Software Requirements Specifications Document

for

University Information System 4.0

**Version 1.0 approved**

**Prepared by;  
Emre Hancı  
Eren Kumru  
Mehmet Yıldız  
Mücahit V. Cumart  
Muhammed Fidan**

**Ankara Unity**

**12/05/2022**

**Table of Contents**

Table of Contents [ii](https://docs.google.com/document/d/1BDzulK4zUeThchvThsBRwOFLtwRZdxjs/edit#heading=h.1y810tw)

Revision History [ii](https://docs.google.com/document/d/1BDzulK4zUeThchvThsBRwOFLtwRZdxjs/edit#heading=h.30j0zll)

1. Introduction 8

1.1. Purpose 8

1.2. Scope 8

1.3. Definitions, acronyms, and abbreviations 8

1.4. References 9

1.5. Overview 10

2. Overall Description 10

2.1. Product Perspective 10  
 [2](https://docs.google.com/document/d/1BDzulK4zUeThchvThsBRwOFLtwRZdxjs/edit#heading=h.2s8eyo1).1.1. System Interfaces 10

2.1.2. User Interfaces 10

2.1.3. Hardware Interfaces 10

2.1.4. Software Interfaces 10

2.1.5. Communication Interfaces 11

2.1.6. Memory Constraints 11

2.1.7. Operations 11

2.1.8. Site Adaptation Requirements 11

2.2. Product Functions 11

2.3. User Characteristics 13

2.4. Constraints 13

2.4.1. Regulatory Policies 13

2.4.2. Hardware Limitations 13

2.4.3. Interfaces to Other Applications 14

2.4.4. Parallel Operation 14

2.4.5. Audit Functions 14

2.4.6. Control Functions 14

2.4.6.1. Input Validity Control 14

2.4.6.2. Integrity Control 14

2.4.7. Higher-order Language Requirements 14

2.4.8. Signal Handshake Protocols 14

2.4.9. Reliability Requirements 14

2.4.10. Criticality of the Application 15

2.4.11. Safety and Security Considerations 15

2.5. Assumptions and Dependencies 15

2.6. Apportioning of Requirements 15

3. Specific Requirements 16

3.1. External Interface Requirements 16

3.1.1. User Interfaces 16

3.1.1.1. Login Page 16

3.1.1.2. Forgot Password 17

3.1.1.3. Change Password 18

3.1.1.4. Homepage - Student 19

3.1.1.5. Homepage - Instructor 20

3.1.1.6. Course Registration 21

3.1.1.7. Add-Delete Course 22

3.1.1.8. Attending Classes 23

3.1.1.9. Assignment List - Student 24

3.1.1.10. Assignment List - Instructor 25

3.1.1.11. Assignment Details - Student 26

3.1.1.12. Assignment Details - Instructor 27

3.1.1.13. Messages 28

3.1.1.14. Course Documents 29

3.1.1.15. Exam List - Student 30

3.1.1.16. Exam List - Instructor 31

3.1.1.17. In - Exam 32

3.1.1.18. Exam Detail 33

3.1.1.19. Note List 34

3.1.1.20. Note Detail 35

3.1.1.21. Surveys 36

3.1.1.22. Community Management 37

3.1.1.23. Profile 38

3.1.1.24. Personal Notifications 39

3.1.1.25. Survey Notifications 40

3.1.1.26. Personal Messages 41

3.1.1.27. Community Messages 42

3.1.1.28. Create Course 43

3.1.1.29. Online Class 44

3.1.1.30. Assignment Create 45

3.1.1.31. Create Survey 46

3.1.1.32. Survey List 47

3.1.1.33. Exam Create 48

3.1.1.34. Exam Update 49

3.1.1.35. Exam Evaluating 50

3.1.1.36. Student List 51

3.1.1.37. Student Assessment 52

3.1.1.38. Student Critique 53

3.1.1.39. Making Announcement 54

3.1.1.40. Update Course 55

3.1.1.41. Student Details 56

3.2. Functions 57

3.2.1. Login 57

3.2.1.1. Functional Requirements-1 57

3.2.1.2. Functional Requirements-2 57

3.2.1.3. Functional Requirements-3 57

3.2.2. Forgot Password 58

3.2.2.1. Functional Requirements-1 58

3.2.2.2. Functional Requirements-2 58

3.2.2.3. Functional Requirements-3 58

3.2.3. Change Password 59

3.2.3.1. Functional Requirements-1 59

3.2.3.2. Functional Requirements-2 59

3.2.3.3. Functional Requirements-3 59

3.2.4. Making Course Registration 60

3.2.4.1. Functional Requirements-1 60

3.2.4.2. Functional Requirements-2 60

3.2.4.3. Functional Requirements-3 60

3.2.5. Add Course 61

3.2.5.1. Functional Requirements-1 61

3.2.5.2. Functional Requirements-2 61

3.2.5.3. Functional Requirements-3 61

3.2.6. Delete Course 62

3.2.6.1. Functional Requirements-1 62

3.2.6.2. Functional Requirements-2 62

3.2.6.3. Functional Requirements-3 62

3.2.7. Attending Class 63

3.2.7.1. Functional Requirements-1 63

3.2.7.2. Functional Requirements-2 63

3.2.7.3. Functional Requirements-3 63

3.2.8. See Assignment Details 64

3.2.8.1. Functional Requirements-1 64

3.2.8.2. Functional Requirements-2 64

3.2.8.3. Functional Requirements-3 64

3.2.9. Submit Assignment 65

3.2.9.1. Functional Requirements-1 65

3.2.9.2. Functional Requirements-2 65

3.2.9.3. Functional Requirements-3 65

3.2.10. Contacting with Course Instructor 66

3.2.10.1. Functional Requirements-1 66

3.2.10.2. Functional Requirements-2 66

3.2.10.3. Functional Requirements-3 66

3.2.11. Viewing Course Documents 67

3.2.11.1. Functional Requirements-1 67

3.2.11.2. Functional Requirements-2 67

3.2.11.3. Functional Requirements-3 67

3.2.12. Downloading Course Documents 68

3.2.12.1. Functional Requirements-1 68

3.2.12.2. Functional Requirements-2 68

3.2.12.3. Functional Requirements-3 68

3.2.13. Attending Exam 69

3.2.13.1. Functional Requirements-1 69

3.2.13.2. Functional Requirements-2 69

3.2.13.3. Functional Requirements-3 69

3.2.14. In Exam Giving Answers 70

3.2.14.1. Functional Requirements-1 70

3.2.14.2. Functional Requirements-2 70

3.2.14.3. Functional Requirements-3 70

3.2.15. In Exam Getting Previous Question 71

3.2.15.1. Functional Requirements-1 71

3.2.15.2. Functional Requirements-2 71

3.2.15.3. Functional Requirements-3 71

3.2.16. In Exam Getting Next Question 72

3.2.16.1. Functional Requirements-1 72

3.2.16.2. Functional Requirements-2 72

3.2.16.3. Functional Requirements-3 72

3.2.17. Finishing Exam 73

3.2.17.1. Functional Requirements-1 73

3.2.17.2. Functional Requirements-2 73

3.2.17.3. Functional Requirements-3 73

3.2.18. Viewing Exam Result 74

3.2.18.1. Functional Requirements-1 74

3.2.18.2. Functional Requirements-2 74

3.2.18.3. Functional Requirements-3 74

3.2.19. Viewing Note List 75

3.2.19.1. Functional Requirements-1 75

3.2.19.2. Functional Requirements-2 75

3.2.19.3. Functional Requirements-3 75

3.2.20. Adding New Note 76

3.2.20.1. Functional Requirements-1 76

3.2.20.2. Functional Requirements-2 76

3.2.20.3. Functional Requirements-3 76

3.2.21. Updating Existing Note 77

3.2.21.1. Functional Requirements-1 77

3.2.21.2. Functional Requirements-2 77

3.2.21.3. Functional Requirements-3 77

3.2.22. Deleting Existing Note 78

3.2.22.1. Functional Requirements-1 78

3.2.22.2. Functional Requirements-2 78

3.2.22.3. Functional Requirements-3 78

3.2.23. Viewing Surveys 79

3.2.23.1. Functional Requirements-1 79

3.2.23.2. Functional Requirements-2 79

3.2.23.3. Functional Requirements-3 79

3.2.24. Attending Surveys 80

3.2.24.1. Functional Requirements-1 80

3.2.24.2. Functional Requirements-2 80

3.2.24.3. Functional Requirements-3 80

3.2.25. Filling Out Surveys 81

3.2.25.1. Functional Requirements-1 81

3.2.25.2. Functional Requirements-2 81

3.2.25.3. Functional Requirements-3 81

3.2.26. Viewing Communities 82

3.2.26.1. Functional Requirements-1 82

3.2.26.2. Functional Requirements-2 82

3.2.26.3. Functional Requirements-3 82

3.2.27. Joining Communities 83

3.2.27.1. Functional Requirements-1 83

3.2.27.2. Functional Requirements-2 83

3.2.27.3. Functional Requirements-3 83

3.2.28. Viewing Communities Messages 84

3.2.28.1. Functional Requirements-1 84

3.2.28.2. Functional Requirements-2 84

3.2.28.3. Functional Requirements-3 84

3.2.29. Leaving Communities 85

3.2.29.1. Functional Requirements-1 85

3.2.29.2. Functional Requirements-2 85

3.2.29.3. Functional Requirements-3 85

3.2.30. Changing Personal Informations 86

3.2.30.1. Functional Requirements-1 86

3.2.30.2. Functional Requirements-2 86

3.2.30.3. Functional Requirements-3 86

3.2.31. Downloading Educational Informations 87

3.2.31.1. Functional Requirements-1 87

3.2.31.2. Functional Requirements-2 87

3.2.31.3. Functional Requirements-3 87

3.2.32. Viewing Notifications 88

3.2.32.1. Functional Requirements-1 88

3.2.32.2. Functional Requirements-2 88

3.2.32.3. Functional Requirements-3 88

3.2.33. Entering Excuse Report 89

3.2.33.1. Functional Requirements-1 89

3.2.33.2. Functional Requirements-2 89

3.2.33.3. Functional Requirements-3 89

3.2.34. Entering Permission Documents 90

3.2.34.1. Functional Requirements-1 90

3.2.34.2. Functional Requirements-2 90

3.2.34.3. Functional Requirements-3 90

3.2.35. Viewing Their Courses 91

3.2.35.1. Functional Requirements-1 91

3.2.35.2. Functional Requirements-2 91

3.2.35.3. Functional Requirements-3 91

3.2.36. Creating Course 92

3.2.36.1. Functional Requirements-1 92

3.2.36.2. Functional Requirements-2 92

3.2.36.3. Functional Requirements-3 92

3.2.37. Updating Course 93

3.2.37.1. Functional Requirements-1 93

3.2.37.2. Functional Requirements-2 93

3.2.37.3. Functional Requirements-3 93

3.2.38. Creating Online Class 94

3.2.38.1. Functional Requirements-1 94

3.2.38.2. Functional Requirements-2 94

3.2.38.3. Functional Requirements-3 94

3.2.39. Publishing Assignments 95

3.2.39.1. Functional Requirements-1 95

3.2.39.2. Functional Requirements-2 95

3.2.39.3. Functional Requirements-3 95

3.2.40. Viewing Submitted Assignments 96

3.2.40.1. Functional Requirements-1 96

3.2.40.2. Functional Requirements-2 96

3.2.40.3. Functional Requirements-3 96

3.2.41. Evaluating Submitted Assignments 97

3.2.41.1. Functional Requirements-1 97

3.2.41.2. Functional Requirements-2 97

3.2.41.3. Functional Requirements-3 97

3.2.42. Creating Survey 98

3.2.42.1. Functional Requirements-1 98

3.2.42.2. Functional Requirements-2 98

3.2.42.3. Functional Requirements-3 98

3.2.43. Viewing Result of Survey 99

3.2.43.1. Functional Requirements-1 99

3.2.43.2. Functional Requirements-2 99

3.2.43.3. Functional Requirements-3 99

3.2.44. Creating Exam 100

3.2.44.1. Functional Requirements-1 100

3.2.44.2. Functional Requirements-2 100

3.2.44.3. Functional Requirements-3 100

3.2.45. Updating Exam 101

3.2.45.1. Functional Requirements-1 101

3.2.45.2. Functional Requirements-2 101

3.2.45.3. Functional Requirements-3 101

3.2.46. Uploading Notes 102

3.2.46.1. Functional Requirements-1 102

3.2.46.2. Functional Requirements-2 102

3.2.46.3. Functional Requirements-3 102

3.2.47. Viewing Student List 103

3.2.47.1. Functional Requirements-1 103

3.2.47.2. Functional Requirements-2 103

3.2.47.3. Functional Requirements-3 103

3.2.48. Making Student Assessment 104

3.2.48.1. Functional Requirements-1 104

3.2.48.2. Functional Requirements-2 104

3.2.48.3. Functional Requirements-3 104

3.2.49. Making Student Critique 105

3.2.49.1. Functional Requirements-1 105

3.2.49.2. Functional Requirements-2 105

3.2.49.3. Functional Requirements-3 105

3.2.50. Viewing Student Contact Informations 106

3.2.50.1. Functional Requirements-1 106

3.2.50.2. Functional Requirements-2 106

3.2.50.3. Functional Requirements-3 106

3.2.51. Sending Announcement Notification 107

3.2.51.1. Functional Requirements-1 107

3.2.51.2. Functional Requirements-2 107

3.2.51.3. Functional Requirements-3 107

3.3. Performance Requirements 108

3.4. Logical Database Requirements 108

3.5. Design Constraints 110

3.6. Software System Attributes 110

3.6.1. Reliability 110

3.6.2. Availability 110

3.6.3. Security 110

3.6.4. Maintainability 110

3.6.5. Portability 110

4. Appendices 111

4.1. Appendix A 111

4.1.1.Data Dictionary 111

4.1.2.Entity Relationship Diagram 116

4.2. Appendix B 117

4.2.1.Process Models of the Designed To-be System 117

