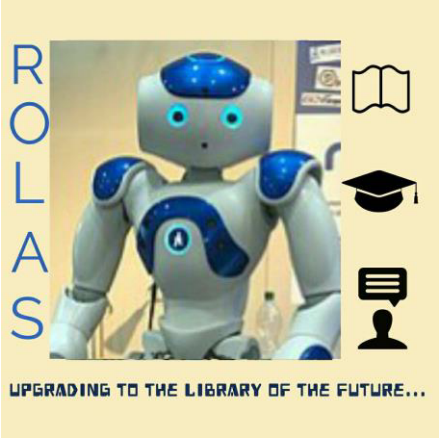
Τεχνολογία Λογισμικού
Τομέας Ηλεκτρονικής και Υπολογιστών
Τμήμα ΗΜΜΥ
Α.Π.Θ.

8ο Εξάμηνο
Άνοιξη 2018



ROboLibrarian Assistant System

Απαιτήσεις Χρηστών

Del.3.1

Version 0.5

Καλαϊτζής Γεώργιος kalaitzg@ece.auth.gr
Καμπελής Ελιέζερ Σολομών eliekamp@ece.auth.gr
Παπαδόπουλος Κωνσταντίνος konserpap@ece.auth.gr
Τοπαλίδης Ευθύμιος eatopalid@ece.auth.gr
Ομάδα 19
24/05/2018

Ιστορικό Αλλαγών

Όνομα	Ημ/νία	Περιγραφή Αλλαγής	Εκδ.
Α. Συμεωνίδης	29/05/2009	Δημιουργία Εγγράφου Προσαρμογή του ESA software engineering standards guidelines (1991) και του εγγράφου SDD document, από τους Bruegge και Dutoit (2004).	0.1
ROLAS	15/05/2018	Συγγραφή κεφαλαίου 1	0.2
ROLAS	17/05/2018	Συγγραφή παραγράφων 2.3-3.0	0.2
ROLAS	23/05/2018	Συγγραφή παραγράφων 2.0-2.2.16	0.4
ROLAS	24/05/2018	Μορφοποίηση εγγράφου	0.5

Μέλη Ομάδας Ανάπτυξης

Μέλη της Ομάδας Ανάπτυξης

Όνομα	ΟΑ	Email
Α. Συμεωνίδης	*	asymeon@issel.ee.auth.gr
Καλαϊτζής Γεώργιος	RO.L.A.S.	kalaitzg@ece.auth.gr
Καμπελής Ελιέζερ Σολομών	RO.L.A.S.	eliekamp@ece.auth.gr
Παπαδόπουλος Κωνσταντίνος	RO.L.A.S.	konserpap@ece.auth.gr
Τοπαλίδης Ευθύμιος	RO.L.A.S.	eatopalid@ece.auth.gr

* **ΕΡΓΑΣΤΗΡΙΑΚΗ ΟΜΑΔΑ** 19

Πίνακας Περιεχομένων

Λίστα Σχημάτων

Εισαγωγικά

0. Εισαγωγή

Το έργο λογισμικού R.O.L.A.S. έχει ως στόχο τον εκσυγχρονισμό της κλασικής δομής διαχείρισης και παροχής υπηρεσιών της βιβλιοθήκης, με αποτέλεσμα την διευκόλυνση τόσο των χρηστών της δομής αυτής όσο και των στελεχών της. Στις προηγούμενες αναφορές, διεκπεραιώθηκε η καταγραφή των απαιτήσεων χρηστών και ο καθορισμός των απαιτήσεων λογισμικού ενώ στο συγκεκριμένο έγγραφο εκπονείται η περιγραφή του συστήματος σε επίπεδο σχεδίασης και αρχιτεκτονικής.

0.1 Στόχος του εγγράφου

Σε αυτό το παραδοτέο στόχος είναι η σχεδίαση του συστήματος της εφαρμογής, χρησιμοποιώντας την κατάλληλη αρχιτεκτονική. Για να συμβεί αυτό πρέπει μα αποδομηθεί το σύστημα σε υποσυστήματα τα οποία περιέχουν κλάσεις που έχουν συνεκτικό εννοιολογικό περιεχόμενο. Στην συνέχεια προδιαγράφονται οι διασυνδέσεις μεταξύ των επιμέρους υποσυστημάτων και περιγράφονται οι ιδιότητες του κάθε τμήματος (διαγράμματα τμημάτων).Επιπλέον, μέσα από τα διαγράμματα ανάπτυξης μοντελοποιούνται τα τμήματα λογισμικού ώστε να γίνεται εκτίμηση των προδιαγραφών των συσκευών αλλά και των πρωτοκόλλων που θα χρησιμοποιηθούν για την επικοινωνία των τμημάτων. Η επιλογή της σωστής αρχιτεκτονικής σε αυτό το σημείο κρίνεται σημαντική καθώς επιδρά στην αξιοπιστία, το κόστος και τον έλεγχο του συστήματος.

0.2 Αναγνωστικό κοινό και τρόπος ανάγνωσης εγγράφου

Η πλήρης κατανόηση του συγκεκριμένου εγγράφου κρίνει επιτακτική την μελέτη των προηγούμενων εγγράφων που αφορούν τις απαιτήσεις χρηστών και τις απαιτήσεις λογισμικού. Το αναγνωστικό κοινό αποτελούν οι εξής ομάδες:

- Σχεδιαστής συστήματος , ο οποίος επικεντρώνεται κυρίως στην λύση της αρχιτεκτονικής που προτάθηκε καθώς επίσης και στις προδιαγραφές του υλικού.
- Προγραμματιστής , ο οποίος επικεντρώνεται στα τμήματα λογισμικού αναλαμβάνοντας την συγγραφή και τον έλεγχο της ορθής λειτουργίας τους.
- Μηχανικός λογισμικού ,που θα αναλάβουν την επέκταση του συστήματος αλλά και την συντήρησή του.

1. **Δυναμική μοντελοποίηση του συστήματος**.

1.1 **Πακέτο επιστροφής βιβλίου από το χρήστη**

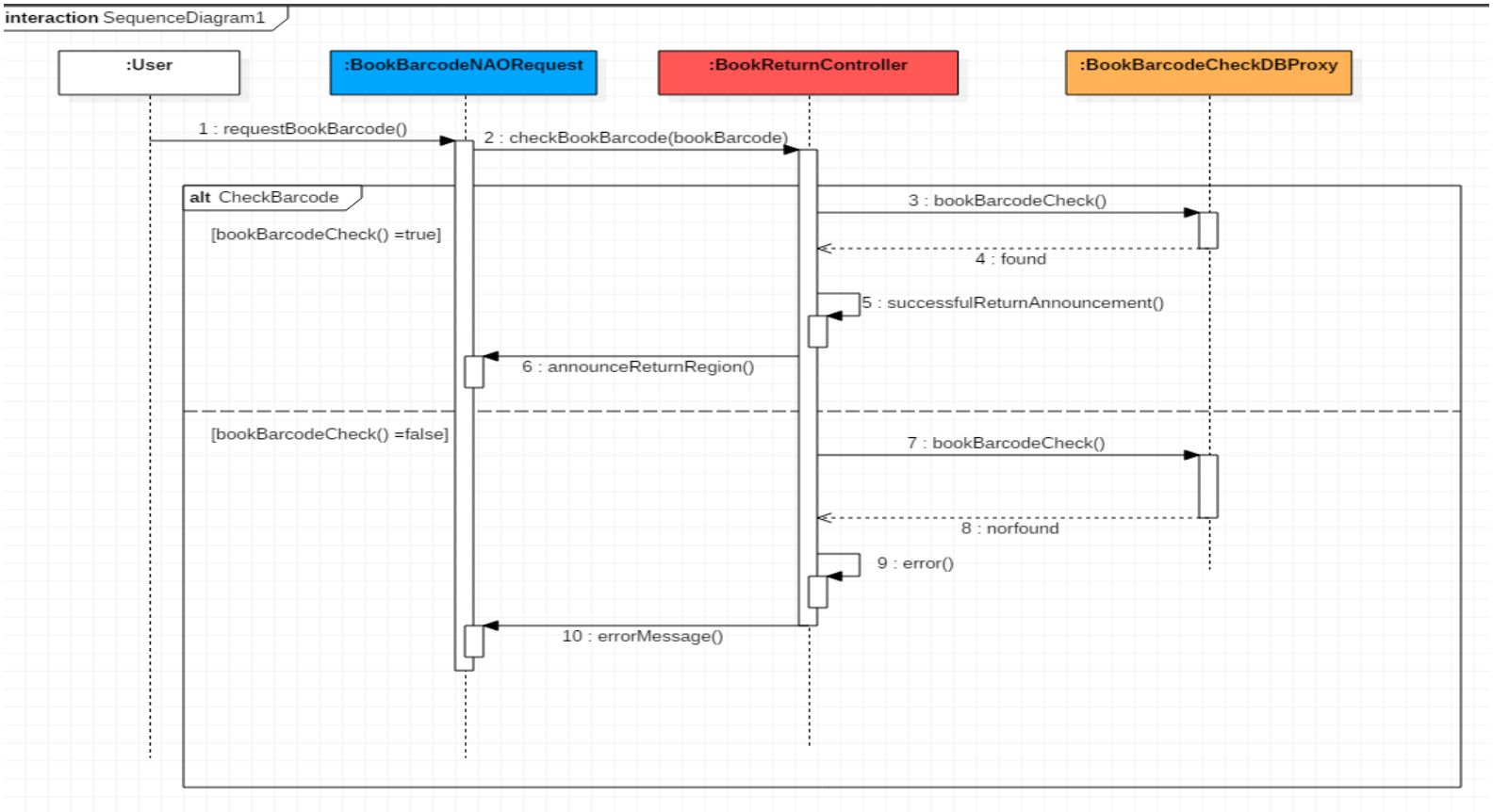
Το πακέτο αυτό περιγράφει την διαδικασία όπου ο εγγεγραμμένος χρήστης της βιβλιοθήκης πραγματοποιεί την επιστροφής βιβλίων που έχει δανειστεί. Η διαδικασία επιστροφής πραγματοποιείται με την επίδειξη του ραβδοκώδικα του βιβλίου στο NAO ώστε να πραγματοποιηθεί από το σύστημα η ταυτοποίηση των στοιχείων του επιστρεφόμενου βιβλίου.

Η λειτουργική απαίτηση που σχετίζεται με αυτό το πακέτο χρήσης είναι:

- <ΛΑ-11> Το σύστημα θα πρέπει να πραγματοποιεί την επιστροφή βιβλίων.

Αφήγηση Σεναρίου

Ο εγγεγραμμένος χρήστης εκφωνεί στο NAO σχετικό μήνυμα που δηλώνει την πρόθεσή του να επιστρέψει κάποιο βιβλίο το οποίο έχει δανειστεί. Στη συνέχεια το σύστημα εκφωνεί (μέσω του NAO) μήνυμα στο οποίο ζητάει από τον εγγεγραμμένο χρήστη να επιδείξει το ραβδοκώδικα του δανεισμένου βιβλίου στο NAO. Η διαδικασία επιστροφής δανεισμένου βιβλίου πραγματοποιείται μέσω της κλάσης BookBarcodeNAORequest που αποτελεί την διεπαφή ανάμεσα στο χρήστη και το σύστημα. Μέσω της μεθόδου requestBookBarcode( ) το σύστημα διαβάζει το ραβδοκώδικα του βιβλίου και στη συνέχεια καλεί την μέθοδο checkBookBarcode(bookBarcode:string) της κλάσης BookReturnController. Η checkBookBarcode() δέχεται ως όρισμα την συμβολοσειρά που αντιπροσωπεύει τον ραβδοκώδικα του βιβλίου και στη συνέχεια καλεί την μέθοδο bookBarcodeCheck() της BookBarcodeCheckDBProxy μέσω της οποίας θα γίνει ο έλεγχος ύπαρξης του βιβλίου μέσω σχετικού ελέγχου του ραβδοκώδικά του στη βάση δεδομένων του συστήματος. Αν ο ραβδοκώδικας του βιβλίου βρεθεί στη βάση δεδομένων τότε η bookBarcodeCheck() επιστρέφει «αληθές» και καλεί την successfulReturnAnnouncement() η οποία με τη σειρά της καλεί την announceReturnRegion(). Μέσω της τελευταίας το σύστημα ανακοινώνει την επιτυχή επιστροφή του βιβλίου και εκφωνεί τον χώρο εναπόθεσης του βιβλίου στον φυσικό χώρο της βιβλιοθήκης μέσω του NAO. Σε περίπτωση που η bookBarcodeCheck() επιστρέψει «ψευδής» καλείται η μέθοδος error() η οποία με τη σειρά της καλεί τη μέθοδο errorMessage() , της κλάσης BookBarcodeNAORequest , και μέσω αυτής εκφωνείται στον χρήστη μήνυμα μη ύπαρξης του ραβδοκώδικα του βιβλίου στη βάση δεδομένων του συστήματος.



1.2 Πακέτο Δανεισμός Βιβλίου

Το πακέτο αυτό περιλαμβάνει τις κλάσεις που μοντελοποιούν τη διαδικασία δανεισμού βιβλίου. Ο χρήστης εισάγει τα στοιχεία του βιβλίου που επιθυμεί να δανειστεί και εφόσον δεν υπάρχουν ληξιπρόθεσμες οφειλές, ολοκληρώνεται με επιτυχία ο δανεισμός.

Οι λειτουργικές απαιτήσεις που σχετίζονται με αυτό το πακέτο χρήσης είναι:

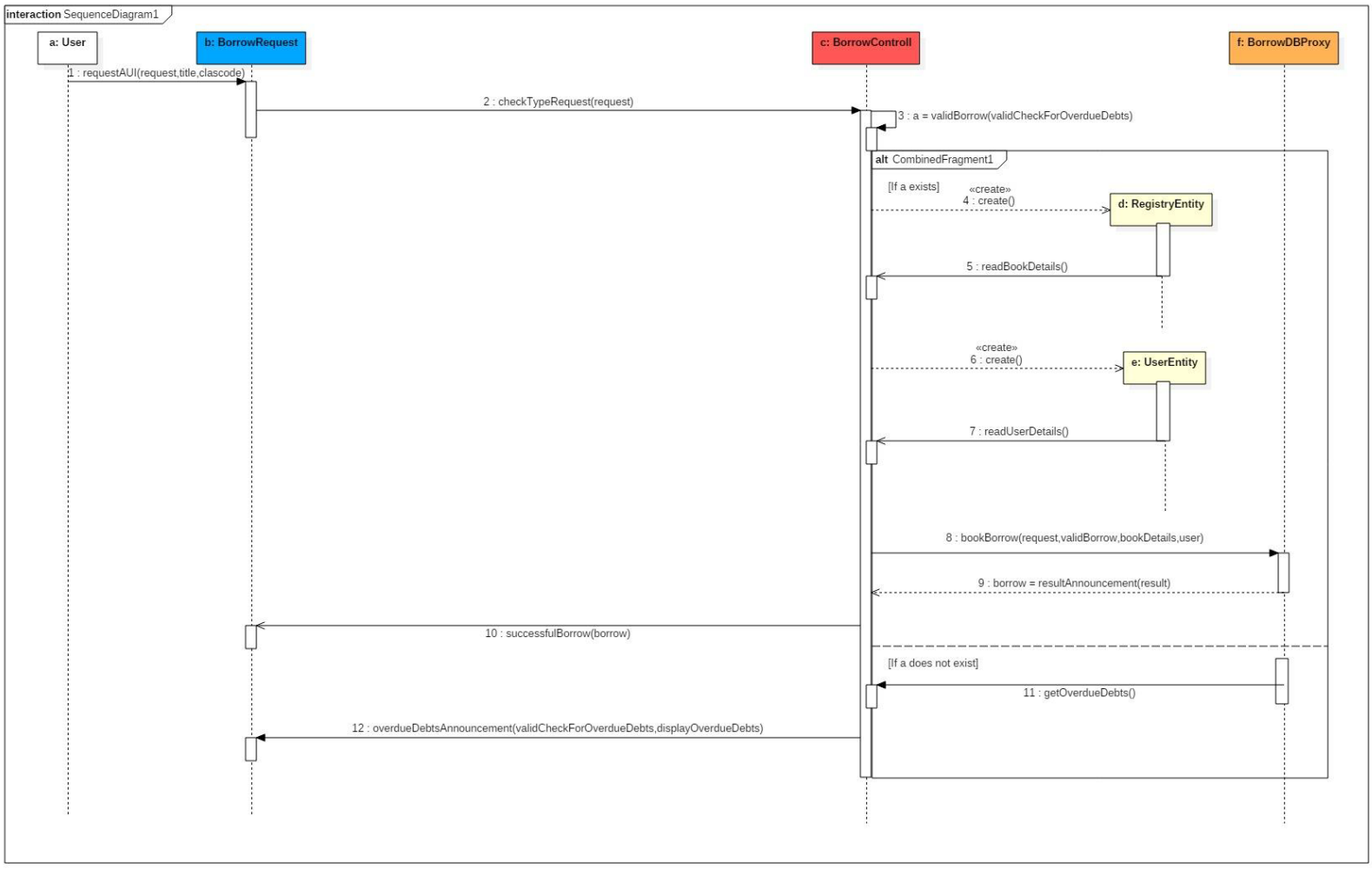
- <ΛΑ-7> Ο εγγεγραμμένος χρήστης πρέπει να έχει τη δυνατότητα να δανειστεί ένα βιβλίο.

