Domain-Model-v1.0

Όνομα Έργου: Train-Up

Κωδικός: Domain-Model

Έκδοση: v1.0

Μέλη Ομάδας

- Σκαραφίγκας Βασίλειος, ΑΜ: 4491.
- Χριστόπουλος Κωνσταντίνος, ΑΜ: 4527.

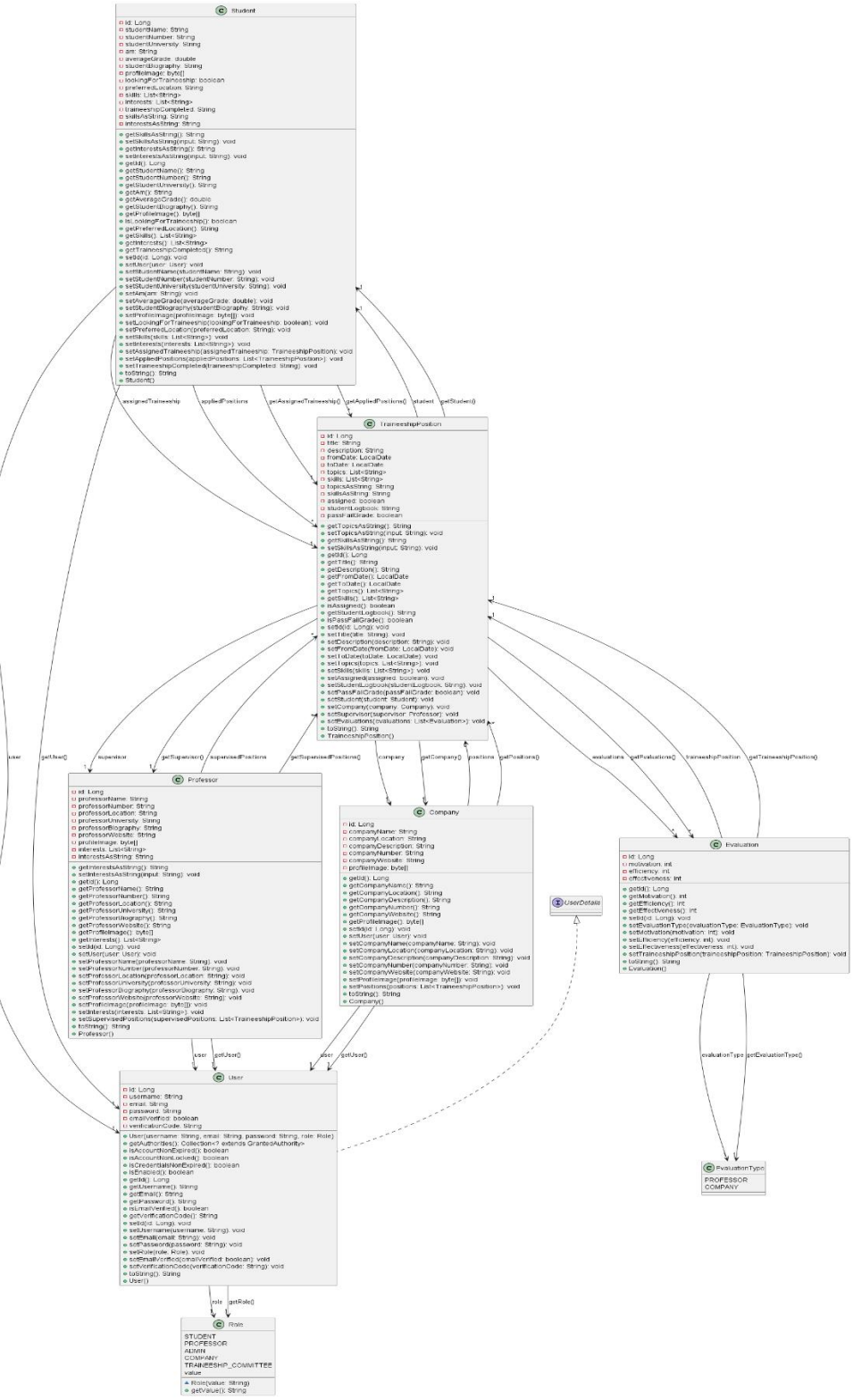
Κατανομή Ρόλων στο παρών τεχνικό κείμενο

- Σκαραφίγκας Βασίλειος: Contributor, Peer Reviewer
- Χριστόπουλος Κωνσταντίνος: Contributor, Peer Reviewer

Στο παρόν τεχνικό κείμενο όλα τα μέλη της ομάδας συνεισέφεραν εξίσου στην περιγραφή των κλάσεων του έργου.