4.2.1.1. Add Course 117

4.2.1.2. Course Registration 117

4.2.1.2.1. Check Added Course Quota 117

4.2.1.3. Add Question To Exam 118

4.2.1.4. Create Exam 118

4.2.1.5. Taking Exam 119

4.2.1.6. Survey Filling Out 119

4.3. Appendix C 120 4.3.1 Class Diagrams 120 4.3.1.1 Login Class Diagram 120

4.3.1.2 System Class Diagram 121

5. Lessons Learned Report 122

**Revision History**

| **Name** | **Date** | **Reason For Changes** | **Version** |
| --- | --- | --- | --- |
| Emre Hancı | 01/04/2022 | Initial Version of Document | v0.1 |
| Emre Hancı | 04/04/2022 | Filled out 1.x and 2.x | v0.2 |
| Emre Hancı | 28/04/2022 | Filled out 3.x (Except 3.4) | v0.3 |
| Eren Kumru | 07/05/2022 | Detailed overall review,  Elaborate on and increase the detail level of all contents where possible (Except 3.1, 3.2, 3.3),  Add references,  Complete check and fix for typos, spelling, grammar, numbering and formatting problems | v0.4 |
| Mücahit Veli Cumart | 09/05/2022 | Filled out 3.1.x source input and destination output table entries. | v0.5 |
| Eren Kumru | 09/05/2022 | Filled out 3.4 and added Data Dictionary | v0.6 |
| Mehmet Yıldız | 09/05/2022 | Update/add some field in 1.x, 2.x, 3.x contents,  Re-write 2.1.5 content,  Corrected/added typographical errors, formatting problems, title hierarchy and emphasis-toning required fields in the document,  Fixed *Table of Contents* table. | v0.7 |
| Mehmet Yıldız | 10/05/2022 | Add Login and System(partial) class diagrams. | v0.8 |
| Muhammed Fidan | 11/05/2022 | Re-write some contents of 3.x,  Corrected errors on the document. | v0.8.1 |
| Ankara Unity Team | 12/05/2022 | Complete review of the document, Lessons Learned Report filled | v1.0 |

1. Introduction

This Software Requirements Specifications document explains what the **University Information System 4.0** is, what it does, and how it should work. Then, the document goes into detail about the environment the system is to be used in, functions, constraints, assumptions, and requirements both on the system and user level. In addition, the document outlines how the prototype will work and how it will depict a system representation.

## Purpose

This document is written for a wide range of audiences, from casual readers to subject experts. Its goal is to provide a clearer picture of the project's system and why it should be implemented. The reader will next have a technical grasp of how each subsystem interacts with the others.

## Scope

The **University Information System 4.0** aims to provide a more easy-to-use education portal and establish more effective communication between course instructors and students.

The System’s:

**First Release** aims to provide an efficient and effective way to process hybrid lectures. Therefore, the first release provides course registration, adding and deleting courses, attending classes, seeing assignment details, submitting assignments, contacting course instructors, and accessing course documents for students. For course instructors, the system also provides posting course information, online lectures, giving assignments, and checking them.

**Second Release** aims to increase the effectiveness of the system by adding more communication-based features. Therefore, the second release plans to provide taking exams, seeing exam results, note-taking, filling out surveys, and community management for students. For course instructors, the system also provides creating course surveys, making and concluding exams, and adding notes and student assessments.

**Third Release** aims to increase the effectiveness of the system that was already established by the first and second releases by getting into administrative features. Therefore, the third release plans to provide changing and downloading personal documents, communicating with in-class students, getting notifications about the school, and entering excuse reports and permission documents for students. For course instructors, the system also provides critiquing students, editing curriculum, accessing student contact information, and making announcements.

## Definitions, Acronyms, and Abbreviations

* SRS: Software Requirements Specifications
* Easy-to-use education portal: the portal which establishes more user-friendly screens gets quick responses and does not have uptime issues.
* Quick response: Because of the overcrowding nature of universities, education portals have accessibility problems on registration weeks and that causes students to take classes that they do not intend to due to late responses. The understanding of a quick response is to return the endpoint request as soon as possible to the student. UIS 4.0 assumes the average quick response time as 0.5 seconds. Providing that quick response is not that easy and requires considering some problems. In Turkey, DPIT[1] rules do not allow storing personal information in a system in another country. On the other hand, the global development operations providers that provide that response times, do not have servers in Turkey, therefore university systems have this quick response issue. Our solution to achieve success is, to have microservers, and the one which stores personal information will work on servers in Turkey, and the other ones will be in other countries with anonymous information. Those microservices will interact with each other with the same primary keys.
* UIS 4.0: University Information System 4.0
* DPIT: Data Protection In Turkey
* PDPA: Personal Data Protection Authority
* Establish more effective communication: Most university systems do not provide all the needed features in one portal. Students and instructors always need to combine some other portal to establish and complete a course portal online. UIS 4.0 aims to provide the most important ones in place therefore students and instructors collaborate and manage their education process in one place.
* Course instructor: Academics who are working in the university.
* Student: People who are registered in university for education.
* Hybrid lecture: Covid-19 pandemic shows us the education process can be evaluated online and there is a business opportunity to satisfy those needs.

## References

1. Turkey’s DPIT by PDPA (<https://www.kvkk.gov.tr/Icerik/5389/Data-Protection-in-Turkey>)
2. Microsoft Fluent Design Principles (<https://www.microsoft.com/design/fluent/>)
3. Laws Of UX (<https://lawsofux.com/>)
4. Ant Design (<https://ant.design/docs/spec/introduce>)
5. Building Secure Software: How to Avoid Security Problems the Right Way. Maddison-Wesley Professional
6. What is ‘Authentication’, The Economic Times (<https://economictimes.indiatimes.com/definition/authentication>)
7. Entity Relationship Diagram with Original Dimensions (<https://drive.google.com/file/d/1tTcOf4LepAFn-Tsp7NeiFc42DnZCo8TN/view?usp=sharing>)
8. Business Process Models with original dimensions (<https://drive.google.com/drive/folders/1eZiIZlEFlLYqj7e3R6KI-RiQCWk4tb-L?usp=sharing>)
9. Mock Screens with original dimensions (<https://drive.google.com/drive/folders/1-9lJIqHVzqdxY6JUBvXw-3FJ0u0ntxXG?usp=sharing>)

## Overview

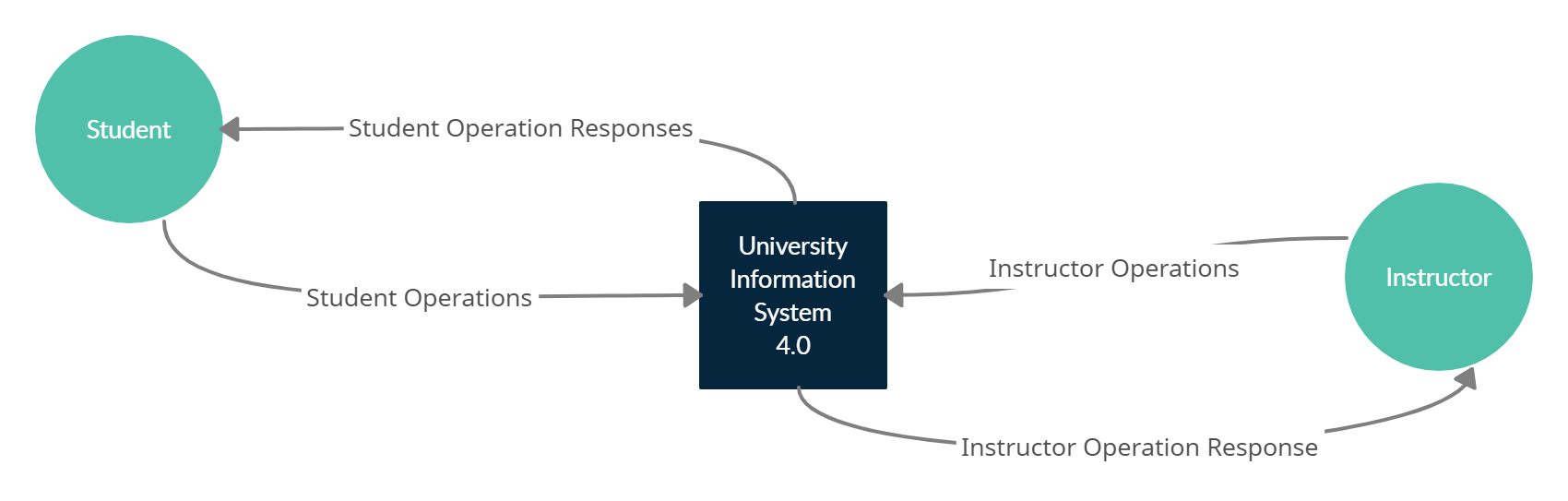
As an overview, the main purpose of **UIS 4.0** is to create a more modern and user-friendly university portal that uses modern technologies as a background.

1. Overall Description

In this section, user, device, software interfaces and other system interfaces, the interaction of main product functions with each other and with the users, assumptions about the user and the system, and system limits will be discussed. Finally, certain criteria that fall outside the current scope of the project will be discussed.

## Product Perspective

The context of this product is that it is a portal for all-inclusive education. The system of this product is then broken down into its subsystems which are lectures, communication, administrative process, and their management.



Context Diagram of System

### 2.1.1. System Interfaces

The application runs on the latest version of Selenium-supported browsers on Windows, Linux, macOS, Android and iOS.

### 2.1.2. User Interfaces

The application GUI provides menus, buttons, inputs, and containers allowing for easy control by a keyboard, a mouse or touch screens interactions.

### 2.1.3. Hardware Interfaces

Not available

### 2.1.4. Software Interfaces

The application allows exporting a list of requirements to MS Excel sheet or PDF format.

The application stores project data in JSON format to enable easy integration with 3rd party applications.

The required software products for using the software is a Web browser.

### 2.1.5. Communication Interfaces

The application consists of a backend with microservice architecture, and user interface layers. These layers provide communication by sharing sensitive data from endpoint points. Therefore, Transmission Control Protocol/Internet Protocol (TCP/IP) will be used in the system. First and foremost, HTTP/HTTPS protocols will be used to transport sensitive data.

In addition, in real-time scenarios such as chatting, it will be realized with the help of real-time microservices with fast communication using Universal Datagram Protocol (UDP).

### 2.1.6. Memory Constraints

Not available

### 2.1.7. Operations

Not available

### 2.1.8. Site Adaptation Requirements

Not available

## Product Functions

The product’s software will be able to perform these major functions:

***For Enrolled Students in the System:***

1. **Make course registration:** They can be registered to the courses that are opened by the course instructors.
2. **Add-Delete course:** They can add or delete courses between add and delete course week dates, which are announced by the university.
3. **Attend classes:** They can sign an attendance document online. After they send a request to sign the attendance, the course instructor needs to approve it.
4. **See assignment details:** They can see assignment details that are assigned by course instructors.
5. **Submit assignments:** They can submit their assignments if course instructors have given an assignment.
6. **Contact course instructors:** They can send and receive messages with course instructors.
7. **Access course documents:** They can access the course documents that are uploaded by course instructors.
8. **Take exams:** They can attend the online exams if course instructors have uploaded any online exams.
9. **See exam results:** They can see the result of either online or paper-based exam results if the course instructor publishes them.
10. **Note-take:** They can take digital notes at any time.
11. **Fill out a survey about the course:** They can fill out surveys if any survey is defined by course instructors.
12. **Community management:** They can see the university community's announcements.
13. **Change/download personal documents:** They can change their personal information and can download documents that are related to their educational life.
14. **Communicate with in-class students:** They can send and receive messages from the other students.
15. **Get notifications about the school:** They can get push notifications sent by the university.
16. **Enter an excuse report and permission document:** They can send their excuse report via online.