Αφήγηση Σεναρίου

Ο χρήστης έχει συνδεθεί στο σύστημα. Έπειτα εκφωνεί την εντολή <<Δανεισμός Βιβλίου>> που γίνεται αντιληπτή από το N.A.O και επεξεργάζεται από τη μέθοδο requestAUI(request, title, clascode) της κλάσης BorrowRequest. Έπειτα ο controller BorrowControll επιβεβαιώνει το είδος του αιτήματος του χρήστη μέσω της μεθόδου checkTypeRequest(request). Η εγκυρότητα του δανεισμού του βιβλίου διασφαλίζεται μέσω της μεθόδου validBorrow(validCheckForOverdueDebts) του controller BorrowControll, όπου αν επιβεβαιωθεί ότι ο χρήστης πληρεί τα κριτήρια για δανεισμό τότε αποθηκεύται η τιμή true στο αντικείμενο a. Τότε ακολουθείται η ροή επιτυχούς δανεισμού όπου ο controller BorrowControll δημιουργεί τα αντικείμενα RegistryEntity, BookEntity και μέσω των μεθόδων readBookDetails(), readUserDetails() αποθηκεύονται τα χαρακτηριστικά του βιβλίου και του αντίστοιχου χρήστη. Στη συνέχεια ο controller θα κατοχυρώσει το δανεισμό μέσω της κλάσης BorrowDBProxy και της μεθόδου bookBorrow(request, validBorrow, bookDetails, user) που αποθηκεύει το δανεισμό στη βάση δεδομένων. Ο επιτυχής δανεισμός ολοκληρώνεται όταν η BorrowDBProxy θα επικοινωνήσει με τον controller για να κληθεί η μέθοδος resultAnnouncement(result) που αποθηκεύει στη μεταβλητή borrow το μήνυμα επιτυχούς δανεισμού που θα εκφωνηθεί στον χρήστη.

Το μήνυμα αυτό μεταβιβάζεται από τον controller στην κλάση BorrowRequest και ο χρήστης θα ακούσει το μήνυμα μέσω της successfulBorrow(borrow).

Στην εναλλακτική ροή όταν η μεταβλητή a έχει τιμή false η κλάση BorrowDBProxy θα καλέσει τη μέθοδο getOverdueDebts() του controller BorrowControll που επιστρέφει τις ληξιπρόθεσμες οφειλές του χρήστη. Οι οφειλές θα ανακοινωθούν από το N.A.O. όταν ο controller BorrowControll καλέσει τη μέθοδο overdueDebtsAnnouncement(validCheckForOverdueDebts, displayOverdueDebts) της κλάσης BorrowRequest.



1.3 **Πακέτο Αναζήτησης Περιγραφής Βιβλίου**

Το πακέτο αυτό αναφέρεται στη διαδικασία αναζήτησης της περιγραφής ενός βιβλίου, στο χώρο της βιβλιοθήκης, από εγγεγραμμένους χρήστες ή επισκέπτες.

Η λειτουργική απαίτηση που σχετίζεται με αυτό το πακέτο χρήσης είναι η:

- <ΛΑ-2> Ο εγγεγραμμένος χρήστης και ο επισκέπτης πρέπει να μπορούν να αναζητήσουν την περιγραφή ενός βιβλίου (Σενάριο Χρήσης 3).

Αφήγηση Σεναρίου

Ο εγγεγραμμένος χρήστης ή ο επισκέπτης, User, εκφωνεί στο ρομπότ ΝΑΟ τη φράση "ΑΝΑΖΗΤΗΣΕ ΠΕΡΙΓΡΑΦΉ" και στη συνέχεια τον τίτλο του βιβλίου που επιθυμεί. Με αυτόν τον τρόπο καλείται η μέθοδος requestListenerAUI(request) και μεταφέρεται στο σύστημά μας η εντολή του χρήστη με την απαραίτητη πληροφορία (επιθυμητή ενέργεια και τίτλος).

Στη συνέχεια, το AUI (Audio User Interface) μέσω της μεθόδου requestBookDesc(registry) επικοινωνεί με τον ελεγκτή DescriptionController.

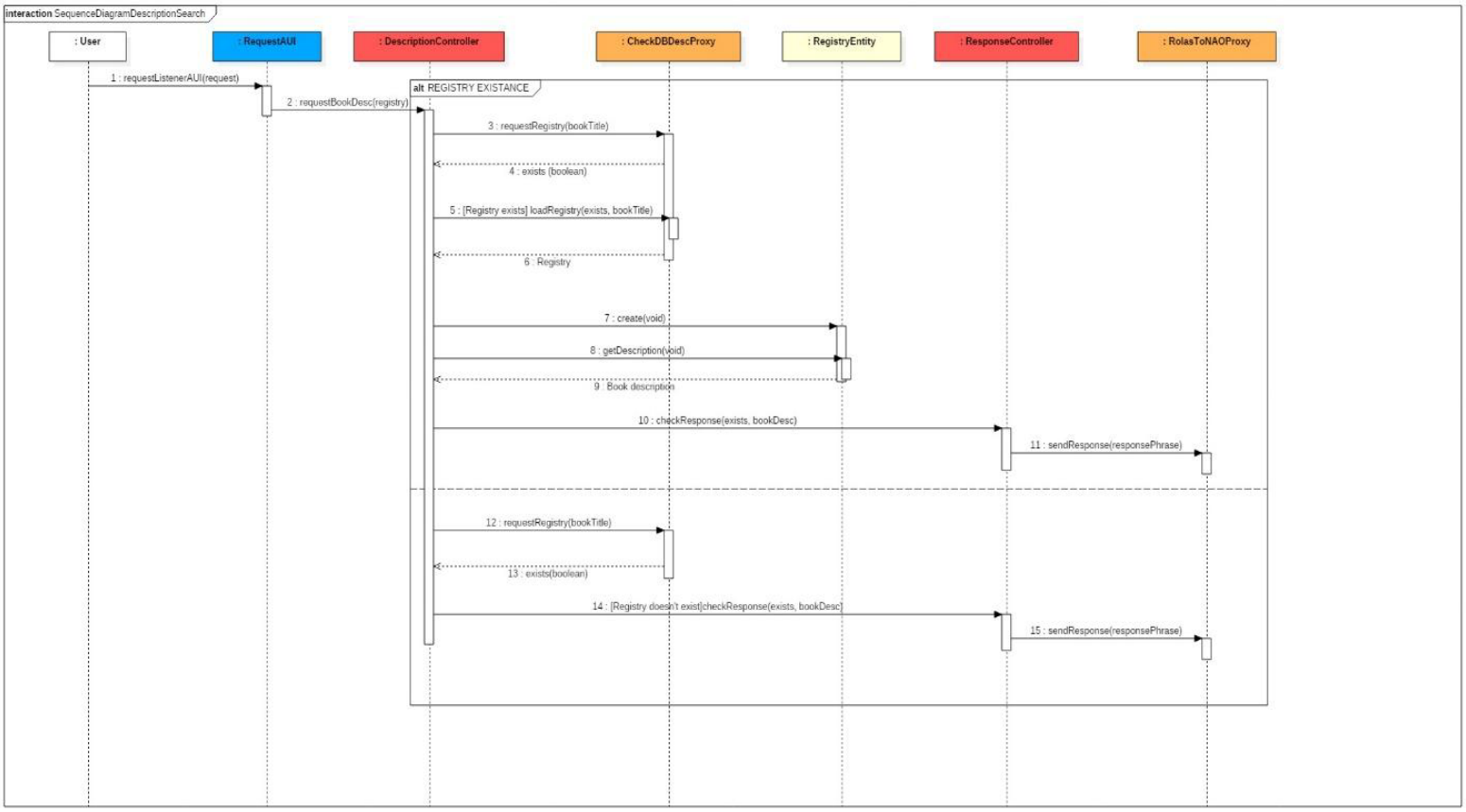
Ο ελεγκτής καλεί τη συνάρτηση requestRegistry(bookTitle) για να ελέγξει μέσω του CheckDBDescProxy εάν υπάρχει η εγγραφή (το βιβλίο) και αν ναι (επιστρέφει exists boolean), ποια είναι αυτή.

Στην περίπτωση που η εγγραφή είναι καταχωρημένη στη βάση δεδομένων Lib_DB, τότε επιστρέφεται στον ελεγκτή μέσω της loadRegistry(exists, bookTitle) η επιθυμητή εγγραφή (Registry). Έπειτα, με την create() δημιουργείται από τον ελεγκτή μία οντότητα τύπου RegistryEntity και διαμέσου της getDescription() επιστρέφεται στον ελεγκτή η περιγραφή του βιβλίου (BookDesc).

Αν η εγγραφή δεν είναι καταχωρημένη στη βάση δεδομένων Lib_DB (εναλλακτική ροή, ALT), τότε επιστρέφεται στον ελεγκτή, μέσω της requestRegistry(bookTitle), μία τιμή boolean (exists) που μας πληροφορεί για την απουσία της εγγραφής.

Στο επόμενο βήμα ο ελεγκτής ResponseController ενημερώνεται από τη συνάρτηση checkResponse(exists, bookDesc) για το είδος και το περιεχόμενο της απάντησης που θα δοθεί στο αρχικό αίτημα του χρήστη για την περιγραφή ενός βιβλίου.

Τέλος, με τη μέθοδο sendResponse(responsePhrase) το RolasToNAOProxy λαμβάνει τη φράση που θα αναλάβει να εκφωνήσει το ΝΑΟ. Στην περίπτωση που η εγγραφή υπάρχει στο σύστημα θα αναγνώσει την περιγραφή του βιβλίου, ενώ στην αντίθετη περίπτωση θα εκφωνήσει "ΤΟ ΒΙΒΛΊΟ ΠΟΥ ΑΝΑΖΗΤΆΤΕ ΔΕΝ ΥΠΆΡΧΕΙ ΣΤΗ ΒΙΒΛΙΟΘΉΚΗ"



1.4 **Πακέτο Αναζήτησης εγγραφών στο σύστημα από τον διαχειριστή**

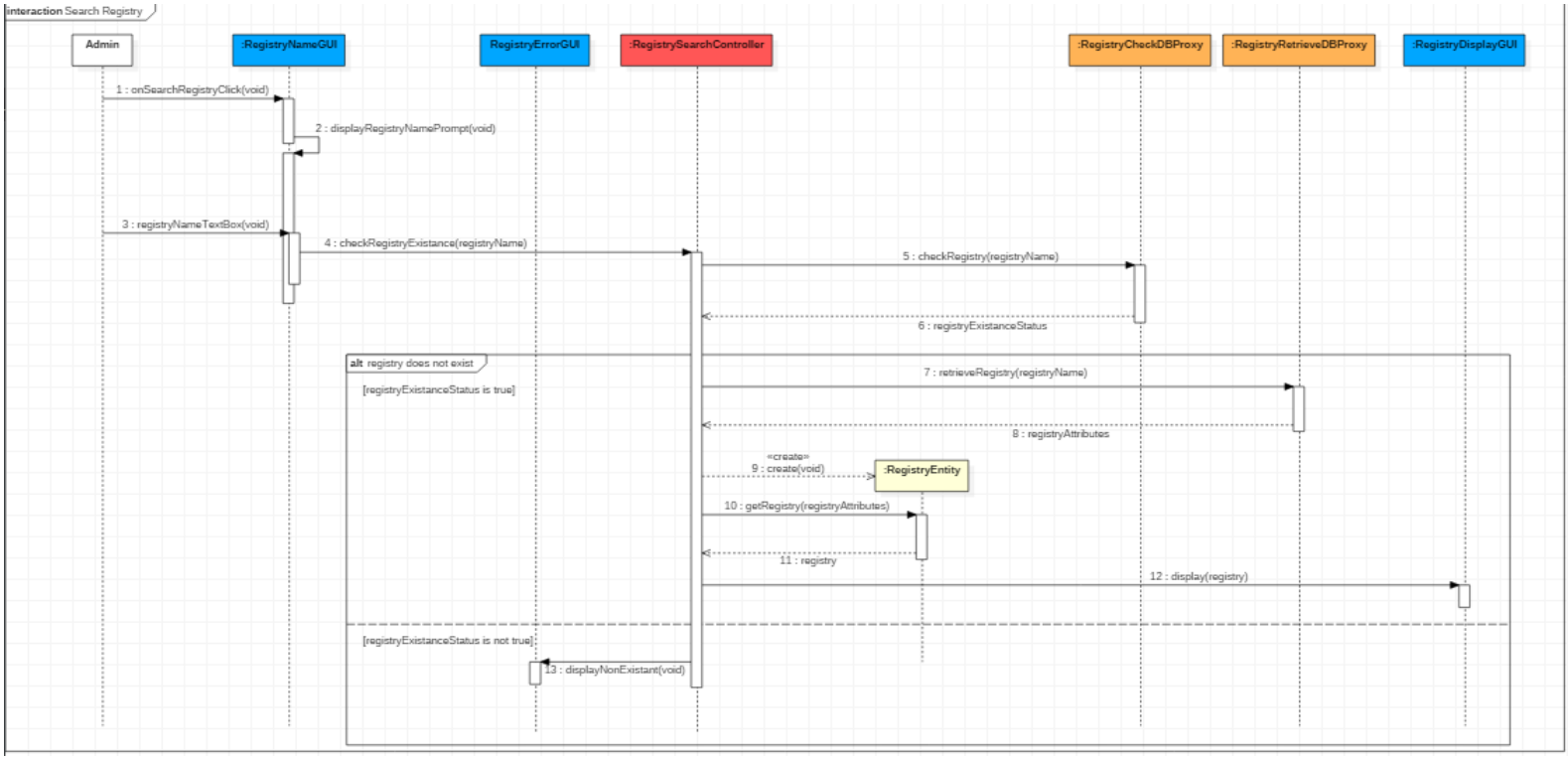
Οι λειτουργικές απαιτήσεις που σχετίζονται με αυτό το πακέτο χρήσης είναι:

- <ΛΑ-9> Ο διαχειριστής πρέπει να μπορεί να αναζητήσει εγγραφές που βρίσκονται στη βάση δεδομένων.

