ΑΛΕΞΑΝΔΡΕΙΟ Τ.Ε.Ι. ΘΕΣΣΑΛΟΝΙΚΗΣ
ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΦΑΡΜΟΓΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Υπηρεσία Sms-Server για το Αλεξάνδρειο ΤΕΙ Θεσσαλονίκης

Του φοιτητή

Οσμαντσικίδη Πέτρου

ΑΜ: 123853

Επιβλέποντες :

Δημήτρης Α. Δέρβος

Ουγιάρογλου Στέφανος

ΠΕΡΙΕΧΟΜΕΝΑ

Κεφάλαιο 1 : Εισαγωγή 5
Κεφάλαιο 2 : Αρχιτεκτονική εφαρμογής 6
Κεφάλαιο 3 : Τεχνολογίες υλοποίησης 8
3.1 : Τεχνολογίες Backend 8
3.1.1 : Java 9
3.1.2 : .NET 9
3.1.3 : Python 10
3.1.4 : Ruby 11
3.1.5 : PHP 11
3.1.6 : Node.js 12
3.1.7 : Επιλογή και αξιοποίηση της Java στην εφαρμογή 12
3.2 : Java EE 13
3.2.1 : Ορολογία 13
3.2.2 : Χρήση της Java EE στην υπηρεσία 13
3.2.3 : Επιλογή και αξιοποίηση Java EE αντί Spring Framework 14
3.3 : Application Servers 14
3.3.1 : Java application servers 15
3.3.2 : Jboss (WildFly) 16
3.3.3 : Glassfish 16
3.3.4 : Apache Tomcat / TomEE 16
3.3.5 : Payara Server 17
3.3.6 : Επιλογή και αξιοποίηση Payara Server στην εφαρμογή 17
3.4 : Dependency Injection 18
3.4.1 : Ορισμός 18
3.4.2 : EJB 3 18
3.4.3 : Αξιοποίηση EJB στην εφαρμογή 20
3.5 : Java Build Tools 20
3.5.1 : Ant 21

3.5.2 : Maven 21
3.5.3 : Gradle 22
3.5.4 : Επιλογή και αξιοποίηση Maven στην εφαρμογή 22
3.6 : Βάση Δεδομένων 24
3.6.1 : Ορισμός 24
3.6.2 : MySQL 24
3.6.3 : Αξιοποιήση της MySQL στο project 25
3.6.4 : JDBC Connection Pooling 26
3.7 : Web Services 27
3.7.1 : REST 28
3.7.2 : SOAP 28
3.7.3 : Επιλογή και αξιοποίηση REST στην εφαρμογή 29
3.8 : Δημιουργία ιστότοπου 30
3.8.1 : HTML 30
3.8.2 : JSP 31
3.8.3 : Σύγκριση και αξιοποίηση HTML στην εφαρμογή 31
3.9 : jQuery 32
3.9.1 : Ορισμός και χαρακτηριστικά 32
3.9.2 : Αξιοποίηση jQuery στην εφαρμογή 33
3.10 : Ajax 33
3.10.1 : Ορισμός και χαρακτηριστικά 33
3.10.2 : Αξιοποίηση στην εφαρμογή 34
3.11 : Bootstrap 35
3.11.1 : Ορισμός και χαρακτηριστικά 35
3.11.2 : Αξιοποίηση στην εφαρμογή 36
3.12 : Cookies 36
3.12.1 : Ορισμός και χαρακτηριστικά 36
3.12.2 : Αξιοποίηση στην εφαρμογή 37
Κεφάλαιο 4 : Υπηρεσίες που εξυπηρετούνται 37
4.1 : Ακαδημαϊκά Θέματα 38

4.1.1 : Ανάκτηση ακαδημαϊκών δεδομένων από φοιτητές 38
4.1.2 : Ενημέρωση ομάδων φοιτητών για ακαδημαϊκά θέματα 38
4.1.3 : Στοχευμένα μηνύματα 39
Κεφάλαιο 5 : Η εφαρμογή 40
5.1 : Ανάκτηση δεδομένων μέσω SMS 40
5.2 : Ενημέρωση ομάδων φοιτητών από καθηγητές 43
5.3 : Ενημέρωση αιμοδοτών για νέες αιμοδοσίες 48
5.4 : Στοχευμένη αποστολή SMS 49
5.5 : Καταχώρηση / Ενημέρωση νούμερου κινητού τηλεφώνου 52
Κεφάλαιο 6 : Τεκμηρίωση, Συντήρηση, Επεκτασιμότητα 54
6.1 : Ρυθμίσεις για χρήση Java ως backend 54
6.2 : Dependencies 56
6.3 : Το API της εφαρμογής 57
6.3.1 : Συνοπτικά 58
6.3.2 : Endpoint SMS Aggregator 59
6.3.3 : Endpoint ιστοσελίδας 60
6.4 : Υλοποίηση διασύνδεσης με SMS Aggregator 65
6.4.1 : Οδηγίες σύνδεσης Error! Bookmark not defined.
6.4.2 : Αποστολή SMS 65
6.4.3 : SMS Forward 68
6.4.4 : Delivery Reports 70
6.5 : Mobile Originated Υπηρεσίες 72
6.5.1 : Αίτημα για εισαγωγή νέας υπηρεσίας στο σύστημα 72
6.5.2 : Ρυθμίσεις υπηρεσίας 74
6.5.3 : Εισαγωγή στο δυναμικό XML 76
6.5.4 : Ανάκτηση δυναμικού XML 78
6.5.5 : Διαδικασία απάντησης ερωτημάτων 79
6.5.6 : Εισαγωγή νέας βάσης για ερωτήματα 82
6.6 : Mobile Terminated Υπηρεσίες 82
6.6.1 : Εισαγωγή νέας υπηρεσίας στην εφαρμογή. 82

6.6.2 : Αποστολή μαζικών μηνυμάτων προς φοιτητές 83
6.6.3 : Αποστολή μαζικών μηνυμάτων προς αιμοδότες 85
6.6.4 : Αποστολή στοχευμένων μηνυμάτων 86
6.7 : Database Logs 86
6.7.1 : Βάση δεδομένων 86
6.7.2 : Χρονοπρογραμματισμένα logs 87
6.8 : Σύστημα αυθεντικοποίησης 88
6.8.1 : Αυθεντικοποίηση χρήστη 88
6.8.2 : Δημιουργία και αξιοποίηση JSON Web Token 90
6.8.3 : Ασφάλεια μεθόδων μέσω Interceptor 91
6.9 : Server Documentation 92
6.9.1 : Διαχείριση Payara Server 92
6.9.2 : Admin Interface Error! Bookmark not defined.
6.9.3 : Deployment της εφαρμογής 93
6.9.4 : Συνδέσεις βάσεων δεδομένων 94
6.9.5 : Server Logs 97
6.9.6 : SSL 99
6.9.7 : Ιστοσελίδα 101
Κεφάλαιο 7 : Επίλογος 101
Κεφάλαιο 8 : Βιβλιογραφία 102

Κεφάλαιο 1 : Εισαγωγή

Το αντικείμενο με το οποίο ασχολείται η παρούσα πτυχιακή εργασία είναι η δημιουργία εφαρμογής η οποία θα λειτουργεί ως sms-server και θα εξυπηρετεί τις ανάγκες άμεσης πληροφόρησης των μελών της ακαδημαϊκής κοινότητας του ΑΤΕΙ/Θ μέσω της υπηρεσίας SMS Aggregator του GUNET (https://sms.gunet.gr/), η οποία παρέχει σε όλα τα μέλη της Ακαδημαϊκής κοινότητας της χώρας την δυνατότητα για αποστολή και λήψη σύντομων γραπτών μηνυμάτων (SMS). Η εφαρμογή που υλοποιείται παρέχει την δυνατότητα για άμεση και προσωποποιημένη ενημέρωση των μελών του της Ακαδημαϊκής κοινότητας μέσω γραπτών προκαθορισμένων μηνυμάτων ερωτήσεων προς αυτήν, καθώς και μέσω της αποστολής μαζικών ή στοχευμένων μηνυμάτων της εφαρμογής προς αυτών.

Συγκεκριμένα, όσον αφορά την άμεση πληροφόρηση μέσω μηνυμάτων ερωτήσεων, η εφαρμογή παρέχει μια σειρά τυποποιημένων ερωτημάτων τα οποία επιστρέφουν απαντήσεις περιορισμένης οπτικής από το περιεχόμενο των βάσεων δεδομένων εφαρμογών του ιδρύματος, όπως η Πυθία και η Εθελοντική Αιμοδοσία. Μέσω αυτών των ερωτημάτων, οι χρήστες μέλη της Ακαδημαϊκής κοινότητας του ΑΤΕΙ/Θ μπορούν να στέλνουν κάποιο από τα καθορισμένα μηνύματα προς την εφαρμογή μέσω κινητού τηλεφώνου με την μορφή SMS, και να παραλαμβάνουν πάλι σε μορφή μηνύματος προσωποποιημένη πληροφόρηση σύμφωνα με το ερώτημα τους και την ύπαρξη η μη της πληροφορίας που ζητήθηκε.

Όσον αφορά την μαζική ή στοχευμένη ενημέρωση των μελών, η εφαρμογή παρέχει την δυνατότητα τους καθηγητές/μέλη διοικητικού προσωπικού να αποστέλλουν προκαθορισμένα ενημερωτικά μηνύματα που αφορούν ακαδημαϊκούς σκοπούς/λειτουργίες προς άλλα μέλη που ανήκουν στην ακαδημαϊκή κοινότητα. Η εφαρμογή χρησιμοποιεί τις βάσεις δεδομένων εφαρμογών του ιδρύματος για την εύρεση των δηλωμένων κινητών τηλεφώνων των μελών, και με συνδυασμό με τις ιδιότητες αυτών μπορεί να στέλνει τα συγκεκριμένα μηνύματα προς όλα αυτά τα μέλη που τους αφορούν. Αυτά τα ενημερωτικά μηνύματα μπορούν να σταλούν είτε προς ομάδες φοιτητών που είναι εγγεγραμμένοι στην βάση της πυθίας, είτε σε ομάδες εθελοντών αιμοδοτών που είναι εγγεγραμμένοι στην βάση της Εθελοντικής αιμοδοσίας. Η πρόσβαση σε αυτές τις λειτουργίες παρέχεται μέσω της web εφαρμογής που έχει αναπτυχθεί για την παροχή τους.

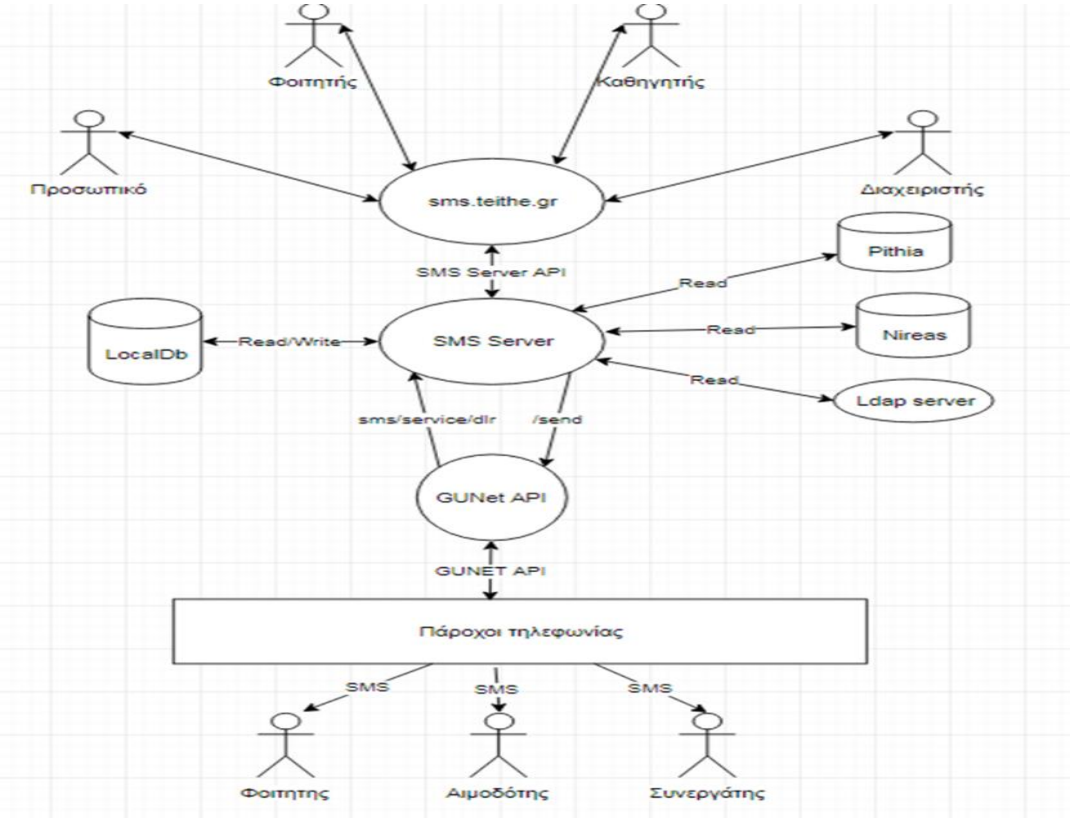
Παράλληλα με τις παραπάνω λειτουργίες που παρέχονται, η εφαρμογή διατηρεί και διαχειρίζεται τα δεδομένα καταγραφής χρήσης αυτών των λειτουργιών, και προσφέρει την δυνατότητα ανάλυσης και επεξεργασίας τους, για σκοπούς καταγραφής και βελτίωσης των ερωτημάτων, καθώς και για την βελτίωση του τρόπου λειτουργίας της ίδιας της εφαρμογής.

Ο σχεδιασμός της εφαρμογής προς τις λειτουργίες που παρέχει, καθώς και η ανάπτυξη της, με την παροχή πρόσβασης στις ακαδημαϊκές βάσεις δεδομένων και την παραχώρηση του server στον οποίο θα εκτελείται, έγινε σε συνεργασία με το Κέντρο Διαχείρισης Δικτύων του ΑΤΕΙ/Θ.

Κεφάλαιο 2 : Αρχιτεκτονική εφαρμογής

Η εφαρμογή λειτουργεί ως διασύνδεση μεταξύ του συστήματος γραμματείας του ιδρύματος (pithia) και της αιμοδοσίας (nireas) με την υπηρεσία αποστολής μηνυμάτων που παρέχει το GUNET (SMS Aggregator) , καθώς και με την εφαρμογή web που δημιουργήθηκε για την παροχή αυτών των υπηρεσιών στους φοιτητές / καθηγητές του ιδρύματος. Η βασική συνδεσμολογία της εφαρμογής φαίνεται παρακάτω (Εικόνα 2.1)

Εικόνα 2.1



Η επικοινωνία με τον SMS Aggregator γίνεται μέσω διαδικτύου προς και από τοκαθορισμένο REST API συνδεσιμότητας που μας παρέχει το GUNET, με διαπιστευτήρια που παρέχονται αποκλειστικά για την επικοινωνία του ΑΤΕΙ/Θ με την εφαρμογή του GUNET. Η πρόσβαση σε αυτό το API είναι δυνατή μόνο εάν η εφαρμογή βρίσκεται σε κάποιο ορισμένο subnet δικτύου, που έχει οριστεί μέσα στο εύρος διευθύνσεων του ΑΤΕΙ/Θ.

Η επικοινωνία της εφαρμογής με την ιστοσελίδα (sms.teithe.gr) γίνεται μέσω του REST API που δημιουργήθηκε και εξηπηρετεί τις ανάγκες τις ιστοσελίδας για την πρόσβαση στις υπηρεσίες που παρέχονται από την εφαρμογή, τα απαραίτητα δεδομένα για την χρήση αυτών των υπηρεσιών ,καθώς και την αυθεντικοποίηση των χρηστών.

Η επικοινωνία με τις βάσεις pithia και nireas γίνεται μέσω του JDBC Protocol. Η επικοινωνία σε αυτές τις βάσεις δεδομένων μπορεί να πραγματοποιηθεί μόνο μέσα από το εσωτερικό δίκτυο του ιδρύματος, όπου και βρίσκεται εγκατεστημένη η εφαρμογή. Το μοντέλο επικοινωνίας με αυτές τις βάσεις είναι Read-Only, και η εφαρμογή μπορεί να λαμβάνει ακαδημαικά δεδομένα σε πραγματικό χρόνο, καθώς και δεδομένα για τους εθελοντές αιμοδότες. Η επικοινωνία με την τοπική βάση είναι επίσης μέσω του JDBC, και είναι Read-Write, και χρησιμοποιείται ώστε να καλυφθούν οι ανάγκες της εφαρμογής για εγγραφή

δεδομένων σε διάφορες λειτουργίες της, που δεν μπορούν να εξηπηρετηθούν λόγω του περιορισμού στην εγγραφή στις ακαδημαικές βάσεις, καθώς και για καταγραφή στοιχείων λειτουργίας της.

Η επικοινωνία με τον LDAP Server (σύστημα ταυτοποίησης χρηστών που χρησιμοποιεί το ΑΤΕΙ/Θ) γίνεται μέσω του πρωτόκολλου LDAP, και χρησιμοποιείται ως εναλλακτικός τρόπος ταυτόποιησης χρηστών στην ιστοσελίδα. Η σύνδεση στον LDAP Server μπορεί να πραγματοποιηθεί μόνο από κάποια εσωτερική IP του δικτύου του ΑΤΕΙ/Θ, όπου και βρίσκεται η εφαρμογή, και υπάρχουν δικαιώματα μόνο ελέγχου διαπιστευτηρίων χρήστη.

Όπως φάινεται και στο σχεδίαγραμμα αρχιτεκτονικής (Εικόνα 2.1) η εφαρμογή δεν επικοινωνεί απέυθειας με κάποιο πρωτοκολλο SMS για την παραλαβή και αποστολή μηνυμάτων. Η επικοινωνία με τους εμπλεκόμενους χρήστες της εφαρμογής για αυτές τις ενέργειες γίνεται μόνο μέσω του API του GUNET, κάτι που σημαίνει ότι η επιτυχημένη παραλαβή και αποστολή εξαρτώνται από την πλατφόρμα του SMS Aggregator και της υλοποίησης τους για επικοινωνία μεσω το πρωτόκολλου SMS με τους χρήστες.

Κεφάλαιο 3 : Τεχνολογίες υλοποίησης

Σε αυτό το κεφάλαιο περιγράφονται οι διάφορες τεχνολογίες που χρησιμοποιήθηκαν για την υλοποίηση της παρούσας πτυχιακής, καθώς και θα γίνει και σύγκριση με τις υπάρχουσες ανάλογες τεχνολογίες που υπήρχαν κατά την υλοποίησή της.

Στις εκάστοτε συγκρίσεις, πρέπει αναλογιστούν οι προϋποθέσεις και οι στόχοι που υπάρχουν για την εφαρμογή, η οποία θα χρησιμοποιηθεί στην παραγωγή και θα διατηρηθεί από τρίτα πρόσωπα, οπότε πρέπει να χρησιμοποιηθούν τεχνολογίες οι οποίες είναι κατανοητές και διαδεδομένες, καθώς και αρκετά ώριμες ώστε να επιτευχθεί η μακροζωία αυτής.

3.1 : Τεχνολογίες Backend

Η σημαντικότερη απόφαση είναι η επιλογή της γλώσσας που θα χρησιμοποιηθεί για την υλοποίηση του backend. Υπάρχουν αρκετές γλώσσες που έχουν την δυνατότητα να υλοποιήσουν αυτά που χρειάζεται η εφαρμογή. Αυτή η επιλογή θα γίνει παίρνοντας υπόψιν τις παρακάτω παραμέτρους.

- Τα χαρακτηριστικά της γλώσσας.
- Το τοπικό οικοσύστημα της.
- Το πρόβλημα που επιλύεται.
- Την υπάρχουσα γνώση της τεχνολογίας.
- Τους πόρους που διατίθενται για την εφαρμογή.

Παρακάτω παρουσιάζονται οι πιο διαδεδομένες γλώσσες δημιουργίας backend εφαρμογών.

3.1.1 : Java

Η Java υπάρχει από το 1995 και είναι ένα από τα κύρια εργαλεία παραγωγής εφαρμογών παραγωγής από την άνθηση του J2EE.

Η μακρά διάρκεια ζωής και η ευρεία χρησιμοποίηση της Java έχει παράγει ένα τεράστιο οικοσύστημα από documentation, βιβλιοθήκες και framework, πολλά από τα όποια στοχεύουν στο ηλεκτρονικό εμπόριο, στην ασφάλεια και σε περίπλοκες αρχιτεκτονικές συναλλαγών. Υπάρχουν έμπειροι προγραμματιστές Java παντού στον κόσμο, είτε ως full time είτε ως συνεργάτες εταιριών. Η open source κοινότητα συμμετείχε στην Java από την αρχή, φτιάχνοντας μία άφθονη αγορά για λύσεις και εργαλεία Java.

Ο αρχικός στόχος της Java ήταν το to “write once, run everywhere”, αν και το αν πέτυχε είναι αντικείμενο debate. Επιπλέον άλλες γλώσσες μπορούν να τρέξουν πάνω στο JVM(Java Virtual Machine) και να εκμεταλλευθούν ήδη υπάρχων βιβλιοθήκες Java.

Χαρακτηριστικά της Java:

- Στατική
- Πολλές γλώσσες πάνω στο JVM
- Καλό documentation
- Μεγάλο community
- Βαριά frameworks
- Multi-Threaded
- Multi-Platform

3.1.2 : .NET

Η πλατφόρμα .NET της Microsoft χρησιμοποιείται σε εταιρικές εφαρμογές που τρέχουν πάνω στο λειτουργικά Microsoft Windows. Για τους χρήστες Linux υπάρχει πλέον δυνατότητα μέσω του Mono Project. Παρόλο που η C# και άλλα κομμάτια του .NET είναι ανοιχτά σε standard ανοιχτού κώδικα, η .NET είναι κυρίως ένα κλειστό οικοσύστημα που ελέγχεται από την Microsoft.

Καθώς έχει υιοθετηθεί ευρέως από εταιρίες, το documentation και η υποστήριξη της γλώσσας είναι πλέον εκτενής. Επειδή το .NET είναι ένα framework που τρέχει πάνω στην εικονική πλατφόρμα γνωστή και ως Common Language Infrastructure(CLI) υπάρχει δυνατότητα να χρησιμοποιηθούν διάφορες γλώσσες, όπως C#, VB.NET, C++, μέχρι και Python ή Ruby. Η C# είναι η πιο διαδομένη από αυτές, ακολουθούμενη από την VB.NET.

Πρέπει να σημειωθεί, ότι ενώ υπάρχουν δωρεάν εργαλεία και βιβλιοθήκες, το οικοσύστημα είναι βασισμένο σε αδειοδοτημένο μοντέλο λογισμικού, που σημαίνει ότι όταν χρειάζονται third-party εργαλεία, θα χρειαστεί να πληρωθούν. Επίσης, καθώς θα πρέπει να χρησιμοποιηθεί το λογισμικό Windows για να τρέξει, αυτό σημαίνει επιπλέον κόστος(σε αντίθεση με το δωρεάν Linux)

- Χαρακτηριστικά .NET
- Στατική
- Πολλαπλές γλώσσες
- Windows OS(Mono στα Linux)
- Καλό documentation & Υποστήριξη
- Multi-Threaded
- Αδειοδοτημένο μοντέλο λογισμικού

3.1.3 : Python

Η Python είναι μία ανοιχτού λογισμικού ερμηνευτική(interpretive) γλώσσα που έχει λάβει ευρείας προτίμησης από την επιστημονική κοινότητα λόγω της ευκολίας εκμάθησής της καθώς και του πλήθους των επιστημονικών βιβλιοθηκών που διαθέτει.

Το Django είναι ένα framework της Python που φτιάχτηκε για την δημιουργία ιστοσελίδων, και πάνω σε αυτό στηρίζονται σελίδες όπως το Pinterest και το Instagram.
Το Django και η Python μπορούν να τρέξουν σε οποιαδήποτε πλατφόρμα, παρόλο που οι προγραμματιστές πρέπει να σχεδιάσουν τα προγράμματά τους για να τρέχουν και σε Windows και σε Linux.

Ένα πρόβλημα της Python, είναι ότι είναι υπερβολικά δύσκολο να κλιμακωθεί σε πολλούς πυρήνες επεξεργαστή πάνω στο ίδιο μηχάνημα. Αυτό συμβαίνει λόγω των περιορισμών του Global Interpreter Lock(GIL) που χρησιμοποιεί. Παρόλα αυτά, είναι πολύ χρήσιμη σε εφαρμογές οι οποίες κλιμακώνονται οριζοντίως σε διάφορους server, κάνοντας την μια πολύ καλή λύση για εφαρμογές του Cloud.

Χαρακτηριστικά Python:

- Δυναμική
- Επιστημονική
- Εύκολη εκμάθηση
- Καλή για εφαρμογές cloud
- Ελαφριά
- Μέτριο documentation
- Single-Threaded

3.1.4 : Ruby

Η Ruby, και συγκεκριμένα το web framework της, η Ruby on Rails είναι επίσης μια δημοφιλής interpreted γλώσσα για την δημιουργία backend εφαρμογών.
Η Ruby παρέχει περιεκτικούς τρόπους εκμάθησης μέσω διαδικτύου, και για αρχάριους και για έμπειρους προγραμματιστές.

Η μεγάλη ζήτηση της σήμανε την δημιουργία πολλών web-based βιβλιοθηκών και εργαλείων που βοηθούν τους προγραμματιστές να εξελίσσουν γρήγορα εφαρμογές, στην μορφή των RubyGems. Πολλές διάσημες εφαρμογές τρέχουν σε Ruby, όπως το Airbnb, το GitHub και το Groupon.

Ένα σημαντικό πρόβλημα της Ruby είναι ότι δεν κλιμακώνεται πολύ καλά στην πλευρά του server όταν υπάρχει μεγάλος αριθμός αιτήσεων. Επίσης, επειδή η Ruby είναι ανοιχτού λογισμικού και κοινότητας, δεν μπορεί να βρεθεί τόσο καλό documentation εύκολα .
Λόγω αυτού, υπάρχουν ανησυχίες ότι θα είναι πιο δύσκολο να διατηρηθεί η Ruby όσο ο αριθμός των βιβλιοθηκών και των γραμμών του κώδικα μεγαλώνει.

Χαρακτηριστικά Ruby:

- Δυναμική
- Εύκολη
- Εύκολη εκμάθηση
- Υποστήριξη Web
- Ελαφριά
- Όρια κλιμάκωσης

3.1.5 : PHP

Μία από τις πιο δημοφιλείς γλώσσες για εφαρμογές διαδικτύου, η PHP έχει ένα τεράστιο οικοσύστημα από προγραμματιστές, frameworks και βιβλιοθήκες. Μεγάλες εταιρίες, όπως το Facebook, WordPress, Twitpic, Imgur χρησιμοποιούν PHP κατακόρων.
Η PHP δεν έχει κανόνες όπως οι compiled γλώσσες, ή αυστηρά standard όπως η Python, αλλά μόνο guidelines από την προγραμματιστική κοινότητα. Ως αποτέλεσμα, τα μεγάλα project δεν έχουν σωστό στήσιμο και μπορεί να γίνουν δύσκολο να διαβαστούν και να συντηρηθούν, πρόβλημα γνωστό και ως ‘Spaghetti Code’.

Χαρακτηριστικά PHP:

- Δυναμική
- Πολλές βιβλιοθήκες
- Μεγάλη προγραμματιστική κοινότητα
- Μεγάλη ιστορία
- Έλλειψη αυστηρού ελέγχου στον κώδικα

3.1.6 : Node.js

Η Node.js είναι μια νέα event-driven γλώσσα, και έχοντας προέλθει από την Javascript, είναι η μοναδική από τις παραπάνω που έχει σχεδιαστεί από την αρχή να παρέχει web-requests χρησιμοποιώντας την javascript από την πλευρά του server. Το non-blocking input/output της παράγει υψηλή απόδοση σε εφαρμογές server. Οι προγραμματιστές καταλαβαίνουν με τη γνώση τους σε client-side Javascript μπορούν να υλοποιήσουν πλέον και την πλευρά του server. Εταιρίες όπως η Yahoo και η LinkedIn πλέον παράγουν κομμάτια των εφαρμογών τους σε Node.js

Επειδή η Node.Js είναι αρκετά νέα, το να βρεις προγραμματιστές με γνώση σε αυτήν ίσως είναι αρκετά δυσκολότερο. Το documentation και οι βιβλιοθήκες είναι σκορπισμένες. Παρόλα αυτά, η κατάσταση ίσως βελτιωθεί εφόσον οι προγραμματιστές JavaScript μεταβούν προς τις αρχιτεκτονικές backend.

Χαρακτηριστικά Node.JS:

- Δυναμική
- Νέα
- Event-Driven
- Περιορισμένο documentation/βιβλιοθήκες
- Μικρότερη κοινότητα

3.1.7 : Επιλογή και αξιοποίηση της Java στην εφαρμογή

Πίνακας 3.1

Java	Στατική, Πολλές γλώσσες πάνω στο JVM,Καλό documentation Μεγάλο community,Βαριά frameworks, Multi-Threaded, Multi-Platform
.NET	Στατική,Πολλαπλές γλώσσες,Windows OS(Mono στα Linux) Καλό documentation & Υποστήριξη, Multi-Threaded Αδειοδοτημένο μοντέλο λογισμικού
Python	Δυναμική, Επιστημονική, Εύκολη εκμάθηση, Καλή για εφαρμογές cloud Ελαφριά, Μέτριο documentation, Single-Threaded
Ruby	Δυναμική, Εύκολη, Εύκολη εκμάθηση, Υποστήριξη Web Ελαφριά,Όρια κλιμάκωσης
PHP	Δυναμική, Πολλές βιβλιοθήκες, Μεγάλη προγραμματιστική κοινότητα Μεγάλη ιστορία,Έλλειψη αυστηρού ελέγχου στον κώδικα

Node.JS	Δυναμική ,Νέα, Event-Driven, Περιορισμένο documentation/βιβλιοθήκες Μικρότερη κοινότητα

Σύμφωνα με τις απαιτήσεις του project, η καλύτερη (προσωπική) επιλογή ήταν η Java, καθώς παρέχει τεράστια υποστήριξη, υπάρχει ένα τεράστιο community με άπειρες λύσεις στο web, ένα καλοδιατηρημένο documentation και μεγάλη υποστήριξη μέσω framework σε εφαρμογές με πολλούς χρήστες, καθώς και δυνατότητα portability .

3.2 : Java EE

3.2.1 : Ορολογία

Η πλατφόρμα Java, Enterprise Edition (Java EE), πρώην Java 2 Platform, Enterprise Edition (J2EE) και πλέον Jakarta EE, είναι ένα σύνολο προδιαγραφών, επεκτείνοντας το Java SE με προδιαγραφές για επιχειρησιακές δυνατότητες, όπως κατανεμημένες υπολογιστικές υπηρεσίες και υπηρεσίες web. Οι εφαρμογές Java EE μπορούν να είναι microservices ή application servers, οι οποίοι χειρίζονται συναλλαγές, ασφάλεια, κλιμάκωση, ταυτόχρονη διαχείριση και διαχείριση των συστατικών τους..

Η Java EE ορίζεται από τις προδιαγραφές της. Η προδιαγραφή καθορίζει τα API και τις αλληλεπιδράσεις τους. Όπως συμβαίνει και με άλλες προδιαγραφές της Java, όπου οι πάροχοι πρέπει να πληρούν ορισμένες απαιτήσεις συμμόρφωσης για να δηλώσουν τα προϊόντα τους ως Java EE.

Παραδείγματα πλαισίων στα οποία χρησιμοποιούνται η Java EE είναι: ηλεκτρονικό εμπόριο, λογιστικά συστήματα τραπεζικών πληροφοριών.

3.2.2 : Χρήση της Java EE στην υπηρεσία

Η Java EE , όπως περιγράφεται παραπάνω, είναι μόνο ένα specification το όποιο οφείλουν να χρησιμοποιούν οι εκάστοτε εταιρίες/framework που το υλοποιούν. Αυτό σημαίνει ότι για να χρησιμοποιηθεί η Java EE για τον σκοπό της εφαρμογής, υπάρχουν δύο τρόποι.

Ο πρώτος είναι μέσω ενός framework , όπως το Spring, το όποιο μπορεί να εγκατασταθεί πάνω σε έναν οποιοδήποτε ικανό Web Server και να λειτουργήσει για αυτόν τον σκοπό.

Ο δεύτερος είναι, χρησιμοποιώντας κάποιον Java Application Server, ο οποίος υλοποιεί από μόνος του τα Java EE Specifications χωρίς να χρειάζεται ένα βαρύ framework από πίσω να κάνει την δουλεία που έχει ήδη έμφυτη.

3.2.3 : Επιλογή και αξιοποίηση Java EE αντί Spring Framework

Και οι δύο τρόποι ήταν ικανοί να φέρουν εκ πέρας τις απαιτήσεις της υπηρεσίας. Τελικώς, επιλέχτηκε να υλοποιηθεί η εφαρμογή χρησιμοποιώντας έναν Application Server (θα γίνει παρουσίαση παρακάτω) αντί για το Spring Framework.

Αυτό έγινε για τους εξής λόγους:

- Η JEE είναι ένα specification, που σημαίνει ότι μπορεί να τρέξει σε οποιονδήποτε application server, άρα επιτυγχάνεται το portability.
- Παρόλο που χρησιμοποιεί κομμάτια του JEE, γενικά η Spring είναι ένας πλήρης application server. Στο παρελθόν ήταν ελαφρία αλλά αυτό πλέον δεν ισχύει, καθώς το framework πλέον έχει μεγαλώσει πάρα πολύ που δεν μπορεί να μικρύνει. Γενικά παρέχει υπερβολικά πολλά πράγματα τα οποία δεν χρειάζονται στην εφαρμογή που υλοποιείται.
- Είναι υπερβολή να χρησιμοποιηθεί Spring ενώ ενώ η εφαρμογή τρέχει, ήδη πάνω σε έναν application server.
- Οι εμπορικοί application server έχουν πλέον πολύ καλή κονσόλα διαχείρισης. Σε θέμα διαχείρισης της εφαρμογής, είναι πολύ πιο φιλικοί στην διαχείριση από κάποιο τμήμα διαχείρισης.
- Η Spring παρόλο που έχει πάρα πολλές δυνατότητες έμφυτες, χρειάζεται αρκετό configuration(σε XML) για να έρθει στα μέτρα που το χρειαζόμαστε και να εκτελέσει τις λειτουργίες ακριβώς όπως ζητείται.

Το Spring Framework ακόμα υπερέχει σε άλλους τομείς, όπως το έμφυτο πλήρες security, και την υποστήριξη σε Big Data/NoSQL βάσεις, αλλά αυτά δεν ήταν τόσο σημαντικά για την εφαρμογή, όσο το portability, η απουσία μεγάλου configuration της εφαρμογής καθώς και η ευκολία διατήρησης ενός application server.

3.3 : Application Servers

Οι application server(διακομιστές λογισμικού) είναι λογισμικό στο οποίο εκτελούνται εφαρμογές ιστού ή desktop. Οι application servers αποτελούνται από web server connectors, γλώσσες προγραμματισμού υπολογιστών, βιβλιοθήκες χρόνου εκτέλεσης(runtime libraries), συνδέσεις βάσεων δεδομένων και τον κώδικα διαχείρισης που απαιτείται για την ανάπτυξη, τη διαμόρφωση, τη διαχείριση και τη σύνδεση αυτών των στοιχείων σε έναν κεντρικό υπολογιστή Web. Ο διακομιστής εφαρμογών εκτελείται πίσω από έναν κεντρικό υπολογιστή [π.χ. Apache ή Microsoft Internet Information Services (IIS)] και (σχεδόν πάντα) μπροστά σε μια βάση δεδομένων SQL (π.χ. PostgreSQL, MySQL ή Oracle). Οι εφαρμογές Web είναι κώδικες υπολογιστών που εκτελούνται σε application serverskαι είναι γραμμένοι στη γλώσσα που

υποστηρίζει ο διακομιστής εφαρμογών και καλεί τις βιβλιοθήκες και τα συστατικά στοιχεία εκτέλεσης που προσφέρει ο διακομιστής εφαρμογών.

Υπάρχουν πολλοί application servers. Η επιλογή επηρεάζει το κόστος, την απόδοση, την αξιοπιστία, την επεκτασιμότητα και τη δυνατότητα συντήρησης μιας διαδικτυακής εφαρμογής.

Οι application servers παρέχουν υπηρεσίες συστήματος με σαφώς καθορισμένο αλλά ιδιόκτητο τρόπο. Οι προγραμματιστές εφαρμογών αναπτύσσουν προγράμματα σύμφωνα με τις προδιαγραφές του application server.

Μια αντίθετη αλλά ανάλογη περίπτωση είναι η πλατφόρμα Java EE. Οι application servers Java EE παρέχουν υπηρεσίες συστήματος σε ένα καλά καθορισμένο, ανοικτό, βιομηχανικό πρότυπο. Οι προγραμματιστές εφαρμογών αναπτύσσουν προγράμματα σύμφωνα με την προδιαγραφή Java EE και όχι σύμφωνα με τον διακομιστή εφαρμογών. Μια εφαρμογή Java EE που αναπτύχθηκε σύμφωνα με το πρότυπο Java EE μπορεί να εφαρμοστεί σε οποιοδήποτε διακομιστή εφαρμογών Java EE.

3.3.1 : Java application servers

Η πλατφόρμα Java, Enterprise Edition ή Java EE ορίζει το βασικό σύνολο των API και των λειτουργιών των Java Application Servers.

Η υποδομή Java EE χωρίζεται σε logical containers:

- EJB Container: Τα Enterprise JavaBeans (EJB) χρησιμοποιούνται για τη διαχείριση συναλλαγών. Σύμφωνα με τις προδιαγραφές της J2EE, η επιχειρησιακή λογική μιας εφαρμογής βρίσκεται στο Enterprise JavaBeans - ένα αρθρωτό στοιχείο διακομιστή που παρέχει πολλά χαρακτηριστικά, συμπεριλαμβανομένης της διαχείρισης συναλλαγών και της βελτίωσης της επεκτασιμότητας των εφαρμογών.
- Web Container: Τα Web modules περιλαμβάνουν διακομιστές και σελίδες JavaServer (JSP).
- JCA Container (Architecture Connector Java EE)
- Πρόγραμμα JMS (υπηρεσία μηνυμάτων Java)

Ορισμένοι διακομιστές εφαρμογών Java αφήνουν εκτός λειτουργίας πολλές λειτουργίες Java EE όπως το EJB και το Java Message Service (JMS), συμπεριλαμβανομένου του Jetty από το Eclipse Foundation. Η εστίασή τους είναι περισσότερο στα Java Servlets και στις σελίδες JavaServer.

Υπάρχουν πολλοί application servers ανοιχτού κώδικα Java που υποστηρίζουν Java EE, συμπεριλαμβανομένου του JOnAS από το Web Object, το WildFly (πρώην JBoss AS) από το JBoss (τμήμα Red Hat), το Geronimo από το Apache, το TomEE από το Apache, από το Desiderata Software, το Enhydra Server από το Enhydra.org, το GlassFish από το Oracle και το Payara Server από το C2B2.

Οι application servers που αναφέρονται παραπάνω εξυπηρετούν κυρίως εφαρμογές ιστού και υπηρεσίες μέσω RMI, EJB, JMS και SOAP. Παρακάτω θα γίνει σύγκριση κάποιων από αυτούς που αναφέρθηκαν.

