Diploma Projects Management App

Sprint Report

Group members:

Surname/First name	**AM**	**Email**
Theodoridis Charalampos	4674	cs04674@uoi.gr
Mponitsis Pantelis	4742	cs04742@uoi.gr
Sidiropoulos Georgios	4789	cs04789@uoi.gr

Contents

1 Introduction

1.1 Purpose

The objective of this project is to develop a Web-based application enabling students to explore a range of available diploma thesis projects offered by professors and apply for those that they find interesting. Additionally, the application provides professors with the ability to assign diploma thesis projects to students with some strategies, supervise the assigned theses projects, as well as evaluate the outcomes.

1.2 Document Structure

The rest of this document is structured as follows. Section 2 describes out Scrum team and specifies the this Sprint's backlog. Section 3 specifies the main design concepts for this release of the project.

2 Scrum team and Sprint Backlog

2.1 Scrum team

Product Owner	Theodoridis Charalampos, Mponitsis Pantelis, Sidiropoulos Georgios
Scrum Master	Theodoridis Charalampos, Mponitsis Pantelis, Sidiropoulos Georgios
Development Team	Theodoridis Charalampos, Mponitsis Pantelis, Sidiropoulos Georgios

2.2 Sprints

Sprint No	Begin Date	End Date	Number of weeks	User stories
1	**March 5th**	**March 12th**	**1**	**U1 , U2 , S1 , P1**
2	**March 12th**	**March 26th**	**2**	**S2 , S3 , P2 , P3 , P5 , P7**
3	**April 1st**	**April 8th**	**1**	**S4 , P6**
4	**April 8th**	**April 15th**	**1**	**P4 , P8 , P9**

3 Use Cases

3.1 Register

Use case ID	Register
Actors	User
Pre conditions	The user does not have an account in the application.
Main flow of events	1. The use case starts when the user selects the "Register" button on the homepage. 2. The system displays a registration form. 3. The user fills in their username, password, and role. 4. The system saves the new user in the database.
Alternative flow 1	If the user selects a username that already exists, the system displays a message, and the registration process needs to be repeated.
Post conditions	The user has an account in the application and can log in to it.

3.2 Login

Use case ID	Login
Actors	User
Pre conditions	The user have an account in the application.
Main flow of events	1. The use case starts when the user selects the "Login" button on the homepage. 2. The system displays a login form. 3. The user fills in their username and password.
Alternative flow 1	If the user enters an incorrect username or password, the system displays a message, and the login process needs to be repeated.
Post conditions	The user depending on their role, can access the main menu of the application.

3.3 SetProfileInformation

Use case ID	SetProfileInformation
Actors	User
Pre conditions	The user is in the registration process.
Main flow of events	1. The use case starts when the system displays a login form with the information the need to fill in according to their role. 2. The user fills in the form details.
Alternative flow 1	If the user is in the main menu of the application can select the "Set personal information" and follow the same process.
Post conditions	The system has stored the user's information and can utilize it for future processes.

3.4 GetListOfAvailableSubjects

Use case ID	GetListOfAvailableSubjects
Actors	Student
Pre conditions	The student has logged in and is currently in the main menu of the application.
Main flow of events	1. The use case starts when the student selects the “Available Subjects” button on the main menu. 2. The system displays a list of all available subjects offered by all professors.
Alternative flow 1	If there are no available subjects, the system displays an empty list.
Post conditions	The student can view the available subjects, read their descriptions and apply for the ones they are interested in.

3.5 GetSubjectInformation

Use case ID	GetSubjectInformation
Actors	Student
Pre conditions	The student is on the page that displays the list of available subjects, which is not empty.
Main flow of events	1. The use case starts when the student selects the “Subject’s information” button on the specific available subject. 2. The system displays a table containing a detailed description of the subject .
Post conditions	The student can read the information about the subject and decide whether they are interested in it.

3.6 ApplyForSubject

Use case ID	ApplyForSubject
Actors	Student
Pre conditions	The student is on the page that displays the list of available subjects, which is not empty.
Main flow of events	1. The use case starts when the student selects the "Apply" button on the specific available subject. 2. The system generates an application that includes the student and saves it to the appropriate field of the specific subject.
Alternative flow 1	If the student has already submitted an application for the specific subject, the system does not create a new application.
Post conditions	The professor can view the application of the student, as well as their details, in the list of applications for the specific subject.

