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

Τμήμα Μηχανικών Πληροφορικής ΑΤΕΙΘ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Υλοποίηση API για διασύνδεση της ηλεκτρονικής γραμματείας φοιτητών με την εφαρμογή Uniway

Του φοιτητή

Μιχαηλούδη Θανάση

Αρ. Μητρώου: 123870

Επιβλέπων καθηγητής

Αδαμίδης Παναγιώτης

Θεσσαλονίκη 2019

ΠΕΡΙΛΗΨΗ

Αντικείμενο της παρούσας πτυχιακής εργασίας είναι η υλοποίηση ενός JSON API για την σύνδεση του ΤΕΙ Θεσσαλονίκης με την ηλεκτρονική πλατφόρμα Uniway. Το Uniway παρέχεται από τη Ελληνική Ακαδημαϊκή Κοινότητα, οργανισμό που συμμετέχουν όλα τα Πανεπιστήμια και τα ΤΕΙ της χώρας και είναι στη διάθεση όλων των μελών της (τα Ελληνικά Πανεπιστήμια και ΤΕΙ). Σκοπός του είναι να προσφέρει ουσιαστικές πληροφορίες για τις σπουδές των φοιτητών έχοντας άμεση πρόσβαση στα επίσημα συστήματα των εκάστοτε ιδρυμάτων. Πιο συγκεκριμένα, οι φοιτητές μπορούν μέσα από την συγκεκριμένη εφαρμογή να ελέγξουν τους βαθμούς τους, να βρούνε τα μαθήματα τα οποία έχουνε δηλώσει στο τρέχον εξάμηνο και το ιστορικό των δηλώσεων τους αλλά και να δούνε τα μαθήματα που περιλαμβάνονται στο πρόγραμμα σπουδών τους. Για το λόγο αυτό, η Ελληνική Ακαδημαϊκή Κοινότητα έχει δημοσιεύσει τον οδηγό σύνδεσης των ιδρυμάτων με το Uniway. Με βάση τον συγκεκριμένο έγγραφο αλλά και τη χρήση του Spring Framework το οποίο είναι ένα ευέλικτο και πανίσχυρο περιβάλλον ανάπτυξης για την δημιουργία ολοκληρωμένων εφαρμογών, έχουμε υλοποιήσει το συγκεκριμένο JSON API.

ABSTRACT

The subject of this dissertation is the implementation of a JSON API for the connection of TEI of Thessaloniki with the Uniway electronic platform. Uniway is provided by GUnet, an organization that participates in all the Universities and TEIs of the country and is available to all its members (Greek Universities and TEIs). Its purpose is to provide meaningful information on student studies by having direct access to the official systems of each institution. In particular, students can, through this application, check their grades, find the lessons they have declared in the current semester and the history of their statements, and also see the courses included in their curriculum. For this reason, GUnet has published the institution linking guide to Uniway. Based on this document and the use of the Spring Framework, which is a flexible and powerful framework for enterprise application deployment, we have implemented this JSON API.

ΕΥΧΑΡΙΣΤΙΕΣ

Ευχαριστώ θερμά όλους τους καθηγητές του Τμήματος Πληροφορικής του Α.Τ.Ε.Ι. Θεσσαλονίκης και το προσωπικό της γραμματείας για τη συνεργασία και βοήθειά τους όλα αυτά τα χρόνια, καθώς και τους συμφοιτητές μου για τις όμορφες στιγμές που μοιραστήκαμε. Ευχαριστώ ιδιαιτέρως τον επιβλέποντα καθηγητή κ. Αδαμίδη Παναγιώτη για την καθοδήγηση και πολύτιμη συμβολή του στην ολοκλήρωση της παρούσας πτυχιακής εργασίας.

ΠΙΝΑΚΑΣ ΠΕΡΙΕΧΟΜΕΝΩΝ

ΚΑΤΑΛΟΓΟΣ ΕΙΚΟΝΩΝ

ΚΑΤΑΛΟΓΟΣ ΠΙΝΑΚΩΝ

ΚΕΦΑΛΑΙΟ 1

ΕΙΣΑΓΩΓΗ

Σκοπός της συγκεκριμένης πτυχιακής εργασίας είναι η υλοποίηση ενός JSON API (Application Programming Interface), το οποίο θα στοχεύει στην σύνδεση του ΤΕΙ Θεσσαλονίκης με την ηλεκτρονική πλατφόρμα Uniway. Το Uniway παρέχεται από την Ελληνική Ακαδημαϊκή Κοινότητα, οργανισμό που συμμετέχουν όλα τα Πανεπιστήμια και τα ΤΕΙ της χώρας και είναι στη διάθεση όλων των μελών της (τα Ελληνικά Πανεπιστήμια και ΤΕΙ). Σκοπός του είναι να προσφέρει ουσιαστικές πληροφορίες για τις σπουδές των φοιτητών έχοντας άμεση πρόσβαση στα επίσημα συστήματα των εκάστοτε ιδρυμάτων. Πιο συγκεκριμένα, οι φοιτητές μπορούν μέσα από την συγκεκριμένη εφαρμογή να ελέγξουν τους βαθμούς τους, να βρούνε τα μαθήματα τα οποία έχουνε δηλώσει στο τρέχον εξάμηνο και το ιστορικό των δηλώσεων τους, αλλά και να δούνε τα μαθήματα που περιλαμβάνονται στο πρόγραμμα σπουδών τους. Η Ελληνική Ακαδημαϊκή Κοινότητα παρέχει ένα έγγραφο μέσω του οποίου αναλύει το προγραμματιστικό περιβάλλον διασύνδεσης της εφαρμογής με το ιδρυματικό Πληροφοριακό Σύστημα Ηλεκτρονικής Γραμματείας Φοιτητών. Το συγκεκριμένο έγγραφο, όπως αναφέρουμε και στο κεφάλαιο 4, παρέχει χρήσιμες πληροφορίες για τις κλήσεις της εφαρμογής, αλλά και τη μορφή των δεδομένων που θα πρέπει αυτή να επιστρέφει .

Μέσα από την υλοποίηση του συγκεκριμένου θέματος θέλουμε να συμβάλουμε στην εξέλιξη του ΤΕΙ. Το Uniway είναι μία συλλογική προσπάθεια, έχει ως στόχο να δώσει λύσεις σε κοινά προβλήματα που αντιμετωπίζουν τα πανεπιστήμια και τα ΤΕΙ της χώρας. Αξιοποιώντας την ευκολία και την αμεσότητα που παρέχει το μέσο, το Uniway έχει σχεδιασθεί για να παρέχει την ελευθερία και την ευελιξία των ανοικτών κοινωνικών δικτύων συνδυάζοντας εξειδικευμένες υπηρεσίες που αφορούν τους φοιτητές,

αντικαθιστώντας έτσι τα ήδη παλαιωμένα συστήματα των ακαδημαϊκών ιδρυμάτων της χώρας .

1.1 ΟΡΓΑΝΩΣΗ ΚΕΙΜΕΝΟΥ

Η συγκεκριμένη ενότητα πραγματεύεται τη διάρθρωση του κειμένου το οποίο αποτελείται από 4 κεφάλαια.

- **Κεφάλαιο 2:** Στο κεφάλαιο αυτό αναφέρουμε τις τεχνολογίες που χρησιμοποιήσαμε για να υλοποιήσουμε την εφαρμογή. Πιο συγκεκριμένα, κάνουμε λόγο για το Spring Framework, τα πλεονεκτήματα του και για το Java Persistence API(JPA).
- **Κεφάλαιο 3:** Στο κεφάλαιο αυτό επικεντρωνόμαστε στην περιγραφή του προβλήματος. Πιο συγκεκριμένα, αναφερόμαστε στην Ελληνική Ακαδημαϊκή Κοινότητα και στο Uniway.
- **Κεφάλαιο 4:** Κάνουμε λόγο για την υλοποίηση του JSON API. Αρχικά, αναφερόμαστε στις κλήσεις της εφαρμογής αλλά και στη μορφή των δεδομένων που επιστρέφονται από αυτή, ενώ στη συνέχεια αναφερόμαστε με περισσότερες λεπτομέρειες στη δομή του κώδικα.
- **Κεφάλαιο 5 :** Στο κεφάλαιο αυτό αναφέρουμε τα συμπεράσματα τα οποία προκύπτουν από την συγκεκριμένη πτυχιακή εργασία.

ΚΕΦΑΛΑΙΟ 2

ΤΕΧΝΟΛΟΓΙΕΣ

Στο τρέχον κεφάλαιο αναλύουμε τις σχετικές τεχνολογίες που έχουμε χρησιμοποιήσει για τη λύση του προβλήματος. Αρχικά, κάνουμε μια γενική αναφορά στους όρους Json, Framework, API και στη συνέχεια αναλύουμε το Spring Framework και το Spring Data JPA. Ειδικότερα, αναφέρουμε τους λόγους για τους οποίος το Spring Framework είναι το πιο διαδεδομένο περιβάλλον ανάπτυξης της Java, τη χρησιμότητα του JPA (Java Persistence API), ενώ στο τέλος κάνουμε αναφορά στις απαραίτητες γνώσεις και εφόδια που χρειάζεται ένας προγραμματιστής για να χτίσει μια εφαρμογή χρησιμοποιώντας το JPA.

2.1 JSON

Σύμφωνα με τη Βικιπαίδεια , JavaScript Object Notation ή Json είναι ένα ανοικτό πρότυπο μορφοποίησης, το οποίο είναι εύκολα κατανοητό από τον άνθρωπο καθώς χρησιμοποιεί κείμενο για τη μετάδοση δεδομένων και συγκεκριμένα ζεύγη χαρακτηριστικών-τιμών. Πρόκειται για ένα πολύ κοινό πρότυπο μορφοποίησης δεδομένων που χρησιμοποιείται για την ασύγχρονη επικοινωνία μεταξύ περιηγητή και διακομιστή.

Η Json υποστηρίζει δύο δομές δεδομένων που χρησιμοποιούνται ευρέως (μεταξύ των γλωσσών προγραμματισμού).

- Μια συλλογή από ζεύγη (όνομα, τιμή). Όπως αντικείμενο, εγγραφή, δομή, λεξικό, πίνακας κατακερματισμού, λίστα κλειδιών ή συσσωματωμένος πίνακας.

- Μια διατεταγμένη λίστα τιμών.

Η Json είναι στενά συνδεδεμένη με τη JavaScript. Ωστόσο, το μεγάλο της πλεονέκτημα είναι ότι πολλές γλώσσες προγραμματισμού μπορούν να δημιουργήσουν και να διαβάσουν αυτή τη μορφή. Το προτέρημα αυτό είναι ένας από τους λόγους για τους οποίους η Json είναι πολύ δημοφιλής ως τρόπο αποθήκευσης, ανάλυσης, ανάγνωσης και ανταλλαγής πληροφοριών σε εφαρμογές και σε υπηρεσίες ιστού.

Η Json δεν είναι μέρος της JavaScript, αλλά έχει υιοθετήσει τον τρόπο γραφής της. Ωστόσο, υπάρχουν σημαντικές διαφορές ανάμεσα στο συντακτικό τους. Στην πρώτη όλες οι λέξεις κλειδιά πρέπει να βρίσκονται μέσα σε εισαγωγικά, ενώ στη δεύτερη οι λέξεις κλειδιά δεν χρειάζεται να βρίσκονται σε εισαγωγικά. Ένας άλλος τομέας όπου η Json διαφέρει από τα αντικείμενα JavaScript είναι οι τύποι τιμών που μπορεί να αποθηκεύσει. Οι βασικοί τύποι τους οποίους χρησιμοποιεί η Json είναι οι εξής :

- Number
- String
- Boolean(true, false)
- Array (Μια διατεταγμένη λίστα από τιμές, χωρισμένες από κόμμα και περικυκλωμένες από {}. Οι τιμές δεν χρειάζεται να είναι του ίδιου τύπου)
- Object (Μια μη ταξινομημένη συλλογή από ζεύγη, χωρισμένες από κόμμα και περικυκλωμένες από {}. Οι τιμές δεν χρειάζεται να είναι του ίδιου τύπου. Ανάμεσα σε κάθε ζεύγος κλειδιού τιμής υπάρχει ο χαρακτήρας :)
- Null

Σύμφωνα με μια αναφορά της επίσημης σελίδας Json με τίτλο *Json: The Fat-Free Alternative to Xml* τόσο η Xml όσο και η Json έχουν ευρεία υποστήριξη για τη δημιουργία, την ανάγνωση και την αποκωδικοποίηση. Η Json προωθείται ως μια εναλλακτική λύση αντί της Xml, καθώς μορφοποιεί τα δεδομένα με ένα πιο συνοπτικό τρόπο ο οποίος εξοικονομεί εύρος ζώνης και βελτιώνει τους χρόνους απόκρισης κατά την αποστολή των μηνυμάτων από και προς το διακομιστή. Αντίθετα η Xml έχει χρησιμοποιηθεί για την περιγραφή δομημένων δεδομένων και τη σειριοποίηση (serialization) αντικειμένων. Υπάρχουν διάφορα πρωτόκολλα που βασίζονται στη Xml και αντιπροσωπεύουν το ίδιο είδος δομών δεδομένων με τη Json για τον ίδιο τύπο ανταλλαγής δεδομένων. Όταν, υπάρχουν δεδομένα που κωδικοποιούνται στη Xml το αποτέλεσμα είναι συνήθως

μεγαλύτερο από την ισοδύναμη κωδικοποίηση στη Json. Ωστόσο, αν τα δεδομένα είναι συμπιεσμένα χρησιμοποιώντας έναν αλγόριθμο όπως το zip, υπάρχει μικρή διαφορά.

Το σχήμα 2.1 δείχνει ένα παράδειγμα αναπαράστασης δεδομένων στη Xml και Json αντίστοιχα.

XML

```
<empinfo>
   <employees>
      <employee>
         <name>James Kirk</name>
         <age>40></age>
      </employee>
      <employee>
         <name>Jean-Luc Picard</name>
         <age>45</age>
      </employee>
      <employee>
         <name>Wesley Crusher</name>
         <age>27</age>
      </employee>
   </employees>
</empinfo>
```

JSON

```
{  "empinfo" :
      {
            "employees" :  [
            {
                  "name" : "James Kirk",
                  "age" : 40,
            },
            {
                  "name" : "Jean-Luc Picard",
                  "age" : 45,
            },
            {
                  "name" : "Wesley Crusher",
                  "age" : 27,
            }
                        ]
      }
}
```

2.1 Αναπαράσταση δεομένων σε JSON και XML

2.2 APLICATON PROGRAMMING INTERFACE (API)

Μια δημοσίευση της ιστοσελίδας MuleSoft με τίτλο *What is an Api? (2016)* αναφέρει ότι API είναι το ακρωνύμιο για το Application Programming Interface, το οποίο είναι μια διασύνδεση λογισμικού που επιτρέπει σε δύο εφαρμογές να συνομιλούν μεταξύ τους. Κάθε φορά που χρησιμοποιούμε μια εφαρμογή, όπως το Facebook, στέλνουμε ένα άμεσο μήνυμα ή ελέγχουμε τον καιρό στο τηλέφωνό μας χρησιμοποιούμε ένα API. Όταν χρησιμοποιούμε μια εφαρμογή στο κινητό μας τηλέφωνο, η εφαρμογή συνδέεται με το

διαδίκτυο και στέλνει δεδομένα σε ένα διακομιστή. Ο διακομιστής ανακτά αυτά τα δεδομένα, τα επεξεργάζεται, εκτελεί τις απαραίτητες ενέργειες και τα στέλνει πίσω στο τηλέφωνό μας. Η εφαρμογή με τη σειρά της επεξεργάζεται τα δεδομένα και μας παρουσιάζει τις πληροφορίες που θέλουμε με ευανάγνωστο τρόπο.

Ένα πιο οικείο παράδειγμα είναι όταν καθόμαστε σε ένα τραπέζι εστιατορίου με ένα μενού επιλογών για να παραγγείλουμε. Η κουζίνα είναι το μέρος του "συστήματος" που θα προετοιμάσει την παραγγελία μας. Αυτό που λείπει είναι ο κρίσιμος σύνδεσμος για την επικοινωνία της παραγγελίας μας με την κουζίνα και την παράδοση του φαγητού μας πίσω στο τραπέζι μας. Εκεί βρίσκεται ο σερβιτόρος ή το API. Ο σερβιτόρος είναι ο αγγελιοφόρος ή το API που παίρνει το αίτημά μας ή παραγγέλνει και λέει στην κουζίνα (το σύστημα) τι να κάνει. Στη συνέχεια ο σερβιτόρος παραδίδει την απάντηση σε εμάς. Στην περίπτωση αυτή είναι το φαγητό.