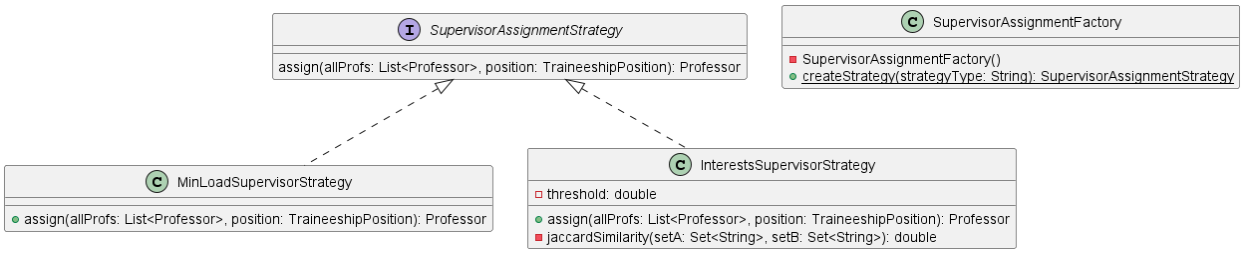
Ο σύνδεσμος για το repository της ομάδας μας στο Github είναι εδώ.

Διάγραμμα Domain Model

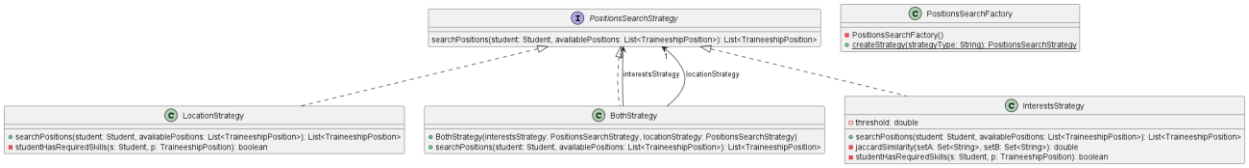


Εικόνα 1 Domain Model Diagram

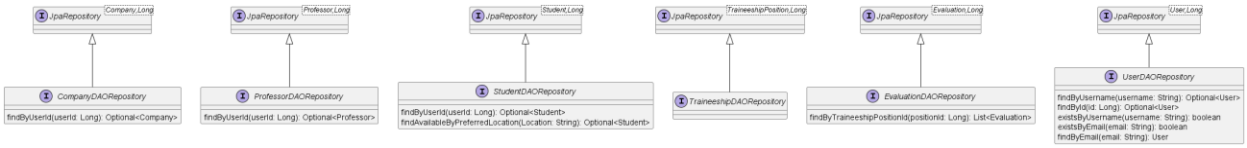
Στο σχήμα της εικόνας 1 παρουσιάζουμε το διάγραμμα κλάσεων του Domain Model. Στην παρούσα έκδοση του συγκεκριμένου τεχνικού κειμένου και στην εικόνα 2 την λογική της στρατηγικής αναζήτησης καθηγητή και στην εικόνα 3 την στρατηγική αναζήτηση των θέσεων πρακτικής άσκησης. Το Model (εικόνα 1) είναι περιγραφικά το σχήμα της βάσης δεδομένων και έχει με μεθόδους accessor.



Εικόνα 2 Διάγραμμα Στρατηγικής για επιλογή καθηγητή



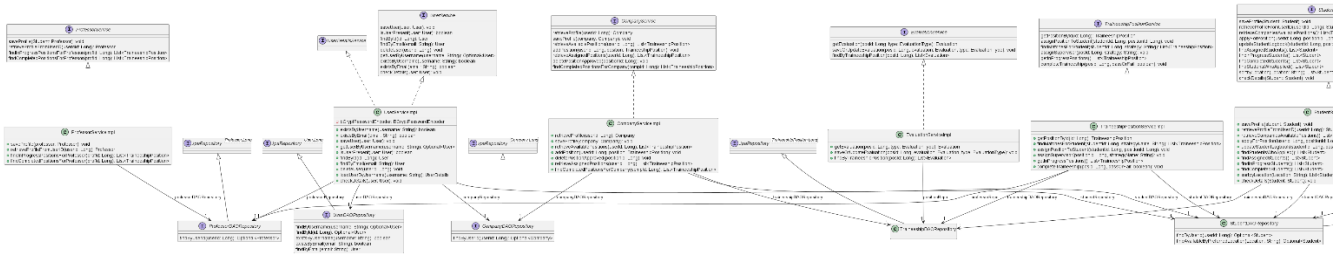
Εικόνα 3 Διάγραμμα Στρατηγικής για εύρεση θέσης



