Practice Portal

Sprint Report

Christos Dimitresis cs04351@uoi.gr

VERSIONS HISTORY

Date	Version	Description	Author
28-04-2025	0.0	Project initialization, Spring Boot setup, and Git repository configuration	Christos Dimitresis
03-05-2025	0.1	Implementation of core entities (Student, Company, Professor, Internship)	Christos Dimitresis
07-05-2025	0.2	Created repositories, DTOs, and basic Thymeleaf templates	Christos Dimitresis
10-05-2025	0.3	Implemented login/registration and role-based access with Spring Security	Christos Dimitresis
13-05-2025	0.4	Developed controllers for students, companies, and professors	Christos Dimitresis
16-05-2025	0.5	Added internship position management and company associations	Christos Dimitresis
20-05-2025	0.6	Implemented matching strategies for assignments and professors (interests, location, combined)	Christos Dimitresis
23-05-2025	0.7	Added student evaluation by companies and professors	Christos Dimitresis
26-05-2025	1.0	Final testing, controller tests, bug fixing, and project submission	Christos Dimitresis

1 Scrum team and Sprint Backlog

1.1 Scrum team

Product Owner	Christos Dimitresis
Scrum Master	Christos Dimitresis
Development Team	Christos Dimitresis

1.2 Sprints

Sprint No	Begin Date	End Date	Number of weeks	User stories
Sprint 1	28/04/2025	26/05/2025	4	US1 – US21

Since the project was developed individually, the Scrum process was followed in a simplified way.

All functionalities were implemented incrementally as part of a single continuous sprint.

The entire scope of the application, including all user stories (US1–US21), was covered during this sprint.

Although formal Sprint Planning, Daily Scrum, and Sprint Review meetings did not take place, the development followed the Agile mindset of progressive feature integration, continuous testing, and iterative refinement based on observed needs. :)

2 Use Cases

Use Case 1 : User Registration

Use case ID	UC1
Actors	Unauthenticated user
Pre conditions	- The user has not registered before. - The user has access to the main entry page (`index.html`) and the registration form.
Main flow of events	1. The use case starts when the user visits the main page (`index.html`). 2. The user selects to register and is redirected to `user_register.html`. 3. The user selects the desired role (student, company, or professor). 4. The system redirects the user to the corresponding registration form: 4.1. `student_registration.html` 4.2. `company_registration.html` 4.3. `professor_registration.html` 5. The use case ends here. The rest of the process is handled in role-specific use cases.
Alternative flow 1	If the submitted data contains errors (e.g., missing required fields, invalid email), the system displays validation error messages and remains on the same registration form.
Alternative flow 2	-
Post conditions	- The user has selected their role and has been redirected to the role-specific registration form.

Use Case 2: User Login

Use case ID	UC2
Actors	Unauthenticated user
Pre conditions	- The user already has a valid account in the system. - The login form is accessible from the main page (index.html).
Main flow of events	1. The use case starts when the user visits the main page (index.html). 2. The user enters their email and password into the login form. 3. The user clicks the "Login" button. 4. The system authenticates the user's credentials using Spring Security. 5. If authentication succeeds: 5.1. The user is redirected to their corresponding dashboard based on role (Student, Company, Professor, Practice Office).
Alternative flow 1	If the user submits invalid credentials (e.g., wrong password), the system displays an error message and remains on the login page.
Alternative flow 2	If the user submits empty fields or invalid email format, the system displays validation errors and does not attempt authentication.
Post conditions	- The user is authenticated and redirected to their role-specific dashboard. - A session is created to maintain the login state.