3.3.2 : Jboss (WildFly)

Η πλατφόρμα JBoss Enterprise Application Platform (JBoss EAP) της Red Hat είναι η υποστηριζόμενη έκδοση του διακομιστή εφαρμογών Wildfly της κοινότητας τους. Πρόκειται για ένα container Java ™ EE 7 που αποτελεί τμήμα της σειράς προϊόντων JBoss Middleware της Red Hat. Με όλα όσα χρειάζεστε για να δημιουργήσετε και να διαχειριστείτε υπηρεσίες βασισμένες στην Java, είναι ένας καλός συνδυασμός λογισμικού ανοικτού κώδικα και υποστηριζόμενου λογισμικού.

Το Red Hat διαθέτει ένα Πρόγραμμα για Προγραμματιστές όπου μπορείτε να εγγραφείτε για να αποκτήσετε πρόσβαση στο EAP JBoss για αναπτυξιακούς σκοπούς. Όταν είστε μέλος του Προγράμματος, έχετε πρόσβαση στα φόρουμ προγραμματιστών της Red Hat, στη βάση γνώσεων, στο υλικό αναφοράς και στο ενημερωτικό δελτίο των προγραμματιστών.

Οφέλη:

- Υποστηρίζεται από την Red Hat, μια εταιρεία αφιερωμένη σε προϊόντα ανοιχτού κώδικα
- Συμμορφώνεται με τις προδιαγραφές Java ™ EE 7
- Περιλαμβάνει μια σειρά υπηρεσιών web, containers και αρχιτεκτονική έτοιμη για cloud
- Περιλαμβάνει ευέλικτα χαρακτηριστικά διαχείρισης, διαμόρφωσης και αυτοματοποίησης

3.3.3 : Glassfish

Το GlassFish είναι ένα έργο ανοικτού κώδικα που ξεκίνησε από την Sun Microsystems. Η Oracle ανέλαβε την ανάπτυξή της, αλλά δεν παρέχει πλέον υποστήριξη. Ακόμα, η κοινότητα GlassFish παραμένει ισχυρή.

Οφέλη:

- Υποστηρίζει Enterprise JavaBeans, JPA, JavaServer Faces, JMS, RMI, σελίδες JavaServer, μικροεφαρμογές και άλλα
- Οι προγραμματιστές μπορούν να δημιουργήσουν φορητές και κλιμακούμενες επιχειρησιακές εφαρμογές που ενσωματώνουν συστήματα παλαιού τύπου
- Ελαφριά εφαρμογή

3.3.4 : Apache Tomcat / TomEE

Ο Apache Tomcat είναι μια εφαρμογή ανοιχτού κώδικα με πολλές τεχνολογίες Java. Είναι αποτέλεσμα μιας συνεργασίας των κορυφαίων προγραμματιστών παγκοσμίως. Είναι από τους παλαιότερους Java application servers.Η έκδοση TomEE υποστηρίζει το specification της JavaEE

Οφέλη:

- Ελαφρύς κώδικας με αποτέλεσμα γρήγορο χρόνο φόρτωσης και ανάπτυξης
- Ευέλικτος, με πολλές επιλογές προσαρμογής
- Σταθερή πλατφόρμα

3.3.5 : Payara Server

Ο Payara server προέρχεται από το GlassFish. Αυτός ο server έχει βελτιστοποιηθεί για παραγωγή και είναι ασφαλής από προεπιλογή. Ο Payara έχει υλοποιήσει τις δικές του βελτιώσεις και διορθώσεις και δεν έχει καμία σχέση με την Oracle. Υπάρχουν σχέδια για την αντιμετώπιση προηγμένων δυνατοτήτων βάσεων δεδομένων, βελτιωμένων διαγνωστικών και πολλά άλλα.

Ο διακομιστής Payara θα είναι πάντα ανοιχτός κώδικας. Ο κώδικας ανήκει σε μη κερδοσκοπική εταιρεία με έδρα το Ηνωμένο Βασίλειο, η οποία είναι αφιερωμένη στη διασφάλιση της συνεχιζόμενης ανάπτυξης και συντήρησης του διακομιστή προς όφελος της κοινότητας χρηστών.

Οφέλη:

- Ο Payara Server αποτελεί άμεση αντικατάσταση για την έκδοση Open Source του GlassFish Server
- Οι τριμηνιαίες κυκλοφορίες περιέχουν διορθώσεις σφαλμάτων, διορθώσεις και βελτιώσεις
- Οι προσαρμοσμένες κονσόλες μπορούν να δημιουργηθούν με βελτιωμένες δυνατότητες αναφοράς και διαχείρισης
- Το προφίλ Payara GitHub υποστηρίζεται από μια ενεργή κοινότητα χρηστών

3.3.6 : Επιλογή και αξιοποίηση Payara Server στην εφαρμογή

Η συνεχής υποστήριξη των απαιτήσεων, η πλήρης υλοποίηση του JavaEE καθώς και η ευκολία χρήσης ήταν και οι λόγοι, προέτρεψαν την χρήση μιας υλοποίησης του Glassfish.
Ο Payara Server όπως αναφέρθηκε και παραπάνω είναι μια υλοποίηση του Glassfish,που όμως παρέχει παραπάνω δυνατότητες, όπως :

- Συχνα releases με bug fixes,καθώς υπάρχει νέα έκδοση κάθε 3 μήνες
- Υποστήριξη πελατών, είτε για την δημιουργία της εφαρμογής είτε κατά την λειτουργία της
- Μηνιαίο patch για θέματα ασφαλείας
- Προσθήκες όπως SQL Logging, υπηρεσία ελέγχου υγείας της εφαρμογής, καταγραφικό διαχειριστή κλπ
- Υποστήριξη Docker

Για τους παραπάνω λόγους και επιλέχθηκε ως application server της εφαρμογής. Παρόλα αυτά, θα μπορούσε εύκολα η εφαρμογή να τρέξει σε οποιοδήποτε από τους προαναφερθέντες application servers, με κάποιες μικρές μετατροπές.

3.4 : Dependency Injection

3.4.1 : Ορισμός

Στη μηχανική λογισμικού, το dependency injection είναι μια τεχνική όπου ένα αντικείμενο (ή στατική μέθοδος) παρέχει τις εξαρτήσεις ενός άλλου αντικειμένου. Μια εξάρτηση είναι ένα αντικείμενο που μπορεί να χρησιμοποιηθεί .Ένα injection είναι η μετάβαση μιας εξάρτησης σε ένα εξαρτώμενο αντικείμενο που θα το χρησιμοποιήσει.

Αυτή η θεμελιώδης απαίτηση σημαίνει ότι απαγορεύεται η χρήση values που παράγονται μέσα από μια κλάση ή από νέες ή στατικές μεθόδους. Ο client πρέπει να δεχτεί τιμές που πέρασαν από έξω. Αυτό επιτρέπει στον client να αναθέσει αυτό το πρόβλημα κάπου αλλού.

Η πρόθεση πίσω από το dependency injection είναι η αποσύνδεση αντικειμένων στο βαθμό που κανένας κώδικας client δεν πρέπει να αλλάξει απλώς και μόνο επειδή ένα αντικείμενο που εξαρτάται από αυτό πρέπει να αλλάξει σε διαφορετικό.

Το dependency injection είναι μία μορφή της ευρύτερης τεχνικής της αντιστροφής του ελέγχου. Όπως και με άλλες μορφές αντιστροφής του ελέγχου, το dependency injection υποστηρίζει την αρχή της αντιστροφής εξάρτησης. Ο client αναθέτει την ευθύνη της παροχής των εξαρτήσεων του σε εξωτερικό κώδικα (τον injector). Δεν επιτρέπεται στον πελάτη να καλέσει τον εξωτερικό κώδικα που κάνει το inject. Είναι ο κώδικας inject που κατασκευάζει τις υπηρεσίες και καλεί τον πελάτη να τις χρησιμοποιήσει. Αυτό σημαίνει ότι ο κώδικάς του πελάτη δεν χρειάζεται να γνωρίζει τον κώδικα όπου έγινε το inject. Ο client δεν χρειάζεται να γνωρίζει πώς να κατασκευάσει τις υπηρεσίες. Ο client δεν χρειάζεται να γνωρίζει ποιες υπηρεσίες χρησιμοποιεί. Ο client πρέπει να γνωρίζει μόνο τις εγγενείς διεπαφές των υπηρεσιών, επειδή αυτές καθορίζουν τον τρόπο με τον οποίο ο client μπορεί να χρησιμοποιήσει τις υπηρεσίες. Αυτό διαχωρίζει τις ευθύνες χρήσης και κατασκευής.

3.4.2 : EJB 3

Το Enterprise JavaBeans (EJB) είναι μια πλατφόρμα για την κατασκευή φορητών, επαναχρησιμοποιήσιμων και κλιμακούμενων επιχειρηματικών εφαρμογών χρησιμοποιώντας τη γλώσσα προγραμματισμού Java. Από την αρχική ενσωμάτωσή της, το EJB έχει προσφερθεί ως μοντέλο ή πλαίσιο που επιτρέπει να δημιουργηθούν εφαρμογές Java για επιχειρήσεις χωρίς να χρειάζεται να επανακατασκευαστούν υπηρεσίες όπως συναλλαγές, ασφάλεια και αυτοματοποιημένη επιμονή για την κατασκευή μιας εφαρμογής.

Το EJB 3 απλοποιεί σημαντικά την ανάπτυξη υιοθετώντας ένα μοντέλο προγραμματισμού POJO. Όπως φαίνεται στο παρακάτω σχήμα, ένας σχολιασμός μετατρέπει ένα απλό POJO σε ένα EJB.

Ένας σχολιασμός μετατρέπει ένα απλό POJO σε ένα EJB Με το EJB 3, το dependency injection έχει απλοποιήσει σημαντικά την πρόσβαση σε πόρους EJB όπως JDBC DataSource, JMS Objects και JPA Entity Manager και υπηρεσίες όπως Timer, User Transaction και Web Services.

Το dependency injection μας επιτρέπει να δηλώσουμε απλώς τις εξαρτήσεις και να αφήσουμε το container EJB να ασχοληθεί με την πολυπλοκότητα της δημιουργίας στιγμιότυπων, την προετοιμασία και την ανάλυση των πόρων και την παροχή αναφορών για υπηρεσίες ή πηγές στους client, όπως απαιτείται.

Παρακάτω (Πίνακας)υπάρχουν κάποιοι τύποι κλάσεων που μπορούν να γίνουν inject σε αντικείμενα.

Πίνακας 3.2

RESOURCES	STATELESS	STATEFUL	MDB	INTERCEPTORS
JDBC DataSource	NAI	NAI	NAI	NAI
JMS Destinations, Connection Factories	NAI	NAI	NAI	NAI
Mail Resources	NAI	NAI	NAI	NAI
UserTransaction	NAI	NAI	NAI	NAI
Environment Entries	NAI	NAI	NAI	NAI
EJBContext	NAI	NAI	NAI	OXI
Timer Service	NAI	OXI	NAI	OXI
Web Service reference	NAI	NAI	NAI	NAI
EntityManager, EntityManagerFactory	NAI	NAI	NAI	NAI

Όπου:

- Stateless : Ένα αντικείμενο το όποιο δεν μπορεί να κρατήσει values στις μεταβλητές του
- Stateful: Ένα αντικείμενο το οποίο κρατάει values στις μεταβλητές του
- Mdb : Ένα message-driven bean που επιτρέπει στις εφαρμογές να επικοινωνούν ασύγχρονα
- Interceptors : Κλάσεις, που καλούνται πριν από άλλες όταν υπάρχει το annotation

Τα παρακάτω annotations(Πίνακας 3.3) μπορούν να χρησιμοποιηθούν σε ένα πεδίο ή μια μέθοδο για dependency injection. Επίσης, μπορούν να χρησιμοποιηθούν σε επίπεδο κλάσης για να ορίσουν τα dependencies που θα χρησιμοποιηθούν αργότερα με το JNDI lookup.

Πίνακας 3.3

ANNOTATIONS	ΧΡΗΣΗ	COMPONENTS ΠΟΥ ΜΠΟΡΟΥΝ ΝΑ ΤΑ ΧΡΗΣΙΜΟΠΟΙΗΣΟΥΝ
javax.annotation.Resource	Dependency injection πόρων όπως DataSource, και αντικείμενα JMS	EJB, Web, Application Client
javax.ejb.EJB	Dependency injection σε Session beans	EJB, Web, Application Client
javax.xml.ws.WebServiceRef	Dependency injection σε Web services	EJB, Web, Application Client
javax.persistence.PersistenceContext	Dependency injection σε container-managed EntityManager	EJB, Web
javax.persistence.PersistenceUnit	Dependency injection σε EntityManagerFactory	EJB, Web

3.4.3 : Αξιοποίηση EJB στην εφαρμογή

Η χρήση EJB σε Java Web Applications βοηθάει τον προγραμματιστή, ώστε να μην ασχολείται με θέματα optimization, πόρων , multithreading κλπ καθώς αυτά υλοποιούνται σύμφωνα με το πρότυπο λειτουργίας της Java EE. Με την χρήση των beans αφήνουμε ένα μεγάλο κομμάτι υλοποίησης και ασφάλειας σε αυτά

Στην εφαρμογή, χρησιμοποιούνται κυρίως στις κλάσεις που υλοποιούν την λογική της υπηρεσίας ώστε να επιτυγχάνεται η καλύτερη διαχείριση πόρων, σε κλάσεις που πραγματοποιούν χρονοπρογραμματισμένες εργασίες όπως καταγραφή και ανάκτηση δεδομένων σε ορισμένα χρονικά διαστήματα , στις κλάσεις διασύνδεσης με βάσεις δεδομένων μέσω αντικειμένων DataSource, καθώς και για την ασφάλεια της εφαρμογής και των υπηρεσιών που παρέχει.

3.5 : Java Build Tools

Προκειμένου να φέρουμε τον κώδικα μας στην παραγωγή, θα πρέπει να έχουμε μια εύχρηστη εικόνα λογισμικού που μπορεί να εκτελεστεί και να αναπτυχθεί. Εκεί μπαίνουν τα εργαλεία κατασκευής. Λαμβάνουν τον πηγαίο κώδικα μας και το καταρτίζουν σε ένα εκτελέσιμο πρόγραμμα. Αυτές τις μέρες, τα εργαλεία κατασκευής χρησιμοποιούνται όχι μόνο για την κατασκευή των εφαρμογών μας, αλλά και με άλλες δυνατότητες όπως διαχείριση εξαρτήσεων(dependency management)

Στον κόσμο της Java, υπάρχουν τρία βασικά εργαλεία κατασκευής: Ant, Maven και Gradle. Αυτά τα εργαλεία έρχονται πριν από το πραγματικό περιβάλλον παραγωγής, αλλά αποτελούν απαραίτητο στοιχείο της διαδικασίας, για αυτό και αναφέρονται εδώ. Οι βασικές διαφορές για αυτά τα εργαλεία

μειώνονται στο μέγεθος της προσπάθειας, της χρηστικότητας και των εξωτερικών συνδέσεων που έχει το καθένα.

3.5.1 : Ant

Το Ant της Apache είναι μια βιβλιοθήκη Java open source και ένα εργαλείο γραμμής εντολών που χρησιμοποιείται για την αυτοματοποίηση διαδικασιών δημιουργίας λογισμικού. Χρησιμοποιείται κυρίως για την κατασκευή εφαρμογών Java. Δημιουργήθηκε το 2000, και είναι το αρχικό εργαλείο κατασκευής στον χώρο της Java που χρησιμοποιείται ακόμα και σήμερα.

Πλεονεκτήματα

- Χρησιμοποιεί βάση XML που σημαίνει ότι λειτουργεί καλά με αυτόματα εργαλεία.
- Μόλις ανοίξει και τρέξει, ο Ant μας δίνει σχεδόν πλήρη έλεγχο
- Το εμπλουτισμένο οικοσύστημα των plugin δημιουργεί πολλές δυνατότητες και είναι εύκολο να δημιουργήσουμε προσαρμοσμένες προσθήκες, αν αυτό που χρειαζόμαστε δεν είναι διαθέσιμο.
- Στερεά και εκτεταμένο documentation

Μειονεκτήματα:

- Η Βάση XML που χρησιμοποιεί σημαίνει λιγότερες δυνατότητες προσαρμογής.
- Η δημιουργία script είναι συχνά πολύ διαφορετική, γεγονός που καθιστά δύσκολη την κατανόηση άλλων project.
- Ως παλαιό καθιερωμένο εργαλείο, η κοινότητα είναι αρκετά νεκρή.

3.5.2 : Maven

Το Apache Maven είναι ένα εργαλείο αυτοματοποίησης κατασκευής, κυρίως για Java, και είναι η πιο δημοφιλής επιλογή για τους προγραμματιστές της Java σήμερα σύμφωνα με τους αριθμούς χρήσης. Είναι ιδανικό για μεγάλες επιχειρήσεις λόγω της πολύ γρήγορης ταχύτητας κατασκευής.

Πλεονεκτήματα:

- Εκτεταμένο οικοσύστημα για plugins.

- Η κοινή δομή μεταξύ των build καθιστά την κατανόηση άλλων project εύκολη.
- Πλήρης υποστήριξη για σχεδόν οποιοδήποτε CI, application server ή εργαλείο IDE.

Μειονεκτήματα:

- Πολλές απαιτήσεις λήψης για εξαρτήσεις και plugins..
- Η προσαρμογή στις απαιτήσεις είναι αδύναμη.

3.5.3 : Gradle

Το Gradle είναι ένα σύστημα αυτοματισμού ανοιχτού κώδικα. Το Gradle στοχεύει να «συνδυάσει τη δύναμη και την ευελιξία του Ant με τη διαχείριση εξάρτησης και τις συμβάσεις του Maven σε έναν πιο αποτελεσματικό τρόπο οικοδόμησης». Τα script build του είναι γραμμένα σε Groovy και όχι σε XML, διαφορετικών πλεονεκτημάτων και μειονεκτημάτων σε σύγκριση με τον Ant ή Maven. Παρά το γεγονός ότι είναι ένα νεότερο εργαλείο σε αυτό το χώρο, θεωρείται εκτεταμένη υιοθεσία.

Πλεονεκτήματα:

- Χρησιμοποιεί βάση DSL σημαίνει ότι έχετε ένα πιο προσαρμόσιμο και εξορθολογισμένο εργαλείο.
- Εξαιρετική τεκμηρίωση και ενεργή κοινότητα.
- Είναι απλό να δημιουργηθούν προσαρμοσμένες προσθήκες.

Μειονεκτήματα:

- Η Βάση DSL σημαίνει ότι υπάρχει ένα λιγότερο απλό και τυποποιημένο εργαλείο.
- Ως νεότερη εργαλείο, το οικοσύστημα για plugins είναι λιγότερο ανεπτυγμένο.
- Ως νεότερο εργαλείο, η υποστήριξή του για εργαλεία CI και application servers δεν είναι τόσο ολοκληρωμένη σε σύγκριση Maven ή Ant.

3.5.4 : Επιλογή και αξιοποίηση Maven στην εφαρμογή

Ant vs Maven/Gradle

- Χρησιμοποιώντας ANT πρέπει να παρέχονται πληροφορίες σχετικά με τη δομή του έργου, ενώ με το Maven/Gradle υπάρχει μια σύμβαση για την τοποθέτηση πηγαίου κώδικα, σύνθετου κώδικα, πακέτων κλπ. Έτσι, δεν χρειάζεται να παρέχονται πληροφορίες για το έργο.
- Το Maven/Gradle έχει έναν κύκλο ζωής της διαδικασίας κατασκευής, ενώ η ANT δεν έχει.

- Το Maven/Gradle είναι ένα πλαίσιο και το Ant είναι απλά ένα εργαλείο.
- Το Ant είναι κυρίως εργαλείο κατασκευής ενώ τα άλλα δυο είναι κυρίως εργαλεία διαχείρισης έργου.

Maven vs Gradle

- Customized Builds. Με το Maven, μπορούν να οριστούν εύκολα τα metadata και οι εξαρτήσεις της εφαρμογής, αλλά η δημιουργία ενός πολύ προσαρμοσμένου build μπορεί να είναι ένας εφιάλτης. Το αρχείο POM μπορεί εύκολα να μεγαλώσει πολύ καθώς το project μεγαλώνει και ίσως είναι αργότερα ένα μη αναγνώσιμο αρχείο XML.
- Dependency management και δομή project. Παρόλα αυτά, το Maven παρέχει απλή αλλά αποτελεσματική διαχείριση εξάρτησης και αφού έχει μια δομή καταλόγου για τα project μας, έχουμε μια τυποποιημένη διάταξη για όλα τα project μας. Χρησιμοποιεί ένα δηλωτικό αρχείο XML για το αρχείο POM του και έχει μια σειρά από plugins που μπορούν να χρησιμοποιηθούν. Το Gradle χρησιμοποιεί τη δομή καταλόγου που υπάρχει στο Maven, αλλά αυτό μπορεί να προσαρμοστεί. Χρησιμοποιεί επίσης την ίδια μορφή GAV που χρησιμοποιεί το Maven για τον εντοπισμό αντικειμένων.
- Plugins και ενσωματώσεις. Το Maven υποστηρίζει επίσης μια ευρεία ποικιλία βημάτων οικοδόμησης του κύκλου ζωής και ενσωματώνεται απρόσκοπτα με εργαλεία τρίτων κατασκευαστών όπως διακομιστές CI, plugins για κάλυψη κώδικα και συστήματα αποθήκευσης τεχνασμάτων. Όσον αφορά τα plugin, υπάρχει όλο και μεγαλύτερος αριθμός διαθέσιμων plugin πλέον και υπάρχουν μεγάλοι προμηθευτές που έχουν plugins συμβατά με το Gradle. Ωστόσο, υπάρχουν ακόμα περισσότερα διαθέσιμα πρόσθετα για το Maven σε σύγκριση με τον αριθμό που διατίθεται για το Gradle.
- Ευκαμψία. Το Gradle, από την άλλη πλευρά, είναι πολύ ευέλικτο και βασίζεται σε ένα script. Τα custom builds θα ήταν εύκολο να γίνουν στο Gradle. Ωστόσο, επειδή ο Gradle είναι ουσιαστικά νεότερο, ο αριθμός των προγραμματιστών που γνωρίζουν το Gradle καλά να είναι περιορισμένος.

Το Maven προτιμήθηκε τελικά έναντι του Gradle, καθώς είναι μια πιο έτοιμη και διαδεδομένη τεχνολογία, που σημαίνει ότι υπάρχει μεγαλύτερη υποστήριξη σε διάφορες τεχνολογίες που χρησιμοποιούνται στην εφαρμογή.

3.6 : Βάση Δεδομένων

3.6.1 : Ορισμός

Μια βάση δεδομένων είναι μια οργανωμένη συλλογή δεδομένων. Μια σχεσιακή βάση δεδομένων, είναι μια συλλογή σχημάτων, πινάκων, ερωτημάτων, αναφορών, προβολών και άλλων στοιχείων. Οι σχεδιαστές βάσεων δεδομένων συνήθως οργανώνουν τα δεδομένα για να μοντελοποιούν πτυχές της πραγματικότητας κατά τρόπο που να υποστηρίζει διαδικασίες που απαιτούν πληροφορίες, όπως (για παράδειγμα) μοντελοποίηση της διαθεσιμότητας δωματίων σε ξενοδοχεία με τρόπο που να υποστηρίζει την εύρεση ενός ξενοδοχείου με κενές θέσεις.

Ένα σύστημα διαχείρισης βάσεων δεδομένων (DBMS) είναι μια εφαρμογή υπολογιστή-λογισμικού που αλληλοεπιδρά με τους τελικούς χρήστες, άλλες εφαρμογές και την ίδια τη βάση δεδομένων για την καταγραφή και ανάλυση δεδομένων. Ένα DBMS γενικού σκοπού επιτρέπει τον ορισμό, τη δημιουργία, την αναζήτηση, την ενημέρωση και τη διαχείριση των βάσεων δεδομένων.

Μία βάση δεδομένων δεν είναι γενικά φορητή σε διαφορετικά DBMS, αλλά διαφορετικά DBMS μπορούν να διαλειτουργούν χρησιμοποιώντας πρότυπα όπως SQL και ODBC ή JDBC για να επιτρέψουν σε μια ενιαία εφαρμογή να λειτουργήσει με περισσότερα από ένα DBMS. Οι μηχανικοί υπολογιστών μπορούν να ταξινομούν τα συστήματα διαχείρισης βάσεων δεδομένων σύμφωνα με τα μοντέλα βάσης δεδομένων που υποστηρίζουν. Μερικές φορές ένα DBMS αναφέρεται χαλαρά ως "βάση δεδομένων".

3.6.2 : MySQL

Η MySQL είναι μια βάση δεδομένων SQL κατάλληλη για μικρές έως μεσαίες ιστοσελίδες. Η βάση δεδομένων είναι ελεύθερη και ανοιχτή με μια εμπορική άδεια (η MySQL ανήκει πλέον στην Oracle αφού αγόρασε τον Sun).

Οι κοινές εφαρμογές για MySQL περιλαμβάνουν εφαρμογές web που βασίζονται σε php και java που απαιτούν back-end αποθήκευσης DB.

Η MySQL χρησιμοποιείται συνήθως με 2 διαφορετικούς μηχανισμούς αποθήκευσης. Ο ένας ονομάζεται MyISAM δεν υποστηρίζει συναλλαγές και αποθηκεύει κάθε πίνακα σε ένα σύνολο 3 αρχείων. Το δεύτερο λέγεται InnoDB το οποίο υποστηρίζει τις συναλλαγές. Αυτός ο μηχανισμός αποθήκευσης αποθηκεύει όλα τα δεδομένα σε ένα ενιαίο σύνολο από bytes ή χρησιμοποιεί ένα σύνολο bytes ανά κατάλογο βάσεων δεδομένων.

Η MySQL έχει ένα μεγάλο πλεονέκτημα, καθώς είναι δωρεάν, είναι συνήθως διαθέσιμο σε κοινό πακέτα φιλοξενίας και μπορεί εύκολα να εγκατασταθεί σε περιβάλλον Linux, Unix ή Windows. Εάν μια εφαρμογή web απαιτεί περισσότερο από τη βάση δεδομένων,, είναι εύκολο να ρυθμίσουμε ίσως στιγμιότυπα της

βάσης δεδομένων που απαιτούν μόνο το κόστος υλικού, σε αντίθεση με τις εμπορικές βάσεις δεδομένων που απαιτούν μία άδεια για κάθε περίπτωση.

Πλεονεκτήματα:

- Είναι εύκολη στη χρήση: Ενώ απαιτείται βασική γνώση της SQL-και οι περισσότερες σχεσιακές βάσεις δεδομένων απαιτούν την ίδια γνώση, η MySQL είναι πολύ εύκολη στη χρήση. Με μόνο μερικές απλές εντολές SQL, μπορείτε να δημιουργήσουμε και να αλληλοεπιδράσουμε με τη MySQL.
- Είναι ασφαλής: η MySQL περιλαμβάνει σταθερά επίπεδα ασφαλείας δεδομένων που προστατεύουν τα ευαίσθητα δεδομένα από εισβολείς. Τα δικαιώματα μπορούν να ρυθμιστούν ώστε να επιτρέπουν ορισμένα ή όλα τα προνόμια στους clients. Οι κωδικοί πρόσβασης είναι κρυπτογραφημένοι.
- Είναι δωρεάν: η MySQL διατίθεται δωρεάν από την τοποθεσία Web της MySQL.
- Είναι γρήγορη: Για λόγους ταχύτητας, οι σχεδιαστές της MySQL αποφάσισαν να προσφέρουν λιγότερα χαρακτηριστικά από άλλους σημαντικούς ανταγωνιστές βάσης δεδομένων, όπως η Oracle . Ωστόσο, παρά το γεγονός ότι διαθέτει λιγότερα χαρακτηριστικά από τα άλλα προϊόντα εμπορικής βάσης δεδομένων, η MySQL εξακολουθεί να προσφέρει όλα τα χαρακτηριστικά που απαιτούνται από τους περισσότερους προγραμματιστές βάσεων δεδομένων.
- Είναι επεκτάσιμη: η MySQL μπορεί να χειριστεί σχεδόν κάθε ποσότητα δεδομένων, μέχρι και 50 εκατομμύρια εγγραφές ή περισσότερες. Το προεπιλεγμένο όριο μεγέθους αρχείου είναι περίπου 4 GB. Ωστόσο, μπορούμε να αυξήσουμε αυτόν τον αριθμό σε ένα θεωρητικό όριο 8 TB δεδομένων.
- Διαχειρίζεται πολύ καλά τη μνήμη: ο MySQL server έχει ελεγχθεί διεξοδικά για την αποφυγή διαρροών μνήμης.
- Λειτουργεί σε πολλά λειτουργικά συστήματα: η MySQL τρέχει σε πολλά λειτουργικά συστήματα, όπως τα Windows ,Linux , πολλές ποικιλίες UNIX (όπως Sun , Solaris , AIX και DEC * UNIX), OS / 2, FreeBSD , και άλλα.
- Υποστηρίζει αρκετές διεπαφές ανάπτυξης: Οι διεπαφές ανάπτυξης περιλαμβάνουν JDBC, ODBC και δέσμες ενεργειών (PHP και Perl), επιτρέποντάς σας να δημιουργήσετε λύσεις βάσεων δεδομένων που εκτελούνται σε όλες τις μεγάλες πλατφόρμες, όπως Linux, UNIX και Windows.

3.6.3 : Αξιοποιήση της MySQL στο project

Για τις ανάγκες της εφαρμογής χρειάζεται η χρησιμοποίηση μιας τοπικής βάσης δεδομένων. Αυτή η βάση, χρησιμοποιείται για 3 σκοπούς:

- Καταγραφή συναλλαγών με GUNET Aggregator: Κάθε φορά που στέλνεται ένα SMS μέσω της υπηρεσίας, η εφαρμογή λαμβάνει πίσω ένα DLR(Delivery Report) από την πλατφόρμα του GUNET. Όποτε υπάρχει η ανάγκη καταγραφής αυτών των DLR για στατιστικούς και λειτουργικούς λόγους (π.χ. να στέλνεται ξανά μήνυμα σε έναν αριθμό στον όποιο παρουσιάζεται σφάλμα κάθε φορά κατά την αποστολή).
- Καταγραφή ενεργειών και αποστολής μηνυμάτων: Καταγράφει τις ενέργειες των χρηστών της ιστοσελίδας, ώστε να υπάρχει επίβλεψη των μηνυμάτων που στέλνονται είτε από καθηγητές είτε μαθητές.
- Βοηθητική writable βάση: Όπως εξηγείται και στην αρχιτεκτονική της υπηρεσίας, δεν υπάρχει πρόσβαση εγγραφής στις βάσεις του ακαδημαϊκού ιδρύματος. Για τις ανάγκες της εφαρμογής όμως, πρέπει να αποθηκεύονται κάποια δεδομένα για τους χρήστες (όπως ανανεωμένο τηλέφωνο, ή προτιμήσεις για την αποστολή μηνυμάτων).

Έχοντας αυτές τις απαιτήσεις, γίνεται σαφές ότι δεν χρειάζεται κάποια βάση με πολλές λειτουργιές. Σε συνδυασμό με την τεράστια υποστήριξη που υπάρχει για την MySQL (ειδικά στην διασύνδεση με Java) και το γεγονός ότι είναι μια εύκολη, δωρεάν, open source λύση, που θα στηθεί σε ένα μηχάνημα Linux, πάρθηκε η απόφαση να χρησιμοποιηθεί ως βάση για το project.

3.6.4 : JDBC Connection Pooling

Η δημιουργία συνδέσεων JDBC είναι δαπανηρή σε πόρους, ιδιαίτερα όταν το JDBC API που χρησιμοποιείται σε περιβάλλον server που συνδέει components. Σε αυτόν τον τύπο περιβάλλοντος, η απόδοση μπορεί να βελτιωθεί σημαντικά όταν χρησιμοποιείται το Connection Pooling. Η συγκέντρωση συνδέσεων σημαίνει ότι οι συνδέσεις επαναχρησιμοποιούνται αντί να δημιουργούνται κάθε φορά που ζητάμε μια σύνδεση. Για να διευκολυνθεί η επαναχρησιμοποίηση της σύνδεσης, χρειάζεται μια μνήμη cache των συνδέσεων βάσης δεδομένων, που ονομάζεται connection pool, συντηρείται από μια ενότητα συγκέντρωσης σύνδεσης ως ένα στρώμα πάνω από οποιοδήποτε JDBC driver.

Η συγκέντρωση συνδέσεων πραγματοποιείται στο παρασκήνιο και δεν επηρεάζει τον κώδικα μιας εφαρμογής. Ωστόσο, η εφαρμογή πρέπει να χρησιμοποιεί ένα αντικείμενο DataSource (ένα αντικείμενο που υλοποιεί τη διασύνδεση DataSource) για να αποκτήσει μια σύνδεση αντί να χρησιμοποιήσει την κλάση του DriverManager. Μια κλάση που υλοποιεί τη διασύνδεση DataSource μπορεί να παρέχει ή όχι συγκέντρωση συνδέσεων. Ένα αντικείμενο DataSource καταχωρείται με μια υπηρεσία ονομασίας JNDI(Java Naming and Directory Interface). Μόλις καταχωρηθεί ένα αντικείμενο DataSource, η εφαρμογή ανακτά την από την υπηρεσία ονομασίας JNDI με τον συνήθη τρόπο.

Για παράδειγμα, χρησιμοποιώντας τον κώδικα παρακάτω (Εικόνα 3.1) μπορεί να αποκτηθεί άμεσα μία σύνδεση στην βάση δεδομένων.

Εικόνα 3.1

```
@Resource(lookup = "jdbc/pithia")
DataSource pithia;
```

Εάν το αντικείμενο DataSource παρέχει connection pooling, η αναζήτηση επιστρέφει μια σύνδεση, αν είναι διαθέσιμη, αλλιώς αν δεν βρει καμία διαθέσιμη, δημιουργεί μια νέα σύνδεση. Η εφαρμογή επωφελείται από την επαναχρησιμοποίηση της σύνδεσης χωρίς να απαιτείται καμία αλλαγή κώδικα. Οι επαναχρησιμοποιούμενες συνδέσεις από το pool συμπεριφέρονται με τον ίδιο τρόπο όπως οι νεοσύστατες φυσικές συνδέσεις. Η εφαρμογή κάνει σύνδεση με τη βάση δεδομένων και η πρόσβαση στα δεδομένα λειτουργεί με τον συνήθη τρόπο. Όταν ολοκληρωθεί η εργασία με τη σύνδεση, η εφαρμογή κλείνει τη σύνδεση και αυτή επιστρέφεται στο pool (Εικόνα 3.1).

Εικόνα 3.2

```
Connection conn = pithia.getConnection();
conn.close();
```

Οι JNDI συνδέσεις της εφαρμογής μας δημιουργούνται και ανακτώνται από τον Application server, που στην περίπτωσή μας είναι ο Payara Server. Αυτό μπορεί να γίνει είτε μέσω του admin panel του (περιγράφεται στο documentation της εφαρμογής) είτε μέσω του domain.xml του application, και μπορούν να ρυθμιστούν καταλλήλως με πολύ παραπάνω επιλογές για την σύνδεση από μια απλή ανάκτηση σύνδεσης από την DriverManager class.

3.7 : Web Services

Τα Web Services είναι εφαρμογές client-server που επικοινωνούν μέσω του HTTP (Hypertext Transfer Protocol) . Όπως περιγράφεται από την Κοινοπραξία World Wide Web (W3C), οι υπηρεσίες web παρέχουν ένα τυπικό μέσο για τη διαλειτουργικότητα μεταξύ εφαρμογών λογισμικού που λειτουργούν σε διάφορες πλατφόρμες και πλαίσια. Οι υπηρεσίες Web χαρακτηρίζονται από τη μεγάλη διαλειτουργικότητα και την ευελιξία τους, καθώς και τις επεξεργάσιμες από μηχανές περιγραφές τους, χάρη στην χρήση των XML. Οι υπηρεσίες Web μπορούν να συνδυαστούν με έναν χαλαρά συνδεδεμένο τρόπο για την επίτευξη σύνθετων λειτουργιών. Τα προγράμματα που παρέχουν απλές υπηρεσίες μπορούν να αλληλοεπιδρούν μεταξύ τους για την παροχή περίπλοκων υπηρεσιών προστιθέμενης αξίας.

Στο εννοιολογικό επίπεδο, ένα Web Service είναι ένα στοιχείο λογισμικού που παρέχεται μέσω ενός τελικού σημείου(endpoint) προσβάσιμου στο δίκτυο. Ο καταναλωτής και ο πάροχος της υπηρεσίας χρησιμοποιούν μηνύματα για την ανταλλαγή αιτημάτων και απάντησης επίκλησης με τη μορφή αυτοτελών εγγράφων.

Σε τεχνικό επίπεδο, οι υπηρεσίες web μπορούν να υλοποιηθούν με διάφορους τρόπους. Οι δύο τεχνολογίες ιστού που χρησιμοποιούνται για την υλοποίησή τους είναι η SOAP και η REST.

3.7.1 : REST

Το REST (Representational State Transfer) είναι πραγματικά ένα Web Services API. Τα REST API βασίζονται σε URI (Uniform Resource Identifier) και το πρωτόκολλο HTTP και χρησιμοποιούν την μορφή JSON για μεταφορά δεδομένων, η οποία είναι συμβατή με όλους τους browser. Ένα REST API μπορεί να είναι απλό να κατασκευαστεί και να κλιμακωθεί , αλλά μπορούν επίσης να είναι τεράστια και περίπλοκα - είναι όλα σχετικά με τον τρόπο κατασκευής τους, και τον σκοπό για τον οποίο έχουν σχεδιαστεί.

Λόγοι για τους οποίους μπορεί να θέλουμε να δημιουργηθεί ένα API που να είναι RESTful περιλαμβάνουν περιορισμούς πόρων, λιγότερες απαιτήσεις ασφαλείας, συμβατότητα προγράμματος-πελάτη προγράμματος περιήγησης, υγεία δεδομένων και κλιμάκωση - πράγματα που ισχύουν πραγματικά για τις υπηρεσίες ιστού.

Μερικές πληροφορίες για το REST:

- Το REST είναι απλό, χάρη στο πρωτόκολλο HTTP.
- Το REST API χρησιμοποιεί μια ενιαία ομοιόμορφη διεπαφή. Αυτό απλοποιεί τον τρόπο με τον οποίο οι εφαρμογές αλληλοεπιδρούν με το API απαιτώντας όλες τις διεπαφές με τον ίδιο τρόπο, μέσω της ίδιας πύλης.
- Το REST βελτιστοποιείται για τον ιστό. Χρησιμοποιώντας το JSON ως μορφή δεδομένων, το καθιστά συμβατό με τα προγράμματα περιήγησης.
- Το REST είναι γνωστό για άριστες επιδόσεις και δυνατότητα κλιμάκωσης. Αλλά, όπως και κάθε τεχνολογία, μπορεί να μπλοκαριστεί ή να χτυπήσει την εφαρμογή μας. Αυτός είναι ο λόγος για τον οποίο έχουν κατασκευαστεί γλώσσες όπως το GraphQL για την αντιμετώπιση προβλημάτων ακόμη και το REST δεν μπορεί να λύσει.

3.7.2 : SOAP

Το πρωτόκολλο SOAP (Simple Access Protocol) είναι ένα πρωτόκολλο από μόνο του και είναι λίγο πιο περίπλοκο, καθορίζοντας περισσότερα πρότυπα από τα REST - πράγματα όπως η ασφάλεια και ο τρόπος με τον οποίο αποστέλλονται τα μηνύματα. Αυτά τα ενσωματωμένα πρότυπα φέρουν λίγο περισσότερη επιβάρυνση, αλλά μπορούν να αποτελέσουν αποφασιστικό παράγοντα για οργανισμούς που απαιτούν πιο ολοκληρωμένα χαρακτηριστικά όσον αφορά την ασφάλεια, τις συναλλαγές και την συμμόρφωση με το ACID (Atomicity, Consistency, Isolation, Durability).