***For Enrolled Course Instructors in the System:***

1. **Provide course information:** They can create courses and fill out course information.
2. **Make classes online:** They can make their lectures as online meetings.
3. **Give assignments and check assignments:** They can give assignments and evaluate the assignments.
4. **Create a course survey:** They can create surveys about the course multiple times.
5. **Make and conclude exams:** They can create and evaluate online exams.
6. **Add notes:** They can add notes as digital.
7. **Student assessment:** They can assess the students who are enrolled in their courses.
8. **Make a critique about students:** They can make critiques about students that are enrolled in their classes.
9. **Curriculum editing:** They can make any changes to the curriculum.
10. **Access student contact information:** They can access students' contact information that is enrolled in their courses.
11. **Make an announcement:** They can make announcements about the course to students that are enrolled in their courses.

## User Characteristics

The following, lists out expectations/assumptions about the users of this product:  
***For Enrolled Student:***

* He/She is legally a student of a university that is currently enrolled in the UIS 4.0.
* He/She does not need to be an expert in using a computer but should have basic knowledge.
* He/She should know how to access the UIS 4.0.
* He/She should be aware of the product functions listed above, as they will be happening in real-time and can be used as an official document.

***For Enrolled Course Instructor:***

* He/She is legally a course instructor of a university that is currently enrolled in the UIS 4.0.
* He/She does not need to be an expert in using a computer but should have a basic knowledge.
* He/She should know how to access the UIS 4.0.
* He/She should be aware of the product functions listed above, as they will be happening in real-time and can be used as an official document.

## Constraints

### 2.4.1. Regulatory Policies Since Turkey has protected personal information with official laws all the personal information should be stored as relevant to Turkey’s DPIT laws. The most important thing that requires consideration is storing those kinds of data in servers that are not in Turkey. While storing them outside of Turkey’s boundaries, the information should be anonymous.

### 2.4.2. Hardware Limitations Since the product will be deployed to cloud services, there are a few hardware limitations.

## Since education portals need a quick response at course registration and add-delete course weeks in the year, servers that run the portal’s applications must have high-end hardware. On the other hand, having high properties on servers has a higher cost. The person responsible for server management should increase the server's plans when the busy weeks come, and then should decrease the server plans for budget savings.

### 2.4.3. Interfaces to Other Applications Since UIS 4.0 has integration with universities' current portals, the system should be able to easily integrate with third-party applications.

## The most important thing with integration is to convert the data from third-party systems to the required data format. The system should convert those data into JSON format.

### 2.4.4. Parallel Operation Since there is a limitation on course quota, a student enrolling in a course system should be able to check the remaining quota in order from the first enrolled to the last one.

### 2.4.5. Audit Functions Since in the best case, there should not be any mistakes while taking courses or any other functionality, the system also should be able to audit incoming requests for any complaints that came from users. 2.4.6. Control Functions Since the UIS 4.0 takes input or gets data from integrations, the data should be controlled, as they are defined.

### 

### 2.4.6.1. Input Validity Control While getting input from users, the system should be able to check if it is a valid input. Some of the validity controls can be done in the frontend such as whether the given email is valid or not. Some validity controls can be done in the backend such as given email and password matches for a user. 2.4.6.2. Integrity Control Since the system will be integrated with third-party applications, the system should be able to check its integrity controls. 2.4.7. Higher-order Language Requirements

### Not available 2.4.8. Signal Handshake Protocols Since there is a WebSocket between the frontend and backend the client should be able to confirm themselves with the handshake protocol.

### 2.4.9. Reliability Requirements Not available

### 2.4.10. Criticality of the Application Since business domain requirements there are a few criticality in the system. Those are;

* Sensitive information that is defined by Turkey’s DPIT[1] securities.
* Having a satisfied uptime ratio especially at the course registration and add-delete weeks.

### 2.4.11. Safety and Security Considerations

Since the business domain has some sensitive information, that kind of data should be stored encrypted.

## Assumptions and Dependencies

Since the hardware subsystems of the system will be provided by cloud services, it is assumed that the chosen service does not have any critical problems.

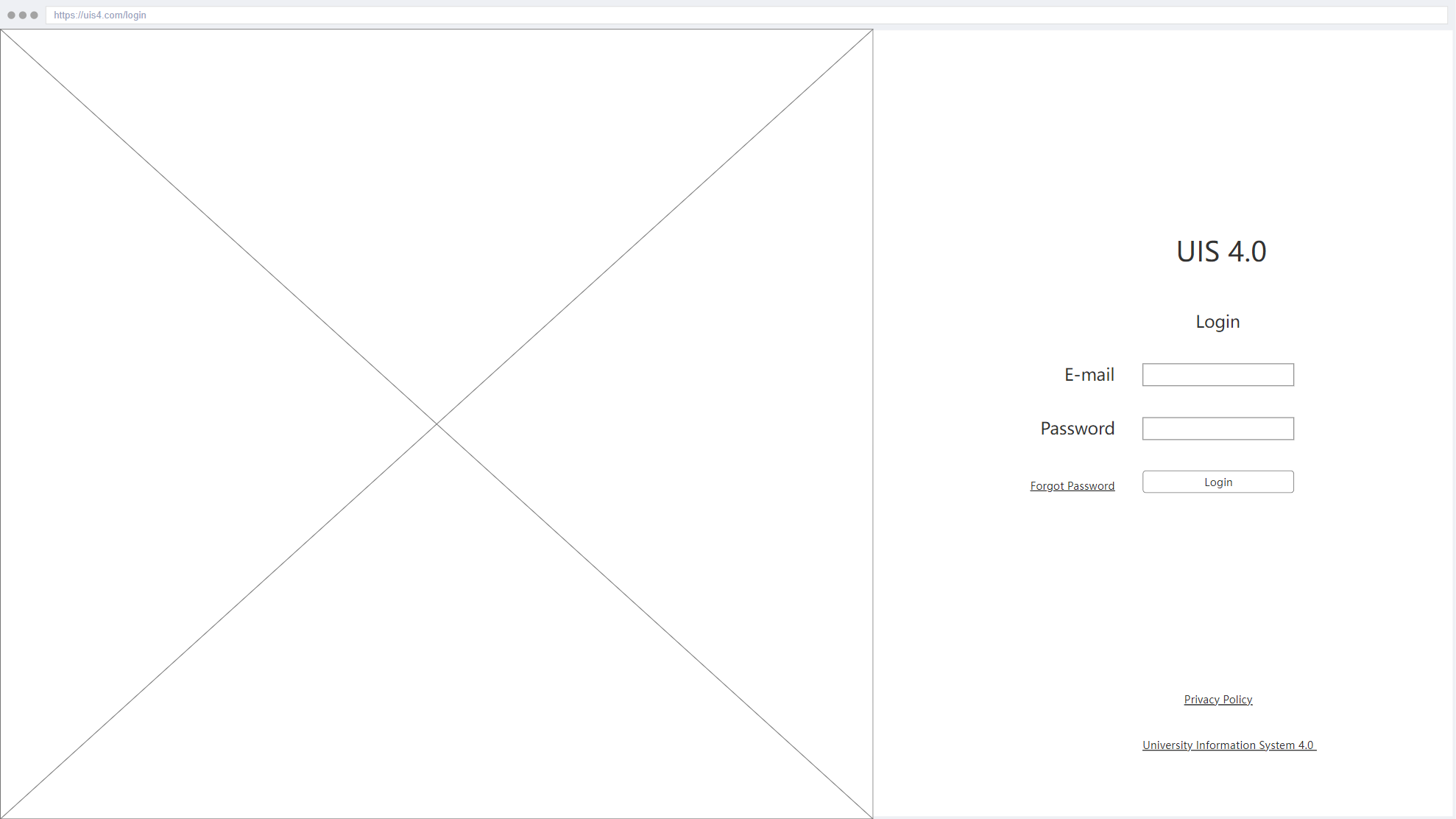
## Apportioning of Requirements

## Requirements that are beyond the scope of this project and will be implemented in future versions are adding a university staff role for not needing to integrate into the university's current information systems, and starting to end the complete education portal product.

3. Specific Requirements

## 3.1. External Interface Requirements

### 3.1.1 User Interfaces[9]



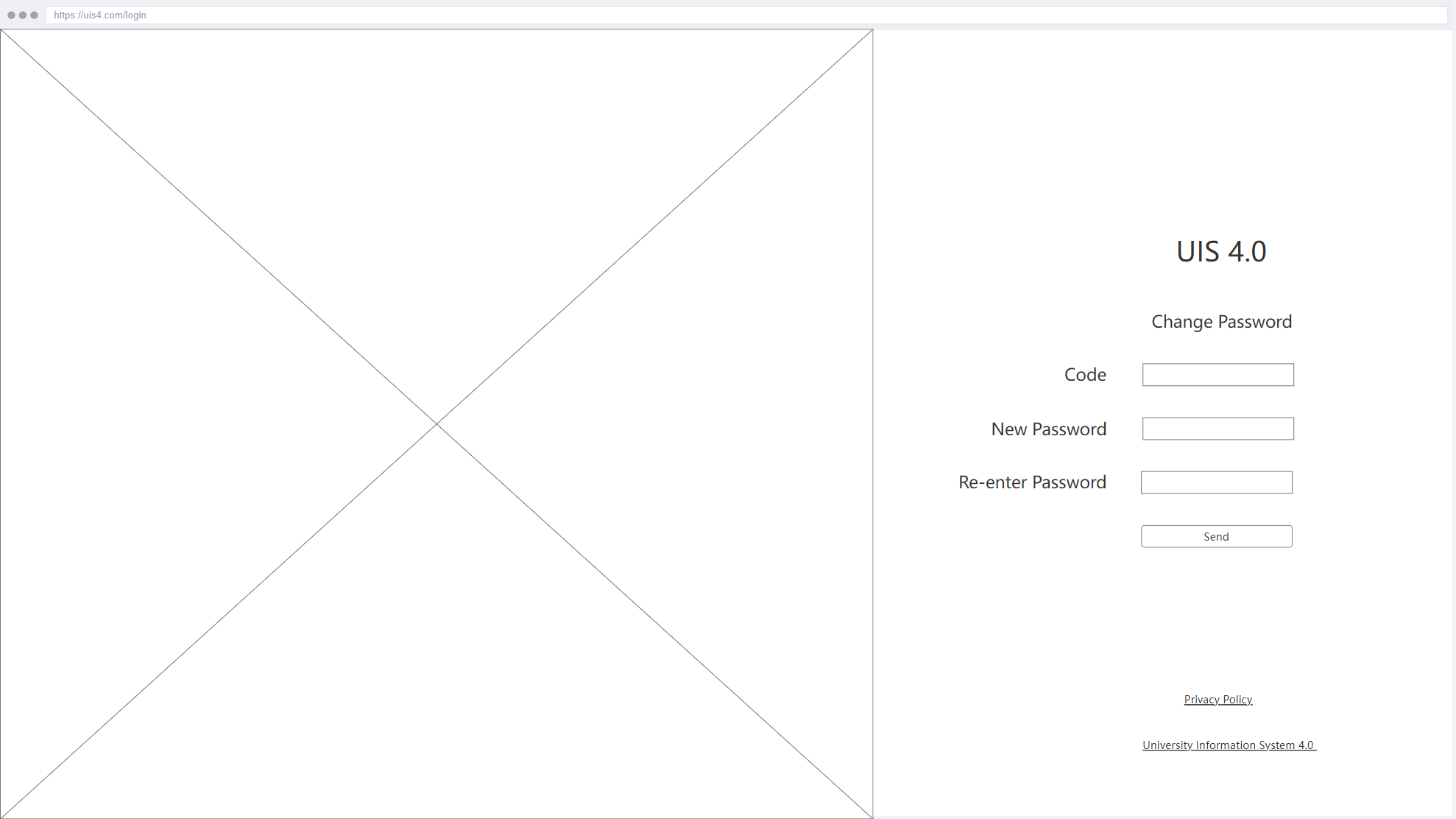
#### 3.1.1.1 Login Page

| **Name of Item** | Login Page |
| --- | --- |
| **Description of Purpose** | Show how to look like login **Login** UI design for Students and Instructors. |
| **Source of Input or Destination of Output** | **Email**, **Password** inputs, **Login** button and **Forgot Password** link, **Privacy Policy** link. |
| **Relationships to Other Inputs / Outputs** | 1. To click **Login** button:    1. if Email and Password is correct, the system will redirect to the homepage according to the user type.    2. If not, the system will give a pop-up error message to the user. 2. To click the **Privacy Policy** link, the system will download the “*Privacy Policy”* document. 3. To click the **Forgot Password**, the system will redirect the *Forgot Password* page. |

