εργασίας δημιουργήσαμε ένα κοινό λογαριασμό στο github, το οποίο βρίσκεται σε αυτό το link:

- https://github.com/javacTeam

Από τις πρώτες ημέρες ενασχόλησης με το έργο είχαμε επικοινωνία με την ομάδα ανάπτυξης καθώς θέσαμε το ερώτημα με ποιο κομμάτι του έργου να ασχοληθούμε για αρχή και πώς να κάνουμε μια συνεισφορά. Η συμβουλή τους ήταν να δούμε παλαιότερα pull requests που απορρίφθηκαν για να καταλάβουμε ποιες είναι οι απαιτήσεις τους. Μας παρότρυναν να υλοποιήσουμε μια αλλαγή και να την υποβάλουμε με την διαδικασία του fork and pull model και θα μας βοηθούσαν με σχόλια για περαιτέρω βελτιώσεις μέχρι τελικά η συνεισφορά να γίνει δεκτή.

Σκοπός ήταν η εισαγωγή ενός καινούριου feature στις ομαδικές συνομιλίες. Όταν σε μια ομαδική συνομιλία στέλναμε ένα μήνυμα και το λάμβανε έστω και ένα άτομο έδειχνε ότι το μήνυμα αυτό ήταν πλέον απεσταλμένο και στα στοιχεία του μηνύματος έδειχνε όλη την λίστα των ατόμων που βρίσκονταν στην συνομιλία ανεξάρτητα από το αν το είχαν λάβει ή όχι. Μετά την συνεισφορά μας, ένα μήνυμα δείχνει ότι έχει σταλεί μόνο αν το λάβουν όλα τα άτομα της συνομιλίας. Σε περίπτωση που δεν το έχουν λάβει όλοι εμφανίζεται στην φυσαλίδα του μηνύματος ο αριθμός των ατόμων που το έχουν λάβει σε σχέση με τον συνολικό αριθμό των μελών της συνομιλίας, ενώ στα στοιχεία του μηνύματος εμφανίζονται με γκρι φόντο όσοι δεν το έλαβαν. Παρακάτω εξηγούμε ποιες αλλαγές κάναμε στον κώδικα σε κάθε κλάση για να το πετύχουμε.

- ✓ MessageRetrievalService :

Δημιουργήσαμε ένα Map που συνδέει για κάθε timestamp του μηνύματος, δηλαδή την ώρα που στάλθηκε, όλους όσους το έχουν παραλάβει(σε μορφή λίστας που περιλαμβάνει τους αριθμούς των παραληπτών).

- ✓ MessageDetailsActivity:

Δημιουργήσαμε 2 νέες μεθόδους τις isReceived και την isNotReceived που δημιουργούν 2 αντικείμενα τύπου Recipients με όσους έχουν παραλάβει το μήνυμα ή όχι αντίστοιχα. Επειδή παρατηρήσαμε ότι μερικές φορές στη λίστα των Recipients κάποιοι αριθμοί παρουσιάζονται σε διαφορετική μορφή από ότι δίνονται από την MessageRetrievalService δημιουργήσαμε μία νέα μέθοδο, την recipientsCorrected, που μετατρέπει τους αριθμούς στην ίδια μορφή (αφαιρεί κενά/ειδικούς χαρακτήρες).

Στο xml αρχείο message_details_activity δημιουργήσαμε ένα νέο ListView το οποίο παρουσιάζει τα ονόματα όσων δεν το έχουν παραλάβει σε ανοιχτό γκρι φόντο. Το ListView αρχικοποιείται στην μέθοδο initializeRecources όπως και τα υπόλοιπα Listviews.

Τέλος στη μέθοδο updateRecipients γίνεται έλεγχος για το αν το έχουν λάβει όλοι οι παραλήπτες συγκρίνοντας το μέγεθος του αντικειμένου που δημιουργείται από την κλάση isReceived με τον αριθμό όλων των παραληπτών (ο αριθμός των παραληπτών δίνεται από την μέθοδο getRecipientsList().size() η οποία είναι μέθοδος της κλάσης Recipients). Αν το μέγεθος είναι ίδιο που σημαίνει ότι το έχουν λάβει όλοι τότε τα ονόματα των παραληπτών στις πληροφορίες του μηνύματος παρουσιάζονται κανονικά. Σε αντίθετη περίπτωση παρουσιάζονται πρώτα όσοι το έχουν λάβει με το παλιό ListView και στη συνέχεια όσοι δεν το έχουν λάβει με το καινούριο ListView (γκρι φόντο).

- ✓ ConversationItem

Δημιουργείται μία boolean μεταβλητή, η isDeliveredToAll, η οποία ελέγχει αν το μήνυμα έχει παραδοθεί σε όλους τους παραλήπτες. Ο παραπάνω έλεγχος γίνεται με τη μέθοδο checkIfDeliveredToAll που δημιουργήσαμε.

Η μέθοδος setStatusIcons είναι υπεύθυνη για να εμφανίσει την εικόνα με το tick στην φυσαλίδα του μηνύματος η οποία υποδηλώνει ότι παραδόθηκε. Πριν τις αλλαγές η εικόνα εμφανιζόταν ακόμα και αν λάμβανε το μήνυμα έστω και ένα άτομο ανεξάρτητα αν αυτό είχε παραδοθεί σε όλους. Εμείς το σχεδιάσαμε έτσι ώστε να εμφανίζεται μόνο στη περίπτωση που το μήνυμα έχει παραδοθεί σε όλους ελέγχοντας την τιμή της μεταβλητής isDeliveredToAll.