Use Case 3: User Logout

Use case ID	UC3
Actors	Authenticated user (Student, Company, Professor, or Practice Office member)
Pre conditions	- The user is currently logged in. - The user has access to the application interface that provides a logout option.
Main flow of events	1. The use case starts when the user clicks on the "Logout" option available on its dashboard. 2. The system terminates the current user session using Spring Security. 3. The user is redirected to the "logout_success.html" page. 4. A confirmation message is displayed, indicating successful logout.
Alternative flow 1	If the session is already invalid or expired, the system redirects the user to the login page.
Alternative flow 2	If the user tries to logout without being authenticated, the logout request is ignored and the user is redirected to the main page or login page.
Post conditions	- The user's session is invalidated. - The user is logged out of the system and cannot access authenticated pages unless they log in again.

Use Case 4: Student Registration Profile Creation

Use case ID	UC4
Actors	Unauthenticated user (who selects role = Student)
Pre conditions	- The user has selected "Student" during initial role selection in Use Case UC1. - The student registration form (`student_registration.html`) is accessible.
Main flow of events	1. The use case starts when the user has been redirected to `student_registration.html` after selecting "Student" in UC1. 2. The system redirects the user to `student_registration.html`. 3. The user fills in their full name, university ID, list of interests, list of skills, and preferred location. 4. The user submits the form. 5. The system validates the input. 6. If all data is valid, the student account is created in the database with the provided profile details. 7. The user is redirected to a success page (`register_success.html`).
Alternative flow 1	If required fields are missing or invalid, the system displays error messages and remains on the registration form.
Alternative flow 2	-
Post conditions	- A new student account is created with a complete profile. - The student can log in and access their dashboard.

Use Case 5: Implicit Application for Traineeship

Use case ID	UC5
Actors	Student (after successful registration)
Pre conditions	- The user has completed the registration process as a Student (UC1 and UC4). - The student account is stored with full profile data in the system.
Main flow of events	1. The use case starts when a student successfully completes their registration. 2. The system considers the registered student as a candidate for traineeship matching. 3. No explicit application action is required by the student. 4. The student will be considered in the next round of automatic traineeship assignment by the practice office.
Alternative flow 1	-
Alternative flow 2	-
Post conditions	- The student is marked as a valid applicant and is eligible to be matched to available traineeship positions.

Use Case 6: Fill in Internship Evaluation (Logbook)

Use case ID	UC6
Actors	Student (with an assigned traineeship)
Pre conditions	- The student is logged in. - A traineeship has been assigned to the student. - The student is on the dashboard page (`dashboard.html`), where the assignment and evaluation option is visible.
Main flow of events	1. The use case starts when the student visits their dashboard (`dashboard.html`). 2. The student sees the assigned traineeship information and clicks the "Evaluate" button. 3. The system redirects the student to the evaluation page (`evaluation.html`). 4. The student fills in the evaluation form describing their traineeship experience. 5. The student submits the form. 6. The system validates the input and stores or overrides the previous evaluation (if any). 7. The student is redirected back to the dashboard page.
Alternative flow 1	If the student submits incomplete or invalid evaluation data, the system shows error messages and remains on the evaluation page.
Alternative flow 2	-
Post conditions	- The student’s evaluation is stored or updated in the system.

Use Case 7: Company Registration Profile Creation

Use case ID	UC7
Actors	Unauthenticated user (who selects role = Company)
Pre conditions	- The user has selected "Company" during initial role selection in Use Case UC1. - The company registration form (`company_registration.html`) is accessible.
Main flow of events	1. The use case starts when the user is redirected to `company_registration.html` after choosing "Company" in UC1. 2. The user fills in the company name and location, along with other required fields (e.g., contact person, email). 3. The user submits the form. 4. The system validates the input. 5. If all data is valid, the company account is created and stored in the database. 6. The user is redirected to `register_success.html`.
Alternative flow 1	If required fields are missing or invalid, the system displays error messages and remains on the registration form.
Alternative flow 2	-
Post conditions	- A new company account is created with a complete profile. - The company can log in and access its dashboard.