#### 3.1.1.2 Forgot Password

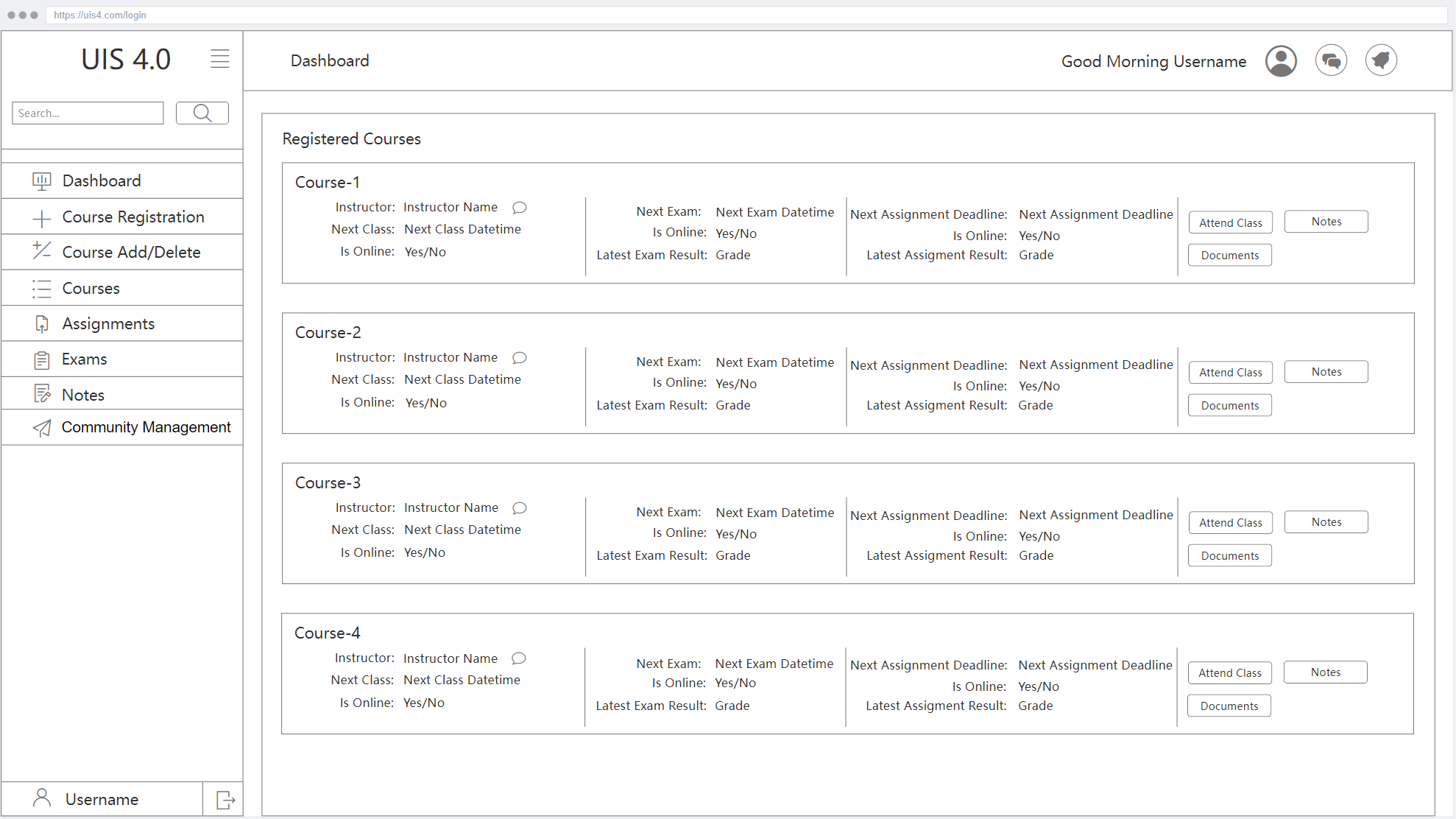
| **Name of Item** | Forgot Password |
| --- | --- |
| **Description of Purpose** | Send code to the user for renewing the password. |
| **Source of Input or Destination of Output** | **Email** input and **Send** button |
| **Relationships to Other Inputs / Outputs** | To click the **Send** button, the system will send a specific code to the user for renewing password. |

#### 3.1.1.3 Change Password

****

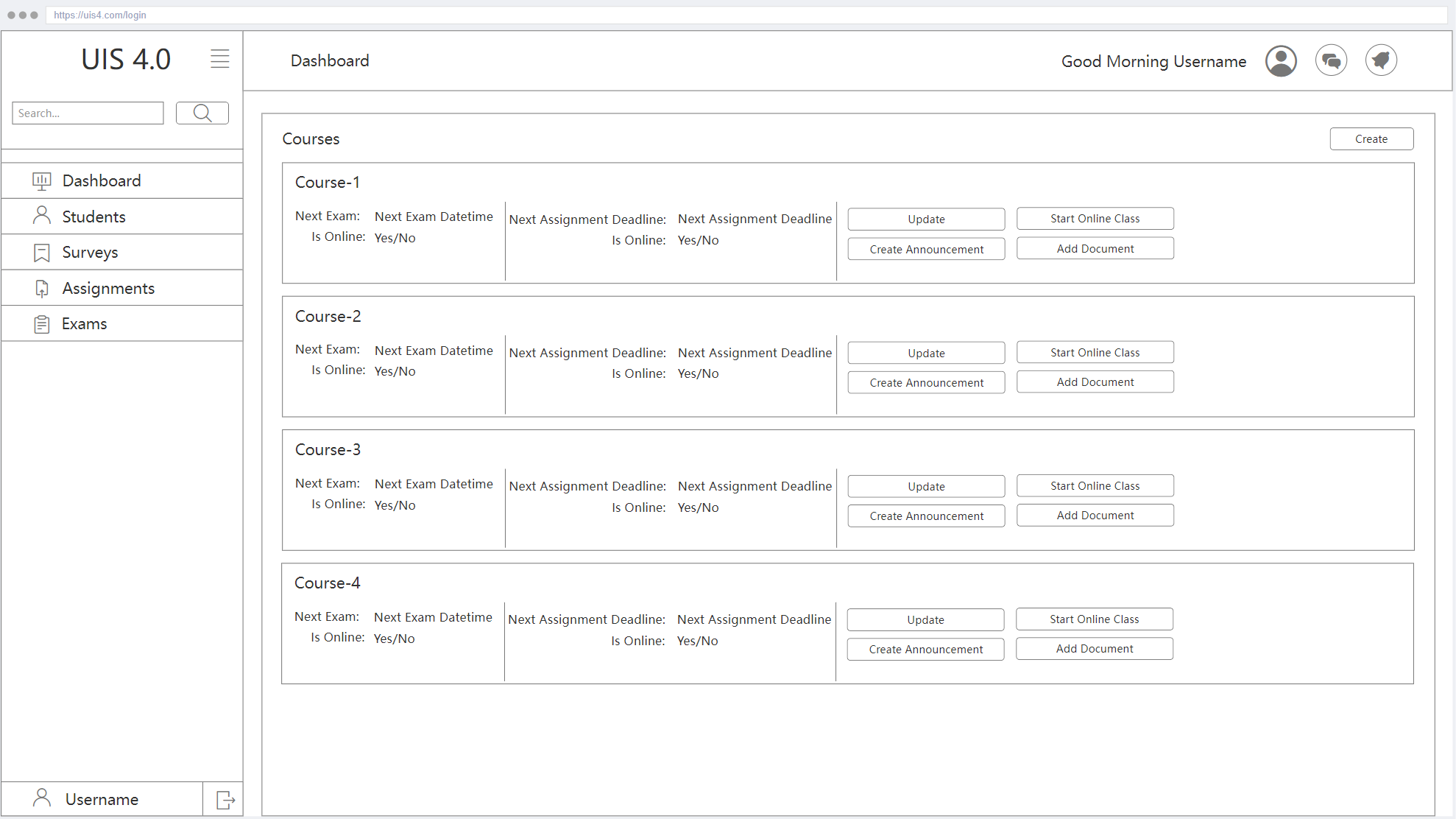
| **Name of Item** | Change Password |
| --- | --- |
| **Description of Purpose** | Change forgotten password. |
| **Source of Input or Destination of Output** | **Code**, **New Password** and **Re–enter Password** inputs, **Send** button |
| **Relationships to Other Inputs / Outputs** | 1. To click the **Send** button,    1. if password inputs are filled correctly and match each other, and code is valid. The system renews the old password with the new password.    2. Otherwise, the system gives a pop-up error message to the user. |

#### 3.1.1.4 Homepage - Student



| **Name of Item** | Homepage - Student |
| --- | --- |
| **Description of Purpose** | Accessing Course Informations |
| **Source of Input or Destination of Output** | **Search-box** input, **Search** button, **Menu** items, Buttons in Registered Courses, **Profile** button, **Messages** button |
| **Relationships to Other Inputs / Outputs** | 1. Typing something in the **Search-box** and clicking the **Search** button returns data equivalent to this search within the system. for example student information, instructor information, course information, etc... 2. When the user presses the buttons on the **Menu** and registered courses, the relevant page is displayed. 3. The page redirects to the profile page when the **Profile** button is pressed. 4. The page redirects to the messages page when the **Messages** button is pressed. |

#### 3.1.1.5 Homepage - Instructor



| **Name of Item** | Homepage - Instructor |
| --- | --- |
| **Description of Purpose** | Accessing Course Information |
| **Source of Input or Destination of Output** | **Search-box** input, **Search** button, **Menu** items, Buttons in courses, **Profile** button, **Messages** button |
| **Relationships to Other Inputs / Outputs** | 1. Typing something in the **Search-box** and clicking the **Search** button returns data equivalent to this search within the system. for example student information, instructor information, course information, etc... 2. When the user presses the buttons on the **Menu** and courses, the relevant page is displayed. 3. The page redirects to the **Profile** page when the profile button is pressed. 4. The page redirects to the messages page when the **Messages** button is pressed. |

#### 3.1.1.6 Course Registration

| **Name of Item** | Course Registration |
| --- | --- |
| **Description of Purpose** | Making course registration |
| **Source of Input or Destination of Output** | Add course button, Delete course button |
| **Relationships to Other Inputs / Outputs** | 1. A request will be sent to the server to add the course, when the Add course button is clicked. 2. A request will be sent to the server to delete the course, when the Delete course button is clicked. |

#### 3.1.1.7 Add-Delete Course

| **Name of Item** | Add-Delete Course |
| --- | --- |
| **Description of Purpose** | Adding or deleting course |
| **Source of Input or Destination of Output** | Add course button, Delete course button |
| **Relationships to Other Inputs / Outputs** | 1. A request will be sent to the server to add the course, when the Add course button is clicked. 2. A request will be sent to the server to delete the course, when the Delete course button is clicked. |

#### 3.1.1.8 Attending Classes

#### 

| **Name of Item** | Attending Classes |
| --- | --- |
| **Description of Purpose** | For sending attending class request |
| **Source of Input or Destination of Output** | Attend Class button |
| **Relationships to Other Inputs / Outputs** | When this button is pressed, the confirmation window appears. If the user approves, a request to join the course is sent. |

#### 3.1.1.9 Assignment List - Student

| **Name of Item** | Assignment List - Student |
| --- | --- |
| **Description of Purpose** | Listing assignments that are registered courses |
| **Source of Input or Destination of Output** | Submit button, Details button |
| **Relationships to Other Inputs / Outputs** | 1. This page shows the details of courses when the Details button is pressed. 2. This page redirects to the submit page when the Submit button is pressed. |

#### 3.1.1.10 Assignment List - Instructor

| **Name of Item** | Assignment List - Instructor |
| --- | --- |
| **Description of Purpose** | For listing submitted assignments |
| **Source of Input or Destination of Output** | Details button |
| **Relationships to Other Inputs / Outputs** | This page shows the details of the student’s assignments when the Details button is pressed. |