Λόγοι για τους οποίους μπορεί να θέλουμε να δημιουργηθεί μια εφαρμογή με SOAP API περιλαμβάνουν υψηλότερα επίπεδα ασφάλειας (π.χ. διεπαφή κινητής εφαρμογής με μια τράπεζα), εφαρμογές ανταλλαγής μηνυμάτων που χρειάζονται αξιόπιστη επικοινωνία ή συμμόρφωση με το ACID.

Μερικές πληροφορίες για το SOAP API:

- Το SOAP έχει αυστηρότερη ασφάλεια. Το WS-Security, πέραν της υποστήριξης SSL, είναι ένα ενσωματωμένο πρότυπο που δίνει στο SOAP κάποιες δυνατότητες ασφάλειας σε επίπεδο επιχείρησης.
- Επανάληψη αποστολής για αξιόπιστη λειτουργία μηνυμάτων. Το REST δεν διαθέτει ένα τυπικό σύστημα ανταλλαγής μηνυμάτων και μπορεί να αντιμετωπίσει τις αποτυχίες επικοινωνίας μόνο με μια νέα προσπάθεια. Το SOAP έχει ενσωματωμένη λογική επιτυχίας / επανάληψης και παρέχει αξιοπιστία από άκρο σε άκρο ακόμη και μέσω των διαμεσολαβητών SOAP.
- Το SOAP έχει ενσωματωμένη συμμόρφωση με το ACID. Η συμμόρφωση με το ACID μειώνει τις ανωμαλίες και προστατεύει την ακεραιότητα μιας βάσης δεδομένων, καθορίζοντας ακριβώς τον τρόπο αλληλεπίδρασης των συναλλαγών με τη βάση δεδομένων. Το ACID είναι πιο συντηρητικό από άλλα μοντέλα συνέπειας των δεδομένων, για αυτό προτιμάται συνήθως όταν χειρίζεται οικονομικές ή άλλως ευαίσθητες συναλλαγές.

3.7.3 : Επιλογή και αξιοποίηση REST στην εφαρμογή

- Το SOAP είναι ένα πρωτόκολλο. Το REST είναι αρχιτεκτονικό στυλ. Κάθε API έχει σχεδιάζεται για να εκθέτει ορισμένες πτυχές της επιχειρηματικής λογικής μιας εφαρμογής σε ένα server και το SOAP χρησιμοποιεί μια διεπαφή υπηρεσίας για να το κάνει αυτό, ενώ το REST χρησιμοποιεί URIs.
- Τα REST API έχουν πρόσβαση σε έναν πόρο για δεδομένα (URI). Τα API SOAP εκτελούν μια ενέργεια. Το REST είναι μια αρχιτεκτονική που έχει περισσότερη ροή δεδομένων. Το SOAP είναι ένα τυποποιημένο πρωτόκολλο για τη μεταφορά δομημένων πληροφοριών που είναι πιο λειτουργικές.
- Το REST επιτρέπει πολλές διαφορετικές μορφές δεδομένων, όπως το Plain Text, το HTML, το XML και το JSON, το οποίο είναι εξαιρετικά κατάλληλο για δεδομένα και παρέχει μεγαλύτερη συμβατότητα με το πρόγραμμα περιήγησης. Το SOAP χρησιμοποιεί μόνο XML.
- Η ασφάλεια χειρίζεται διαφορετικά. Το SOAP υποστηρίζει το WS-Security, το οποίο είναι εξαιρετικό στο επίπεδο των μεταφορών και είναι λίγο πιο ολοκληρωμένο από το SSL και πιο ιδανικό για την ενσωμάτωση σε εργαλεία ασφάλειας σε επιχειρήσεις. Και οι δύο υποστηρίζουν το SSL για ασφάλεια από άκρο σε άκρο και το REST μπορεί να χρησιμοποιήσει την ασφαλή έκδοση του πρωτοκόλλου HTTP, το HTTPS.
- Το SOAP απαιτεί περισσότερο εύρος ζώνης. Το REST απαιτεί λιγότερους πόρους (ανάλογα με το API). Υπάρχει λίγο περισσότερη επιβάρυνση με το SOAP από την πύλη, λόγω του τύπου μεταφοράς του φορτίου. Επειδή το REST χρησιμοποιείται κυρίως για υπηρεσίες ιστού, η ελαφριά βαρύτητά του είναι ένα πλεονέκτημα σε αυτά τα σενάρια.
- Οι κλήσεις REST μπορούν να αποθηκευτούν προσωρινά, σε αντίθεση με τις κλήσεις SOAP. Τα δεδομένα μπορούν να επισημανθούν ως αποθηκευμένα στο αρχείο, πράγμα που σημαίνει ότι μπορεί να επαναχρησιμοποιηθεί από το πρόγραμμα περιήγησης αργότερα χωρίς να χρειάζεται να ξεκινήσει ένα άλλο αίτημα πίσω στο διακομιστή. Αυτό εξοικονομεί χρόνο και πόρους.

- Ένα API είναι κατασκευασμένο για να χειρίζεται το ωφέλιμο φορτίο της εφαρμογής σας και το REST και το SOAP το κάνουν διαφορετικά. Ένα ωφέλιμο φορτίο είναι τα δεδομένα που αποστέλλονται μέσω του Διαδικτύου και όταν ένα ωφέλιμο φορτίο είναι "βαρύ" απαιτεί περισσότερους πόρους. Το REST τείνει να χρησιμοποιεί HTTP και JSON, τα οποία ελαφρύνουν το ωφέλιμο φορτίο. Το SOAP βασίζεται περισσότερο στην XML

Γενικά, όταν πρόκειται για την δημιουργία ενός API για υπηρεσίες ιστού, η RESTful αρχιτεκτονική φαίνεται να είναι η καλύτερη επιλογή, εκτός από περιπτώσεις όπως για μια επιχείρηση που υποστηρίζεται από περισσότερους πόρους, χρειάζεται εξαιρετική ασφάλεια και έχει περισσότερες απαιτήσεις, όπου το SOAP είναι μια καλύτερη λύση.

3.8 : Δημιουργία ιστότοπου

Λόγω των αναγκών της εφαρμογής, ήταν αναγκαίο να δημιουργηθεί ένας ιστότοπος μέσω του όποιου, οι χρήστες της εφαρμογής, είτε πρόκειται για καθηγητές, είτε για προσωπικό του ιδρύματος είτε για φοιτητές, θα πρέπει μέσω αυτού να μπορούν να αλληλοεπιδράσουν με τον server, για να έχουν μια διαδραστική επαφή με την εφαρμογή.

Οι απαιτήσεις της ιστοσελίδας ήταν οι εξής:

- Γρήγορο
- Ασφαλές (Με πλήρη ταυτοποίηση χρήστη)
- Ευέλικτο (χρησιμοποιήσιμο από κινητές συσκευές)
- Προσαρμοποιήσιμο στις ανάγκες κάθε είδους χρήστη

Στην περίπτωσή μας υπήρχαν δύο εναλλακτικές. Είτε να χρησιμοποιηθεί το γνωστό τρίπτυχο HTML-CSS-Javascript , είτε , καθώς ο server είναι ένας Java application server να καταφύγουμε στην λύση του JSP (Java Server Pages).

3.8.1 : HTML

HTML σημαίνει HyperText Markup Language. Είναι η κύρια γλώσσα για την ανάπτυξη ιστοσελίδων. Η HTML γράφεται με τη χρήση στοιχείων HTML, τα οποία αποτελούνται από ετικέτες, κυρίως με ετικέτα ανοίγματος και ετικέτα κλεισίματος. Τα δεδομένα μεταξύ αυτών των ετικετών είναι συνήθως το περιεχόμενο. Ο κύριος στόχος της HTML είναι να επιτρέπει στα προγράμματα περιήγησης ιστού να ερμηνεύουν και να εμφανίζουν το περιεχόμενο που γράφεται μεταξύ των ετικετών. Οι ετικέτες έχουν σχεδιαστεί για να περιγράφουν το περιεχόμενο της σελίδας. Το HTML έρχεται με προκαθορισμένες ετικέτες. Επιτρέπουν σε κάποιον να εισάγει μαζί εικόνες, κείμενο, βίντεο, φόρμες και άλλα κομμάτια περιεχομένου σε συνεκτική ιστοσελίδα.

Τα στοιχεία της HTML είναι τα βασικά δομικά στοιχεία όλων των ιστοτόπων. Η HTML επιτρέπει την ενσωμάτωση εικόνων και αντικειμένων στην ιστοσελίδα. Μπορεί επίσης να χρησιμοποιηθεί για τη

δημιουργία διαδραστικών μορφών. Επίσης παρέχει τα μέσα για τη δημιουργία δομημένων εγγράφων. Το κάνει αυτό υποδηλώνοντας διαρθρωτική σημασιολογία για κείμενο όπως επικεφαλίδες, παραγράφους, λίστες, συνδέσμους, εισαγωγικά και άλλα στοιχεία. Ωστόσο, πλέον οι ιστοσελίδες σπάνια σχεδιάζονται χρησιμοποιώντας μόνο HTML. Το HTML επιτρέπει στον προγραμματιστή να ενσωματώνει σενάρια γραμμένα σε γλώσσες όπως JavaScript, και χρησιμοποιούν την τεχνολογία CSS για τον έλεγχο της παρουσίασης, της μορφοποίησης και διάταξης

3.8.2 : JSP

Το JSP (Java Server Pages) χρησιμοποιείται κυρίως για την ανάπτυξη δυναμικών ιστοσελίδων. Η τεχνολογία JSP επιτρέπει την ταχεία ανάπτυξη και την εύκολη συντήρηση των εν λόγω πλούσιων σε πληροφορία, δυναμικών ιστοσελίδων. Οι ιστοσελίδες JSP βασίζονται σε HTML, XML ή άλλους τύπους εγγράφων. Χρειάζονται επίσης έναν συμβατό web server που υποστηρίζει web servlet, όπως Apache Tomcat ή Jetty, για να τρέξει.

Το JSP κυκλοφόρησε για πρώτη φορά από την Sun Microsystems το 1999. Η Sun Microsystems αναλήφθηκε τελικά από την Oracle Corporation. Το JSP θεωρείται ότι είναι παρόμοιο με το PHP. Ωστόσο, το JSP βασίζεται στη γλώσσα προγραμματισμού Java. Καθώς το JSP είναι ένα πρόγραμμα Java, πρέπει να εκτελεστεί μέσα σε μια Java Virtual Machine (JVM), ειδικά μία που ενσωματώνεται στο λειτουργικό σύστημα του κεντρικού υπολογιστή του διακομιστή για να παρέχει ένα αφηρημένο περιβάλλον ουδέτερης πλατφόρμας.

Το JSP επιτρέπει επίσης την ταχεία ανάπτυξη εφαρμογών που βασίζονται στο Web και είναι ανεξάρτητες από την πλατφόρμα. Επιπλέον, η τεχνολογία του JSP επιτρέπει στους σχεδιαστές να αλλάξουν τη συνολική διάταξη της σελίδας χωρίς να αλλάξουν το υποκείμενο δυναμικό περιεχόμενο. Το JSP μπορεί επίσης να χρησιμοποιηθεί ανεξάρτητα ή ως το στοιχείο προβολής ενός σχεδιασμού MVC(Model View Controller)

3.8.3 : Σύγκριση και αξιοποίηση HTML στην εφαρμογή

- Οι σελίδες JSP προσθέτουν κώδικα από την πλευρά του server σε μια σελίδα HTML.
- Το JSP δημιουργεί δυναμικές σελίδες, ενώ το HTML δημιουργεί στατικές σελίδες.
- Το JSP είναι μια γλώσσα δέσμης ενεργειών από την πλευρά του server, ενώ το HTML είναι μια γλώσσα δέσμης ενεργειών από την πλευρά του client.
- Οι σελίδες HTML δίνουν έμφαση στην εμφάνιση, τη σημασιολογία και τη διάταξη των πληροφοριών στο πρόγραμμα περιήγησης. Οι σελίδες JSP μπορούν να επικαλεστούν ενσωματωμένες λειτουργίες από τον server.

- Το HTML εκτελείται στο πρόγραμμα περιήγησης στο Web. Το JSP τρέχει απευθείας στον Web Server και στον τοπικό JVM.
- Το HTML παρέχει ένα μέσο για να περιγράψει τη δομή των πληροφοριών που βασίζονται σε κείμενο σε ένα έγγραφο. Το JSP παρέχει μια δυναμική διεπαφή για τη συνεχή αλλαγή των δεδομένων και ενεργοποιεί δυναμικά τις ενέργειες του server.
- Το HTML φορτώνεται πιο γρήγορα καθώς εκτελείται στο τοπικό μηχάνημα. Το JSP χρειάζεται κάποιο χρόνο για να φορτώσει καθώς πρέπει να αλληλοεπιδράσει με τον Web Server.

Παρόλο που η τεχνολογία JSP είναι εύκολο να συνδεθεί με τον Java server που έχει δημιουργηθεί έτσι και αλλιώς, δεν καλύπτει όλες τις απαιτήσεις που έχουν τεθεί παραπάνω, καθώς είναι γενικά μια ‘δυσκίνητη’ τεχνολογία, που θέλει αρκετά μεγάλη παραμετροποίηση για να φτάσει τα standard που θέτονται στην σημερινή εποχή. Επίσης, δεν έχει την τεράστια υποστήριξη και community που θα έχουμε χρησιμοποιώντας τον κλασικό τρόπο δημιουργίας ιστοσελίδων.

Επιλέγοντας να μην χρησιμοποιήσουμε JSP, αυτό σημαίνει ότι πρέπει η ιστοσελίδα να επικοινωνεί με κάποιον τρόπο με τον server. Αυτό γίνεται μέσω δημιουργίας API στον server, και επικοινωνία σε αυτόν μέσω javascript και ανάλογων τεχνολογιών.

3.9 : jQuery

3.9.1 : Ορισμός και χαρακτηριστικά

Η jQuery είναι μια βιβλιοθήκη JavaScript που έχει σχεδιαστεί για να απλοποιεί τον προγραμματισμό ιστοσελίδων από την πλευρά του client της HTML. Δημιουργήθηκε τον Ιανουάριο του 2006 στο BarCamp NYC από τον John Resig. Χρησιμοποιείται από το περισσότερο από το 31% των 10.000 ιστοτόπων, και η δημοφιλέστερη βιβλιοθήκη JavaScript που χρησιμοποιείται σήμερα.

Η jQuery δεν είναι γλώσσα, αλλά είναι ένας σωστά γραμμένος κώδικας JavaScript. Όπως αναφέρεται στην επίσημη ιστοσελίδα του jQuery, "είναι μια γρήγορη και συνοπτική βιβλιοθήκη JavaScript που απλοποιεί τη μετάδοση εγγράφων HTML, τη διαχείριση συμβάντων, την κίνηση και τις αλληλεπιδράσεις του Ajax για γρήγορη ανάπτυξη ιστού".

Η jQuery παίρνει πολλές συνήθεις εργασίες που απαιτούν πολλές σειρές κώδικα JavaScript και τις περιβάλλει σε μεθόδους που μπορούμε να καλέσουμε με μία μόνο γραμμή κώδικα. Επίσης απλοποιεί επίσης πολλά από τα περίπλοκα πράγματα από τη JavaScript, όπως οι κλήσεις AJAX και η χειραγώγηση του DOM. Ουσιαστικά είναι μια βιβλιοθήκη με πολύ συμπαγές και σωστά γραμμένο κώδικα JavaScript που αυξάνει την παραγωγικότητα του προγραμματιστή επιτρέποντάς του να επιτύχει την κρίσιμη λειτουργικότητα UI γράφοντας πολύ μικρό κώδικα.

Συνοψίζοντας:

- Βοηθάει στη βελτίωση της απόδοσης της εφαρμογής
- Βοηθά στην ανάπτυξη συμβατών ιστοσελίδων με προγράμματα περιήγησης
- Βοηθά στην εφαρμογή κριτικής λειτουργικότητας που σχετίζεται με το UI χωρίς να γράφουμε εκατοντάδες γραμμές κώδικα
- Είναι επεκτάσιμη - η jQuery μπορεί να επεκταθεί για να εφαρμόσει προσαρμοσμένη συμπεριφορά
- Δεν χρειάζεται να μάθουμε νέα σύνταξη για τη χρήση του jQuery, γνωρίζοντας ότι η απλή σύνταξη JavaScript είναι αρκετή
- Απλός και καθαρότερος κώδικας, δεν χρειάζεται να γράφουμε αρκετές γραμμές κώδικα για να επιτύχουμε πολύπλοκες λειτουργίες

Η βιβλιοθήκη jQuery περιέχει τις ακόλουθες λειτουργίες:

- χειρισμός HTML / DOM
- χειρισμός CSS
- Μέθοδοι συμβάντων HTML
- Εφέ και κινούμενα σχέδια
- AJAX
- Βοηθητικά προγράμματα

3.9.2 : Αξιοποίηση jQuery στην εφαρμογή

Η JQuery χρησιμοποιήθηκε σε μεγάλό βαθμό για την διασύνδεση της ιστοσελίδας με την εφαρμογή, καθώς οι βιβλιοθήκες που μας παρέχει μειώνουν κατά πολύ τον κώδικα που χρειάζεται για να κατασκευαστούν απλές λειτουργίες, μαζί με τις περιπτώσεις λάθους που μπορεί να προκύψουν. Επίσης παρέχει συμβατότητα με άλλες τεχνολογίες της εφαρμογής που περιγράφονται παρακάτω, όπως το Ajax και το Bootstrap.

3.10 : Ajax

3.10.1 : Ορισμός και χαρακτηριστικά

Το Ajax είναι ένα client-side script που επικοινωνεί με και από ένα server / βάση δεδομένων χωρίς την ανάγκη για ανανέωση ή πλήρη ανανέωση σελίδας. Το Ajax είναι συνήθως ένας γενικός όρος για διάφορες τεχνικές JavaScript που χρησιμοποιούνται για τη σύνδεση έναν web server δυναμικά, χωρίς απαραίτητα να φορτώνει πολλές σελίδες. Πιο συγκεκριμένα, αναφέρεται στη χρήση αντικειμένων XmlHttpRequest για να αλληλοεπιδράσει με έναν web server δυναμικά μέσω JavaScript.

Παρακάτω παρουσιάζονται τα κύρια οφέλη από τη χρήση του Ajax στις εφαρμογές ιστού:

- Callbacks: Το Ajax χρησιμοποιείται για την πραγματοποίηση κλήσης, πραγματοποιώντας ένα γρήγορο ταξίδι προς και από τον server για να ανακτήσει και / ή να αποθηκεύσει δεδομένα χωρίς να αναρτήσει ολόκληρη τη σελίδα πίσω στο server. Με την αποτυχία πλήρους επιστροφής και αποστολής όλων των δεδομένων φόρμας στο server, η χρήση του δικτύου ελαχιστοποιείται και επιτυγχάνεται ταχύτερη λειτουργία . Σε τοποθεσίες και τοποθεσίες με περιορισμένο εύρος ζώνης, αυτό μπορεί να βελτιώσει σημαντικά την απόδοση του δικτύου. Τις περισσότερες φορές, τα δεδομένα που αποστέλλονται προς και από τον server είναι ελάχιστα. Με τη χρήση επανάκλησης, ο server δεν χρειάζεται να επεξεργαστεί όλα τα στοιχεία φόρμας. Με την αποστολή μόνο των απαραίτητων δεδομένων, υπάρχει περιορισμένη επεξεργασία στο server.
- Πραγματοποίηση ασύγχρονων κλήσεων: Το Ajax επιτρέπει την πραγματοποίηση ασύγχρονων κλήσεων σε έναν web server. Αυτό επιτρέπει στο πρόγραμμα περιήγησης του πελάτη να αποφύγει την αναμονή για την άφιξη όλων των δεδομένων προτού επιτρέψει στον χρήστη να πραγματοποιήσει ενέργειες στην ιστοσελίδα.
- Αυξημένη ταχύτητα: Ο κύριος σκοπός του Ajax είναι να βελτιώσει την ταχύτητα, την απόδοση και τη χρηστικότητα μιας εφαρμογής στο διαδίκτυο. Ένα εξαιρετικό παράδειγμα του Ajax είναι η δυνατότητα χαρακτηρισμού ταινιών στο Netflix. Ο χρήστης βαθμολογεί μια ταινία και η προσωπική της βαθμολογία για την ταινία αυτή θα αποθηκευτεί στη βάση δεδομένων τους χωρίς να περιμένει την ανανέωση ή την επαναφόρτωση της σελίδας. Αυτές οι αξιολογήσεις ταινιών αποθηκεύονται στη βάση δεδομένων τους χωρίς να αναρτάται ολόκληρη η σελίδα πίσω στο διακομιστή.

Το Ajax θα πρέπει να χρησιμοποιείται οπουδήποτε σε μια web εφαρμογή όπου μικρές ποσότητες πληροφοριών θα μπορούσαν να αποθηκευτούν ή να ανακτηθούν από το διακομιστή χωρίς να αναρτώνται ολόκληρες οι σελίδες. Ένα καλό παράδειγμα είναι η επικύρωση των δεδομένων πραγματοποιείται data validation. Ένα άλλο παράδειγμα θα ήταν η αλλαγή των τιμών σε ένα dropdown menu, που βασίζεται σε άλλες εισόδους της εκάστοτε φόρμας.

Ένα άλλο παράδειγμα είναι οι συμβουλές κειμένου και κουτιά κειμένου με αυτόματη συμπλήρωση. Ο client πληκτρολογεί δύο γράμματα και μια λίστα με όλες τις τιμές που ξεκινούν με αυτά τα γράμματα εμφανίζονται παρακάτω. Μια επιστροφή κλήσης γίνεται σε μια υπηρεσία ιστού που θα ανακτήσει όλες τις τιμές που ξεκινούν με αυτούς τους χαρακτήρες. Αυτό είναι ένα φανταστικό χαρακτηριστικό που θα ήταν αδύνατο χωρίς το Ajax .

3.10.2 : Αξιοποίηση στην εφαρμογή

Μια απαίτηση των σύγχρονων εφαρμογών είναι οι κλήσεις σε διαδικασίες διασύνδεσης να γίνεται ασύγχρονα, έτσι ώστε ο χρήστης της εφαρμογής να μην ‘κοκαλώνει’ σε κάποιο σημείο της εφαρμογής περιμένοντας να τελειώσει η εκάστοτε διαδικασία. Επίσης, όταν χρειάζεται να γίνει μια ανανέωση στην

ιστοσελίδα σε συγκεκριμένα κομμάτια αυτής, δεν χρειάζεται να γίνεται μια ολική ανανέωση της. Αυτά μπορούν να επιτευχθούν μόνο με την χρήση του Ajax.
Στην εφαρμογή μας, όλες οι κλήσεις γίνονται ασύγχρονα και τα δεδομένα ανανεώνονται μέσω του Ajax, για την καλύτερη εμπειρία του χρήστη. Επίσης η χρήση και οι περιπτώσεις σφαλμάτων σε κάποια διασύνδεση είναι πλήρως υλοποιημένα, κάτι κάνει την υλοποίηση διασύνδεσης αρκετά εύκολη και ευανάγνωστη από άλλους προγραμματιστές.

3.11 : Bootstrap

3.11.1 : Ορισμός και χαρακτηριστικά

Το Bootstrap είναι ένα ισχυρό πλαίσιο front-end για ταχύτερη και ευκολότερη ανάπτυξη ιστοσελίδων. Περιλαμβάνει πρότυπα σχεδίασης βασισμένα σε HTML και CSS για κοινά στοιχεία διεπαφής χρήστη όπως Forms, Buttons, Tables, Navigations, Dropdowns, Alerts, Modals, Tabs, Accordion, Carousel και πολλές άλλες καθώς και προαιρετικές επεκτάσεις JavaScript.

Κυριότερα, το bootstrap δίνει τη δυνατότητα να δημιουργηθεί ένα responsive layout με πολύ λιγότερη προσπάθεια.

Πλεονεκτήματα του bootstrap:

- Εξοικονόμηση χρόνου – Μπορεί να εξοικονομηθεί πολύς χρόνος και προσπάθεια χρησιμοποιώντας τα προκαθορισμένα πρότυπα και κλάσεις σχεδιασμού του Bootstrap και να επικεντρωθεί ο προγραμματιστής σε άλλες εργασίες ανάπτυξης.
- Responsive Design - Χρησιμοποιώντας το Bootstrap μπορούν εύκολα να δημιουργηθούν σχέδια που ανταποκρίνονται στις ανάγκες μας. Οι λειτουργίες για responsive design του Bootstrap κάνουν τις ιστοσελίδες μας να εμφανίζονται πιο κατάλληλα σε διαφορετικές συσκευές και αναλύσεις οθόνης .
- Συνεπής σχεδιασμός - Όλα τα στοιχεία Bootstrap μοιράζονται τα ίδια πρότυπα σχεδίασης και στυλ μέσω μιας κεντρικής βιβλιοθήκης, έτσι ώστε τα σχέδια και τα σχέδια των ιστοσελίδων μας να είναι συνεπή σε όλη την ανάπτυξή τους.
- Εύκολο στη χρήση - Το Bootstrap είναι πολύ εύκολο στη χρήση. Οποιοσδήποτε με τις βασικές γνώσεις HTML και CSS μπορεί να ξεκινήσει την ανάπτυξη με το Bootstrap.
- Συμβατό με προγράμματα περιήγησης - Το Bootstrap δημιουργείται για σύγχρονα προγράμματα περιήγησης, που σημαίνει ότι είναι συμβατό με όλα τα σύγχρονα δημοφιλή προγράμματα περιήγησης όπως Mozilla Firefox, Google Chrome, Safari, Internet Explorer και Opera.

3.11.2 : Αξιοποίηση στην εφαρμογή

Η εφαρμογή πρέπει να είναι διαδραστική (responsive) και ο μόνος τρόπος να επιτευχθεί αυτό είναι μέσω του Bootstrap. Το bootstrap παρέχει components που από μόνα τους υλοποιούν αυτή την απαίτηση, καθώς προσαρμόζονται στις ανάγκες της εκάστοτε συσκευής από την οποία είναι προσβάσιμα. Αυτό σημαίνει ότι ένας χρήστης μίας συσκευής με μικρή οθόνη θα έχει την ίδια ευκολία πρόσβασης στην ιστοσελίδα όσο ένας χρήστης επιτραπέζιου υπολογιστή.

Το bootstrap μέσω μιας extra βιβλιοθήκης μας παρέχει επίσης νέα components για μηνύματα χρήσης (alert box) και loaders, τα οποία ακολουθούν και υλοποιούν ένα ενιαίο μοντέρνο σχεδιασμό και λειτουργίες, και είναι συμβατά με όλους τους browser με τον ίδιο ενιαίο τρόπο.
Η συγκεκριμένη βιβλιοθήκη (bootbox) και τα συστατικά της χρησιμοποιούνται για όλες τις προαναφερθέντες υπηρεσίες ελέγχου της ιστοσελίδας.

3.12 : Cookies

3.12.1 : Ορισμός και χαρακτηριστικά

Μπορούμε να σκεφτούμε τα cookies ως αρχεία κειμένου, τα οποία αποθηκεύονται στον υπολογιστή μας. Κατόπιν αιτήματος ενός webserver, το πρόγραμμα περιήγησης δημιουργεί ένα τέτοιο αρχείο.

Τα cookies έχουν κάποια συγκεκριμένα χαρακτηριστικά:

- Ο webserver μπορεί να διαβάσει και να γράψει περιεχόμενο από και προς αυτό το αρχείο. Οι webservers μπορούν να έχουν πρόσβαση μόνο σε cookies που έχουν οριστεί σε δικό τους domain.
- Σύμφωνα με το πρωτόκολλο HTTP, τα cookies δεν μπορούν να είναι μεγαλύτερα από 4096 Bytes (4KB) το καθένα.
- Υπάρχει ένα όριο στον αριθμό των cookies ανά domain. Ο αριθμός διαφέρει ανά πρόγραμμα περιήγησης, ωστόσο, το γενικά χρησιμοποιούμενο όριο είναι είκοσι cookies.
- Υπάρχει ένα όριο στο συνολικό αριθμό των cookies στο σκληρό δίσκο του πελάτη. Αυτός ο αριθμός διαφέρει επίσης ανά πρόγραμμα περιήγησης, αλλά συνήθως περιορίζεται σε περίπου 300 cookies. Όταν ξεπεραστεί αυτός ο αριθμός, ένα παλιότερο cookie διαγράφεται πριν δημιουργηθεί ένα νέο.
- Τα cookies έχουν ημερομηνία λήξης. Αυτή η ημερομηνία έχει οριστεί έτσι ώστε το πρόγραμμα περιήγησης να μπορεί να διαγράψει τα παλιά cookies όταν δεν χρειάζονται πλέον. Εάν η ημερομηνία λήξης είναι κενή, το cookie θα διαγραφεί όταν κλείσει η σύνδεση με τον server. Αυτό συμβαίνει όταν το παράθυρο ή η καρτέλα του ιστότοπου κλείνει από το χρήστη ή όταν ο χρήστης κλείσει ολόκληρο το πρόγραμμα περιήγησης. Αυτά τα cookies, μερικές φορές καλούνται

cookies περιόδου λειτουργίας, χρησιμοποιούνται κυρίως για την αποθήκευση προσωρινών ρυθμίσεων.

Σήμερα, τα cookies χρησιμοποιούνται για σχεδόν κάθε σκοπό. Μπορούμε να χρησιμοποιηθούν για την αποθήκευση των ρυθμίσεων χρήστη, όπως όνομα, γλώσσα, τοποθεσία ή μέγεθος οθόνης. Αυτό μπορεί να βελτιώσει την ποιότητα της υπηρεσίας που παρέχεται σε έναν πελάτη, επειδή μπορούμε να βελτιστοποιηθεί η υπηρεσία για έναν πελάτη και να υπάρχει μνήμη για αυτήν τη βελτιστοποίηση στο μέλλον. Για παράδειγμα, μπορούμε να αποθηκευτεί η προτιμώμενη γλώσσα του πελάτη σε ένα cookie και, στη συνέχεια, να εμφανιστεί το περιεχόμενο του ιστότοπου στην προτιμώμενη γλώσσα κάθε φορά που ο πελάτης επισκέπτεται τον επισκέπτεται.

3.12.2 : Αξιοποίηση στην εφαρμογή

Η ιστοσελίδα και οι υπηρεσίες που χρησιμοποιούνται χρειάζονται κάποιο σύστημα ταυτοποίησης και ασφάλειας. Καθώς η HTML είναι στατική, οι πληροφορίες που χρειάζονται δεν μεταφέρονται κατά την πλοήγηση στα διάφορα κομμάτια του ιστότοπου. Μέσω των cookies επιλύεται αυτό το πρόβλημα, καθώς κατά την είσοδο κάθε χρήστη στην ιστοσελίδα, μετά την ταυτοποίηση του μέσω του συστήματος αυθεντικοποίησης, παρέχεται με ένα ‘token’ αυθεντικοποίησης, και μέσω αυτού προσαρμόζεται το περιεχόμενο που εμφανίζεται, καθώς και οι υπηρεσίες που μπορεί να χρησιμοποιηθούν. Επίσης, το συγκεκριμένο token εμπεριέχεται μέσα σε ένα cookie, το οποίο έχει ορισμένη διάρκεια ζωής (2 ώρες), ώστε ο χρήστης να παραμένει συνδεμένος και να μην χρειάζεται να ταυτοποιείται συνέχεια στην ιστοσελίδα.

Κεφάλαιο 4 : Υπηρεσίες που εξυπηρετούνται

Η παρούσα εφαρμογή έχει δημιουργηθεί με σκοπό την άμεση ενημέρωση προσώπων για λειτουργίες του ιδρύματος του ΑΤΕΙ/Θ. Στην αρχική υλοποίησή της υποστηρίζει την ενημέρωση φοιτητών για ακαδημαϊκά θέματα, είτε μέσω ερωτημάτων από τους ίδιους μέσω αποστολής SMS προς την εφαρμογή, είτε μέσω της παραλαβής ενημερωτικών μηνυμάτων από την εφαρμογή που έχουν αποσταλεί από τους καθηγητές των μαθημάτων που παρακολουθούν. Επίσης, υποστηρίζεται η ενημέρωση των εθελοντών αιμοδοτών για τις αιμοδοσίες που πραγματοποιούνται από το ιατρείο του ΑΤΕΙ/Θ, είτε μέσω ερωτημάτων από αυτούς από την εφαρμογή, είτε μέσω μαζικών ενημερωτικών μηνυμάτων της εφαρμογής προς αυτούς για επικείμενες αιμοδοσίες. Επιπρόσθετα, υποστηρίζεται πιλοτικά η αποστολή στοχευμένων προκαθορισμένων μηνυμάτων προς μεμονωμένα τηλέφωνα, που αφορούν λειτουργίες του ιδρύματος.

4.1 : Ακαδημαϊκά Θέματα

4.1.1 : Ανάκτηση ακαδημαϊκών δεδομένων από φοιτητές

Οι φοιτητές του ΑΤΕΙ/Θ μπορούν να ενημερωθούν για προσωπικά δεδομένα που υπάρχουν στην ακαδημαϊκή βάση του ιδρύματος Pithia. Οι φοιτητές, αφού εγγράφουν στην υπηρεσία και επιβεβαιώσουν το τηλέφωνο τους, μπορούν να στείλουν κάποιο από τα προκαθορισμένα μηνύματα(Πίνακας 4.1) στο νούμερο της εφαρμογής (54584) και να λάβουν άμεσα πληροφορίες που τους αφορούν.

Παράδειγμα μηνύματος : TEITHE VATHMOS ΔΙΚΤΥΑ Η/Υ

Πίνακας 4.1

Μήνυμα	Παράμετρος	Περιγραφή
TEITHE VATHMOS	Όνομα μαθήματος	Επιστρέφει τον βαθμό του χρήστη στο μάθημα που δίνει ως παράμετρο
TEITHE VATHMOS KSEXWR	Όνομα μαθήματος	Επιστρέφει τους βαθμούς του χρήστη για την θεωρία και το εργαστήριο για το μάθημα που δίνει ως παράμετρο
TEITHE DM		Επιστρέφει τις διδακτικές μονάδες του χρήστη
TEITHE DM PTYXIO		Επιστρέφει τις διδακτικές μονάδες που πρέπει να συμπληρωθούν από τον χρήστη για πτυχίο
TEITHE EGGRAFI	Όνομα μαθήματος	Επιστρέφει επιβεβαίωση εγγραφής του χρήστη στο μάθημα που δίνει ως παράμετρο
TEITHE EGGRAFI ERGASTIRIO	Όνομα μαθήματος	Επιστρέφει το τμήμα του εργαστηρίου που δόθηκε, καθώς και τις ώρες που πραγματοποιείται
TEITHE PITHIALOGIN		Επιστρέφει τα διαπιστευτήρια του χρήστη για το σύστημα της πυθίας

4.1.2 : Ενημέρωση ομάδων φοιτητών για ακαδημαϊκά θέματα

Οι καθηγητές του τμήματος μπορούν να στείλουν ενημερωτικά μηνύματα προς τους μαθητές που είναι εγγεγραμμένοι σε κάποιο από τα μαθήματα που διδάσκουν στο τρέχον διδακτικό εξάμηνο. Η υπηρεσία είναι διαθέσιμη για όλους τους καθηγητές του ΑΤΕΙ/Θ, και τα τηλέφωνα των μαθητών βρίσκονται αυτόματα από την βάση του ιδρύματος. Με την αποστολή μηνυμάτων επιτυγχάνεται η έγκυρη ενημέρωση των φοιτητών άμεσα για κρίσιμα θέματα ή ανακοινώσεις των μαθημάτων. Η μοναδική προϋπόθεση είναι η ύπαρξη των τηλεφώνων των φοιτητών στην βάση της πυθίας (η καταχώρηση αυτή

γίνεται κατά την εγγραφή τους στο ίδρυμα) ή στην τοπική βάση της εφαρμογής, μέσα από την ιστοσελίδα της(sms.teithe.gr).

Η εφαρμογή, καθώς χρησιμοποιεί την διασύνδεση του SMS Aggregator για την αποστολή αυτών των μηνυμάτων, έχει δυνατότητα να παρέχει μόνο συγκεκριμένα προκαθορισμένα μηνύματα προς αποστολή, με την δυνατότητα να οριστούν παράμετροι σε θέσεις που έχουν οριστεί σε αυτά τα μηνύματα. Παρακάτω παρουσιάζονται τα διαθέσιμα μηνύματα που μπορούν να σταλούν για ενημερωτικούς σκοπούς.

Ακύρωση/Αναπλήρωση μαθήματος:

- Το μάθημα [όνομα μαθήματος] της [ημερομηνία] αναβάλλεται.
- Το μάθημα [όνομα μαθήματος] της [ημερομηνία] αναβάλλεται και θα πραγματοποιηθεί στις [ημερομηνία].
- Θα γίνει αναπλήρωση του μαθήματος [όνομα μαθήματος] την [ημερομηνία].

Ανάθεση/Παρουσίαση εργασιών:

- Έχει ανακοινωθεί εργασία για το μάθημα [όνομα μαθήματος]. Προθεσμία εώς [ημερομηνία].
- Η προθεσμία για την εργασία στο μάθημα [όνομα μαθήματος] λήγει στις [ημερομηνία].
- Θα γίνει παρουσίαση εργασιών για το μάθημα [όνομα μαθήματος] στις [ημερομηνία].

Γενικές ανακοινώσεις:

- Διαβάστε σημαντική ανακοίνωση για το μάθημα [όνομα μαθήματος] στο [τοποθεσία ανακοίνωσης].
- Έχουν ανακοινωθεί βαθμολογίες για το μάθημα [όνομα μαθήματος].

4.1.3 : Στοχευμένα μηνύματα

Οι καθηγητές και τα μέλη του διοικητικού προσωπικού μπορούν να στείλουν μηνύματα ακαδημαϊκού χαρακτήρα προς επιλεγμένους παραλήπτες της επιλογής τους. Μέσω της ιστοσελίδας, μπορούν επίσης να κάνουν αναζήτηση για διάφορα καταχωρημένα νούμερα κινητών τηλεφώνων που υπάρχουν στις βάσεις του ιδρύματος μέσω ονόματος ή όνομα χρήστη στην ακαδημαϊκή βάση. Τα μηνύματα αυτά, όπως συμβαίνει και με τα μαζικά μηνύματα ενημέρωσης φοιτητών είναι προκαθορισμένα με δυνατότητα αλλαγής παραμέτρων μέσα σε αυτά. Προς το παρόν υποστηρίζονται μόνο μηνύματα που υποστηρίζουν την ενημέρωση για θέματα πτυχιακών. Τα μηνύματα φαίνονται παρακάτω.