Use Case 8: View Company's Posted Traineeship Positions

Use case ID	UC8
Actors	Company (authenticated user)
Pre conditions	- The company is logged in successfully. - The company has posted at least one traineeship position. - The company is redirected to their dashboard page (`dashboard.html`).
Main flow of events	1. The use case starts when the company logs into the application. 2. The system redirects the company to their dashboard (`dashboard.html`). 3. The system retrieves and displays the list of traineeship positions posted by the company. 4. Each position is shown with its title, description, and other metadata.
Alternative flow 1	If the company has not posted any positions yet, the dashboard displays an informative message.
Alternative flow 2	-
Post conditions	- The company can view all its posted traineeship positions. - The company may proceed to manage (edit/delete) them or add new ones.

Use Case 9: View Assigned Traineeship Positions

Use case ID	UC9
Actors	Company (authenticated user)
Pre conditions	- The company is logged in successfully. - At least one of the company's posted traineeship positions has been matched to a student by the practice office.
Main flow of events	1. The use case starts when the company logs in and lands on the dashboard (`dashboard.html`). 2. The system checks if there are assigned (matched) traineeship positions for the company. 3. If there are, the system hides the list of unmatched posted positions. 4. The dashboard displays only the list of assigned positions, including student details.
Alternative flow 1	If no positions have been matched yet, the system displays the posted (unmatched) positions as usual.
Alternative flow 2	-
Post conditions	- The company can view the list of their traineeship positions that have been assigned to students. - The company is able to evaluate each assigned student through the provided evaluation interface.

Use Case 10: Add New Traineeship Position

Use case ID	UC10
Actors	Company (authenticated user)
Pre conditions	- The company is logged in. - The company is viewing their dashboard (`dashboard.html`).
Main flow of events	1. The use case starts when the company clicks the "Add New Position" button on the dashboard. 2. The system redirects the user to the new position form (`new_position.html`). 3. The company fills in the position details: 3.1. Title 3.2. Description of the work to be done 3.3. Start and end dates 3.4. Required skills 3.5. Related interests/topics 4. The company submits the form. 5. The system validates the input and stores the new traineeship position in the database. 6. The user is redirected back to the dashboard. 7. The newly added position is now visible in the list of posted positions.
Alternative flow 1	If the submitted form contains invalid or missing data (e.g., missing dates or required fields), the system shows error messages and remains on the form.
Alternative flow 2	If the position violates internal validation rules (e.g., end date before start date), the system prevents submission and displays feedback.
Post conditions	- A new traineeship position is created and associated with the company. - The position appears immediately in the dashboard under the list of posted positions.

Use Case 11: Delete Posted Traineeship Position

Use case ID	UC11
Actors	Company (authenticated user)
Pre conditions	- The company is logged in and viewing their dashboard (`dashboard.html`). - The dashboard displays one or more posted traineeship positions.
Main flow of events	1. The use case starts when the company views their dashboard. 2. The company identifies a traineeship position they want to delete. 3. The company clicks the "Delete" button on that position's card. 4. The system sends a POST request to delete the specific position. 5. The position is deleted from the database. 6. The dashboard is automatically updated to remove the deleted position from the list.
Alternative flow 1	-
Alternative flow 2	-
Post conditions	- The selected traineeship position is permanently removed from the database. - The position no longer appears in the dashboard view.

Use Case 12: Submit Evaluation for Assigned Student

Use case ID	UC12
Actors	Company (authenticated user with assigned students)
Pre conditions	- The company is logged in. - One or more traineeship positions have been assigned to students. - The company dashboard displays the assigned traineeships.
Main flow of events	1. The use case starts when the company views its dashboard (`dashboard.html`). 2. For each assigned student, an "Evaluate" button is available. 3. The company clicks the "Evaluate" button for a specific student. 4. The system redirects the user to the evaluation page (`company/evaluation.html`). 5. The company fills in the evaluation form: 5.1. Motivation (1 to 5) 5.2. Effectiveness (1 to 5) 5.3. Efficiency (1 to 5) 5.4. Optional comments 6. The company submits the form. 7. The system validates the input and saves the evaluation. 8. If an evaluation already exists for the same assignment, it is updated (overridden). 9. The user is redirected back to the dashboard.
Alternative flow 1	If the submitted form contains invalid data (e.g., missing scores), the system displays validation messages and stays on the evaluation page.
Alternative flow 2	-
Post conditions	- The evaluation for the selected student is stored or updated in the system. - The company can re-access the evaluation at any time and resubmit if needed.