#### 3.1.1.11 Assignment Details - Student

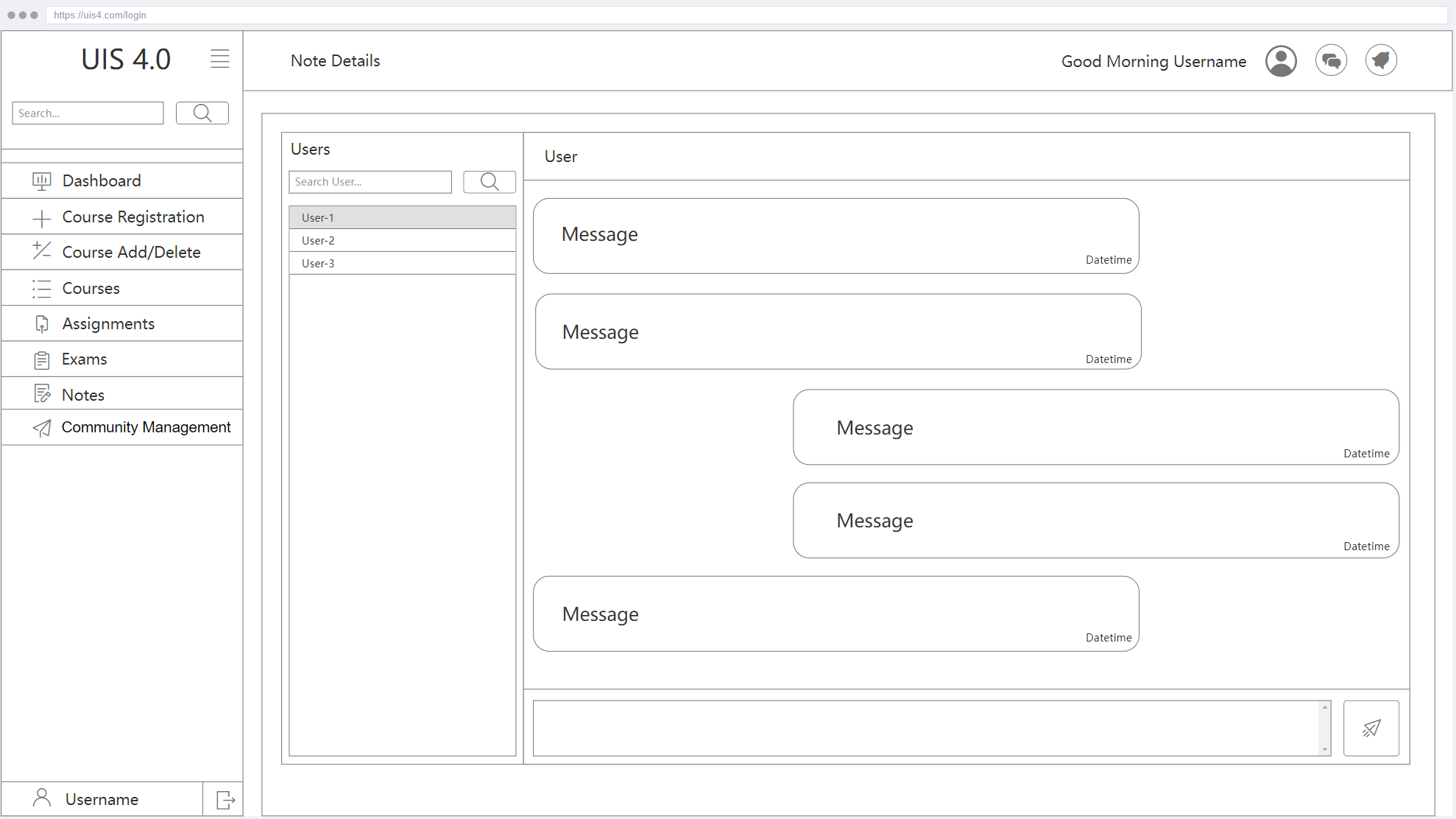
#### 

| **Name of Item** | Assignment Details - Student |
| --- | --- |
| **Description of Purpose** | For viewing details of assignments |
| **Source of Input or Destination of Output** | Download Detail button, Submit button |
| **Relationships to Other Inputs / Outputs** | 1. This page shows the details of the assignment when the download details button is pressed. 2. This page sends a request to the server for the assignment when the submit button is pressed. |

#### 3.1.1.12 Assignment Details - Instructor

| **Name of Item** | Assignment Details - Instructor |
| --- | --- |
| **Description of Purpose** | For evaluating assignment |
| **Source of Input or Destination of Output** | View assignment button, Grade input, Evaluate button |
| **Relationships to Other Inputs / Outputs** | 1. This page shows the assignment when the view assignment button is pressed. 2. The grade input is for the grade of the assignment which is given by the instructor to the student. 3. The Evaluate button is for sending the result of the assignment to the server. |

#### 3.1.1.13 Messages



| **Name of Item** | Messages |
| --- | --- |
| **Description of Purpose** | Viewing messages and sending messages |
| **Source of Input or Destination of Output** | Search input, Message input, Send button |
| **Relationships to Other Inputs / Outputs** | 1. This page displays the entries related to the searched input. 2. The message input is for the message which the user wants to send. 3. This button sends the message to the other user. |

#### 3.1.1.14 Course Documents

#### 

| **Name of Item** | Course Documents |
| --- | --- |
| **Description of Purpose** | Viewing course documents |
| **Source of Input or Destination of Output** | Download button |
| **Relationships to Other Inputs / Outputs** | The download button allows the user to view and download the course document. |

#### 3.1.1.15 Exam List - Student

#### 

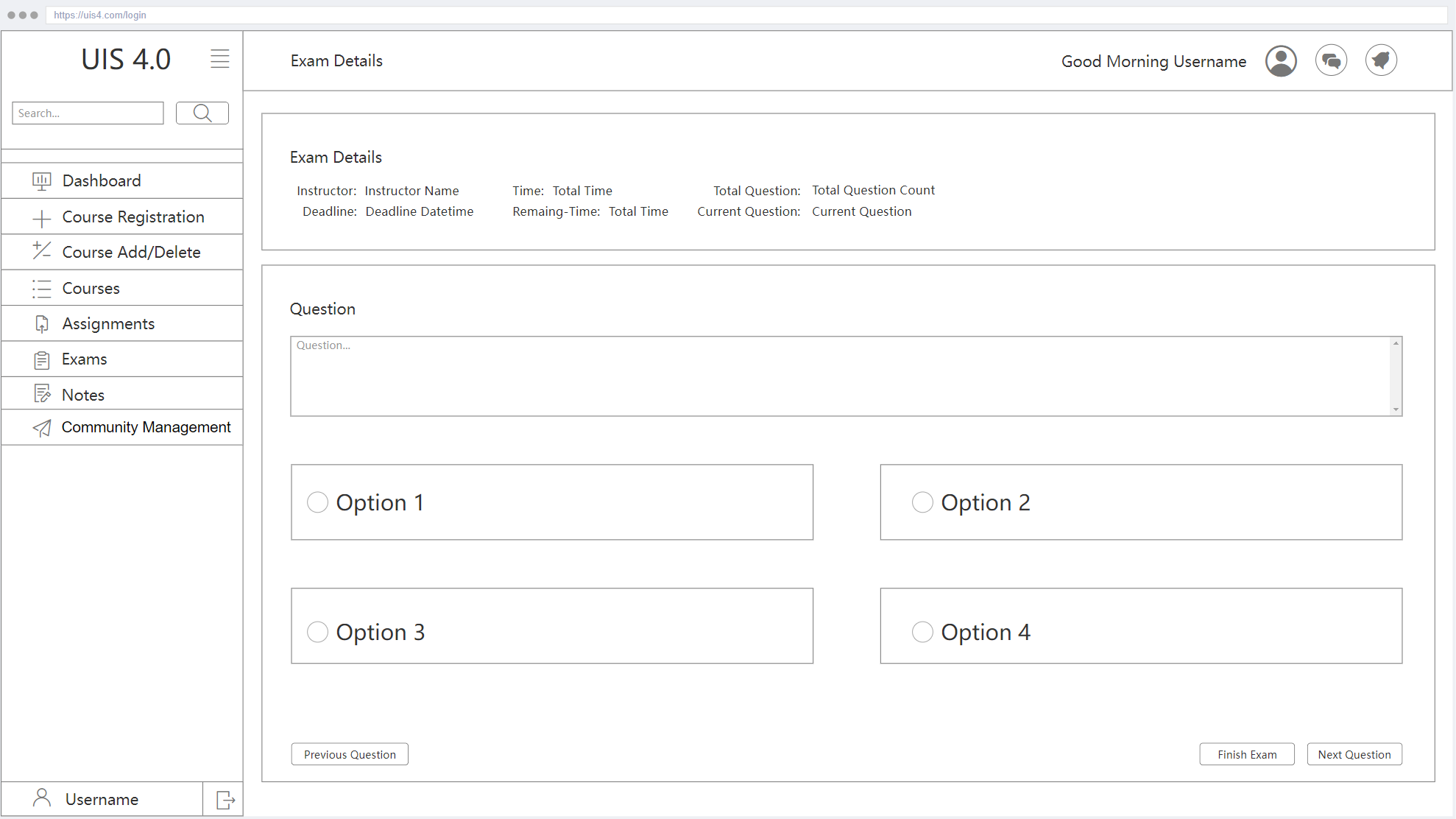
| **Name of Item** | Exam List - Student |
| --- | --- |
| **Description of Purpose** | Viewing Exams in List |
| **Source of Input or Destination of Output** | Start button |
| **Relationships to Other Inputs / Outputs** | The start button is for the user to start the relevant exam. When the start button is pressed, confirmation is requested with a pop-up. If the user confirms the pop-up, the exam page opens. |

#### 3.1.1.16 Exam List - Instructor

#### 

| **Name of Item** | Exam List - Instructor |
| --- | --- |
| **Description of Purpose** | Viewing Exam List |
| **Source of Input or Destination of Output** | Create button, Evaluate button, Update button |
| **Relationships to Other Inputs / Outputs** | 1. The create button allows for creating a new exam. 2. The instructor can evaluate the students' exams with the evaluate button. Exams are opened when the Evaluate button is pressed. 3. The page opens the relevant exam editing page when the instructor clicks the update button |

#### 3.1.1.17 In-Exam



| **Name of Item** | In-Exam |
| --- | --- |
| **Description of Purpose** | Giving Answers to Questions in Exam |
| **Source of Input or Destination of Output** | Options, Previous Question button, Finish Exam button, Next Question button |
| **Relationships to Other Inputs / Outputs** | 1. This page displays the next question of the exam if the next question button is pressed, and the previous question of the exam if the previous question button is pressed. 2. This page sends the user’s exam to the server when the user clicks the finish exam button. |

#### 3.1.1.18 Exam Detail

#### 

| **Name of Item** | Exam Detail |
| --- | --- |
| **Description of Purpose** | Viewing Details of Exam |
| **Source of Input or Destination of Output** | Start exam button |
| **Relationships to Other Inputs / Outputs** | This page redirects the user to a new in-exam page. |

#### 3.1.1.19 Note List

#### 

| **Name of Item** | Note List |
| --- | --- |
| **Description of Purpose** | Viewing List of Notes |
| **Source of Input or Destination of Output** | Update button, Delete button, Add Notes button |
| **Relationships to Other Inputs / Outputs** | 1. This page redirects the user to a new note detail page when the add notes button is clicked. 2. This page redirects the user to a relevant note detail page when the update button is clicked. 3. The delete button deletes the relevant note when clicked. |

#### 3.1.1.20 Note Detail

| **Name of Item** | Note Detail |
| --- | --- |
| **Description of Purpose** | Updating a Note or Create New |
| **Source of Input or Destination of Output** | Note input, Save notes button |
| **Relationships to Other Inputs / Outputs** | This page redirects the user to the note list page when the save notes button is clicked. |

#### 3.1.1.21 Surveys

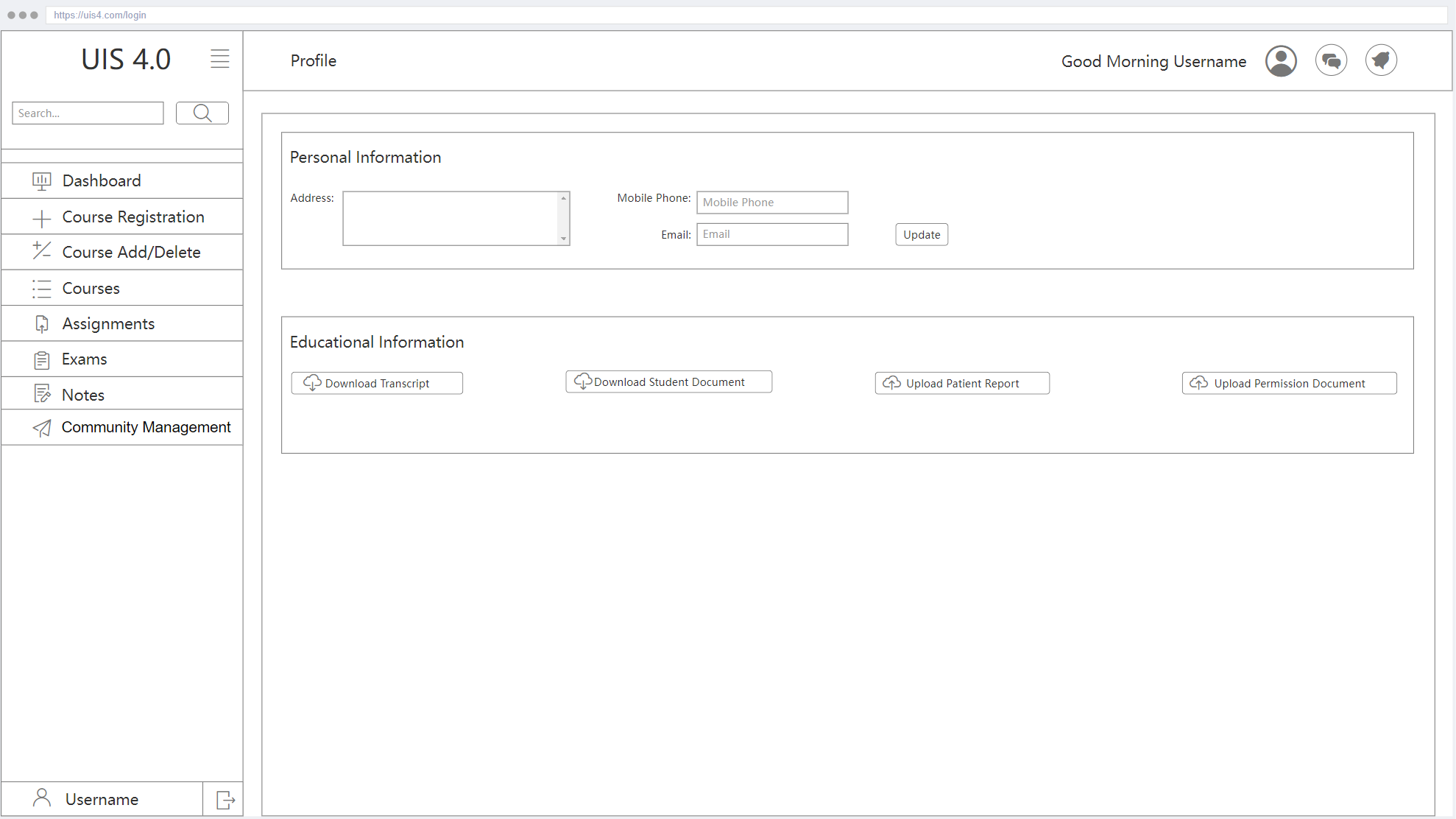
#### 

| **Name of Item** | Surveys |
| --- | --- |
| **Description of Purpose** | Giving Answers to Surveys |
| **Source of Input or Destination of Output** | Question options, Submit Survey button |
| **Relationships to Other Inputs / Outputs** | The submit survey button is for submitting answers of the survey to the server. |