Παρουσίαση πτυχιακών:

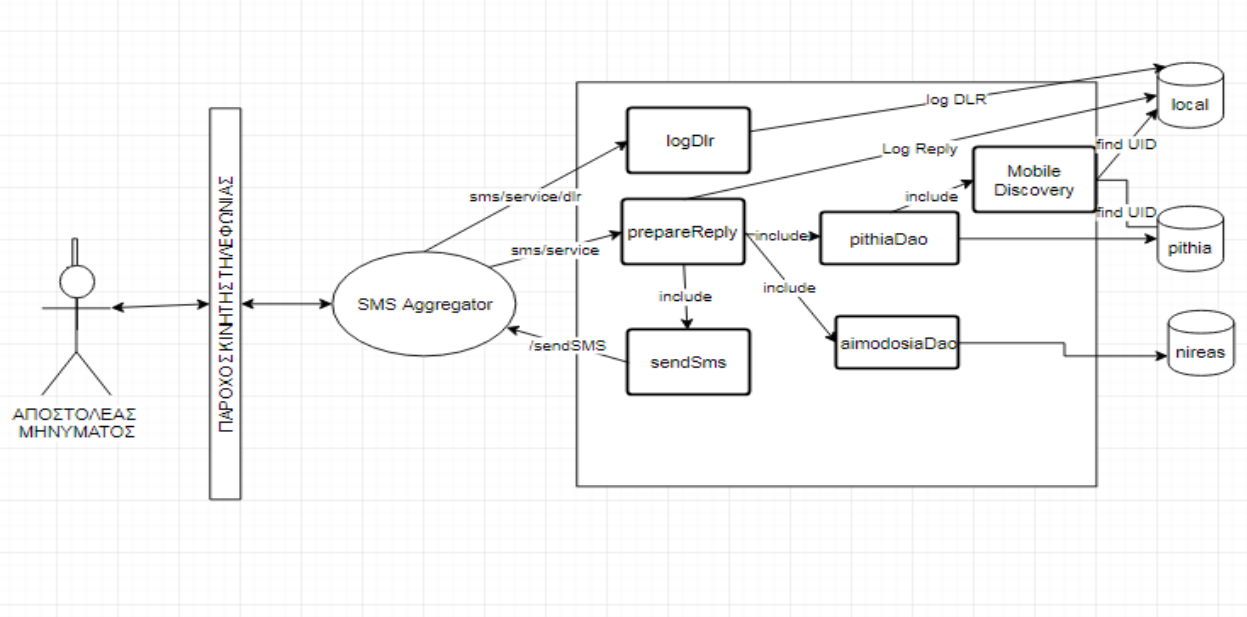
- Είστε επιβλέπων για την πτυχιακή εργασία του [φοιτητής] στις [ημερομηνία].
- Η παρουσίαση της πτυχιακής σας εργασίας είναι στις [ημερομηνία].

Κεφάλαιο 5 : Η εφαρμογή

5.1 : Ανάκτηση δεδομένων μέσω SMS

Η υπηρεσία ανάκτησης δεδομένων είναι ανοιχτή σε όλους τους χρήστες που κατέχουν ακαδημαϊκό κωδικό και είναι εγγεγραμμένοι στην υπηρεσία του SMS Aggregator. Αυτοί οι χρήστες μπορούν να χρησιμοποιήσουν οποιαδήποτε από τις διαθέσιμες υπηρεσίες για ανάκτηση προσωπικών δεδομένων τους, εφόσον αυτά υπάρχουν μέσω των διαθέσιμων ερωτημάτων. Παρακάτω παρουσιάζεται το διάγραμμα χρήσης της υπηρεσίας (Εικόνα 5.1Εικόνα 5.8).

Εικόνα 5.1



Η διαδικασία ανάκτησης δεδομένων από την εφαρμογή ξεκινάει από ένα γραπτό μήνυμα του χρήστη προς το προκαθορισμένο νούμερο (54584) που έχει οριστεί απο το GUNET. Ο χρήστης μπορεί να στείλει μόνο προκαθορισμένα μηνύματα (με ή χωρις παραμέτρους), τα οποία μπορεί να βρει στον οδηγό μηνυμάτων που υπάρχει στην ιστοσελίδα της εφαρμογής (sms.teithe.gr)(Εικόνα 5.2,Εικόνα 5.3)

Εικόνα 5.2

ΑΚΑΔΗΜΑΙΚΑ ΜΗΝΥΜΑΤΑ

ΜΗΝΥΜΑ	ΠΕΡΙΓΡΑΦΗ
TEITHE VATHMOS <ΟΝΟΜΑ ΜΑΘΗΜΑΤΟΣ>	ΕΠΙΣΤΡΕΦΕΙ ΤΟΝ ΒΑΘΜΟ ΤΟΥ ΧΡΗΣΤΗ ΣΤΟ ΣΥΓΚΕΚΡΙΜΕΝΟ ΜΑΘΗΜΑ
TEITHE VATHMOS KSEXWR<ΟΝΟΜΑ ΜΑΘΗΜΑΤΟΣ>	ΕΠΙΣΤΡΕΦΕΙ ΤΟΥΣ ΒΑΘΜΟΥΣ ΤΟΥ ΧΡΗΣΤΗ ΓΙΑ ΘΕΩΡΙΑ ΚΑΙ ΕΡΓΑΣΤΗΡΙΟ ΓΙΑ ΤΟ ΣΥΓΚΕΚΡΙΜΕΝΟ ΜΑΘΗΜΑ
TEITHE DM	ΕΠΙΣΤΡΕΦΕΙ ΤΙΣ ΔΙΔΑΚΤΙΚΕΣ ΜΟΝΑΔΕΣ ΤΟΥ ΧΡΗΣΤΗ
TEITHE DM PTYXIO	ΕΠΙΣΤΡΕΦΕΙ ΤΙΣ ΔΙΔΑΚΤΙΚΕΣ ΜΟΝΑΔΕΣ ΠΟΥ ΥΠΟΛΟΙΠΟΝΤΑΙ ΓΙΑ ΠΤΥΧΙΟ
TEITHE EGGRAFI <ΟΝΟΜΑ ΜΑΘΗΜΑΤΟΣ>	ΕΠΙΣΤΡΕΦΕΙ ΕΠΙΒΕΒΑΙΩΣΗ ΕΓΓΡΑΦΗΣ ΣΤΟ ΣΥΓΚΕΚΡΙΜΕΝΟ ΜΑΘΗΜΑ
TEITHE EGGRAFI ERGASTIRIO<ΟΝΟΜΑ ΜΑΘΗΜΑΤΟΣ>	ΕΠΙΣΤΡΕΦΕΙ ΤΟ ΕΡΓΑΣΤΗΡΙΟ ΚΑΙ ΤΙΣ ΩΡΕΣ ΠΟΥ ΠΡΑΓΜΑΤΟΠΟΙΕΙΤΑΙ
TEITHE PITHIALOGIN	ΕΠΙΣΤΡΕΦΕΙ ΤΑ ΔΙΑΠΙΣΤΕΥΤΗΡΙΑ ΤΟΥ ΧΡΗΣΤΗ

Εικόνα 5.3

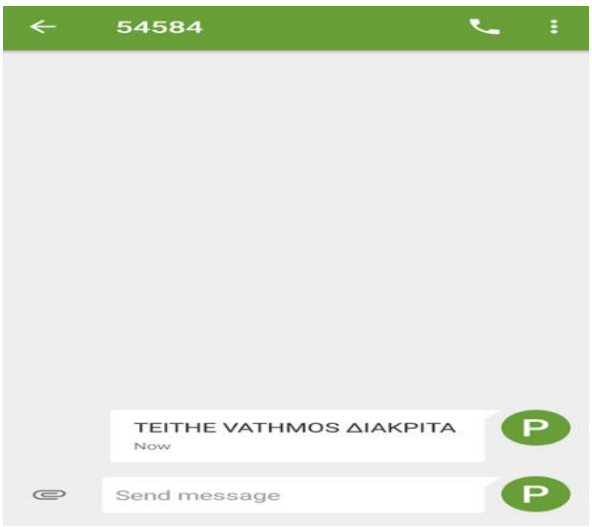
ΜΗΝΥΜΑΤΑ ΑΙΜΟΔΟΣΙΑΣ

ΜΗΝΥΜΑ	ΠΕΡΙΓΡΑΦΗ
TEITHE AIMODOSIA	ΕΓΓΡΑΦΗ ΤΟΥ ΧΡΗΣΤΗ ΣΤΗΝ ΥΠΗΡΕΣΙΑ ΑΥΤΟΜΑΤΗΣ ΑΠΟΣΤΟΛΗΣ ΜΗΝΥΜΑΤΩΝ ΑΙΜΟΔΟΣΙΑΣ
TEITHE AIMODOSIA STOP	ΔΙΑΚΟΠΗ ΑΥΤΟΜΑΤΩΝ ΜΗΝΥΜΑΤΩΝ ΑΠΟ ΤΗΝ ΥΠΗΡΕΣΙΑ ΤΗΣ ΑΙΜΟΔΟΣΙΑΣ
TEITHE AIMODOSIA TELEYTAIA	ΕΠΙΣΤΡΕΦΕΙ ΤΗΝ ΗΜΕΡΟΜΗΝΙΑ ΤΗΣ ΤΕΛΕΥΤΑΙΑΣ ΦΟΡΑΣ ΠΟΥ ΕΔΩΣΕ ΑΙΜΑ Ο ΧΡΗΣΤΗΣ
TEITHE AIMODOSIA FIALES	ΕΠΙΣΤΡΕΦΕΙ ΤΟΝ ΑΡΙΘΜΟ ΦΙΑΛΩΝ ΠΟΥ ΔΙΚΑΙΟΥΤΑΙ Ο ΧΡΗΣΤΗΣ
TEITHE AIMODOSIA FIALKATANAL	ΕΠΙΣΤΡΕΦΕΙ ΤΟΝ ΑΡΙΘΜΟ ΤΩΝ ΦΙΑΛΩΝ ΑΙΜΑΤΟΣ ΠΟΥ ΕΧΟΥΝ ΚΑΤΑΝΑΛΩΘΕΙ ΑΠΟ ΤΟΝ ΧΡΗΣΤΗ

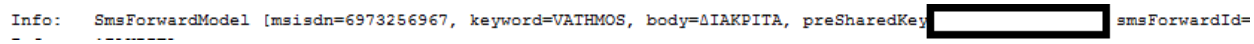
Εφόσον το μήνυμα ξεκινάει με “TEITHE” και έχει κάποιο από τα keyword που αναφέρονται(παράδειγμα μηνύματος Εικόνα 5.4), θα παραληφθεί από την πλατφόρμα του GUNET, και θα προωθηθεί κατάλληλα στο ορισμένο endpoint της εφαρμογής μας. (Εικόνα 5.5)

Σημείωση : Για να αποσταλεί ένα τέτοιο μήνυμα ο χρήστης πρέπει να είναι εγγεγραμμένος στην υπηρεσία του GUNET, διαφορετικά δεν του επιτρέπεται να χρησιμοποιήσει την υπηρεσία.

Εικόνα 5.4



Εικόνα 5.5



Εκεί, ύστερα από επεξεργασία του μηνύματος ως προς την ακεραιότητα / ασφάλεια του (που ορίζονται από προκαθορισμένα κλειδιά τα οποία έχουμε ανταλλαχθεί με τον SMS Aggregator), το μήνυμα θα επεξεργαστεί από την ανάλογη διεργασία, αναλόγως τον τύπο υπηρεσίας που ανήκει (υπηρεσία ακαδημαϊκών μηνυμάτων ή αιμοδοσίας).

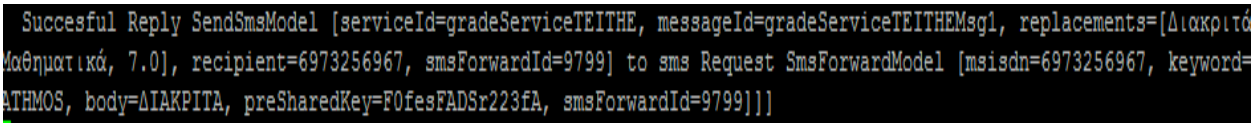
Σε περίπτωση που πρόκειται για ακαδημαϊκό μήνυμα, θα πρέπει να βρεθεί ο χρήστης στον οποίο αντιστοιχεί το νούμερο κινητού τηλεφώνου από το οποίο στάλθηκε το μήνυμα, ώστε να τον ταυτόποιθεί και να μπορούν να εκτελεσθούν τα αντίστοιχα ερωτήματα στην βάση του ιδρύματος. Αυτό γίνεται μέσω μιας βοηθητικής κλάσης 'Mobile Discovery' η οποία επιστρέφει το πραγματικό ID του χρήστη, κάνοντας αναζήτηση πρώτα στην τοπική βάση της εφαρμογής (στην περίπτωση που ο χρήστης έχει επιβεβαιώσει το νούμερο του) και εφόσον δεν το βρει, στην βάση της Πυθίας.

Σε περίπτωση που πρόκειται αιμοδοσίας, τα διάφορα ερωτήματα γίνονται απευθείας μέσω του νούμερου κινητού τηλεφώνου του χρήστη στην βάση της αιμοδοσίας, οπότε δεν χρειάζεται κάποια επιπλέον διαδικασία.

Και στις δύο παραπάνω περιπτώσεις, η διαδικασία συνεχίζεται με την αποστολή ερωτήματος στις αντίστοιχες βάσεις τους. Εάν το ερώτημα ολοκληρωθεί επιτυχώς, τότε προετοιμάζεται το ανάλογο μήνυμα απάντησης με αποδέκτη τον χρήστη. Εφόσον υπάρξει κάποιο σφάλμα στην ανάκτηση δεδομένων (μη εύρεση ID χρήστη, δεδομένων προς απάντηση, κλπ) του αποστέλλεται το μήνυμα λάθος που έχει οριστεί για κάθε υπηρεσία.

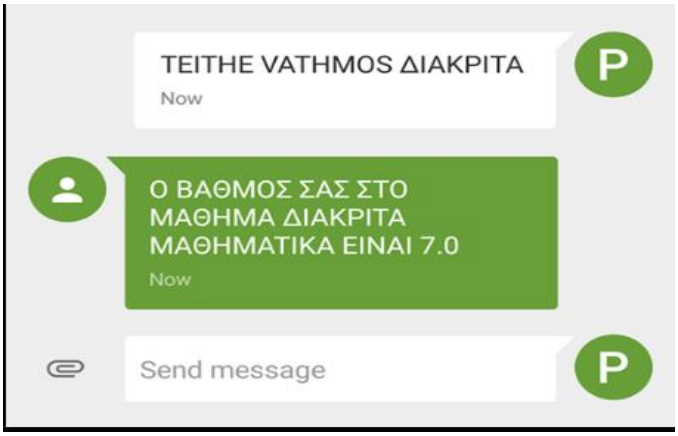
Εφόσον ετοιμαστεί το μήνυμα βρίσκοντας τα κατάλληλα replacements για τα εκάστοτε μηνύματα, καλείται η διασύνδεση που παρέχεται από το GUNET για να σταλθεί η απάντηση στον χρήστη. Εφόσον δεν υπάρχει κάποιο σφάλμα στις συνθήκες αποστολής, καταγράφεται στα αρχεία καταγραφής του server το παρακάτω μήνυμα(Εικόνα 5.6).

Εικόνα 5.6



Εφόσον γίνει η αποστολή, καταγράφεται η απάντηση που λάβαμε για την αποστολή του μηνύματος στην μνήμη του προγράμματος ,καθώς και το ίδιο το μήνυμα. Ο χρήστης που έστειλε το μήνυμα παραλαμβάνει την απάντηση στο κινητό του τηλέφωνου (Εικόνα 5.7).

Εικόνα 5.7



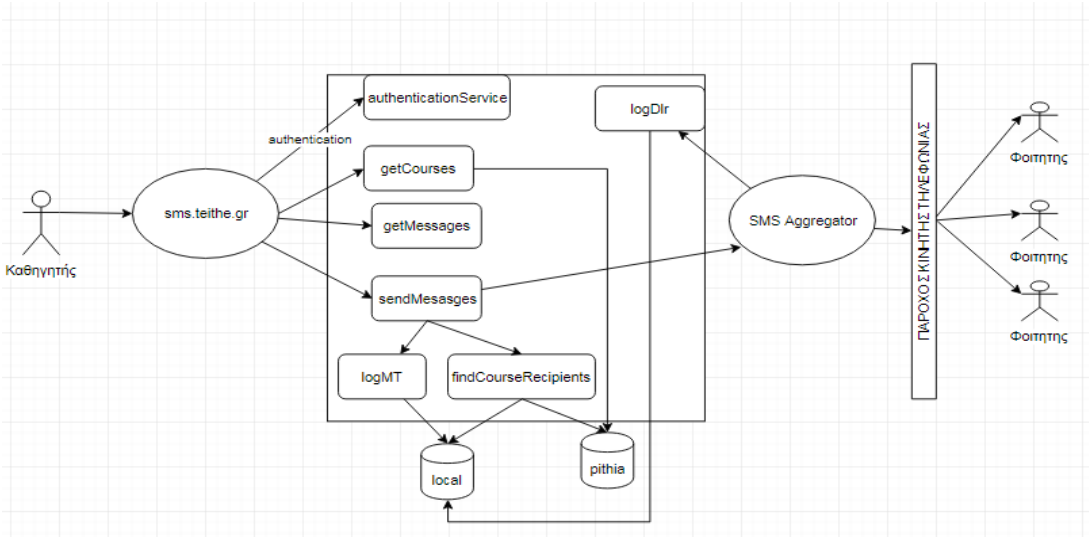
Αφού γίνει η αποστολή του μηνύματος από τον SMS Aggregator , στέλνεται στην εφαρμογή ένα Delivery Report (DLR) από τον SMS Aggregator για την πραγματική κατάσταση αποστολής του μηνύματος. Η διεύθυνση διαδικτύου στην οποία αποστέλλεται αυτό το μήνυμα ορίζεται στο κατά το αίτημα αποστολής μηνύματος που στέλνει η εφαρμογή μας. Όταν το μήνυμα καταφθάνει στην εφαρμογή μας, καταγράφεται και αυτό στην μνήμη του προγράμματος.

Τελικά ,ανά τακτά ορισμένα χρονικά διαστήματα, το πρόγραμμα αποθηκεύει τα δεδομένα των απεσταλμένων μηνυμάτων καθώς και τα DLR στην τοπική βάση, και αδειάζει την προσωρινή μνήμη.

5.2 : Ενημέρωση ομάδων φοιτητών από καθηγητές

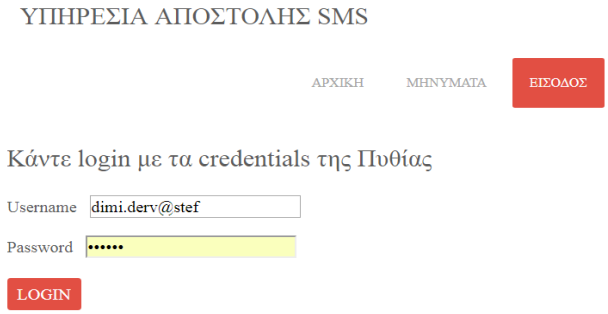
Πρόσβαση στην συγκεκριμένη υπηρεσία παρέχεται μόνο σε καθηγητές, καθώς τα μηνύματα μπορούν να σταλούν μόνο σε ομάδες φοιτητών που ανήκουν σε μαθήματα. Παρακάτω παρουσιάζεται το διάγραμμα χρήσης της υπηρεσίας (Εικόνα 5.8).

Εικόνα 5.8



Η διαδικασία ξεκινάει μέσω της ιστοσελίδας sms.teithe.gr, οπου ο καθηγητής πρέπει να κάνει είσοδο στην ιστοσελίδα και να πιστοποιηθεί, ώστε να έχει πρόσβαση για την αποστολή μηνυμάτων.(Εικόνα 5.9)

Εικόνα 5.9

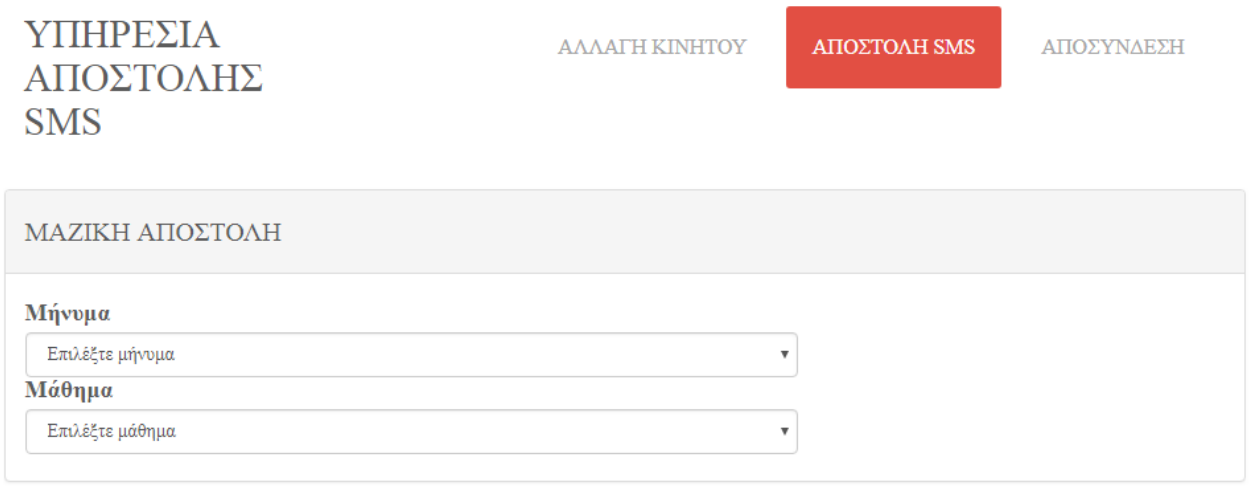


Πατώντας το κουμπί login, στέλνεται ένα αίτημα στο endpoint αυθεντικοποίησης της εφαρμογής μας. Αυτό, θα προσπαθήσει να ταυτοποιήσει τον χρήστη με 2 τρόπους. Αρχικά μέσω της βάσης της πυθίας, χρησιμοποιώντας ένα view που επιστρέφει ΝΑΙ ή ΟΧΙ εάν ο συνδυασμός username/password είναι σωστός. Εάν η απάντηση είναι αρνητική, γίνεται αναζήτηση στο σύστημα LDAP του ΑΤΕΙ/Θ, ελέγχοντας εάν το password που δόθηκε ισχύει για τον υπάρχον χρήστη.

Παρόλο που το σύστημα εισόδου μέσω LDAP λειτουργεί κανονικά για LDAP λογαριασμούς, ένα γίνει είσοδος με αυτό δεν υπάρχει δυνατότητα να αντιστοιχιστεί με τα υπάρχοντα μαθήματα αργότερα, οπότε χρησιμοποιείται μόνο για άλλους σκοπούς (ενημέρωση κινητού τηλεφώνου, περιγράφεται στην Ενότητα 5.5 :).

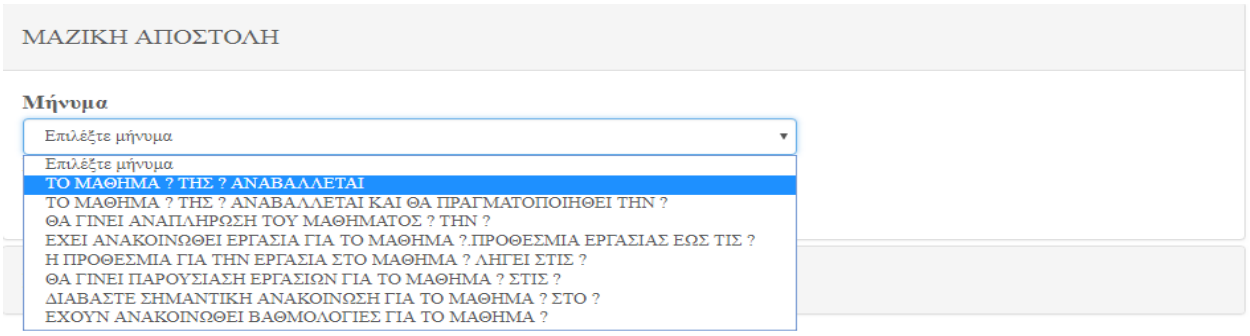
Εφόσον η σύνδεση είναι επιτυχής, επιστρέφεται στον χρήστη σε μορφή Cookie ένα token αυθεντικοποίησης. Μέσω αυτού του token μπορεί στην συνέχεια να χρησιμοποιεί τις υπηρεσίες που του παρέχονται, καθώς και να λαμβάνει προσωποποιημένες πληροφορίες (τα μαθήματα που διδάσκει) . Ύστερα, γίνεται redirect στην σελίδα από την οποία μπορεί να στείλει τα μαζικά μηνύματα ενημέρωσης σε φοιτητές.(Εικόνα 5.10)

Εικόνα 5.10



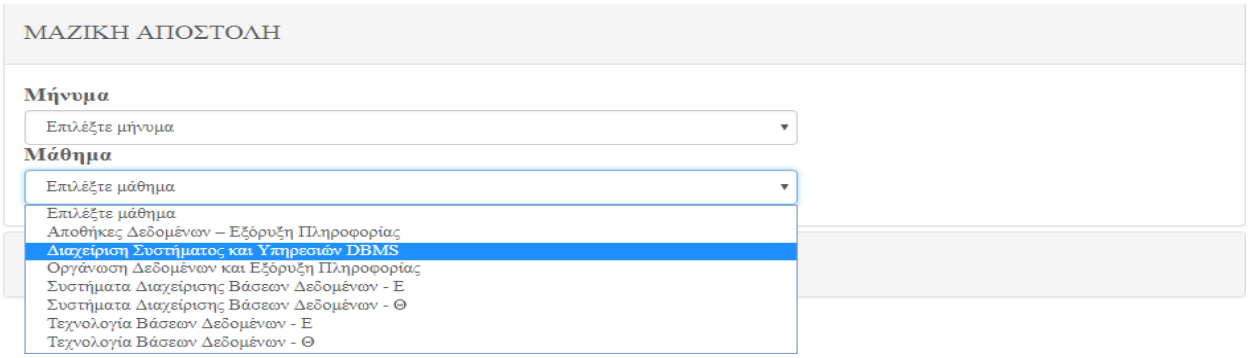
Κατά την φόρτωση της ιστοσελίδας, πραγματοποιούνται 2 αιτήματα προς την εφαρμογή.
Το 1ο είναι για να μας φέρει δυναμικά τα μηνύματα τα οποία μπορεί να στείλει ο καθηγητής(Εικόνα 5.11).

Εικόνα 5.11



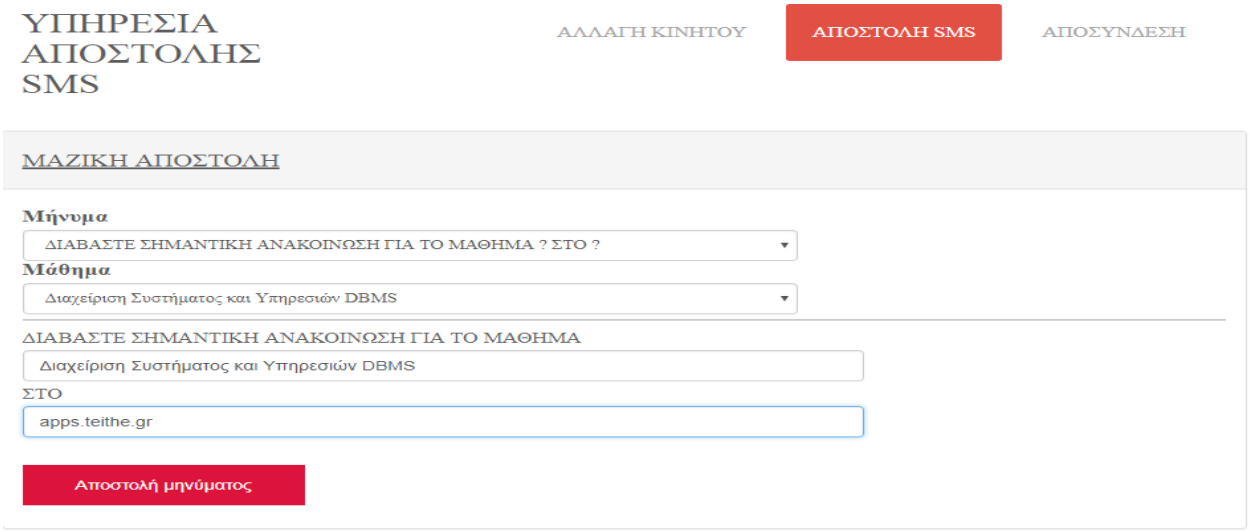
Το 2ο είναι για να μας φέρει τα μαθήματα τα οποία διδάσκει ο καθηγητής την συγκεκριμένη περίοδο (τα δεδομένα παρέχονται από την βάση της πυθίας (Εικόνα 5.12).

Εικόνα 5.12



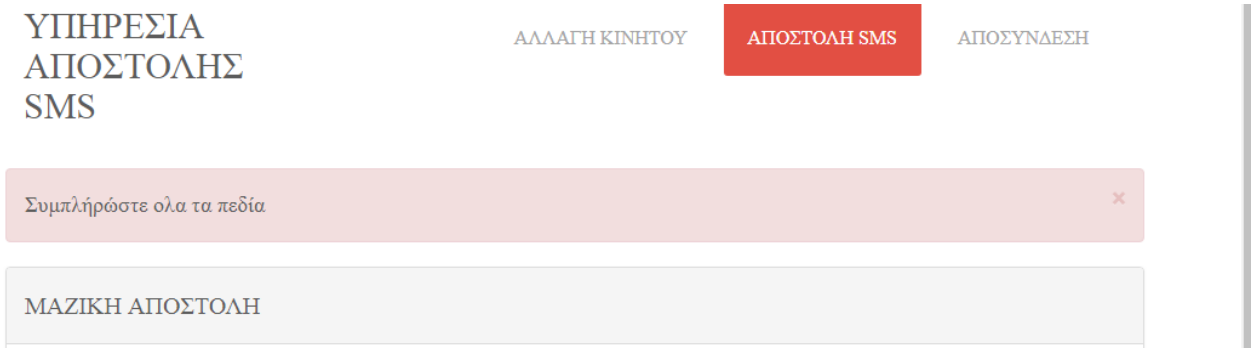
Επιλέγοντας στοιχεία από τα παραπάνω dropdown menu, εμφανίζονται δυναμικά τα πεδία που μπορεί να αλλάξει ο καθηγητής ώστε να στείλει το μήνυμα.(Εικόνα 5.13)

Εικόνα 5.13



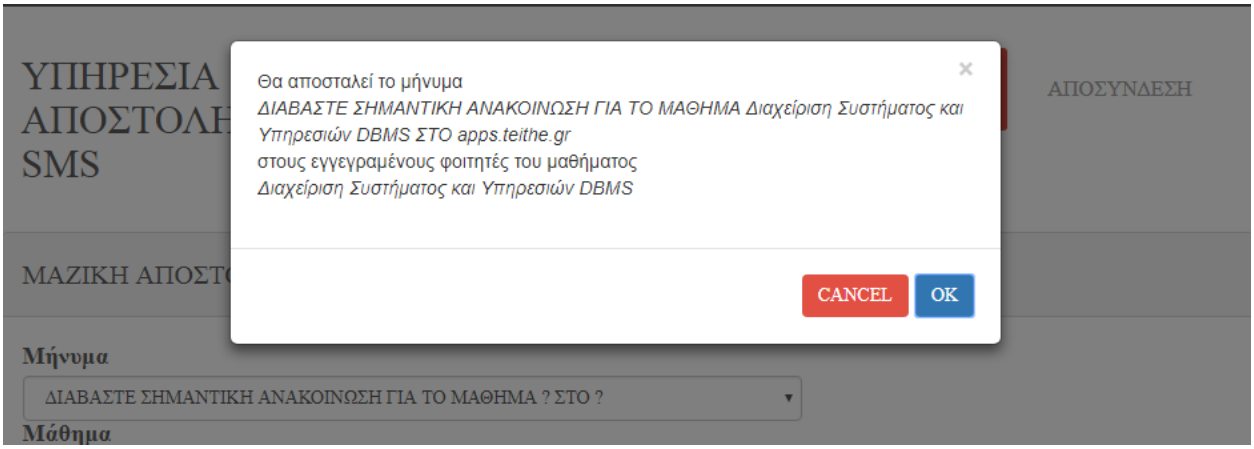
Πατώντας το κουμπί αποστολή, γίνεται έλεγχος για συμπλήρωση όλων των πεδίων (και εμφάνιση προειδοποιητικού μηνύματος εάν υπάρχει κάποιο σφάλμα(Εικόνα 5.14).

Εικόνα 5.14



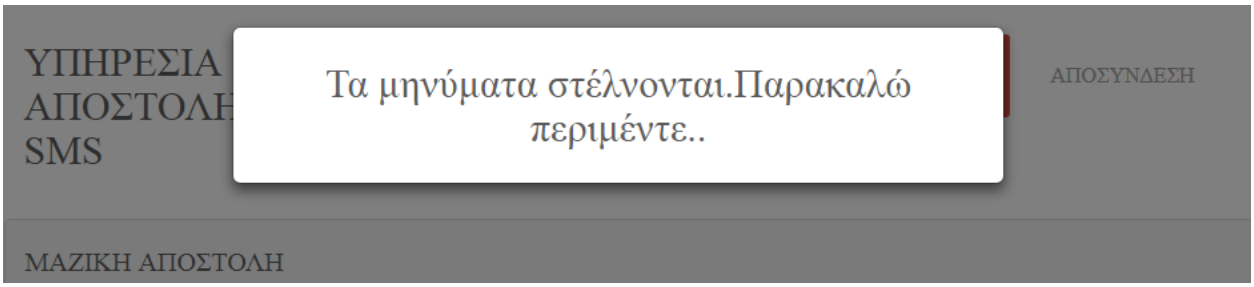
Εφόσον όλα είναι σωστά, αφού πατηθεί το κουμπί «Αποστολή μηνύματος» , εμφανίζεται ένα Alert επιβεβαίωσης(Εικόνα 5.15).

Εικόνα 5.15



Πατώντας το OK εμφανίζεται ένα loading screen μέχρι να ολοκληρωθεί η διαδικασία.(Εικόνα 5.16)

Εικόνα 5.16

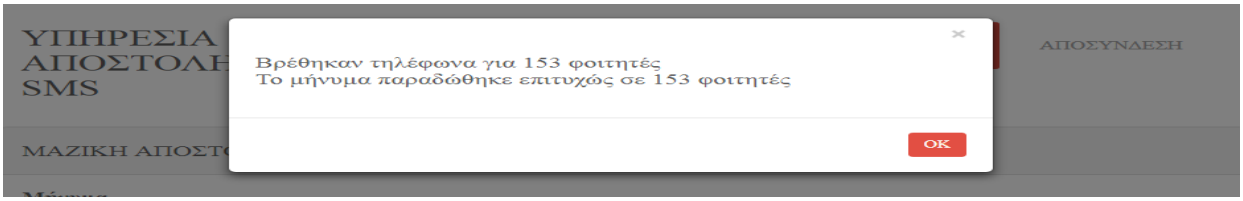


Όσο συμβαίνει αυτό, η ιστοσελίδα στέλνει το αίτημα αποστολής αυτών των μηνυμάτων στην εφαρμογή. Η εφαρμογή με την σειρά της, έχει να εκτελέσει τις εξής διαδικασίες:

- Εύρεση τηλεφώνων των μαθητών του επιλεγμένου μαθήματος, κάνοντας συνδυασμό μεταξύ της τοπικής βάσης, που μας παρέχει τα ενημερωμένα από τους φοιτητές τηλέφωνα, και τα δηλωμένα στην βάση της πυθίας τηλέφωνα.
- Αποστολή μηνυμάτων με παράλληλο τρόπο .
- Καταγραφή του αιτήματος και κατάστασης αποστολής των μηνυμάτων .

Εφόσον τελειώσει η εκτέλεση στην εφαρμογή, επιστρέφεται το αποτέλεσμα στον χρήστη (καθηγητή)(Εικόνα 5.17).

Εικόνα 5.17

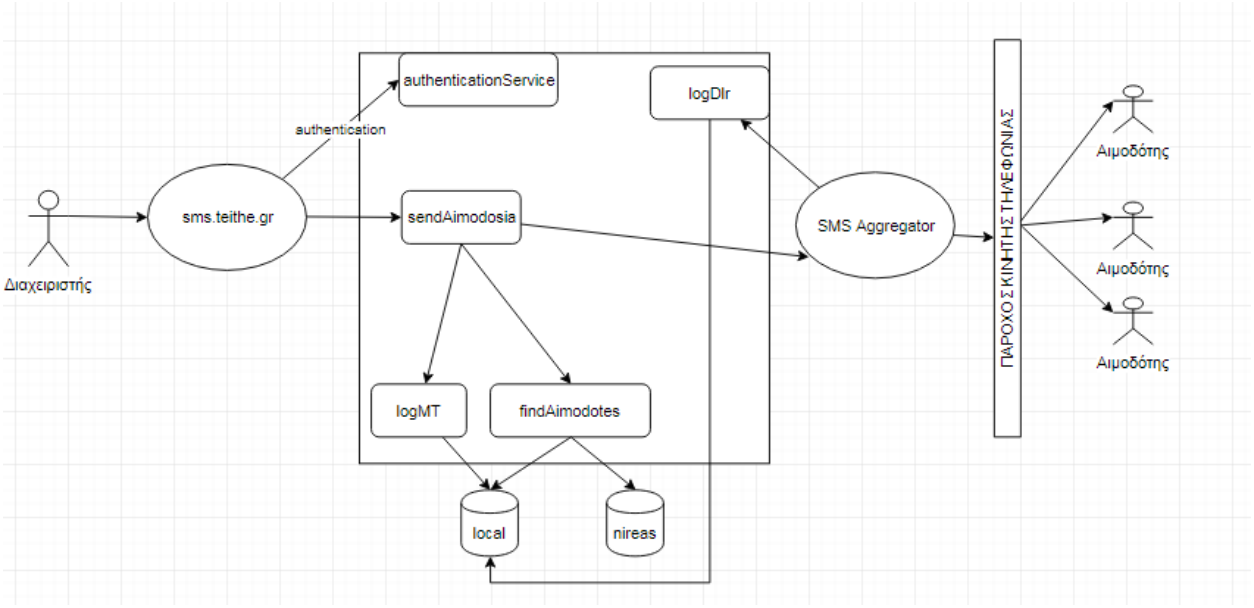


Στην συνέχεια, η εφαρμογή θα λάβει ξεχωριστά DLR για κάθε μήνυμα που στάλθηκε. Αυτά τα μηνύματα DLR αποθηκεύονται στην βάση μαζί με την καταγραφή της ενέργειας, ανά τακτά χρονικά διαστήματα.

5.3 : Ενημέρωση αιμοδοτών για νέες αιμοδοσίες

Η πρόσβαση στην υπηρεσία παρέχεται όνο στον διαχειριστή του κέντρου δικτύου του ιδρύματος (για λόγους ελέγχου χρήσης). Παρακάτω παρουσιάζεται το διάγραμμα χρήστης της υπηρεσίας(Εικόνα 5.18).

Εικόνα 5.18



Η διαδικασία αποστολής ξεκινάει με την είσοδο του χρήστη στην υπηρεσία μέσω της σελίδας login στην ιστοσελίδα. Η διαδικασία είναι ίδια όπως αυτή που ακολουθείται με τους καθηγητές, αλλά ο διαχειριστής πρέπει να χρησιμοποιήσει τα διαπιστευτήρια που έχει στον LDAP Server του ιδρύματος. Εφόσον ταυτοποιηθεί, θα του επιστραφεί αναλόγως το authentication token, και μπορεί πλέον να έχει πρόσβαση στην ειδική σελίδα αποστολής για αιμοδοσία(Εικόνα 5.19).