3.7 GetProfessorAvailableSubjects

Use case ID	GetProfessorAvailableSubjects
Actors	Professor
Pre conditions	The professor has logged in and is currently in the main menu of the application.
Main flow of events	1. The use case starts when the professor selects the "My Available Subjects" button on the main menu. 2. The system displays a list of all available subjects offered by the professor.
Alternative flow 1	If the professor has no available subjects, the system displays an empty list.
Post conditions	The professor can view the subjects they offer, add a new subject, update or delete a subject or assign a subject to a student.

3.8 AddSubject

Use case ID	AddSubject
Actors	Professor
Pre conditions	The professor is on the page that displays the list of available subjects that they offer.
Main flow of events	1. The use case starts when the professor selects the "Add Subject" button on the top of the page. 2. The system displays a form that contains the information of the new subject which the professor needs to fill in. 3. The professor completes the details of the new subject. 4. The system saves the new subject in the database.
Post conditions	The professor and the students can view the new subject in the application.

3.9 DeleteSubject

Use case ID	DeleteSubject
Actors	Professor
Pre conditions	The professor is on the page that displays the list of available subjects that they offer and the list is not empty.
Main flow of events	1. The use case starts when the professor selects the "Delete" button on the specific subject. 2. The system deletes the subject from the database along with all the applications made for it.
Post conditions	The professor and the students cannot view anymore the subject in the application.

3.10 GetSubjectApplicationList

Use case ID	GetSubjectApplicationList
Actors	Professor
Pre conditions	The professor is on the page that displays the list of available subjects that they offer and the list is not empty.
Main flow of events	1. The use case starts when the Professor selects the "Applications" button on the specific available subject. 2. The system displays a list of applications for the specific subject. Each application includes the student's details.
Alternative flow 1	If there are no applications for the specific subject, the system displays an empty list.
Post conditions	The professor can view the applications of the subject.

3.11 AssignSubject

Use case ID	AssignSubject
Actors	Professor
Pre conditions	The professor is on the page that displays the list of available subjects that they offer and the list is not empty.
Main flow of events	1. The use case starts when the Professor selects the "Assign Subject" button on the specific available subject. 2. The system displays a page containing the assignment strategies. 3. The professor selects a strategy. 4. The system finds the best applicant from the applications for the subject. 5. The system deletes the specific subject from the database, along with all the applications that have been made for it. 6. The system displays the page with the assigned diploma thesis subjects of the professor, where the subject that was assigned is included. Also, this subject contains information about the assigned student as well.
Alternative flow 1	If there are no applications for the specific subject, the system does not assign the subject and displays the page with the professor's diploma thesis subjects where the specific subject is not present.
Post conditions	The professor can add the grades of the new diploma thesis, update them or delete the specific diploma thesis subject.

3.12 GetProfessorThesis

Use case ID	GetProfessorThesis
Actors	Professor
Pre conditions	The professor has logged in and is currently in the main menu of the application.
Main flow of events	1. The use case starts when the professor selects the "My Assigned Thesis Subjects" button on the main menu. 2. The system displays a list of diploma thesis subjects supervised by the professor.
Alternative flow 1	If the professor has no supervise diploma thesis subjects, the system displays an empty list.
Post conditions	The professor can view the diploma thesis subjects that supervise, add the grades of a diploma thesis subject, update them or delete a specific diploma thesis subject.

3.13 AddGrades

Use case ID	AddGrades
Actors	Professor
Pre conditions	The professor is on the page that displays the list of diploma thesis subjects that they supervise and the list is not empty.
Main flow of events	1. The use case starts when the professor selects the "Add grades" button on the specific diploma thesis subject. 2. The system displays a form that contains the grades of the diploma thesis subject which the professor needs to fill in. 3. The professor completes the grades of the diploma thesis subject. 4. The system the final grade of the diploma thesis subject based on a weighted average formula. 5. The system displays the list of diploma thesis subjects supervised by the professor, including the specific diploma thesis subject with updated grades and a filled final grade.
Alternative flow 1	If the diploma thesis subject already has grades, the system updates them and calculate a new final grade.
Post conditions	The professor can view the diploma thesis subject with the updated grades.