#### 3.1.1.22 Community Management

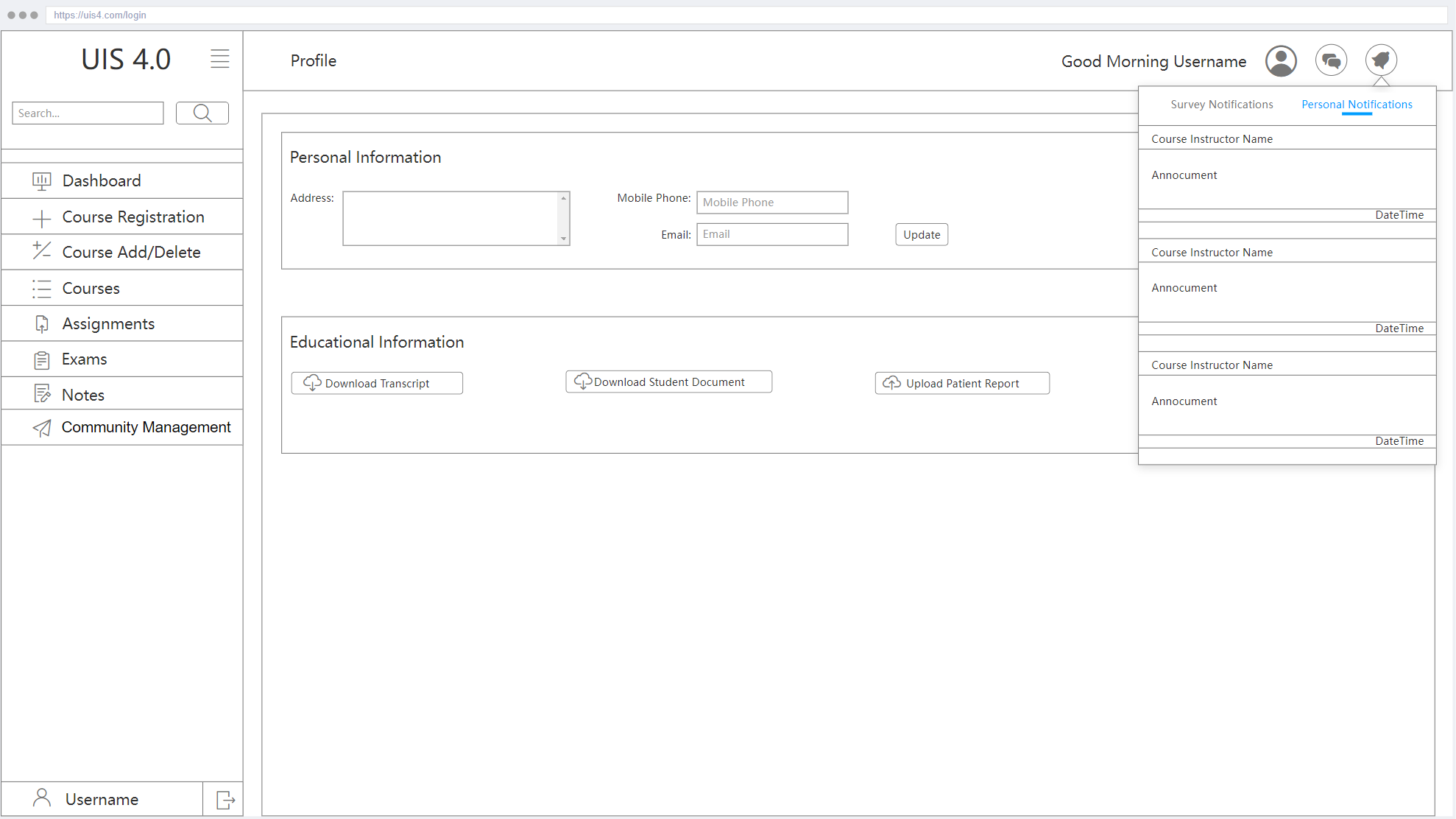
| **Name of Item** | Community Management |
| --- | --- |
| **Description of Purpose** | Viewing Communities |
| **Source of Input or Destination of Output** | Join button, Leave Button |
| **Relationships to Other Inputs / Outputs** | 1. This page displays a confirmation window on the screen. If the user confirms, a request is sent to the server for joining the Community. 2. This page displays a confirmation window on the screen. If the user confirms, a request is sent to the server for leaving the Community. |

#### 3.1.1.23 Profile



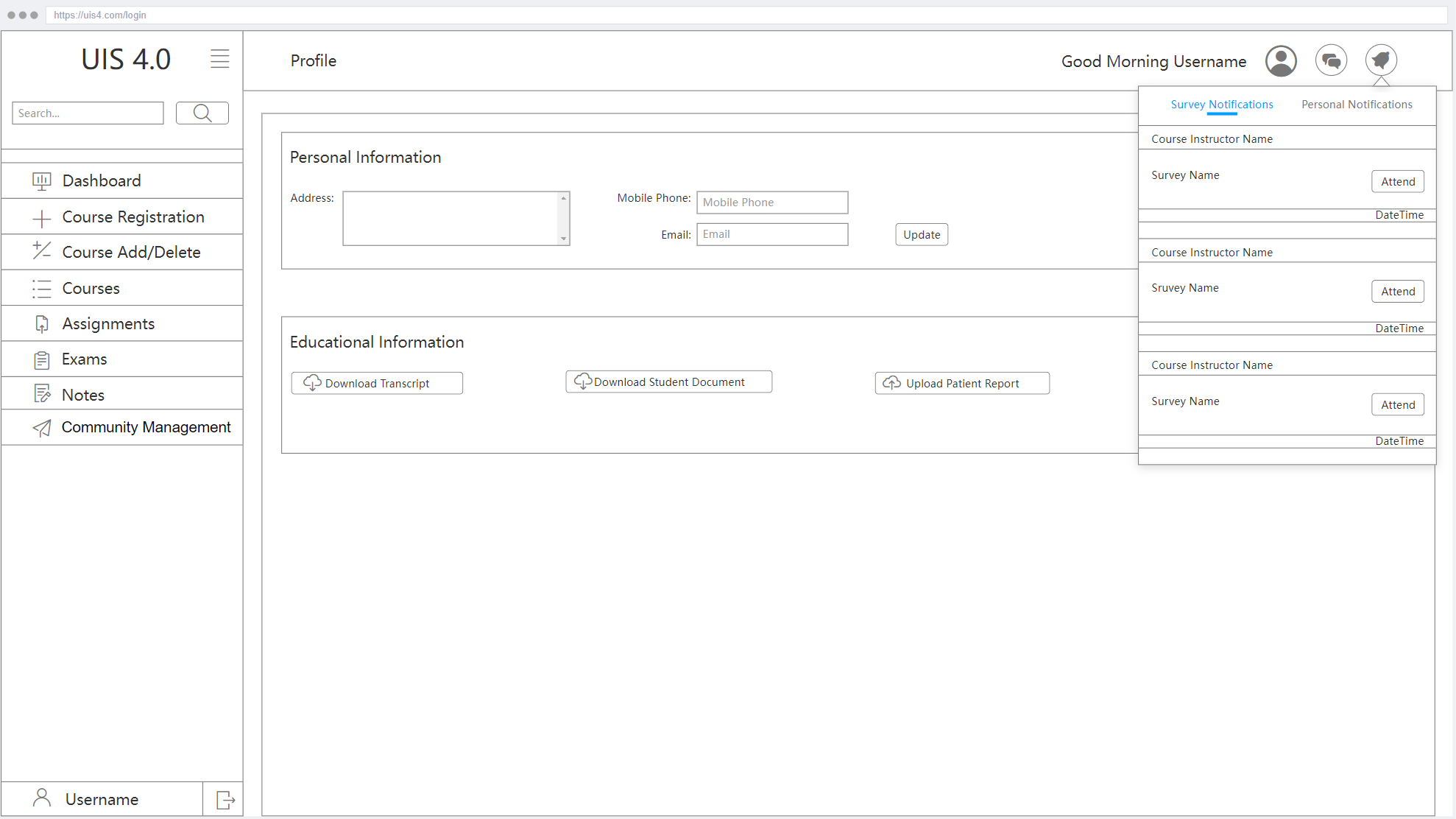
| **Name of Item** | Profile |
| --- | --- |
| **Description of Purpose** | Viewing Personal and Educational Information |
| **Source of Input or Destination of Output** | Personal information inputs, Update button, Download Transcript, Download Student Document, Upload Patient Report, Upload Permission Document |
| **Relationships to Other Inputs / Outputs** | 1. This page updates personal information if there is a change in this information when the user clicks the update button. 2. This page displays related documents when the download or upload ‘document’ buttons are clicked. |

#### 3.1.1.24 Personal Notifications



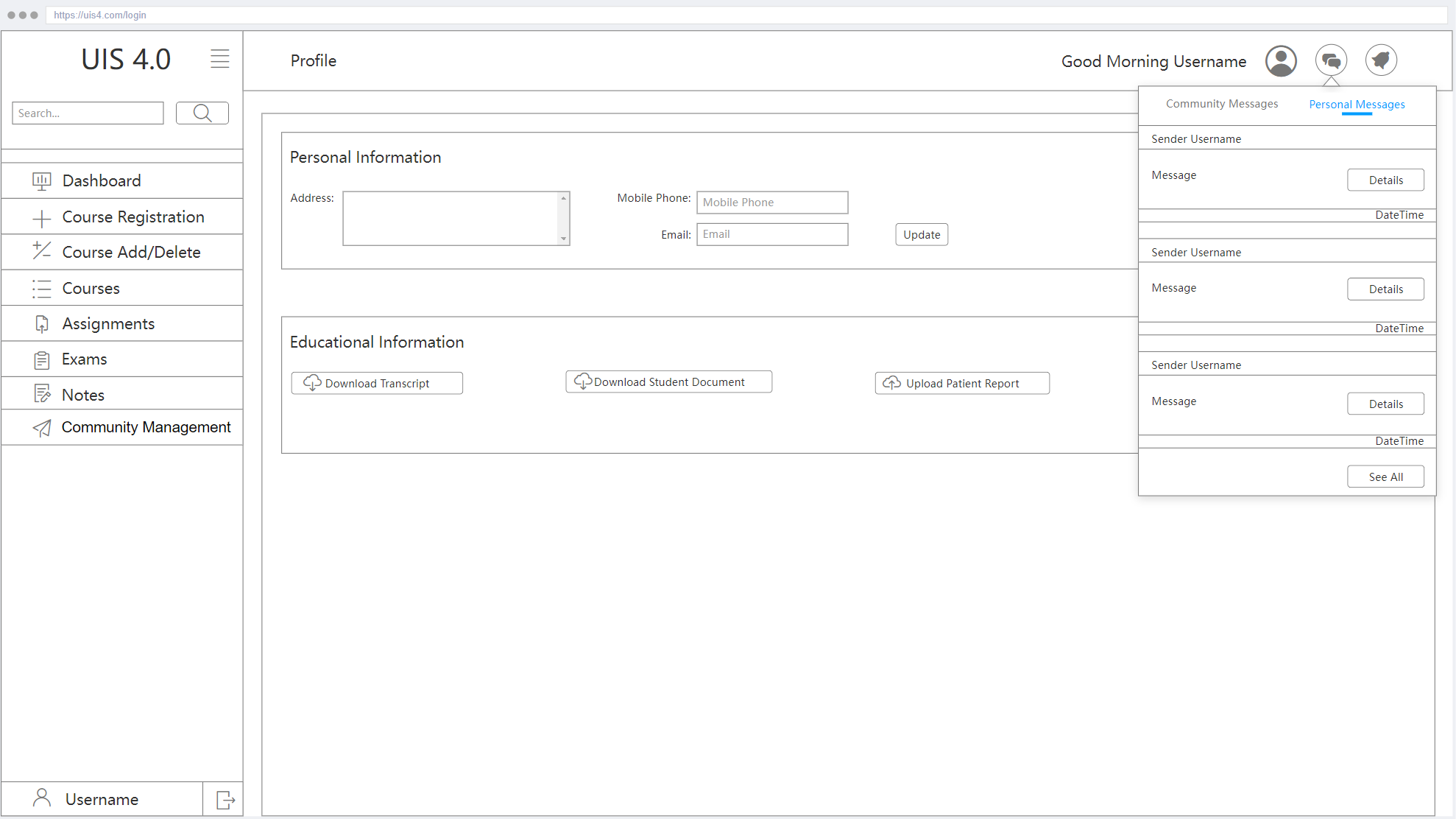
| **Name of Item** | Personal Notifications |
| --- | --- |
| **Description of Purpose** | Viewing Personal Notifications |
| **Source of Input or Destination of Output** | - |
| **Relationships to Other Inputs / Outputs** | - |