Με αφορμή το παραπάνω παράδειγμα αξίζει να αναφερθούμε σε ορισμένα χαρακτηριστικά για τα σύγχρονα API:

- Τα σύγχρονα API συμμορφώνονται με πρότυπα (HTTP και REST), είναι φιλικά προς τον προγραμματιστή, εύκολα, προσβάσιμα και κατανοητά.
- Αντιμετωπίζονται περισσότερο σαν προϊόντα. Είναι σχεδιασμένα για κατανάλωση από συγκεκριμένο κοινό (π.χ. προγραμματιστές για κινητά), είναι τεκμηριωμένα και έχουν εκδοθεί με τρόπο ώστε οι χρήστες να έχουν ορισμένες προσδοκίες όσον αφορά τη συντήρηση και τον κύκλο ζωής τους.
- Επειδή είναι τυποποιημένα έχουν ισχυρότερη πειθαρχία για την ασφάλεια και τη διακυβέρνηση, καθώς και παρακολουθούνται και διαχειρίζονται για απόδοση και κλίμακα.
- Το σύγχρονα API έχουν τον δικό τους κύκλο ζωής ανάπτυξης λογισμικού, σχεδιασμού, δοκιμών, κατασκευής, διαχείρισης και εκδόσεων.

2.3 FRAMEWORK

Ο Nikola Kolev με μια δημοσίευση του στην ιστοσελίδα Jaxenter με τίτλο *The pros and cons of using a framework (2017)* αναφέρεται στη χρησιμότητα του πλαισίου ανάπτυξης,

αλλά και στα πλεονεκτήματα και στα μειονεκτήματα τα οποίο μπορεί αυτό να έχει. Ως πλαίσιο λογισμικού ορίζεται ένα επαναχρησιμοποιήσιμο περιβάλλον λογισμικού που παρέχει ιδιαίτερη λειτουργικότητα, ως μέρος μιας μεγαλύτερης πλατφόρμας λογισμικού. Αυτό διευκολύνει την ανάπτυξη εφαρμογών λογισμικού, προϊόντων και λύσεων. Τα πλαίσια λογισμικού ενδέχεται να περιλαμβάνουν προγράμματα υποστήριξης, μεταγλωττιστές, βιβλιοθήκες κώδικα, σύνολα εργαλείων και διεπαφές προγραμματισμού εφαρμογών (API) που συγκεντρώνουν όλα τα διαφορετικά στοιχεία για να επιτρέψουν την ανάπτυξη ενός έργου ή ενός συστήματος. Τα πλαίσια λογισμικού είναι παρόμοια με τις βιβλιοθήκες αλλά έχουν κάποια χαρακτηριστικά που τα ξεχωρίζουν από αυτές.

- **Αντιστροφή ελέγχου:** Σε ένα πλαίσιο λογισμικού η ροή ελέγχου του συνολικού προγράμματος υπαγορεύεται από το πλαίσιο λογισμικού και όχι από τον καλούντα. Αυτό είναι το αντίθετο από το πώς λειτουργούν οι βιβλιοθήκες, όπου η ροή ελέγχου υπαγορεύεται από τον καλούντα.
- **Επεκτασιμότητα:** Οι προγραμματιστές μπορούν να προσθέσουν εξειδικευμένο κώδικα για να παρέχουν συγκεκριμένη λειτουργικότητα.
- **Μη τροποποιήσιμος κώδικας πλαισίου:** Ο κώδικας ενός πλαισίου δεν μπορεί να τροποποιηθεί αλλά μπορεί να δεχθεί επεκτάσεις που έχουν υλοποιηθεί από το χρήστη. Με άλλα λόγια, οι χρήστες μπορούν να επεκτείνουν το ίδιο το πλαίσιο αλλά δεν μπορούν να τροποποιήσουν τον κώδικα του.

Το ίδιο το πλαίσιο λογισμικού παρέχει μια συγκεκριμένη ροή ελέγχου. Αυτό που πρέπει να κάνουμε είναι να υλοποιήσουμε την επιχειρησιακή λογική της εφαρμογής και το πλαίσιο θα κάνει τα υπόλοιπα, απαλλάσσοντας το προγραμματιστή από τις συνδέσεις βάσεων δεδομένων, τις συναλλαγές, τα πρωτόκολλα μεταφοράς δεδομένων κλπ.

Ένας ακόμα λόγος για τον οποίο είναι χρήσιμο ένα πλαίσιο λογισμικού είναι οι δοκιμές (Unit tests). Κάθε έργο θα πρέπει να έχει δοκιμές. Ωστόσο, είναι μια αρκετά επίπονη διαδικασία, διότι θα πρέπει πρώτα να εκτελέσουμε τα διάφορα tests, να συλλέξουμε αποτελέσματα, να παρουσιάσουμε τα αποτελέσματα, να συγκρίνουμε τα αποτελέσματα μεταξύ των δοκιμαστικών περιόδων κ.λπ. Οι προγραμματιστές δεν πρέπει να χάνουν χρόνο για την υλοποίηση αυτής της λειτουργικότητας. Αντίθετα, ένα πλαίσιο ελέγχου μονάδων (Unit testing framework) θα πρέπει να παρέχει σημεία, όπου οι προγραμματιστές μπορούν να συνδέσουν τις δοκιμές τους και το πλαίσιο θα φροντίσει τα υπόλοιπα.

Ένα πλαίσιο ελέγχου μονάδων(Unit testing framework) είναι ένα απλό παράδειγμα ενός πλαισίου. Το Spring Framework περιέχει ένα τεράστιο σύνολο εργαλείων για την κατασκευή μεγάλων εφαρμογών με διαχείριση συναλλαγών, ασφάλεια και πολλές άλλες λειτουργίες.

2.4 SPRING FRAMEWORK

Η Java είναι μία από τις πιο δημοφιλείς γλώσσες προγραμματισμού στον κόσμο ακόμα και σήμερα. Ο λόγος για την σταθερή δημοτικότητά της οφείλεται στα ισχυρά εργαλεία ανάπτυξης προγραμμάτων και εφαρμογών τα οποία αυτή διαθέτει. Για οποιαδήποτε εμπορική ανάπτυξη έργου ένα περιβάλλον ανάπτυξης είναι απαραίτητο για την οργανωμένη υλοποίηση. Όταν πρόκειται για την Java, υπάρχουν πολλά δημοφιλή πλαίσια που ανταποκρίνονται σε διαφορετικές προτιμήσεις προγραμματιστών. Μερικά από αυτά τα δημοφιλή πλαίσια είναι το Spring, Struts, Hibernate, JSF, Vaadin, GWT, Grails και άλλα. Από όλα αυτά τα περιβάλλοντα ανάπτυξης το Spring Framework είναι το πιο δημοφιλές και πιο ευρέως χρησιμοποιούμενο σε όλο τον κόσμο.

Σύμφωνα με τη δημοσίευση της Gupta Kitty στην ιστοσελίδα Freelancinggig με τίτλο *What is Spring more popular than other Java frameworks* (2018), οι λόγοι για τους οποίους το Spring είναι πιο δημοφιλής από τα άλλα περιβάλλοντα ανάπτυξης της Java είναι οι εξής:

- **Φιλικό προς τον προγραμματιστή:** Ο λόγος για τον οποίο οι περισσότεροι προγραμματιστές προτιμούν το Spring για οποιοδήποτε τύπο και μέγεθος των εφαρμογών τους, οφείλεται στα φιλικά χαρακτηριστικά του. Είναι ευέλικτο και δίνει τη δυνατότητα στον προγραμματιστή να συνδυάσει διαφορετικές τεχνολογίες και πλαίσια λογισμικού για να παράγει μια ολοκληρωμένη και αποδοτική εφαρμογή .
- **Ενσωμάτωση με την τεχνολογία Front End**: Ένα πλαίσιο ανάπτυξης εφαρμογών πρέπει να συνεργάζεται αρμονικά τόσο με τις τεχνολογίες που χρησιμοποιούνται για την ανάπτυξη του πελάτη (web client), όσο και με τις τεχνολογίες που χρησιμοποιούνται για την ανάπτυξη του εξυπερετητή (server). Αυτήν την ευκολία

μας την παρέχει το Spring. Υποστηρίζει όλες τις σύγχρονες τεχνολογίες front-end, όπως Struts, Angular Js, jQuery, κτλ. Επίσης υπάρχουν πολλά διαθέσιμα plugins που μπορούμε να ενσωματώσουμε σε αυτό. Αυτή η απεριόριστη επιλογή plugins, οφείλεται στο ότι το Spring έχει τη μεγαλύτερη κοινοτική υποστήριξη προηγμένων και έμπειρων προγραμματιστών.

- **Θεματοστρεφής προγραμματισμος (Aspect Oriented Programming):** Aspect Oriented Programming είναι μια νέα ιδέα που έχει εισαχθεί από το Spring. Υπάρχουν πολλά πλαίσια λογισμικού που βασίζονται στο AOP (Aspect oriented programming) αλλά το Spring είναι το καταλληλότερο. Επιτρέπει στους προγραμματιστές να αναπτύξουν εφαρμογές με τέτοιο τρόπο, ώστε η επιχειρηματική λογική να παραμένει χωρισμένη από τις υπηρεσίες του συστήματος. Παρέχει διεπαφές, όπως το Spring Core, το Spring IOC και το Spring AOP, ώστε να μπορούν να αναπτυχθούν ξεχωριστά και να ενσωματωθούν διαφορετικά κομμάτια για να επιτευχθεί η τέλεια επιχειρησιακή λογική.
- **Το πλαίσιο MVC:** Το μοντέλο MVC βοηθά στην ευέλικτη ανάπτυξη, την εύκολη δημιουργία unit tests και στην ακόμα πιο εύκολη εγκατάσταση, συντήρηση και αντιμετώπιση προβλημάτων. Επίσης μας δίνει τη δυνατότητα να διαιρέσουμε ένα μεγάλο έργο σε ενότητες και να ορίσουμε διαφορετικές ομάδες για να τις ολοκληρώσουν εγκαίρως.

2.4.1 Πλεονεκτήματα Και Μειονεκτήματα Του Spring Framework

Το Spring Framework, όταν εισήχθη, γνώρισε τεράστια δημοτικότητα και εκτίμηση από τους προγραμματιστές. Χάρη στην εύκολη ανάπτυξη επαναχρησιμοποιήσιμου κώδικα με υψηλή απόδοση και των υπόλοιπων χαρακτηριστικών του, οι προγραμματιστές μπορούνε να αναπτύξουνε κάθε είδους εφαρμογή στη java. Σύμφωνα με τη δημοσίευση της Chandra Shekhar στην ιστοσελίδα Onlinetutorialspoint με τίτλο *Advantages of Spring Framework (2015)* ορισμένα από τα πλεονεκτήματα χρήσης του Spring είναι:

- Το Spring δίνει τη δυνατότητα στους προγραμματιστές να αναπτύξουν επιχειρησιακές εφαρμογές χρησιμοποιώντας POJOs (Plain Old Java Object). Το όφελος από την ανάπτυξη εφαρμογών που χρησιμοποιούν το POJO είναι ότι έχουμε την επιλογή να χρησιμοποιήσουμε ένα servlet container. Με τον όρο servlet container εννοούμε ένα μηχανισμό, ο οποίος αναπαράγει ιστοσελίδες στη μεριά του διακομοστή, χρησιμοποιώντας δυναμικά Java.
- Το Spring έρχεται με κάποιες από τις υπάρχουσες τεχνολογίες, όπως το πλαίσιο ORM, J2EE, JDK χρονοδιακόπτες κλπ. Οπότε, δεν χρειάζεται να ενσωματώσουμε αυτές τις τεχνολογίες.
- Το Spring έχει τη δυνατότητα να διαχειρίζεται τα αντικείμενα και να κάνει διαθέσιμες τις υπηρεσίες τους όπου χρειάζεται. Με αυτό τον τρόπο οι εφαρμογές ιστού έχουν πρόσβαση στα ίδια αντικείμενα.
- Πολλές από τις βέλτιστες πρακτικές που χρησιμοποιεί το Spring έχουν αποτελέσει πρότυπα σχεδίασης σε διάφορες εφαρμογές.
- Μπορεί να χρησιμοποιηθεί για την ανάπτυξη διαφορετικών τύπων εφαρμογών, όπως οι αυτόνομες εφαρμογές, οι αυτόνομες εφαρμογές GUI, οι εφαρμογές Web και οι μικροεφαρμογές.
- Το Spring παρέχει ενσωματωμένες υπηρεσίες middleware, όπως Connection pooling, Διαχείριση συναλλαγών κ.λπ.

Ωστόσο, πριν χρησιμοποιήσουμε το Spring πρέπει να γνωρίζουμε τους περιορισμούς του. Μια δημοσίευση της ιστοσελίδας Right place for right person (2017) αναφέρει ως μειονεκτήματα του Spring Framework τα εξής:

- Ένα από τα σημαντικότερα προβλήματα που αντιμετωπίζει το Spring είναι ότι είναι πολύπλοκο. Υπάρχουν παράλληλοι μηχανισμοί οι οποίοι είναι χρήσιμοι για την εκτέλεση της εφαρμογής. Αυτό το γεγονός προκαλεί σύγχυση στους προγραμματιστές και τους αναγκάζει να ξοδέψουν αρκετό χρόνο για να κατανοήσουν αυτούς τους μηχανισμούς και το τρόπο λειτουργίας τους.
- Απαιτεί μεγάλη διαμόρφωση. Για να μπορέσει ένας προγραμματιστής να χρησιμοποιήσει ορισμένες από τις τεχνολογίες που προσφέρει το Spring, χρειάζεται να διαμορφώσει αρκετά xml αρχεία για να είναι λειτουργική η εφαρμογή. Επομένως, οι προγραμματιστές θα πρέπει εκ των προτέρων να γνωρίζουν τη δομή και τη λειτουργία του XML.

- Δεν υπάρχουν σαφείς οδηγίες σε θέματα, όπως επιθέσεις scripting (cross-site scripting) ή επιθέσεις πλαστογράφησης αιτήσεων(cross-site request forgery).

2.4.2 Spring Framework και Struts

Το Struts Framework είναι η υλοποίηση του Model-View-Controller (MVC) design pattern για το Java Single Page(JSP). Το Struts διατηρείται ως μέρος του έργου Apache Jakarta και είναι ανοικτού κώδικα. Επίσης είναι κατάλληλο για την υλοποίηση οποιουδήποτε μεγέθους εφαρμογής. [21]

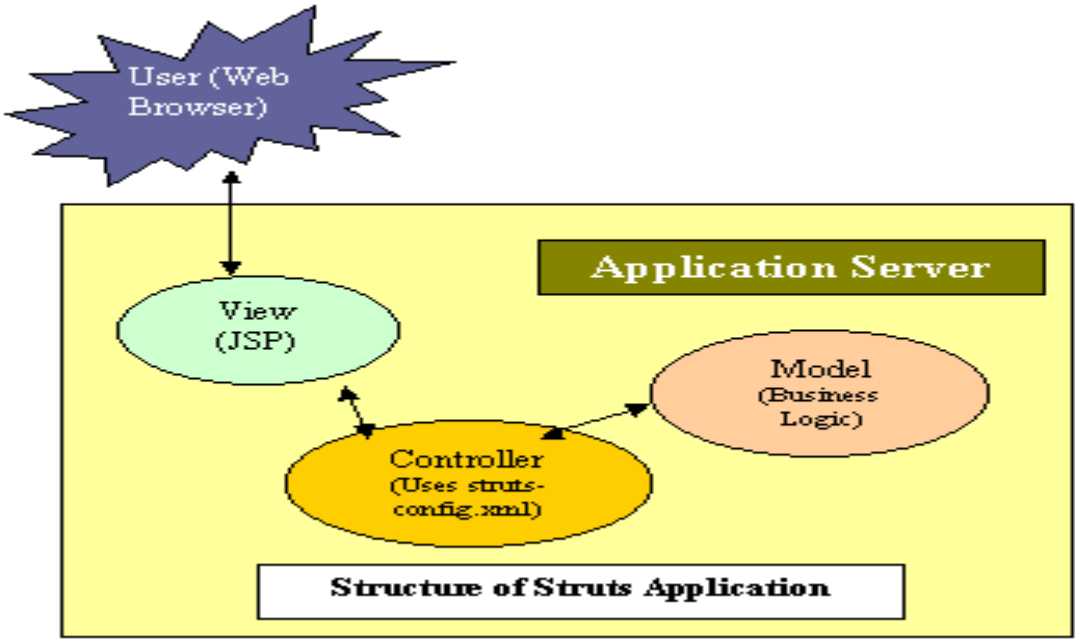
Το Struts παρέχει τρία βασικά στοιχεία:

- Ένας χειριστής αιτήματος που παρέχεται από τον προγραμματιστή της εφαρμογής και έχει αντιστοιχιστεί σε ένα τυπικό URI.
- Ένας χειριστής απόκρισης που μεταφέρει τον έλεγχο σε έναν άλλο πόρο που ολοκληρώνει την απόκριση.
- Μια βιβλιοθήκη ετικετών που βοηθά τους προγραμματιστές να δημιουργούν διαδραστικές εφαρμογές.

Το Struts βοηθά στην ανάπτυξη των web εφαρμογών. Είναι ένα από τα πιο δημοφιλή πλαίσια για εφαρμογές που βασίζονται στον ιστό. Τα χαρακτηριστικά αυτού του τύπου πλαισίου είναι:

- Αξιόπιστη αρχιτεκτονική
- Βοηθά στην ανάπτυξη εφαρμογών οποιουδήποτε μεγέθους
- Εύκολη σχεδίαση

Στην εικόνα 2.2 παρουσιάζεται η δομή που πρόκειται να έχει μια εφαρμογή υλοποιημένη με το Struts Framework.