Εικόνα 5.19

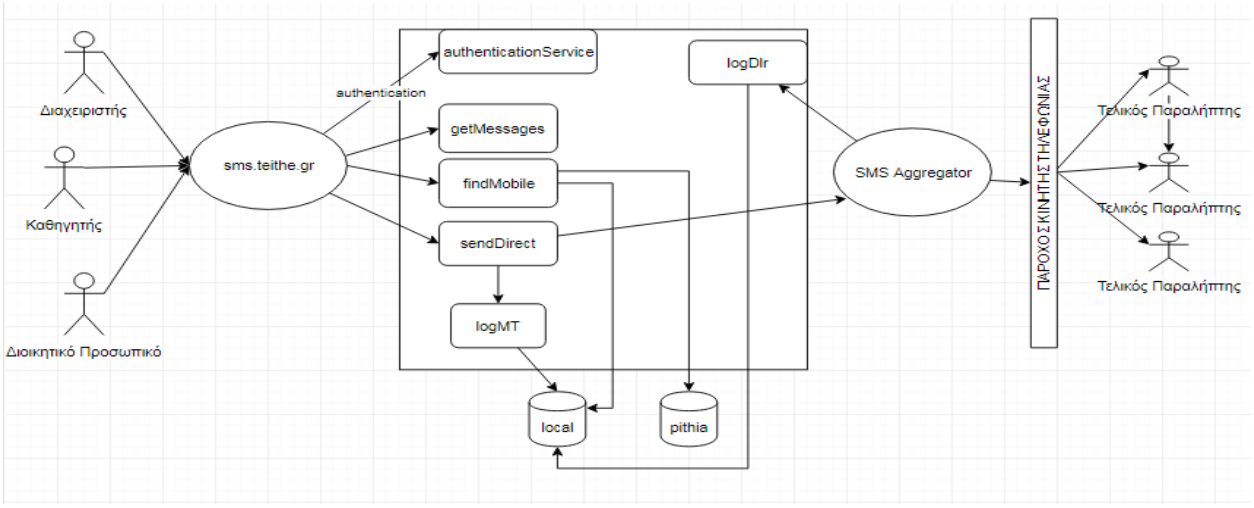
Εισάγοντας την ημερομηνία και πατώντας το κουμπί αποστολή, αποστέλλονται το ενημερωτικό μήνυμα προς τους εγγεγραμμένους χρήστες στην βάση τις αιμοδοσίας (nireas),με την προϋπόθεση ότι δεν έχουν δώσει αίμα τους 6 μήνες πριν από την ημερομηνία που εισάχθηκε, καθώς και στους χρήστες που έχουν εγγραφεί στα ενημερωτικά μηνύματα της αιμοδοσίας μέσω SMS, οι οποίοι είναι αποθηκευμένοι στην τοπική βάση της εφαρμογής

Η ενέργεια και τα ενημερωτικά DLR καταγράφονται με τον ίδιο τρόπο που αναφέρθηκε στην αποστολή ενημερωτικών μηνυμάτων σε φοιτητές.

5.4 : Στοχευμένη αποστολή SMS

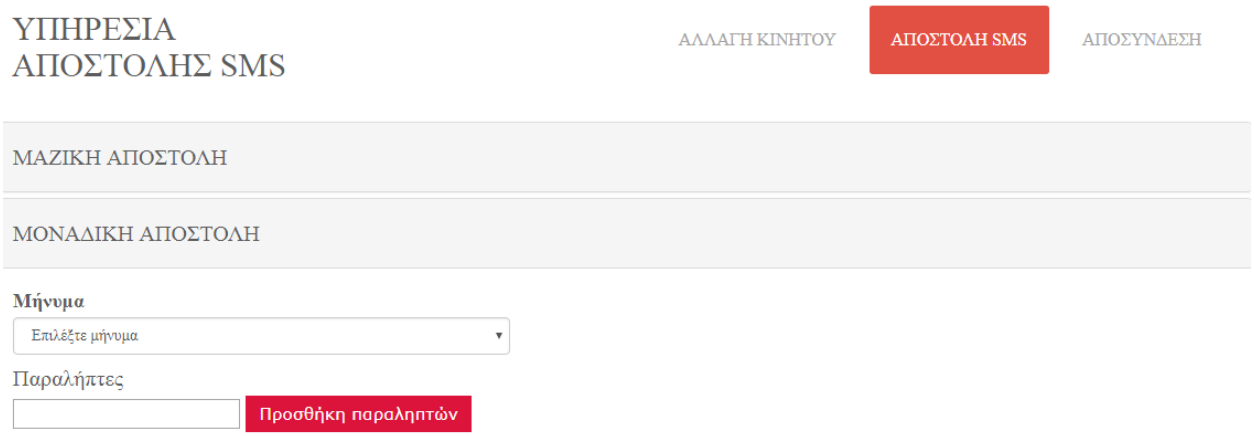
Πρόσβαση στην υπηρεσία αποστολής στοχευμένων μηνυμάτων έχουν ο διαχειριστής, οι καθηγητές, καθώς και τα μέλη του διοικητικού προσωπικού. Μέσω αυτής της υπηρεσίας μπορούν να σταλούν μηνύματα προς ορισμένους παραλήπτες, καθώς και να γίνει αναζήτηση σε καταχωρημένα τηλέφωνα μέσω της ιστοσελίδας για αποστολή σε αυτά. Παρακάτω παρουσιάζεται το διάγραμμα χρήσης της υπηρεσίας (Εικόνα 5.20)

Εικόνα 5.20



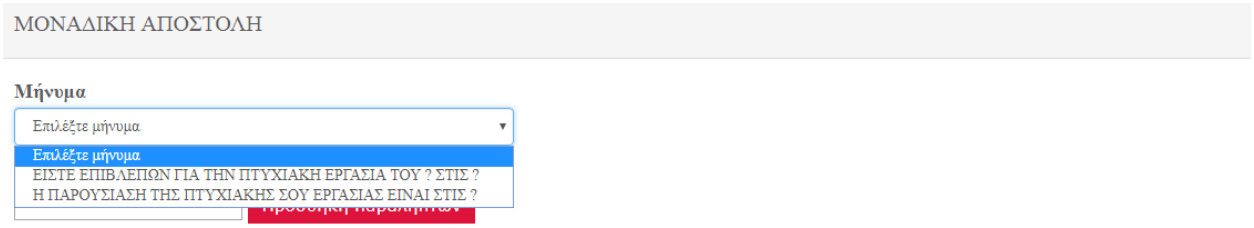
.Η διαδικασία ξεκινάει με την είσοδο και αυθεντικοποίηση του χρήστη στην ιστοσελίδα. Ο χρήστης μπορεί να κάνει είσοδο είτε με τα διαπιστευτήρια της Πυθίας (Καθηγητές) είτε μέσω LDAP(διαχειριστής και διοικητικό προσωπικό). Εφόσον είναι επιτυχής, μπορεί να έχει πρόσβαση στην σελίδα αποστολής μηνυμάτων (Εικόνα 5.21).

Εικόνα 5.21



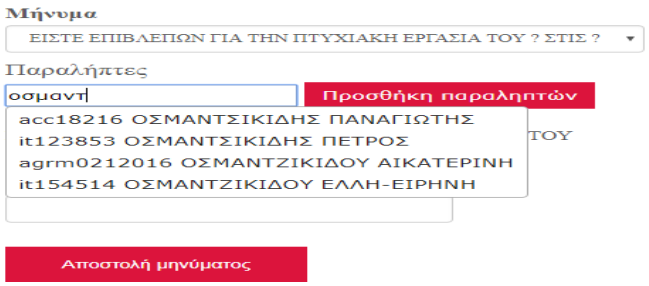
Η σελίδα στέλνει αίτημα στον server για να φορτώσει δυναμικά τα διαθέσιμα μηνύματα που μπορεί να στείλει ο χρήστης(Εικόνα 5.22).

Εικόνα 5.22



Ο χρήστης στην συνέχεια μπορεί να πληκτρολογήσει στο πεδίο των παραληπτών στοιχεία ώστε να αναζητήσει τα τηλέφωνα των παραληπτών στους οποίους θέλει να στείλει το επιλεγμένο μήνυμα.

Εικόνα 5.23

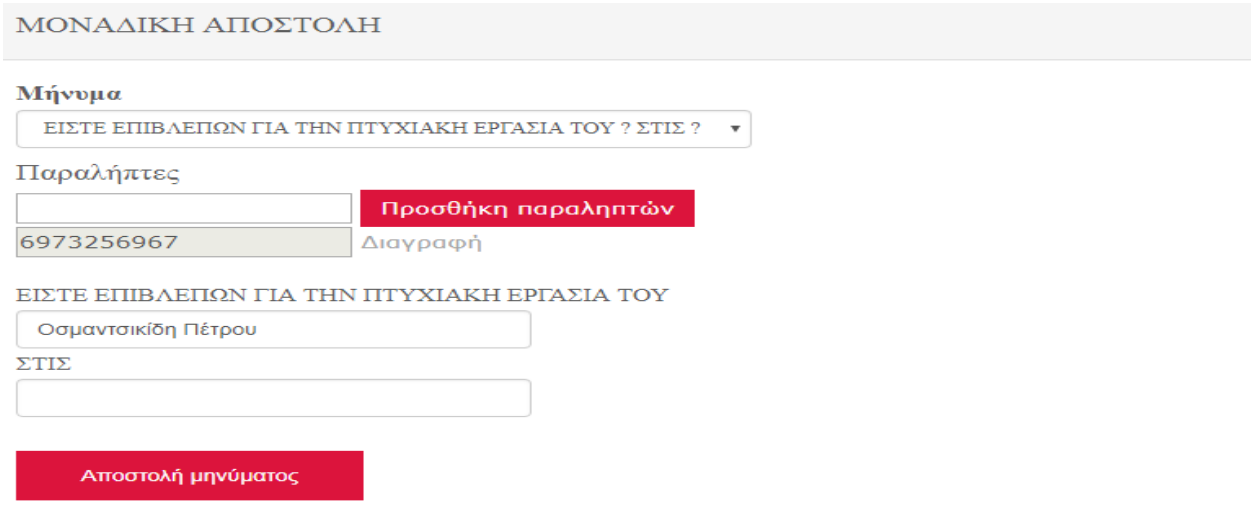


Η αναζήτηση μπορεί με τους εξής τρόπους:

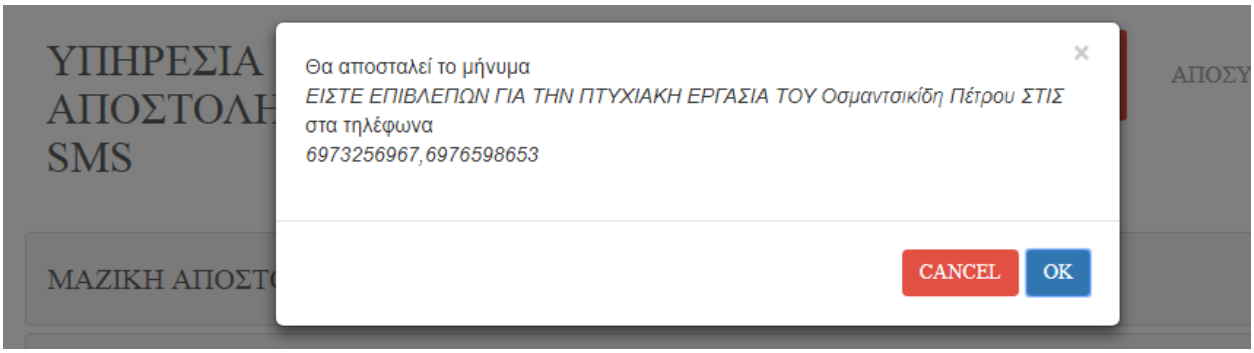
- Τηλέφωνο (όπου εμφανίζονται τα στοιχεία για επιβεβαίωση)
- User ID
- Ονοματεπώνυμο

Και γίνεται δυναμικά, σε κάθε χαρακτήρα που πληκτρολογείται. Η εφαρμογή κάνει αναζήτηση στην τοπική βάση και στην πυθία για τα τηλέφωνα, δίνοντας προτεραιότητα στα επιβεβαιωμένα της τοπικής βάσης. Πατώντας την κουμπί «Προσθήκη Παραληπτών» γίνεται προσθήκη του παραλήπτη στην λίστα αποστολέων(Εικόνα 5.24).Στην συνέχεια, πατώντας αποστολή εμφανίζεται ένα Alert για επιβεβαίωση του μηνύματος και των αποδεκτών(Εικόνα 5.25).

Εικόνα 5.24



Εικόνα 5.25

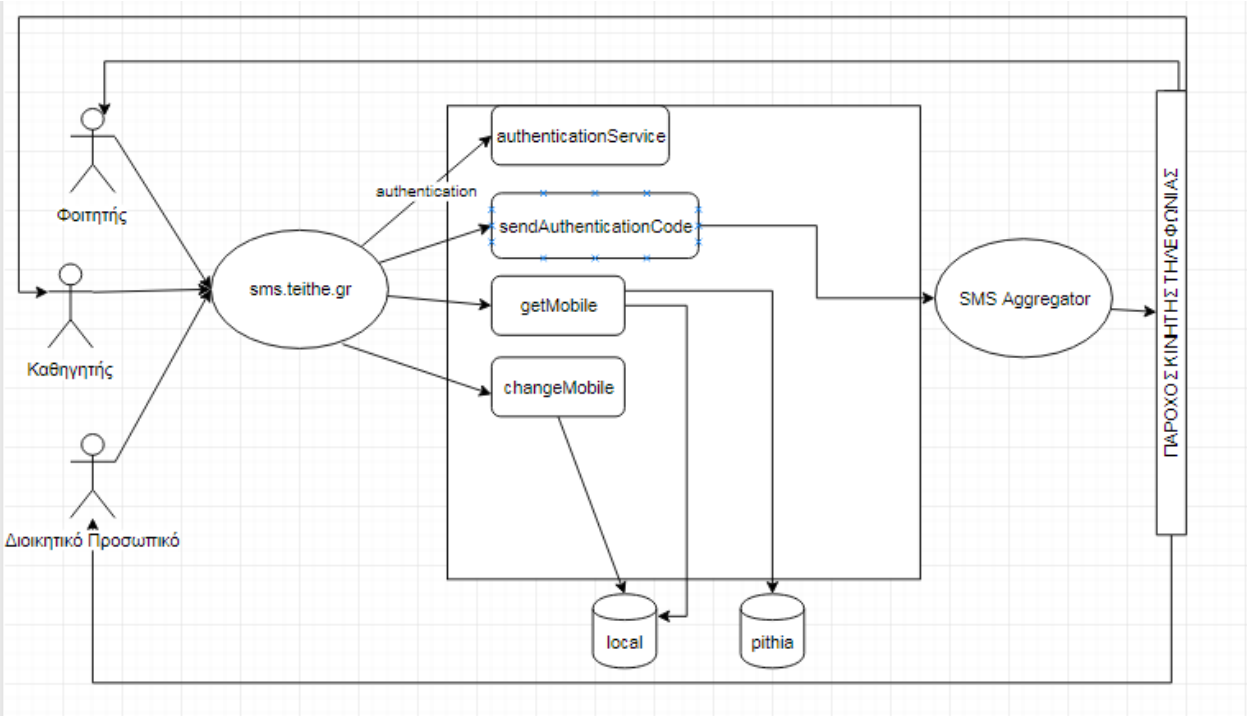


Η εφαρμογή στέλνει τα μηνύματα στους επιλεγμένους παραλήπτες, και επιστρέφεται στον χρήστη το αποτέλεσμα αυτής της ενέργειας. Στην συνέχεια καταγράφεται η συγκεκριμένη ενέργεια, καθώς και τα DLR όπως και στις προηγούμενες υπηρεσίες που παρουσιάστηκαν.

5.5 : Καταχώρηση / Ενημέρωση νούμερου κινητού τηλεφώνου

Όπως αναφέρθηκε στις προηγούμενες υπηρεσίες, χρησιμοποιείται η τοπική βάση της εφαρμογής για την αναζήτηση και εύρεση ταυτοποιημένων κινητών τηλεφώνων. Αυτό γίνεται λόγω μη ύπαρξης μερικών τηλεφώνων στην βάση στης πυθίας, καθώς και για την ενημέρωση των ήδη υπάρχοντών. Πρόσβαση για αλλαγή έχουν όλοι οι ταυτοποιημένοι χρήστες της εφαρμογής. Παρακάτω παρουσιάζεται το διάγραμμα χρήσης της υπηρεσίας(Εικόνα 5.26).

Εικόνα 5.26

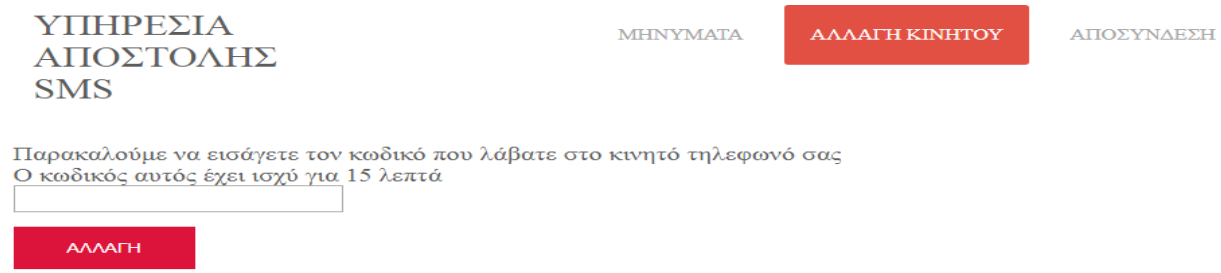


Η διαδικασία ξεκινά με την είσοδο και αυθεντικοποίηση μέσω της ιστοσελίδας. Ο χρήστης μπορεί να κάνει είσοδο είτε μέσω Πυθίας είτε μέσω LDAP. Εφόσον είναι επιτυχής, μπορεί να μεταβεί στην σελίδα αλλαγής κινητού τηλεφώνου(Εικόνα 5.27).

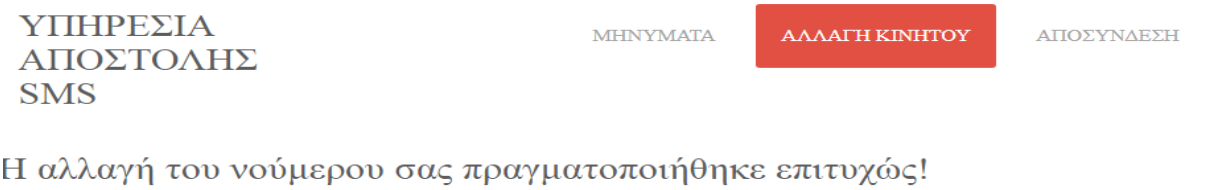
Εικόνα 5.27

Κατά την πλοήγηση σε αυτήν την σελίδα, γίνεται ανάκτηση του τρέχοντος δηλωμένου κινητού του τηλεφώνου στην τοπική βάση, και αν δεν βρεθεί, στην βάση της πυθίας. Εφόσον ο χρήστης δηλώσει κάποιο 10-ψηφιο νούμερο, ο μπορεί να πατήσει «Αποστολή μηνύματος επιβεβαίωσης» για να του αποσταλεί ένας τυχαίος κωδικός επιβεβαίωσης για να γίνει η αλλαγή(Εικόνα 5.28), και θα του αποσταλεί ο κωδικός επιβεβαίωσης στο κινητό που δηλώθηκε στην προηγούμενη σελίδα. Πατώντας αλλαγή, ελέγχεται η εγκυρότητα αυτού του κωδικού και αναλόγως καταχωρεί τον ανανεωμένο αριθμό κινητού στην τοπική βάση, και επιστρέφεται στην ιστοσελίδα το αποτέλεσμα της ενέργειας(Εικόνα 5.29).

Εικόνα 5.28



Εικόνα 5.29

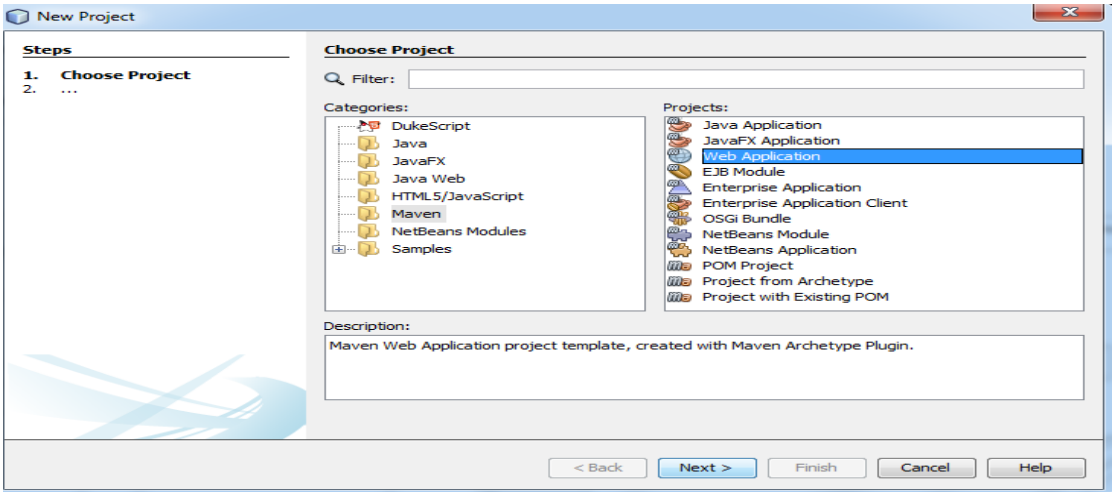


Κεφάλαιο 6 : Τεκμηρίωση, Συντήρηση, Επεκτασιμότητα

6.1 : Ρυθμίσεις για χρήση Java ως backend

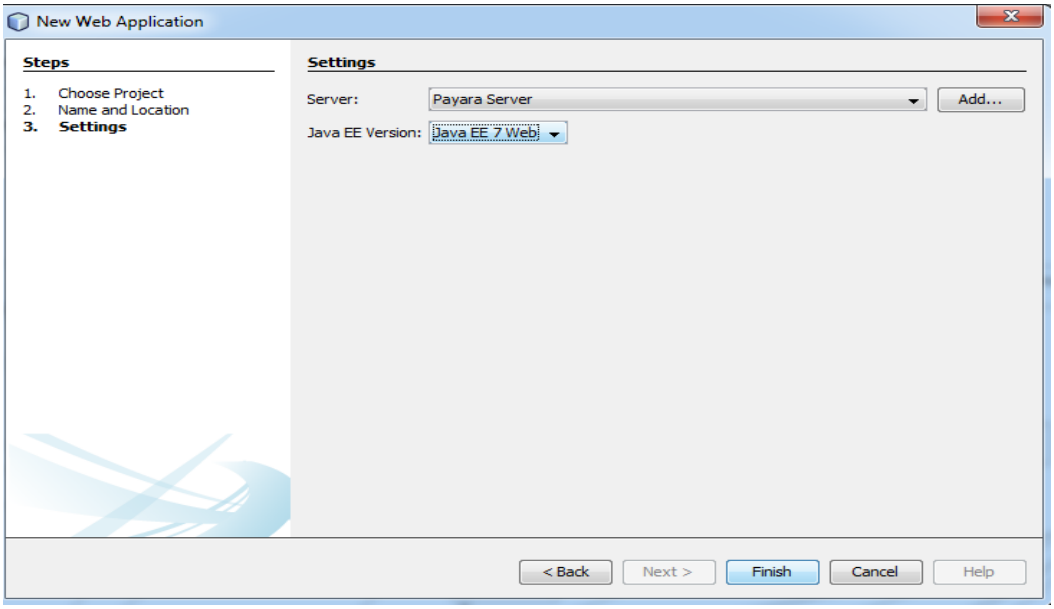
Για να χρησιμοποιηθεί η Java ως backend εφαρμογή, πρέπει να γίνουν κάποιες ρυθμίσεις στην παραγωγή του εκτελέσιμου αρχείου. Αρχικά, χρησιμοποιώντας οποιοδήποτε IDE, πρέπει να οριστεί το project ως 'Web Application'. Επειδή πρόκειται να χρησιμοποιηθεί το Maven, θα πρέπει να οριστεί και αυτό στις αρχικές ρυθμίσεις του project(Εικόνα 6.1).

Εικόνα 6.1



Στην συνέχεια θα ζητηθεί να οριστεί και ένας application server(Εικόνα 6.2) πάνω στον οποίο θα τρέξει το η εφαρμογή, καθώς και η έκδοση Java EE που θα χρησιμοποιηθεί.(Η οποία θα ρυθμιστεί και στο configuration file του Maven).

Εικόνα 6.2



Μετά από αυτό, πριν από την δημιουργία των Java κλάσεων που ουσιαστικά θα υλοποιούν τα webservices της εφαρμογής, πρέπει να οριστεί ένα deployment descriptor, που ορίζεται από το αρχείο web.xml και πρέπει να βρίσκεται στον φάκελο \WebContent\WEB-INF(Εικόνα 6.3).

Εικόνα 6.3

```
<web-app xmlns="http://xmlns.jcp.org/xml/ns/javaee" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee
                http://xmlns.jcp.org/xml/ns/javaee/web-app_3_1.xsd"
        version="3.1">

        <display-name>Archetype Created Web Application</display-name>

        <servlet>
                <servlet-name>jersey-serlvet</servlet-name>
                <servlet-class>org.glassfish.jersey.servlet.ServletContainer</servlet-class>
                <init-param>
                        <param-name>jersey.config.server.provider.packages</param-name>
                        <param-value>com.smsserver</param-value>
                </init-param>
                <load-on-startup>1</load-on-startup>
        </servlet>

        <servlet-mapping>
                <servlet-name>jersey-serlvet</servlet-name>
                <url-pattern>/service/*</url-pattern>
        </servlet-mapping>
```

Η πρώτη ρύθμιση του servlet, αφορά τα πακέτα μέσα στα οποία θα ψάχνει η Java για να αντιστοιχήσει τα διάφορα webservices. Στην συγκεκριμένη περίπτωση είναι όλο το πακέτο του *com.smsserver.*
Η δεύτερη ρύθμιση για το servlet-mapping, αφορά την τοποθεσία αυτών των webservices. Όπως φαίνεται και στην εικόνα, είναι ρυθμισμένο να τα παρουσιάζει στην διεύθυνση */service/* .

Αυτές είναι οι μοναδικές ρυθμίσεις που χρειάζονται για την λειτουργιά της εφαρμογής ως backend. Μέσω annotations στον κώδικα (όπως θα παρουσιαστούν παρακάτω σε παραδείγματα υλοποίησης) μπορούν πολύ γρήγορα και χωρίς περεταίρω ρυθμίσεις να δημιουργηθούν webservices που πληρούν τις απαραίτητες απαιτήσεις της εφαρμογής.

6.2 : Dependencies

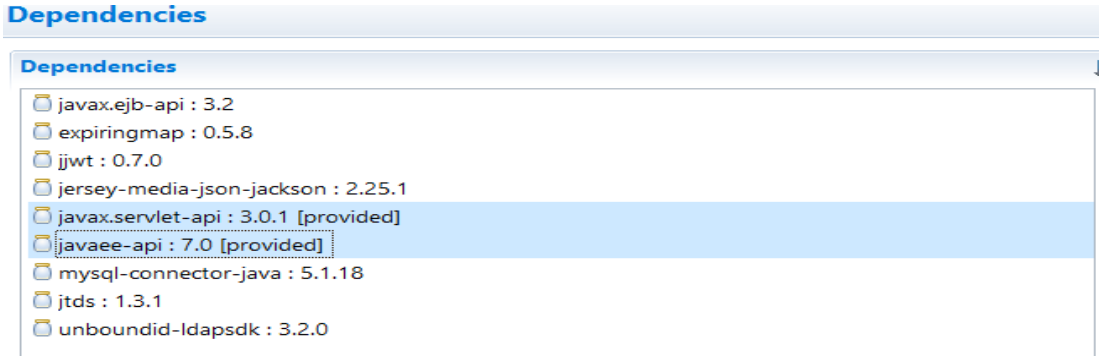
Επειδή η εφαρμογή στηρίζεται στο Maven, η προσθήκη και η διαχείριση βιβλιοθηκών Java αναλαμβάνεται εξ ολοκλήρου από αυτό. Αυτό γίνεται με την μορφή dependencies(εξαρτήσεων), που σημαίνει ότι απλά προσθέτοντας ένα dependency στο αρχείο ρυθμίσεων του Maven(pom.xml), το Maven κατεβάζει την βιβλιοθήκη μαζί με οποιαδήποτε άλλη χρειάζεται για να λειτουργήσει. Επειδή μεγαλώνοντας η εφαρμογή χρειάζονται παραπάνω εξαρτήσεις από βιβλιοθήκες, υπάρχει περίπτωση μερικές βιβλιοθήκες να συμπίπτουν και να δημιουργήσουν προβλήματα συμβατότητας από διάφορα component που τις χρησιμοποιούν. Αυτό το πρόβλημα το διαχειρίζεται άμεσα το Maven, καθώς μπορεί να λύνει τέτοιου είδους conflict.

Ένα Maven dependency ως συνήθως μπορεί να αντιστοιχιστεί μέσω ενός link στο επίσημο repository του Maven, από όπου και μπορεί να ανακτηθεί οποιαδήποτε στιγμή από οποιοδήποτε μηχάνημα. Υπάρχει δυνατότητα όμως, να οριστεί και κάποιο dependency σε κάποιο τοπικό αρχείο. Παρακάτω φαίνεται ένα παράδειγμα προσθήκης dependency από το repository του Maven στο pom.xml και οι εξαρτήσεις που χρησιμοποιεί η εφαρμογή(Εικόνα 6.4,Εικόνα 6.5).

Εικόνα 6.4

```
<dependency>
        <groupId>org.glassfish.jersey.media</groupId>
        <artifactId>jersey-media-json-jackson</artifactId>
        <version>2.25.1</version>
</dependency>
```

Εικόνα 6.5



Αναλυτικά τα dependencies που χρησιμοποιεί η εφαρμογή:

- javax.ejb-api: Παρέχει τις EJB λειτουργίες της εφαρμογής. Παρόλο που οι περισσότερες από αυτές παρέχονται και από τον application server, δεν είναι οι πιο πρόσφατες και δεν κάλυπταν όλες τις ανάγκες της εφαρμογής, οπότε χρησιμοποιείται η εξωτερική βιβλιοθήκη.
- expiringmap: Είναι μια custom βιβλιοθήκη η οποία υλοποιεί με thread-safe τρόπο το ConcurrentMap της Java. Χρησιμοποιείται από την εφαρμογή για την διαγραφή στοιχείων από κάποιον πίνακα ύστερα από κάποιο timeout που ορίζουμε, και για τον ορισμό πινάκων με μέγιστο μέγεθος και αντικατάσταση του παλαιότερου στοιχείου.
- jjwt: Η βιβλιοθήκη JSON Web Token παρέχει την δυνατότητα να δημιουθούν hashed web token για αυθεντικοποίηση, με δυνατότητα προσθήκης στοιχείων (όπως username, role, timeout) και γρήγορο έλεγχο αυτών των στοιχείων κατά τους διάφορους ελέγχους ταυτοποίησης της εφαρμογής. Η συγκεκριμένη βιβλιοθήκη χρειάζεται και το dependency του Jackson για να λειτουργήσει, κάτι που λύνεται αυτόματα από το Maven.
- jersey-media-json-jackson: Το συγκεκριμένο dependency περιλαμβάνει αρκετές βιβλιοθήκες που θα χρησιμοποιηθούν στην εφαρμογή. Χρησιμοποιείται αντί για του MOXy που είναι ο default τρόπος για μετατροπή αρχείων JSON σε και από Java objects, λόγω των λειτουργιών που υποστηρίζει (απόκρυψη και μετονομασία πεδίων κατά την μετατροπή). Επίσης είναι απαραίτητη όπως αναφέρθηκε παραπάνω για την λειτουργία του JJWT.
- mysqlconnector-java: Χρησιμοποιείται για την διασύνδεση JDBC με τις MySQL βάσεις
- jdts: Είναι μια open source JDBC βιβλιοθήκη διασύνδεσης Java με MS-SQL Servers. Προτιμήθηκε αντί της επίσημης βιβλιοθήκης που μας παρέχεται από την Microsoft καθώς εκείνη, καθώς δεν είναι ανοιχτού λογισμικού δεν παρέχεται από το Maven.
- unboundid-ldap: Χρησιμοποιείται για την διασύνδεση στον LDAP Server, που είναι αναγκαία για ταυτοποίηση των χρηστών μέσω εκείνης της υπηρεσίας. Προσφέρει αρκετά πλεονεκτήματα σε σχέση με την υλοποίηση της Java, καθώς προσφέρει ένα δυνατό και εύχρηστο API χρήσης που καλύπτει όλες τις ανάγκες της εφαρμογής, καθώς έχει υλοποιημένη υποστήριξη multi-threading και connection-pooling.

Στο αρχείο ρυθμίσεων πρέπει να προστεθούν και δύο επιπλέον dependencies

- javax.servlet-api
- javaee-api

Παρόλο που ο application server παρέχει τα συγκεκριμένα dependencies, την ώρα του build όπου ουσιαστικά γίνεται compile και test η εφαρμογή, αυτά είναι απαραίτητα. Χρησιμοποιώντας το tag <provided> το Maven θεώρει πως υπάρχουν, και εκτελεί φυσιολογικά το build θεωρώντας όμως ότι θα υπάρχουν στον server στον οποίο θα εκτελεστεί η εφαρμογή.

6.3 : Το API της εφαρμογής

Το API που παρέχεται από την εφαρμογή χρησιμοποιείται για κλήσεις από τον SMS Aggregator, καθώς και από την ιστοσελίδα η οποία χρησιμοποιεί λειτουργίες της εφαρμογής. Αναλόγως την κρισιμότητα της

εκάστοτε ενέργειας, τα διάφορα endpoint είναι προστατευμένα για εξουσιοδοτημένη χρήση από τους εξουσιοδοτημένους χρήστες.

Στον Πίνακας 6.1 παρουσιάζονται όλα τα endpoint που παρέχονται. Στα endpoint τα οποία έχουν σημειωθεί να είναι 'Secured', πρέπει να αποστέλλεται μαζί με το αίτημα το authentication token το οποίο δημιουργείται κατά την αυθεντικοποίηση ενός χρήστη μέσω του endpoint 'sms/authentication'. Ο τρόπος που γίνεται αυτό είναι προσθέτοντας στο header του αιτήματος το παρακάτω πεδία:

- Key:Authorization
- Value:Bearer <token>

Σε περίπτωση που δεν υπάρχει το συγκεκριμένο header σε ένα secured μήνυμα, θα επιστραφεί ως response στο αίτημα το μήνυμα '401 Unauthorized'. Παρακάτω, παρουσιάζονται ξεχωριστά ανα endpoint τα διάφορα συγκεκριμένα headers που χρειάζονται για κάθε μοναδικό endpoint.

6.3.1 : Συνοπτικά

Πίνακας 6.1

Endpoint	Type	Secured	Role	Χρήση
sms/service	POST	Όχι	-	Χρησιμοποιείται από τον SMS Aggregator για προώθηση MO μηνυμάτων χρηστών
sms/service/dlr/	POST	Όχι	-	Χρησιμοποιείται από τον SMS Aggregator για αποστολή των Delivery Report
sms/authentication	POST	Όχι	-	Ταυτοποιεί τον χρήστη για χρήση της ιστοσελίδας
sms/site/moservices	GET	Όχι	-	Ανάκτηση των περιγραφών των MO υπηρεσιών που υποστηρίζονται
sms/site/mtservices/moodle	GET	Ναι	Staff,Admin	Ανάκτηση των μαζικών MT υπηρεσιών που υποστηρίζονται
sms/site/mtservices/direct	GET	Ναι	Staff,Admin	Ανάκτηση των direct MT υπηρεσιών που υποστηρίζονται
sms/site/courses	GET	Ναι	Staff,Admin	Ανάκτηση των μαθημάτων που διδάσκει κάθε καθηγητής

sms/site/users/{search}	GET	Ναι	Staff,Admin	Ανάκτηση τηλεφώνων χρηστών με βάση την παράμετρο search
sms/site/mobile/	GET	Ναι	ALL	Ανάκτηση καταχωρημένου τηλεφώνου ενός χρήστη
sms/site/mobile/	PUT	Ναι	ALL	Αλλαγή/Καταχώρηση τηλεφώνου του χρήστη
sms/site/mobile /sendConfirmation/{mobile}	GET	Ναι	ALL	Αποστολή κωδικού επαλήθευσης σε χρήστη μετα από αίτημα αλλαγής τηλεφώνου
sms/send/moodle	POST	Ναι	Staff,Admin	Αποστολή μαζικών μηνυμάτων σε κάποιο επιλεγμένο μάθημα
sms/send/direct	POST	Ναι	Staff,Admin	Αποστολή direct μηνυμάτων σε επιλεγμένα τηλέφωνα
sms/send/aimodosia	POST	Ναι	Admin	Αποστολή μαζικών μηνυμάτων σε αιμοδότες

6.3.2 : Endpoint SMS Aggregator

Sms/service

Χρήση : Χρησιμοποιείται από τον SMS Aggregator για προώθηση MO μηνυμάτων χρηστών. Το συγκεκριμένο endpoint ορίζεται μέσω του διαχειριστικού του SMS Aggregator για να λαμβάνει τα μηνύματα που αφορούν το ΑΤΕΙ/Θ, τα οποία προωθούνται σε αυτό.

Type: POST

Headers:*Content-Type = application/json; charset=utf-8*

Consumes : Json	Produces : Json
"MSISDN": Το τηλέφωνο του χρήστη "keyword": Το keyword της υπηρεσίας "body": Το υπόλοιπο μήνυμα που στάλθηκε "pre-shared key": Το προκαθορισμένο κλειδί ασφαλείας της υπηρεσίας “forward-id”:Ο αριθμός προώθησης του MO μηνύματος	“result”: Το αποτέλεσμα της ενέργειας(0 ή 1) «errorCode”: Το id του error “error”:Περιγραφή του error

Sms/Service/Dlr

Χρήση : Χρησιμοποιείται από τον SMS Aggregator για αποστολή των Delivery Report. Η διευθυνση του συγκεκριμένου endpoint συμπεριλαμβάνεται στο πεδίο dlr-url κατά την κλήση της αποστολής μηνυμάτων προς το GUNET.

Type: POST
Headers:
Content-Type = application/json; charset=utf-8

Consumes : JSON	Produces
“serviceId”:Το ID της υπηρεσίας που κλήθηκε «recipient”:Ο χρήστης στον οποίο στάλθηκε το μήνυμα «status”:Η κατάσταση του μηνύματος «error”: Ο κωδικός αριθμός της κατάστασης	204 No content

6.3.3 : Endpoint ιστοσελίδας

Sms/Authentication

Χρήση : Ταυτοποιεί τον χρήστη για χρήση της ιστοσελίδας. Επιστρέφει τον ρόλο και το authentication token του χρήστη, σύμφωνα με τα οποία θα μπορεί να χρησιμοποιήσει τις @Secured υπηρεσίες που παρέχει το API.

Type: POST
Headers:
Content-Type = application/json; charset=utf-8

Consumes : JSON	Produces : JSON
“username”: Το όνομα του χρήστη «password”: Ο κωδικός του χρήστη	«username”: Το όνομα του χρήστη “role”: Ο ρόλος του χρήστη “token”:Το token με το οποίο θα χρησιμοποιεί τα secure endpoint. Το συγκεκριμένο token γίνεται αντιστοίχιση στην εφαρμογή για την προέλευση κάθε περαιτέρω κλήσης “confirmedPhone”:0 ή 1, αν ο χρήστης έχει επιβεβαιωμένο τηλέφωνο στην υπηρεσία

Sms/Site/moservices

Χρήση : Ανάκτηση των περιγραφών των MO υπηρεσιών που υποστηρίζονται. Είναι ελεύθερο προς χρήση και επιστρέφει αυτόματα όλα τα διαθέσιμα μηνύματα ερωτήσεων που έχουν καθοριστεί στην εφαρμογή.

Type: GET

Produces : JSON List
“keyword”:Το keyword της κάθε υπηρεσίας “description”: Η περιγραφή κάθε υπηρεσίας

sms/site/mtservices/moodle

Χρήση : Ανάκτηση των μαζικών MT υπηρεσιών που υποστηρίζονται. Είναι ελεύθερο προς χρήση και επιστρέφει αυτόματα όλα τα διαθέσιμα μηνύματα για μαζική αποστολή που έχουν καθοριστεί στην εφαρμογή.