#### 3.1.1.25 Survey Notifications



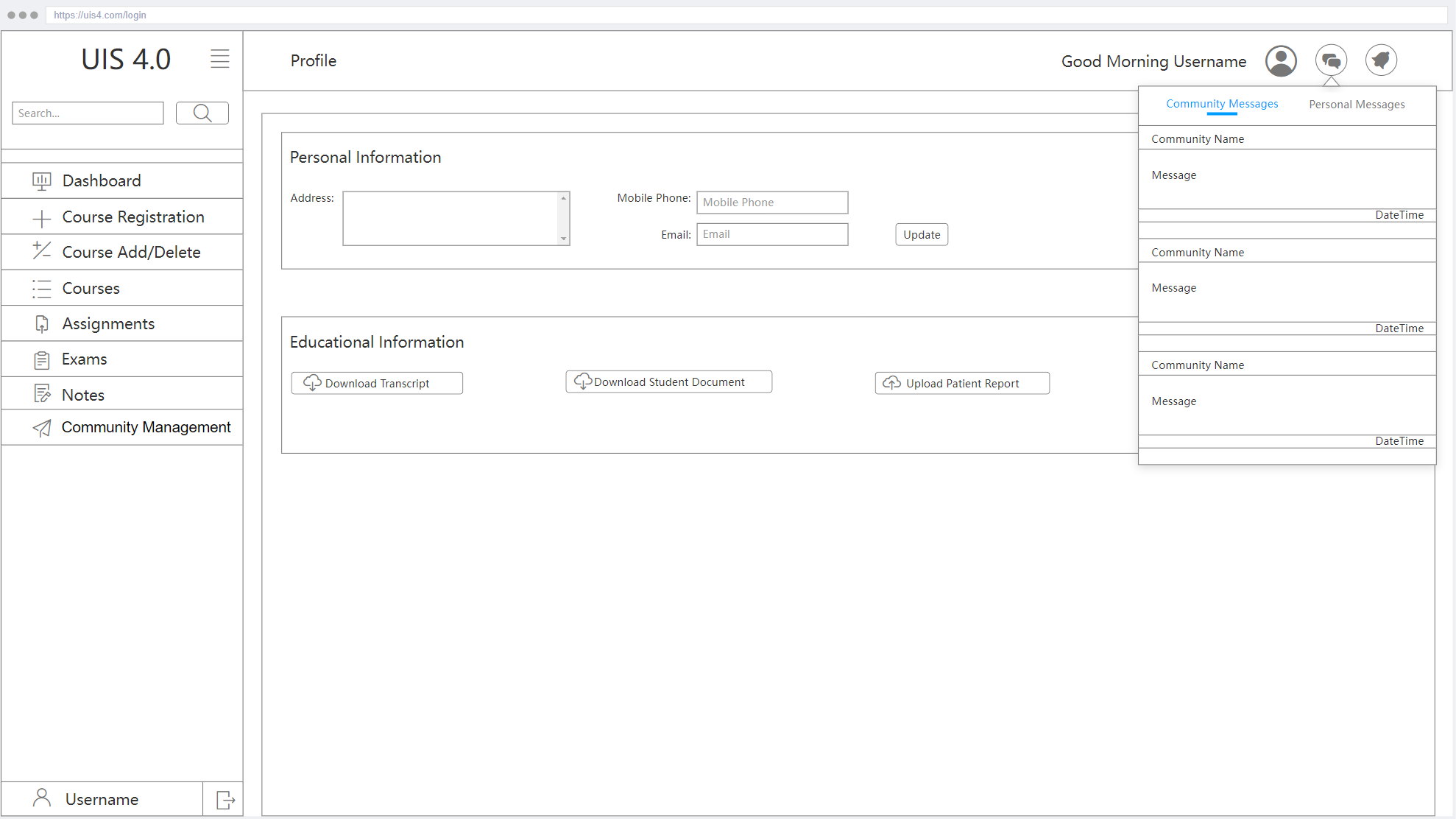
| **Name of Item** | Survey Notifications |
| --- | --- |
| **Description of Purpose** | Viewing Survey Notifications |
| **Source of Input or Destination of Output** | - |
| **Relationships to Other Inputs / Outputs** | - |

#### 3.1.1.26 Personal Messages



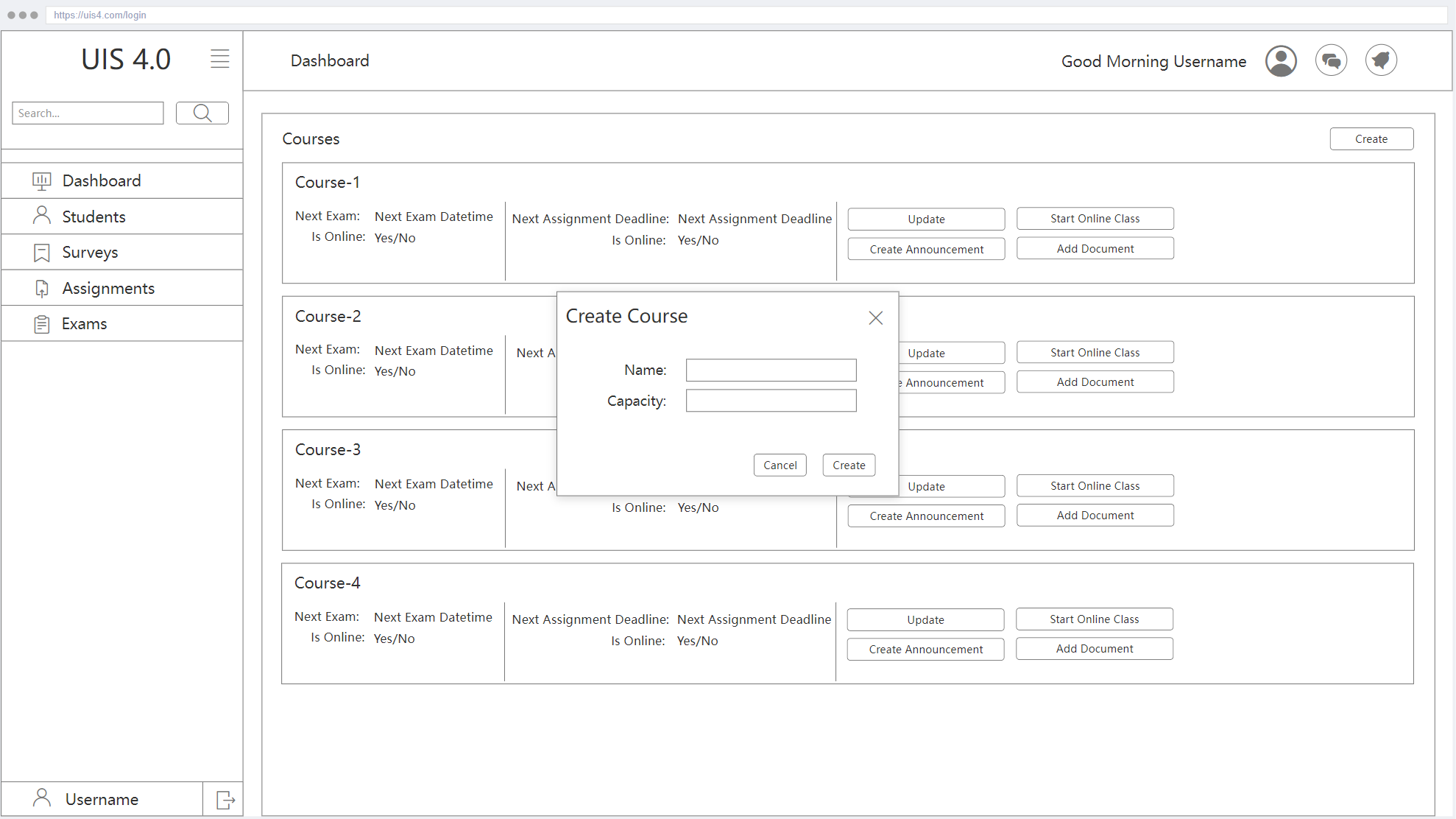
| **Name of Item** | Personal Messages |
| --- | --- |
| **Description of Purpose** | Viewing Personal Messages |
| **Source of Input or Destination of Output** | Details button, See All button |
| **Relationships to Other Inputs / Outputs** | 1. This page displays details of related messages when its details button is clicked. 2. This page redirects the user to a new messages page if the see all button is clicked. |

#### 3.1.1.27 Community Messages



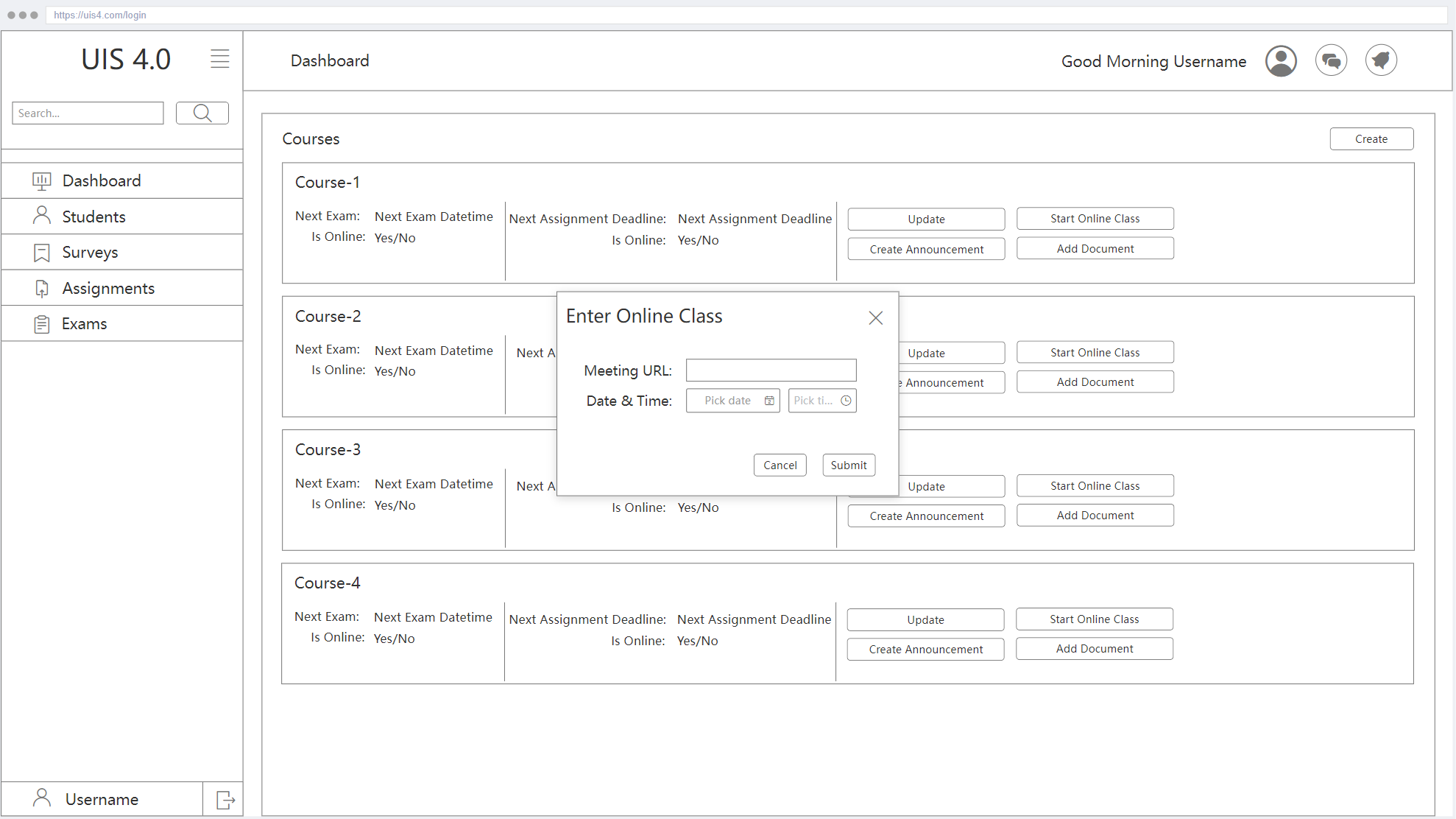
| **Name of Item** | Community Messages |
| --- | --- |
| **Description of Purpose** | Viewing Community Messages |
| **Source of Input or Destination of Output** | - |
| **Relationships to Other Inputs / Outputs** | - |