Αφήγηση Σεναρίου

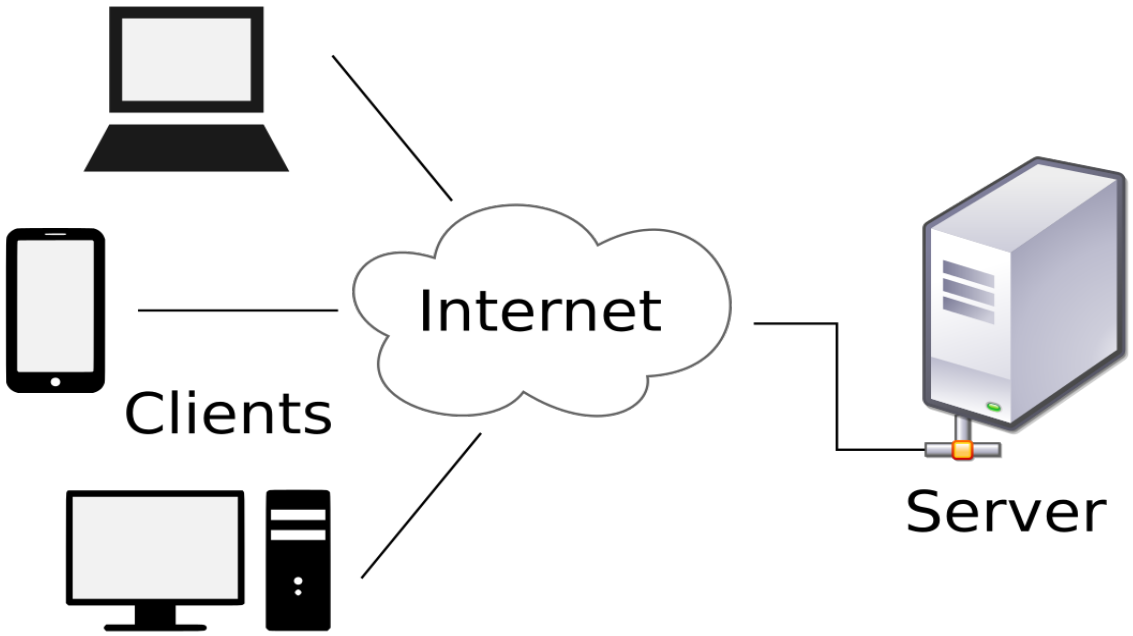
Το σενάριο αυτό πυροδοτείτε έπειτα από επιλογή του κουμπιού «Αναζήτηση Εγγραφών» από τον διαχειριστή μέσω της μεθόδου onSearchRegistryClick της RegistryNameGUI.Στη συνέχεια εμφανίζεται η οθόνη προτροπής για την εισαγωγή του ονόματος της εγγραφής που επιθυμεί να αναζητήσει ο διαχειριστής. Αφού ο διαχειριστής εισάγει το όνομα της εγγραφής στο πεδίο κειμένου της προτρεπτικής οθόνης καλείται η συνάρτηση checkRegistryExistance του RegistrySearchController με όρισμα το όνομα που εισήγαγε ο διαχειριστής στο πεδίο κειμένου της προτρεπτικής οθόνης που δημιουργεί η μέθοδος registryNameTextBox του οριακού αντικειμένου RegistryNameGUI . Έπειτα καλείται από τον RegistrySearchController η μέθοδος checkRegistry του RegistryCheckDBProxy με όρισμα το όνομα της εγγραφής (registryName), πραγματοποιείται αναζήτηση ύπαρξης της συγκεκριμένης εγγραφής στη βάση δεδομένων μετά την λήξη της οποίας η μέθοδος επιστρέφει μία λογική τιμή που αντιστοιχεί στην κατάσταση ύπαρξης της εγγραφής στην βάση δεδομένων (registryExistanceStatus). Εάν η εγγραφή υπάρχει τότε γίνεται κλήση της μεθόδου retrieveRegistry του RegistryRetrieveDBProxy με όρισμα το όνομα της εγγραφής .

Μια λίστα με τα χαρακτηριστικά της εγγραφής ανακτάται από την βάση δεδομένων και επιστρέφεται από τη μέθοδο retrieveRegistry στον RegistrySearchController. Στη συνέχεια δημιουργείται η οντότητα RegistryEntity και έπειτα καλείται από τον RegistrySearchController η μέθοδος getRegistry της Registry Entity με όρισμα την λίστα που ανακτήθηκε παραπάνω. Η μέθοδος αυτή επιστρέφει την εγγραφή σαν αντικείμενο τύπου RegistryEntity. Ο RegistrySearchController καλεί την μέθοδο display, με όρισμα την εγγραφή τύπου RegistryEntity, της RegistryDisplayGUI , η οποία εμφανίζει την οθόνη με τις πληροφορίες (χαρακτηριστικά) της εγγραφής. Σε περίπτωση που η εγγραφή δεν υπάρχει στην βάση δεδομένων ο RegistrySearchController καλεί την displayNonExistant του RegistryErrorGUI , η οποία προβάλλει το μήνυμα της ανεπιτυχούς εύρεσης της εγγραφής.



2. **Προτεινόμενη Αρχιτεκτονική Λογισμικού**

Για το σύστημα Robolibrarian Assistant System, που αναπτύσσουμε στα πλαίσια αυτού του μαθήματος, χρησιμοποιήσαμε την αρχιτεκτονική client/server για τον βασικό λόγο πως παρέχει τη δυνατότητα σε πολλούς χρήστες να χρησιμοποιούν ταυτόχρονα την ίδια εφαρμογή και τα αρχεία που είναι αποθηκευμένα σε ένα server. Αυτός ο τύπος αρχιτεκτονικής αντιπροσωπεύει την υλοποίησή μας καθώς έχουμε πολλούς χρήστες να αιτούνται δανεισμούς, δηλαδή αιτούνται πρόσβαση σε αρχεία που είναι αποθηκευμένα σε έναν κεντρικό υπολογιστή.



2.1 **Αρχιτεκτονική πελάτη-διακομιστή**(client-server)

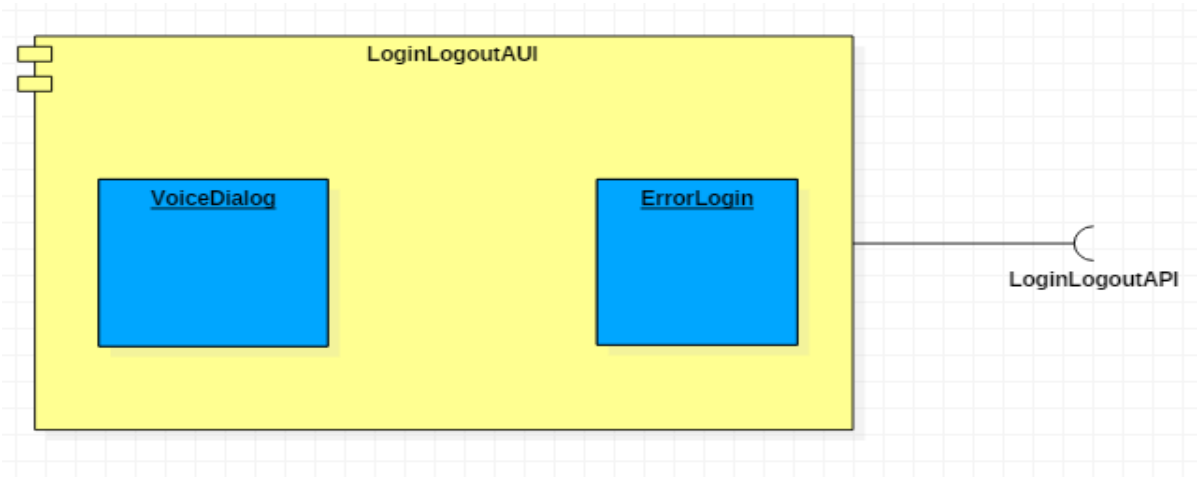
Η αρχιτεκτονική client/server χαρακτηρίζεται από την δυνατότητα συνεργασίας προγραμμάτων μεταξύ πελάτη και εξυπηρετητή σε μία εφαρμογή, στη συγκεκριμένη περίπτωση στο σύστημά μας. Ο server παρέχει συναρτήσεις-υπηρεσίες σε έναν ή περισσότερους χρήστες, που έχουν προηγουμένως υποβάλλει αίτηση για αυτές. Ο server διαθέτει σε κοινή χρήση στους χρήστες προγράμματα, δεδομένα, αποθηκευτικούς χώρους ακόμη και επεξεργαστές που αποτελούν τις λεγόμενες υπηρεσίες. Στο σύστημά μας έχουμε τρεις διαφορετικούς τύπους χρηστών (admin, guest user, registered) οπότε αναλόγως επιλέγεται και το αντίστοιχο τμήμα της υπηρεσίας που θα παραχθεί.

2.2 **Αποδόμηση συστήματος**

Γενικά τα υποσυστήματα διακρίνονται ως: Συλλογή κλάσεων, συνδέσεων, λειτουργιών, γεγονότων και περιορισμών που είναι διασυνδεδεμένα.
Συνεπώς στην παράγραφο αυτή παρουσιάζουμε το διαχωρισμό του συστήματος σε επιμέρους υποσυστήματα τα οποία είναι υπεύθυνα για την υλοποίηση ξεχωριστών λειτουργιών.
Σχεδιάζουμε λοιπόν το σύστημά μας με στόχο την υψηλή συνεκτικότητα και τη χαμηλή σύζευξη όσο είναι δυνατόν.

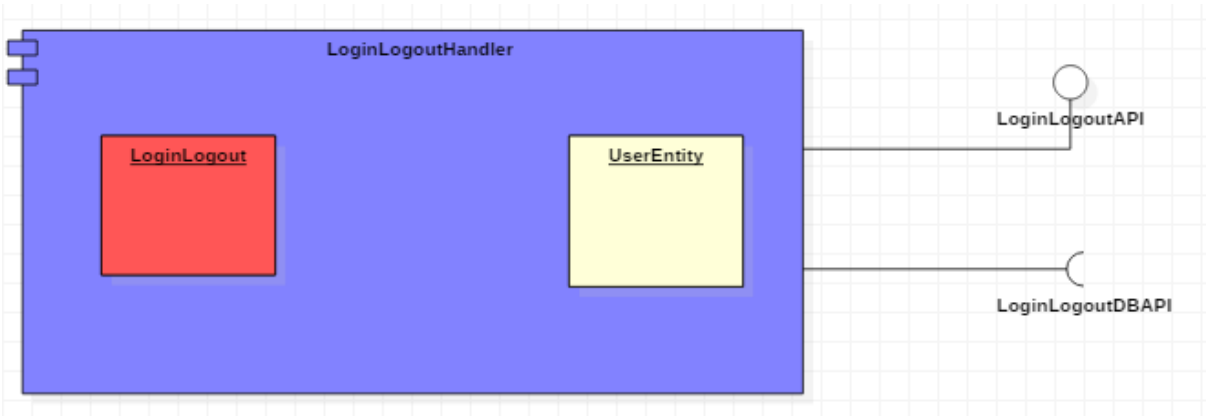
2.2.1 **Υποσύστημα** LoginLogoutAUI

Το υποσύστημα αυτό αποτελείται από τις διεπαφές που ο χρήστης του συστήματος έρχεται σε αλληλεπίδραση κατά την είσοδό του στο σύστημα. Αποτελείται ουσιαστικά από την κλάση VoiceDialog , μέσω της οποίας ο χρήστης ζητάει να συνδεθεί ή να αποσυνδεθεί από το σύστημα , και η ErrorLogin η οποία είναι υπεύθυνη για την εκφώνηση μηνύματος σφάλματος σε περίπτωση που τα στοιχεία που έχουν δοθεί από το χρήστη στο σύστημα είναι εσφαλμένα.



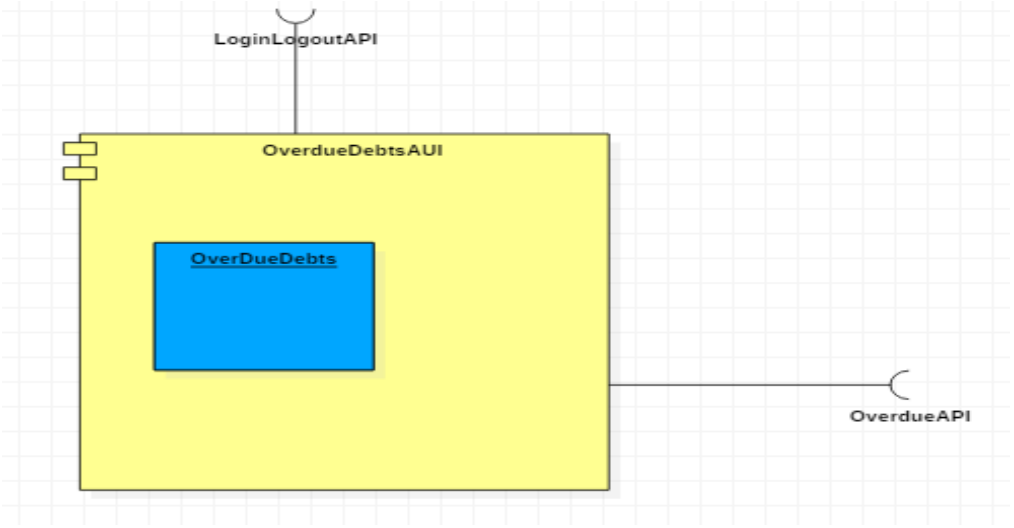
2.2.2 **Υποσύστημα** LoginLogoutHandler

Το συγκεκριμένο υποσύστημα περιέχει τα τμήματα του ελεγκτή (LoginLogout) που είναι υπεύθυνο για την διαχείριση των δεδομένων που ο χρήστης δίνει στο σύστημα κατά την είσοδό του καθώς επίσης και για την διεκπεραίωση της διαδικασίας εξόδου του εγγεγραμμένου χρήστη από το σύστημα. Ενώ η κλάση UserEntity περιλαμβάνει τις πληροφορίες που αφορούν το χρήστη.



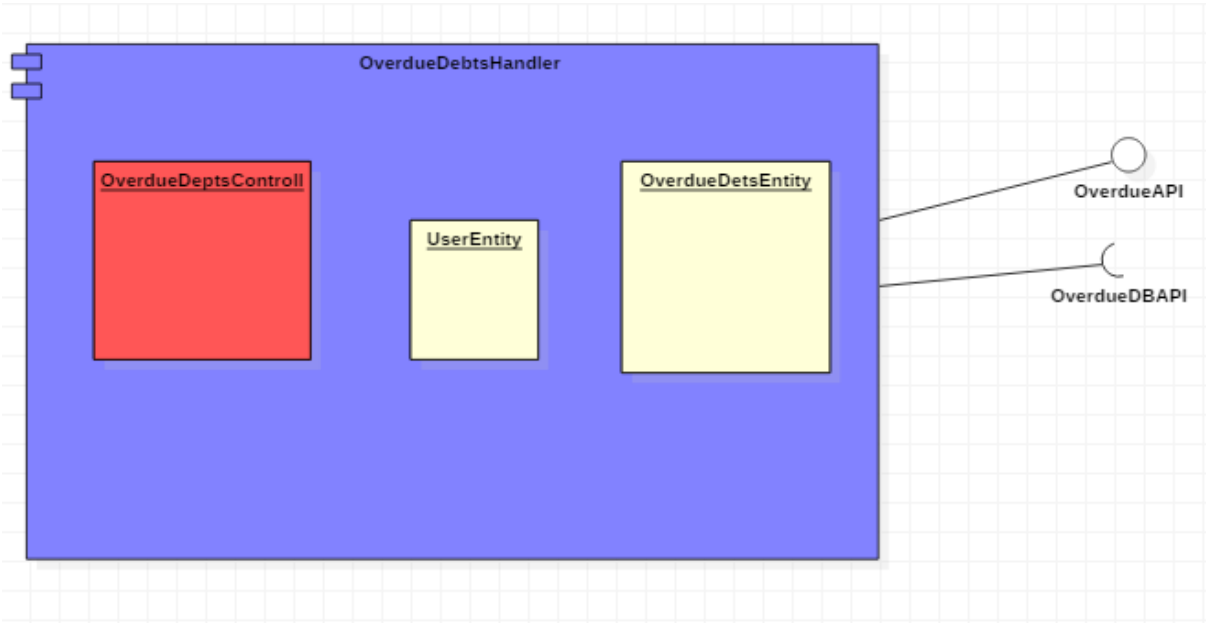
2.2.3 **Υποσύστημα** OverdueDebtsAUI

Το υποσύστημα αυτό είναι υπεύθυνο για τον έλεγχο των ληξιπρόθεσμων οφειλών του εγγεγραμμένου χρήστη. Μέσω της κλάσης OverdueDebts ο χρήστης εισέρχεται στο σύστημα και στη συνέχεια το σύστημα επιστρέφει την ημερομηνία δανεισμού του βιβλίου ώστε ανάλογα μέσω του OverdueDebtsHandler όπου θα γίνει ο έλεγχος για την ύπαρξη ληξιπρόθεσμων οφειλών.



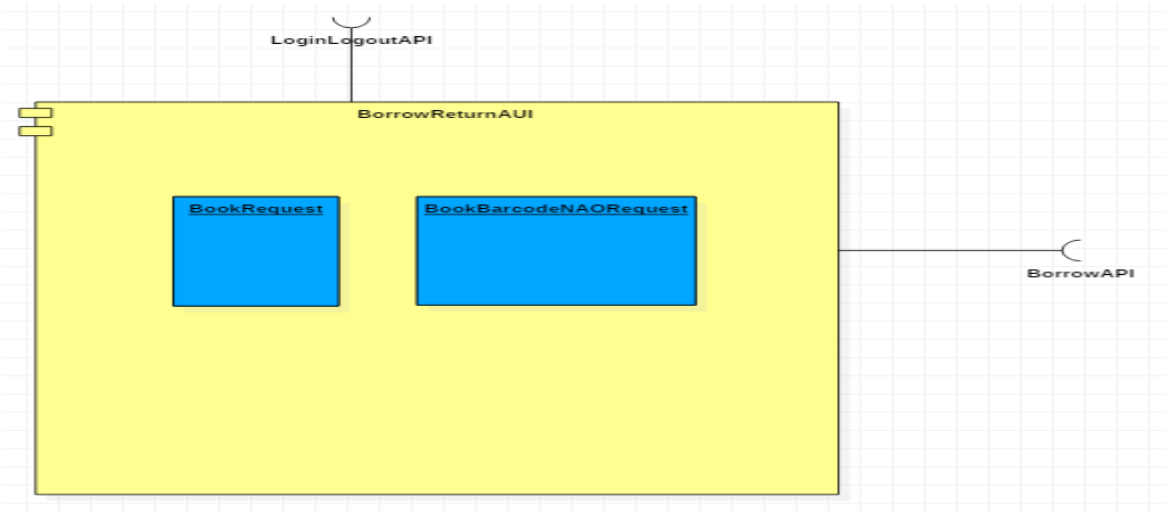
2.2.4 **Υποσύστημα** OverdueDebtsHandler

Το υποσύστημα αυτό είναι υπεύθυνο για την εύρεση των ληξιπρόθεσμων οφειλών του χρήστη. Στην κλάση οντοτήτων OverdueDetsEntity κρατιόνται οι ημερομηνίες δανεισμού βιβλίων του χρήστη ,και μέσω της OverdueDeptsControll γίνεται ο έλεγχος των στοιχείων αυτών προκειμένου να εκφωνηθεί από το σύστημα ανάλογο μήνυμα. Επίσης στην κλάση οντοτήτων UserEntity περιλαμβάνονται τα στοιχεία του εγγεγραμμένου χρήστη.