Type: GET

Produces : JSON List
“message”:Ο κωδικός του MT μηνύματος “description”: Η περιγραφή κάθε μηνύματος

sms/site/mtservices/direct

Χρήση : Ανάκτηση των direct MT υπηρεσιών που υποστηρίζονται. Είναι ελεύθερο προς χρήση και επιστρέφει αυτόματα όλες τις διαθέσιμα μηνύματα για στοχευμένες αποστολές που έχουν καθοριστεί στην εφαρμογή.

Type: GET

Secured : Staff , Admin

Produces : JSON List
“message”:Ο κωδικός του MT μηνύματος “description”: Η περιγραφή κάθε μηνύματος

sms/site/courses

Χρήση : Ανάκτηση των μαθημάτων που διδάσκει κάθε καθηγητής. Επιστρέφει μία λίστα από τα μαθήματα που διδάσκει ο καθηγητής το τρέχον ακαδημαϊκό εξάμηνο και έχουν τουλάχιστον 1 εγγεγραμμένο μαθητή.

Type: GET
Secured: Staff , Admin

Produces : JSON list
LIST: “courseName ”:Ο κωδικός του κάθε μαθήματος “courseId ”: Η περιγραφή κάθε μαθήματος

sms/site/users/{search}

Χρήση : Ανάκτηση τηλεφώνων χρηστών με βάση την παράμετρο search. Η παράμετρος search μπορεί να είναι το ονοματεπώνυμο, το userID του χρήστη στην βάση της πυθίας, ή το κινητό του χρήστη.

Type: GET
Secured : Staff , Admin

Produces : JSON List
LIST: “name ”: Το username κάθε χρήστη που πληρεί την αναζήτηση “mobile”: Το τηλέφωνο του “fullname”: Το ονοματεπωνυμό του

sms/site/mobile

Χρήση : Ανάκτηση καταχωρημένου τηλεφώνου ενός χρήστη. Επιστρέφεται το καταχωρημένο νούμερο κινητού τηλεφώνου του χρήστη που καλεί το endpoint.

Type: GET
Secured : ALL

Produces: String – Το νούμερο του κινητού τηλεφώνου

sms/site/mobile

Χρήση : Αλλαγή/Καταχώρηση τηλεφώνου του χρήστη. Μέσω του endpoint στέλνεται το verificationCode που έχει λάβει ο χρήστης κατά την αίτηση αλλαγή του τηλεφώνου, και αν είναι επιτυχής πραγματοποιείται η ανανέωση που ζητήθηκε.

Type: PUT
Headers:
Content-Type = application/json; charset=utf-8
Secured : ALL

Consumes : JSON	Produces
"verification": Ο κωδικός επαλήθευσης που έχει σταλεί στον χρήστη για επιβεβαίωση	200 OK 417 EXPECTATION FAILED

sms/site/mobile/sendConfirmation/{mobile}

Χρήση : Αποστολή κωδικού επαλήθευσης σε χρήστη μετα από αίτημα αλλαγής τηλεφώνου. Επιστρέφεται ένα verification code που θα χρησιμοποιηθεί για την αλλαγή του κινητού τηλεφώνου του χρήστη.

Type: GET
Secured: ALL

Consumes: mobile – το κινητό στο οποίο θα σταλεί ο κωδικός επιβεβαίωσης

Produces: 204 NO CONTENT

Sms/Send/Moodle

Χρήση : Αποστολή μαζικών μηνυμάτων σε κάποιο επιλεγμένο μάθημα. Ο χρήστης που καλεί αυτό το endpoint μπορεί να στείλει το επιλεγμένο μήνυμα στους μαθητές του course που έχει επιλεχθεί. Του επιστρέφεται το αποτέλεσμα της ενέργειας.

Type: POST
Headers:
Content-Type = application/json; charset=utf-8
Secured : Staff , Admin

Consumes : JSON	Produces : JSON
"course": Το ID του μαθήματος "messageId":Το ID του μηνύματος «replacements"(Array): Τα πεδία αντικατάστασης του μηνύματος	"send": Ο αριθμός των παραληπτών στους οποίους στάλθηκε το μήνυμα "delivered": Ο αριθμός των παραληπτών στους οποίους παραδόθηκε το μήνυμα

Sms/Send/Direct

Χρήση : Αποστολή direct μηνυμάτων σε επιλεγμένα τηλέφωνα. Ο χρήστης μπορεί να στείλει το μήνυμα που έχει επιλέξει στην λίστα παραληπτών που δηλώνεται στο array Recipients.

Type: POST
Headers:
Content-Type = application/json; charset=utf-8
Secured : Staff , Admin

Consumes : JSON	Produces : JSON
"recipients"(Array):Λίστα παραληπτών για το μήνυμα "messageId":Το ID του μηνύματος «replacements"(Array): Τα πεδία αντικατάστασης του μηνύματος	"send": Ο αριθμός των παραληπτών στους οποίους στάλθηκε το μήνυμα "delivered": Ο αριθμός των παραληπτών στους οποίους παραδόθηκε το μήνυμα

Sms/Send/Aimodosia

Χρήση : Αποστολή μαζικών μηνυμάτων σε αιμοδότες.Είναι προσβάσιμο μόνο από τον διαχειριστή της υπηρεσίας, και στέλνει αυτόματα σε όλους τους δηλωμένους εθελοντές αιμοδότες που έχουν καταχωρήσει αριθμό κινητού τηλεφώνου.

Type: POST
Headers:
Content-Type: text/plain
Secured : Admin

Consumes : String	Produces : JSON
"Date": Η ημερομηνία της επόμενης αιμοδοσίας	"send": Ο αριθμός των παραληπτών στους οποίους στάλθηκε το μήνυμα "delivered": Ο αριθμός των παραληπτών στους οποίους παραδόθηκε το μήνυμα

6.4 : Υλοποίηση διασύνδεσης με SMS Aggregator

6.4.1 : Αποστολή SMS

Η send SMS μέθοδος παρέχεται από την πλατφόρμα του GUNET Aggregator και γίνεται consume από τα Πανεπιστημιακά Ιδρύματα. Το endpoint της υπηρεσίας είναι το https://sms-services.gunet.gr:9999/sendSMS.
Το request πρέπει να είναι POST και το Content Type: application/json . Ακολουθεί ένα παράδειγμα μιας κλήσης προς την δοκιμαστική υπηρεσία gradeService.

```
{
  "serviceId": "gradeService",
  "messageId": "testMessage",
  "replacements": [
    "Προγραμματισμός Ι",
    "7"
  ],
  "recipient": "6901234567",
  "institution": "TEITHE",
  "pre-shared key": "F0fesFADSr223fA",
  "dlr-url": "https://teithe.gr/dlrs",
  "sms-forward-id": "230"
}
```

Ακολουθούν και οι ενδεικτικές απαντήσεις του SMS Aggregator στο παραπάνω request

Επιτυχές:

```
{
  "serviceId": "gradeService",
  "errorCode": "",
  "error": ""
} Ανεπιτυχές:
```

```
{
   "serviceId": "gradeService",
   "errorCode": "E-002",
   "error": "Unknown Service"
}
```

Για αυτόν τον λόγο, πρέπει να δημιουργηθούν δύο Java κλάσεις, μία για την παραγωγή του JSON που στέλνεται στο endpoint της εφαρμογή(Εικόνα 6.6), και μία για την απάντηση που δέχεται(Εικόνα 6.7). Αυτές είναι απλές POJO κλάσεις, οι όποιες μετατρέπονται σε JSON με την βοήθεια της βιβλιοθήκης Jackson.

Κλάση SendSmsModel

Εικόνα 6.6

```
@XmlRootElement
public class SendSmsModel {

        private String serviceId;// id tis ipiresias pou periexei to to
                                 // prokathorismeno minima pros apostoli
        private String messageId;// to id toy minimatos
        private String[] replacements; // se periptwsi p thelei array
        private String recipient;// username i msisdn toy xristi
        private final String institution = "TEITHE";// ypoxrewtiko gia teithe
        private final String preSharedKey = GetPropertyValues.getProperties().getProperty("preSharedKey");
        private final String dlrUrl = GetPropertyValues.getProperties().getProperty("dlrUrl");

        private String smsForwardId;// efoson ginei se dinexeia enos forward
                                                    // request-proeraitiko
```

Εικόνα 6.7

```
@XmlRootElement
public class SmsResponseModel {

        private String serviceId;// to id tis ipiresias poy klithike
        private String errorCode;// to id toy error
        private String error;// to minima lathous unknown service, unknown message, unknown
                                // recipient, unknown institution, invalid pre-shared key,
                                // unauthorized sender address, user not opted in, user
                                // deactivated service, user credits expired, service
                                // deactivated, general error etc.
```

Τα properties preSharedKey και dlrUrl διαβάζονται και ορίζονται από το αρχείο config.properties στην αρχική εκτέλεση της εφαρμογής, έτσι ώστε να μπορούν να παραμετροποιηθούν χωρίς την αλλαγή του πηγαίου κώδικα.

Παρατηρείται, ότι οι μεταβλητές δεν αντιστοιχούν πλήρως σε αυτές που ζητάει η υπηρεσία του GUNET (preSharedKey αντί για pre-shared-key, dlrUrl αντί για dlr-url, smsForwardId αντί για sms-forward-id). Αυτό συμβαίνει επειδή η Java δεν επιτρέπει να οριστούν μεταβλητές που περιέχουν παύλες (-) στο όνομά τους. Η λύση έρχεται με την χρησιμοποίηση του annotation @JsonProperty πάνω από τις GET μεθόδους των μεταβλητών (Εικόνα 6.8), από τις οποίες φτιάχνεται το XML, καθώς οι μεταβλητές της κλάσεις είναι private.

Εικόνα 6.8

```
@JsonProperty("sms-forward-id")
public String getSmsForwardId() {
    return smsForwardId;
}

@JsonProperty("pre-shared key")
public String getPreSharedKey() {
    return preSharedKey;
}

@JsonProperty("dlr-url")
public String getDlrUrl() {
    return dlrUrl;
}
```

Η τελική αποστολή γίνεται μέσω της κλάσης GunetServices(Εικόνα 6.9), η οποία χρησιμοποιεί την βιβλιοθήκη του Jersey Client, η οποία παρέχεται μέσα σε ένα dependency, μαζί με το Jackson που αναφέρθηκε παραπάνω. Επειδή η υπηρεσία του GUNET υποστηρίζει SSL, η Java δεν επιτρέπει να πραγματοποιηθεί κλήση σε ασφαλές πρωτόκολλο εάν δεν περαστεί χειροκίνητα το πιστοποιητικό μέσα στο JKS της. Αυτό δημιουργεί δεδομένα προβλήματα, καθώς πρέπει συνεχώς να γίνεται έλεγχος για τυχόν αλλαγές / λήξη αυτού του πιστοποιητικού.

Για αυτό τον λόγο, δημιουργήθηκε μία επέκταση της κλάσης του Jersey Client(Εικόνα 6.10), η οποία αγνοεί τον παραπάνω περιορισμό. Αυτό δεν δημιουργεί προβλήματα ασφαλείας, καθώς καλώντας το συγκεκριμένο endpoint δεν λαμβάνονται κρίσιμα δεδομένα, και η επικοινωνία συνεχίζει να είναι κρυπτογραφημένη.

Στην εκτέλεση της μεθόδου αποστολής(Εικόνα 6.9), και πριν αποκτηθεί η σύνδεση στο endpoint, το αντικείμενο SendSmsModel μετατρέπεται στο αντίστοιχο JSON String μέσω της κλάσης ObjectMapper(Jackson lib), και το περνάει ως παράμετρος στην κλήση. Αφού ολοκληρωθεί αυτή η κλήση στην υπηρεσία, η εφαρμογή λαμβάνει το response, και το μετατρέπει σε Java object κλάσης SendResponseModel, και το επιστρέφει από την μέθοδο στο σημείο από το οποίο καλέστηκε, όπου και θα καταγραφεί.

Εικόνα 6.9

```
@Stateful
public class GunetServices {

	Client client;
	public SmsResponseModel sendSingleSms (SendSmsModel sendSms) {
		try {

			client = TrustAllClient.IgnoreSSLClient();
			WebTarget webTarget = client.target(GetPropertyValues.getProperties().getProperty("gunetUrl"));

			String json= new ObjectMapper().writeValueAsString(sendSms);
			// POST method
			Response response = webTarget.request().accept(MediaType.APPLICATION_JSON)
					.post(Entity.entity(json, MediaType.APPLICATION_JSON), Response.class);
			// check response status code
			if (response.getStatus() != 200) {
			    throw new RuntimeException("Failed : HTTP error code : "
				    + response.getStatus());
			}
			// display response
			SmsResponseModel smsResponse = response.readEntity(SmsResponseModel.class);
			client.close();
			return smsResponse;

		  } catch (Exception e) {
			e.printStackTrace();
			return null;
		  }
	}
```

Εικόνα 6.10

```
public class TrustAllClient
{
        public static Client IgnoreSSLClient() throws Exception {
                SSLContext sslcontext = SSLContext.getInstance("TLS");
                sslcontext.init(null, new TrustManager[]{new X509TrustManager() {
                    public void checkClientTrusted(X509Certificate[] arg0, String arg1) throws CertificateException {}
                    public void checkServerTrusted(X509Certificate[] arg0, String arg1) throws CertificateException {}
                    public X509Certificate[] getAcceptedIssuers() { return new X509Certificate[0]; }

                }}, new java.security.SecureRandom());
                return ClientBuilder.newBuilder().sslContext(sslcontext).hostnameVerifier((s1, s2) -> true).build();
            }
}
```

6.4.2 : SMS Forward

Κάθε υπηρεσία περιέχει έναν αριθμό keywords τα οποία μπορεί να χρησιμοποιήσει ο τελικός χρήστης, στέλνοντας τα με SMS στην πλατφόρμα. Το κομμάτι αυτό των υπηρεσιών ονομάζεται Mobile Originated (MO). Όταν η πλατφόρμα του GUNET λαμβάνει ένα SMS με κάποιο keyword, τότε προωθεί στο SMS σε κάποιο προκαθορισμένο εξωτερικό σύστημα μέσω Web Service. Για αυτό πρέπει να δημιουργηθεί ένα endpoint, το όποιο θα κάνει consume την αποστολή SendSms του web service του GUNET.

Το JSON που αποστέλλεται από το GUNET έχει την μορφή :

```
{
"MSISDN": "6901234657",
   "keyword": "BA",
   "body": "ΗΛΕΚΤΡΟΝΙΚΗ",
   "pre-shared key": "FOfesFADSr223fA",
   "sms-forward-id": "123"
}
```

Για την εύρυθμη λειτουργία του συστήματός τους, αναμένονται οι παρακάτω απαντήσεις σε αυτά τα αιτήματα:

Επιτυχές:

```
{
   "result": "0",
   "errorCode": ""
   "error": ""
}
```

Ανεπιτυχές:

```
{
```

```
        "result": "1",
  "errorCode": "error-code-1"
   "error": "error-description"
}
```

Παρακάτω παρουσιάζονται η κλάση η οποία θα περιέχει τα δεδομένα του μηνύματος SMS Forward(Εικόνα 6.11) που λαμβάνει η εφαρμογή, καθώς και η κλάση που θα περιέχει τα δεδομένα που θα αποσταλούν προς τον SMS Aggregator ως αποτέλεσμα της διαδικασίας της εφαρμογής(Εικόνα 6.12). Όπως και στην αποστολή SMS, χρησιμοποιούνται annotations για να συμβαδίζουν τα πεδία JAVA με τα αντίστοιχα του JSON.

Εικόνα 6.11

```
public class SmsForwardModel {

        // Jsonproperty gia to jackson, gia na kanei consume to service
        // SerializedName gia to gson, gia opiadipote metatropi json-object
        @XmlElement(name = "MSISDN")
        private String msisdn;// to kinito tilefwno toy apostolea-xristi
        private String keyword;// to string me to opoio ksekinaei to minima
        private String body;// self explanatory
        private String preSharedKey;// to key p exei simfwnithei me to gunet
        private String smsForwardId;
```

Εικόνα 6.12

```
public class SmsForwardResponseModel {

        // apostelnetai pisw sto gunet otan mas kanei smsforward
        private int result;// an diaxristike swsta 0. alliws 1
        private String errorCode;// to id toy error
        private String error;// to minima lathous unknown service, unknown message, unknown
                             // recipient, unknown institution, invalid pre-shared key,
                             // unauthorized sender address, user not opted in, user
                             // deactivated service, user credits expired, service
                             // deactivated, general error etc.
```

Αφού κατασκευαστούν οι model κλάσεις που αντιστοιχούν στα JSON πεδία, πρέπει να κατασκευαστεί και το ουσιαστικό endpoint της εφαρμογής(Εικόνα 6.13), το οποίο θα καλείται από την υπηρεσία του GUNET. Αυτό συμβαίνει χρησιμοποιώντας τις λειτουργίες της JavaEE, που παρέχονται από τον application server.

Εικόνα 6.13

```
@Path("/")
public class Gunet {

    @EJB
    MobileOriginated mobileOriginated;
    @EJB
    Logs logs;

    @POST
    @Produces(MediaType.APPLICATION_JSON)
    @Consumes(MediaType.APPLICATION_JSON)
    public SmsForwardResponseModel smsForward(SmsForwardModel smsRequest) {
        SmsForwardResponseModel smsResponse = mobileOriginated.reply(smsRequest);
        return smsResponse;
    }
```

Το @Path("/") χρησιμοποιείται για την εγγραφή της κλάσης ως web service. Στην συγκεκριμένη περίπτωση έχει οριστεί στην root τοποθεσία των webservice, η οποία έχει οριστεί μέσα από το αρχείο web.xml.

Το @POST σημαίνει ότι η συγκεκριμένη μέθοδος δέχεται μόνο αιτήματα POST

Το @Produces(Media.Type.Application_Json) σημαίνει ότι επιστρέφει τα δεδομένα σε μορφή JSON. Επίσης, λόγω της ύπαρξης του συγκεκριμένου annotation, η Java θα κάνει αυτόματα την μετατροπή του Java Object σε αντίστοιχο JSON String.

Το @Consumes(Media.Type.Application_Json)σημαίνει ότι η μέθοδος επιβάλλει τα δεδομένα να σταλούν σε μορφή JSON, και αναλόγως μετά και αυτό κάνει μετατροπή του Json string σε Java object.

Η μέθοδος επεξεργάζεται το αίτημα, παράγει την απάντηση σε αυτό (στις περισσότερες περιπτώσεις περιλαμβάνει αποστολή μηνύματος πίσω στον αρχικό αποστολέα) και στέλνει πίσω στο GUNET το status αυτής της ενέργειας.

6.4.3 : Delivery Reports

Ο SMS Aggregator, κατά την αποστολή SMS επιστρέφει μήνυμα στην εφαρμογή για το αν το μήνυμα στάλθηκε, όμως η συγκεκριμένη απάντηση δεν είναι η πραγματική επιβεβαίωση παραλαβής του μηνύματος, αλλά μόνο η κατάσταση αποστολής του μηνύματος (αν όντως δόθηκε η εντολή για αποστολή του μηνύματος).
Μετά την πραγματική αποστολή του κάθε μηνύματος, επιστρέφει σε ένα endpoint που καθορίζεται κατά την αποστολή κάθε μηνύματος και υλοποιείται από την εφαρμογή το πραγματικό delivery report, που παρέχει πλήρη στοιχεία για την παράδοση ή όχι του κάθε μηνύματος.

Το JSON μήνυμα που προωθείται στην εφαρμογή έχει την εξής δομή:

Επιτυχής παράδοση μηνύματος:

```
{
  "serviceId": "gradeService",
  "recipient": "6901234657",
  "status": "DELIVRD"
}
```


Ανεπιτυχής παράδοση μηνύματος:


```
{
  "serviceId": "gradeService",
  "recipient": "6901234657",
  "status": "ERROR",
  "error": "80"
}
```


Παρακάτω παρουσιάζεται η κλάση η οποία θα περιέχει τα δεδομένα του DLR μηνύματος(Εικόνα 6.14), καθώς και η μέθοδος η οποία θα λειτουργήσει ως endpoint για την αποστολή αυτών των DLR μηνυμάτων(Εικόνα 6.15).

Εικόνα 6.14

```
@XmlRootElement
public class DlrRequestModel {



    private String serviceId;//to Id tis ipiresias poy klithike
    private String recipient;//o xristis p parelave to minima
    private String status;//DELIVRD', 'ERROR', 'EXPIRED', 'PENDING', 'SENT', 'SUBMITTED', 'UNDELI
    private String error;//0-999
```

Εικόνα 6.15

```
@Path("dlr/")
@POST
@Consumes(MediaType.APPLICATION_JSON)
public void getDlr(DlrRequestModel dlrReq) {// methodos gia na pairnoyme
    logs.logDlr(dlrReq);
}
```

Το path καθορίζεται από την κλάση μέσα στην οποία περιέχεται η μέθοδος. Καθώς αυτή η κλάση έχει το @Path(“/”) όπως φαίνεται και στο παράδειγμα του SMS Forward, το endpoint θα είναι στο service/dlr/. Η μέθοδος είναι void, οπότε η απάντηση που επιστρέφει κατά την κλήση της είναι πάντα το 204:NO CONTENT

6.5 : Mobile Originated Υπηρεσίες

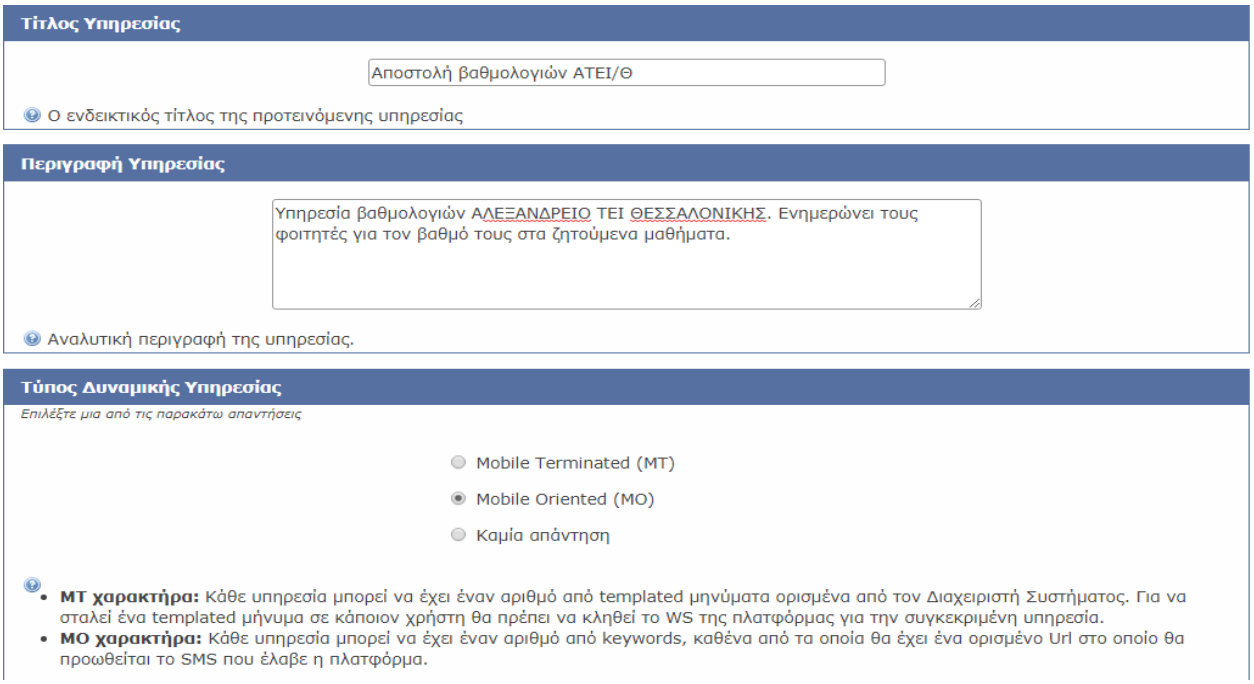
Όπως παρουσιάστηκε και στα προηγούμενα κεφάλαια, οι Mobile Originated υπηρεσίες είναι οι υπηρεσίες, στις οποίες η διαδικασία των υπηρεσιών ξεκινάει από τον χρήστη. Για την δημιουργία μίας τέτοιας υπηρεσίας, πρέπει η υπηρεσία καθώς και όλες οι παράμετροι αυτής (όπως keyword, προκαθορισμένες απαντήσεις της, endpoint της εφαρμογής, τύπος υπηρεσίας κλπ) να οριστούν στον SMS Aggregator και στην συνέχεια να υλοποιηθούν από την εφαρμογή.

6.5.1 : Αίτημα για εισαγωγή νέας υπηρεσίας στο σύστημα

Το αίτημα για δημιουργία μιας νέας υπηρεσίας μπορεί να γίνει μέσω του link που μας παρέχει το gunet για αυτόν τον σκοπό *(http://survey.gunet.gr/index.php/371887/lang-el)*

Παρακάτω παρουσιάζεται το αίτημα για εισαγωγή της υπηρεσίας 'Αποστολή Βαθμολογιών ΑΤΕΙ/Θ' . Αφού γίνει είσοδος στο survey link που αναφέρεται παραπάνω, και μετά την εισαγωγή των στοιχείων του ιδρύματος, παρουσιάζεται η σελίδα για την εισαγωγή της νέας υπηρεσίας (Εικόνα 6.16).

Εικόνα 6.16



Απαιτείται Επιβεβαίωση Κινητού;

○ Ναι ◉ Όχι ○ Καμία απάντηση

Η ερώτηση σχετίζεται με το αν η υπηρεσία απαιτεί επιβεβαίωση κινητού (εγγραφή 1ου επιπέδου) για το χρήστη. Στην περίπτωση που η υπηρεσία έχει οριστεί ως Opt-In υπηρεσία και επιχειρηθεί να χρησιμοποιηθεί η υπηρεσία για χρήστη που δεν είναι Opted-In τότε η πλατφόρμα θα ενημερώνει κατάλληλα. Στόχος είναι να επιβεβαιωθεί το κινητό του χρήστη.

Απαιτείται Ενεργοποίηση Χρήστη;

○ Ναι ◉ Όχι ○ Καμία απάντηση

Η ερώτηση σχετίζεται με το αν η υπηρεσία απαιτεί ενεργοποίηση (εγγραφή 2ου επιπέδου) από το χρήστη. Στόχος είναι να ενεργοποιηθεί η υπηρεσία εκ μέρους του χρήστη για να μπορεί να λαμβάνει σχετικά μηνύματα.

Keywords

Keyword 1	VATHMOS
Keyword 2	ΒΑΘ
Keyword 3	
Keyword 4	
Keyword 5	

Ποια keywords θα χρησιμοποιεί η συγκεκριμένη υπηρεσία;

Δυναμικά Πεδία

Δυναμικό Πεδίο 1	course
Δυναμικό Πεδίο 2	vathmos1
Δυναμικό Πεδίο 3	vathmos2
Δυναμικό Πεδίο 4	
Δυναμικό Πεδίο 5	

Καθορισμός δυναμικών πεδίων για την χρήση τους στα σχετικά ενδεικτικά μηνύματα.

Ενδεικτικό Κείμενο Μηνύματος (Παράδειγμα 1):

Ο ΒΑΘΜΟΣ ΣΑΣ ΣΤΟ ΜΑΘΗΜΑ [course ,1] ΕΙΝΑΙ [vathmos1 ,2]

Να συμπληρωθεί το ενδεικτικό κείμενο του μηνύματος συμπεριλαμβανομένου των δυναμικών πεδίων που ορίστηκαν παραπάνω. Παράδειγμα: ΓΙΑ ΤΗΝ ΕΝΕΡΓΟΠΟΙΗΣΗ ΤΟΥ ΗΛΕΚΤΡΟΝΙΚΟΥ ΣΑΣ ΛΟΓΑΡΙΑΣΜΟΥ, ΠΗΓΑΙΝΕΤΕ [ΔΥΝΑΜΙΚΟ ΠΕΔΙΟ, 1] ΚΑΙ ΕΙΣΑΓΕΤΕ ΤΟ PIN: [ΔΥΝΑΜΙΚΟ ΠΕΔΙΟ,2]

Ενδεικτικό Κείμενο Μηνύματος (Παράδειγμα 2):

Ο ΒΑΘΜΟΣ ΣΑΣ ΣΤΟ ΜΑΘΗΜΑ[course,1] ΕΙΝΑΙ [vathmos1,2] ΣΤΗΝ ΘΕΩΡΙΑ ΚΑΙ [vathmos2,3] ΣΤΟ ΕΡΓΑΣΤΗΡΙΟ

Να συμπληρωθεί το ενδεικτικό κείμενο του μηνύματος συμπεριλαμβανομένου των δυναμικών πεδίων που ορίστηκαν παραπάνω. Παράδειγμα:

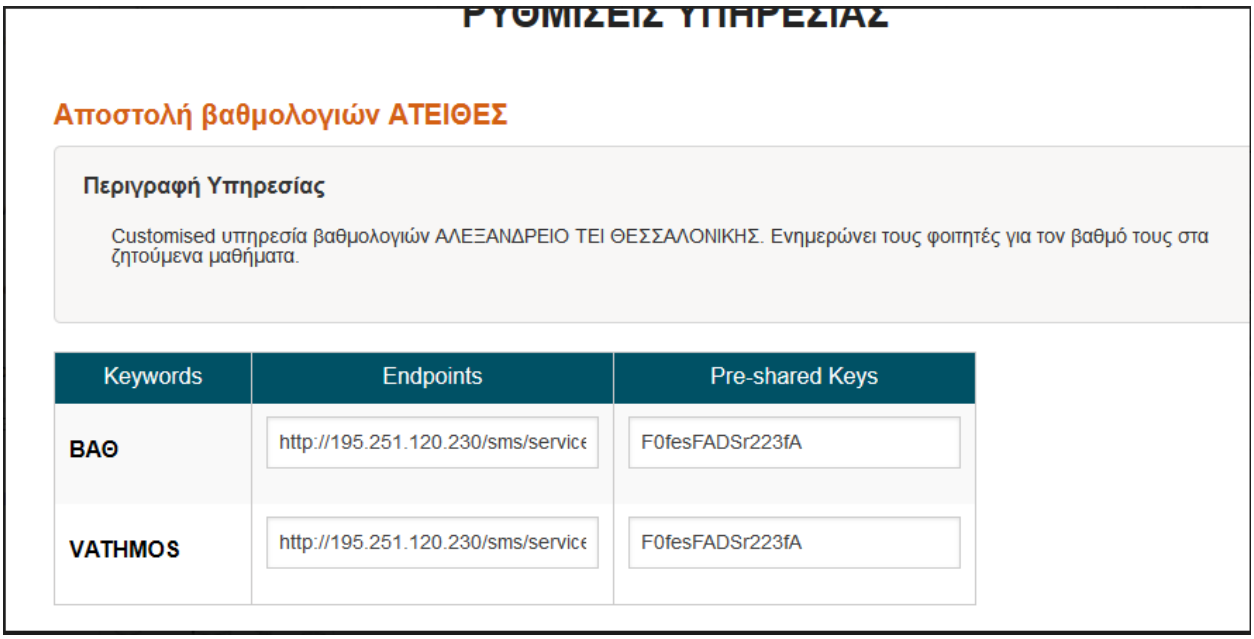
Επεξήγηση των παραπάνω ρυθμίσεων:

- Ο τίτλος και η περιγραφή της υπηρεσίας εμφανίζονται μόνο στην ιστοσελίδα του SMS Aggregator (sms.gunet.gr) και είναι πληροφοριακά πεδία.
- Ο τύπος της υπηρεσίας πρέπει να οριστεί ως Mobile Originated, όπως και ο τύπος της υπηρεσίας.
- Επιβεβαίωση κινητού τηλεφώνου : Ορίζεται ως 'ΟΧΙ' καθώς η επιβεβαίωση του κινητου τηλεφώνου γίνεται με διαδικασίες της εφαρμογής μας.
- Ενεργοποίηση χρήστη : Ορίζεται ως 'ΟΧΙ'. Σε περίπτωση που οριστεί ως 'ΝΑΙ', η υπηρεσία θα είναι απενεργοποιημένη από προεπιλογή, και οι χρήστες θα πρέπει να την ενεργοποιήσουν χειροκίνητα κάθε φορά από την ιστοσελίδα του SMS Aggregator.
- Keywords: Τα keywords με τα οποία θα αναγνωρίζεται η υπηρεσία. Όταν ένας χρήστης στέλνει μήνυμα που ξεκινάει με κάποιο από τα ορισμένα keywords, θα αντιστοιχίζεται με αυτά στον SMS Aggregator και αυτός στην συνέχεια θα στείλει το ανάλογο μήνυμα στην εφαρμογή.
- Δυναμικά πεδία και ενδεικτικό κείμενο μηνύματος : Τα μηνύματα που μπορούν να σταλούν , όπως έχει αναφερθεί ήδη, είναι προκαθορισμένα, εκτός από κάποια ορισμένα δυναμικά πεδία μέσα σε αυτά. Αυτό γίνεται δημιουργώντας αυτά τα μηνύματα με την μορφή που φαίνεται στην Εικόνα 6.5

6.5.2 : Ρυθμίσεις υπηρεσίας

Αφότου γίνει η υποβολή, το GUNET θα δημιουργήσει την υπηρεσία που ζητήθηκε, και θα εμφανιστεί στο διαχειριστικό του SMS Aggregator στον διαχειριστή της υπηρεσίας του ιδρύματος. Η υπηρεσία όμως δεν έχει ρυθμίσεις για το endpoint αποστολής, τα preshared key ασφαλείας κλπ. Παρακάτω (Εικόνα 6.17) εμφανίζονται οι ρυθμίσεις της υπηρεσίας 'Αποστολή βαθμολογιών ΑΤΕΙ/Θ' όπως εμφανίζονται στον διαχειριστή του ιδρύματος.

Εικόνα 6.17



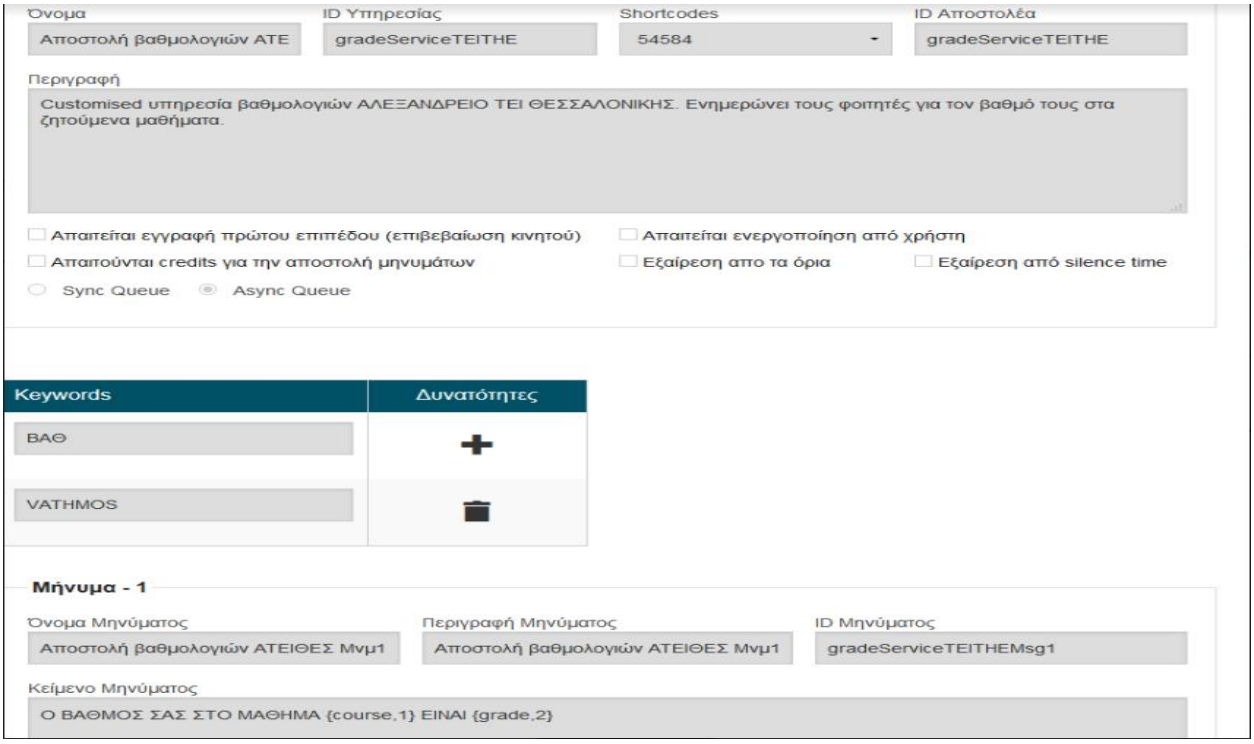
Στο πεδίο Endpoint εισάγεται η διεύθυνση του endpoint της εφαρμογής, στην οποία καταλήγουν τα μηνύματα της υπηρεσίας προς το ίδρυμα του ΑΤΕΙ/Θ. Αυτή η διεύθυνση έχει τοποθετηθεί στο *https://195.251.120.230/sms/service*

Στο πεδίο Pre-Shared keys ορίζεται ένα προκαθορισμένο κλείδι, το οποίο θα στέλνεται μαζί με κάθε μήνυμα από τον SMS Aggregator προς την εφαρμογή. Μέσω αυτού του κλειδιού μπορεί να γίνει επαλήθευση ότι το μήνυμα όντως προέρχεται από το GUNET και όχι από κάποια τρίτη εφαρμογή.

Στην συνέχεια, μέσα από το διαχειριστικό(Εικόνα 6.18), μπορεί να γίνει ανάκτηση των διάφορων πληροφοριών που χρειάζονται για την υλοποίηση της υπηρεσίας από την εφαρμογή. Στην περίπτωση των MO υπηρεσιών, χρειάζονται:

- Το ID της υπηρεσίας
- Τα ID των μηνυμάτων
- Τα καταχωρημένα keywords
- Τα Shortcodes (τα νούμερα στα οποία γίνεται η αποστολή μηνυμάτων από τον χρήστη)

Εικόνα 6.18



6.5.3 : Εισαγωγή στο δυναμικό XML

Έχοντας πλέον όλες τις απαραίτητες πληροφορίες για την υπηρεσία, η υπηρεσία μπορεί πλέον να υλοποιηθεί από την εφαρμογή. Για την αποφυγή επεξεργασίας του πηγαίου κώδικα και δημιουργία νέου πακέτου εφαρμογής κάθε φορά που γίνεται προσθήκη ή αλλαγή μίας υπηρεσίας, οι υπηρεσίες υλοποιούνται μέσα σε ειδικό XML, που φορτώνεται κατά την εκκίνηση της εφαρμογής. Παρακάτω (Εικόνα 6.19) φαίνεται το παράδειγμα εισαγωγής της υπηρεσίας ‘Αποστολή βαθμολογιών ΑΤΕΙ/Θ’ σε αυτό το αρχείο.