3.14 DeleteThesis

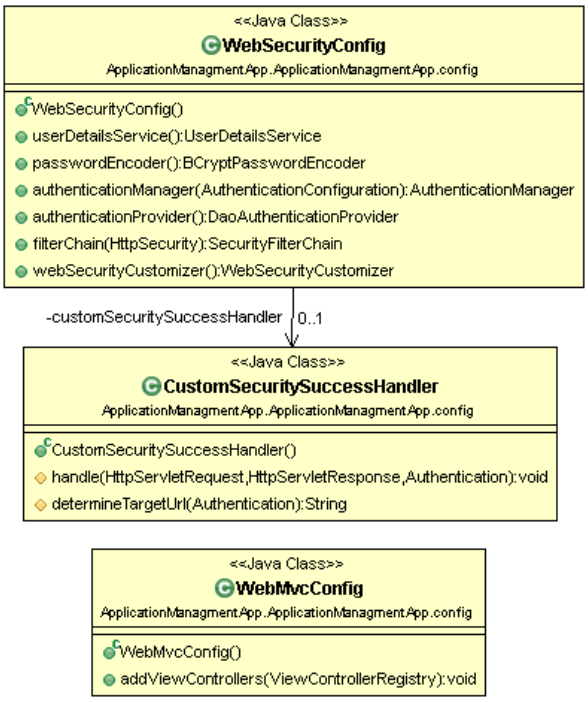
Use case ID	DeleteThesis
Actors	Professor
Pre conditions	The professor is on the page that displays the list of diploma thesis subjects that they supervise and the list is not empty.
Main flow of events	1. The use case starts when the professor selects the "Delete" button on the specific diploma thesis subject. 2. The system deletes the diploma thesis subject from the database.
Post conditions	The professor cannot view and supervise anymore the diploma thesis subject in the application.