2.2 Η δομή μιας εφαρμογής μέσω Struts

- **Model**: Το μοντέλο περιλαμβάνει τον πυρήνα της λειτουργικότητας της εφαρμογής. Συγκρατεί την κατάσταση της εφαρμογής. Μερικές φορές η μόνη λειτουργικότητα που περιέχει είναι η κατάσταση της εφαρμογής.
- **View**: Η προβολή δίνει την παρουσίαση του μοντέλου. Είναι η εμφάνιση της εφαρμογής. Η προβολή θα πρέπει να ενημερώνεται όταν προκύπτουν αλλαγές στο μοντέλο.
- **Controller**: Ο ελεγκτής αντιδρά στην είσοδο του χρήστη. Δημιουργεί και ρυθμίζει το μοντέλο.

Ορισμένα από τα χαρακτηριστικά για τα οποία τα Struts έχει γίνει τόσο δημοφιλής είναι τα εξής :

- Είναι ανοιχτού κώδικα
- Υλοποιεί την αρχιτεκτονική μοντέλου JSP 2(Java Single Page)
- Αποθηκεύει πληροφορίες δρομολόγησης εφαρμογών (routing) και χαρτογράφησης αιτήσεων (request mapping) σε ένα μόνο αρχείο, struts-config.xml
- Το Struts αλληλεπιδρά μόνο με τα επίπεδα view και controller. Το επίπεδο model αφήνεται στον προγραμματιστή

- OGNL και ValueStack: Το ακρωνύμιο OGNL (Object Graph Navigation Language) είναι μια ισχυρή γλώσσα έκφρασης που χρησιμοποιείται για την αναφορά και τον χειρισμό δεδομένων.

Τέλος, αναφέρουμε τα δομικά στοιχεία από τα οποία αποτελείται το Struts Framework.

- **Χειριστής αποτελεσμάτων:** Ο χειριστής αποτελεσμάτων στέλνει το αίτημα για προβολή
- **Custom ετικέτες:** Οι custom ετικέτες χρησιμοποιούνται για την απόδοση του δυναμικού περιεχομένου
- **Stack Value:** Το Stack Value επιτρέπει στη γλώσσα έκφρασης να βρει τιμές ιδιοτήτων σε πολλά αντικείμενα.
- **Γλώσσα έκφρασης:** Η γλώσσα έκφρασης χρησιμοποιείται για τη λήψη και τη ρύθμιση ιδιοτήτων των αντικειμένων Java
- **Ελεγκτής:** Ο ελεγκτής λαμβάνει όλα τα εισερχόμενα αιτήματα. Η κύρια λειτουργία του είναι η χαρτογράφηση ενός URI αιτήματος σε μια συγκεκριμένη κλάση δράσης
- **Αρχείο Struts-config.xml:** Αυτό το αρχείο περιλαμβάνει όλες τις πληροφορίες δρομολόγησης και διαμόρφωσης για την εφαρμογή.
- **Action classes:** Είναι ευθύνη του έργου να δημιουργήσει αυτές τις κλάσεις. Δρουν ως γέφυρες μεταξύ URI που καλούνται από το χρήστη. Οι ενέργειες επεξεργάζονται ένα αίτημα και επιστρέφουν ένα αντικείμενο το οποίο προσδιορίζει το επόμενο συστατικό στοιχείο που πρέπει να κληθεί. Είναι μέρος του στρώματος του ελεγκτή, όχι του επιπέδου του μοντέλου.
- **Προβολή πόρων:** Η προβολή πόρων περιλαμβάνει σελίδες Java Server, σελίδες HTML, αρχεία JavaScript και Stylesheet, JavaBeans και ετικέτες JSP Struts.
- **Action Forms:** Απλοποιούν σε μεγάλο βαθμό την επικύρωση της φόρμας χρήστη καταγράφοντας τα δεδομένα χρήστη από το αίτημα HTTP. Ενεργούν ως τείχος προστασίας μεταξύ των ιστοσελίδων και της εφαρμογής. Αυτά τα στοιχεία επιτρέπουν την επικύρωση της εισόδου του χρήστη πριν προχωρήσει σε μια ενέργεια. Εάν η είσοδος δεν είναι έγκυρη μπορεί να εμφανιστεί μια σελίδα με σφάλμα. [19]

Τόσο το Struts, όσο και το Spring Framework είναι δυο από τα πιο δημοφιλής πλαίσια λογισμικού. Το καθένα έχει τα πλεονεκτήματα και τις διαφορές του. Με αφορμή τη δημοσίευση της ιστοσελίδας Educba με τίτλο *Spring vs Struts (2018),* αναφέρουμε 5 από τις βασικότερες διαφορές ανάμεσα στο Struts και το Spring Framework.

Πίνακας 2.1 Διαφορές ανάμεσα στο Spring και το Struts Framework

Spring Framework	**Struts Framework**
Είναι ένα πλαίσιο εφαρμογής το οποίο υλοποιεί το Di(Dependence Injection) και IOC(Inverse of control)	Είναι ένα πλαίσιο ανοιχτού κώδικα το οποίο επεκτείνει το MVC(Model-View-Controller) πλαίσιο
Έχει πολυεπίπεδη αρχιτεκτονική	Δεν έχει πολυεπίπεδη αρχιτεκτονική
Είναι ένα πλαίσιο με λίγες απαιτήσεις	Είναι ένα πλαίσιο με πολλές απαιτήσεις
Μπορεί εύκολα να ενσωματώσει άλλες τεχνολογίες όπως JDBC ή ORM	Μπορεί να ενσωματώσει άλλες τεχνολογίες αλλά χρειάζεται ο προγραμματιστής να γράψει έξτρα κώδικα
Έχει μεγαλύτερη ευελιξία συγκριτικά με το Struts	Είναι λιγότερο ευέλικτο από το Spring

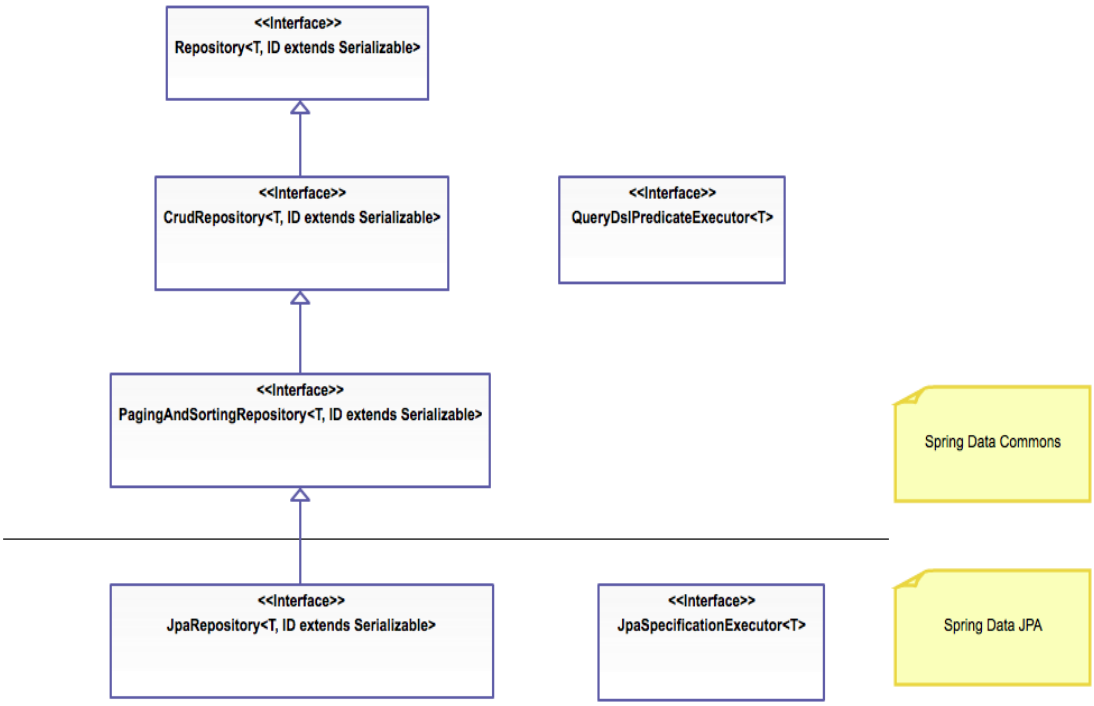
2.4.3 Spring Data JPA

Το Spring Data JPA ανήκει στο Spring Data και πρόκειται για ένα πλαίσιο ανάπτυξης το οποίο διευκολύνει την υλοποίηση αποθετηρίων (repositories) τα οποία βασίζονται στο JPA. Βοηθάει ιδιαίτερα στην κατασκευή εφαρμογών οι οποίες αναπτύσσονται με το Spring και χρησιμοποιούν τεχνολογίες πρόσβασης δεδομένων. Η υλοποίηση ενός στρώματος πρόσβασης δεδομένων μιας εφαρμογής είναι μία δύσκολη διαδικασία, καθώς χρειάζεται να γραφτεί αρκετός κώδικας ακόμα και για την εκτέλεση απλών ερωτημάτων. Το Spring Data JPA στοχεύει στη σημαντική βελτίωση της εφαρμογής σε επίπεδο πρόσβασης δεδομένων, μειώνοντας την προσπάθεια στο ποσό που πραγματικά χρειάζεται.

Έτσι, το μόνο που χρειάζεται να κάνει ο προγραμματιστής είναι να γράψει τις διεπαφές αποθετηρίου και το Spring θα τα παρέχει αυτόματα στην εφαρμογή. [17]

Οι διεπαφές οι οποίες προσφέρονται από το Spring Data αναφέρονται παρακάτω.

- **CrudRepository**: Παρέχει λειτουργίες CRUD (δημιουργίας, διαγραφής, ενημέρωσης) για τη διαχειριζόμενη οντότητα.
- **PagingAndSortingRepository**: Δηλώνει τις μεθόδους που χρησιμοποιούνται για την ταξινόμηση και την αποτύπωση σελίδων σε οντότητες που έχουν ανακτηθεί από τη βάση δεδομένων.
- **QueryDslPredicateExecutor**: Δεν είναι διεπαφή. Δηλώνει τις μεθόδους που χρησιμοποιούνται για την ανάκτηση οντοτήτων από τη βάση δεδομένων χρησιμοποιώντας αντικείμενα QueryDsl Predicate.
- **JpaRepository:** Είναι ένα repository συγκεκριμένης αποθήκευσης JPA που συνδυάζει τις μεθόδους που δηλώνονται από τις κοινές διεπαφές πίσω από μια ενιαία διεπαφή.
- **JpaSpecificationExecutor**: Δεν είναι διεπαφή. Δηλώνει τις μεθόδους που χρησιμοποιούνται για την ανάκτηση οντοτήτων από τη βάση δεδομένων. [8]



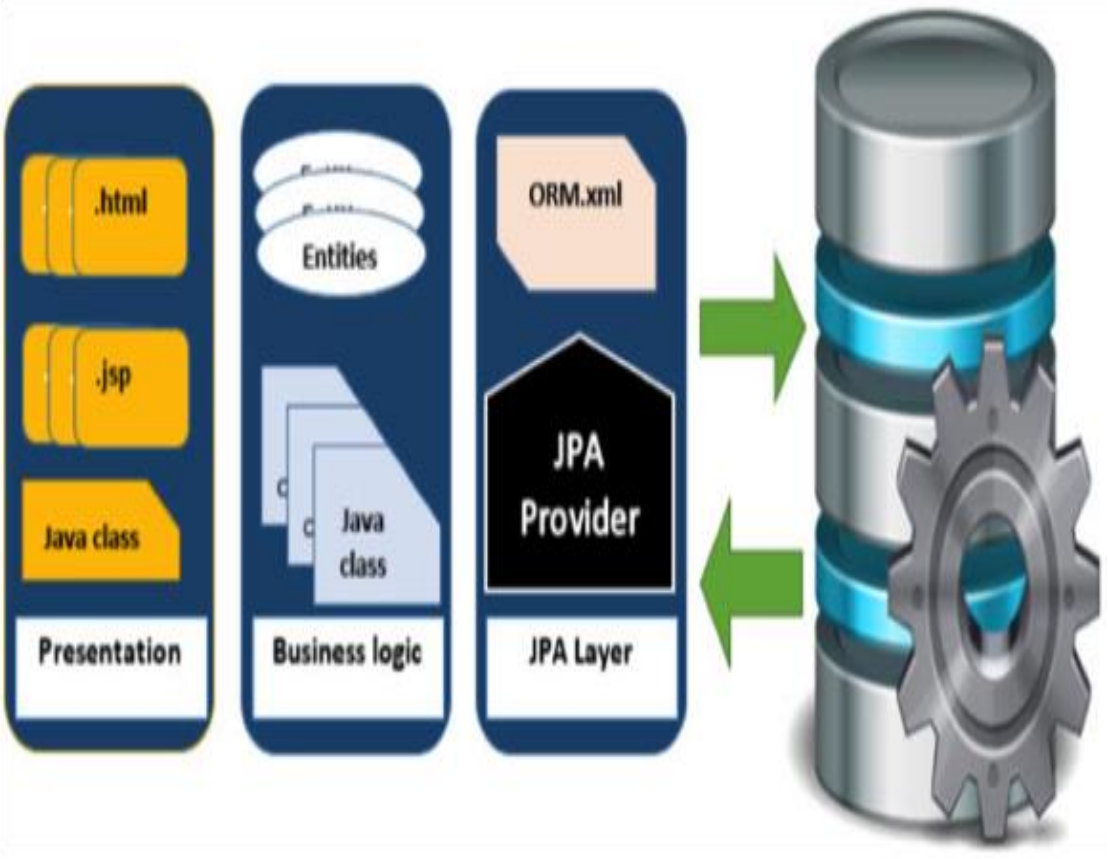
2.3 Η ιεραρχία των repositories

2.5 JAVA PERSISTENCE API (JPA)

Παρά τις διαθέσιμες τεχνολογίες διαχείρισης των δεδομένων, οι προγραμματιστές εφαρμογών καταβάλλουν προσπάθεια για να εκτελέσουν αποτελεσματικά τις λειτουργίες μιας βάσης δεδομένων. Γράφουν αρκετό κώδικα ή χρησιμοποιούν τα κατάλληλα πλαίσια ανάπτυξης για να αλληλεπιδράσουν με τη βάση δεδομένων, ενώ με τη χρήση του JPA το βάρος της αλληλεπίδρασης με τη βάση δεδομένων μειώνεται σημαντικά. Αποτελεί γέφυρα μεταξύ μοντέλων αντικειμένων (πρόγραμμα Java) και σχεσιακών μοντέλων (πρόγραμμα βάσης δεδομένων). Τα σχεσιακά αντικείμενα παρουσιάζονται σε μορφή πίνακα, ενώ τα αντικειμενικά μοντέλα παρουσιάζονται σε ένα διασυνδεδεμένο γράφημα μορφής αντικειμένου. Κατά την αποθήκευση και την ανάκτηση ενός μοντέλου αντικειμένου από μια σχεσιακή βάση δεδομένων, εμφανίζονται αναντιστοιχίες εξαιτίας των ακόλουθων λόγων:

- **Γραμμικότητα:** Το μοντέλο αντικειμένων έχει μεγαλύτερη ακρίβεια από το σχεσιακό μοντέλο.
- **Υπό Τύποι:** Κληρονομικότητα. Δεν υποστηρίζονται από όλους τους τύπους σχεσιακών βάσεων δεδομένων.
- **Συσχετισμοί:** Τα σχεσιακά μοντέλα δεν μπορούν να καθορίσουν πολλαπλές σχέσεις, σε αντίθεση με τα μοντέλα αντικειμένων.
- **Πλοήγηση δεδομένων:** Η πλοήγηση δεδομένων μεταξύ αντικειμένων σε ένα δίκτυο αντικειμένων είναι διαφορετική και στα δύο μοντέλα.