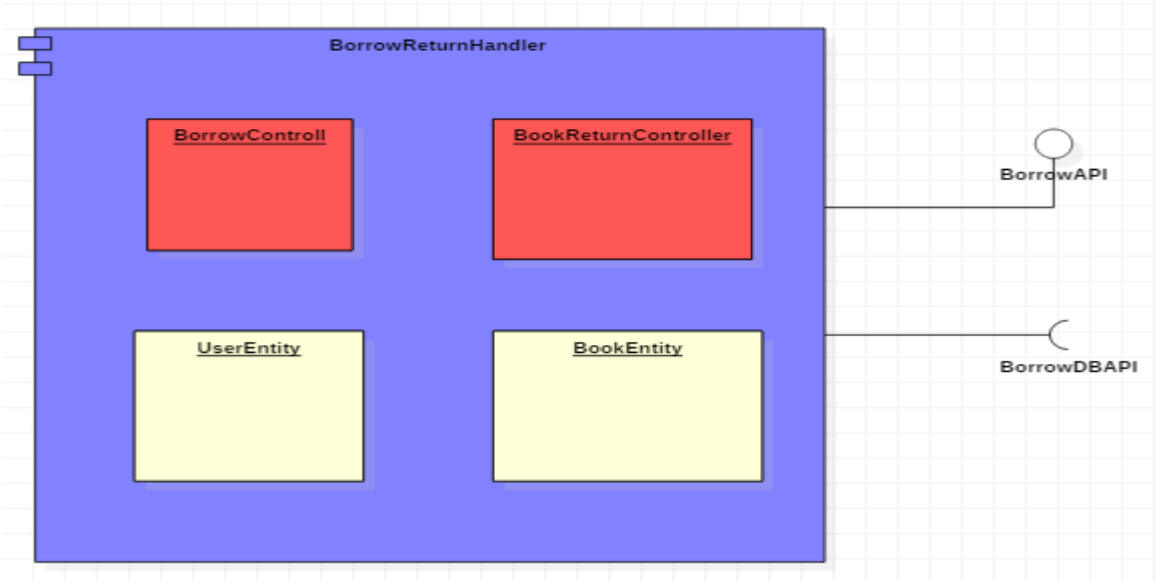
2.2.5 **Υποσύστημα** BorrowReturnAUI

Το υποσύστημα BorrowReturnAUI περιέχει τις κλάσεις BookRequest και BookBarcodeNAORequest. Η πρώτη κλάση δέχεται από τον χρήστη τον τίτλο ή τον ταξινομικό κωδικό του βιβλίου που αυτός επιθυμεί να δανειστεί και επιστρέφει τις διαθέσιμες πληροφορίες για το βιβλίο αυτό, ελέγχει επίσης αν εκκρεμούν κάποιοι δανεισμοί του εγγεγραμμένος χρήστης προκειμένου να του επιτρέψει να δανειστεί κάποιο νέο βιβλίο σε περίπτωση όπου δεν υπάρχει κάποια ληξιπρόθεσμη οφειλή του. Ενώ η BookBarcodeNAORequest είναι υπεύθυνη για την επιστροφή των βιβλίων από τον χρήστη. Μέσω αυτής διαβάζεται από το σύστημα ο ραβδοκώδικας του επιστρεφόμενου βιβλίου και στη συνέχεια μετά από σχετικό έλεγχο των στοιχείων αυτών εκφωνεί στον χρήστη μήνυμα σχετικό με το αποτέλεσμα της διαδικασίας επιστροφής.



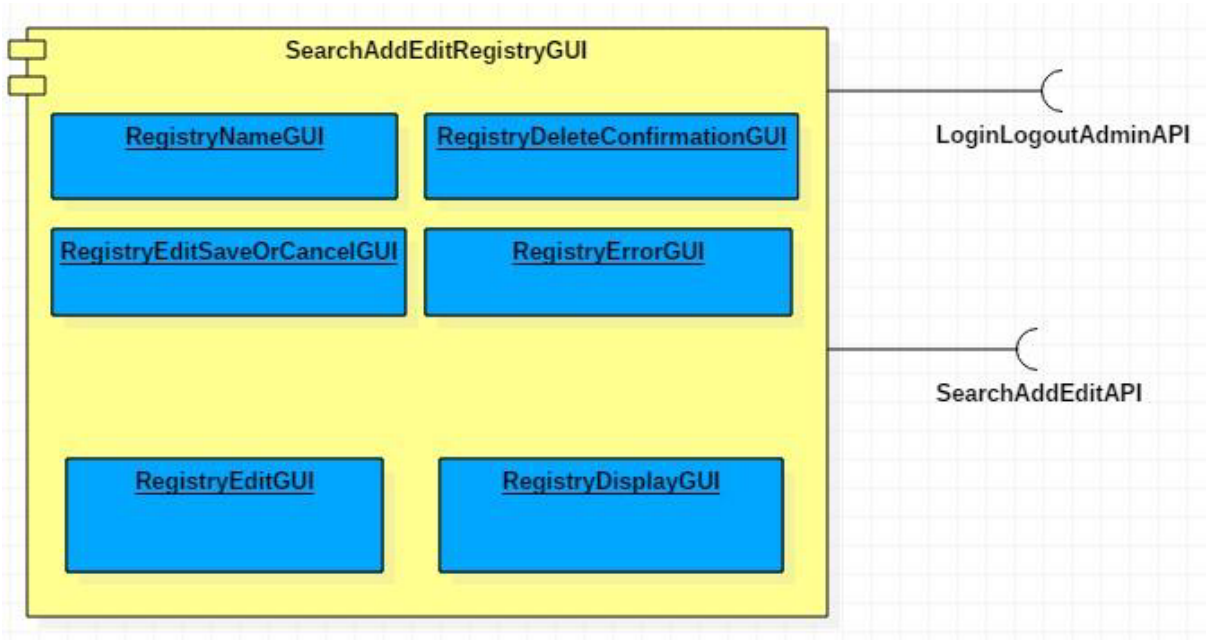
2.2.6 **Υποσύστημα** BorrowReturnHandler

Μέσω του υποσυστήματος αυτού επιτυγχάνεται από την BorrowControll ο έλεγχος των του τύπου του αιτήματος που έχει κατατεθεί από τον χρήστη αν είναι δυνατός ο δανεισμός κάποιου βιβλίου, διαβάζει τα χαρακτηριστικά του χρήστη και του βιβλίου που πρόκειται να δανειστεί και τέλος αποθηκεύει το αποτέλεσμα για επιτυχή δανεισμό ή μη. Και μέσω της BookreturnController γίνεται ο έλεγχος επιστροφής του βιβλίου. Αντίστοιχα στις κλάσεις οντοτήτων UserEntity και της BookEntity περιέχουν τα στοιχεία του εγγεγραμμένου χρήστη και τα στοιχεία του βιβλίου (προς δανεισμό ή επιστροφή) αντίστοιχα.



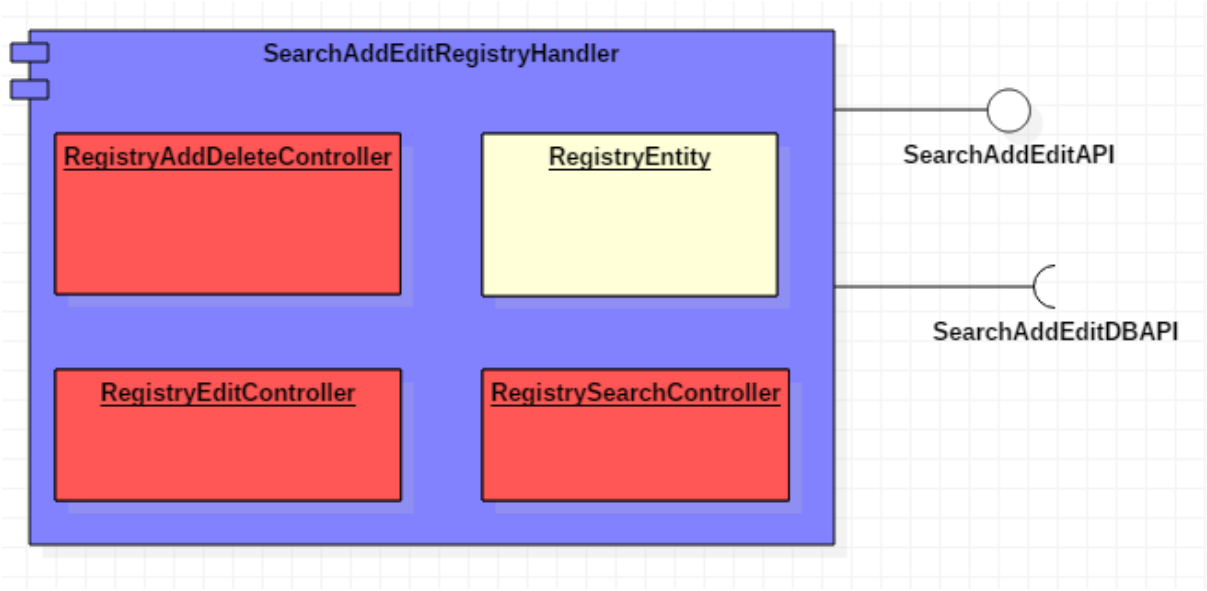
2.2.7 **Υποσύστημα** SearchAddEditRegistryGUI

Το υποσύστημα αυτό αποτελείται από εκείνες τις κλάσεις που μοντελοποιούν τη διεπαφή μέσω της οποίας ο admin διαχειρίζεται τις εγγραφές βιβλίων στη βάση δεδομένων. Πρώτα όμως η διεπαφή αυτή ζητάει να έχει συνδεθεί στο σύστημα. Η searchAddEditAPI είναι οι υπηρεσίες που ζητάει η διεπαφή αυτή για προσθήκη, αναζήτηση και επεξεργασία εγγραφών βιβλίων. Με την RegistryNameGUI ο admin προσθέτει/αφαιρεί εγγραφές, με την RegistryEditSaveOrCancelGUI ο admin αποθηκεύει οριστικά ή όχι μια επεξεργασμένη εγγραφή, με την RegistryDeleteConfirmationGUI ο admin επιβεβαιώνει τη διαγραφή μίας εγγραφής από τη βάση δεδομένων, με την RegistryErrorGUI ο admin ενημερώνεται για τη μη ύπαρξη μίας εγγραφής στη βάση δεδομένων και τέλος με τις RegistryEditGUI και RegistryDisplayGUI ο admin μπορεί να επεξεργαστεί τα πεδία της εγγραφής αλλά και να δει τα περιεχόμενά τους αντίστοιχα.



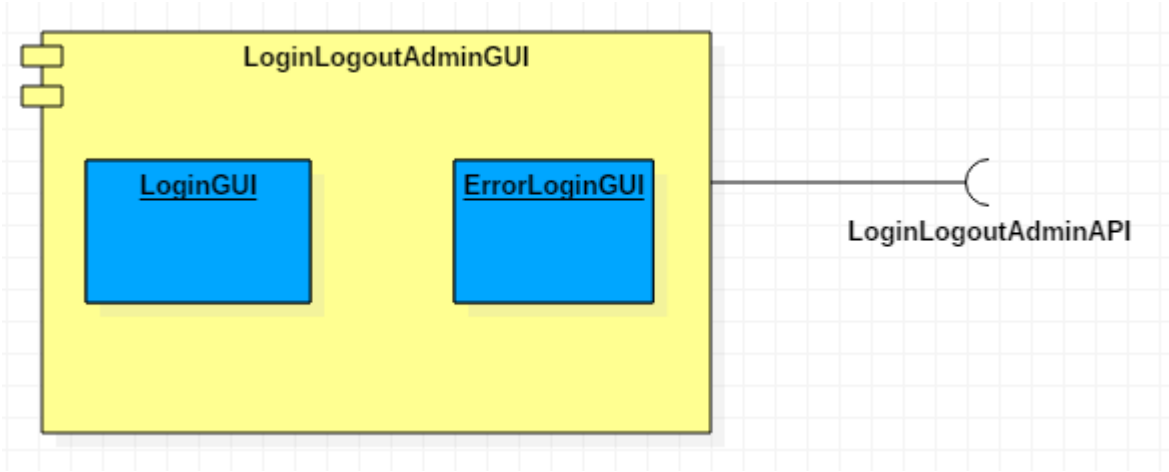
2.2.8 **Υποσύστημα** SearchAddEditRegistryHandler

Το συγκεκριμένο υποσύστημα παρέχει στο SearchAddEditRegistryGUI μέσω τις υπηρεσίας SearchAddEditAPI τις υπηρεσίες που αναφέρθηκαν παραπάνω και τις οποίες πρώτα ζητάει από την DataBase μέσω της SearchAddEditDBAPI. Συγκεκριμένα η κλάση RegistryAddDeleteController προσθέτει/αφαιρεί εγγραφές στη βάση δεδομένων, το RegistryEntity περιέχει τις πληροφορίες της εγγραφής ενός βιβλίου, ο RegistryEditController επεξεργάζεται τις εγγραφές στη βάση δεδομένων και τέλος ο RegistrySearchController αναζητεί εγγραφές που βρίσκονται στη βάση δεδομένων.



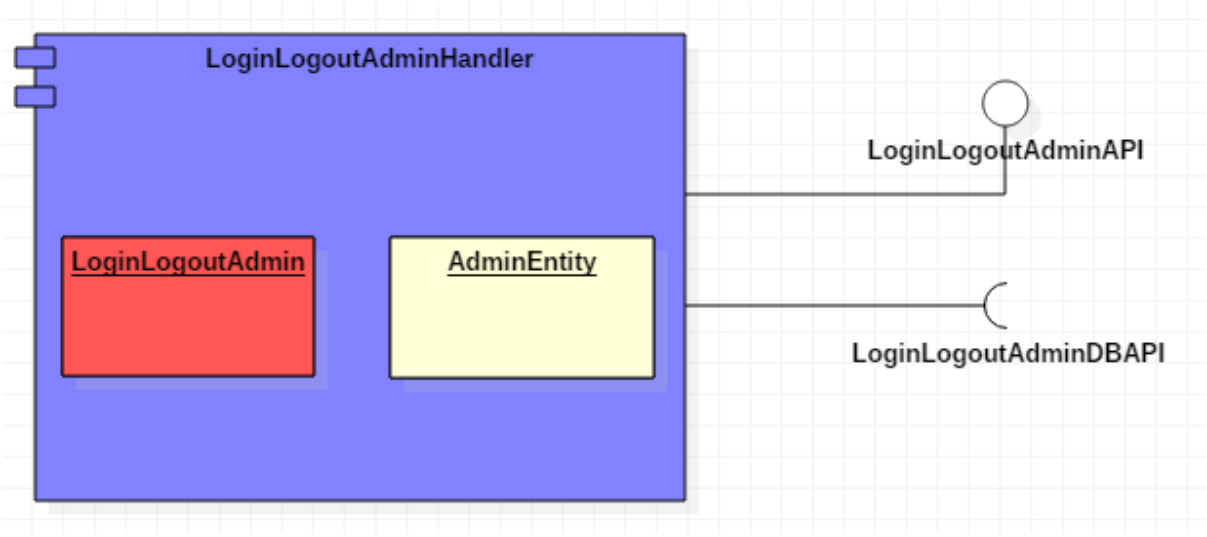
2.2.9 **Υποσύστημα** LoginLogoutAdminGUI

Το υποσύστημα αυτό αποτελεί μία διεπαφή του χρήστη με το σύστημα και του παρέχει τη δυνατότητα σύνδεσης/αποσύνδεσης σε αυτό. Το υποσύστημα αυτό απαιτεί την υπηρεσία που επιστρέφει τα στοιχεία του διαχειριστή που πρόκειται να συνδεθεί. Η κλάση LoginGUI αντιπροσωπεύει τη σελίδα σύνδεσης του διαχειριστή στο σύστημα και η κλάση ErrowLoginGUI τη σελίδα επιστροφής των σφαλμάτων σύνδεσης του διαχειριστή.