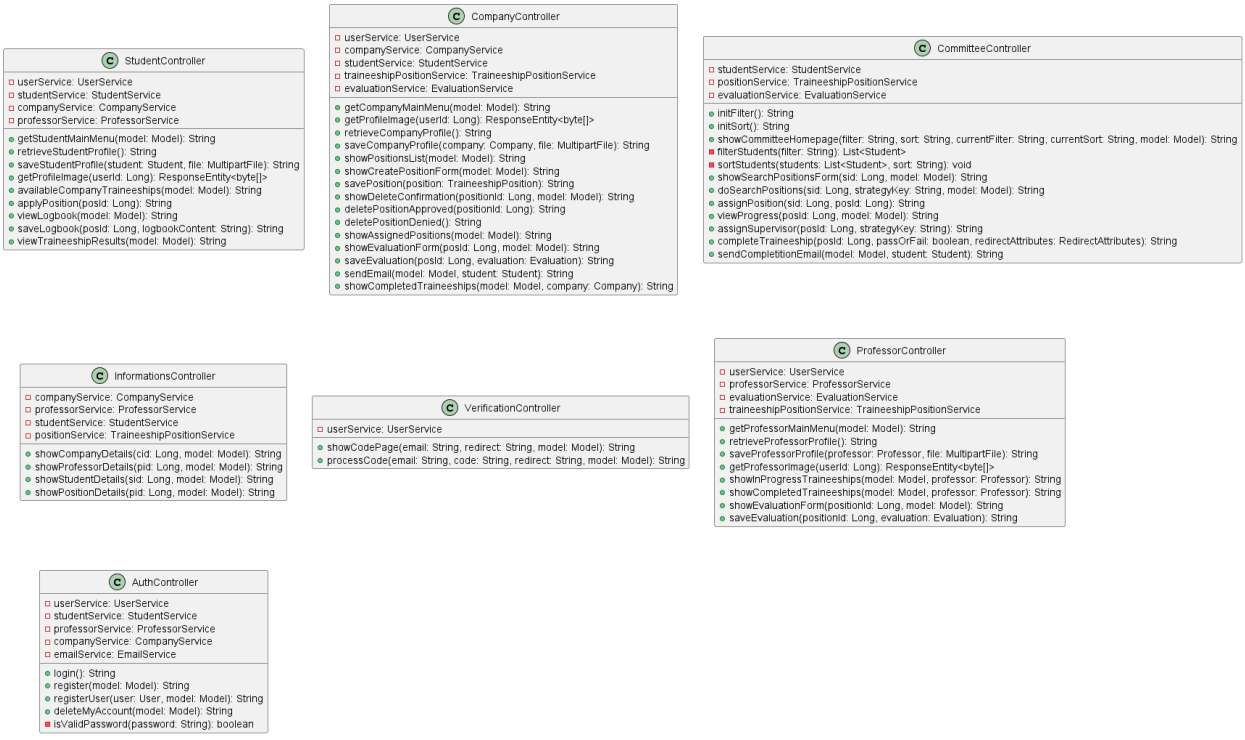
Εικόνα 4 Διάγραμμα DAO

Εικόνα 5 Διάγραμμα Service

Η εικόνα 6 όπως είχαμε αναφέρει στην προηγούμενη έκδοση δείχνει την χρήση του dao με τα strategies. Επειδή το διάγραμμα αν προσθέταμε κι το domain θα ήταν μεγάλο όμως θεωρητικά κάθε κλάση του domain συνδέεται στο κατάλληλο service που χρειάζεται την κλάση αυτή σαν αντικείμενο.