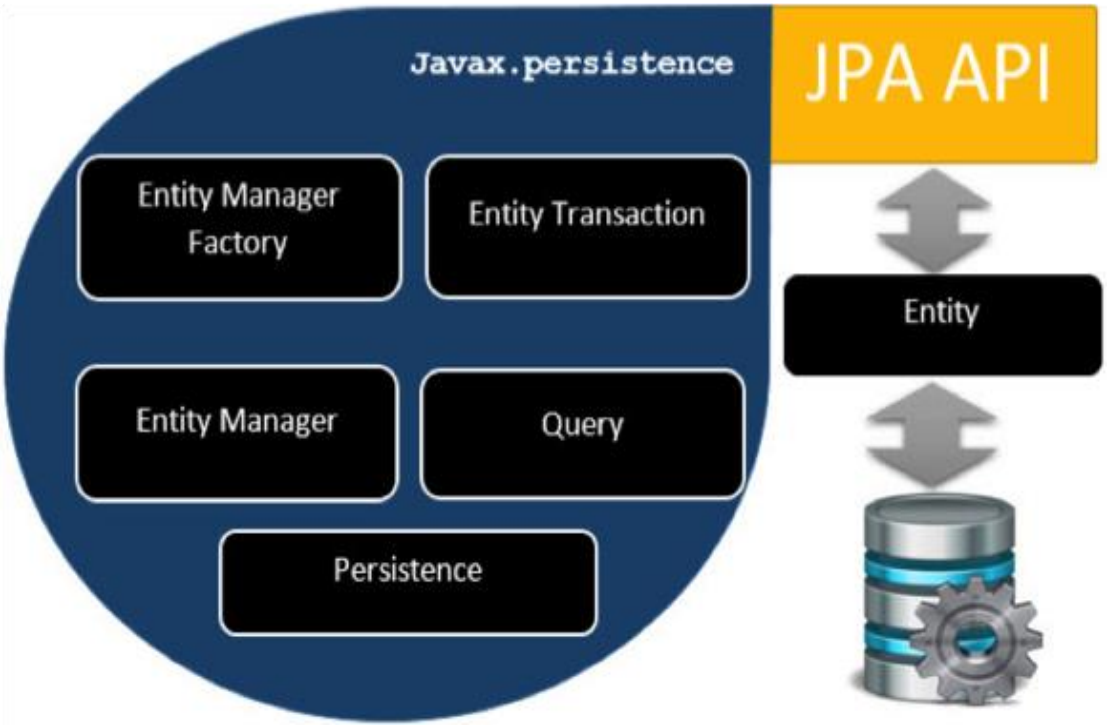
Το Java Persistence API είναι μια συλλογή από κλάσεις και μεθόδους για την αποθήκευση (persistence) δεδομένων σε μια βάση δεδομένων. Για να μειωθεί ο κώδικας για τη διαχείριση των σχεσιακών αντικειμένων οι προγραμματιστές χρησιμοποιούν διάφορους JPA προμηθευτές για την εύκολη αλληλεπίδραση με την βάση δεδομένων. Το γεγονός ότι το JPA είναι ένα API ανοιχτού κώδικα έχει οδηγήσει εταιρείες, όπως η Oracle, το Redhat, το Eclipse κ.λπ., να παρέχουν νέα προϊόντα υλοποιώντας με το δικό τους τρόπο τη λειτουργικότητα του JPA. Μερικά από τα προϊόντα είναι το Hibernate, Eclipselink, Toplink,Spring Data JPA, κλπ. [7]



2.4 Επίπεδο εφαρμογής του JPA

2.5.1 Η Αρχιτεκτονική Του JPA

Το Java Persistence API είναι υπεύθυνο για την αποθήκευση των επιχειρηματικών οντοτήτων, ως σχεσιακές οντότητες. Μας δείχνει πώς μπορούμε να ορίσουμε οντότητες (Plain Old Java Object) και πώς να διαχειριστούμε οντότητες με σχέσεις.



2.5 Η αρχιτεκτονική επιπέδου κλάσεων του JPA

Πίνακας 2.2 Περιγραφή των κλάσεων του JPA

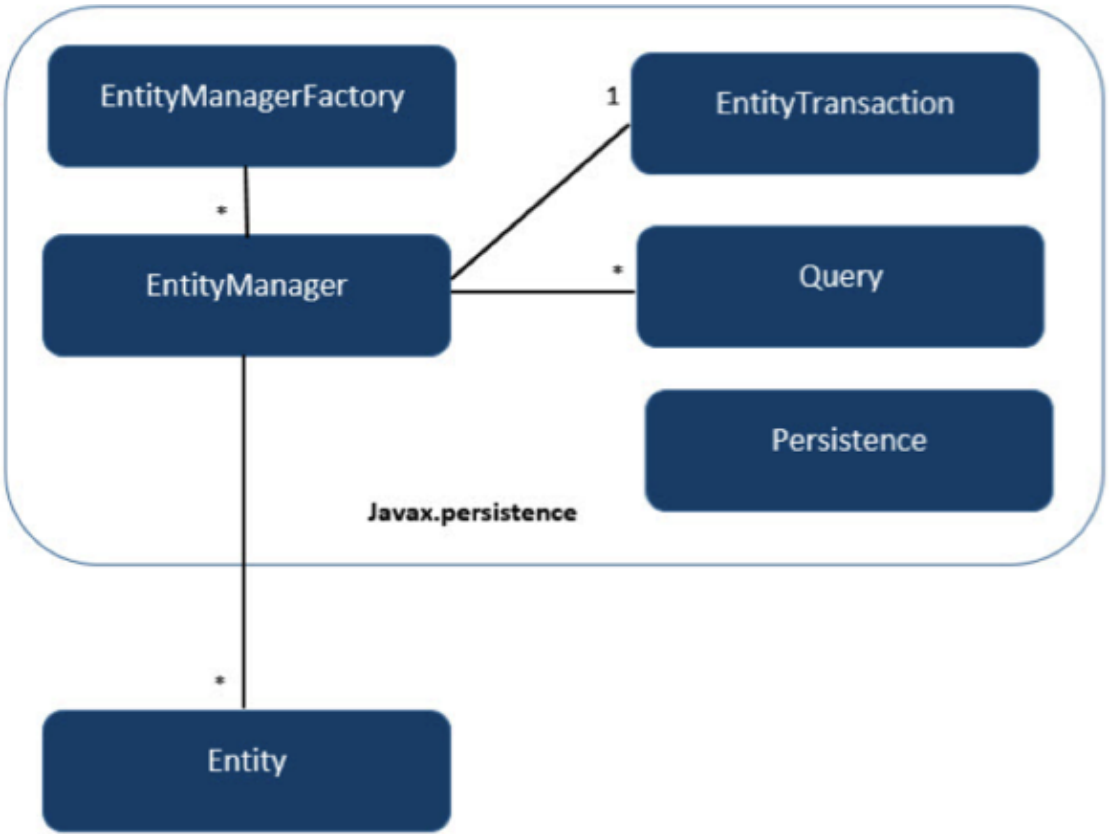
ΚΛΑΣΕΙΣ	**ΠΕΡΙΓΡΑΦΗ**
Entity Manager Factory	Είναι μια factory κλάση του Entity Manager. Δημιουργεί και διαχειρίζεται πολλά instances του Entity Manager.
Entity Manager	Πρόκειται για μια διεπαφή η οποία διαχειρίζεται τις λειτουργίες αποθήκευσης ενός αντικείμενου. Λειτουργεί σαν factory για τα Query instances.
Entity	Entities είναι τα persistence αντικείμενα τα οποία αποθηκεύονται σαν εγγραφές στη βάση.
Entity Transaction	Έχει σχέση ένα προς ένα με τον Entity Manager. Οι λειτουργίες για κάθε ένα Entity Manager διατηρούνται από την κλάση Entity Transaction.
Persistence	Αυτή η κλάση περιέχει στατικές μεθόδους για την απόκτηση του Entity Manager Factory instance.
Query	Αυτή η διεπαφή υλοποιείται από κάθε JPA vendor για την απόκτηση σχεσιακών αντικειμένων που πληρούν τα κριτήρια.

Οι κλάσεις και οι διεπαφές του πίνακα 2.2 χρησιμοποιούνται για την αποθήκευση οντοτήτων σε μια βάση δεδομένων ως εγγραφές. Βοηθούν τους προγραμματιστές να μειώσουνε τις προσπάθειές τους να γράψουν κώδικα για την αποθήκευση δεδομένων σε

μια βάση δεδομένων, ώστε να μπορούν να επικεντρωθούν σε πιο σημαντικές δραστηριότητες, όπως είναι η αντιστοίχιση ενός πίνακα στη βάση με μία κλάση. [8]

2.5.2 Σχέσεις Κλάσεων (JPA)

Στην εικόνα 2.5 απεικονίζεται η αρχιτεκτονική και οι σχέσεις μεταξύ των κλάσεων και των διεπαφών. Η εικόνα 2.6 δείχνει τη σχέση μεταξύ τους.



2.6 Οι σχέσεις μεταξύ των κλάσεων και των διεπαφών του JPA

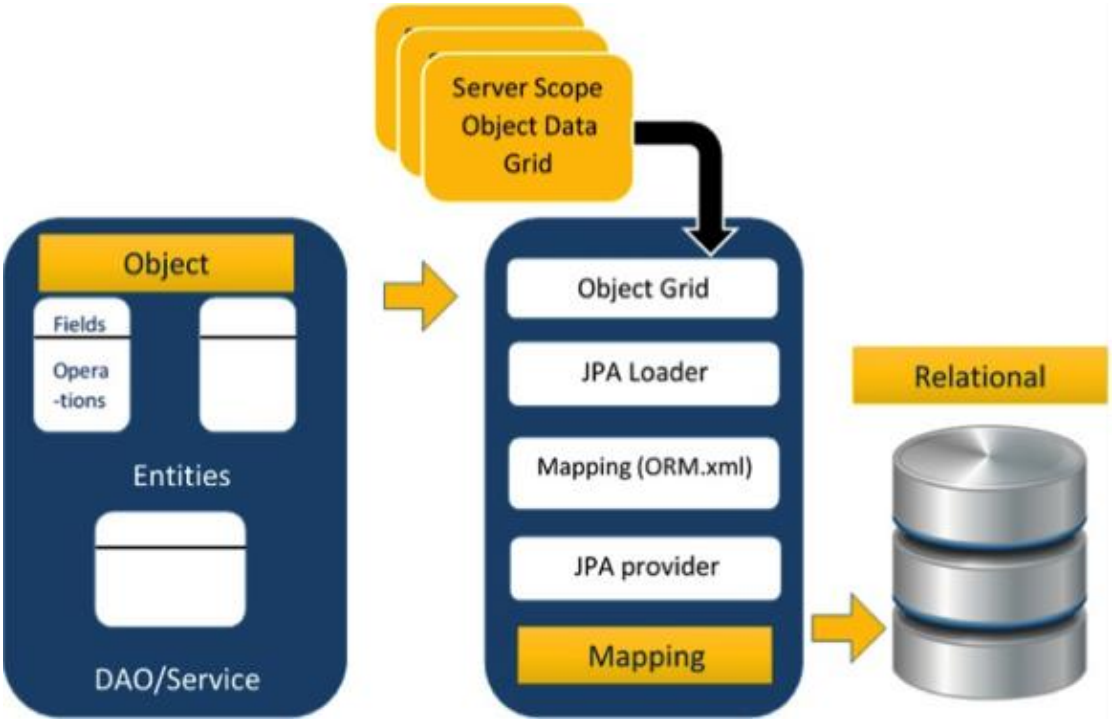
- Η σχέση μεταξύ Entity Manager Factory και Entity Manager είναι ένα προς πολλά.
- Η σχέση μεταξύ Entity Manager και Entity Transaction είναι ένα προς ένα. Για κάθε λειτουργία Entity Manager υπάρχει μια παρουσία Entity Transaction.
- Η σχέση μεταξύ του Entity Manager και του Query είναι ένα προς πολλά. Πολλοί αριθμοί ερωτημάτων(queries) μπορούν να εκτελεστούν χρησιμοποιώντας μία ενότητα Entity Manager.

- Η σχέση μεταξύ του Entity Manager και της οντότητας είναι ένα προς πολλά. Ένας Entity Manager μπορεί να διαχειριστεί πολλές οντότητες. [8]

2.5.3 Object Relational Mapping (ORM)

Το ORM είναι υπεύθυνο για την μετατροπή δεδομένων από τύπο αντικειμένου σε σχεσιακό τύπο και αντίστροφα. Το κύριο χαρακτηριστικό του ORM είναι η χαρτογράφηση ή η δέσμευση ενός αντικειμένου και των δεδομένων του με μία όψη στη βάση δεδομένων. Κατά τη χαρτογράφηση (mapping) πρέπει να εξετάσουμε τα δεδομένα, τον τύπο των δεδομένων και τις σχέσεις τους. Μερικά από τα χαρακτηριστικά του ORM είναι:

- **Κληρονομικότητα και Αντικειμενοστρέφεια:** Έχουμε τη δυνατότητα να αξιοποιήσουμε τις έννοιες της κληρονομικότητας και της αντικειμενοστρέφειας.
- **Υψηλή απόδοση:** Έχει πολλές τεχνικές και τεχνικές κλειδώματος.
- **Αξιόπιστο:** Είναι εξαιρετικά σταθερό και χρησιμοποιείται από πολλές εφαρμογές.



2.7 Η αρχιτεκτονική του ORM

Σύμφωνα με την εικόνα 2.7 η αντικειμενική σχεσιακή χαρτογράφηση (Object Relational Mapping) χωρίζεται σε τρεις φάσεις. Η πρώτη φάση ονομάζεται φάση δεδομένων αντικειμένων (object data phase). Περιλαμβάνει κλάσεις(classes) , διεπαφές(interfaces) και άλλες υπηρεσίες(services). Είναι το κύριο επιχειρηματικό στρώμα, το οποίο έχει λειτουργίες που βασίζονται στην επιχειρησιακή λογική και διάφορα χαρακτηριστικά.

Η δεύτερη φάση ονομάζεται φάση χαρτογράφησης (mapping) ή persistence phase. Περιέχει τον JPA vendor, το αρχείο χαρτογράφησης (ORM.xml), τον JPA Loader και το πλέγμα αντικειμένων(Object grid).

- **JPA vendor:** Είναι οι διάφορες υλοποιήσεις του JPA όπως Eclipselink, Toplink, Hibernate.
- **Αρχείο χαρτογράφησης:** Το αρχείο αντιστοίχισης (ORM.xml) περιέχει την αντιστοιχία μεταξύ των δεδομένων μιας κλάσης και των δεδομένων σε μια σχεσιακή βάση δεδομένων.
- **JPA Loader:** Ο JPA Loader λειτουργεί σαν μνήμη cache.
- **Πλέγμα αντικειμένων(Object grid):** Πρόκειται για μια προσωρινή τοποθεσία που μπορεί να αποθηκεύσει ένα αντίγραφο σχεσιακών δεδομένων. Όλα τα ερωτήματα (queries) κατά της βάσης δεδομένων πραγματοποιούνται πρώτα στα δεδομένα του πλέγματος αντικειμένων. Μόνο μετά τη δέσμευση επηρεάζονται τα δεδομένα της βάσης.

Η τρίτη φάση είναι η φάση των σχεσιακών δεδομένων. Περιέχει τα σχεσιακά δεδομένα που είναι λογικά συνδεδεμένα με το επιχειρηματικό στοιχείο. Όπως αναφέρθηκαμε παραπάνω, μόνο όταν το επιχειρηματικό στοιχείο δεσμεύει τα δεδομένα αποθηκεύονται φυσικά στη βάση δεδομένων. Μέχρι τότε τα τροποποιημένα δεδομένα αποθηκεύονται σε μια προσωρινή μνήμη με μορφή πλέγματος. Η διαδικασία λήψης των δεδομένων είναι ίδια με εκείνη της αποθήκευσης των δεδομένων. [9]

2.5.4 Αντιστοίχιση Οντοτήτων Με Πίνακες Βάσης Δεδομένων

Mapping.xml είναι ένα αρχείο με σκοπό να καθοδηγήσει τον JPA vendor να αντιστοιχίσει τις κλάσεις οντοτήτων με τους πίνακες της βάσης δεδομένων. Στην εικόνα 2.8 αναπαριστάται η κλάση οντότητας ενός υπαλλήλου.

```
public class Employee {

        private int eid;
        private String ename;
        private double salary;
        private String deg;

        public Employee(int eid, String ename, double salary, String deg) {
                super( );
                this.eid = eid;
                this.ename = ename;
                this.salary = salary;
                this.deg = deg;
        }
        public Employee( ){
                super();
        }
        public int getEid( ) {
                return eid;
        }
        public void setEid(int eid)  {
                this.eid = eid;
        }
        public String getEname( ) {
                return ename;
        }
        public void setEname (String ename) {
                this.ename = ename;
        }
        public double getSalary( ){
                return salary;
        }
        public void setSalary(double salary) {
                this.salary = salary;
        }
        public String getDeg( ) {
                return deg;
        }
        public void setDeg(String deg) {
                this.deg = deg;
        }}
```

2.8 Η κλάση POJO της ενότητας Employee

```
<? xml version="1.0" encoding="UTF-8" ?>
<entity-mappings xmlns="http://java.sun.com/xml/ns/persistence/orm"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://java.sun.com/xml/ns/persistence/orm
                        http://java.sun.com/xml/ns/persistence/orm_1_0.xsd"
                        version="1.0">
    <description> XML Mapping file</description>
    <entity class="Employee">
        <table name="EMPLOYEETABLE"/>
        <attributes>
            <id name="eid">
                <generated-value strategy="TABLE"/>
            </id>
            <basic name="ename">
                <column name="EMP_NAME" length="100"/>
            </basic>
            <basic name="salary">
            </basic>
            <basic name="deg">
            </basic>
        </attributes>
    </entity>
</entity-mappings>
```

2.9 Το αρχείο XML της κλάσης Employee

Η εικόνα 2.9 απεικονίζει τη χαρτογράφηση της κλάσης υπαλλήλου με τον πίνακα της βάσης δεδομένων. Σε αυτό το αρχείο έχουμε τις εξής ετικέτες :

- **<entity-mappings>**: Η ετικέτα ορίζει τον ορισμό του σχήματος.
- **<description>**: Η ετικέτα παρέχει μια περιγραφή σχετικά με την εφαρμογή.
- **<entity>**: Η ετικέτα ορίζει την κλάση οντότητας που θέλουμε να μετατρέψουμε σε πίνακα μιας βάσης δεδομένων.
- **<table>**: Η ετικέτα ορίζει το όνομα του πίνακα. Σε περίπτωση που έχουμε ταυτόσημα ονόματα, τόσο για την οντότητα, όσο και για τον πίνακα, τότε αυτή η ετικέτα δεν είναι απαραίτητη.
- **<attributes>**: Η ετικέτα ορίζει τα χαρακτηριστικά (πεδία σε έναν πίνακα).
- **<id>**: Η ετικέτα ορίζει το πρωτεύον κλειδί του πίνακα. Η ετικέτα <generated-value> καθορίζει τον τρόπο εκχώρησης της τιμής του πρωτεύοντος κλειδιού, όπως αυτόματος ή χειροκίνητος.
- **<basic>**: Η ετικέτα χρησιμοποιείται για τον καθορισμό των υπόλοιπων χαρακτηριστικών του πίνακα.

