
Diploma Projects Management App

Sprint Report

<AS TEAM>

<Archontis Nestoras-4747 & Spyridon Chalidias-4830>

VERSIONS HISTORY

Date	Version	Description	Author
20/5/2023	v1.0	Final report based on deliverable code	Archontis Nestoras Spyridon Chalidias

1 Introduction

This document provides information concerning the **1-5** sprint of the project.

1.1 Purpose

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	Archontis Nestoras
Scrum Master	Spyridon Chalidias
Development Team	Archontis Nestoras, Spyridon Chalidias

2.2 Sprints

<List below the sprints that you performed and the user stories that have been realized in each Sprint>

Sprint No	Begin Date	End Date	Number of weeks	User stories
1	5 th of March	19 of March	2 weeks	<p>As a professor I want to be able to make an account and log in to that account.</p> <p>As a student I want to be able to make an account and log into that account</p>
2	20 th of March	4 th of April	2 weeks	<p>As a professor I want to be able to put my information to my account and to publish the diploma thesis that are available.</p> <p>As a student I want to be able to put my information to my account and see the available diploma thesis.</p>
3	5 th of April	19 th of April	2 weeks	<p>As a professor I want to be able to see al the students that assigned for each diploma thesis.</p> <p>As a student I want to be able to assign for the diploma thesis.</p>
4	20 th of April	4 th of May	2 weeks	<p>As a professor I want to be able to choose a strategy in order to assign the diploma thesis to the student.</p>
5	5 th of May	19 th of May	2weeks	<p>As wa professor I want to be able to put grades to the student.</p> <p>As a student I want to be able to see if I have a diploma thesis and what my grade is.</p>

3 Use Cases

<Specify the concrete Use Cases that describe the interaction of the user with the applications, as derived from the abstract user stories. Give a **UML Use Case diagram** and the **detailed use case descriptions**.>

3.1 <Create Professor Account>

Use case ID	UC 1
Actors	professor
Pre conditions	-
Main flow of events	<ol style="list-style-type: none">1. The use case starts when the professor presses the 'sign up' button2. The professor provides necessary details to create an account (username,password)3. The professor selects the property of "PROFESSOR"4. The professor presses the button 'sign up' in order to create the account
Alternative flow 1	If the username already exists an error message will appear
Post conditions	The account is created and redirect him to login page

3.2 <Create Student Account>

Use case ID	UC 2
Actors	student
Pre conditions	-
Main flow of events	<ol style="list-style-type: none">1. The use case starts when the student presses the 'sign up' button2. The student provides necessary details to create an account (username,password)3. The student selects the property of "STUDENT"4. The student presses the 'sign up' button in order to create the account

Alternative flow 1	If the username already exists an error message will appear
Post conditions	The account is created and redirect him to login page

3.3 <Professor Login>

Use case ID	UC 3
Actors	professor
Pre conditions	The professor has already created an account
Main flow of events	<ol style="list-style-type: none"> 1. The use case starts when the professor presses the 'Login' button 2. The professor provides the credentials (username,password) 3. The professor presses the button 'Login' order to connect to the account
Alternative flow 1	If the credentials are not correct, an error message will appear.
Post conditions	The professor has logged in to the account

3.4 <Fill Information of Professor>

Use case ID	UC 4
Actors	professor
Pre conditions	The professor has already logged into his account
Main flow of events	<ol style="list-style-type: none"> 1. The use case starts when the professor presses the 'Info' button 2. The professor writes his information 3. The professor presses the 'Save' button in order to save in the DB his new personal information
Alternative flow 1	The professor can change the info.