Τέλος θέλαμε να παρουσιάζεται στη φυσαλίδα του μηνύματος ο αριθμός όσων έχουν παραλάβει το μήνυμα, συγκριτικά με τον αριθμό των μελών που συμμετέχουν στην συνομιλία. Αποφασίσαμε ότι ο καλύτερος τρόπος είναι να παρουσιάζεται στην εξής μορφή :

(παραλήπτες) / (μέλη συνομιλίας)

Οι αριθμοί αυτοί αλλάζουν δυναμικά με το που το μήνυμα παραδοθεί σε ένα καινούριο μέλος. Τους παραλήπτες τους παίρνουμε από την κλάση MessageRecord με την μέθοδο getReceiptCount, ενώ τα μέλη της συνομιλίας από PushGroupSendJob με τη μέθοδο getAllRecipients() που δημιουργήσαμε εμείς.

Παρατηρήσαμε ότι με τις παραπάνω αλλαγές παρουσιαζόταν πρόβλημα αν έστελνες μήνυμα σε ένα άτομο. Οπότε προσθέσαμε έναν έλεγχο για το αν η συνομιλία είναι ομαδική ή όχι. Ο έλεγχος αυτός γίνεται στην κλάση ConversationActivity και αποθηκεύει το αποτέλεσμα σε μια boolean μεταβλητή, την isGroupConvAct.

Για να μπορέσουμε να υλοποιήσουμε την συνεισφορά χρειάστηκε να δούμε πως υλοποιείται εις βάθος μια τυπική αποστολή μηνύματος και να μελετήσουμε αρκετές κλάσεις που συνδέονται με αυτή την διαδικασία. Υπήρχε ένα μεγάλο διάστημα που διαβάζαμε τον κώδικα για να τον κατανοήσουμε διότι είχε πολύ λίγα σχόλια. Μόλις βρήκαμε τις μεθόδους που μας ενδιέφεραν σχεδιάσαμε την υλοποίηση του έργου. Κατά την διάρκεια της υλοποίησης, προέκυψαν πολλά προβλήματα λόγω της χρήσης πολλών κλάσεων και της σειράς εκτέλεσης των μεθόδων.

Προσπαθήσαμε κατά το δυνατόν να συμβαδίσουμε τα πρότυπα υλοποίησης της ομάδας ανάπτυξης όσο αναφορά τον τρόπο γραφής του κώδικα (χρήση ήδη υπαρχόντων μεθόδων όπου επιτρεπόταν, σωστή στοίχιση, σωστή ονοματοθεσία μεταβλητών και μεθόδων και επαναχρησιμοποίηση κώδικα στην xml). Όλες οι μέθοδοι και οι περισσότερες μεταβλητές που εισάγαμε τεκμηριώνονται από java docs ή από απλά σχόλια αντίστοιχα για να μπορέσει η ομάδα ανάπτυξης να καταλάβει αμέσως τι υλοποιήσαμε.

Η συνεισφορά του έργου ολοκληρώθηκε με επιτυχία από άποψη λειτουργικότητας και εμφάνισης. Παρ' όλα αυτά δημιουργήθηκαν σε κάποιες ιδιάζουσες περιπτώσεις κάποια minor bugs στην εμφάνιση τα οποία μαζί με τα σχόλια της ομάδας ανάπτυξης ευελπιστούμε ότι θα τα διορθώσουμε σύντομα. Το μέγεθος της συνεισφοράς μας ήταν 163 γραμμές κώδικα και χρειάστηκε να σβήσουμε και 4 γραμμές από τον προϋπάρχοντα κώδικα. Το μέγεθος αυτό θεωρούμε ότι είναι αξιόλογο καθώς είχε αυξημένη πολυπλοκότητα λόγω των συσχετίσεων πολλών κλάσεων και μεθόδων και είχε σαν στόχο την υλοποίηση μιας νέας λειτουργίας.

Για την διαχείριση του έργου εργαστήκαμε με το git. Η διαδικασία που ακολουθήσαμε ήταν ότι αρχικά κάναμε ένα fork του έργου στο δικό μας προφίλ στο github, μετά κλωνοποιήσαμε αυτό το fork τοπικά και υλοποιήσαμε τις όποιες αλλαγές είχαμε να κάνουμε σε ένα δικό μας branch (η μεθοδολογία αυτή προτάθηκε από την ομάδα ανάπτυξης καθώς θα εργαζόμασταν σε ένα καινούργιο feature). Αφότου κάναμε τα διάφορα commits (ελέγχοντας πάντα ότι το έργο μας είναι συγχρονισμένο με το κυρίως έργο-git rebase) κάναμε push στο δικό μας fork και ύστερα κάναμε pull το branch μας στο master του κυρίως έργου. Το όνομα του branch που δημιουργήσαμε είναι το : groupconversation_displaying_receivers_improvement και βρίσκεται στο παρακάτω link:

- https://github.com/javacTeam/TextSecure/tree/groupconversation_displaying_receivers_improvement

όσο αναφορά το pull request η ομάδα ανάπτυξης μας είπε ότι δεν διαθέτει το χρόνο για να το κάνει review εφόσον ενδέχεται να έχει κάποια λάθη και μας παρότρυναν να το ολοκληρώσουμε και να το ξανά υποβάλουμε. Το link είναι:

- https://github.com/WhisperSystems/TextSecure/pull/3357

όσο αναφορά το ιστολόγο της ομάδας όπου έγινε ανάρτηση του τελικού προγράμματος εργασίας βρίσκεται στο εξής link :

- http://javacteam.blogspot.gr/