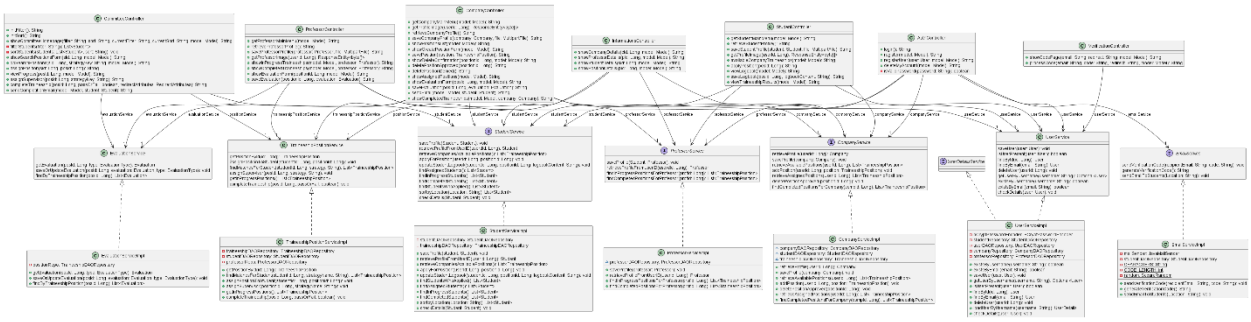


Εικόνα 6 Διάγραμμα Service με dao και strategies



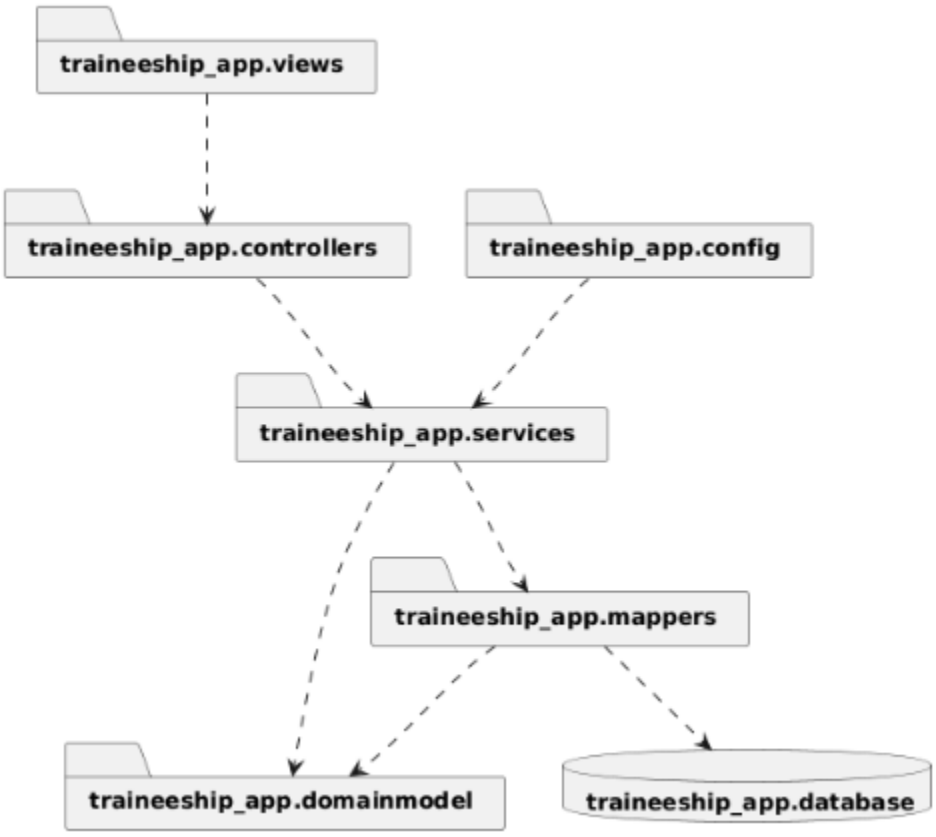
Εικόνα 7 Διάγραμμα Controller

Στην εικόνα 8 φαίνεται η σύνδεση του controller με το service. Θεωρητικά το μοντέλο είναι μεγαλύτερο καθώς θεωρούμε ως service την εικόνα 6.



Εικόνα 8 Διάγραμμα Controller με service

Στην εικόνα 9 φαίνεται η αρχιτεκτονική του project με όλα τα πακέτα κλάσεων με την λογική τους σύνδεση.