Εικόνα 6.19

```
<service>
    <serviceId>gradeServiceTEITHE</serviceId>
    <keywords>
        <keyword>VATHMOS</keyword>
    </keywords>
    <preSharedKey>F0fesFADSr223fA</preSharedKey>
    <database>pithia</database>
    <message>
        <keyword></keyword>
        <messageId>gradeServiceTEITHEMsg1</messageId>
        <numberOfReplacements>2</numberOfReplacements>
        <query>SELECT ctitle,CONVERT(DECIMAL(10,1),grade) as grade1
            FROM
            [eUniCentral].[dbo].[v_SMS_gradeService]
            where username=? and
            ctitle LIKE ?
            order by acyear desc
        </query>
        <likePosition>2</likePosition>
        <description>ΕΠΙΣΤΡΕΦΕΙ ΤΟΝ ΒΑΘΜΟ ΤΟΥ ΧΡΗΣΤΗ ΣΤΟ ΣΥΓΚΕΚΡΙΜΕΝΟ ΜΑΘΗΜΑ
        </description>
        <userInput>ONOMA ΜΑΘΗΜΑΤΟΣ</userInput>
    </message>

    <message>
        <keyword>KSEXWR</keyword>
        <messageId>gradeServiceTEITHEMsg2</messageId>
        <numberOfReplacements>3</numberOfReplacements>
        <query>

            DECLARE @USERNAME VARCHAR(50)
```

- serviceID: Το ID της υπηρεσίας και έχει ανακτηθεί από το διαχειριστικό του SMS Aggregator όπως παρουσιάστηκε παραπάνω.
- Keywords: Τα διάφορα keywords της υπηρεσίας
- preSharedKey: Το κλειδί ασφαλείας που έχει οριστεί για την υπηρεσία από τον διαχειριστή του ιδρύματος
- database: Η προεπιλεγμένη βάση για τα ερωτήματα της υπηρεσίας.
- message: Πέραν από τις ρυθμίσεις της υπηρεσίας, πρέπει να οριστεί κάθε μήνυμα που υλοποιείται.
    - Keyword: Τα διάφορα ερωτήματα για αυτά τα μηνύματα, ξεκινούν με το keyword της υπηρεσίας, και μπορούν να συνεχιστούν στο μήνυμα. ΠΧ για να γίνει ερώτημα για τους βαθμούς ξεχωριστά, πρέπει να σταλεί το μήνυμα «TEITHE VATHMOS KSEXWR”. Το μήνυμα στέλνεται στην υπηρεσία που έχει keyword «VATHMOS” , και στην συνέχεια επεξεργάζεται από αυτήν που έχει αυτό το δευτερεύον keyword. Εάν το keyword του μηνύματος του είναι κενό, τότε κάθε αίτημα που δεν περιέχει ως δευτερεύον κάποιο από τα ορισμένα, επεξεργάζεται από αυτό(πχ “TEITHE VATHMOS” ή “TEITHE VATHMOS ΔΙΑΚΡΙΤΑ»
    - messageId: Το ID του και έχει ανακτηθεί από το διαχειριστικό του SMS Aggregator όπως παρουσιαστήκε παραπάνω.
    - numberOfReplacements : Το σύνολο των replacement για το συγκεκριμένο μήνυμα. Στο παραδειγμά της αποστολής βαθμολογιών, τα replacements είναι 2 (μάθημα και βαθμός)
    - query: Το ερώτημα που γίνεται προς την επιλεγμένη βάση. Τα ορισμένα replacements θα παραχθούν από το πρώτο αποτέλεσμα αυτού του ερωτήματος, με την σειρά που εμφανίζονται στο SELECT. Σημείωση : Στα ερωτήματα που γίνονται στην βάση της

Πυθίας, ως πρώτη παράμετρος του ερωτήματος πρέπει να είναι το username. Αντίστοιχα, στα ερωτήματα της Αιμοδοσίας, πρέπει να είναι το mobile.
    - likePosition: Εάν το ερώτημα που έχει οριστεί στο query περιέχει παραμέτρους LIKE, η θέση τους στο ερώτημα πρέπει να οριστεί εδώ. Εάν όχι, αυτή η παράμετρος μπορεί να αγνοηθεί.
    - Database: Εάν θέλουμε να χρησιμοποιηθεί μια διαφορετική βάση από την προεπιλεγμένη της υπηρεσίας, μπορει να τεθεί σε αυτό το σημείο και να την κάνει override.
    - Description: Η περιγραφή που θα εμφανίζεται στην ιστοσελίδα της εφαρμογής.
    - userInput: Κομμάτι της παραπάνω περιγραφής, για τα μηνύματα στα οποία πρέπει ο χρήστης να δώσει κάποια παράμετρο
- errorMessage: Το μήνυμα λάθους που έχει οριστεί κατά την δημιουργία της υπηρεσίας. Εμφανίζεται εάν παρουσιαστεί οποιοδήποτε σφάλμα σε κάποιο από τα υπόλοιπα ερωτήματα της υπηρεσίας
    - messageId: messageId: Το ID του και έχει ανακτηθεί από το διαχειριστικό του SMS Aggregator όπως παρουσιαστήκε παραπάνω.
    - numberOfReplacements: Το σύνολο των replacements για αυτό το μήνυμα. Στις περισσότερες περιπτώσεις αυτός ο αριθμός είναι 0.

6.5.4 : Ανάκτηση δυναμικού XML

Εφόσον εισαχθεί μια υπηρεσία στο αρχείο με τον τρόπο που παρουσιάστηκε, πρέπει να φορτωθεί από την εφαρμογή. Αυτό γίνεται μέσω της κλάσης Services.java (Εικόνα 6.20). Πρόκειται για μια κλάση Singleton (που ορίζεται από το annotation @Singleton), που σημαίνει ότι κατά την δημιουργία και χρησιμοποίηση της με οποιονδήποτε τρόπο από την εφαρμογή, δημιουργείται μόνο και μόνο ένα instance της. Το συγκεκριμένο instance δημιουργείται κατά την εκκίνηση της εφαρμογής, μέσω των annotation @Startup (που ορίζεται στην ίδια την κλάση) και @PostConstruct (που ορίζεται στην μέθοδο που θα εκτελεστεί).

Κατά την εκτέλεση της μεθόδου loadMobileOriginated , πραγματοποιούνται 3 βασικές λειτουργίες:

- Γίνεται populate του πίνακα (hashMap) mobileOriginatedServices. Μέσω του JaxB γίνεται unmarshal το αρχείο XML που χρησιμοποιήσαμε προηγουμένως σε Java Objects, τα οποία καταχωρούνται με τα keywords τους στο HashMap. Εδώ αξίζει να σημειωθεί, ότι τα keywords που αποστέλλονται την εφαρμογή από τον SMS Aggregator, είναι όλα με ελληνικούς χαρακτήρες (πλην των αγγλικών που δεν υπάρχουν στο ελληνικό αλφάβητο), για αυτόν τον λόγο γίνεται και η μετατροπή των ορισμένων από την εφαρμογή keywords στην αντίστοιχη μορφή.

- Γίνεται populate του πίνακα (ArrayList) descriptions, που χρησιμοποιείται για την παρουσίαση των MO υπηρεσιών στην ιστοσελίδα. Αυτό σημαίνει, ότι κατά την είσοδο μιας νέας υπηρεσίας στην εφαρμογή, αυτομάτως ανανεώνονται και τα δεδομένα που υπάρχουν στην ιστοσελίδα.
- Υπολογισμός των παραμέτρων του ερωτήματος. Για κάθε μήνυμα υπολογίζονται οι παράμετροι που χρησιμοποιούνται από το ερώτημα του χωρίς να χρειάζεται να γίνει χειροκίνητη εισαγωγή στο XML αρχείο.

Εικόνα 6.20

```
@Startup
@Singleton
public class Services {

	private HashMap<String, MobileTerminatedService> mobileTerminatedServices = new HashMap<String, MobileTerminatedService>();
	private HashMap<String, MobileOriginatedService> mobileOriginatedServices = new HashMap<String, MobileOriginatedService>();
	private ArrayList<ServiceDescription> descriptions = new ArrayList<ServiceDescription>();
	private ArrayList<SmsTemplate> smsTemplatesMoodle = new ArrayList<SmsTemplate>();
	private ArrayList<SmsTemplate> smsTemplatesDirect = new ArrayList<SmsTemplate>();

	@PostConstruct
	private void init() {
		this.loadMobileOriginated();
		this.loadMobileTerminated();
	}
	public void loadMobileOriginated() {
		try {
			JAXBContext jc = JAXBContext.newInstance(com.smsserver.services.models.mobileoriginated.Root.class);
			Unmarshaller unmarshaller = jc.createUnmarshaller();
			ClassLoader classLoader = Services.class.getClassLoader();
			File questionXML = new File(classLoader.getResource("mobileoriginated.xml").getFile());

			com.smsserver.services.models.mobileoriginated.Root moRoot = (com.smsserver.services.models.mobileoriginated.Root) unmars
					.unmarshal(questionXML);

			for (MobileOriginatedService s : moRoot.getMobileOriginatedService()) {
				for (String keyword : s.getKeywords().keywordList){
					mobileOriginatedServices.put(EnToGSM7Converter.EnglishToGSM7(keyword), s);
				}
				for (com.smsserver.services.models.mobileoriginated.Message m : s.getMessages()) {
					descriptions.add(new ServiceDescription(
							"ΤΕΙΤΗΕ "+s.getKeywords().keywordList.get(0) + " " + m.getFullKeyword(), m.getDescriptior

					m.computeQueryParams();
				}
			}
```

6.5.5 : Διαδικασία απάντησης ερωτημάτων

Αρχικά, όπως έχει παρουσιαστεί στην ενότητα 6.4.3, έχει υλοποιηθεί το endpoint το οποίο έχει οριστεί για την αποστολή των μηνυμάτων από τον SMS Aggregator προς την εφαρμογή. Με την παραλαβή κάποιου τέτοιου μηνύματος, αρχίζει η διαδικασία αποκωδικοποίησης του και προετοιμασία του μηνύματος που περιέχει την απάντηση στον χρήστη που έστειλε το μήνυμα. Η κλάση που διαχειρίζεται αυτήν την διαδικασία είναι η *MobileOriginated.java.*

Κύρια μέθοδος της διαδικασίας

Από την μέθοδο που υλοποιεί το endpoint, καλείται η μέθοδος reply(). Μέσα σε αυτήν, ακολουθούνται οι εξής ενέργειες:

- Έλεγχος για ύπαρξη του keyword που περιέχεται στο απεσταλμένο μήνυμα στο HashMap με τις υπηρεσίες MO, που δημιουργήθηκε προηγουμένως.
- Ανάκτηση των δεδομένων της υπηρεσίας, για να χρησιμοποιηθούν μεταγενέστερα.
- Έλεγχος εγκυρότητας του preSharedKey που είναι δηλωμένο στην εφαρμογή για την υπηρεσία, με το preSharedKey που υπάρχει στο μήνυμα που έχει γίνει προώθηση στην εφαρμογή.
- Προετοιμασία του μηνύματος απάντησης. Σε περίπτωση σφάλματος, η μέθοδος αυτή πετάει ένα generic exception και δεν υπάρχει απάντηση προς τον χρήστη.
- Αποστολή του μηνύματος απάντησης προς τον χρήστη, και καταγραφή του response που παράγει ο SMS Aggregator για την επιτυχή ή όχι αποστολή του.
- Καταγραφή της διαδικασίας στην βάση της εφαρμογής.

Παρακάτω παρουσιάζεται ο Java κώδικας(Εικόνα 6.21).

Εικόνα 6.21

```
public SmsForwardResponseModel reply(SmsForwardModel smsRequest) {
    if (!services.getMobileOriginatedServices().containsKey(smsRequest.getKeyword()))
        return new SmsForwardResponseModel(false, "Wrong keyword", "0");

    MobileOriginatedService mobileOriginatedService = services.getMobileOriginatedServices().get(smsRequest.getKeyword());

     if(!mobileOriginatedService.getPreSharedKey().equals(smsRequest.getPreSharedKey()))
     return new SmsForwardResponseModel(false, "Wrong preSharedKey", "0");

    try {
        SendSmsModel sms = null;
        sms = prepareReply(smsRequest, mobileOriginatedService);

        SmsResponseModel response = gunet.sendSingleSms(sms);
        if(response.getError().equals(""))
            LOGGER.log(Level.INFO, "Succesful Reply "+sms+" to sms Request "+smsRequest);
        else
            LOGGER.log(Level.INFO, "SMS not delivered "+sms,response);
        logs.logMobileOriginated(smsRequest, sms, response);
        return new SmsForwardResponseModel(true);
    } catch (Exception e) {
        LOGGER.log(Level.SEVERE, "SMS not sent "+ smsRequest,e);
        return new SmsForwardResponseModel(false, e.getMessage(), "0");
    }

}
```

Μέθοδος προετοιμασίας μηνύματος

Από την προηγούμενη μέθοδο, καλείται η μέθοδος prepareReply(), η οποία όπως αναφέρθηκε και προηγουμένως προετοιμάζει την απάντηση που θα σταλεί στον χρήστη. Η μέθοδος δέχεται ως παραμέτρους το απεσταλμένο μήνυμα του χρήστη, και τις παραμέτρους της υπηρεσίας στην οποία ανήκει αυτό το μήνυμα. Στην συνέχεια εκτελεί τις εξής ενέργειες:

- Μετατροπή του μηνύματος που στάλθηκε μέσω SMS από GSM7 μορφή σε Unicode. Το κείμενο που περιέχεται στο body, έχοντας μορφή GSM7 αντικαθιστά ελληνικούς χαρακτήρες που

συνυπάρχουν στο αγγλικό αλφάβητο σε αγγλικούς. Τα ερωτήματα της εφαρμογής όμως στηρίζονται στους ελληνικούς χαρακτήρες, οπότε γίνεται μετατροπή σε μια unicode μορφή. Πρέπει να σημειωθεί, ότι με αυτόν τον τρόπο, πλέον δεν υποστηρίζονται ερωτήματα ,όπου η παράμετρος είναι γραμμένη με αγγλικούς χαρακτήρες.

- Εύρεση του αντίστοιχου μηνύματος μέσω του οποίου θα πραγματοποιηθεί η απάντηση. Πραγματοποιείται αναζήτηση για το εάν υπάρχει κάποιο keyword της υπηρεσίας που ταιριάζει με αυτό του μηνύματος. Εφόσον δεν βρεθεί, ελέγχει εάν υπάρχει κάποιο μήνυμα που δεν έχει ορισμένο keyword (κενό). Εάν και πάλι δεν βρεθεί, τότε η μέθοδος πετάει exception, και επιστρέφει.
- Γίνεται έλεγχος για το είδος της υπηρεσίας. Προς το παρόν υπάρχουν 2 τύποι υπηρεσιών:
    - Ακαδημαϊκές υπηρεσίες. Σε αυτήν την περίπτωση, πρέπει να βρεθεί πρώτα το ID του χρήστη, για την πραγματοποίηση των ερωτημάτων στην βάση της Πυθίας μέσω της μεθόδου queryPithia της κλάσης pithiaDao.
    - Υπηρεσίες αιμοδοσίας. Σε αντίθεση με τις ακαδημαικές υπηρεσίες, η ταυτόποιηση του χρήστη για τα ερωτήματα γίνεται μέσω του νούμερου του κινητού του τηλεφώνου(MSISDN) απευθείας, μέσω της μεθόδου queryAimodosia της κλάσης aimodosiaDao.
- Σε περίπτωση αποτυχίας εύρεσης απάντησης στα ερωτήματα των υπηρεσιών, η εφαρμογή στέλνει ως απάντηση το ορισμένο μήνυμα λάθους της υπηρεσίας
- Επιστρέφεται το μήνυμα προς αποστολή.

Παρακάτω φαίνεται ο Java κωδίκας (Εικόνα 6.22).

Εικόνα 6.22

```
private SendSmsModel prepareReply(SmsForwardModel smsRequest, MobileOriginatedService mobileOriginatedService)
        throws Exception {
    String body=GsmConverter.Gsm7ToString(smsRequest.getBody());
    String[] userParameters = body.split("\\s+");;
    int extra = 0;
    Message message = null;
    if (!smsRequest.getBody().equals("")) {
        for (Message mes : mobileOriginatedService.getMessages()) {
            if (mes.getKeyword().equalsIgnoreCase(userParameters[0])) {
                message = mes;
                extra++;
            }
        }
    }
    if(message==null){
        for (Message mes : mobileOriginatedService.getMessages())
            if (mes.getKeyword().equals(""))
                message = mes;
    }
    if (message == null)
        throw new Exception("No message found with default keyword");

    String[] replacements = null;
    SendSmsModel sms;
    try {
        if (mobileOriginatedService.getDatabase().equalsIgnoreCase("pithia")){
            String authenticator = discovery.getUsername(smsRequest.getMsisdn());
            replacements = pithiaDao.queryPithia(message, authenticator, userParameters,extra);
        }
        else if(mobileOriginatedService.getDatabase().equalsIgnoreCase("nireas")){
            replacements=aimodosiaDao.queryAimodosia(message, smsRequest.getMsisdn(), userParameters,extra);
        }
    } catch (Exception e) {
        message = mobileOriginatedService.getErrorMessage();
        replacements=new String[0];
        LOGGER.log(Level.INFO, "SMS forward unsuccesful reply ",e);
    }
    sms = new SendSmsModel(mobileOriginatedService.getServiceId(), message.getMessageId(), replacements,
            smsRequest.getMsisdn(), smsRequest.getSmsForwardId());
    return sms;
```

6.5.6 : Εισαγωγή νέας βάσης για ερωτήματα

Σε περίπτωση που χρειαστεί να γίνει προσθήκη μίας νέας βάσης για ανάλογες υπηρεσίες, πρέπει να ακολουθηθεί η εξής διαδικασία:

- Προσθήκη της νέας βάσης ως JDBC Connection pool και απόδοση JNDI name μέσα στις ρυθμίσεις του application server, όπως περιγράφεται στην ενότητα 6.9.4
- Δημιουργία κλάσης dao της νέας βάσης δεδομένων. Αυτή η κλάση μπορεί να είναι αντίγραφο των ήδη υπάρχοντών κλάσεων που υλοποιούν την εκτέλεση ερωτημάτων. Η μόνη διαφοροποίηση πρέπει να είναι η βάση (ή βάσεις) στις οποίες εκτελούνται τα ερωτήματα, που δηλώνεται ως μεταβλητή στην κλάση, αλλάζοντας το όνομα στο οποίο γίνεται αναζήτηση στο πεδίο @Resource.
- Εισαγωγή της λογικής της νέας κλάσης στην μέθοδο prepareReply (ενότητα 6.5.5), στο σημείο που γίνεται έλεγχος για το είδος της υπηρεσίας
- Ορισμός της νέας βάσης ερωτημάτων στις εκάστοτε υπηρεσίες που υπάρχουν στο αρχείο mobileoriginated.xml (Ενότητα 6.5.3), στο πεδίο database, είτε σε ολόκληρη την υπηρεσία είτε σε μεμονωμένα μηνύματα .

6.6 : Mobile Terminated Υπηρεσίες

6.6.1 : Εισαγωγή νέας υπηρεσίας στην εφαρμογή.

Η εισαγωγή μιας νέας MT υπηρεσίας στην εφαρμογή γίνεται με παρόμοιο τρόπο με αυτόν που γίνεται για της MO υπηρεσίες(ενότητες 6.5.1,6.5.2,6.5.3). Οι υπηρεσίες MT χρειάζονται λιγότερες ρυθμίσεις για την διασύνδεση με τον SMS Aggregator, καθώς η επικοινωνία ξεκινάει από την εφαρμογή, οπότε δεν χρειάζονται οι ρυθμίσεις για την επικοινωνία του SMS Aggregator προς την εφαρμογή, παρα μόνο η επικοινωνία για την αποστολή μηνυμάτων.

Αρχικά, πρέπει να γίνει η δημιουργία της υπηρεσίας στον SMS Aggregator. Στο survey που συμπληρώνεται, πρέπει να επιλεχθεί η επιλογή ‘Mobile Terminated’. Τα πεδία στα οποία συμπληρώνονται τα keywords πρέπει να είναι κενά. Οι υπόλοιπες ρυθμίσεις είναι ίδιες με αυτές που ορίζονται και στις MO υπηρεσίες.

Οι ρυθμίσεις της υπηρεσίας που βρίσκονται στο διαχειριστικό του SMS Aggregator, έχουν επίσης ελάχιστες ρυθμίσεις. Για τις MT υπηρεσίες πρέπει να οριστεί μόνο το preSharedKey, το οποίο στέλνεται μαζί με οποιοδήποτε μήνυμα από την εφαρμογή προς τον SMS Aggregator κατά την εκτέλεση των αιτημάτων αποστολής μηνυμάτων. Από το διαχειριστικό γίνεται ανάκτηση των ID της υπηρεσίας και τον μηνυμάτων, τα οποία θα οριστούν στην εφαρμογή μεταγενέστερα.

Κάθε νέα υπηρεσία και οι παράμετροι της πρέπει να οριστούν στην εφαρμογή. Η διαδικασία είναι ίδια με τις MO υπηρεσίες, αλλά δηλώνεται σε ένα διαφορετικό αρχείο που περιέχει τις MT υπηρεσίες, το οποίο ονομάζεται mobileterminated.xml. Παρακάτω (Εικόνα 6.23) εμφανίζεται το παράδειγμα εισαγωγής της υπηρεσίας 'cancelCourse' σε αυτό το αρχείο.

Εικόνα 6.23

```
<service>
     <serviceId>cancelCourse</serviceId>
     <message>
          <messageId>cancelCourseMsg1</messageId>
          <message>ΤΟ ΜΑΘΗΜΑ ? ΤΗΣ ? ΑΝΑΒΑΛΛΕΤΑΙ</message>
     </message>
     <message>
          <messageId>cancelCourseMsg2</messageId>
          <message>ΤΟ ΜΑΘΗΜΑ ? ΤΗΣ ? ΑΝΑΒΑΛΛΕΤΑΙ ΚΑΙ ΘΑ ΠΡΑΓΜΑΤΟΠΟΙΗΘΕΙ ΤΗΝ ?
          </message>
     </message>
     <message>
          <messageId>cancelCourseMsg3</messageId>
          <message>ΘΑ ΓΙΝΕΙ ΑΝΑΠΛΗΡΩΣΗ ΤΟΥ ΜΑΘΗΜΑΤΟΣ ? ΤΗΝ ?</message>
     </message>
     <preSharedKey>123789</preSharedKey>
     <type>pithia</type>
</service>
```

- serviceId: Το ID της υπηρεσίας, όπως έχει ανακτηθεί από το διαχειριστικό του SMS Aggregator.
- Message: Το μήνυμα το οποίο έχει δηλωθεί στην υπηρεσία
    - messageId: Το ID του μηνύματος, όπως έχει ανακτηθεί από το διαχειριστικό του SMS Aggregator
    - message: Το μήνυμα το οποίο έχει οριστεί προς αποστολή. Το μήνυμα που ορίζεται εδώ όμως, δεν έχει ουσιαστική επίδραση στο τελικό μήνυμα που θα καταλήξει τελικά στους χρήστες, παρα μόνον ο αριθμός των replacements του μηνύματος, που ορίζονται από το σύνολο των ? που υπάρχουν σε αυτό. Ο σκοπός ύπαρξης αυτού του μηνύματος σε αυτό το σημείο είναι για την εμφάνιση αυτού στην ιστοσελίδα της εφαρμογής.
- preSharedKey: Το preSharedKey που έχει οριστεί στις ρυθμίσεις της υπηρεσίας στο διαχειριστικό
- type: Ο τύπος της υπηρεσίας. Μπορεί να είναι είτε 'pithia' , για τις αποστολές μαζικών μηνυμάτων προς φοιτητές, είτε 'direct' για τις στοχευμένες αποστολές. Αναλόγως την επιλογή, η υπηρεσία είναι διαθέσιμη προς επιλογή στην ιστοσελίδα και την ίδια την εφαρμογή.

6.6.2 : Αποστολή μαζικών μηνυμάτων προς φοιτητές

Η εφαρμογή υποστηρίζει την αποστολή μαζικών μηνυμάτων ενημέρωσης προς φοιτητές που είναι εγγεγραμμένοι σε κάποιο μάθημα του ΑΤΕΙ/Θ από τους καθηγητές που το διδάσκουν. Η ενέργεια αποστολής μηνυμάτων μπορεί να γίνει μόνο από την ιστοσελίδα, η οποία επικοινωνεί με το API της εφαρμογής για την πραγματοποίηση αυτής της ενέργειας. Παρακάτω (Εικόνα 6.24) παρουσιάζεται η μέθοδος που υλοποιεί το συγκεκριμένο endpoint.

Εικόνα 6.24

```
@Path("/massPithia")
@POST
@Produces(MediaType.APPLICATION_JSON)
@Consumes(MediaType.APPLICATION_JSON)
@Secured({Role.STAFF,Role.ADMIN})
public Response sendMoodle(SendSmsRequestMoodle req) throws URISyntaxException {
    Principal principal = securityContext.getUserPrincipal();
    req.setProfessor(principal.getName());

    try {
        return Response.ok(mobileTerminated.sendMassPithia(req)).build();
    } catch (Exception e) {
        e.printStackTrace();
        return Response.status(Status.INTERNAL_SERVER_ERROR).build();
    }

}
```

Η συγκεκριμένη μέθοδος, λαμβάνει ως παραμέτρους το ID του μαθήματος όπως είναι δηλωμένο στην πυθία, το ID του μηνύματος προς αποστολή και τα replacements του μηνύματος προς αποστολή. Μέσω του securityContext βρίσκει αυτόματα το ID του καθηγητή που εκτέλεσε αυτήν την ενέργεια, και προωθεί την διαδικασία στην μέθοδο sendMassPithia της κλάσης mobileTerminated.

Η μέθοδος sendMassPithia (Εικόνα 6.25) έχει τις ίδιες παραμέτρους όπως η προηγούμενη μέθοδος. Ξεκινώντας η διαδικασία αποστολής μηνυμάτων, η εφαρμογή αναζητά τα νούμερα των κινητών τηλεφώνων στα οποία θα σταλεί το επιλεγμένο μήνυμα. Αυτό γίνεται συνδυάζοντας τα νούμερα των τηλεφώνων των φοιτητών που είναι δηλωμένοι στο επιλεγμένο μάθημα προς αποστολή, στην βάση της πυθίας, σε συνδυασμό με τα τηλέφωνα που είναι δηλωμένα στην τοπική βάση της εφαρμογής. Αφού βρεθεί η λίστα με τα τηλέφωνα, γίνεται ανάκτηση των δεδομένων της MT υπηρεσίας μέσω του messageID του μηνύματος προς αποστολή. Σύμφωνα με αυτά τα δεδομένα, πραγματοποιείται η δημιουργία του μηνύματος προς αποστολή, το οποίο γίνεται προώθηση προς αποστολή από την μέθοδο sendSmsParallel(). Η συγκεκριμένη μέθοδος στέλνει το μήνυμα με έναν παράλληλο διαδικαστικό τρόπο προς τον SMS Aggregator στην λίστα με τα τηλέφωνα των φοιτητών, που είχε ανακτηθεί προηγουμένως. Παίρνει ως παραμέτρους το σύνολο των thread παραλληλισμού(στο παράδειγμα μας 2) και το Boolean true/false, όπου θέτοντας το 'true' πραγματοποιείται δοκιμαστική αποστολή. Με την ολοκλήρωση της αποστολής, πραγματοποιείται καταγραφή της ενέργειας αποστολής στον τοπική βάση της εφαρμογής, και επιστρέφεται για ενημέρωση του χρήστη (καθηγητή) το αποτέλεσμα της αποστολής.

Εικόνα 6.25

```
public SendReport sendMassPithia(SendSmsRequestMoodle request) throws Exception {

    ArrayList<String> mobileNumbers = discovery.getMobileMass(request.getCourse());

    String serviceId = services.getMobileTerminatedServices().get(request.getMessageId()).getServiceId();

    SendSmsModel sms = new SendSmsModel(serviceId, request.getMessageId(), request.getReplacements());
    int delivered = sendSmsParallel(sms, mobileNumbers, 2,true);
    logs.logMobileTerminated(request.getCourse(), request.getProfessor(), sms, mobileNumbers.size(), delivered);
    return new SendReport(mobileNumbers.size(), delivered);

}
```

6.6.3 : Αποστολή μαζικών μηνυμάτων προς αιμοδότες

Η αποστολή μαζικών μηνυμάτων προς τους εθελοντές αιμοδότες πραγματοποιείται μέσω της ιστοσελίδας, η οποία στέλνει την εντολή για την αποστολή των μηνυμάτων μέσω του endpoint διασύνδεσης με την εφαρμογή, το οποίο υλοποιείται από την μέθοδο sendAimodosia(Εικόνα 6.26). Αυτή η μέθοδος είναι προσβάσιμη μόνο από ταυτοποιημένο χρήστη με δικαιώματα διαχειριστή, και παίρνει ως παράμετρο μόνο ένα String που θα χρησιμοποιηθεί ως replacement στην ημερομηνία της αιμοδοσίας.

Εικόνα 6.26

```
@Path("/aimodosia")
@POST
@Produces(MediaType.APPLICATION_JSON)
@Consumes(MediaType.TEXT_PLAIN)
@Secured(Role.ADMIN)
public Response sendAimodosia(String date) throws URISyntaxException {
    try {
        return Response.ok(mobileTerminated.sendAimodosia(date)).build();
    } catch (Exception e) {
        e.printStackTrace();
        return Response.status(Status.INTERNAL_SERVER_ERROR).build();
    }
}
```

Η διαδικασία προωθείται στην μέθοδο sendAimodosia(Εικόνα 6.27) της κλάσης mobileTerminated. Η διαδικασία ξεκινάει κάνοντας ανάκτηση των τηλεφώνων στα οποία θα σταλεί το ενημερωτικό μήνυμα για την επικείμενη αιμοδοσία. Τα τηλέφωνα αυτά αποτελούνται από αυτά των εθελοντών αιμοδοτών που δεν έχουν δώσει αίμα τουλάχιστον 6 μήνες πριν την ημερομηνία που δηλώθηκε, καθώς και των τηλεφώνων που έχουν εγγραφεί στα ενημερωτικά μηνύματα της υπηρεσίας της αιμοδοσίας μέσω γραπτού μηνύματος. Αφού γίνει αυτή η ανάκτηση, προετοιμάζεται το μήνυμα προς αποστολή με τα στοιχεία της υπηρεσίας της αιμοδοσίας. Αυτά τα στοιχεία δεν περιέχονται στο αρχείο mobileterminated.xml, αλλά ορίζονται χειροκίνητα σε αυτό το σημείο του κώδικα. Στην συνέχεια γίνεται η αποστολή των μηνυμάτων με την μέθοδο sendSmsParallel, και καταγραφή της ενέργειας στην τοπική βάση.

Εικόνα 6.27

```
public SendReport sendAimodosia(String date) throws Exception {

    ArrayList<String> mobileNumbers = aimodosia.getMobileNumbers(date);

    SendSmsModel sms = new SendSmsModel("aimodosia", "aimodosiaMsg5", new String[] { date });
    int delivered = sendSmsParallel(sms, mobileNumbers, 2, false);
    logs.logMobileTerminated("AIMODOSIA", "ADMIN", sms, mobileNumbers.size(), delivered);
    sms = null;
    return new SendReport(mobileNumbers.size(), delivered);

}
```

6.6.4 : Αποστολή στοχευμένων μηνυμάτων

Η αποστολή στοχευμένων μηνυμάτων γίνεται με παρόμοιο τρόπο με αυτόν που παρουσιάστηκε στην αποστολή μαζικών μηνυμάτων προς φοιτητές(Ενότητα 6.6.2). Σε αυτήν την υπηρεσία όμως, αντί να δηλωθεί το μάθημα προς αποστολή, το endpoint λαμβάνει απευθείας την λίστα των τηλεφώνων προς αποστολή(Εικόνα 6.28). Η υπηρεσία είναι προσβάσιμη από τους χρήστες οι οποίοι έχουν ταυτοποιηθεί με ρόλο STAFF ή ADMIN, και η εφαρμογή βρίσκει αυτόματα μέσω του authentication Token το user ID τους για την πραγματοποίηση της ενέργειας αποστολής.

Εικόνα 6.28

```
@Path("/direct")
@POST
@Produces(MediaType.APPLICATION_JSON)
@Consumes(MediaType.APPLICATION_JSON)
@Secured({Role.STAFF,Role.ADMIN})
public Response sendDirect(SendSmsRequestDirect req) throws URISyntaxException {
    Principal principal = securityContext.getUserPrincipal();
    req.setSender (principal.getName());

    try {
        return Response.ok(mobileTerminated.sendDirect(req)).build();
    } catch (Exception e) {
        return Response.status(Status.INTERNAL_SERVER_ERROR).build();
    }

}
```

```
public class SendSmsRequestDirect {


    private String[] recipients;
    private String sender;
    private String messageId;
    private String[] replacements;
```

6.7 : Database Logs

Η εφαρμογή καταγράφει κάθε είδους ενέργεια αποστολής μηνυμάτων στην τοπική βάση, πέρα από τα logs του συστήματος. Αυτό συμβαίνει για να επιτευχθεί πλήρης έλεγχος της χρήσης της εφαρμογής από τους χρήστες της, είτε πρόκειται για MO είτε για MT υπηρεσίες. Για την βέλτιστη απόδοση του προγράμματος, η καταγραφή των δεδομένων δεν γίνεται συνεχώς στην βάση δεδομένων, αλλά έχει οριστεί να πραγματοποιείται ανά τακτά ορισμένα διαστήματα.

Στην βάση καταγράφονται επίσης και όλα τα μηνύματα DLR που λαμβάνει η εφαρμογή από τον SMS Aggregator μετά από κάθε αποστολή μηνύματος SMS. Προς το παρόν αυτά τα μηνύματα είναι μόνο πληροφοριακά, αλλά μελλοντικά θα μπορούσαν να αξιοποιηθούν για αναγνώριση λανθασμένων/αποκλεισμένων κινητών τηλεφώνων, καθώς και την στατιστική επιτυχία της αποστολής μηνυμάτων της εφαρμογής.

6.7.1 : Βάση δεδομένων

Σύμφωνα με τα παραπάνω δεδομένα, έχουν κατασκευαστεί 3 πίνακες στην MySQL βάση δεδομένων της εφαρμογής, για την καταγραφή αυτών των στοιχείων:

- Πίνακας mobile_originated:

id	mobile	keyword	body	serviceId	messageId	replacements	smsstatus	time
178	6973256967	VATHMOS	KSEXWR ΔΙΚΤΥΑ	gradeServiceTEITHE	gradeServiceTEITHEMsg2	Δεξιότητες Επικοινωνίας/Κοιν...	Delivered	2018-05-22 14:44:32
179	6973256967	VATHMOS	KSEXWR ΔΙΚΤΥΑ Η/Υ	gradeServiceTEITHE	gradeServiceTEITHEMsg2	Δίκτυα Η/Υ, 5.0, 6.5	Delivered	2018-05-22 14:44:32
180	6973256967	VATHMOS	ΔΙΑΚΡΙΤΑ	gradeServiceTEITHE	gradeServiceTEITHEMsg1	Διακριτά Μαθηματικά, 7.0	Delivered	2018-05-22 15:00:00

- Πίνακας mobile_terminated:

id	list	user	serviceId	messageId	replacements	time	sentTo	received
235	AIMODOSIA	ADMIN	aimodosia	aimodosia...	test 5	2018-05-09 18:40:00	4	4
236	1269-M201	dimiderv@stef	assignmentAnnoun...	assignme...	Αποθήκες Δεδομένων – Εξόρυξη Πληροφορίας, ΤΕΤΑΡΤ...	2018-05-09 19:40:00	0	0
237	269-5603	dimiderv@stef	moodleAnnounce...	moodleAn...	Προσοχή: Αγνοήστε αυτό το sms, είναι ΤΕΣΤ νέας υπη...	2018-05-09 19:50:00	161	161

- Πίνακας dlrs:

id	serviceId	recipient	status	error	time
1	serviceid	6973256967	DELIVRD	ok	2017-07-28 15:27:25
2	gradeService	6973256967	ERROR	80	2017-07-28 15:27:25
3	gradeService	6973256967	DELIVRD	NULL	2017-07-28 15:27:25

Σημειώσεις για τους παραπάνω πίνακες:

- Τα πεδία id είναι auto-increment.
- Τα πεδία time προσθέτονται αυτόματα, χρησιμοποιόντας το CURRENT_TIMESTAMP σε οποιαδήποτε νέα εγγραφή ή update.

6.7.2 : Χρονοπρογραμματισμένη καταγραφή

Η καταγραφή στην βάση γίνεται μέσω της κλάσης ScheduledBackupLogs(Εικόνα 6.29). Πρόκειται για μια κλάση Singleton, που σημαίνει ότι δημιουργείται μόνο ένα instance της κατά την λειτουργία του εφαρμογής. Η κλάση παρέχει διεπαφές μέσω των οποίων οι διάφορες μέθοδοι της εφαρμογής μπορούν να προσθέτουν αντικείμενα καταγραφής στους αντίστοιχες ArrayList της κλάσης.

Μέσω του annotation bean @Schedule στην μέθοδο backup, η εφαρμογή έχει ρυθμιστεί να αδειάζει τις λίστες με τα αντικείμενα καταγραφής(Εικόνα 6.30)στην βάση στο $10^{ο}$ λεπτό κάθε ώρας. Μέσω του annotation @PreDestroy, η μέθοδος backup καλείται κατά τον εκάστοτε τερματισμό της εφαρμογής με σκοπό την αποφυγή απώλειας δεδομένων.

Εικόνα 6.29

```
@Singleton
public class ScheduledBackupLogs {

    @Resource(lookup = "jdbc/localdb")
    DataSource localdb;

    private ArrayList<DlrRequestModel> deliveryReports = new ArrayList<DlrRequestModel>();
    private ArrayList<MobileOriginatedLogs> _MOLogs = new ArrayList<MobileOriginatedLogs>();
    private ArrayList<MobileTerminatedLogs> _MTLogs = new ArrayList<MobileTerminatedLogs>();

    @Schedule(minute = "*/10", hour = "*")
    @PreDestroy
    public void backup() {
        System.out.println("Backing up..");
        backupMO();
        backupMT();
        backupDlrs();

    }
```

Εικόνα 6.30

```
private void backupMT() {
    if (_MTLogs.isEmpty())
        return;
    try (Connection conn = localdb.getConnection();
            PreparedStatement stmt = conn.prepareStatement(
                    "INSERT INTO mobile_terminated(list,user,serviceId,messageId,replacements,sentTo,received) values (?,?,?,?,?,?,?)
        for (MobileTerminatedLogs mt : _MTLogs) {
            stmt.setString(1, mt.getList());
            stmt.setString(2, mt.getUser());
            stmt.setString(3, mt.getServiceId());
            stmt.setString(4, mt.getMessageId());
            stmt.setString(5, mt.getReplacements());
            stmt.setInt(6, mt.getSentTo());
            stmt.setInt(7, mt.getReceived());
            stmt.executeUpdate();
        }
        System.out.println("MobileTerminated Logs Backed Up");
        conn.close();
    } catch (Exception e) {
        e.printStackTrace();
        System.out.println("MobileTerminated Logs Backup Interrupted");
    }
    _MTLogs.clear();

}
```

6.8 : Υποσύστημα αυθεντικοποίησης

Οι υπηρεσίες που είναι διαθέσιμες στην εφαρμογή είναι προστατευμένες από κακόβουλη χρήση μέσω της απαίτησης αυθεντικοποίησης και ταυτοποίησης των χρηστών σε κάθε βήμα. Αυτό επιτυγχάνεται μέσω της προστασίας των endpoint που εκτελούν τις διάφορες διεργασίες της εφαρμογής, με την χρήση του authentication token όπως περιγράφεται και στην ενότητα παρουσίασης του API (6.3). Η προστασία αυτή μπορεί να οριστεί σε ολόκληρες κλάσεις ή μεμονωμένα σε μεθόδους μέσω των custom annotation @Secured, που χρησιμοποιεί το authentication token που λαμβάνεται κατά την αρχική αυθεντικοποίηση του χρήστη μέσω της ιστοσελίδας.

6.8.1 : Αυθεντικοποίηση χρήστη

Η αυθεντικοποίηση των χρηστών και η απόκτηση του authentication token πραγματοποιείται από το endpoint /service/authentication. Η κλάση που υλοποιεί το endpoint ονομάζεται Authentication(Εικόνα 6.31) και εκτός απο τον έλεγχο στοιχείων και παραχώρηση πρόσβασης στον χρήστη, προσθέτει και

περεταίρω στοιχεία στην μνήμη της εφαρμογής για τον μεταγενέστερο έλεγχο της πρόσβασης του χρήστη.