- **<column-name>:** Η ετικέτα χρησιμοποιείται για τον καθορισμό του ονόματος μιας στήλης του πίνακα. [11]

2.5.5 Annotations

Στο πίνακα 2.3 αναφέρομε ορισμένα από τα annotations που χρησιμοποιεί το JPA. [11]

Πίνακας2.3 Annotations του JPA

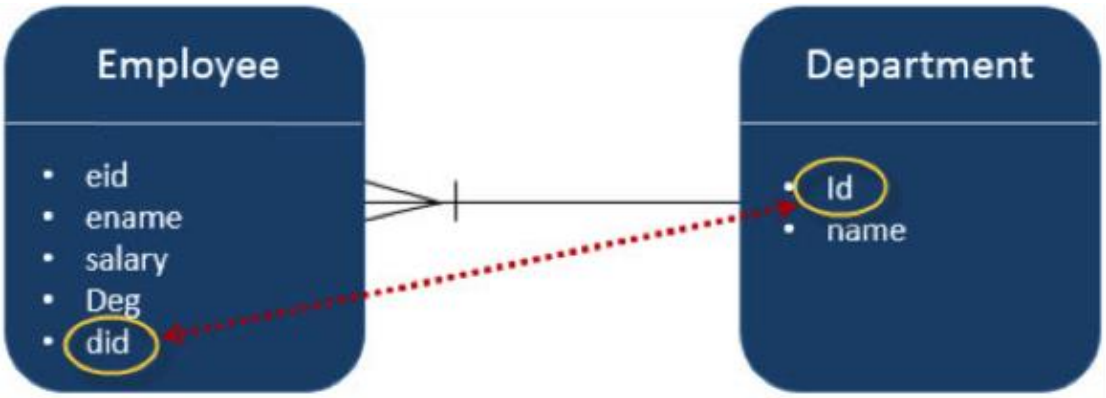
ANNOTATION	**ΠΕΡΙΓΡΑΦΗ**
@Entity	Δηλώνει την κλάση ως entity
@Table	Δηλώνει το όνομα του σχήματος
@Basic	Καθορίζει ρητά τα πεδία χωρίς περιορισμούς.
@Embedded	Καθορίζει τις ιδιότητες της κλάσης ή της οντότητας η οποία είναι instance μιας ενσωματωμένης κλάσης.
@Id	Καθορίζει την ιδιότητα που χρησιμοποιείται για την ταυτότητα (πρωτεύον κλειδί ενός πίνακα) της κλάσης.
@GeneratedValue	Καθορίζει τον τρόπο με τον οποίο μπορεί να αρχικοποιηθεί το χαρακτηριστικό ταυτότητας, όπως αυτόματα ή χειροκίνητα.
@Transient	Καθορίζει εκείνες τις τιμές που δεν αποθηκεύονται ποτέ στη βάση δεδομένων.
@Column	Το όνομα της στήλης του πίνακα
@SequenceGenarartor	Καθορίζει την τιμή του πεδίου το οποίο έχει σαν annotation το @GeneratedValue. Δημιουργεί μια ακολουθία.
@TableGenerator	Καθορίζει τη γεννήτρια τιμών για το πεδίο το οποίο έχει το annotation @GeneratedValue. Δημιουργεί έναν πίνακα με τυχαίες τιμές .
@JoinColumn	Καθορίζει τη σχέση μεταξύ μιας οντότητας ή μίας συλλογής οντοτήτων. Αυτό χρησιμοποιείται σε σχέσεις ένα προς πολλά και πολλά προς ένα.
@AccessType	Αυτός ο τύπος annotation χρησιμοποιείται για τον ορισμό του τύπου πρόσβασης. Εάν ο τύπος πρόσβασης είναι (FIELD), τότε η πρόσβαση εμφανίζεται στο πεδίο. Αν είναι (PROPERTY), τότε η πρόσβαση εμφανίζεται στην ιδιότητα.
@UniqueConstraint	Καθορίζει τα πεδία και τους μοναδικούς περιορισμούς για τον κύριο ή δευτερεύοντα πίνακα.
@ManyToMany	Ορίζει μια σχέση πολλά προς πολλά (many-to-many) μεταξύ των πινάκων.
@ManyToOne	Ορίζει μια σχέση πολλά προς ένα (many-to-one) μεταξύ των πινάκων.
@OneToMany	Ορίζει μια σχέση ένα προς πολλά (one-to-many) μεταξύ των πινάκων.

ANNOTATION	ΠΕΡΙΓΡΑΦΗ
@OneToOne	Ορίζει μια σχέση ένα προς ένα (one-to-one) μεταξύ των πινάκων.
@NamedQueries	Καθορίζει τη λίστα με τα ονόματα ερωτημάτων.

2.5.6 Σχέσεις Οντοτήτων

Η σχέση πολλά προς ένα (Many-To-One) μεταξύ των οντοτήτων υπάρχει όταν μια οντότητα (στήλη ή ομάδα στηλών) αναφέρεται σε μια άλλη οντότητα (στήλη ή ομάδα στηλών) που περιέχει μοναδικές τιμές. Στις σχεσιακές βάσεις δεδομένων οι σχέσεις αυτές εφαρμόζονται με τη χρήση ξένου κλειδιού μεταξύ των πινάκων.

Έστω ότι έχουμε μια σχέση ανάμεσα στην οντότητα υπάλληλος και στην οντότητα τμήμα. Από την σχέση υπαλλήλου προς τμήμα προκύπτει η σχέση πολλά προς ένα. Αυτό σημαίνει ότι κάθε αρχείο υπαλλήλου περιέχει ένα αναγνωριστικό τμήματος, το οποίο πρέπει να είναι πρωτεύον κλειδί στον πίνακα του τμήματος.



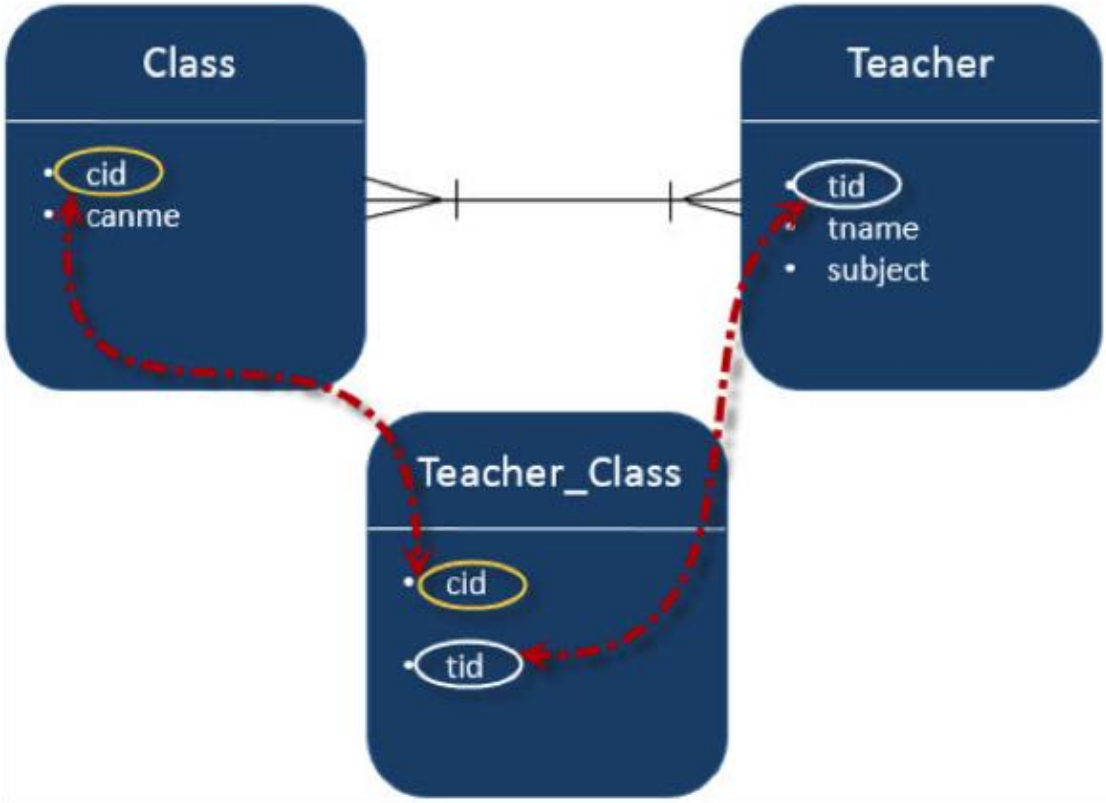
2.10 Σχέση many -to-one μεταξύ των πινάκων Employee και Τμήματος

Στη σχέση ένα προς πολλά (One-To-Many) κάθε σειρά μιας οντότητας αναφέρεται σε πολλές εγγραφές μιας άλλης οντότητας. Στο παραπάνω παράδειγμα αν υποθέσουμε ότι οι πίνακες υπαλλήλων και τμημάτων συνδέονται με έναν αντίστροφο τρόπο, τότε η σχέση γίνεται σχέση ένα προς πολλά.

Στη σχέση ένα προς ένα (One-To-One) ένα στοιχείο μπορεί να συνδεθεί με ένα μόνο άλλο στοιχείο. Σημαίνει, ότι κάθε σειρά μιας οντότητας αναφέρεται σε μία και μοναδική σειρά

μιας άλλης οντότητας. Σύμφωνα με το παραπάνω παράδειγμα, αν συνδέσουμε τον υπάλληλο και το τμήμα με έναν μοναδικό τρόπο, τότε η σχέση γίνεται σχέση ένα προς ένα. Κάθε εργαζόμενος ανήκει σε ένα μόνο τμήμα.

Η σχέση πολλά προς πολλά (Many-To-Many) συναντάται όταν μια ή περισσότερες σειρές από μια οντότητα, σχετίζονται με περισσότερες από μία σειρές σε μια άλλη οντότητα . Σύμφωνα με την εικόνα 2.11 τόσο ο πίνακας τάξη, όσο και ο πίνακας δάσκαλος έχουν μία σχέση πολλά προς πολλά. [12]



2.11 Σχέση many-to-many μεταξύ των κλάσεων Teacher και Class

2.5.7 JPQL

Η JPQL δεν είναι SQL. Παρά τις ομοιότητες μεταξύ των δυο γλωσσών στη δομή αλλά και στις λέξεις κλειδιά που χρησιμοποιούν, υπάρχουν αρκετές διαφορές. Αξίζει να

σημειώσουμε, ότι βασικότερη διαφορά είναι ο αντικειμενοστραφής χαρακτήρας της JPQL, ο οποίος απαιτεί και ένα διαφορετικό τρόπος σκέψης.

Η JPQL είναι μια γλώσσα η οποία εκτελεί ερωτήματα σε οντότητες. Αντί για πίνακες και σειρές που διαχειρίζεται η SQL, η JPQL διαχειρίζεται οντότητες και αντικείμενα. Μας παρέχει έναν τρόπο να εκφράσουμε ερωτήματα στις οντότητες με βάση τον τρόπο που έχουμε δηλώσει τα entities και τις μεταξύ τους σχέσεις και όχι με βάση τη φυσική σχέση που έχουν οι πίνακες στη βάση δεδομένων. Ορισμένοι από τους λόγους που χρησιμοποιούμε τη JPQL πάνω από τη SQL είναι η φορητότητα, καθώς η JPQL μεταφράζεται εύκολα σε SQL, αλλά και το γεγονός το ότι δεν χρειαζόμαστε να γνωρίζουμε με ακρίβεια τις αντιστοιχίες των entities που έχουμε δημιουργήσει με τη βάση δεδομένων. Ωστόσο, η υιοθέτηση της JPQL δεν σημαίνει ότι θα χάσουμε βασικές λειτουργίες της SQL, καθώς υπάρχουν πλήθος λειτουργιών που υποστηρίζονται από τη πρώτη.

Τα ερωτήματα (queries) εμπίπτουν σε μία από τις τέσσερις κατηγορίες:

- **Select queries:** Ανακτούν ένα ή περισσότερα entities φιλτράροντας τα αποτελέσματα όπως απαιτείται.
- **Aggregate queries:** Τα συγκεντρωτικά ερωτήματα (aggregate queries) είναι παραλλαγές από επιλεγμένα ερωτήματα (select queries) που ομαδοποιούν τα αποτελέσματα και παράγουν συνοπτικά δεδομένα. Τα ερωτήματα ονομάζονται μερικές φορές ερωτήματα αναφοράς, αφού επικεντρώνονται κατά κύριο λόγο στη δημιουργία δεδομένων
- **Update, Delete queries:** Η ενημέρωση και η διαγραφή ερωτημάτων χρησιμοποιούνται για την τροποποίηση ή την κατάργηση ολόκληρων συνόλων οντοτήτων.

Στα ερωτήματα οι οντότητες αναφέρονται με το όνομα. Εάν μια οντότητα δεν έχει ρητώς κατονομαστεί (για παράδειγμα χρησιμοποιώντας την ιδιότητα name του annotation @Entity), χρησιμοποιείται το όνομα της κλάσης. Αυτό το όνομα είναι το αφηρημένο όνομα σχήματος της οντότητας στο πλαίσιο ενός ερωτήματος. Οι οντότητες αποτελούνται από μία ή περισσότερες ιδιότητες οι οποίες εφαρμόζονται ως πεδία. Τέλος, είναι σημαντικό να σημειωθεί ότι τα ερωτήματα δεν είναι ευαίσθητα σε πεζά-κεφαλαία.

Η επιλογή ερωτημάτων (select queries) είναι ο πιο σημαντικός τύπος ερωτήματος ώστε να διευκολύνεται η μαζική ανάκτηση δεδομένων από το βάση δεδομένων. Η συνολική μορφή ενός επιλεγμένου ερωτήματος (select queries) έχει ως εξής:

SELECT <select expression> FROM <key> [WHERE <conditional expression>] [ORDER BY <order by clause>]

Η απλούστερη μορφή ενός ερωτήματος (select query) αποτελείται από δύο υποχρεωτικά τμήματα. Το τμήμα SELECT που ορίζει τη μορφή των αποτελεσμάτων των ερωτημάτων και το τμήμα FROM, το οποίο καθορίζει την οντότητα ή τις οντότητες από τις οποίες θα προκύψουν τα αποτελέσματα. Ένα παράδειγμα είναι το εξής:

SELECT e FROM Employee e

Η δομή αυτού του ερωτήματος είναι παρόμοια με ένα ερώτημα SQL αλλά με μερικές σημαντικές διαφορές. Η πρώτη διαφορά είναι ότι το τμήμα του ερωτήματος που ορίζεται στο τμήμα FROM δεν είναι ένας πίνακας, αλλά μία οντότητα και στην περίπτωση του παραπάνω παραδείγματος η οντότητα του εργαζομένου. Όπως και στην SQL, έτσι και εδώ χρησιμοποιούμε το e ως ψευδώνυμο (alias). Το e είναι γνωστό ως μεταβλητή ταυτοποίησης και είναι το κλειδί με το οποίο θα αναφερθεί η οντότητα στη συνέχεια του ερωτήματος select. Σε αντίθεση με τα ερωτήματα σε SQL όπου ένα ψευδώνυμο πίνακα είναι προαιρετικό, στη JPQL είναι υποχρεωτικό για να έχουμε πρόσβαση στις υπόλοιπες μεταβλητές .