Εικόνα 9 Διάγραμμα Controller με service

Περιγραφή κλάσεων

Στην παρακάτω περιγραφή των κλάσεων στοχεύουμε να φανούν οι βασικές υποψήφιες κλάσεις. Καθώς θα προχωρήσει η ανάλυση και η υλοποίηση, ενδέχεται να προστεθούν ή να αφαιρεθούν κλάσεις ή να γίνουν πιο εξιδεικευμένες οι σχέσεις.

A. Βασικό Μοντέλο (Domain Model):

- **User**

Η βασική οντότητα που αναπαριστά κάθε εγγεγραμμένο χρήστη της πλατφόρμας.

- **Πεδία**: id, username, email, password, emailVerified, verificationCode.
- **Σχέσεις**:
  - 1–1 με Role (για τον ρόλο: STUDENT, PROFESSOR, COMPANY, ADMIN, TRAINEESHIP_COMMITTEE).
  - 1–1 με Student ή Professor ή Company μέσω των associations “user” (καθώς κάθε ειδικός τύπος χρήστη έχει και εγγραφή σε User).
- **Ρόλος στα Use Cases**: Κεντρικός κόμβος για σύνδεση/εγγραφή, έλεγχο δικαιωμάτων και πλοήγηση ανάλογα με τον ρόλο.

- **Role**

Enum που καθορίζει τον τύπο/εξουσιοδότηση ενός User.

- **Τιμές**:
  - STUDENT
  - PROFESSOR
  - COMPANY
  - ADMIN
  - TRAINEESHIP_COMMITTEE
- **Χρήση**: Ελέχει ποια μενού και ποιες λειτουργίες είναι προσβάσιμες σε κάθε χρήστη.

- **Student**

Εξειδίκευση του User για φοιτητές.

- **Πεδία**: studentName, studentNumber, studentUniversity, am, averageGrade, studentBiography, profileImage, lookingForTraineeship, preferredLocation, skills, interests, traineeshipCompleted, κ.ά.
- **Σχέσεις**:
  - 1–N με TraineeshipPosition (μέσω appliedPositions) — οι θέσεις που υπέβαλε αίτηση.
  - 1–1 με TraineeshipPosition (μέσω assignedTraineeship) — η θέση που του ανατέθηκε.
  - 1–N με Evaluation (μέσω evaluations) — οι αξιολογήσεις που έλαβε.
- **Ρόλος στα Use Cases**: Υποβολή αιτήσεων, ενημέρωση logbook, προβολή αποτελεσμάτων και προφίλ.

- **Profesor**

Εξειδίκευση του User για καθηγητές-επόπτες.

- **Πεδία**: professorName, professorNumber, professorLocation, professorUniversity, professorBiography, profileImage, interests.
- **Σχέσεις**:
  - 1–N με TraineeshipPosition (μέσω supervisedPositions) — οι θέσεις που επιβλέπει.
  - 1–N με Evaluation (μέσω evaluations) — οι αξιολογήσεις που έχει κάνει.
- **Ρόλος στα Use Cases**: Αξιολόγηση φοιτητών, προβολή θέσεων υπό επίβλεψη και ενημέρωση προφίλ.

- **Company**

Εξειδίκευση του User για εταιρείες.

- **Πεδία**: companyName, companyLocation, companyDescription, companyNumber, companyWebsite, profileImage.
- **Σχέσεις**:
  - 1–N με TraineeshipPosition (μέσω positions) — οι θέσεις πρακτικής που έχει αναρτήσει.
- **Ρόλος στα Use Cases**: Ανάρτηση, τροποποίηση και διαγραφή θέσεων πρακτικής, προβολή προφίλ εταιρείας.

- **TraineeshipPosition**

Αντικείμενο που αναπαριστά μια διαθέσιμη ή ανατεθειμένη θέση πρακτικής.

- **Πεδία**: title, description, fromDate, toDate, topics, skills, assigned (flag), studentLogbook, passFailGrade, κ.ά.
- **Σχέσεις**:
  - N–1 με Company (η εταιρεία που την προσφέρει).
  - N–1 με Student (μέσω assignedTraineeship) — ο φοιτητής που την έχει αναλάβει.
  - N–1 με Professor (μέσω supervisor) — ο επιβλέπων καθηγητής.
  - 1–N με Evaluation (μέσω evaluations) — οι αξιολογήσεις εταιρείας και καθηγητή.
- **Ρόλος στα Use Cases**: Κέντρο όλων των αλληλεπιδράσεων: αναζήτηση-φιλτράρισμα (strategies), υποβολή αιτήσεων, ανάθεση, ενημέρωση logbook, αξιολογήσεις.