4 Design

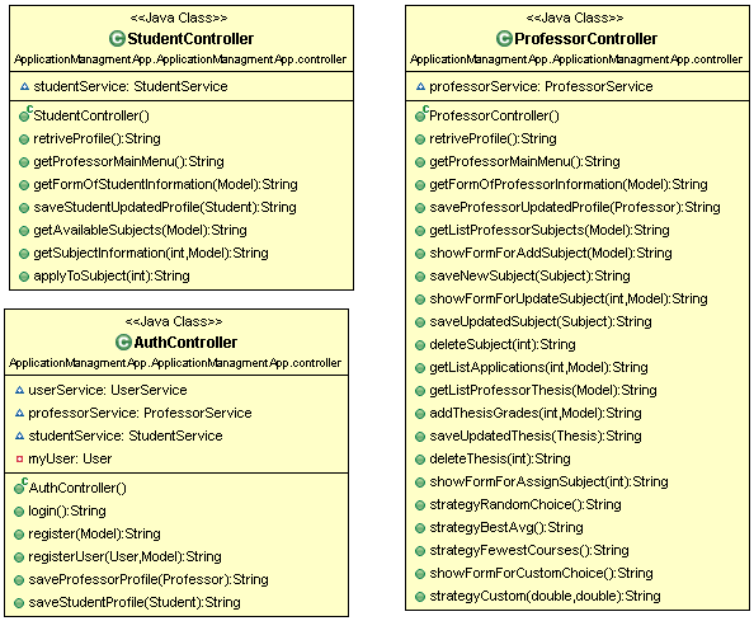
4.1 Architecture

- An overall UML package diagram

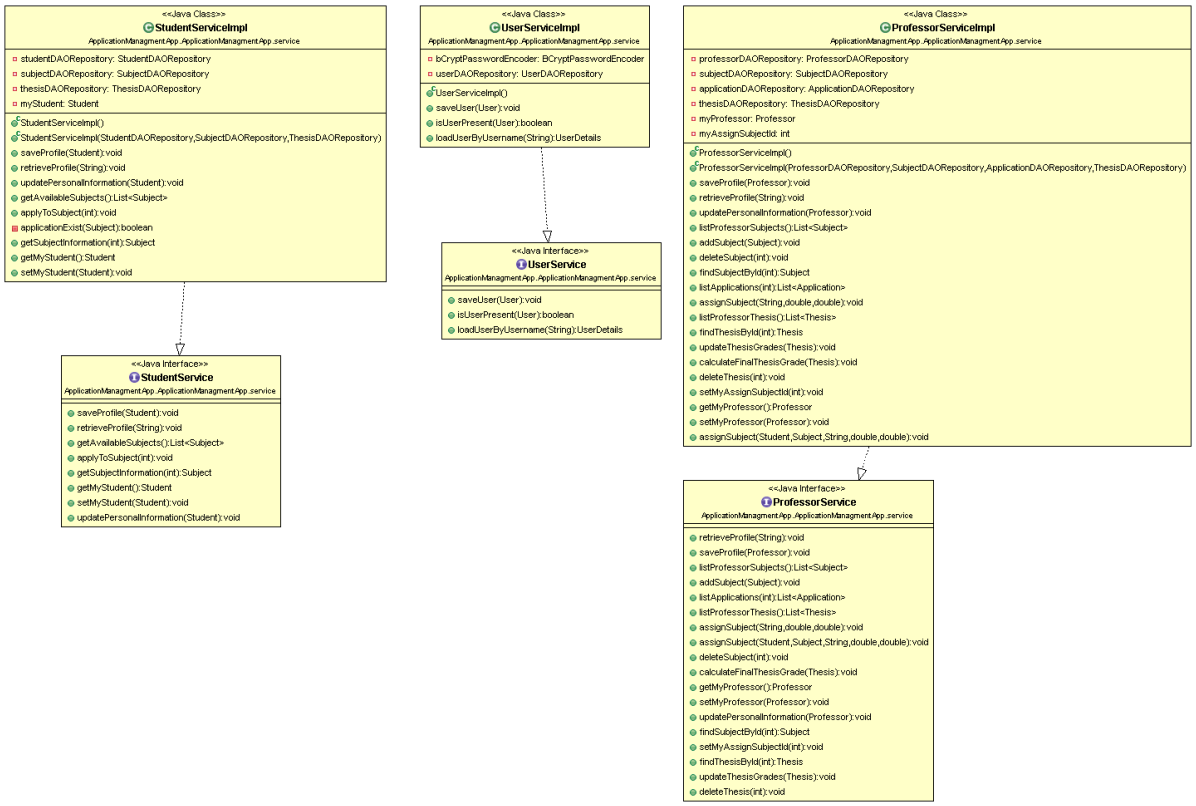
- UML class diagram for the classes of package config.



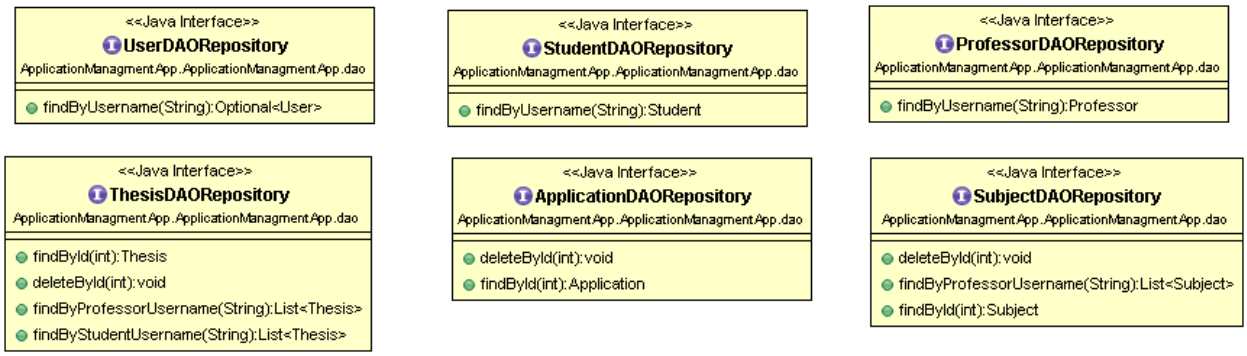
- UML class diagram for the classes of package controller.