Post conditions	The professor has put his information to the account.
------------------------	---

3.5< Student Login>

Use case ID	UC 5
Actors	student
Pre conditions	The student has already created an account
Main flow of events	<ol style="list-style-type: none"> 1. The use case starts when the student presses the 'Login' button 2. The student provides the credentials (username,password) 3. the student presses the button 'Login' in order to connect to the account
Alternative flow 1	If the credentials are not correct, an error message will appear
Post conditions	The student has logged in to the account

3.6 <Fill Information of Student>

Use case ID	UC 6
Actors	student
Pre conditions	The student has already logged into his account
Main flow of events	<ol style="list-style-type: none"> 1. The use case starts when the student presses the info button 2. The student writes his information 3. The student presses the 'Save' button in order to save in the DB his new personal information
Alternative flow 1	The student can change the info.
Post conditions	The student has put his information to the account.

3.7<Manage subjects>

Use case ID	UC 7
Actors	professor
Pre conditions	The professor has already logged in to the account
Main flow of events	<ol style="list-style-type: none">1. The use case starts when the professor presses the 'List' button2. The professor can see the list of his own subjects3. If the professor press the 'Add Subject' button<ol style="list-style-type: none">3.1.The professor provides the diploma thesis subject details(title, objectives)3.2.The professor presses the 'Save' button to save the diploma thesis4. If the professor press the 'Delete' button<ol style="list-style-type: none">4.1. a warning message will pop4.2. the specific subject will be removed from DB
Post conditions	A new diploma thesis subject has been created or has been deleted or none of the above actions are done

3.8 < Apply for thesis>

Use case ID	UC 8
Actors	student
Pre conditions	The student has already logged in to the account
Main flow of events	<ol style="list-style-type: none">1. The use case starts when the student presses the 'Subjects' button in order for the available thesis subjects to appear2. The student can press the 'More' button in order to see more information for the specific subject3. The Student presses the 'Apply' button in order to assigne to the specific subject
Alternative flow 1	If the student has already enrolled the button will disappear.
Alternative flow 2	If the student has already a thesis the buttons will also disappear in order not to apply for other thesis.

Post conditions	A new application for a diploma thesis is created
------------------------	---

3.9 < Manage applications>

Use case ID	UC 9
Actors	professor
Pre conditions	The professor has already logged in to the account
Main flow of events	<ol style="list-style-type: none"> 1. The use case starts when the professor presses the 'Applications' button in order to see the applications(subject_name, student_name) 2. The professor can select one of the four options 3. The professor press the 'Assign' button in order to give the specific subject to a student
Alternative flow 1	If any student meets the criteria then the thesis is assigned to him
Alternative flow 2	Else a warning message will pop and no action will take place
Post conditions	A student has been chosen for a diploma thesis or not.

3.10 < Grade thesis>

Use case ID	UC 10
Actors	professor
Pre conditions	The professor has already logged in to the account
Main flow of events	<ol style="list-style-type: none"> 1. The use case starts when the professor presses the 'Assigned Diploma' button in order to appear a list of students with diploma thesis. 2. If the professor press the 'Grade' button <ol style="list-style-type: none"> 2.1. the professor grades the thesis in 3 different areas of the project 2.2.the professor presses the 'Submit' button and the total grade is calculated
Alternative flow 1	The professor can change the grade and the total grade will be updated