- **Evaluation**

Καταγράφει το αποτέλεσμα μιας αξιολόγησης φοιτητή σε θέση πρακτικής.

- **Πεδία**: motivation, efficiency, effectiveness.
- **Σχέσεις**:
    - N–1 με TraineeshipPosition (σε ποια θέση αναφέρεται).
    - N–1 με EvaluationType (προέλευση: FROM_PROFESSOR ή FROM_COMPANY).
- **Ρόλος στα Use Cases**: Αποθήκευση και προβολή βαθμολογιών από καθηγητή ή εταιρεία, απαραίτητο για την τελική "finalize traineeship".

➢ **EvaluationType**

Enum με τις δύο πηγές αξιολόγησης:

- FROM_PROFESSOR
- FROM_COMPANY
  Χρησιμοποιείται για να ξεχωρίζουν οι αξιολογήσεις κατά προβολή και λογική ολοκλήρωσης.

Β. DAO:

Το πακέτο dao αποτελείται από διάφορους ορισμούς διεπαφής που προέρχονται από το γενικό Interface JpaRepository που παρέχεται από το Spring Boot. Αυτές οι διεπαφές μας επιτρέπουν να εκτελούμε βασικές λειτουργίες βάσης δεδομένων που αντιστοιχίζουν τα δεδομένα της εφαρμογής που είναι αποθηκευμένα στη βάση δεδομένων στη μνήμη αντικείμενα των κλάσεων μοντέλου τομέα. Βασικά, εδώ εφαρμόζουμε το μοτίβο Data Mapper από Martin Fowler's catalog of EAA patterns. Για κάθε οντότητα του domain έχουμε κι από μια dao κλάση για την ανάκτηση δεδομένων από την βάση.

Γ. Services:

Τα service χρησιμοποιούν τις μεθόδους του DAO για να τραβάνε τα δεδομένα από την βάση σε συνδυασμό με τις κατάλληλες οντότητες και είναι συνενωμένοι με τους controllers που τα χρησιμοποιούν.

- **UserService**: Κλάση που παρέχει λειτουργίες για εγγραφή (signup), σύνδεση (login) και έξοδο (logout) των οντοτήτων **User**.
- **CompanyService**: Κλάση που είναι υπεύθυνη για τη διαχείριση του προφίλ των εταιρειών, τη δημιουργία νέων θέσεων πρακτικής, την προβολή των διαθέσιμων και των ανατεθειμένων θέσεων, καθώς και την αξιολόγηση φοιτητών που εργάζονται σε αυτές τις θέσεις.
- **StudentService**: Κλάση που παρέχει λειτουργίες για τη δημιουργία και ενημέρωση του προφίλ ενός φοιτητή, την αναζήτηση/εμφάνιση διαθέσιμων θέσεων, την υποβολή αίτησης (apply) σε θέση πρακτικής, καθώς και την ενημέρωση ενός logbook (ημερολογίου πρακτικής).
- **ProfessorService**: Κλάση που είναι υπεύθυνη για τη δημιουργία/ενημέρωση του προφίλ καθηγητή και την αξιολόγηση των φοιτητών σε θέσεις που επιβλέπει.
- **CommitteeService**: Κλάση που αφορά την επιτροπή πρακτικής. Παρέχει λειτουργίες για την ανάθεση θέσεων πρακτικής σε φοιτητές (βάσει διαφορετικών κριτηρίων), την ανάθεση καθηγητών ως επιβλεπόντων (επίσης με εναλλακτικές στρατηγικές) και, τέλος, την ολοκλήρωση ή ακύρωση μιας θέσης πρακτικής (pass/fail).
- **EmailService**: Κλάση που αφορά τις ενημερώσεις με email, είτε για υπενθυμίσεις καταχώρησης βαθμολογιών, για επιβεβαίωση λογαριασμού και για ειδοποίηση φοιτητών για την τελική τους βαθμολογία και όπου αλλού κριθεί κατάλληλο.
- **TraineeshipPositionService:** Κλάση υπεύθυνη για τη διαχείριση των θέσεων πρακτικής. Παρέχει λειτουργίες για την εύρεση διαθέσιμων θέσεων, την ανάθεση θέσης σε φοιτητή με ταυτόχρονη ενημέρωση της κατάστασής του, την ανάθεση επιβλέποντα καθηγητή βάσει επιλεγμένης στρατηγικής, καθώς και την παρακολούθηση της προόδου των θέσεων (π.χ. αν είναι σε εξέλιξη). Περιλαμβάνει επίσης τη διαδικασία ολοκλήρωσης μιας θέσης πρακτικής, η οποία προϋποθέτει την ύπαρξη αξιολογήσεων και από την εταιρεία και από τον επιβλέποντα καθηγητή. Η κλάση βασίζεται στη χρήση στρατηγικών (pattern Strategy) για ευέλικτη επιλογή θέσεων και καθηγητών.
- **EvaluationService**: Κλάση που διαχειρίζεται τις αξιολογήσεις θέσεων πρακτικής. Παρέχει λειτουργίες για την αναζήτηση, δημιουργία ή ενημέρωση αξιολογήσεων ανά τύπο (επιβλέπων ή εταιρεία), καθώς και για την προβολή όλων των αξιολογήσεων που έχουν καταχωρηθεί για μια συγκεκριμένη θέση. Εξασφαλίζει ότι κάθε θέση μπορεί να έχει έως και μία αξιολόγηση από κάθε αξιολογητή και υποστηρίζει την ενσωμάτωση των αποτελεσμάτων αξιολόγησης στην τελική κρίση της πρακτικής.