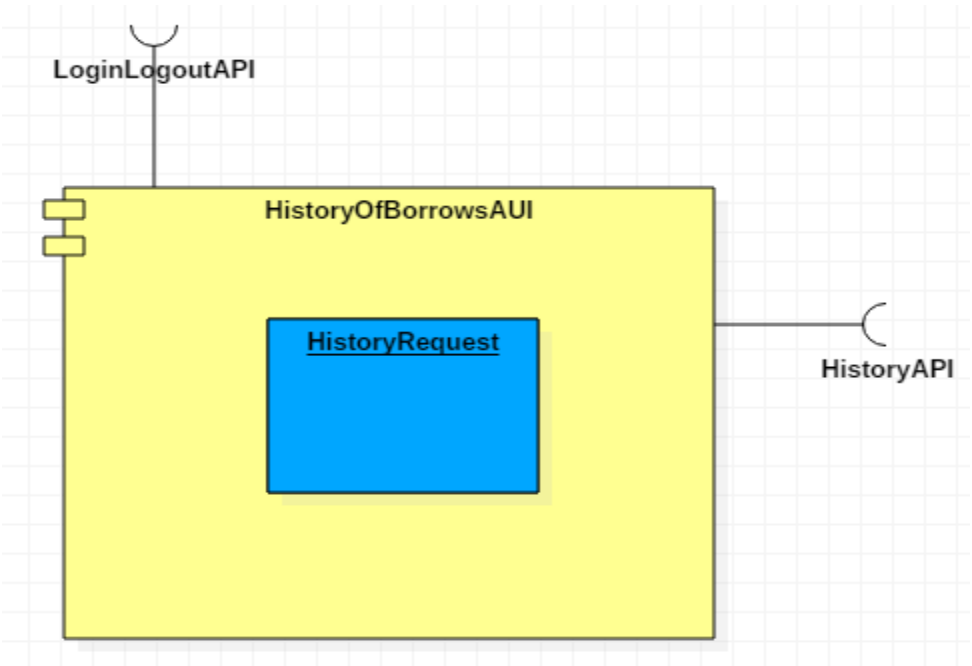
2.2.10 **Υποσύστημα** LoginLogoutAdminHandler

Το υποσύστημα αυτό παρέχει την υπηρεσία LoginLogoutAdminAPI, που περιλαμβάνει τα στοιχεία του διαχειριστή που συνδέεται στο σύστημα, στο υποσύστημα LoginLogoutAdminGUI και τα οποία παραλαμβάνει μέσω της LoginLogoutAdminDBAPI από την database. Η κλάση LoginLogoutAdmin ελέγχει τη σύνδεση/αποσύνδεση του admin στο σύστημα. Τέλος η κλάση AdminEntity περιέχει τα στοιχεία ορισμού του διαχειριστή.



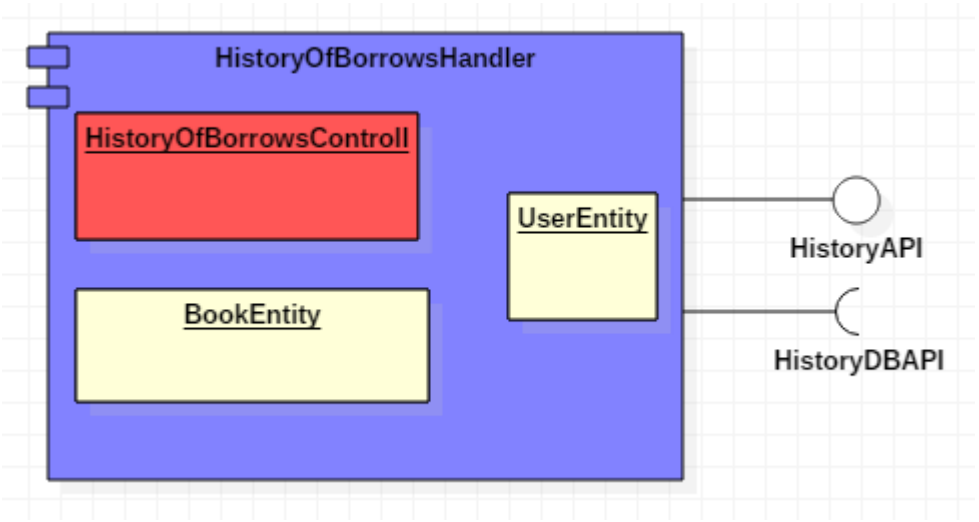
2.2.11 **Υποσύστημα** HistoryOfBorrowsAUI

Το υποσύστημα αυτό έχει ως σκοπό να προσκομίσει το ιστορικό δανεισμών του χρήστη. Σε αυτό βοηθάει η διεπαφή HistoryAPI που αιτείται των πληροφοριών αυτών μέσω του controller αρχικά και έπειτα ο controller μέσω της βάσης δεδομένων. Απαραίτητη προϋπόθεση είναι η διεπαφή LoginLogoutAdmin να έχει πραγματοποιήσει τη σύνδεση του χρήστη στο σύστημα. Η κλάση HistoryRequest λειτουργεί με σκοπό να ενημερωθεί ο χρήστης μέσω του N.A.O. για το ιστορικό δανεισμών του.



2.2.12 **Υποσύστημα** HistoryOfBorrows**H**andler

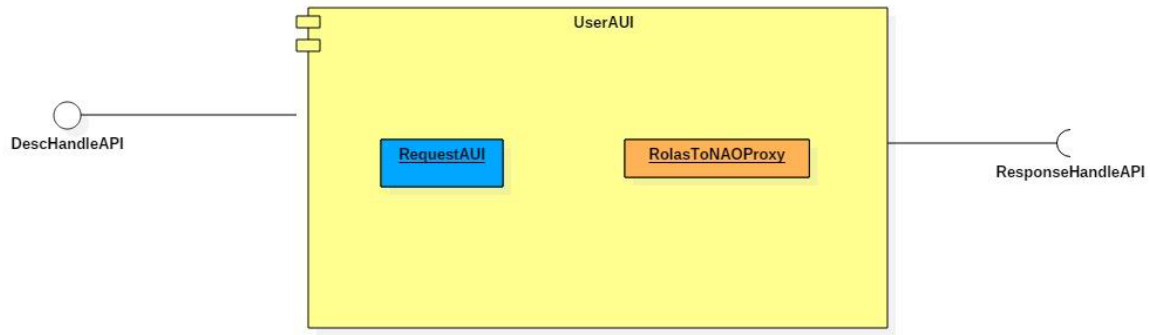
Το υποσύστημα HistoryOfBorrowsHandler μοντελοποιεί τον έλεγχο για την ορθή εκφώνηση του ιστορικού δανεισμών του χρήστη από το N.A.O. Διαθέτει τις διεπαφές HistoryAPI που παρέχει στο HistoryOfBorrowsAUI την υπηρεσία να το ενημερώσει με το ιστορικό δανεισμών, το οποίο όμως πρώτα προσκόμισε μέσω της διεπαφής HistoryDBAPI από τη Βάση Δεδομένων. Ο ελεγκτής HistoryOfBorrowsControll εκκινεί τις διαδικασίες για αναζήτηση του ιστορικού δανεισμών, ελέγχοντας ταυτόχρονα για ύπαρξη ή μη προηγούμενων δανεισμών. Οι κλάσεις UserEntity και BookEntity περιέχουν τις πληροφορίες του χρήστη και των βιβλίων που έχει δανειστεί στο πρόσφατο παρελθόν.



2.2.13 **Διάγραμμα Τμημάτων** DescPackage

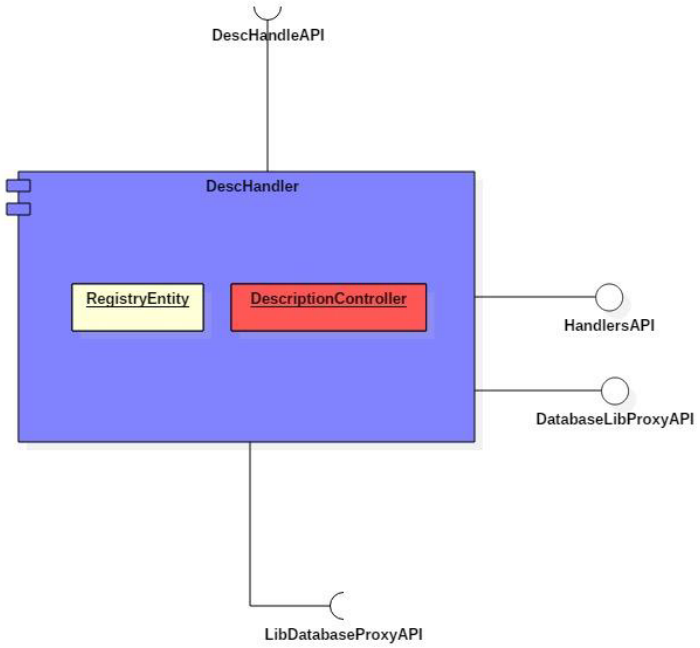
2.2.13.1 **Υποσύστημα** UserAUI

Το παρακάτω υποσύστημα περιέχει τις κλάσεις RequestAUI και RolasToNAOProxy. Το υποσύστημα αυτό είναι υπεύθυνο για την επικοινωνία του χρήστη με το σύστημα μέσω του NAO, δηλαδή την υποδοχή των εντολών του χρήστη και την εκφώνηση της απάντησης του συστήματος.



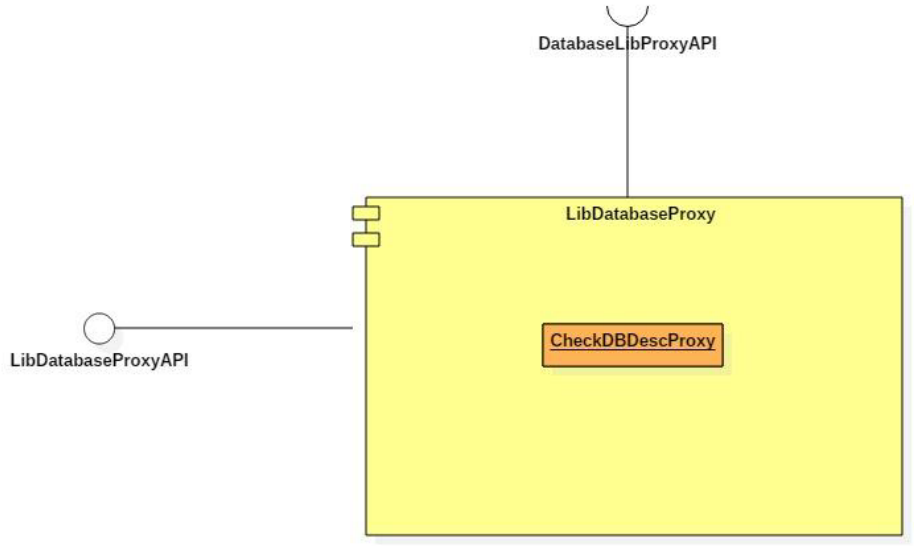
2.2.13.12 **Υποσύστημα** DescHandler

Το παρακάτω υποσύστημα περιέχει τις κλάσεις RegistryEntity και DescriptionController. Το υποσύστημα αυτό είναι υπεύθυνο για την λήψη των πληροφοριών της εντολής του χρήστη, την επικοινωνία με τη βάση δεδομένων (και για το query, αλλά και για το answer), καθώς και για τη σύνδεση με το υποσύστημα ResponseHandler.



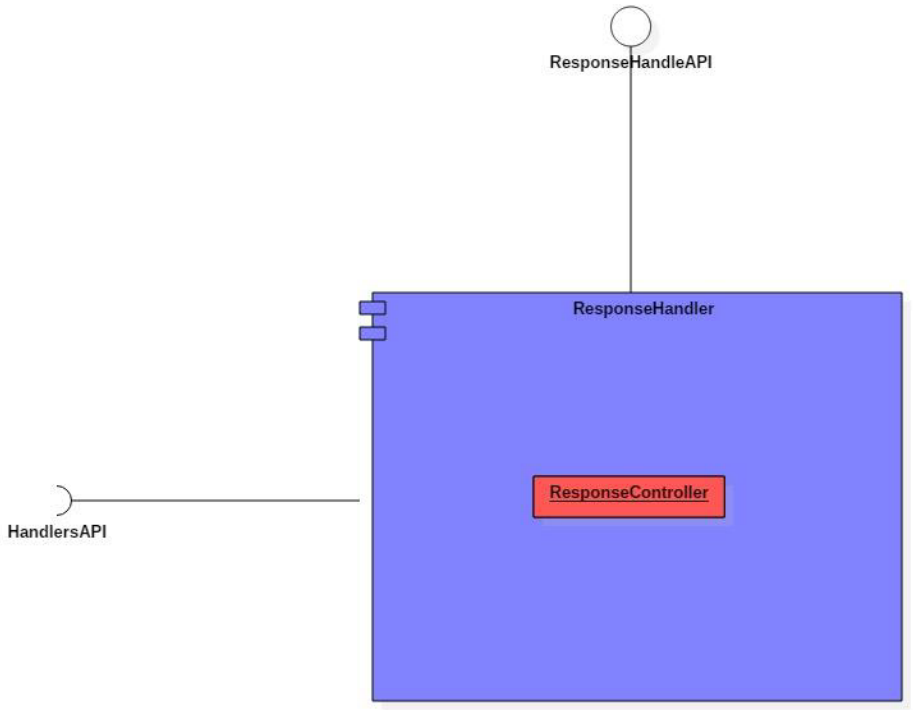
2.2.13.3 **Υποσύστημα** LibDatabaseProxy

Το παρακάτω υποσύστημα περιέχει την κλάση CheckDBDescProxy. Το υποσύστημα αυτό είναι υπεύθυνο για την επικοινωνία με το DescHandler και τη βάση δεδομένων.

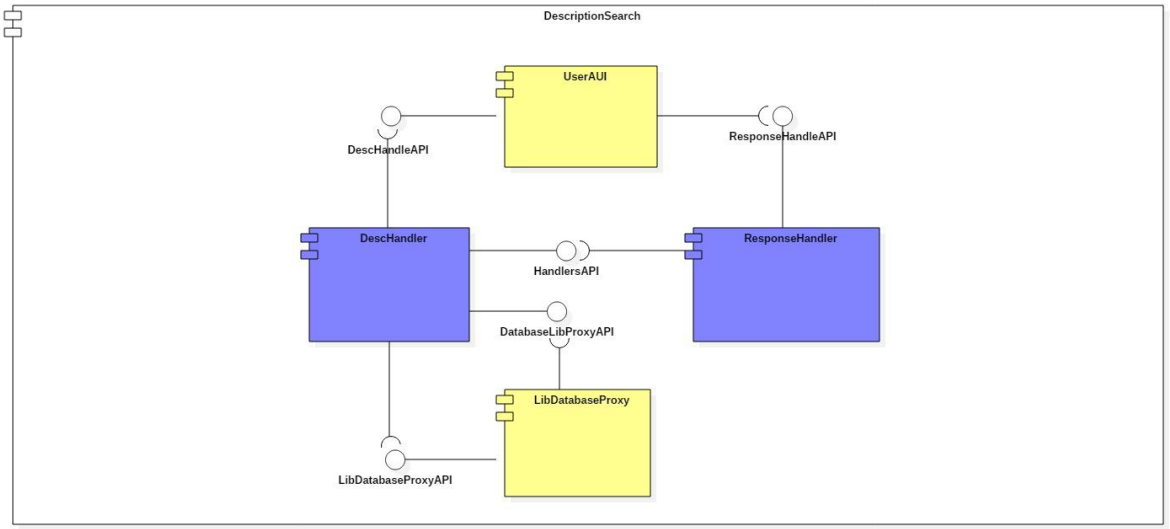


2.2.13.4 **Υποσύστημα** ResponseHandler

Το παρακάτω υποσύστημα περιέχει την κλάση ResponseController. Το υποσύστημα αυτό είναι υπεύθυνο για την επικοινωνία με το υποσύστημα DescHandler από το οποίο λαμβάνει την κατάλληλη απάντηση που θα εκφωνήσει, καθώς και με το υποσύστημα AUI στο οποίο επικοινωνεί την απάντηση αυτή.