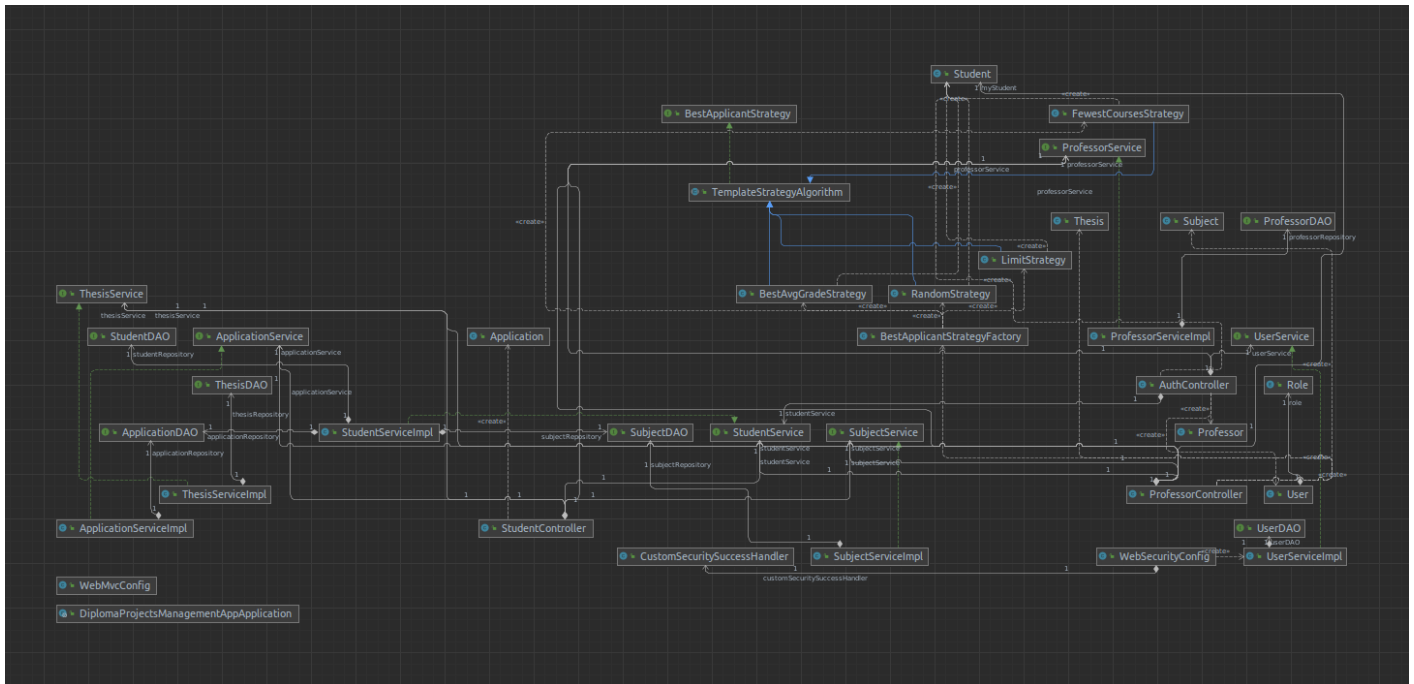
Post conditions	The professor has graded a student.
----------------------------	-------------------------------------



4 Design

4.1 Architecture use cases

<Specify the overall architecture for this release in terms of a **UML package diagram**.>



4.2 Design

<Specify the detailed design for this release in terms of **UML class diagrams**.>

<Document the classes that are included in this release in terms of CRC cards according to the template that is given below.>

Class Name: ProfessorController	
Responsibilities: <ul style="list-style-type: none">▪ Manage professor's dashboard▪ Manage professors' information▪ Manage subjects list▪ Add new subjects▪ Delete subjects▪ Manage application list▪ Assign best applicant to subject▪ Manage assigned diploma list▪ Set and save grade	Collaborations: <ul style="list-style-type: none">▪ ProfessorService▪ SubjectService▪ ApplicationService▪ StudentService▪ ThesisService▪ User▪ BestApplicantStrategyFactory and BestApplicantStrategy▪ Thesis, Student, Application, Subject, Professor

Class Name: StudentController	
Responsibilities: <ul style="list-style-type: none">▪ Manage student's dashboard▪ Manage student's information▪ Not assigned thesis list▪ More info for a subject▪ Assign for a thesis	Collaborations: <ul style="list-style-type: none">▪ StudentService▪ SubjectService▪ ProfessorService▪ ApplicationService▪ ThesisService▪ User▪ Student▪ Subject▪ Professor▪ Application▪ Thesis