Η δεύτερη διαφορά είναι ότι το τμήμα SELECT σε αυτό το παράδειγμα δεν απαριθμεί τα πεδία του πίνακα. Αντίθετα, η μεταβλητή ταυτοποίησης η οποία παρατίθεται για να γίνει το ερώτημα υποδηλώνει ότι ο τύπος αποτελέσματος του ερωτήματος είναι η οντότητα του υπαλλήλου και όχι ένα πλήθος γραμμών του πινάκα. Καθώς ο επεξεργαστής ερωτημάτων επεξεργάζεται το σύνολο αποτελεσμάτων που επιστρέφεται από τη βάση δεδομένων, μετατρέπει αυτά τα αποτελέσματα σε ένα σύνολο οντοτήτων. Η μέθοδος getResultList () της διεπαφής Query θα επιστρέψει μια συλλογή από μηδέν ή περισσότερα αντικείμενα υπαλλήλου μετά την εκτέλεση του ερωτήματος. Παρά τις διαφορές στη δομή και τη σύνταξη, κάθε ερώτημα μεταφράζεται σε SQL μέσα από ένα μηχανισμό ο οποίος δημιουργεί μια βέλτιστη αναπαράσταση SQL του ερωτήματος JPQL. Το προκύπτον SQL ερώτημα είναι αυτό που πραγματικά εκτελείται στη βάση δεδομένων. [13]

2.6 ΕΠΙΛΟΓΟΣ

Αδιαμφισβήτητα το Spring Framework είναι το πιο διαδεδομένο πλαίσιο ανάπτυξης της Java. Κάλλιστα, θα μπορούσε κάποιος να ισχυριστεί ότι το Spring είναι το πιο διαδεδομένο πλαίσιο ανάπτυξης, διότι είναι το πιο πλήρες. Ωστόσο, αυτός δεν είναι ο μόνος λόγος για τον οποίο το Spring έχει αυξήσει τη δημοτικότητα του. Ακόμα και σήμερα που έχουν αναπτυχθεί αρκετά πλαίσια ανάπτυξης στην Java, το Spring συνεχίζει να υπερτερεί σε χαρακτηριστικά, όπως είναι η γρήγορη ανάπτυξη, ο οργανωμένος κώδικας , η δυνατότητα επαναχρησιμοποίησης κώδικα, η απόδοση και η ενσωμάτωση του σε τρίτα συστήματα. Το τελευταίο χαρακτηριστικό είναι αυτό που το έχει ξεχωρίσει από τα υπόλοιπα πλαίσια. Μας δίνει τη δυνατότητα να αξιοποιήσουμε και άλλα πλαίσια και να δημιουργήσουμε μία στιβαρή και αξιόλογη εφαρμογή. Με αφορμή αυτό το προτέρημα έχουμε αξιοποιήσει στη συγκεκριμένη υλοποίηση το Spring Data JPA και όλες τις λειτουργίες τις οποίες παρέχει, με σκοπό να διαχειριστούμε πιο αποτελεσματικά την επικοινωνία και την ανταλλαγή δεδομένων μεταξύ της βάσης και της εφαρμογής.

ΚΕΦΑΛΑΙΟ 3

ΠΕΡΙΓΡΑΦΗ ΠΡΟΒΛΗΜΑΤΟΣ

Το κεφάλαιο αυτό πραγματεύεται την περιγραφή του προβλήματος. Αρχικά, κάνουμε μια αναφορά στην Ελληνική Ακαδημαϊκή Κοινότητα και στις υπηρεσίες που προσφέρει στα μέλη της, ενώ στη συνέχεια εστιάζουμε στο Uniway. Πιο συγκεκριμένα περιγράφουμε τι ακριβώς είναι το Uniway, ποιος είναι ο λόγος ύπαρξής του και τι έχει να προσφέρει στους φοιτητές. Τέλος, κάνουμε μια μικρή αναφορά στην υπηρεσία καταλόγου.

3.1 UNIWAY

Η Ελληνική Ακαδημαϊκή Κοινότητα (GUnet) είναι μία αστική εταιρεία μη κερδοσκοπικού χαρακτήρα με μέλη όλα τα ακαδημαϊκά ιδρύματα (20 Πανεπιστήμια και 16 ΤΕΙ). Αποτελεί ένα δίκτυο συνεργασίας των ΑΕΙ και των στελεχών τους προκειμένου να καταγράφονται οι κοινές ανάγκες, να σχεδιάζονται και να υλοποιούνται κοινές λύσεις και να πραγματοποιείται ανταλλαγή και διάχυση τεχνογνωσίας. Οι στόχοι της Ελληνικής Ακαδημαϊκής Κοινότητας προσδιορίζονται από τις ευρύτερες δικτυακές ανάγκες και επιδιώξεις της ακαδημαϊκής κοινότητας της χώρας, με σκοπό την εξυπηρέτηση της έρευνας και της εκπαίδευσης. Οι υπηρεσίες που παρέχει απευθύνονται προς τα μέλη της, τους ακαδημαϊκούς χρήστες αλλά και το ευρύ κοινό. [3]

Το Uniway είναι μια υπηρεσία που παρέχεται από το δίκτυο ελληνικών πανεπιστημίων και διατίθεται σε όλα τα μέλη της. Η ανάπτυξη του χρηματοδοτήθηκε από το Εθνικό Στρατηγικό Πλαίσιο Αναφοράς 2007-2015, ως μέρος μιας εθνικής πρωτοβουλίας με στόχο

την ανάπτυξη προηγμένων υπηρεσιών δικτύου στον ακαδημαϊκό κόσμο. Βασικός πυλώνας της λειτουργίας του Uniway Students Mobile Hub είναι η δυνατότητα συμμετοχής των φοιτητών στα ηλεκτρονικά ερωτηματολόγια αξιολόγησης μαθημάτων και η διασύνδεση του με τα αντίστοιχα ιδρυματικά πληροφοριακά συστήματα για την ενεργοποίηση ενός επιπλέον καναλιού συμμετοχής. Πιο συγκεκριμένα, είναι μια πλατφόρμα επικοινωνίας, που απευθύνεται σε φοιτητές και είναι διαθέσιμη σε περιβάλλον Android και iOS. Αξιοποιώντας την ευκολία και την αμεσότητα που παρέχει το μέσο έχει σχεδιασθεί για να παρέχει την ελευθερία και την ευελιξία των ανοικτών κοινωνικών δικτύων παράλληλα με εξειδικευμένες υπηρεσίες που αφορούν τους φοιτητές των Πανεπιστημίων και ΤΕΙ της χώρας. Στο πλαίσιο αυτό ενσωματώνει ένα βασικό πυλώνα πληροφόρησης που είναι τα ιδρυματικά πληροφοριακά συστήματα Ηλεκτρονικής Γραμματείας, με στόχο να παρέχει έγκαιρη και έγκυρη ενημέρωση για θέματα που αφορούν τη φοίτηση, την παρακολούθηση μαθημάτων, το πρόγραμμα σπουδών κ.α.. Η επικοινωνία της εφαρμογής με τις ιδρυματικές εφαρμογές είναι μόνο ανάγνωσης και πραγματοποιείται μέσω ενός ενδιάμεσου κόμβου διασύνδεσης, που διεκπεραιώνει θέματα διαλειτουργικότητας και αυθεντικοποίησης με τις ιδρυματικές υποδομές. Η ανάπτυξή του παρέχει πρόσβαση σε κάθε φοιτητή στα ακόλουθα: [4]

- **Βαθμολογίες:** Οι επίσημες βαθμολογίες ανά περίοδο, ανά χρονιά και μάθημα.
- **Ανακοινώσεις:** Ανακοινώσεις και ειδοποιήσεις, της Γραμματείας ενός Ιδρύματος.
- **Πρόγραμμα:** Το πρόγραμμα σπουδών.
- **Αξιολογήσεις Μαθημάτων:** Αξιολογήσεις μαθημάτων μέσα από εστιασμένα ερωτηματολόγια.
- **Δηλώσεις:** Οι δηλώσεις μαθημάτων.
- **Κοινωνική Δικτύωση:** Επικοινωνία εύκολα και γρήγορα με τους συμφοιτητές του, δωρεάν αποστολή γραπτών μηνυμάτων, δημιουργία ομάδων επικοινωνίας, κ.α.

3.2 ΥΠΗΡΕΣΙΑ ΚΑΤΑΛΟΓΟΥ LDAP

Στο πλαίσιο των δράσεων Διαχείρισης Ηλεκτρονικού Μητρώου Χρηστών της Ελληνικής Ακαδημαϊκής Κοινότητας για την ανάπτυξη σύγχρονων υποδομών διαχείρισης ηλεκτρονικών ταυτοτήτων, έχει αναπτυχθεί ένας πρότυπος μηχανισμός διασύνδεσης των συστημάτων ηλεκτρονικής γραμματείας φοιτητών με την υπηρεσία καταλόγου χρηστών του ιδρύματος (LDAP). Ο μηχανισμός αυτός εξασφαλίζει τη διαρκή και σε πραγματικό χρόνο ενημέρωση των στοιχείων που αφορούν τους ηλεκτρονικούς λογαριασμούς του ιδρύματος από το σύστημα διαχείρισης φοιτητών, εξασφαλίζοντας την ορθότητα και την ακρίβεια στην απόδοση των στοιχείων.

Ο συνήθης ρόλος των Υπηρεσιών Καταλόγου (LDAP) είναι η υλοποίηση του κέντρου ταυτοποίησης και αυθεντικοποίησης μέσα σε ένα οργανισμό για το σύνολο των χρηστών του. Έτσι για παράδειγμα είναι αναμενόμενο, τόσο η υπηρεσία Ηλεκτρονικού Ταχυδρομείου, όσο και μια διοικητική εφαρμογή να βασίζεται για την ταυτοποίηση και αυθεντικοποίηση των χρηστών τους στην Υπηρεσία Καταλόγου του ιδρύματος. Αυτός ο ειδικός ρόλος των Υπηρεσιών Καταλόγου συχνά δημιουργεί την παρανόηση ότι η Υπηρεσία Καταλόγου είναι η πρωτογενής πηγή για το σύνολο των στοιχείων που παρέχει. Αυτό είναι λάθος, καθώς εξ' ορισμού οι Υπηρεσίες Καταλόγου διαχειρίζονται πρωτογενώς μόνο ένα μικρό υποσύνολο των στοιχείων που παρέχουν στις δικτυακές υπηρεσίες, όπως για παράδειγμα το uid, το user password, το email κ.α.. Σπάνια οι υπηρεσίες καταλόγου θα δούμε να υλοποιούν το πρωτεύον σύστημα αποθήκευσης στοιχείων μιας επιχειρησιακής εφαρμογής αντικαθιστώντας για παράδειγμα μια βάση δεδομένων. Ακριβώς, με αντίστοιχη λογική το πληροφοριακό σύστημα δεν μπορεί να θεωρηθεί ότι πρωτογενώς διαχειρίζεται την διεύθυνση email των φοιτητών, ή τη βαθμίδα των διδασκόντων, ακόμα και αν τα στοιχεία αυτά περιλαμβάνονται στη βάση δεδομένων του συστήματος. Αντίθετα, η έγκυρη πηγή πληροφορίας για τη διεύθυνση email είναι η Υπηρεσία Καταλόγου, καθώς αυτή ερωτούν τα συστήματα Ηλεκτρονικού Ταχυδρομείου για τη δρομολόγηση των emails, ενώ η έγκυρη πηγή πληροφορίας για τη βαθμίδα των διδασκόντων είναι το Πληροφοριακό Σύστημα Διαχείρισης Προσωπικού. Προκειμένου λοιπόν να διασφαλιστεί η εγκυρότητα και η ποιότητα των στοιχείων που συγκεντρώνονται στην ιδρυματική Υπηρεσία Καταλόγου είναι απαραίτητο να εντοπισθούν πρώτα ποια είναι τα πρωτογενή Πληροφοριακά Συστήματα για κάθε στοιχείο πληροφορίας που παρέχουν, δηλαδή να

εντοπισθούν τα συστήματα, τα οποία αποτελούν διοικητικά και διαχειριστικά μοναδική έγκυρη πηγή άντλησης της πληροφορίας. Αυτή η προσέγγιση αποτελεί την Αρχή Έγκυρης Πηγής. Η συμμόρφωση με την Αρχή Έγκυρης Πηγής πρακτικά σημαίνει ότι τα πρωτογενή στοιχεία που συντηρούνται σε κάθε πληροφοριακό σύστημα πρέπει να επαναχρησιμοποιούνται από άλλα πληροφοριακά συστήματα συμβάλλοντας α) στην ακρίβεια και ομοιογένεια των διαθέσιμων στοιχείων και β) στην απλοποίηση των διαδικασιών για το τελικό χρήστη αποφεύγοντας την επαναληπτική παροχή των ιδίων στοιχείων σε κάθε νέα υπηρεσία που απολαμβάνει. Ακολουθώντας την παραπάνω αρχή θεωρούμε ότι το Πληροφοριακό Σύστημα Ηλεκτρονικής Γραμματείας Φοιτητών ενός ιδρύματος είναι το πρωτογενές σύστημα για:

- Στοιχεία που αφορούν τους φοιτητές για τα οποία το πληροφοριακό σύστημα αποτελεί το μοναδικό σημείο εισόδου στο ίδρυμα.
- Στοιχεία που αφορούν στο τμήμα ή το πρόγραμμα σπουδών φοίτησης.
- Στοιχεία που αφορούν στη φοιτητική κατάσταση.
- Προσωπικά στοιχεία ταυτότητας, όπως το όνομα, το επώνυμο κ.α.

Για τους ειδικούς σκοπούς της Υπηρεσίας Καταλόγου, μόλις ένα μικρό υποσύνολο των διαθέσιμων στοιχείων είναι χρήσιμα για την ταυτοποίηση και εξουσιοδότηση σε δικτυακές υπηρεσίες. Για παράδειγμα, δεν υπάρχει λόγος απεικόνισης στοιχείων, όπως η τρέχουσα δήλωση μαθημάτων του φοιτητή κ.α., τα οποία θα ενδιέφεραν ένα πολύ μικρό ποσοστό εφαρμογών. Αντίθετα, στοιχεία του χρήστη, όπως το επίπεδο φοίτησης (προπτυχιακό, μεταπτυχιακό, διδακτορικό) ή αν βρίσκεται σε κατάσταση αναστολής φοίτησης που θα ήταν χρήσιμα για την εξουσιοδότηση της πρόσβασης σε περισσότερες από μια εφαρμογές μπορούν να συμμετέχουν στον πυρήνα των στοιχείων του χρήστη που γίνονται διαθέσιμα μέσω της Υπηρεσίας Καταλόγου. [3]

3.3 ΕΠΙΛΟΓΟΣ

Αναντίρρητα, η δημιουργία της Ελληνικής Ακαδημαϊκής Κοινότητας είναι προς όφελος των μελών της. Μέσα από κοινές ανάγκες και προβλήματα αναπτύσσονται υπηρεσίες και μοιράζεται τεχνογνωσία που μόνο θετικά θα μπορούσε να επηρεάσει το σύνολο των ιδρυμάτων. Το Uniway είναι αποτέλεσμα αυτής της συλλογικής προσπάθειας, το οποίο

έχει ως στόχο τη δημιουργία μιας ενιαίας πλατφόρμας επικοινωνίας για να παρέχει έγκαιρη και έγκυρη ενημέρωση των φοιτητών για θέματα που αφορούν τη φοίτηση, την παρακολούθηση μαθημάτων και το πρόγραμμα σπουδών.

ΚΕΦΑΛΑΙΟ 4

ΥΛΟΠΟΙΗΣΗ

Το κεφάλαιο τέσσερα πραγματεύεται την υλοποίηση του θέματος. Αρχικά, θα αναφερθούμε στις κλήσεις της εφαρμογής, ενώ στη συνέχεια θα μιλήσουμε για τη δομή του κώδικα. Πιο συγκεκριμένα, θα κάνουμε λόγο για τη μορφή των κλήσεων αλλά και τη μορφή των δεδομένων που θα επιστρέφονται μέσω αυτών. Τέλος, θα αναφερθούμε με περισσότερες λεπτομέρειες για την υλοποίηση της εφαρμογής μέσω του Spring Framework.

4.1 ΚΛΗΣΕΙΣ ΕΦΑΡΜΟΦΗΣ

Η αλληλεπίδραση του χρήστη με μία εφαρμογή έχει ως απόρροια την ανταλλαγή δεδομένων. Ο χρήστης μέσα από διάφορες ενέργειες κάνει κλήσεις προς την εφαρμογή για να του παρέχει πληροφορίες που αυτός ζήτησε. Αυτές τις ενέργειες θα μπορούσαμε να τις χαρακτηρίσουμε ως κλήσεις της εφαρμογής. Στη συγκεκριμένη υλοποίηση οι κλήσεις της εφαρμογής περιλαμβάνουν τα στοιχεία του φοιτητή, το πρόγραμμα σπουδών, τη βαθμολογία και τις δηλώσεις μαθημάτων.