Δ. Στρατιγική Ανάλυση (strategies)

Η ανάθεση των θέσεων στους μαθητές πραγματοποιείται μέσω εναλλακτικών στρατηγικών που λαμβάνουν υπόψη τις ενδιαφέροντα των μαθητών, την προτιμώμενη θέση ή και τα δύο. Για να γίνουν οι εναλλακτικές στρατηγικές εναλλάξιμες και να διευκολύνουμε την επέκταση της εφαρμογής με περισσότερες στρατηγικές, βασιζόμαστε στο πρότυπο στρατηγικής GoF, δηλ. οι στρατηγικές ανάθεσης θέσης υλοποιούνται σε διαφορετικές κλάσεις που παρέχουν την ίδια διεπαφή. Η υπηρεσία επιτροπής δημιουργεί την κατάλληλη στρατηγική χρησιμοποιώντας ένα παραμετροποιημένο "factory". Η ανάθεση καθηγητών σε θέσεις πρακτικής άσκησης υλοποιείται επίσης μέσω εναλλακτικών στρατηγικών που λαμβάνουν υπόψη τα ενδιαφέροντα ή τον φόρτο εργασίας των καθηγητών (Εικόνα 2). Όπως και προηγουμένως, για να γίνουν οι εναλλακτικές στρατηγικές εναλλάξιμες και να διευκολύνουμε την επέκταση της εφαρμογής με περισσότερες στρατηγικές, βασιζόμαστε στο GoF μοτίβο στρατηγικών, δηλαδή οι στρατηγικές ανάθεσης καθηκόντων επόπτη υλοποιούνται σε διαφορετικές κλάσεις που παρέχουν την ίδια διεπαφή.

Συγκεκριμένα:

1. *Στρατηγική αναζήτησης πρακτικής θέσης από φοιτητή:*

- **PositionsSearchStrategy** (interface): Ορίζει τη μέθοδο searchPositions(...) για φιλτράρισμα λίστας διαθέσιμων θέσεων ανά φοιτητή.
- **InterestsStrategy**: Υλοποιεί PositionsSearchStrategy φιλτράροντας θέσεις με βάση το “similarity” ανάμεσα στα topics της θέσης και τα interests του φοιτητή. Ο παράγων threshold καθορίζει το ελάχιστο όριο ομοιότητας.
- **LocationStrategy**: Υλοποιεί PositionsSearchStrategy φιλτράροντας θέσεις με βάση τη γεωγραφική τοποθεσία της εταιρείας σε σχέση με την preferredLocation του φοιτητή.
- **BothStrategy**: Υλοποιεί PositionsSearchStrategy συνδυάζοντας δύο άλλες στρατηγικές (Interests + Location), επιστρέφοντας μόνο τις θέσεις που πληρούν και τα δύο κριτήρια (τομή αποτελεσμάτων).
- **PositionsSearchFactory**: Στατικό “εργοστάσιο”: αναλαμβάνει, βάσει ενός αλφαριθμητικού strategyType, να επιστρέψει την κατάλληλη υλοποίηση PositionsSearchStrategy (INTERESTS, LOCATION ή BOTH).