Αρχικά, η μέθοδος authenticateUser παίρνει ως παράμετρο το username και το password του χρήστη. Μέσω της μεθόδου authUser, γίνεται προσπάθεια για την ταυτοποίηση των στοιχείων που δώθηκαν, καθώς και εύρεση του επιπέδου πρόσβασης του χρήστη. Σε πρώτο στάδιο γίνεται προσπάθεια για αυτόν τον έλεγχο στην βάση της πυθίας, μέσω ενός view το οποίο σε περίπτωση επιτυχίας επιστρέφει τον ρόλο του χρήστη σύμφωνα με τα στοιχεία που δόθηκαν (staff ή stud). Σε περίπτωση αποτυχίας, γίνεται μια δεύτερη προσπάθεια στην εφαρμογή LDAP, όπου αρχικά γίνεται ανάκτηση του ρόλου του χρήστη, και στην συνέχεια προσπάθεια ελέγχου του κωδικού πρόσβασης στο username που έχει δοθεί.

Στην συνέχεια παράγεται το token αυθεντικοποίησης που θα χρησιμοποιηθεί για τις διάφορες απαιτήσεις ασφαλείας της εφαρμογής(Περιγράφεται πιο αναλυτικά παρακάτω (Ενότητα 6.8.2). Ο χρήστης προστίθεται στην λίστα με τους συνδεδεμένους χρήστες μέσω της κλάσης AuthenticatedUsers, και τελικά επιστρέφεται η απάντηση του endpoint, που περιέχει τον ρόλο του χρήστη (admin, staff ή stud) μαζί με το authentication token. Η απάντηση αυτή αποθηκεύεται σε 2 cookies στην ιστοσελίδα, στο role και στο token που χρησιμοποιούνται σε κάθε μεταγενέστερο αίτημα.

Εικόνα 6.31

```
@Path("/authentication")
public class Authentication {
    @EJB
    Ldap ldap;
    @EJB
    PithiaLogin pithiaLogin;
    @EJB
    AuthenticatedUsers authUsers;

    @POST
    @Produces(MediaType.APPLICATION_JSON)
    @Consumes(MediaType.APPLICATION_JSON)
    public Response authenticateUser(User user) throws URISyntaxException {
        try {
            user.setRole(authUser(user));
            user.setToken(Token.issueToken(user));
            authUsers.put(user);
            return Response.ok(user).build();

        } catch (Exception e) {
            return Response.status(Response.Status.UNAUTHORIZED).build();

        }
    }

    private String authUser(User user) throws Exception {
        try {
            return pithiaLogin.performAuthentication(user);
        } catch (Exception ex) {
            String role= ldap.performAuthentication(user).toLowerCase();
            return role;
        }
    }
}
```

6.8.2 : Δημιουργία και αξιοποίηση JSON Web Token

Η δημιουργία του authentication token γίνεται μέσω της βιβλιοθήκης JSON Web Token. Πρόκειται για ένα hash 176 χαρακτήρων, στο οποίο μπορούν να κωδικοποιηθούν διάφορα δεδομένα με ένα συμμετρικό κλειδί, τα οποία μπορούν να αποδικωποιηθούν με πολύ γρήγορο τρόπο έχοντας αυτό το κλειδί.

Η παραγωγή του token πραγματοποιείται κατά την αυθεντικοποίηση του χρήστη. Εφόσον είναι επιτυχής, παράγεται ένα τέτοιο token για χρήση, και του επιστρέφεται. Αυτό γίνεται μέσω της κλάσης Token(Εικόνα 6.32) η οποία παράγει το token,θέτοντας στην πληροφορία το username του χρήστη (θα χρειαστεί αργότερα για την αντιστοίχιση με τον ρόλο του) καθώς και τον χρόνο λήξης του(που έχει οριστεί στις 2 ώρες). Αφού οριστούν, κωδικοποιείται με το ορισμένο signingKey και επιστρέφεται στον χρήστη.

Εικόνα 6.32

```
public static String issueToken(User u) {
    String jwtToken = createJWT("ATEITH SMS SERVER", u.getUsername());
    return jwtToken;
}
private static String createJWT(String issuer, String subject) {
    // The JWT signature algorithm we will be using to sign the token
    SignatureAlgorithm signatureAlgorithm = SignatureAlgorithm.HS256;
    String signingKey = "dummySigningKey";
    long nowMillis = System.currentTimeMillis();
    Date now = new Date(nowMillis);
    long ttlMillis = 7200000;// 2 hours

    // Let's set the JWT Claims
    JwtBuilder builder = Jwts.builder().setIssuedAt(now).setSubject(subject).setIssuer(issuer)
            .signWith(signatureAlgorithm, signingKey);

    // if it has been specified, let's add the expiration
    if (ttlMillis >= 0) {
        long expMillis = nowMillis + ttlMillis;
        Date exp = new Date(expMillis);
        builder.setExpiration(exp);
    }

    // Builds the JWT and serializes it to a compact, URL-safe string
    return builder.compact();
}
```

Αργότερα, θα χρειαστεί να γίνει αποδικωποίηση του token. Η μέθοδος validate(Εικόνα 6.33), ελέγχει αυτόματα την εγκυρότητα ή όχι του token που έχει λάβει. Σε περίπτωση που το token δεν έχει παραχθεί από την εφαρμογή ή έχει λήξει, πετάει το ανάλογο exception. Σε περίπτωση που από το token είναι έγκυρο, επιστρέφεται το username στον οποίο ανήκει το token.

Εικόνα 6.33

```
public static String validateToken(String token) throws Exception {
    return parseJWT(token);
}
private static String parseJWT(String jwt) {

    // This line will throw an exception if it is not a signed JWS (as
    // expected)
    Claims claims = Jwts.parser().setSigningKey("dummySigningKey").parseClaimsJws(jwt).getBody();
    return claims.getSubject();

}
```

6.8.3 : Ασφάλεια μεθόδων μέσω Interceptor

Παρόλου που η εφαρμογή χρησιμοποιεί token-based αυθεντικοποίηση, δεν στηρίζεται στον standard μηχανισμό ασφαλείας της Java EE που ορίζεται από το web.xml. Η λύση που χρησιμοποιείται είναι custom χρησιμοποιώντας μόνο το JAX-RS API, και μπορεί να λειτουργήσει σε οποιαδήποτε εφαρμογή το υποστηρίζει.

Ο client που χρησιμοποιεί το ασφαλές endpoint, πρέπει να ορίσει το HTTP Authorization Header του request. Για παράδειγμα :

Authorization: Bearer: eyJhbGciOiJIUzI1NiJ9.eyJpYXQiOjE1MjY0ODM4ODEsInN1Yil6ImRpbWlkZXJ2QHN0ZWYiLCJpc3MiOiJBVEVJVEggU01TIFNFUlZFUiIsImV4cCI6MTUyNjQ5MTA4MX0.rD0hiO4kdkPGKhjLOdyKJuYvzGVwzNSnflHFe6E77rc

Το JAX-RS παρέχει τα meta-annotation @NameBinding , τα οποία χρησιμοποιούνται για την παραγωγή άλλων annotation που μπορούν να χρησιμοποιηθούν ως filters ή interceptors σε κλάσεις η μεθόδους. Στην εφαρμογή ορίζεται το annotation @Secured(Εικόνα 6.34).

Εικόνα 6.34

```
@NameBinding
@Retention(RetentionPolicy.RUNTIME)
@Target({ElementType.TYPE, ElementType.METHOD})
public @interface Secured {
    Role[] value() default {};
}
```

Το παραπάνω annotation θα οριστεί πάνω από μία filter κλάση, η οποία θα κάνει intercept τα διάφορα request των μεθόδων που έχουν ορισμένο το @Secure. Αρχικά καλείται η AuthenticationFilter κλάση(Εικόνα 6.35), η οποία ελέγχει την ύπαρξη ή όχι έγκυρου token, με priority 1000 (η μέθοδος του endpoint έχει πάντα priority 5000). Σε περίπτωση μη έγκυρου token, επιστρέφεται το response 401: Unauthorized.

Εικόνα 6.35

```
@Secured
@Provider
@Priority(1000)
public class AuthenticationFilter implements ContainerRequestFilter {
    public void filter(ContainerRequestContext requestContext) throws IOException {
        // Get the HTTP Authorization header from the request
        String authorizationHeader =
            requestContext.getHeaderString(HttpHeaders.AUTHORIZATION);

        // Check if the HTTP Authorization header is present and formatted correctly
        if (authorizationHeader == null || !authorizationHeader.startsWith("Bearer ")) {
            throw new NotAuthorizedException("Authorization header must be provided");
        }
        // Extract the token from the HTTP Authorization header
        String token = authorizationHeader.substring("Bearer".length()).trim();
        try {
            String username=Token.validateToken(token);
            requestContext.setSecurityContext(new MySecurityContext(username));
        } catch (Exception e) {
            requestContext.abortWith(
                Response.status(Response.Status.UNAUTHORIZED).build());
        }

    }
```

Στην συνέχεια καλείται η δεύτερη filter κλάση, η AuthorizationFilter. Η συγκεκριμένη κλάση έχει priority 2000, και καλείται πάντα μετα την AuthenticationFilter. Αρχικά, κάνει ανάκτηση των ρόλων που έχουν δηλωθεί μέσω του annotation στην μέθοδο που θα γίνει secure (π.χ. @Secured(Role.Staff)) . Στην συνέχεια ελέγχει εάν ο χρήστης που κάλεσε την μέθοδο έχει πρόσβαση σε αυτή, μέσω του ρόλου που του έχει ανατεθεί(Εικόνα 6.37). Η εύρεση του ρόλου του γίνεται μέσω της κλάσης AuthenticatedUsers(), στην οποία περιέχονται όλοι οι αυθεντικοποιημένοι χρήστες μαζί με το επίπεδο πρόσβασής τους (role). Κάθε χρήστης μπορεί να παραμείνει συνδεμένος στην εφαρμογή έως 2 ώρες, οπότε και μετά διαγράφεται. Η ανάκτηση του username του χρήστη ώστε να αναζητηθεί στον πίνακα γίνεται μέσω της αποδικωποίησης του token.

Εικόνα 6.37

```
private void checkPermissions(List<Role> allowedRoles, String username
    User user=authUsers.get(username);
    for(Role r:allowedRoles)
        if(user.getRole().equalsIgnoreCase(r.name()))
                return;

    throw new Exception();
}
```

Εικόνα 6.36

```
@Singleton
public class AuthenticatedUsers {
    private Map<String, User>  authUsers;
    @PostConstruct
    public void init(){
        authUsers = ExpiringMap.builder()
                .maxSize(300)
                .expiration(2, TimeUnit.DAYS)
                .build();
    }
    public  void put(User u) {
        authUsers.putIfAbsent(u.getUsername(), u);
    }
    public User get(String username) {
        return authUsers.get(username);
    }
    public boolean contains(String username){
        return authUsers.containsKey(username);
    }
```

Εφόσον δημιουργηθούν οι παραπάνω κλάσεις, μπορεί σε οποιαδήποτε μέθοδο ή κλάση να οριστεί το annotation @Secured. Σε περίπτωση που δεν υπάρχει παράμετρος (π.χ. @Secured(Role.Staff)), αυτό σημαίνει ότι όλοι οι ρόλοι έχουν πρόσβαση. Η παρουσία του annotation σε κάποια μέθοδο κάνει override αυτό που πιθανόν υπάρχει στην κλάση.

6.9 : Server Documentation

Η εφαρμογή είναι εγκατεστημένη πάνω σε ένα virtual machine, το οποίο βρίσκεται και ανήκει στο κέντρο δικτύου (NOC) του ΑΤΕΙ/Θ. Το συγκεκριμένο μηχάνημα έχει ως λειτουργικό σύστημα το Debian GNU/Linux 8 και είναι προσβάσιμο μέσω SSL από το εσωτερικό δίκτυο του ιδρύματος, στην τοποθεσία 195.251.120.230.

6.9.1 : Διαχείριση Payara Server

Ο application server είναι εγκατεστημένος στο μηχάνημα στην τοποθεσία /home/noc/payara41
Η εκκίνηση του γίνεται μέσω του εκτελέσιμου αρχείου payara41/bin/admin ,εκτελώντας το με δικαιώματα διαχειριστή (Εικόνα 6.38).

Επίσης, μέσω ενός cron job, ο server είναι προγραμματισμένος να επανεκκίνηται κάθε ημέρα στις 05:00, για αποφυγή σφαλμάτων.

Εικόνα 6.38

```
noc@vmGNUGK:~/payara41/bin$ sudo ./asadmin stop-domain
Waiting for the domain to stop .
Command stop-domain executed successfully.
noc@vmGNUGK:~/payara41/bin$ sudo ./asadmin start-domain
Waiting for domain1 to start .........................
Successfully started the domain : domain1
domain  Location: /home/noc/payara41/glassfish/domains/domain1
Log File: /home/noc/payara41/glassfish/domains/domain1/logs/server.log
Admin Port: 4848
Command start-domain executed successfully.
noc@vmGNUGK:~/payara41/bin$
```

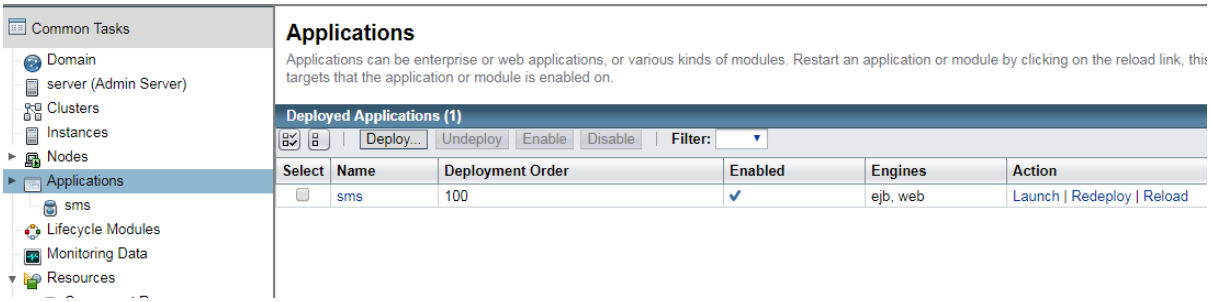
6.9.2 : Διαχειριστική Διεπαφή

Ο Payara Server δίνει την δυνατότητα για την διαχείριση του μέσω ενός διαχειριστικού web-interface, που βρίσκεται ενσωματωμένο στην έκδοση που παρέχεται. Αυτό είναι προσβάσιμο μέσω του port : 4848. Το συγκεκριμένο διαχειριστικό απαιτεί διαπιστευτήρια για τον διαχειριστή, τα οποία ορίζονται μέσω κονσόλας, χρησιμοποιώντας την εντολή *./asadmin change-admin-password* . Ο application server δεν υποστηρίζει την απομακρυσμένη πρόσβαση από προεπιλογή, αλλά για τις ανάγκες της εφαρμογής έχει ρυθμιστεί να την επιτρέπει. Αυτό γίνεται μέσω της εντολής asadmin enable-secure-admin.

6.9.3 : Εγκατάσταση της εφαρμογής

Ο πιο εύκολος τρόπος να γίνει deploy η εφαρμογή είναι μέσω της διαχειριστικής κονσόλας.
Το αρχείο .WAR που έχει παραχθεί προηγουμένως, μπορεί είτε να ανέβει μέσω FTP στο VPN μηχάνημα μας και να χρησιμοποιηθεί απευθείας τοπικά από τον application server, είτε να γίνει upload και deploy αμέσως μετά μέσω του διαχειριστικού. (Εικόνα 6.39,Εικόνα 6.40)

Εικόνα 6.39



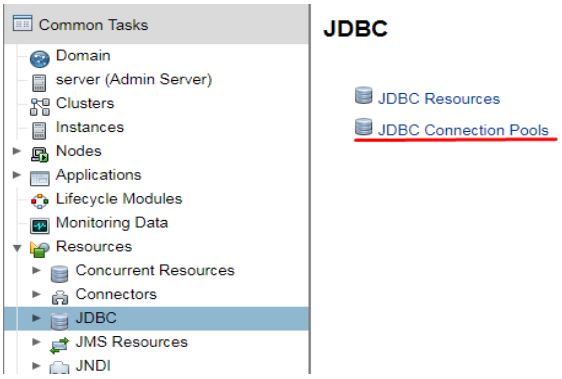
Εικόνα 6.40

Το Web Application ξεκινάει απευθείας με το πάτημα του πλήκτρου ‘OK’, χωρίς να χρειάζεται κάποια επανεκκίνηση του application server.Με τον ίδιο τρόπο, πατώντας το ‘Redeploy’ μπορούν να ανεβούν και να γίνουν deploy οι νέες εκδόσεις της εφαρμογής. Πατώντας το ‘Reload’, γίνεται επανεκκίνηση του web application (το οποίο είναι χρήσιμο σε περίπτωση έχει γίνει κάποια ανανέωση στα configurations της εφαρμογής, και δεν είναι αναγκαίο να επανεκιννηθεί ολόκληρος ο application server)

6.9.4 : Συνδέσεις βάσεων δεδομένων

Η εφαρμογή χρησιμοποιεί JDBC Connection pools για τις συνδέσεις της με τις βάσεις που επικοινωνεί. Για να μπορεί να τα χρησιμοποιήσει, αυτά πρέπει να οριστούν στο JNDI μέσα από τον application server. Αρχικά, πρέπει να δημιουργηθεί η σύνδεση στην βάση. Ο πιο εύκολος τρόπος είναι μέσα από το διαχειριστικό του application server(Εικόνα 6.41), αν και μπορεί να γίνει μέσω επεξεργασίας του domain.xml που βρίσκεται στον φάκελο config του domain.

Εικόνα 6.41



Παρακάτω (Εικόνα 6.42,Εικόνα 6.43) παρουσιάζεται ένα παράδειγμα ρυθμίσεων για δημιουργία νέων συνδέσεων, μέσω της υπάρχουσας σύνδεση στην βάση του ιδρύματος Πυθία.

Εικόνα 6.42

Εικόνα 6.43

Το connection validation χρησιμοποιείται έτσι ώστε να υπάρχει δυνατότητα επανασύνδεσης σε περίπτωση που χαθεί η σύνδεση με την βάση. Επίσης, πρέπει να χρησιμοποιούνται ως driver DataSource κλάσεις.

Για τις βάσεις που χρησιμοποιούνται ,και για την εύρυθμη λειτουργία του προγράμματος πρέπει να οριστούν κάποια επιπλέον properties σε κάθε connection:

SQL-Server :

- JDBC30DataSource = true (Υποχρεωτικό καθώς ο JTDS driver που χρησιμοποιείται από τον Payara δεν υποστηρίζει preparedStatements από προεπιλογή)
- xaEmulation = true (Για να λειτουργεί ως DataSource, και έτσι είναι προσβάσιμο ως connection pool)

MySQL Server:

- allowMultiQueries=true (Για να μπορούμε να τρέχουμε πολλαπλά query μέσω των preparedStatement)
- useUnicode=true&characterEncoding=utf8 (Για να διαβάζουμε και να γράφουμε unicode)
- rewriteBatchedStatements=true (Για batching των παρόμοιων statement)

Μετά την εισαγωγή του εκάστοτε JDBC Connection, πρέπει να γίνει σύνδεση με σε ένα JDBC Resource(Εικόνα 6.44), έτσι ώστε να είναι προσβάσιμο από το πρόγραμμα μέσω του JNDI Lookup.

Εικόνα 6.44

6.9.5 : Server Logs

Ο Payara Server καταγράφει πληροφορίες σχετικά με τα συμβάντα που συμβαίνουν και καταγράφει αυτές τις πληροφορίες χρησιμοποιώντας τον μηχανισμό καταγραφής σε αρχεία καταγραφής (Log files).

Τα log files καταγράφονται από προεπιλογή στα ακόλουθα αρχεία: (Πίνακας 6.2)

Πίνακας 6.2

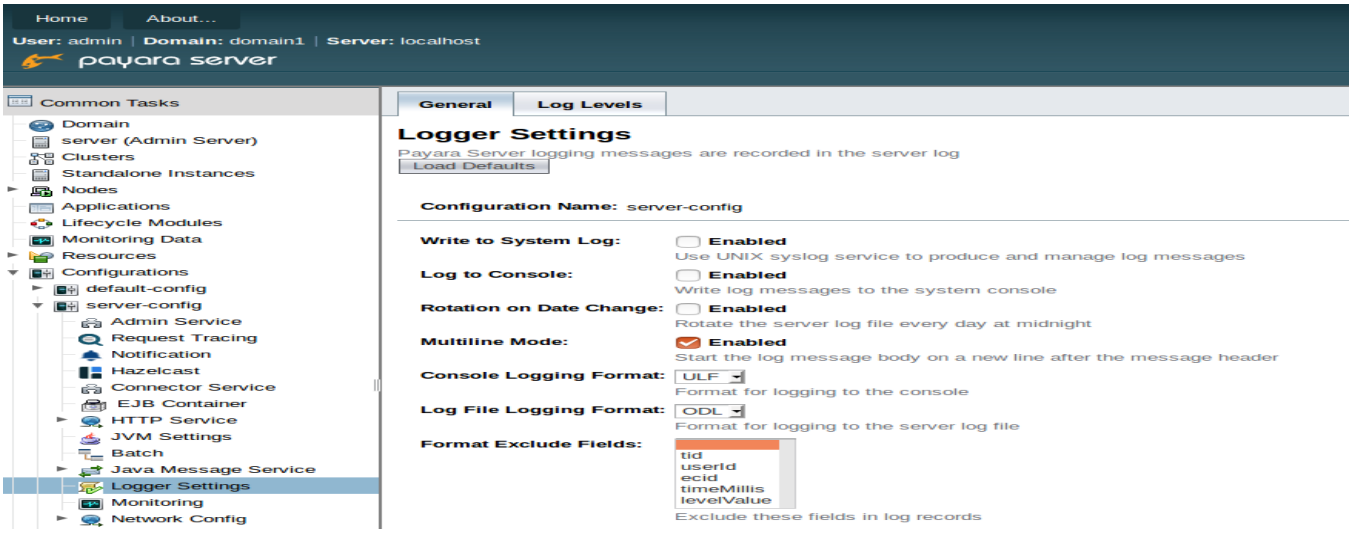
Instance	Αρχείο καταγραφής και τοποθεσία
Domain Admin Server (DAS)	domain-dir/logs/server.log

Standalone server instance	instance-dir/logs/server.log
Cluster instance	instance-dir/logs/cluster.log

Εάν κάποιο instance ξεκινήσει χρησιμοποιώντας την επιλογή -verbose, συνεχίζει να εκτελείται στο προσκήνιο και καταγράφει επίσης πληροφορίες στην κονσόλα. Ο Payara Server χρησιμοποιεί το Java Logging (JUL) για τη μορφοποίηση και την παραγωγή αρχείων καταγραφής. Το προεπιλεγμένο αρχείο ρυθμίσεων isdomain-dir / config / logging.properties.

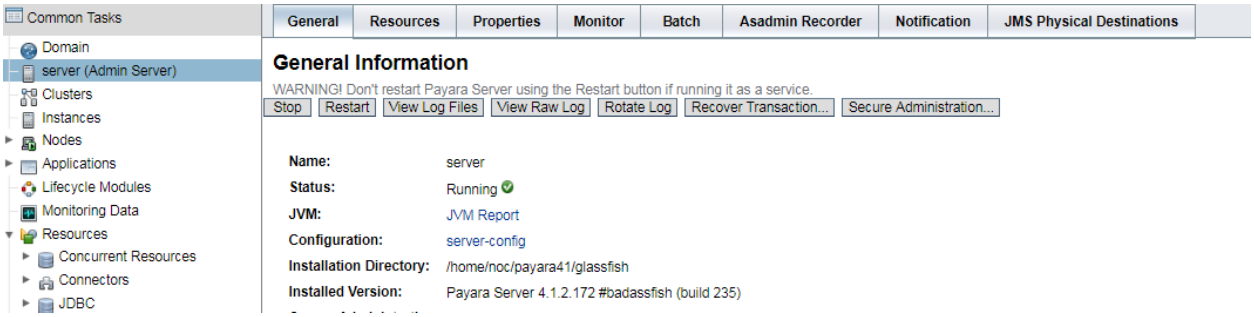
Η καταγραφή μπορεί να ρυθμιστεί με εντολές asadmin (set-log-attributes, set-log-file format, set-log-levels, delete-log-levels, ...). Μπορεί επίσης να ρυθμιστεί στην Κονσόλα διαχειριστή, στις Ρυθμίσεις (Configurations), στην ενότητα Ρυθμίσεις καταγραφικού (Logger Settings) (Εικόνα 6.45).

Εικόνα 6.45



Τα log files είναι προσβάσιμα, είτε μέσω του διαχειριστικού(Εικόνα 6.46), είτε μέσω του αρχείου server.log, που βρίσκεται στην τοποθεσία domain/logs/server.log.

Εικόνα 6.46



Σε περίπτωση που ο server υποστεί σε κάποιο απροσδόκητο σφάλμα, αυτό δεν καταγράφεται Για αυτόν τον σκοπό, πρέπει να θέσουμε ένα επιπλέον log configuration για το JVM. Αυτό γίνεται μέσω της προσθήκης των παρακάτω γραμμών (Εικόνα 6.47) στο domain.xml που βρίσκεται στο config φάκελο του domain μας. Πλέον, εμφανίζεται και το αρχείο jvm.log στον φάκελο με τα server logs.

Εικόνα 6.47

```
<jvm-options>-XX:LogFile=${com.sun.aas.instanceRoot}/logs/jvm.log</jvm-options>
<jvm-options>-XX:+LogVMOutput</jvm-options>
```

6.9.6 : SSL

Ο application server κρυπτογραφεί όλη την κίνηση από/προς αυτόν μέσω του πρωτοκόλλου SLL.
Το security certificate που χρησιμοποιούνται έχουν εκδοθεί από το κέντρο δικτύου του TEI, και είναι τα παρακάτω τρία αρχεία:

- **smsserevercert.pem** (περιέχει το πιστοποιητικό του διακομιστή σε μορφή BASE64 και κωδικοποίηση PEM)
- **smsserver.key** (περιέχει το ιδιωτικό κλειδί του παραπάνω πιστοποιητικού σε κωδικοποίηση PEM)
- **cachain.pem** (περιέχει όλα τα πιστοποιητικά, το ένα κάτω από το άλλο, ξεκινώντας από το πιστοποιητικό της Αρχής έκδοσης του πιστοποιητικού εξυπηρετητή μέχρι κάποια Κορυφαία Αρχή Πιστοποίησης)

Ο Payara και πολλές άλλες εφαρμογές Java αναμένουν να ανακτήσουν πιστοποιητικά SSL / TLS από ένα Java Key Store (JKS). Αυτές συνήθως έχουν προεγκατεστημένο ένα Keytool που βοηθάει να:

- δημιουργηθεί ένα νέο JKS με ένα νέο ιδιωτικό κλειδί..
- δημιουργηθεί ένα αίτημα υπογραφής πιστοποιητικού (CSR) για το ιδιωτικό κλειδί σε αυτό το JKS.
- εισαχθεί ένα πιστοποιητικό που έχει ληφθεί για αυτήν την (CSR) στο JKS.

Το Keytool δεν μας επιτρέπει την εισαγωγή ενός υπάρχον ιδιωτικού κλειδιού για το οποίο υπάρχει ήδη ένα πιστοποιητικό. Για αυτόν τον λόγο πρέπει να ακολουθηθεί η παρακάτω διαδικασία.
Αρχικά, το κλειδί και το πιστοποιητικό να μετατραπούν σε μορφή .der
Αυτό μπορεί να γίνει μέσω της βιβλιοθήκης του OpenSSL και τρέχοντας τις παρακάτω εντολές:
openssl pkcs8 -topk8 -nocrypt -in smsserver.key -inform PEM -out key.der -outform DER

openssl x509 -in smsserevercert.pem -inform PEM -out cert.der -outform DER

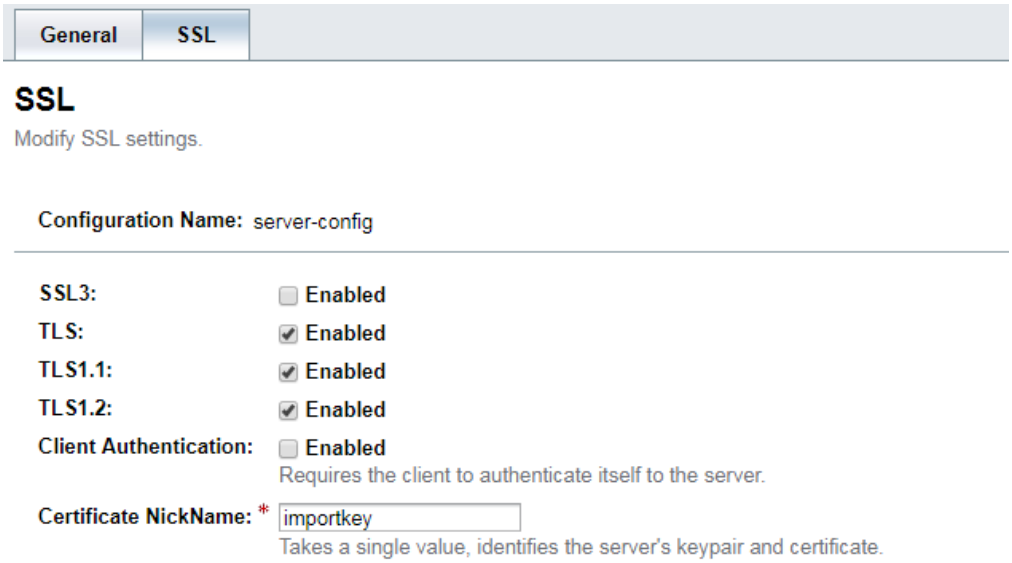
Πλέον, εφόσον υπάρχουν αυτά τα 2 αρχεία, μπορεί να γίνει χρήση μιας βοηθητικής java κλάσης, η οποία θα τα κάνει import στο JKS. Αυτή η κλάση λέγεται ImportKey.class, και βρίσκεται στον φάκελο SSL στο /home/noc. Τρέχοντας το java αρχείο(Εικόνα 6.48) γίνεται το import στο JKS.

Εικόνα 6.48

```
noc@vmGNUGK:~$ java ImportKey key.der cert.der
Using keystore-file : /home/user/keystore.ImportKey
One certificate, no chain.
Key and certificate stored.
Alias:importkey  Password:importkey
```

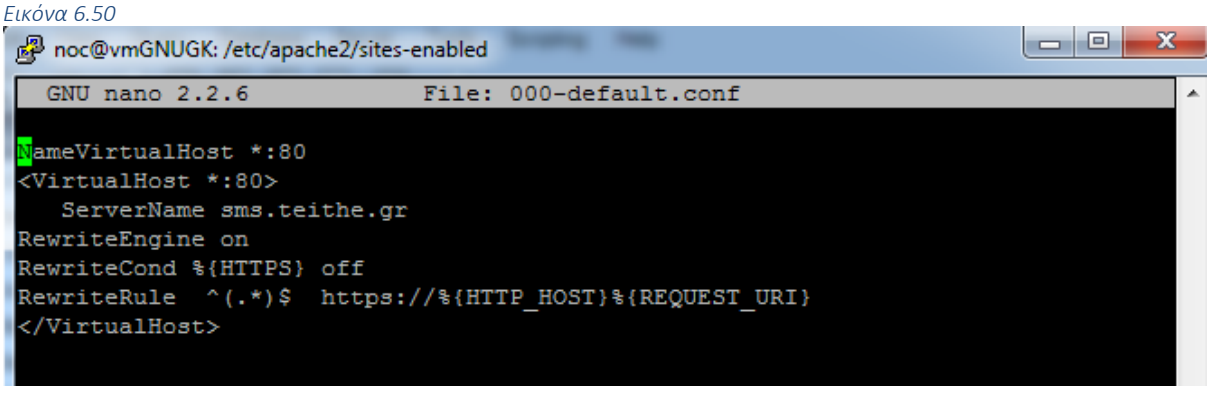
Στην συνέχεια, πρέπει να οριστεί το συγκεκριμένο πιστοποιητικό στην secure port του Payara. Αυτό επιτυγχάνεται μέσω της διαχειριστικής κονσόλας(Εικόνα 6.49).

Εικόνα 6.49



Τέλος, πρέπει να οριστεί όλη η κίνηση του server να περνάει από την ασφαλή πόρτα του application server. Αυτό μπορεί να επιτευχθεί χρησιμοποιώντας την λειτουργία Virtual Host του Apache που υπάρχει πάνω στο μηχάνημα (Εικόνα 6.50). Πλέον, κάθε request στην πόρτα 80 του μηχανήματος, κάνει redirect στην secure πόρτα 443

Εικόνα 6.50



6.9.7 : Ιστοσελίδα

Η ιστοσελίδα είναι ξεχωριστή από την εφαρμογή, κάτι που σημαίνει ότι δεν γίνεται deploy μαζί με το .war αρχείο. Εξαρτάται όμως από τον application server για τα δεδομένα της, και πρέπει να γίνει deploy στον ίδιο application server από τον οποίο εξαρτάται, για λόγους ασφαλείας (δεν επιτρέπεται π.χ. να στηθεί πάνω σε έναν Apache και να επικοινωνεί με τον application server).

Τα αρχεία της ιστοσελίδας είναι εγκατεστημένα στον φάκελο payara41/glassfish/domains/domain1/docroot. Οποιαδήποτε αλλαγή στον κώδικα αυτών των αρχείων θα αποφέρει άμεση αλλαγή στην εμφάνιση/λειτουργικότητα της ιστοσελίδας, χωρίς να χρειάζεται κάποια επανεκκίνηση του application server. Η ιστοσελίδα είναι προσβάσιμη μέσω του domain *https://sms.teithe.gr* .

Κεφάλαιο 7 : Επίλογος

Ο στόχος της παρούσας πτυχιακής ήταν να δημιουργηθεί μια αξιόπιστη ,διατηρήσιμη και εύκολα επεκτάσιμη εφαρμογή η οποία θα εξυπηρετεί όλες της υπηρεσίες που παρέχονται από τον SMS Aggregator. Για την παρουσίαση της καθώς και για την αρχική της λειτουργία έχουν υλοποιηθεί και είναι σε λειτουργία αρκετές υπηρεσίες ,οι οποίες είναι ήδη έτοιμες να εξυπηρετήσουν τις ανάγκες της Ακαδημαικής κοινότητας του ΑΤΕΙ/Θ ως προς τον σκόπο που έχουν δημιουργηθεί, ο όποιος είναι πρωταρχικά η παροχή γρήγορης και αξιόπιστης ενημέρωσης προς τα δρώμενα του ΑΤΕΙ/Θ στα οποία είναι ενδιαφερόμενα τα μέλη της.

Η εφαρμογή από την αρχή σχεδιάστηκε και αναπτύχθηκε με σκοπό την επεκτασιμότητα και την διαλειτουργικότητα ως προς τον τρόπο που μπορεί να χρησιμοποιηθεί. Αρχικά, η Ακαδημαϊκή βάση δεδομένων που χρησιμοποιεί (Πυθία) χρησιμοποιείται κατά κόρων από τα περισσότερα Ακαδημαϊκά ιδρύματα της χώρας, οπότε με ελάχιστες διαφοροποιήσεις, μπορεί να γίνει αφομοίωση της εφαρμογής από αυτά. Στην συνέχεια, οι τεχνολογίες που χρησιμοποιούνται είναι ευρέως διαδεδομένες, καθώς τα διάφορα πακέτα που χρησιμοποιούνται είναι ανοιχτού κώδικα και ελευθέρα, ώστε να μπορεί εύκολα να γίνει επέκταση των λειτουργιών της εφαρμογής. Τελικά, η προσθήκη νέων ερωτημάτων/μηνυμάτων γίνεται μέσω ρυθμίσεων σε αρχεία xml, οπότε δεν χρειάζεται τροποποίηση του πηγαίου κώδικα για την επέκταση της εφαρμογής.

Μελλοντικά, πέρα από την επέκταση των λειτουργιών της εφαρμογής, μέσω του API διασύνδεσης της εφαρμογής, υπάρχει δυνατότητα κάποια τρίτη εφαρμογή να χρησιμοποιήσει τις λειτουργίες του sms-server, ενσωματώνοντάς τες πάνω σε κάποια νέα ενιαία πλατφόρμα του ΑΤΕΙ/Θ, ή σε κάποια ήδη υπάρχουσα (pithia ή moodle) για πιο άμεση πρόσβαση στα μέλη του ακαδημαϊκού ιδρύματος.

Κλείνοντας, θα ήθελα να ευχαριστήσω θερμά τους επιβλέποντές μου καθηγητή κ. Δημήτριο Δέρβο και Δρ. Στέφανο Ουγιάρογλου, μέλος ΕΔΙΠ του Τμήματος, για την καθοδήγησή τους ως προς τις λειτουργικές

ανάγκες της εφαρμογής, την σχεδίαση της web εφαρμογής και γενικότερα τις συμβουλές τους ως προς την εκπόνηση αυτής της πτυχιακής εργασίας, καθώς και τον διευθυντή του ΚΔΔ του ΑΤΕΙ/Θ κ. Αναστάσιο Φιλέλη ο οποίος παρείχε όλα τα απαραίτητα εφόδια για την υλοποίηση της εφαρμογής, μέσω παροχής της πρόσβασης στις βάσεις δεδομένων του ιδρύματος και του server στον οποίο φιλοξενείται η εφαρμογή, καθώς και με τις τεχνικές γνώσεις που προσέφερε στο κομμάτι της διαχείρισης της εφαρμογής.

Κεφάλαιο 8 : Βιβλιογραφία

https://aws.amazon.com/blogs/startups/choosing-the-right-programming-language-for-your-startup/

https://shadow-soft.com/open-source-application-servers/

https://info.payara.fish/payara-server-glassfish-comparison

https://www.novell.com/documentation/nw65/web_mysql_nw/data/aj5bj52.html

https://www.progress.com/tutorials/jdbc/jdbc-jdbc-connection-pooling

https://stackify.com/soap-vs-rest/

https://dzone.com/articles/why-im-using-java-ee

http://www.baeldung.com/ant-maven-gradle

https://devopscube.com/list-of-popular-open-source-java-build-tools/

https://www.upwork.com/hiring/development/soap-vs-rest-comparing-two-apis/

https://www.developer.com/java/data/understanding-jdbc-connection-pooling.html

https://www.sequetech.com/ajax-technology/

https://www.tutorialrepublic.com/twitter-bootstrap-tutorial/bootstrap-introduction.php

https://code.tutsplus.com/tutorials/an-introduction-to-cookies--net-12482

https://dzone.com/refcardz/dependency-injection-in-ejb3?chapter=1

http://www.differencebetween.info/difference-between-html-and-jsp

https://el.wikipedia.org/wiki/HTML

https://en.wikipedia.org/wiki/Java_Platform,_Enterprise_Edition

https://en.wikipedia.org/wiki/Database

https://stackoverflow.com/questions/26777083/best-practice-for-rest-token-based-authentication-with-jax-rs-and-jersey

https://stackoverflow.com/questions/12163947/ldap-how-to-authenticate-user-with-connection-details

https://stackoverflow.com/questions/2642777/trusting-all-certificates-using-httpclient-over-https

https://stackoverflow.com/questions/42373642/how-to-authenticate-users-in-jersey

https://stackoverflow.com/questions/12860289/import-ssl-certificate-in-glassfish

http://open.gunet.gr/apis/sms-services/

https://docs.oracle.com/javase/jndi/tutorial/ldap/security/ldap.html

https://docs.oracle.com/javaee/7/api/javax/json/package-summary.html

https://jersey.github.io/

https://github.com/jwtk/jjwt

https://github.com/pingidentity/ldapsdk

https://github.com/jhalterman/expiringmap

https://github.com/milesibastos/jTDS