Use Case 13: Professor Registration Profile Creation

Use case ID	UC13
Actors	Unauthenticated user (who selects role = Professor)
Pre conditions	- The user has selected "Professor" during initial role selection in Use Case UC1. - The professor registration form (`professor_registration.html`) is accessible.
Main flow of events	1. The use case starts when the user is redirected to `professor_registration.html` after selecting "Professor" in UC1. 2. The user fills in the professor profile: 2.1. Full name 2.2. List of interests 3. The user submits the form. 4. The system validates the input. 5. If all data is valid, the professor account is created in the database. 6. The user is redirected to `register_success.html`.
Alternative flow 1	If the form contains missing or invalid fields, the system shows error messages and remains on the registration form.
Alternative flow 2	-
Post conditions	- A new professor account is created with a complete profile. - The professor can log in and access their dashboard.

Use Case 14: View Supervised Traineeship Positions

Use case ID	UC14
Actors	Professor (authenticated user)
Pre conditions	- The professor is logged in. - At least one student has been matched to the professor through the practice office assignment process.
Main flow of events	1. The use case starts when the professor logs into the system. 2. The system redirects the professor to their dashboard (`dashboard.html`). 3. The system retrieves and displays the list of students assigned to the professor. 4. For each student, basic trainee information is displayed (e.g., name, position, company). 5. The dashboard dynamically updates based on the assignment data in the system.
Alternative flow 1	If the professor has no assigned students, the dashboard displays a message indicating that no students have been assigned yet.
Alternative flow 2	-
Post conditions	- The professor can view all assigned students and their traineeship details.

Use Case 15: Professor Evaluation for Traineeship

Use case ID	UC15
Actors	Professor (authenticated user with assigned students)
Pre conditions	- The professor is logged in. - The professor has at least one assigned (matched) student through the traineeship matching process. - The dashboard displays the assigned students along with the option to evaluate them.
Main flow of events	1. The use case starts when the professor views the dashboard (`dashboard.html`). 2. Each supervised student has an "Evaluate" button. 3. The professor clicks "Evaluate" for a specific student. 4. The system redirects to the evaluation form (`professor/evaluation.html`). 5. The professor fills in the evaluation: 5.1. Student motivation (1 to 5) 5.2. Student effectiveness (1 to 5) 5.3. Student efficiency (1 to 5) 5.4. Company facilities (1 to 5) 5.5. Company guidance (1 to 5) 6. The professor submits the form. 7. The system validates and stores the evaluation. 8. If an evaluation already exists, it is overridden. 9. The professor is redirected back to the dashboard.
Alternative flow 1	If any required field is missing or invalid, the system stays on the form and displays error messages.
Alternative flow 2	-
Post conditions	- The professor's evaluation of the student is updated. - The dashboard reflects the presence of an evaluation.

Use Case 16: View All Students Who Applied for a Traineeship

Use case ID	UC16
Actors	Practice Office member (authenticated user)
Pre conditions	- The practice office user is logged in. - One or more students have registered in the system. - Registration by a student is interpreted as an expression of interest in receiving a traineeship.
Main flow of events	1. The use case starts when the practice office accesses their dashboard (`dashboard.html`). 2. The user clicks on "Manage Students", which leads to `students/dashboard.html`. 3. The system retrieves and displays a list of all registered students. 4. For each student, the system shows profile details such as full name, university ID, skills, interests, and preferred location. 5. The list is automatically updated as new students register.
Alternative flow 1	If no students have registered yet, the system displays a message such as "No registered students found".
Alternative flow 2	-
Post conditions	- The practice office can view and manage all students who are eligible for traineeship assignment. - The practice office can delete a student from the list if .