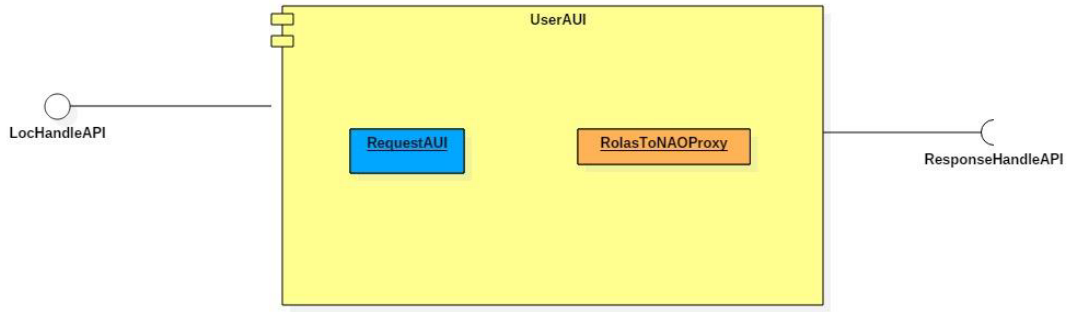
Ολικό διάγραμμα Τμημάτων DescPackage :



2.2.14 **Διάγραμμα Τμημάτων** LocationPackage

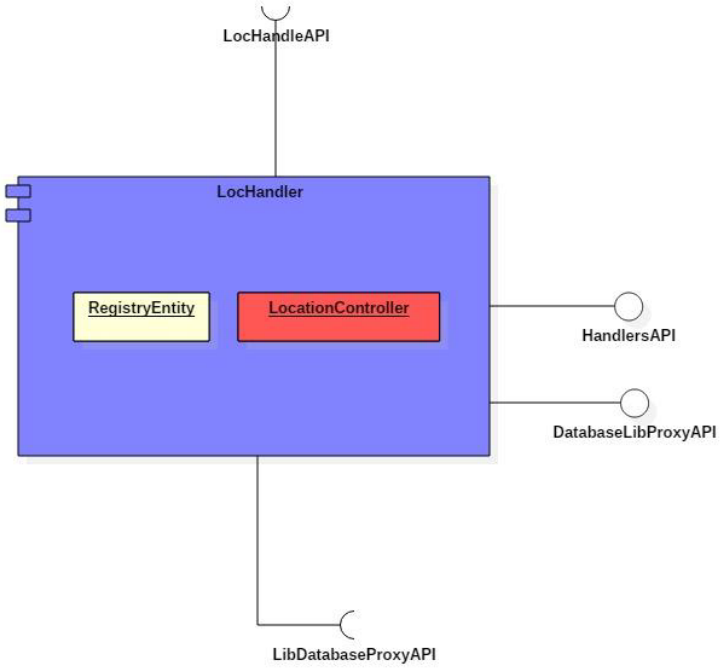
2.2.14.1 **Υποσύστημα** UserAUI

Το παρακάτω υποσύστημα περιέχει τις κλάσεις RequestAUI και RolasToNAOProxy. Το υποσύστημα αυτό είναι υπεύθυνο για την επικοινωνία του χρήστη με το σύστημα μέσω του NAO, δηλαδή την υποδοχή των εντολών του χρήστη και την εκφώνηση της απάντησης του συστήματος.



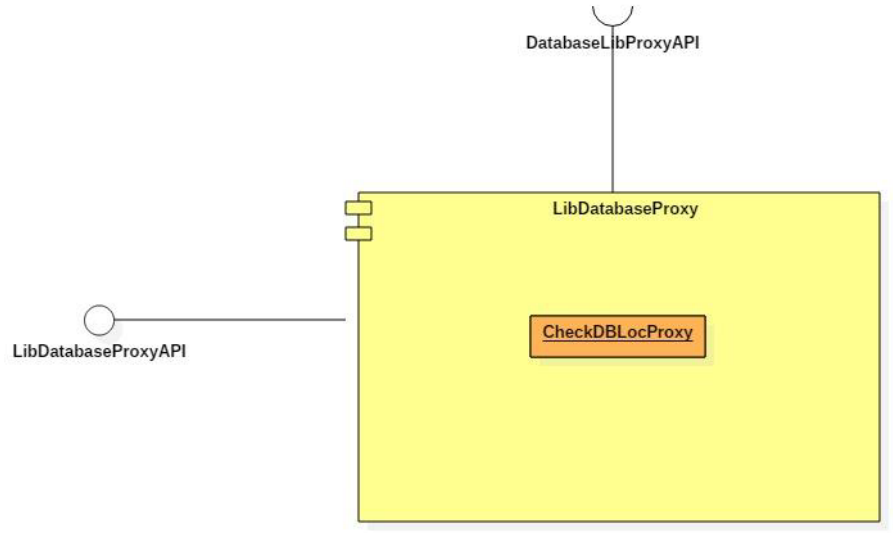
2.2.14.2 **Υποσύστημα** LocHandler

Το παρακάτω υποσύστημα περιέχει τις κλάσεις RegistryEntity και LocController. Το υποσύστημα αυτό είναι υπεύθυνο για τη λήψη των πληροφοριών της εντολής του χρήστη, την επικοινωνία με τη βάση δεδομένων (και για το query, αλλά και για το answer), καθώς και για τη σύνδεση με το υποσύστημα ResponseHandler.



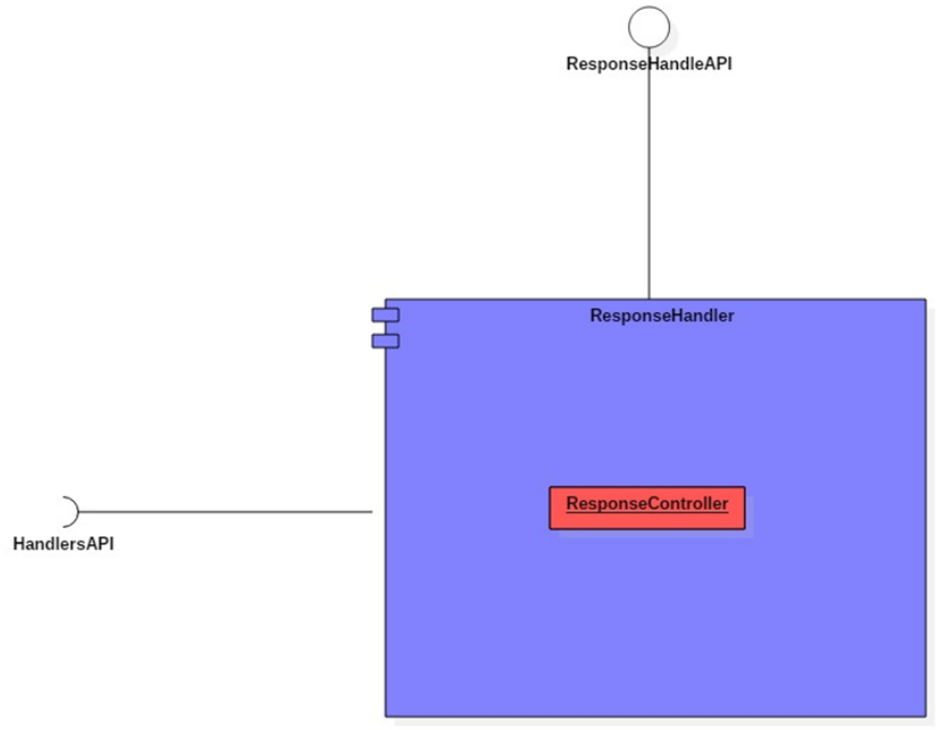
2.2.14.3 **Υποσύστημα** LibDatabaseProxy

Το παρακάτω υποσύστημα περιέχει την κλάση CheckDBLocProxy. Το υποσύστημα αυτό είναι υπεύθυνο για την επικοινωνία με το LocHandler και τη βάση δεδομένων.

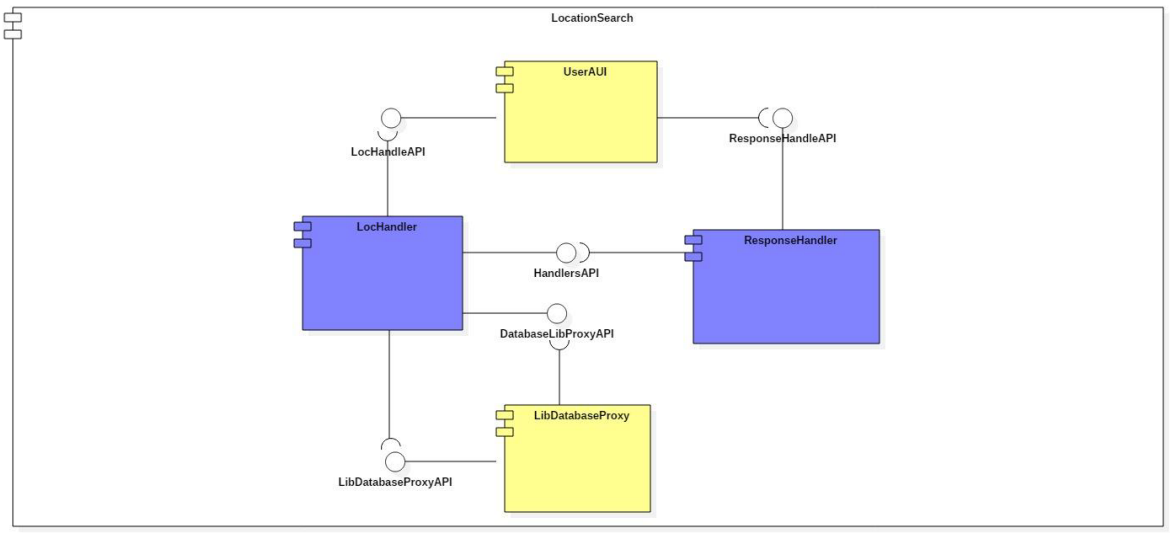


2.2.14.4 **Υποσύστημα** ResponseHandler

Το παρακάτω υποσύστημα περιέχει την κλάση ResponseController. Το υποσύστημα αυτό είναι υπεύθυνο για την επικοινωνία με το υποσύστημα LocHandler από το οποίο λαμβάνει την κατάλληλη απάντηση που θα εκφωνήσει, καθώς και με το υποσύστημα AUI στο οποίο επικοινωνεί την απάντηση αυτή.



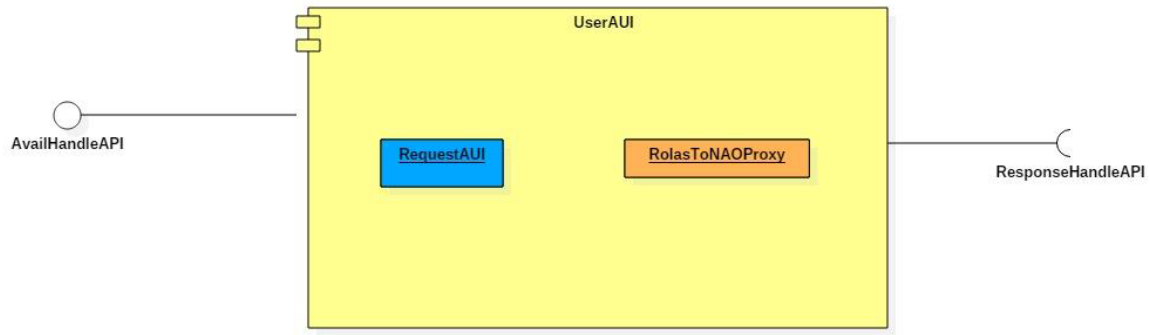
Ολικό διάγραμμα Τμημάτων LocationPackage



2.2.15 **Διάγραμμα Τμημάτων** AvailabilityPackage

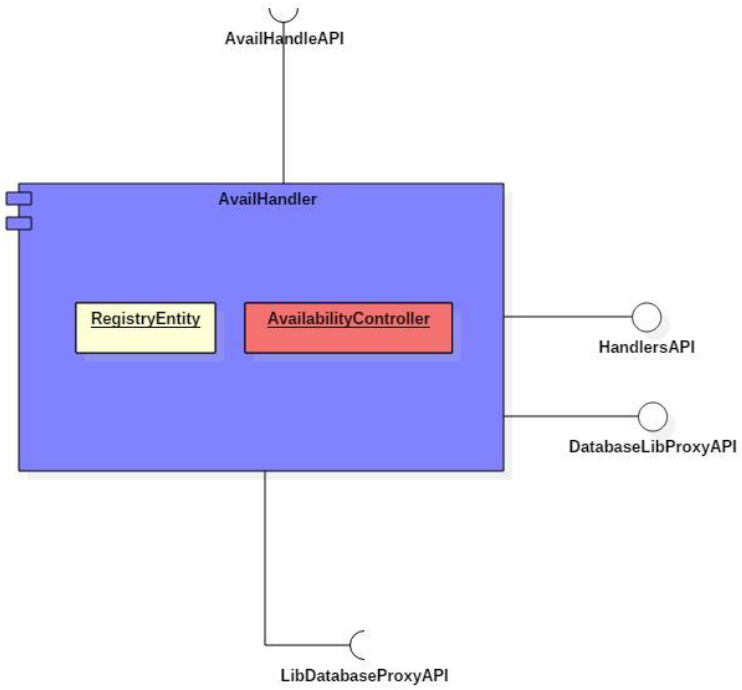
2.2.15.1 **Υποσύστημα** UserAUI

Το παρακάτω υποσύστημα περιέχει τις κλάσεις RequestAUI και RolasToNAOProxy. Το υποσύστημα αυτό είναι υπεύθυνο για την επικοινωνία του χρήστη με το σύστημα μέσω του NAO, δηλαδή την υποδοχή των εντολών του χρήστη και την εκφώνηση της απάντησης του συστήματος.



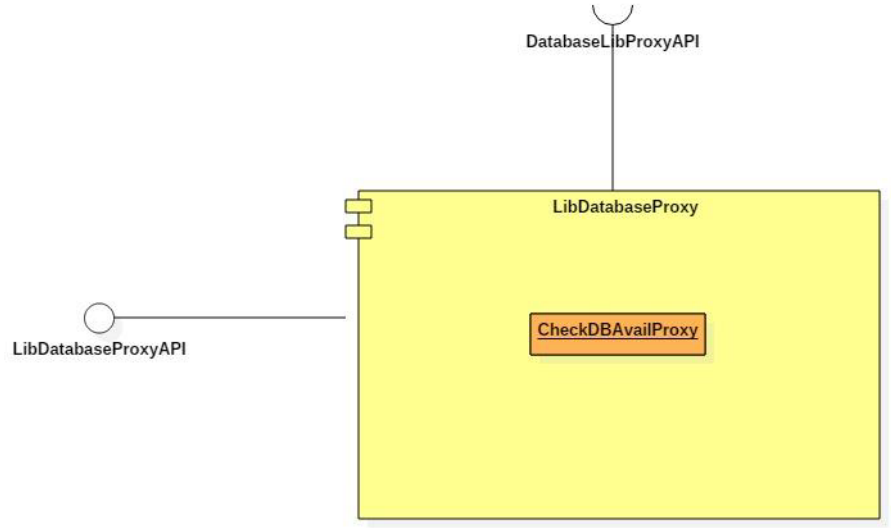
2.2.15.2 **Υποσύστημα** AvailHandler

Το παρακάτω υποσύστημα περιέχει τις κλάσεις RegistryEntity και LocationController. Το υποσύστημα αυτό είναι υπεύθυνο για την λήψη των πληροφοριών της εντολής του χρήστη, την επικοινωνία με τη βάση δεδομένων (και για το query, αλλά και για το answer), καθώς και για τη σύνδεση με το υποσύστημα ResponseHandler.