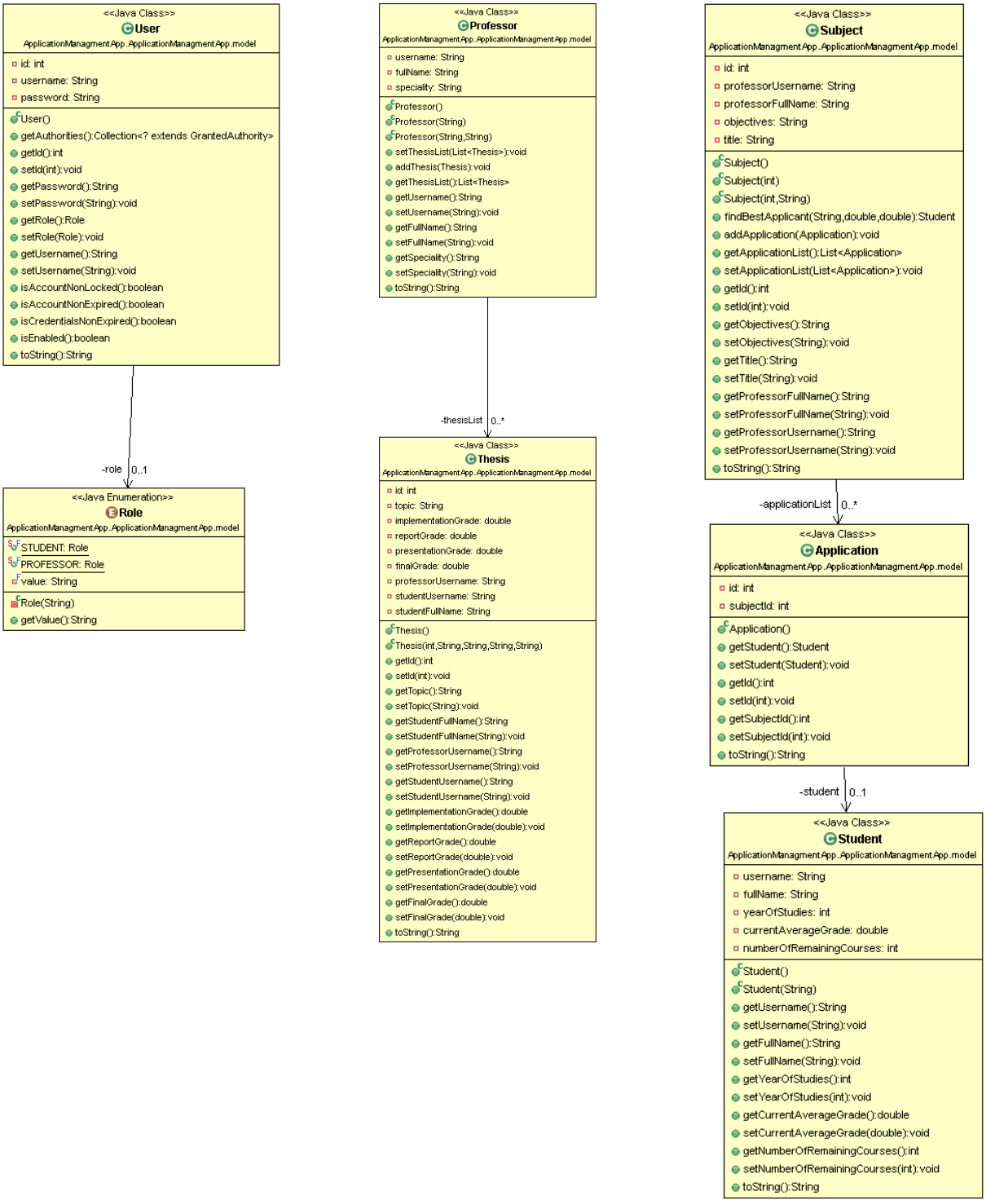
- UML class diagram for the classes of package service.