Use Case 17–19 (Combined): Execute Matching and Assignments

Use case ID	UC17_19
Actors	Practice Office member (authenticated user)
Pre conditions	- The practice office user is logged in. - There are registered students, available traineeship positions, and eligible professors in the system.
Main flow of events	1. The use case starts when the practice office opens their dashboard (`dashboard.html`). 2. The user selects: 2.1. A matching strategy for student-to-position (e.g., based on interests, skills, or location) 2.2. A matching strategy for assigning a professor to each traineeship (e.g., based on interest or load) 3. The user clicks the "Execute Matching" button. 4. The system processes all students and positions based on the selected strategies. 5. For each compatible student-position pair: 5.1. The student is assigned to the position. 5.2. A supervising professor is assigned to the traineeship. 6. The number of matched assignments is updated on the dashboard (e.g., matched count). 7. Assigned internships become visible in the respective dashboards (student, company, professor).
Alternative flow 1	If no suitable matches are found for any student or position, no assignment is performed and the dashboard remains unchanged.
Alternative flow 2	-
Post conditions	- Successfully matched students are assigned to traineeship positions. - Each position is linked to a supervising professor. - The dashboard updates the number of assigned internships. - Unmatched students or positions remain unaffected.

Use Case 20: View All Traineeship Assignments (Matched Internships)

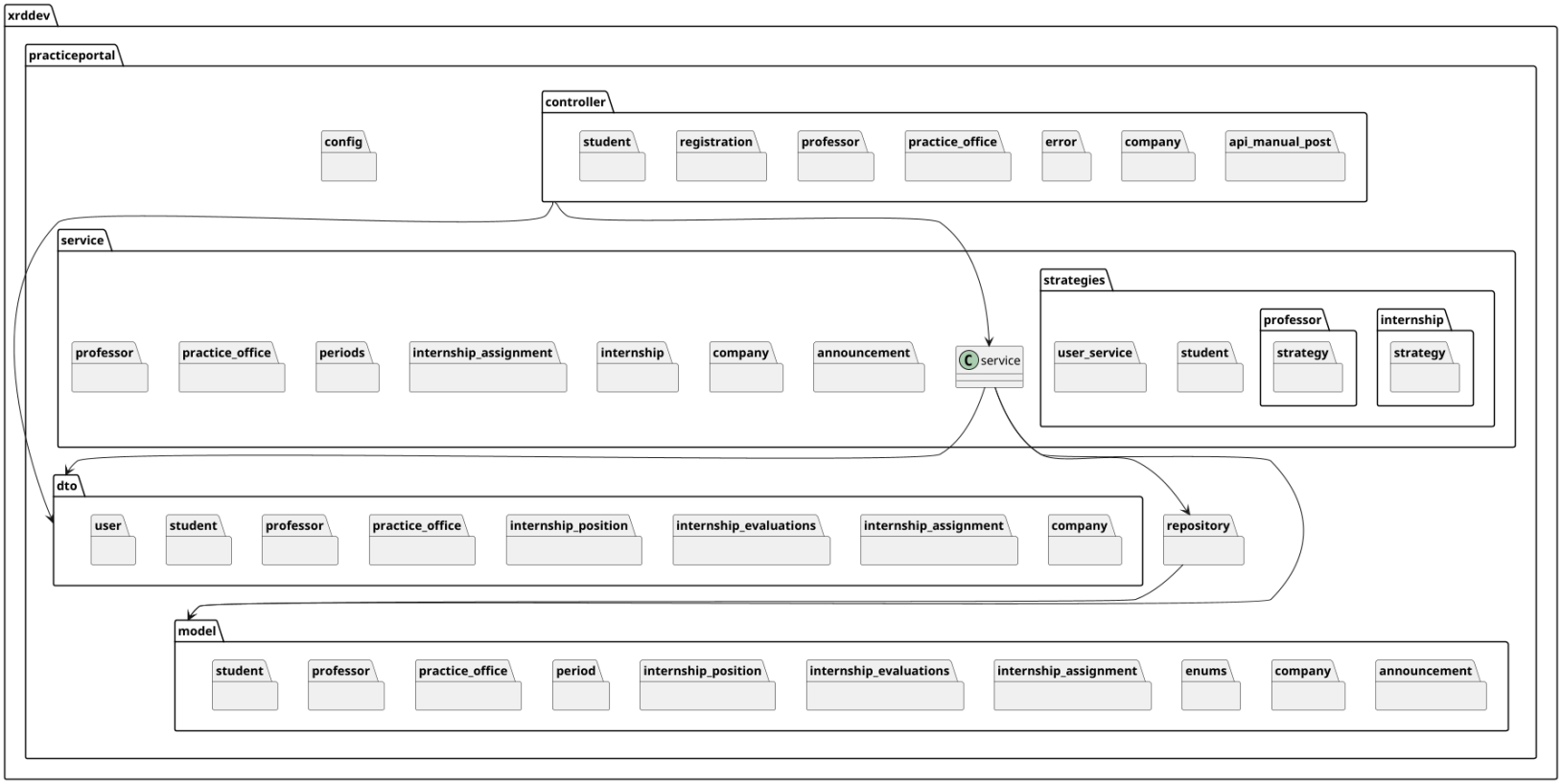
Use case ID	UC20
Actors	Practice Office member (authenticated user)
Pre conditions	- The practice office user is logged in. - At least one traineeship assignment has been created through the matching process.
Main flow of events	1. The use case starts when the practice office user opens the dashboard. 2. The user clicks on the "Manage Assignments" button. 3. The system redirects to the `assigned-internships` page. 4. The system retrieves and displays the list of matched assignments. 5. For each assignment, the following are displayed: 5.1. Student details (name, email, department, interests, skills) 5.2. Assigned position and company info 5.3. Supervising professor 5.4. Matching strategies and assignment date 6. The user may click "See All Evaluations" for each assignment to review feedback.
Alternative flow 1	If no assignments exist, the system displays a message such as "No traineeship assignments found."
Alternative flow 2	-
Post conditions	- The practice office has access to all currently matched traineeship assignments. - The office can review evaluation data and track assignment details.