4.1.1 Στοιχεία Φοιτητή

Μια από τις κλήσεις η οποία είναι διαθέσιμη στην εφαρμογή είναι τα στοιχεία του φοιτητή. Για να μπορέσει ένας φοιτητής να λάβει πληροφορίες σχετικά με το όνομα του, το επίθετο του αλλά και τα υπόλοιπα στοιχεία τα οποία είναι διαθέσιμα για αυτόν, αρκεί να κάνει τη συγκεκριμένη κλήση στην εφαρμογή. Τα στοιχεία, τα οποία είναι διαθέσιμα προς το φοιτητή, φαίνονται στο πίνακα 4.1. (GUnet, 2015)

Πίνακας 4.1 Στοιχεία φοιτητή

ΣΤΟΙΧΕΙΑ	**ΠΕΡΙΓΡΑΦΗ**
AcademicID	Μοναδικό Id σύμφωνα με το https://academicid.gunet.gr
RegistrationID	Αριθμός Μητρώου φοιτητή. Μοναδικός για το σύνολο των φοιτητών του ιδρύματος
DepartmentID	Μοναδικός κωδικός του τμήματος φοίτησης
DepartmentEn	Η επίσημη ονομασία του τμήματος στα Αγγλικά
DepartmentEl	Η επίσημη ονομασία του τμήματος στα Ελληνικά
ProgramID	Μοναδικός κωδικός του προγράμματος σπουδών που ακολουθεί ο φοιτητής
ProgramEn	Η επίσημη ονομασία του προγράμματος σπουδών στα Αγγλικά
ProgramEl	Η επίσημη ονομασία του προγράμματος σπουδών στα Ελληνικά
FirstNameEn	Το όνομα του φοιτητή με απόδοση σε αγγλικούς χαρακτήρες
FirstNameEl	Το όνομα του φοιτητή με απόδοση σε ελληνικούς χαρακτήρες
LastNameEn	Το επώνυμο του φοιτητή με απόδοση σε αγγλικούς χαρακτήρες
LastNameEl	Το επώνυμο του φοιτητή με απόδοση σε ελληνικούς χαρακτήρες
FatherNameEn	Το όνομα πατρός του φοιτητή με απόδοση σε αγγλικούς χαρακτήρες
FatherNameEl	Το όνομα πατρός του φοιτητή με απόδοση σε ελληνικούς χαρακτήρες
BirthDate	Η ημερομηνία γέννησης του φοιτητή σύμφωνα με το πρότυπο ISO 8601:2004 (π.χ. 19950924)
Citizenship	Η ιθαγένεια / υπηκοότητα του φυσικού προσώπου (2-γράμματη κωδικοποίηση του ISO 3166)

ΣΤΟΙΧΕΙΑ	ΠΕΡΙΓΡΑΦΗ
Gender	Το φύλο του φοιτητή σύμφωνα με το πρότυπο ISO/IEC 5218:2004 (π.χ.19950924)
EnrollmentType	Κατηγοριοποίηση του φοιτητή με βάση το επίπεδο του προγράμματος σπουδών που είναι εγγεγραμμένος
EnrollmentTerm	Περίοδος φοίτησης του φοιτητή (π.χ. 8)
AttendanceType	Τύπος φοίτησης φοιτητή (π.χ. full-time)
EnrollmentStatus	Τρέχουσα κατάσταση φοιτητή (π.χ. active)
EnrollmentStatusDate	Η ημερομηνία από την οποία βρίσκεται σε ισχύ η κατάσταση που αναφέρεται από το πεδίο enrollmentStatus
InscriptionAcYear	Το ακαδημαϊκό έτος εγγραφής του φοιτητή (σε μορφή YYYY)
InscriptionTerm	Η ακαδημαϊκή περίοδος εγγραφής του φοιτητή (π.χ. 1)
LoginName	Όνομα χρήστη δικτυακού λογαριασμού φοιτητή
Email	Email
Phone	Τηλέφωνο επικοινωνίας
Mobile	Τηλέφωνο επικοινωνίας

4.1.2 Πρόγραμμα Σπουδών Φοιτητή

Μια από τις κλήσεις η οποία είναι διαθέσιμη στην εφαρμογή είναι το πρόγραμμα σπουδών του φοιτητή. Για να μπορέσει ένας φοιτητής να λάβει πληροφορίες σχετικά με τα μαθήματα που είναι διαθέσιμα στο τρέχον πρόγραμμα σπουδών αλλά και για τους καθηγητές, οι οποίοι διδάσκουν τα συγκεκριμένα μαθήματα, αρκεί να κάνει τη συγκεκριμένη κλήση στην εφαρμογή. Τα στοιχεία τα οποία είναι διαθέσιμα προς το φοιτητή, φαίνονται στο πίνακα 4.2. (GUnet, 2015)

Πίνακας 4.2 Πρόγραμμα σπουδών φοιτητή

ΣΤΟΙΧΕΙΑ	ΠΕΡΙΓΡΑΦΗ
Id	Μοναδικό Id για το μάθημα
DisplayCode	Κωδικός μαθήματος

ΣΤΟΙΧΕΙΑ	**ΠΕΡΙΓΡΑΦΗ**
Title	Τίτλος
Description	Περιγραφή μαθήματος
Semester	Περιγραφή εξαμήνου
SemesterType	0:Εαρινό, 1:Χειμερινό
ProgramId	Μοναδικός κωδικός του προγράμματος σπουδών
CourseType	Τύπος μαθήματος (υποχρεωτικό, προαιρετικό, ...)
WeekHours	Ώρες εβδομαδιαίως
TotalHours	Συνολικές ώρες
PassGrade	Βαθμός βάσης
Units	Πιστωτικές μονάδες
Ects	Μονάδες ECTS
Coefficient	Συντελεστής βαρύτητας
Role	Ρόλος(Υπεύθυνος Μαθήματος, Υπεύθυνος Εργαστηρίου, Διδάσκων)
AcadenicId	Μοναδικό Id σύμφωνα με το https://academicid.gunet.gr
LoginName	Username λογαριασμού καθηγητή
DisplayName	Ονοματεπώνυμο
Email	Email
Phone	Τηλέφωνο επικοινωνίας

4.1.3 Βαθμολογία Φοιτητή

Μια από τις κλήσεις η οποία είναι διαθέσιμη στην εφαρμογή είναι η βαθμολογία του φοιτητή. Για να μπορέσει ένας φοιτητής να λάβει πληροφορίες σχετικά με τα μαθήματα και τις αντίστοιχες βαθμολογίες τους που έχει δώσει σε κάθε εξεταστική περίοδο, αρκεί να κάνει τη συγκεκριμένη κλήση στην εφαρμογή. Τα στοιχεία, τα οποία είναι διαθέσιμα προς το φοιτητή, φαίνονται στο πίνακα 4.3. (GUnet, 2015)

Πίνακας 4.3 Βαθμολογία φοιτητή

ΣΤΟΙΧΕΙΑ	**ΠΕΡΙΓΡΑΦΗ**
ExamPeriodId	Μοναδικό Id για την εξεταστική περίοδο
Description	Περιγραφή εξεταστικής
AcademicYear	Ακαδημαϊκό έτος
Id	Μοναδικό Id για το βαθμό / μάθημα
DisplayCode	Κωδικός μαθήματος
Title	Τίτλος
CourseType	Τύπος μαθήματος (υποχρεωτικό, προαιρετικό, ...)
Semester	Περιγραφή εξαμήνου
Grade	Βαθμός
Passed	Boolean: Προβιβάσιμος Βαθμός
includedInDegree	Boolean: Προσμετράται στο Πτυχίο
isExam	Boolean true: Από εξέταση, false: Από αναγνώριση
PassGrade	Βαθμός βάσης
Units	Πιστωτικές μονάδες
Ects	Μονάδες ECTS
Coefficient	Συντελεστής βαρύτητας

4.1.4 Δηλώσεις Μαθημάτων Των Φοιτητών

Μια από τις κλήσεις, η οποία είναι διαθέσιμη στην εφαρμογή είναι οι δηλώσεις μαθημάτων του φοιτητή. Για να μπορέσει ένας φοιτητής να λάβει πληροφορίες σχετικά με τα μαθήματα τα οποία είναι διαθέσιμα την συγκεκριμένη ακαδημαϊκή περίοδο, αρκεί να κάνει τη συγκεκριμένη κλήση στην εφαρμογή. Τα στοιχεία, τα οποία είναι διαθέσιμα προς το φοιτητή, φαίνονται στο πίνακα 4.4. (GUnet, 2015)

Πίνακας 4.4 Δηλώσεις μαθημάτων φοιτητών

ΣΤΟΙΧΕΙΑ	**ΠΕΡΙΓΡΑΦΗ**
RegistrationPeriodId	Μοναδικό Id για την περίοδο
AcademicYear	Περιγραφή περιόδου
Description	Ακαδημαϊκό Έτος
Id	Μοναδικό Id για το μάθημα
DisplayCode	Κωδικός μαθήματος
Title	Τίτλος
CourseType	Τύπος μαθήματος
Semester	Περιγραφή εξαμήνου
PassGrade	Βαθμός Βάσης
Units	Πιστωτικές μονάδες
Ects	Μονάδες ECTS
Coefficient	Συντελεστής βαρύτητας

4.1.5 Φίλτρα Προγράμματος

Μία από τις κλήσεις, η οποία είναι διαθέσιμη στην εφαρμογή είναι τα φίλτρα προγράμματος. Τα φίλτρα είναι κλήσεις οι οποίες μπορούν να εφαρμοστούν σε ένα πρόγραμμα σπουδών για να περιοριστεί ο επιστρεφόμενος αριθμός μαθημάτων σε κάποιο ακαδημαϊκό έτος. Τα φίλτρα τα οποία είναι διαθέσιμα προς το φοιτητή φαίνονται στο πίνακα 4.5. (GUnet, 2015)

Πίνακας 4.5 Φίλτρα προγράμματος

ΦΙΛΤΡΑ	**ΠΕΡΙΓΡΑΦΗ**
Μαθήματα προγράμματος	Επιστρέφει τα μαθήματα που προσφέρονται σε ένα πρόγραμμα σπουδών σε κάποιο συγκεκριμένο ακαδημαϊκό έτος

ΦΙΛΤΡΑ	ΠΕΡΙΓΡΑΦΗ
Μαθήματα διδάσκοντα	Επιστρέφει τα μαθήματα τα οποία έχει αναθέσει κάποιος καθηγητής σε κάποιο ακαδημαϊκό έτος
Ύπαρξη βαθμολογίας	Επιστρέφει εάν έχει κατατεθεί βαθμολογία για κάποιο μάθημα σε κάποιο ακαδημαϊκό έτος
Λίστα φοιτητών για το μάθημα	Επιστρέφει μια λίστα με τους φοιτητές που έχουν δηλώσει κάποιο μάθημα σε κάποιο ακαδημαϊκό έτος

4.2 ΟΝΤΟΤΗΤΕΣ ΕΦΑΡΜΟΓΗΣ

Μία από τις αρχές του προγραμματισμού είναι η σωστή μοντελοποίηση των οντοτήτων της εφαρμογής. Σύμφωνα με τη μοντελοποίηση οι διάφορες έννοιες μέσα σε μία εφαρμογή αναπαριστώνται από οντότητες. Για παράδειγμα, αν υλοποιούσαμε μία εφαρμογή η οποία θα διαχειρίζεται αυτοκίνητα, το αυτοκίνητο θα ήταν μια οντότητα ή καλύτερα μια κλάση μέσα στην εφαρμογή. Στη δικιά μας περίπτωση οι οντότητες της εφαρμογής είναι ο καθηγητής, ο φοιτητής, το μάθημα, ο βαθμός, η εξεταστική περίοδος και η περίοδος εγγραφής.

Examperiod είναι η οντότητα η οποία αντιπροσωπεύει την εξεταστική περίοδο. Η εξεταστική περίοδος αποτελείται από τα χαρακτηριστικά τα οποία φαίνονται στο πίνακα 4.6. (GUnet, 2015)

Πίνακας 4.6 Examperiod

ΣΤΟΙΧΕΙΑ	ΠΕΡΙΓΡΑΦΗ
ExamPeriodId	Μοναδικό Id για την εξεταστική περίοδο
Description	Περιγραφή εξεταστικής
AcademicYear	Ακαδημαϊκό έτος

Registrationperiod είναι η οντότητα η οποία αντιπροσωπεύει την περίοδο δήλωσης των μαθημάτων. Η περίοδος αποτελείται από τα χαρακτηριστικά τα οποία φαίνονται στο πίνακα 4.7. (GUnet, 2015)

Πίνακας 4.7 Registrationperiod

ΣΤΟΙΧΕΙΑ	**ΠΕΡΙΓΡΑΦΗ**
RegistrationPeriodId	Μοναδικό Id για την περίοδο
Description	Περιγραφή περιόδου
AcademicYear	Ακαδημαϊκό έτος

Η οντότητα teacher αντιπροσωπεύει τον καθηγητή. Αποτελείται από τα χαρακτηριστικά τα οποία φαίνονται στο πίνακα 4.8. (GUnet, 2015)

Πίνακας4.8 Teacher

ΣΤΟΙΧΕΙΑ	**ΠΕΡΙΓΡΑΦΗ**
AcademicId	Μοναδικό Id σύμφωνα με το https://academicid.gunet.gr
DisplayName	Όνομα χρήστη δικτυακού λογαριασμού καθηγητή
LoginName	Όνομα χρήστη δικτυακού λογαριασμού καθηγητή
Email	Email
Phone	Τηλέφωνο επικοινωνίας

Student είναι η οντότητα η οποία αντιπροσωπεύει τον φοιτητή. Η οντότητα φοιτητής αποτελείται από τα χαρακτηριστικά, τα οποία φαίνονται στο πίνακα 4.9. (GUnet, 2015)

Πίνακας 4.9 Student

ΣΤΟΙΧΕΙΑ	**ΠΕΡΙΓΡΑΦΗ**
AcademicID	Μοναδικό Id σύμφωνα με το https://academicid.gunet.gr
RegistrationID	Αριθμός Μητρώου φοιτητή. Μοναδικός, για το σύνολο των φοιτητών του ιδρύματος
DepartmentID	Μοναδικός κωδικός του τμήματος φοίτησης
DepartmentEn	Η επίσημη ονομασία του τμήματος στα Αγγλικά
DepartmentEl	Η επίσημη ονομασία του τμήματος στα Ελληνικά
ProgramID	Μοναδικός κωδικός του προγράμματος σπουδών φοιτητής

ΣΤΟΙΧΕΙΑ	**ΠΕΡΙΓΡΑΦΗ**
ProgramEn	Η επίσημη ονομασία του προγράμματος σπουδών στα Αγγλικά
ProgramEl	Η επίσημη ονομασία του προγράμματος σπουδών στα Ελληνικά
FirstNameEn	Το όνομα του φοιτητή με απόδοση σε αγγλικούς χαρακτήρες
FirstNameEl	Το όνομα του φοιτητή με απόδοση σε ελληνικούς χαρακτήρες
LastNameEn	Το επώνυμο του φοιτητή με απόδοση σε αγγλικούς χαρακτήρες
LastNameEl	Το επώνυμο του φοιτητή με απόδοση σε ελληνικούς χαρακτήρες
FatherNameEn	Το όνομα πατρός του φοιτητή με απόδοση σε αγγλικούς χαρακτήρες
FatherNameEl	Το όνομα πατρός του φοιτητή με απόδοση σε ελληνικούς χαρακτήρες
BirthDate	Η ημερομηνία γέννησης του φοιτητή σύμφωνα με το πρότυπο ISO 8601:2004 (π.χ. 19950924)
Citizenship	Η ιθαγένεια / υπηκοότητα του φυσικού προσώπου (2-γράμματη κωδικοποίηση του ISO 3166)
Gender	Το φύλο του φοιτητή σύμφωνα με το πρότυπο ISO/IEC 5218:2004 (π.χ.19950924)
EnrollmentType	Κατηγοριοποίηση του φοιτητή με βάση το επίπεδο του προγράμματος σπουδών που είναι εγγεγραμμένος
EnrollmentTerm	Περίοδος φοίτησης του φοιτητή (π.χ. 8)
AttendanceType	Τύπος φοίτησης φοιτητή (π.χ. full-time)
EnrollmentStatus	Τρέχουσα κατάσταση φοιτητή (π.χ. active)
EnrollmentStatusDate	Η ημερομηνία από την οποία βρίσκεται σε ισχύ η κατάσταση που αναφέρεται από το πεδίο enrollmentStatus
InscriptionAcYear	Το ακαδημαϊκό έτος εγγραφής του φοιτητή (σε μορφή YYYY)
InscriptionTerm	Η ακαδημαϊκή περίοδος εγγραφής του φοιτητή (π.χ. 1)
LoginName	Όνομα χρήστη δικτυακού λογαριασμού φοιτητή
Email	Email
Phone	Τηλέφωνο επικοινωνίας
Mobile	Τηλέφωνο επικοινωνίας

Lesson είναι η οντότητα η οποία αντιπροσωπεύει ένα μάθημα μέσα στο πρόγραμμα σπουδών. Το μάθημα αποτελείται από τα χαρακτηριστικά τα οποία φαίνονται στο πίνακα 4.10. (GUnet, 2015)

Πίνακας 4.10 Lesson

ΣΤΟΙΧΕΙΑ	**ΠΕΡΙΓΡΑΦΗ**
Id	Μοναδικό Id για το μάθημα
DisplayCode	Κωδικός μαθήματος
Title	Τίτλος
Description	Περιγραφή μαθήματος
Semester	Περιγραφή εξαμήνου
SemesterType	0:Εαρινό, 1:Χειμερινό
ProgramId	Μοναδικός κωδικός του προγράμματος σπουδών
CourseType	Τύπος μαθήματος (υποχρεωτικό, προαιρετικό, ...)
WeekHours	Ώρες εβδομαδιαίως
TotalHours	Συνολικές ώρες
PassGrade	Βαθμός βάσης
Units	Πιστωτικές μονάδες
Ects	Μονάδες ECTS
Coefficient	Συντελεστής βαρύτητας
Role	Ρόλος(Υπεύθυνος Μαθήματος, Υπεύθυνος Εργαστηρίου, Διδάσκων)
AcadenicId	Μοναδικό Id σύμφωνα με το https://academicid.gunet.gr
LoginName	Username λογαριασμού καθηγητή
DisplayName	Ονοματεπώνυμο
Email	Email
Phone	Τηλέφωνο επικοινωνίας

Grade είναι η οντότητα η οποία αντιπροσωπεύει το βαθμό ενός μαθήματος στο πρόγραμμα σπουδών. Ο βαθμός αποτελείται από τα χαρακτηριστικά τα οποία φαίνονται στο πίνακα 4.11. (GUnet, 2015)