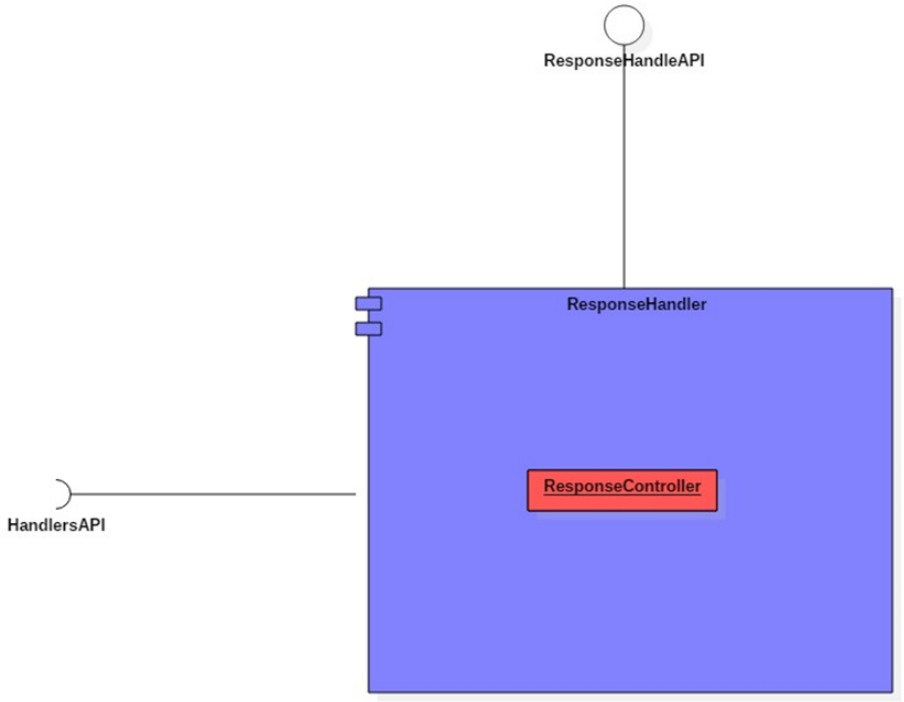
2.2.15.3**Υποσύστημα** LibDatabaseProxy

Το παρακάτω υποσύστημα περιέχει την κλάση CheckDBAvailProxy. Το υποσύστημα αυτό είναι υπεύθυνο για την επικοινωνία με το AvailHandler και τη βάση δεδομένων.

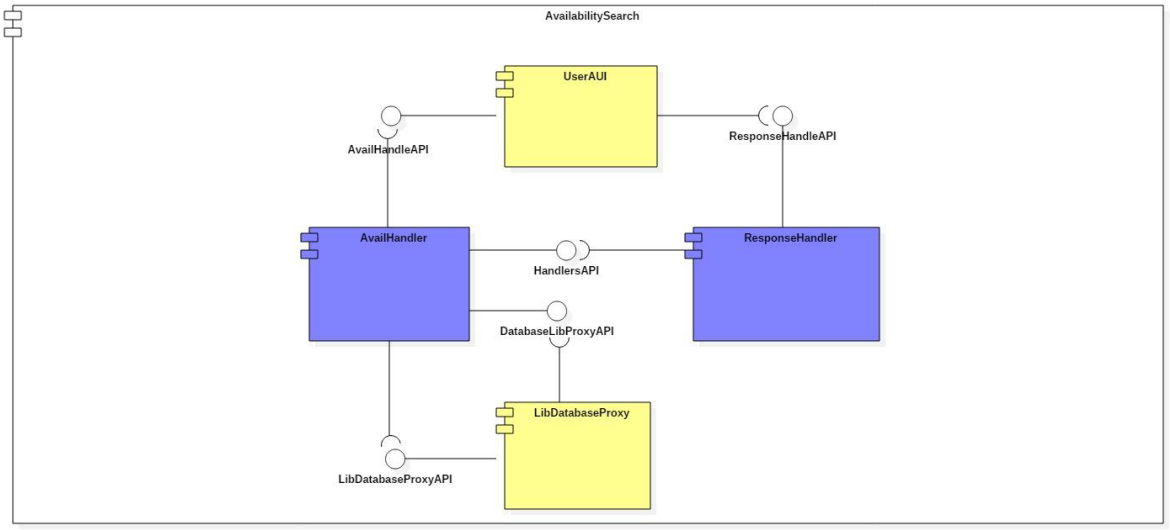


2.2.15.4 **Υποσύστημα** ResponseHandler

Το παρακάτω υποσύστημα περιέχει την κλάση ResponseController. Το υποσύστημα αυτό είναι υπεύθυνο για την επικοινωνία με το υποσύστημα AvailHandler από το οποίο λαμβάνει την κατάλληλη απάντηση που θα εκφωνήσει, καθώς και με το υποσύστημα AUI στο οποίο επικοινωνεί την απάντηση αυτή.

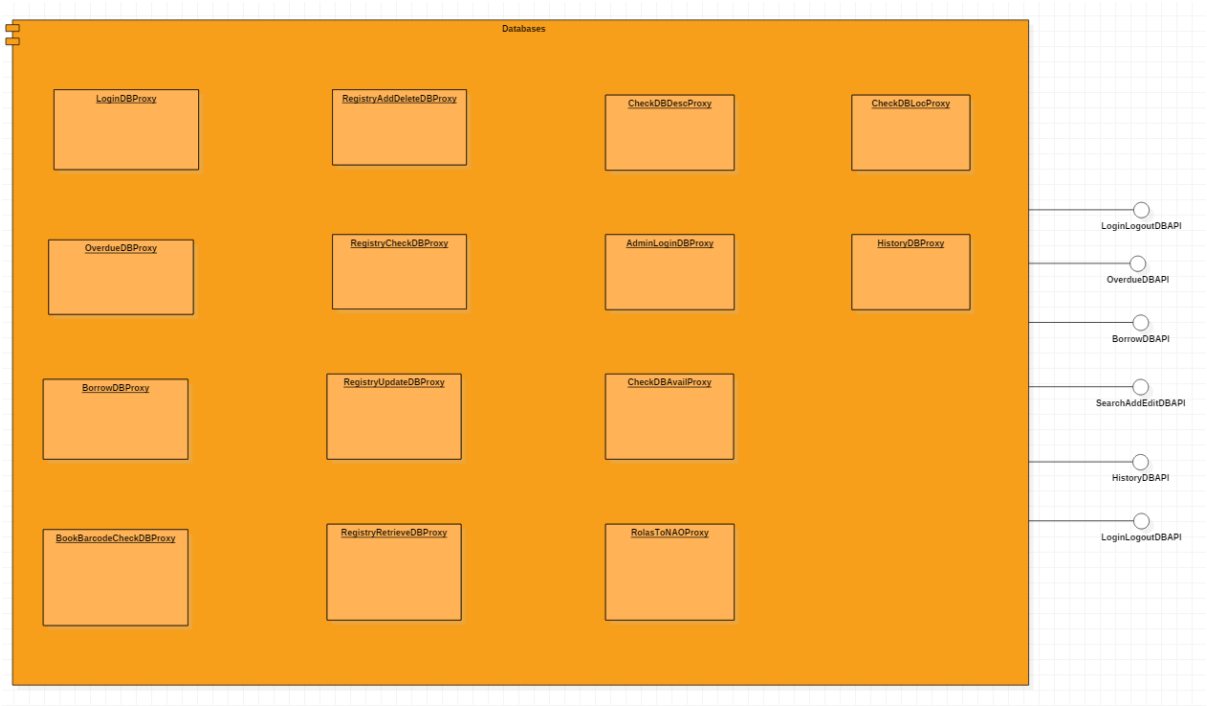


Ολικό διάγραμμα Τμημάτων AvailabilityPackage :

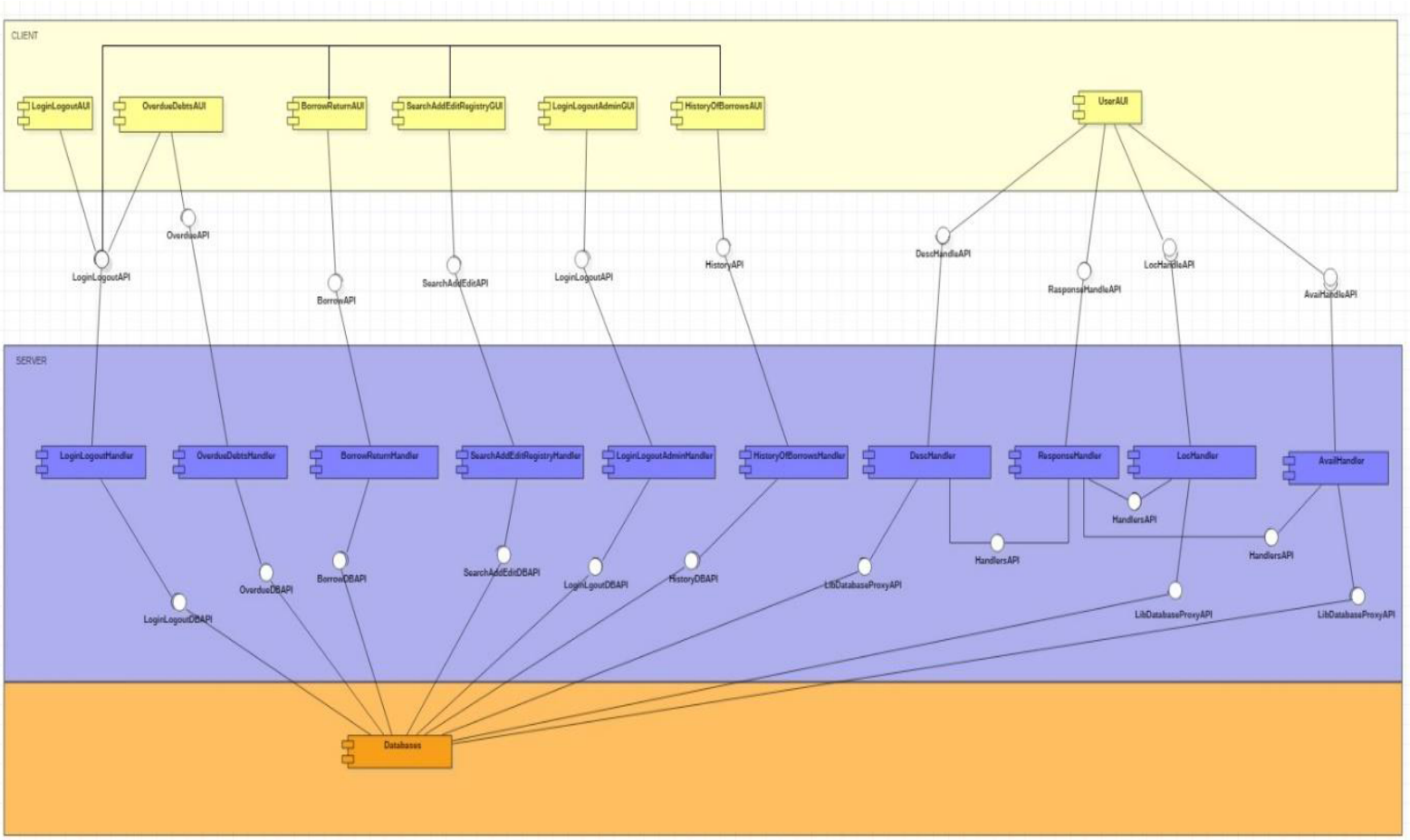


2.2.16 **Υποσύστημα** Databases

Όλα τα υποσυστήματα που προαναφέρθηκαν προκειμένου να διεκπεραιώσουν τις λειτουργίες τους επικοινωνούν με μία βάση δεδομένων. Στο παρακάτω σχήμα παρουσιάζονται όλα τα databaseProxy που έχουν δημιουργηθεί προκειμένου να υπάρξει διασύνδεση των υποσυστημάτων με τη βάση δεδομένων του συστήματος. Αναλυτικότερα έχουμε: την LoginDBProxy που εξυπηρετεί την διαδικασία εισόδου του χρήστη στο σύστημα, την OverdueDBProxy εξυπηρετεί τον έλεγχο των ληξιπρόθεσμων οφειλών , την BorrowDBProxy για την διαδικασία δανεισμού βιβλίων, την BookBarcodeCheckDBProxy που είναι υπεύθυνη για τον έλεγχο και την εύρεση του δοσμένου ραβδοκώδικα στην database ,την RegistryAddDeleteDBProxy για την προσθήκη νέων εγγραφών και την αφαίρεση άλλων από τον Admin του συστήματος , την RegistryCheckDBProxy για τον έλεγχο ύπαρξης της εγγραφής ,την RegistryUpdateDBProxy αποθηκεύει την εγγραφή που επεξεργάστηκε ο διαχειριστής στη βάση δεδομένων, την RegistryRetrieveDBProxy που ανακτά τις εγγραφές , την CheckDBDescProxy που είναι υπεύθυνη για την αναζήτηση περιγραφής ενός βιβλίου, την AdminLoginDBProxy που χρησιμοποιείται για την εύρεση των ονομάτων και των χαρακτηριστικών αριθμών των διαχειριστών του συστήματος , την CheckDBAvailProxy που χρησιμοποιείται για τον έλεγχο της διαθεσιμότητας ενός βιβλίου, RolasToNAOProxy που είναι υπεύθυνο για την αποστολή της κατάλληλης φωνητικής εντολής που θα εκφωνήσει ο NAO στον χρήστη, CheckDBLocProxy που εξυπηρετεί την διαδικασία αναζήτησης της τοποθεσίας ενός βιβλίου και η HistoryDBProxy που είναι υπεύθυνη για την εύρεση του ιστορικού δανεισμών των βιβλίων αν αυτό είναι δυνατό.



Παρακάτω παρατίθεται το διάγραμμα τμημάτων του συστήματος που παρουσιάζει τις εξαρτήσεις μεταξύ όλων των υποσυστημάτων που το αποτελούν.

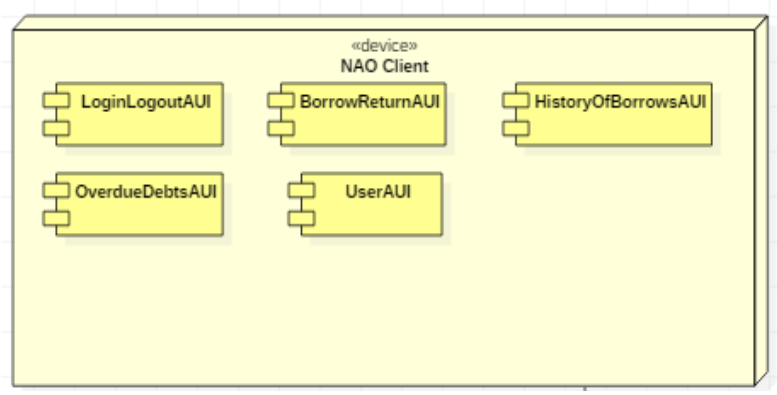


2.3 **Απεικόνιση Υλικού/Λογισμικού**

Παρουσίαση Υλικού/Λογισμικού με κατάλληλα διαγράμματα UML, με σκοπό την αποσαφήνιση του τρόπου διασύνδεσης των τμημάτων του συστήματος αλλά και την συσχέτιση τους με το υλικό του συστήματος.

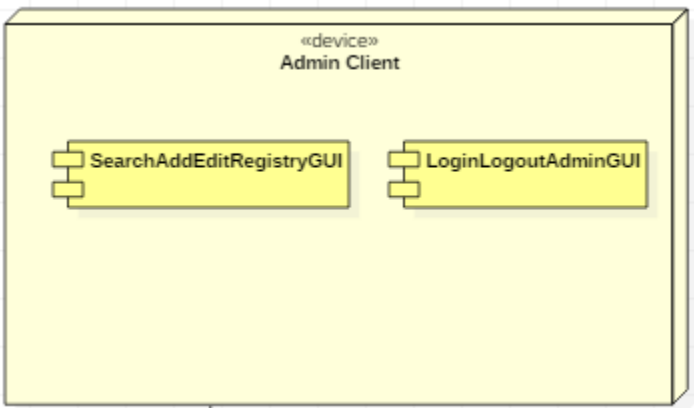
2.3.1 NAO Client

Πρόκειται για τον κόμβο του συστήματος που περιέχει τις διεπαφές που επικοινωνούν με τον εγγεγραμμένο χρήστη αλλά και τον επισκέπτη χρήστη. Ουσιαστικά αφορά τις διεπαφές του NAO με τους δύο παραπάνω χρήστες μέσω των ακουστικών διεπαφών χρήστη (AUI). Η χρήση του γίνεται μέσω της φυσικής συσκευής του Ν.Α.Ο και των περιφερειακών του. Οι διεπαφές αυτές εξυπηρετούν την διαδικασία επικοινωνίας των αιτήσεων του χρήστη προς το σύστημα αλλά και την ενημέρωση του χρήστη από το ίδιο το σύστημα.