2. Στρατηγική αναζήτησης επιβλέποντα καθηγητή σε κάποια θέση από τον «committee»

- **SupervisorAssignmentStrategy(interface)**: Ορίζει τη μέθοδο assign(...) που επιλέγει καθηγητή για μια θέση πρακτικής.

- **InterestsSupervisorStrategy**: Υλοποιεί SupervisorAssignmentStrategy επιλέγοντας τον καθηγητή με τη μεγαλύτερη similarity ανάμεσα στα interests του καθηγητή και στα topics της θέσης, πάνω από ένα όριο threshold.
- **MinLoadSupervisorStrategy**: Υλοποιεί SupervisorAssignmentStrategy αναθέτοντας τον καθηγητή με τον ελάχιστο τρέχοντα φόρτο (μικρότερος αριθμός ήδη επιβλεπόμενων θέσεων).
- **SupervisorAssignmentFactory**: Στατικό "εργοστάσιο": δημιουργεί την κατάλληλη SupervisorAssignmentStrategy (π.χ. INTERESTS ή MIN_LOAD) με βάση παράμετρο strategyType, κρύβοντας την λογική επιλογής από τον καλούντα.

E. Controller

Το πακέτο *MVC controllers* αποτελείται από κλάσεις ελεγκτών, οι οποίες λαμβάνουν την είσοδο του χρήστη από τις *views* της εφαρμογής, εκτελούν συγκεκριμένες λειτουργίες υπηρεσιών (βλ. επόμενο σημείο) για να τροποποιήσουν αντικείμενα του μοντέλου τομέα, και ενημερώνουν τις *views* της εφαρμογής.

Το πακέτο ελεγκτών περιλαμβάνει επτά διαφορετικούς ελεγκτές (Εικόνα 7). Ο AuthController και VerificationController είναι υπεύθυνος για την εγγραφή, ταυτοποίηση και την είσοδο των χρηστών. Οι CompanyController, StudentController ProfessorController, CommitteeController είναι υπεύθυνοι για την αλληλεπίδραση με εταιρείες, φοιτητές, καθηγητές και μέλη της επιτροπής, αντίστοιχα. Ο InformationsController είναι υπεύθυνος για την ανάκτηση και προβολή στοιχείων εταιρίας , καθηγητή, φοιτητή και θέσης αλλά για τους άλλους χρήστες που θέλουν να δουν κάποια πληροφορία πιο αναλυτικά. Οι ελεγκτές βασίζονται στις αντίστοιχες υπηρεσίες των services πακέτων για τον χειρισμό των αντικειμένων των domain αντικειμένων και την ενημέρωση των προβολών. Οι προβολές βασίζονται τη μηχανή προτύπων Thymeleaf με προβολή αρχείων html για τα παράθυρα στο πακέτο view.

F. Views:

Το πακέτο *views* υλοποιεί την αλληλεπίδραση του χρήστη (UI) με την εφαρμογή σε συνεργασία με τη στρώση του *Controller*. Συνήθως, αυτή η στρώση αποτελείται από ένα σύνολο στατικών και δυναμικών ιστοσελίδων. Οι δυναμικές ιστοσελίδες είναι επίσης γνωστές ως *template views* (βλ. πρότυπο Template View από τον κατάλογο EAA patterns

του Martin Fowler). Μια δυναμική HTML σελίδα αποδίδει δεδομένα από αντικείμενα του μοντέλου τομέα σε HTML, βάσει ενσωματωμένων δεικτών. Οι ελεγκτές της εφαρμογής είναι υπεύθυνοι για τη μεταφορά των κατάλληλων αντικειμένων του μοντέλου τομέα στη στρώση *view*.

Εργαλεία που χρησιμοποιήθηκαν

Η συγγραφή του παρόντος τεχνικού κειμένου έγινε με την χρήση του Microsoft Word. Τα διαγράμματα υλοποιήθηκαν με τη χρήση του εργαλείου plant-uml στο περιβάλλον του eclipse.