Class Name: Application	
Responsibilities: <ul style="list-style-type: none"> ▪ Maintains application id ▪ Maintains subjectid ▪ Maintains studentid ▪ Provides getter/setter methods for application properties 	Collaborations: <ul style="list-style-type: none"> ▪ Interacts with the database ▪ Used by `ApplicationService` ▪ Used by `StudentController`

Class Name: Professor	
Responsibilities: <ul style="list-style-type: none"> ▪ Maintains professor id ▪ Maintains professor fullname ▪ Maintains studentid ▪ Maintains professor speciality ▪ Provides getter/setter methods for application properties 	Collaborations: <ul style="list-style-type: none"> ▪ Interacts with the database ▪ Used by `ProfessorService` ▪ Used by `StudentController`

Class Name: Student	
Responsibilities: <ul style="list-style-type: none"> ▪ Maintains student id ▪ Maintains student fullname ▪ Maintains year of studies ▪ Maintains current average grade ▪ Maintains number of remaining courses for graduation ▪ Maintains assigned status ▪ Provides getter/setter methods for student properties 	Collaborations: <ul style="list-style-type: none"> ▪ Interacts with the database ▪ Used by `StudentService` ▪ - Used by `StudentController`

Class Name: Subject	
Responsibilities: <ul style="list-style-type: none"> ▪ Maintains subject id ▪ Maintains subject title ▪ Maintains subject objectives ▪ Maintains subject supervisor ▪ Maintains assigned status ▪ Provides getter/setter methods for application properties 	Collaborations: <ul style="list-style-type: none"> ▪ Interacts with the database ▪ Used by `SubjectService` ▪ Used by `StudentController`

Class Name: Thesis	
Responsibilities: <ul style="list-style-type: none"> ▪ Maintains thesis id ▪ Maintains subjectsid ▪ Maintains studentid ▪ Maintains implementationgrade ▪ Maintains reportgrade ▪ Maintains presentationgrade ▪ Maintains final grade ▪ Provides getter/setter methods for application properties 	Collaborations: <ul style="list-style-type: none"> ▪ Interacts with the database ▪ Used by `ThesisService`

Class Name: User	
Responsibilities: <ul style="list-style-type: none"> ▪ Maintains user id ▪ Maintains username ▪ Maintains password ▪ Maintains role ▪ Maintains subjectsid ▪ Maintains studentid ▪ Provides getter/setter methods for application properties 	Collaborations: <ul style="list-style-type: none"> ▪ Interacts with the database ▪ Used by `UserDetailsService` ▪ Used by Spring Security

5 Testing

The test in Dao are done in a specific database. That means that the tests are succesfull only with that database. The screenshots of the database are bellow.

```
1 • |SELECT * FROM diplomatikh_directory.users;
```

Result Grid					Filter Rows:	Edit:	Export/Import:	
	id	user_name	password	role				
▶	1	p	\$2a\$10\$gRzhnYfIO/qWxPOwcmvs7.RYaBv8On...	PROFESSOR				
	2	s	\$2a\$10\$6dKz3Q/k.qtkjE6Ed2GlEul8tt4qIadBOs...	STUDENT				
	3	ss	\$2a\$10\$sIMhGLOepXbUxvL50hyZke4DkItvSsoa...	STUDENT				
	4	sss	\$2a\$10\$IRoft2QN25zDbtvOmHsjyOfEf74KX98...	STUDENT				
	5	ssss	\$2a\$10\$8WnuuE61ATkvvNznkXzj9ukTCP5SDHg...	STUDENT				
	6	ssssss	\$2a\$10\$2NWOWdGbaqanu7D7dvquBe3X2PWF...	STUDENT				
	7	pppp	\$2a\$10\$8/rdEogNckSmuEOjO/5W4e.1JzuFcw7...	PROFESSOR				
	8	sssss	\$2a\$10\$tre7g6v9I/NjvgxyzFNwg.rQHN4C6qwg...	STUDENT				
	9	aaa	\$2a\$10\$WiJeL07bO3MeLflR7FS0yeFInvib1HJM...	STUDENT				
	10	aa	\$2a\$10\$PeNxNppyQbhBT0Dvhi9EIOYju74jiINiQ...	STUDENT				
•	NULL	NULL	NULL	NULL				


```
1 • SELECT * FROM diplomatikh_directory.student;
```