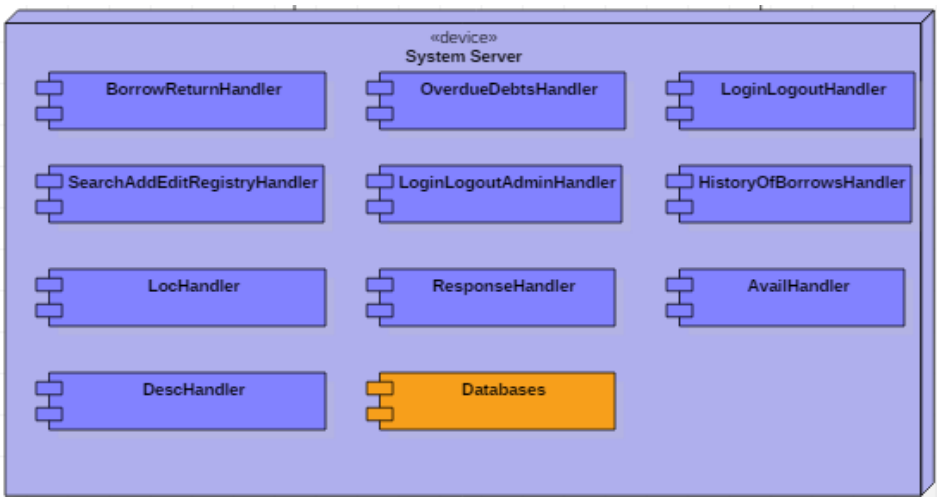
2.3.2 Admin Client

Πρόκειται για τον κόμβο του συστήματος που αφορά την διεπαφή του διαχειριστή με την ιστοσελίδα διαχείρισης της βιβλιοθήκης μέσω της οποίας μπορεί να προσθέσει/αφαιρέσει, να αναζητήσει και να επεξεργαστεί τις εγγραφές του συστήματος. Η χρήση του γίνεται μέσω οποιασδήποτε συσκευής που μπορεί να συνδεθεί στο διαδίκτυο και επακόλουθα μπορεί να συνδεθεί στην ιστοσελίδα διαχείρισης. Οι διεπαφές λαμβάνουν δεδομένα από το διαχειριστή και τα μεταφέρουν στο System Server , όπως επίσης προβάλλουν τα χαρακτηριστικά εγγραφών που αιτείται ο διαχειριστής.



2.3.3 System Server

Πρόκειται για τον κόμβο που αποτελεί που αποτελεί τον κορμό του συστήματος και επικοινωνεί με την βάση δεδομένων με σκοπό την αποθήκευση και την ανάκτηση των αποθηκευμένων στην εξωτερική βάση δεδομένων που αφορούν τις αιτήσεις των χρηστών αλλά και του διαχειριστή. Περιέχει handlers υποσυστήματα που ελέγχουν συνολικά το σύστημα όπως επίσης proxies υποσυστήματα που πλαισιώνουν το σύστημα όσον αφορά την επικοινωνία με την βάση δεδομένων.



Στη συνέχεια παρατίθενται οι ελάχιστες και οι προτεινόμενες απαιτήσεις σε επίπεδο χρήστη και server:

	Minimum Requirements	Recommended Requirements
client(admin)	CPU:Intel Core i3-7100U Processor RAM: 1GB	CPU:Intel Core i5-8400 Processor RAM:2GB
server	CPU:Intel Core i7-7500U Processor RAM: 4GB	CPU:Intel Core i7-8700K Processor RAM: 8GB

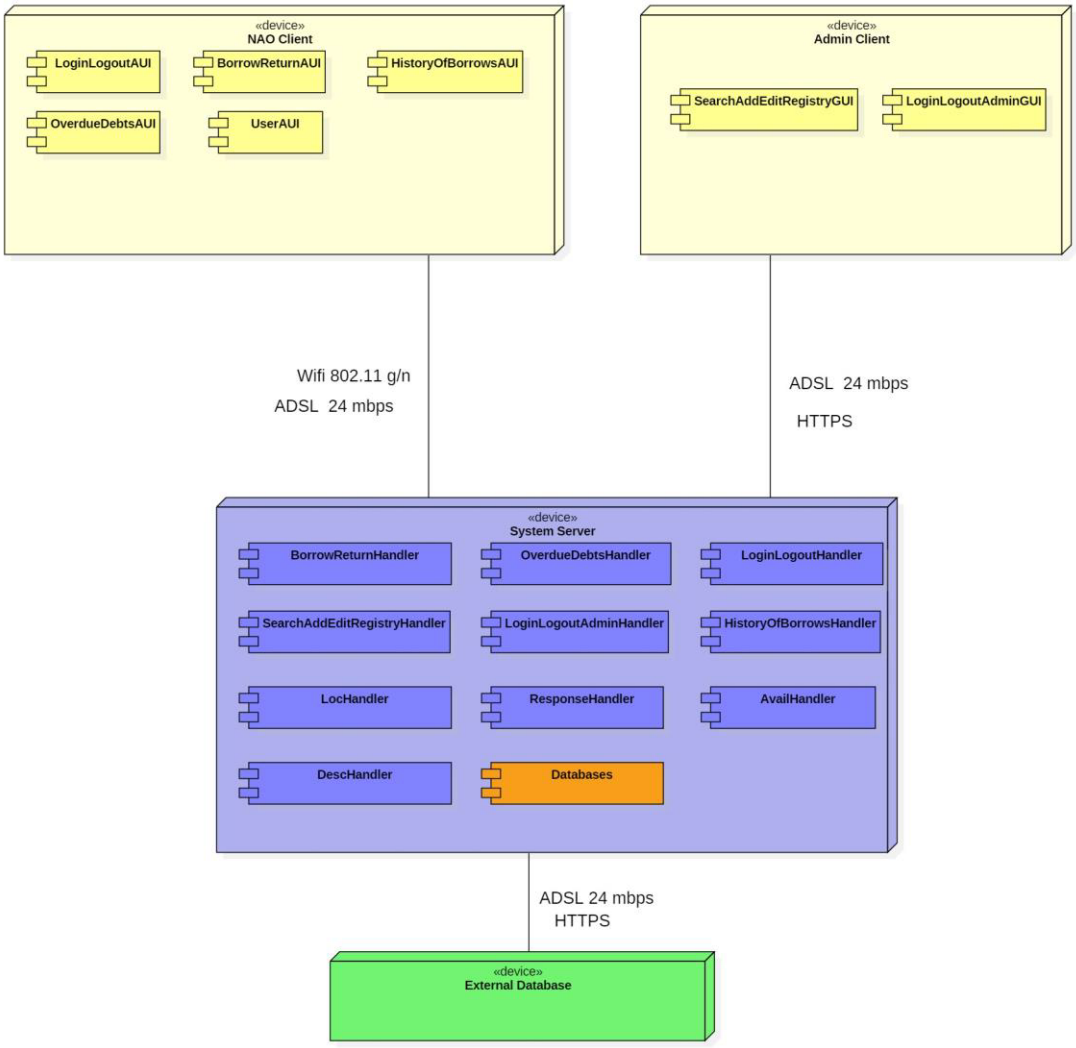
2.3.4 **Συνολικό διάγραμμα ανάπτυξης**

Σε αυτή την παράγραφο παρουσιάζεται ο τρόπος επικοινωνίας μεταξύ των κόμβων του συστήματος.

Ο κόμβος NAO Client με τον κόμβο System Server καθώς επίσης ο κόμβος Admin Client με τον κόμβο System Server επικοινωνούν μέσω των πρωτοκόλλων 802.11 g/n και HTTPS, αντίστοιχα. Οι γραμμές σύνδεσης των κόμβων είναι ADSL μέχρι 24 mbps.

Ενώ ο κόμβος System Server επικοινωνεί με την εξωτερική βάση δεδομένων μέσω του εμπορικού συστήματος διαχείρισης βάσεων δεδομένων MySQL.

Το πρωτόκολλο LAN αναφέρεται στο τοπικό δίκτυο διασύνδεσης υπολογιστών ενώ το πρωτόκολλο HTTPS στο πρωτόκολλο HTTP που όμως έχει επιπλέον στοιχεία ασφαλείας όπως είναι το SSL (Secure Sockets Layer) το οποίο κρυπτογραφεί τα δεδομένα.



2.4 Έλεγχος Πρόσβασης και Ασφάλεια

Ο τρόπος πρόσβασης κάθε χρήστη στο σύστημα διαφέρει:

- Ο επισκέπτης συνδέεται στο σύστημα εκφωνώντας απλά την επιθυμητή εντολή προς το NAO, δηλαδή ουσιαστικά δεν πραγματοποιείται κάποια μορφή αυθεντικοποίησης.
- Ο εγγεγραμμένος χρήστης εισάγεται στο σύστημα με την επίδειξη της ειδικής κάρτας μέλους στο NAO.
- Ο διαχειριστής συνδέεται με την επιβεβαίωση ενός ονόματος χρήστη και ενός κωδικού πρόσβασης, τα οποία εισάγει στη σελίδα πρόσβασης.

Σε συστήματα πολλών χρηστών, όπως το δικό μας, διαφορετικοί δράστες έχουν πρόσβαση σε διαφορετική λειτουργικότητα και διαφορετικά δεδομένα.
Ο έλεγχος πρόσβασης και η ασφάλεια περιγράφουν τη μοντελοποίηση των χρηστών του συστήματος με τη μορφή ενός πίνακα πρόσβασης (access matrix), όπως παρουσιάζεται παρακάτω.

Admin User	LoginGUI	SearchAddEditRegistryPackage	RegistryDeleteConfirmationGUI()
	display()	RegistryNameGUI()	RegistryDeleteConfirmation-GUI()
	backButton()	RegistryNameTextBox()	okButtonClick()
		DisplayNamePromptMes-sage()	DisplayDeleteConfirmation-Message()
	RegistryEditSaveOrCancelGUI	RegistryErrorGUI	RegistryEditGUI
	RegistryEditSaveOrCancel-GUI()	RegistryErrorGUI()	RegistryEditGUI()
	saveRegistryButtonClick()	DisplayNonExistenceMes-sage()	
	cancelButtonClick()		
	displaySavePromptMes-sage()		
	RegistryDisplayGUI		
	RegistryDisplayGUI()		
	display()		
Registered User	VoiceDialog	ErrorLogin	LoginBoundary (OverdueDebts)

	login()	Naoerrorvoice()	OverdueDebts()
	logout()		
	BorrowRequest	BookBarcodeNAORequest	RequestAUI (Loca-tion,Description,Availability)
	requestAUI()	requestBookBarcode()	requestListenerAUI()
	HistoryRequest		
	requestAUI()		
Guest User	RequestAUI (Loca-tion,Description,Availability)		
	requestListenerAUI()		

Η σύνδεση ενός εγγεγραμμένου χρήστη μέσω της επίδειξης της κάρτας μέλους και της σάρωσης του κωδικού QR ακολουθείται από τη μεταφορά της πληροφορίας αυτής στη Library Database (Lib_Database) για την αυθεντικοποίηση του χρήστη. Η μεταφορά αυτή πραγματοποιείται ασύρματα μέσω του πρωτοκόλλου 802.11 και κρυπτογράφησης τύπου SSL.
Με τα ίδια πρωτόκολλα υλοποιείται και η μεταφορά του ονόματος χρήστη και του κωδικού πρόσβασης του διαχειριστή στη βάση δεδομένων, κατά την είσοδο του στο σύστημα.
Η κρυπτογράφηση και η ασφάλεια της Lib_Database είναι παράμετροι που σχετίζονται με την ασφάλεια του δικού μας συστήματος, αλλά όντας εξωτερικό σύστημα, δεν έχουμε δικαιοδοσία καθορισμού της δομής της.

**Υπενθυμίζουμε ότι η* Lib_Database *είναι η βάση δεδομένων του συστήματός μας, δηλαδή ένας κεντρικός υπολογιστής που έχει αποθηκευμένο ένα σύνολο δεδομένων, όπως τα στοιχεία των μελών της βιβλιοθήκης, τα βιβλία που αυτοί έχουν δανειστεί κ.α. , καθώς και τη βάση δεδομένων με τα στοιχεία αυθεντικοποίησης τους. Στην ουσία ενυπάρχουν υπο-βάσεις δεδομένων με διαφορετική λειτουργικότητα.*

2.5 **Οριακές Συνθήκες**

Προκειμένου να εξασφαλίσουμε την αξιοπιστία του συστήματος μας πρέπει να προσδιορίσουμε πλήρως την απόκρισή του στις οριακές συνθήκες.

2.5.1 **Τερματισμός συστήματος**

Το σύστημα τερματίζεται όταν ο διαχειριστής επιλέξει μέσα από την αρχική σελίδα της ιστοσελίδας διαχείρισης της βιβλιοθήκης το πλήκτρο «Τερματισμός Ν.Α.Ο». Εάν κατά την επιλογή του τερματισμού λειτουργίας το σύστημα εξυπηρετεί κάποιον χρήστη ή βρίσκεται σε διαδικασία μεταβολής των στοιχείων της βάσης δεδομένων τότε η διαδικασία τερματισμού τίθεται σε αναμονή. Μόλις διεκπεραιωθούν οι διαδικασίες μεταβολής ή εξυπηρετηθούν οι χρήστες τότε το σύστημα αποδεσμεύεται από τα εξωτερικά συστήματα και αποσυνδέει τον χρήστη.

Σύντομη Περιγραφή:	Στόχος του σεναρίου χρήσης είναι ο τερματισμός λειτουργίας του συστήματος.
Πυροδότηση Δραστηριότητας:	Επιλογή του πλήκτρου «Τερματισμός Ν.Α.Ο».
Προϋπόθεση:	Ο διαχειριστής πρέπει να έχει εκκινήσει το σύστημα.

Βασική Ροή:		
Γραμμή	**Ενέργεια χρήστη συστήματος**	**Απάντηση Συστήματος**
1	Ο χρήστης πατάει το πλήκτρο «Τερματισμός Ν.Α.Ο».	Το σύστημα αποσυνδέεται από τα εξωτερικά συστήματα.
2		Το σύστημα αποσυνδέει τον χρήστη από το σύστημα.
Μετέπειτα κατάσταση:	Το σύστημα έχει τερματιστεί.	

2.5.2 **Διακοπή τροφοδοσίας**

Σε περίπτωση διακοπής τροφοδοσίας της συσκευής ενδέχεται να υπάρξει απώλεια πληροφοριών που δεν έχουν αποθηκευτεί στη βάση δεδομένων.

2.5.3 **Σφάλματα Λογισμικού**

Κατά τη διάρκεια αποθήκευσης ή ανάκτησης των εγγραφών από τη βάση δεδομένων ενδέχεται να υπάρξουν σφάλματα λογισμικού. Τα σφάλματα αυτά προκαλούνται από την απουσία σύνδεσης του συστήματος με την εξωτερική βάση δεδομένων κατά τη διάρκεια αποθήκευσης ή ανάκτησης από τη βάση δεδομένων. Σε μία τέτοια κατάσταση το σύστημα προσπαθεί να συνδεθεί ξανά στη βάση δεδομένων, εμφανίζει μήνυμα στην αρχική σελίδα της ιστοσελίδας που συνδέεται ο διαχειριστής και ενημερώνει με ηχητικό μήνυμα μέσω του Ν.Α.Ο. τους χρήστες. Στη συνέχεια το σύστημα επανεκκινεί.

Σύντομη Περιγραφή:	Στόχος του σεναρίου χρήσης είναι η αντιμετώπιση σφαλμάτων λογισμικού.
Πυροδότηση Δραστηριότητας:	Το σύστημα δεν μπορεί να συνδεθεί με τα εξωτερικά συστήματα.
Προϋπόθεση:	Ο διαχειριστής πρέπει να έχει εκκινήσει το σύστημα.

Βασική Ροή:		
Γραμμή	**Ενέργεια χρήστη συστήματος**	**Απάντηση Συστήματος**
1	Ο χρήστης αιτείται από το σύστημα την επιστροφή ή το δανεισμό βιβλίου.	Το σύστημα αδυνατεί να συνδεθεί στην εξωτερική βάση δεδομένων.
		Το σύστημα επιχειρεί ξανά να επικοινωνήσει με την βάση δεδομένων.
2		Το σύστημα εμφανίζει μήνυμα σχετικό με το σφάλμα στον διαχειριστή.
3		Το σύστημα ανακοινώνει στον χρήστη μέσω του Ν.Α.Ο μήνυμα σχετικό με το σφάλμα που συνέβη.
4		Το σύστημα επανεκκινεί.
Μετέπειτα κατάσταση:	Το σύστημα βρίσκεται στην κατάσταση αναμονής νέου χρήστη.	

3. **Πίνακας ιχνηλασιμότητας εγγράφων Σχεδίασης και Απαιτήσεων Λογισμικού**

Οι αλλαγές που σημειώθηκαν κατά τη μετάβαση από το έγγραφο Απαιτήσεων Λογισμικού στο έγγραφο Σχεδίασης Συστήματος φαίνονται παρακάτω :

1. Στην κλάση BorrowControll προσθέσαμε τη μέθοδο getOverdueDebts() όπως φαίνεται και από το διάγραμμα ακολουθιών του πακέτου Borrow.
2. Στην κλάση RegistryEntity προσθέσαμε τη μέθοδο getRegistryAttributes() όπως φαίνεται και από το διάγραμμα ακολουθιών.

4. **Παράρτημα** Ι – **Ανοιχτά Θέματα**

Δεν παρουσιάστηκαν κάποια ανοικτά θέματα σε αυτή τη φάσης της ανάπτυξης του συστήματός μας.