Use Case 21: Monitor Internship Evaluations

Use case ID	UC21
Actors	Practice Office member (authenticated user)
Pre conditions	- The practice office user is logged in. - One or more internship assignments have been created. - At least one of the involved parties (student, professor, or company) has submitted an evaluation.
Main flow of events	1. The use case starts when the practice office accesses the "Manage Assignments" page. 2. The user selects an assignment and clicks "See All Evaluations". 3. The system redirects to the `evaluations.html` page for the selected assignment. 4. The system displays: 4.1. Company's evaluation of the student (motivation, effectiveness, efficiency, overall grade, comments) 4.2. Professor's evaluation of the student and hosting company 4.3. Student's evaluation of the internship 5. The evaluations are shown in dedicated panels. 6. Below each evaluation panel, the system always displays a static explanatory message (placeholder), regardless of whether the evaluation has been submitted or not. 7. The user clicks the "Back to Dashboard" button to return to the main Practice Office dashboard.
Alternative flow 1	-
Alternative flow 2	-
Post conditions	- The practice office can view the evaluations submitted for a specific traineeship assignment.

3 Design

3.1 Architecture



3.2 CRC CARDS

Class Name: InterestsAndSkillsMatching	
Responsibilities: ● Implement internship assignment strategy based on both interests and skills. ● Compare each student with available positions using Jaccard similarity. ● Filter positions that exceed similarity threshold. ●Select the first compatible position and create an InternshipAssignment. ● Mark the assignment with the used strategy and current date.	**Collaborations:** ● Student ● InternshipPosition ● InternshipAssignment ● SimilarityMetrics ● InternshipMatchingOptions

Class Name: MinLoadMatching	
Responsibilities: ●Assign professors to internship assignments by selecting the one with the **minimum current load**. ● Annotate each assignment with the selected professor and the strategy used. ● Ensure a balanced distribution of assignments among available professors.	**Collaborations:** ● ProfessorService ● Professor ● InternshipAssignment ● ProfessorMatchingOptions

Class Name: InternshipAssignmentServiceImpl	
Responsibilities: ● Retrieve internship assignments by: ○ student email, ○ professor email, ○ company email, ○ assignment and user identifiers. ● Convert assignments to InternshipAssignmentDashboardDto views. ● Handle evaluation submissions from: ○ students, ○ companies, ○ professors. ● Create and update the appropriate evaluation entity inside the assignment. ● Retrieve combined evaluations grouped by assignment. ● Delete assignments by ID.	**Collaborations:** ● InternshipAssignmentRepository ● InternshipAssignmentDashboardDto ● StudentInternshipEvaluation ● CompanyInternshipEvaluation ● ProfessorInternshipEvaluation ● StudentInternshipEvaluationDashboardDto ● CompanyInternshipEvaluationDashboardDto ●ProfessorInternshipEvaluationDashboardDto ●CombinedInternshipEvaluationDashboardDto ● InternshipAssignmentMapper ●InternshipAssignment

Class Name: CompanyService	
Responsibilities: ● Register a new company with profile and internship coordinator details. ● Securely encode and store the company's password. ●Retrieve a company by email or ID for authentication or management. ●Update a company's profile information (name, address, contact, coordinator). ●Return company data mapped as DashboardDto or EditDto. ●Count all registered companies in the system. ● Delete companies by ID. ●Provide a list of all companies as dashboard DTOs.	**Collaborations:** ● CompanyRepository ● PasswordEncoder ● CompanyDashboardDto ● CompanyEditDto ● Company

Class Name: ProfessorService	
Responsibilities: ● Register new professors using profile and credential data. ● Retrieve a professor by email and map to various DTOs (e.g., EditDto, DashboardDto). ● Update professor profile information (name, interests). ● Count the total number of professors in the system. ● Delete professors by ID (used in admin panel). ● Return a list of all professors (raw or mapped). ● Securely hash passwords using BCryptPasswordEncoder.	**Collaborations:** ● ProfessorRepository ● BCryptPasswordEncoder ● ProfessorEditDto ● ProfessorDashboardDto

Class Name: StudentServiceImpl	
Responsibilities: ● Register a new student, hash their password, and persist their profile. ● Update student profile data (name, year, skills, interests, preferred location). ● Retrieve student information by email or ID. ● Map student entities to dashboard or edit DTOs. ● Provide a list of all students or their mapped DTOs. ● Count the total number of students. ● Delete students by ID (admin operation).	**Collaborations:** ● StudentRepository ● BCryptPasswordEncoder ● StudentDashboardDto ● StudentEditDto

Class Name: PracticeOfficeAdminServiceImpl	
Responsibilities: ● Register a new Practice Office admin using email and password. ● Persist the new admin entity with role PRACTICE_OFFICE. ● Find an admin by email for authentication or lookup purposes. ●Convert a PracticeOfficeAdmin entity to a PracticeOfficeAdminDto.	**Collaborations:** ● PracticeOfficeAdminRepository ● PracticeOfficeAdminDto ● PracticeOfficeAdmin

Class Name: InternshipPositionServiceImpl	
Responsibilities: ●Create a new internship position using company identity and position data. ●Update or delete an internship position only if it belongs to the authenticated company. ●Retrieve internship positions (by ID, company ID, or email), mapped to DashboardDto or EditDto. ●Count the total number of internship positions in the system. ●Delete internship positions by ID or by validated ownership. ●Provide all internship positions as full entities or as DTO view	**Collaborations:** ● InternshipPositionRepository ● CompanyService ●InternshipPositionDashboardDto ●InternshipPositionEditDto ●Company