<						
Result Grid						
Filter Rows:						
Edit:						
Export/Import:						
Wrap Cell Content:						
	user_id	full_name	year_of_studies	current_average_grade	number_of_remaining_courses_for_graduation	assigned
▶	2	s	NULL	NULL	NULL	1
	3	ss	8	2.4	9	1
	4	sss	9	2.1	9	1
	5	a	4	6	5	0
	6	John	NULL	NULL	NULL	0
	8	NULL	NULL	NULL	NULL	1
	9	sagfd	5	5	5	0
	10	NULL	NULL	NULL	NULL	0
*	NULL	NULL	NULL	NULL	NULL	NULL

```
1 • SELECT * FROM diplomatikh_directory.professor;
```

<			
Result Grid			
Filter Rows:			
Edit:			
	user_id	full_name	speciality
▶	1	NULL	NULL
	7	NULL	NULL
*	NULL	NULL	NULL

```
1 • SELECT * FROM diplomatikh_directory.subjects;
```






<

Result Grid   Filter Rows: Edit:   

	id	title	objectives	supervisor	assigned
▶	3	c		1	1
	4	aaa		1	1
	5	bbb		1	1
	6	qqq		1	0
	7	www		1	0
	8	eee		1	1
	10	ooo		7	1
	11	iii	iii	7	0
*	NULL	NULL	NULL	NULL	NULL

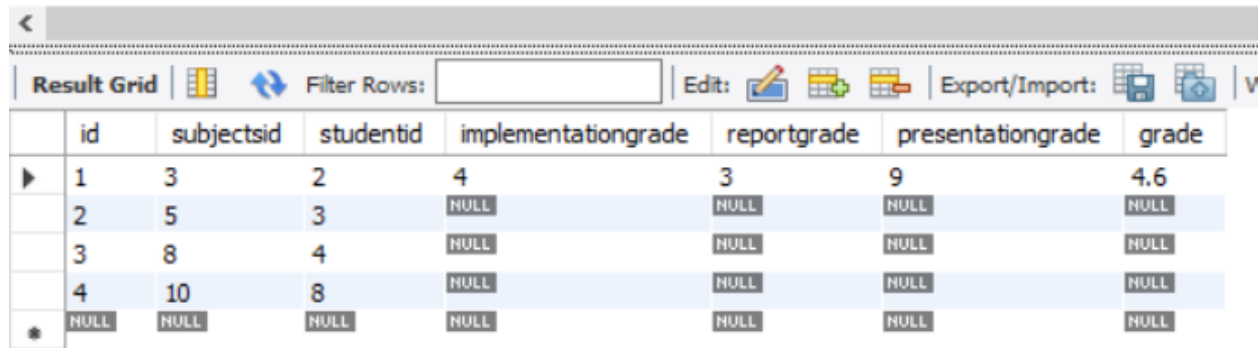
```
1 • SELECT * FROM diplomatikh_directory.application;
```

<

Result Grid   Filter Rows: Edit:    E

	id	subjectsid	studentid
▶	8	11	9
	9	6	9
*	NULL	NULL	NULL

```
1 • SELECT * FROM diplomatikh_directory.thesis;
```



	id	subjectsid	studentid	implementationgrade	reportgrade	presentationgrade	grade
▶	1	3	2	4	3	9	4.6
	2	5	3	NULL	NULL	NULL	NULL
	3	8	4	NULL	NULL	NULL	NULL
	4	10	8	NULL	NULL	NULL	NULL
*	5	NULL	NULL	NULL	NULL	NULL	NULL

6 Others

In the application.properties change the username and password for the DB.