Πίνακας 4.11 Grade

ΣΤΟΙΧΕΙΑ	**ΠΕΡΙΓΡΑΦΗ**
ExamPeriodId	Μοναδικό Id για την εξεταστική περίοδο
Description	Περιγραφή εξεταστικής
AcademicYear	Ακαδημαϊκό έτος
Id	Μοναδικό Id για το βαθμό / μάθημα
DisplayCode	Κωδικός μαθήματος
Title	Τίτλος
CourseType	Τύπος μαθήματος (υποχρεωτικό, προαιρετικό, ...)
Semester	Περιγραφή εξαμήνου
Grade	Βαθμός
Passed	Boolean: Προβιβάσιμος Βαθμός
includedInDegree	Boolean: Προσμετράται στο Πτυχίο
isExam	Boolean true: Από εξέταση, false: Από αναγνώριση
PassGrade	Βαθμός βάσης
Units	Πιστωτικές μονάδες
Ects	Μονάδες ECTS
Coefficient	Συντελεστής βαρύτητας

4.3 ΔΟΜΗ ΤΟΥ ΚΩΔΙΚΑ

Για να μπορέσουμε να υλοποιήσουμε τις οντότητες που αναφέραμε στην ενότητα 4.2 και να επιστρέψουμε τα δεδομένα στην κατάλληλη μορφή σύμφωνα με την ενότητα 4.1, κάνουμε χρήση του Spring Framework. Ορίζουνε κλάσεις οι οποίες είναι υπεύθυνες για την αντιστοίχηση αιτημάτων (request mapping). Με τον όρο αντιστοίχηση αιτημάτων εννοούμε το ταίριασμα ενός http αιτήματος, το οποίο φθάνει στον εξυπηρετητή (server) από την εφαρμογή. Για το λόγο αυτό το Spring μας παρέχει ορισμένα annotations. Στη προκειμένη περίπτωση για να δηλώσουμε τους ελεγκτές χρησιμοποιήσαμε το annotation (@Controller). Στη εφαρμογή έχουμε τους εξής ελεγκτές (controllers):

- **StudentController**: Είναι ο ελεγκτής ο οποίος επεξεργάζεται τα αιτήματα που σχετίζονται με το φοιτητή (/sis/v1/student/{registrationId}).
- **CurriculumController**: Είναι ο ελεγκτής ο οποίος επεξεργάζεται αιτήματα που σχετίζονται με το πρόγραμμα σπουδών. Πιο συγκεκριμένα επεξεργάζεται αιτήματα που σχετίζονται με τα μαθήματα τα οποία είναι διαθέσιμα στο τρέχον ακαδημαϊκό έτος (/sis/v1/curriculum/{registrationId}), τα διάφορα φίλτρα τα οποία μπορούν να εφαρμοστούν σε ένα πρόγραμμα σπουδών για να περιοριστεί ο επιστρεφόμενος αριθμός μαθημάτων σε κάποιο ακαδημαϊκό έτος (/sis/v1/lessonfilters/{programId}/{academicYear}) ή τα μαθήματα τα οποία διδάσκει κάποιος καθηγητής (/sis/v1/teacherlessons/1/loginName/academicYear).
- **StudentGradesController**: Είναι ο ελεγκτής ο οποίος επεξεργάζεται τα αιτήματα που σχετίζονται με τη βαθμολογία του φοιτητή. Αιτήματα όπως η βαθμολογία των μαθημάτων (sis/v1/grades/{registrationId}) ή το αν έχει κατατεθεί βαθμολογία για κάποιο μάθημα σε κάποιο ακαδημαϊκό έτος (sis/v1/lessongrades/{academicYear}/{semesterType}/{id}).
- **StudentRegistrationController**: Είναι ο ελεγκτής ο οποίος επεξεργάζεται τα αιτήματα που σχετίζονται με τις δηλώσεις των μαθημάτων του φοιτητή (sis/v1/courses/registrationId).

Οι παραπάνω ελεγκτές χρησιμοποιούν διάφορες υπηρεσίες (services). Το Spring μας παρέχει το annotation (@Service) για να ορίσουμε κλάσεις οι οποίες εκτελούν διάφορες υπηρεσίες, όπως είναι η εκτέλεση υπολογισμών, κλήσεις σε εξωτερικά συστήματα (APIS), εφαρμογή της επιχειρησιακής λογικής. Στη εφαρμογή έχουμε τις εξής υπηρεσίες:

- **StudentService**
- **StudentRegistrationService**
- **StudentGradesService**
- **CurriculumService**
- **StaffService**

Οι παραπάνω υπηρεσίες είναι υπεύθυνες για να καλούν τα κατάλληλα αποθετήρια. Αποθετήριο (@Repository) είναι ένα Spring annotation το οποίο προσδιορίζει ότι η συγκεκριμένη κλάση είναι ένα αποθετήριο. Ως αποθετήριο, ορίζουμε εκείνες της κλάσεις

οι οποίες έχουνε πρόσβαση στη βάση δεδομένων. Στη εφαρμογή έχουμε τα εξής αποθετήρια:

- **CurriculumRepository**
- **StudentRepository-**
- **StudentGradesRepository**
- **StudentRegistrationRepository**
- **StaffRepository**

Ο πίνακας 5 δείχνει κάποιους ενδεικτικούς χρόνους απόκρισης στις κλήσεις της εφαρμογής.

Πίνακας 5 Χρόνοι απόκρισης της εφαρμογής

ΚΛΗΣΕΙΣ ΕΦΑΡΜΟΦΗΣ	**ΧΡΟΝΟΣ ΑΠΟΚΡΙΣΗΣ**
Στοιχεία φοιτητή	339ms
Πρόγραμμα σπουδών φοιτητή	1.5s
Βαθμολογία φοιτητή	1s
Δηλώσεις μαθημάτων φοιτητή	2s
Μαθήματα προγράμματος	400ms
Μαθήματα διδάσκοντα	400ms
Ύπαρξη βαθμολογίας	400ms
Λίστα φοιτητών για το μάθημα	635ms

4.4 ΕΓΚΑΤΑΣΤΑΣΗ ΕΦΑΡΜΟΓΗΣ

‘Οσο αναφορά το πρακτικό κομμάτι, για να μπορέσουμε να υλοποιήσουμε την εφαρμογή χρησιμοποιήσαμε ένα περιβάλλον ανάπτυξης κώδικα το οποίο ονομάζεται Intelij IDEA. Το συγκεκριμένο περιβάλλον ανάπτυξης κώδικα συνεργάζεται με το gradle, το οποίο είναι ένα αυτοματοποιημένο σύστημα που διευκολύνει στο χτίσιμο της εφαρμογής. Το gradle μας κάνει διαθέσιμο ένα αρχείο (build.gradle), μέσα στο οποίο μπορούμε να δηλώσουμε όλες εκείνες τις βιβλιοθήκες τις οποίες χρειάζεται η εφαρμογή για να λειτουργήσει.

Επίσης μας δίνει τη δυνατότητα μέσα από τις διάφορες διεργασίες (task) να παράγουμε ένα εκτελέσιμο αρχείο στο οποίο είναι ενσωματωμένες οι βιβλιοθήκες τις οποίες χρειάζεται. Ακολουθώντας τα παραπάνω βήματα η εφαρμογή είναι έτοιμη για να την ανεβάσουμε (deploy) στο μηχάνημα το οποίο μας έχει κάνει διαθέσιμο η σχολή. Τα βήματα τα οποία ακολουθούμε για να κάνουμε deploy την εφαρμογή είναι τα εξής:

- Ανοίγουμε ένα τερματικό σταθμό και πληκτρολογούμε την εντολή ssh με την διεύθυνση ip και το συνθηματικό για να μπορέσουμε να συνδεθούμε με το μηχάνημα το οποίο βρίσκεται στη συγκεκριμένη διεύθυνση. Για παράδειγμα εκτελούμε την εντολή ssh username@ip.
- Στη συνέχεια εκτελούμε την εντολή scp για να ανεβάσουμε (upload) την εφαρμογή στο μηχάνημα. Για παράδειγμα scp <local path> <host path>.
- Κάνουμε εγκατάσταση την java.
- Εκτελούμε την εφαρμογή.

4.5 ΕΠΙΛΟΓΟΣ

Το κεφάλαιο τέσσερα πραγματεύεται την υλοποίηση του JSON API. Έχοντα ως βασικό εργαλείο το Spring Framework αλλά και τον οδηγό που μας έχει κάνει διαθέσιμο η Ελληνική Ακαδημαϊκή Κοινότητα, αποσκοπούμε στη μελλοντική διασύνδεση του ιδρυματικού πληροφοριακού συστήματος ηλεκτρονικής γραμματείας φοιτητών με την εφαρμογή Uniway.

ΚΕΦΑΛΑΙΟ 5

ΣΥΜΠΕΡΑΣΜΑΤΑ

Το Uniway παρέχεται από τη Ελληνική Ακαδημαϊκή Κοινότητα, οργανισμό που συμμετέχουν όλα τα Πανεπιστήμια και τα ΤΕΙ της χώρας και είναι στη διάθεση όλων των μελών της (τα Ελληνικά Πανεπιστήμια και ΤΕΙ). Σκοπός του είναι να προσφέρει ουσιαστικές πληροφορίες για τις σπουδές των φοιτητών έχοντας άμεση πρόσβαση στα επίσημα συστήματα των εκάστοτε ιδρυμάτων. Οι φοιτητές μπορούν μέσα από την συγκεκριμένη εφαρμογή να ελέγξουν τους βαθμούς τους, να βρούνε τα μαθήματα τα οποία έχουνε δηλώσει στο τρέχον εξάμηνο και το ιστορικό των δηλώσεων τους, αλλά και να δούνε τα μαθήματα που περιλαμβάνονται στο πρόγραμμα σπουδών τους. Για να είναι εφικτή η σύνδεση των συστημάτων των ιδρυμάτων με το Uniway απαιτείται η υλοποίηση από πλευράς του ιδρύματος ενός συγκεκριμένου τρόπου επικοινωνίας (γνωστού ως Application Programming Interface) που επιτρέπει την επικοινωνία απευθείας από υπολογιστή σε υπολογιστή. Για το λόγο αυτό, η Ελληνική Ακαδημαϊκή Κοινότητα έχει δημοσιεύσει τον οδηγό σύνδεσης των ιδρυμάτων με το Uniway, ο οποίος περιέχει πληροφορίες για τις κλήσεις της εφαρμογής αλλά και τη μορφή των δεδομένων που θα επιστρέφονται μέσω αυτών.

Με βασικό εργαλείο το Spring Framework έχουμε υλοποιήσει όλες τις κλήσεις της εφαρμογής οι οποίες αναφέρονται στον οδηγό σύνδεσης των ιδρυμάτων. Χρησιμοποιούμε το Spring Framework γιατί είναι ένα ολοκληρωμένο περιβάλλον ανάπτυξης enterprise εφαρμογών με αρκετές δυνατότητες. Ακόμα και σήμερα, που έχουν αναπτυχθεί αρκετά πλαίσια ανάπτυξης κώδικα στην Java, το Spring συνεχίζει να υπερτερεί σε χαρακτηριστικά όπως είναι η γρήγορη ανάπτυξη, ο οργανωμένος κώδικας , η δυνατότητα επαναχρησιμοποίησης κώδικα, η απόδοση και η ενσωμάτωση του σε τρίτα συστήματα.

Το τελευταίο χαρακτηριστικό είναι που το έχει ξεχωρίσει από τα υπόλοιπα πλαίσια. Μας δίνει τη δυνατότητα να αξιοποιήσουμε άλλα πλαίσια ανάπτυξης και να δημιουργήσουμε μία στιβαρή και αξιόλογη εφαρμογή. Με αφορμή αυτό το προτέρημα έχουμε αξιοποιήσει στη συγκεκριμένη υλοποίηση το Spring Data JPA, με σκοπό να διαχειριζόμαστε πιο αποτελεσματικά την επικοινωνία και την ανταλλαγή των δεδομένων μεταξύ της βάσης και της εφαρμογής

Τέλος αξίζει να αναφέρουμε την προσπάθεια της Ελληνικής Ακαδημαϊκής Κοινότητας να προσφέρει κοινές λύσεις και καινοτομίες (Uniway) σε όλα τα μέλη της. Τα περισσότερα ιδρύματα στην χώρα μας έχουνε υλοποιήσει τα δικά τους συστήματα για την ενημέρωση των φοιτητών τους τα οποία λειτουργούν ανεξάρτητα μεταξύ τους. Αντίθετα το Uniway είναι μία εφαρμογή ενημέρωσης που θα τη χρησιμοποιούν όλα τα ιδρύματα της χώρας. Ωστόσο η υλοποίηση του απαιτεί χρόνο και αρκετή προσπάθεια από το κάθε ίδρυμα, ενώ ενδέχεται να υπάρξουν κάποια ιδρύματα που δεν θα μπορέσουν να συνδεθούν για διάφορους τεχνικούς λόγους. Για να μπορέσουν τα εξαλειφτούν αυτές οι δυσκολίες θα ήταν καλύτερα η Ελληνική Ακαδημαϊκή Κοινότητα να υλοποιήσει μια πιο ολοκληρωμένη εφαρμογή , χωρίς να περιπλέκει τρίτους και στην προκειμένη περίπτωση τα ακαδημαϊκά ιδρύματα. Τα ιδρύματα με τη σειρά τους θα χρειάζονται αρκετό χρόνο για να υλοποιήσουν τον οδηγό διασύνδεσης, χρησιμοποιώντας τις τεχνολογίες που γνωρίζουν καλύτερα και τα δικά τους μηχανήματα με αποτέλεσμα η εφαρμογή να μην έχει συνοχή και να υπάρχουν αυξομειώσεις στην απόδοση μεταξύ των ιδρυμάτων.

ΒΙΒΛΙΟΓΡΑΦΙΑ

[1] Βικιπαίδεια. (2018, August 19). JSON. Retrieved October 21, 2018, from https://el.wikipedia.org/w/index.php?title=JSON&oldid=7178831

[2] Chandrashekhar. (2015, September 27). Advantages of Spring Framework. Retrieved October 29, 2018, from https://www.onlinetutorialspoint.com/spring/advantages-of-spring-framework.html

[3] GUnet – Greek Universities Network. (n.d.). Retrieved February 26, 2019, from https://www.gunet.gr/

[4] GUnet Uniway SIS API 1-0-3.pdf. (n.d.). Retrieved from https://community.gunet.gr/sites/default/files/GUnet%20Uniway%20SIS%20API%201-0-3.pdf

[5] Gupta, K. (2018, April 26). Why is Spring More Popular Than Other Java Frameworks? Retrieved October 29, 2018, from https://www.freelancinggig.com/blog/2018/04/26/spring-popular-java-frameworks

[6] Nikola lolev,(2017) The pros and cons of using a framework. (n.d.). Retrieved November 17, 2018, from https://jaxenter.com/pros-cons-using-framework-135901.html

[7] JPA Introduction. (n.d.). Retrieved February 25, 2018, from https://www.tutorialspoint.com/jpa/jpa_introduction.htm

[8] JPA - Architecture. (n.d.). Retrieved February 25, 2018, from https://www.tutorialspoint.com/jpa/jpa_architecture.htm

[9] JPA - ORM Components (n.d.). Retrieved February 25, 2018, from https://www.tutorialspoint.com/jpa/jpa_orm_components.htm

[10] JPA - Entity Managers (n.d.). Retrieved February 25, 2018, from https://www.tutorialspoint.com/jpa/jpa_entity_managers.htm

[11] JPA - Quick Guide (n.d.). Retrieved February 25, 2018, from https://www.tutorialspoint.com/jpa/jpa_quick_guide.htm

[12] JPA - Entity Relationships (n.d.). Retrieved February 25, 2018, from https://www.tutorialspoint.com/jpa/jpa_entity_relationships.htm

[13] JPA - JPQL (n.d.). Retrieved February 25, 2018, from https://www.tutorialspoint.com/jpa/jpa_jpql.htm

[14] JSON: The Fat-Free Alternative to XML (n.d.). Retrieved February 17, 2018, from https://www.json.org/xml.html

[15] Spring Data JPA Tutorial: Introduction. (n.d.). Retrieved October 29, 2018, from https://www.petrikainulainen.net/programming/spring-framework/spring-data-jpa-tutorial-introduction/

[16] Spring framework advantages and disadvantages. (n.d.). Retrieved October 29, 2018, from http://r4r.co.in/java/spring/spring_tutorial/advantages_and_disadvantages.shtml

[17] Spring projects. (n.d.). Retrieved October 20, 2018, from https://spring.io/projects/spring-data-jpa#overview

[18] Spring vs Struts - Find Out The 6 Important Differences. (2018, October 11). Retrieved October 29, 2018, from https://www.educba.com/spring-vs-struts/

[19] Struts Framework. (n.d.). Retrieved November 17, 2018, from https://tutorialseye.com/struts-framework.html

[20] What is an API? (Application Programming Interface). (2016, December 7). Retrieved October 29, 2018,

from https://www.mulesoft.com/resources/api/what-is-an-api

[21] What is Struts. (n.d.). Retrieved November 17, 2018,

from https://tutorialseye.com/what-is-struts.html

.