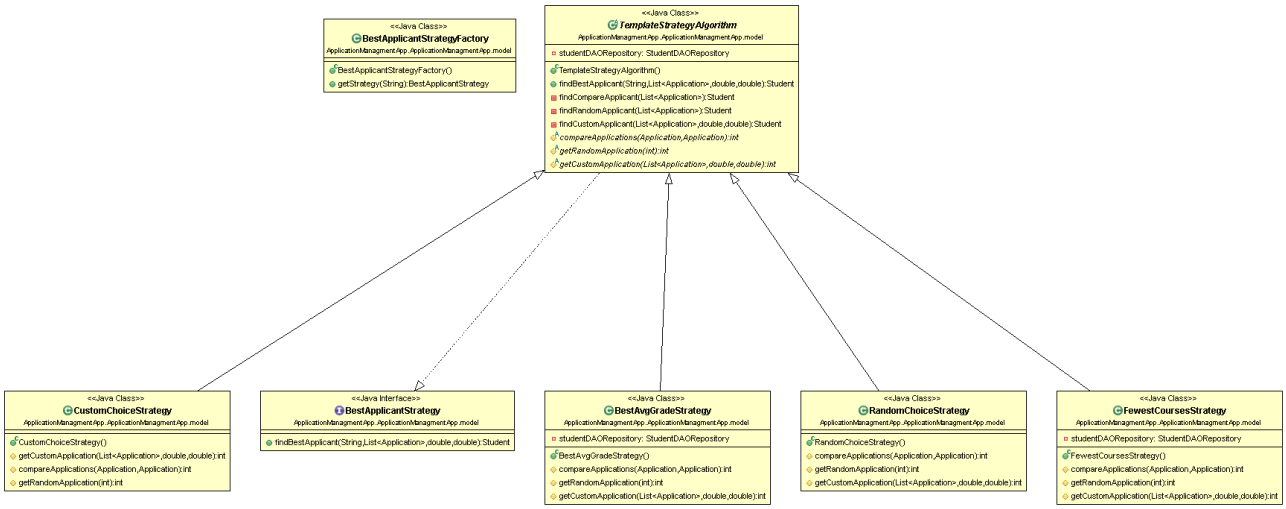


- UML class diagram for the classes of package dao.



- UML class diagram for the classes of package model.





4.2 Design

Class Name: ProfessorController	
Responsibilities: ▪ This class serves as a controller of the Spring Framework for the "Professor" entity.	**Collaborations:** -

Class Name: StudentController	
Responsibilities: ▪ This class serves as a controller of the Spring Framework for the "Professor" entity.	**Collaborations:** -

Class Name: StudentServiceImpl	
Responsibilities: ▪ This class implements StudentService interface. ▪ This class serves as a service of the Spring Framework for the "Student" entity. ▪ This class includes the methods that implement the user stories of a Student.	**Collaborations:** -

Class Name: ProfessorServiceImpl	
Responsibilities: ▪ This class implements ProfessorService interface. ▪ This class serves as a service of the Spring Framework for the "Professor" entity. ▪ This class includes the methods that implement the user stories of a Professor.	**Collaborations:** -

Class Name: Professor	
Responsibilities: ▪ This class holds information for a Professor.	**Collaborations:** ▪ This class has access to the Thesis class.

Class Name: Student	
Responsibilities: ▪ This class holds information for a Student.	**Collaborations:** -

Class Name: Subject	
Responsibilities: ▪ This class holds information for a Subject. ▪ This class finds the best applicant whom the subject will be assigned.	**Collaborations:** ▪ This class has access to the Application class.

Class Name: Application	
Responsibilities: ▪ This class holds information for an Application.	**Collaborations:** ▪ This class has access to the Student class.

Class Name: Thesis	
Responsibilities: ▪ This class holds information for a diploma thesis subject.	**Collaborations:** -

Class Name: BestApplicantStrategyFactory	
Responsibilities: ▪ This class serves as a factory that creates objects for the strategy pattern of subject assignment.	**Collaborations:** -

Class Name: TemplateStrategyAlgorithm	
Responsibilities: ▪ This class implements BestApplicantStrategy interface and the method defined in it.	**Collaborations:** -

Class Name: CustomChoiceStrategy	
Responsibilities: ▪ This class extends TemplateStrategyAlgorithm class and implements the abstract methods defined in it. ▪ This class implements the subject assignment strategy called “Custom Choice”.	**Collaborations:** -

Class Name: BestAvgGradeStrategy	
Responsibilities: ▪ This class extends TemplateStrategyAlgorithm class and implements the abstract methods defined in it. ▪ This class implements the subject assignment strategy called “Best Average Grade”.	**Collaborations:** -

Class Name: RandomChoiceStrategy	
Responsibilities: ▪ This class extends TemplateStrategyAlgorithm class and implements the abstract methods defined in it. ▪ This class implements the subject assignment strategy called “Random Choice Strategy”.	**Collaborations:** -

Class Name: FewestCoursesStrategy	
Responsibilities: ▪ This class extends TemplateStrategyAlgorithm class and implements the abstract methods defined in it. ▪ This class implements the subject assignment strategy called “Fewest Courses Strategy.”.	**Collaborations:** -