#### 3.1.1.28 Create Course



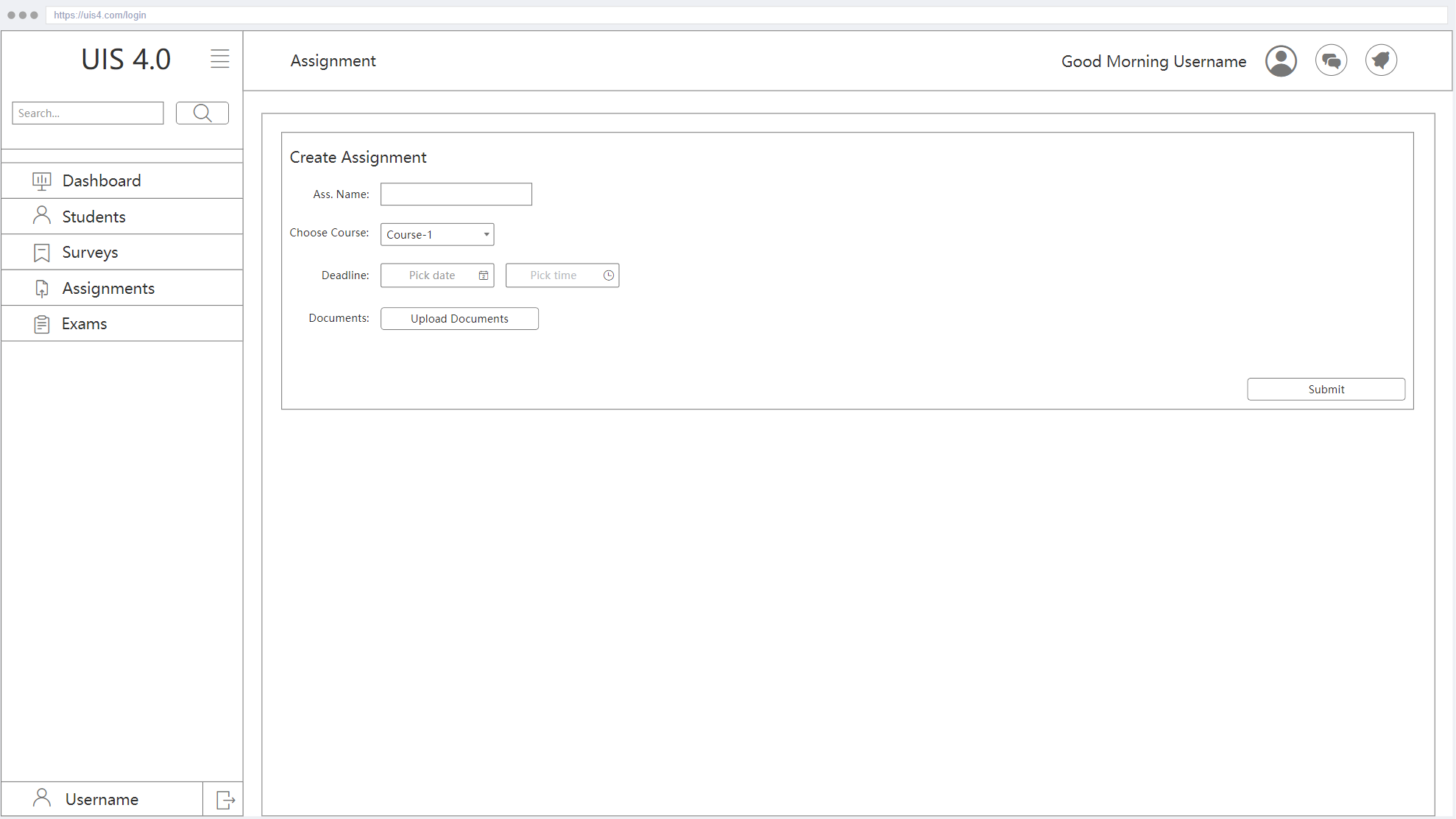
| **Name of Item** | Create Course |
| --- | --- |
| **Description of Purpose** | For Creating New Course |
| **Source of Input or Destination of the Output** | Name input , Capacity input, Create and Cancel buttons |
| **Relationships to Other Inputs / Outputs** | This page sends a request for creating a new course with a given name and capacity when the create button is clicked. |

#### 3.1.1.29 Online Class



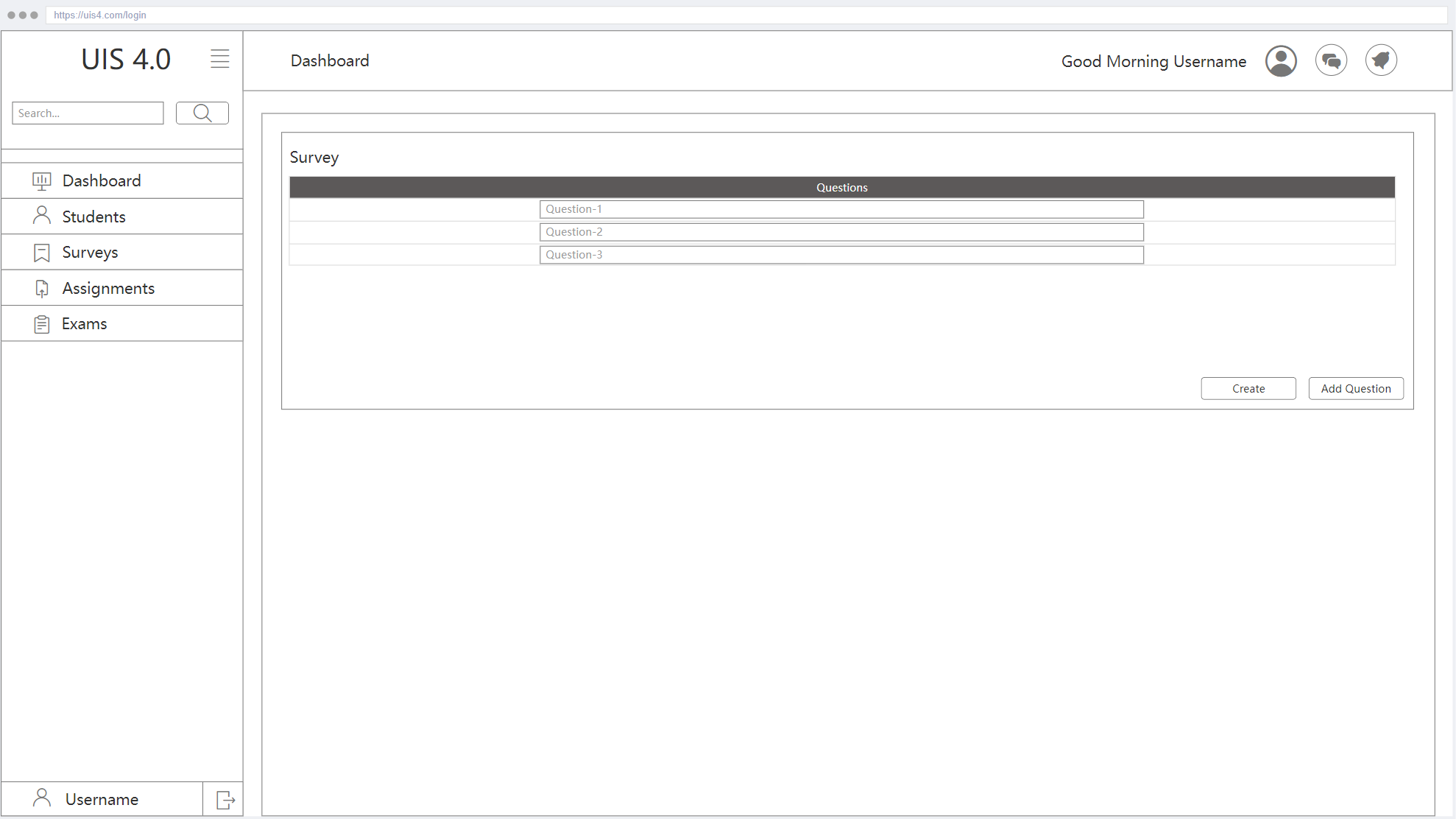
| **Name of Item** | Online Class |
| --- | --- |
| **Description of Purpose** | For Updating an Exist Course |
| **Source of Input or Destination of Output** | Meeting URL input, Date and Time input, Submit and Cancel button |
| **Relationships to Other Inputs / Outputs** | This page updates the online class information of the related course with the given inputs if the user clicks the submit button. |

#### 3.1.1.30 Assignment Create



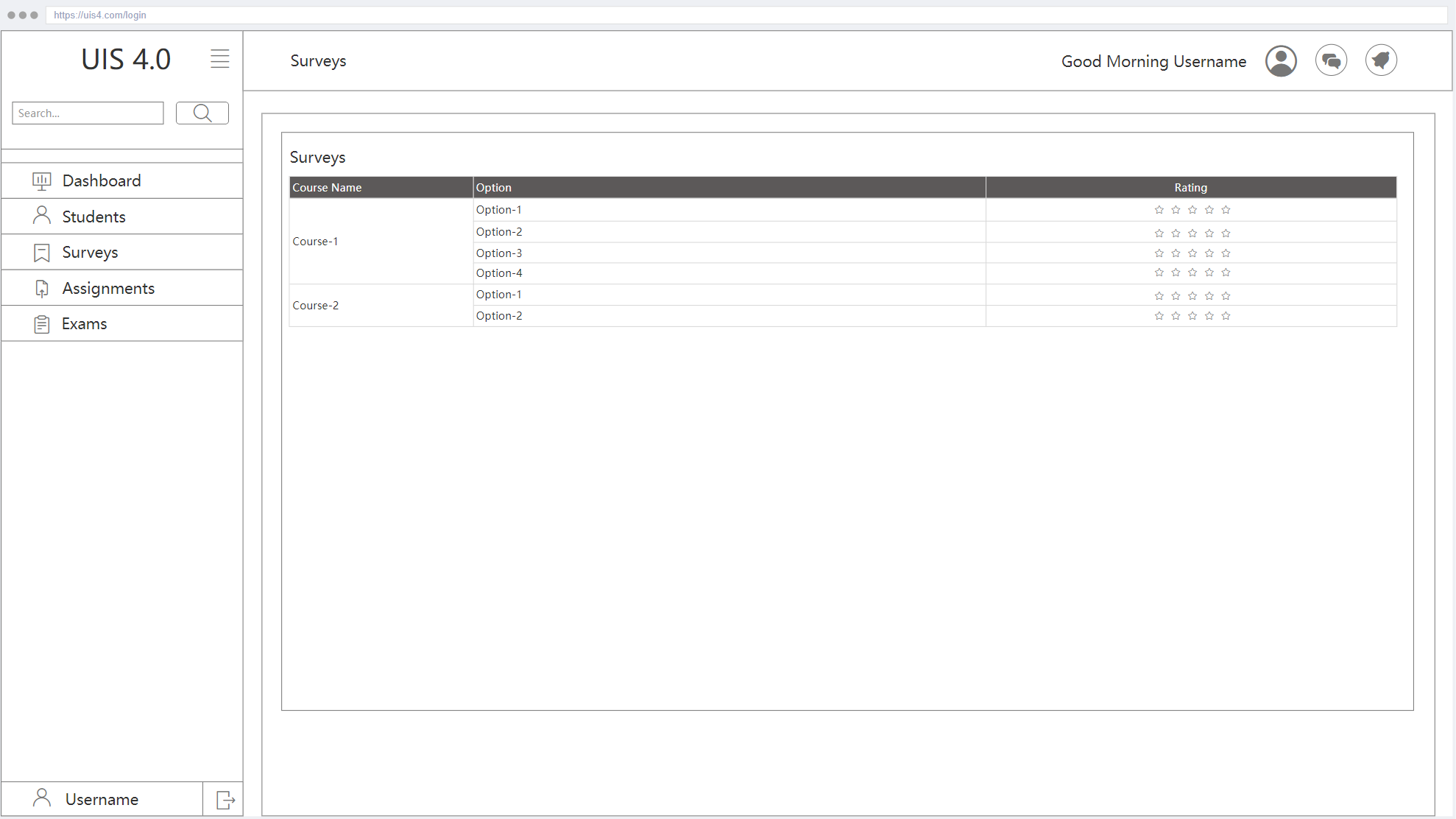
| **Name of Item** | Assignment Create |
| --- | --- |
| **Description of Purpose** | For Creating Assignment |
| **Source of Input or Destination of Output** | Ass. Name input, Course options, Deadline input, Upload Documents button, Submit button |
| **Relationships to Other Inputs / Outputs** | This page creates a new assignment with given information if the submit button is clicked. The upload documents button is for uploading any documents for assignment. |

#### 3.1.1.31 Creating Survey



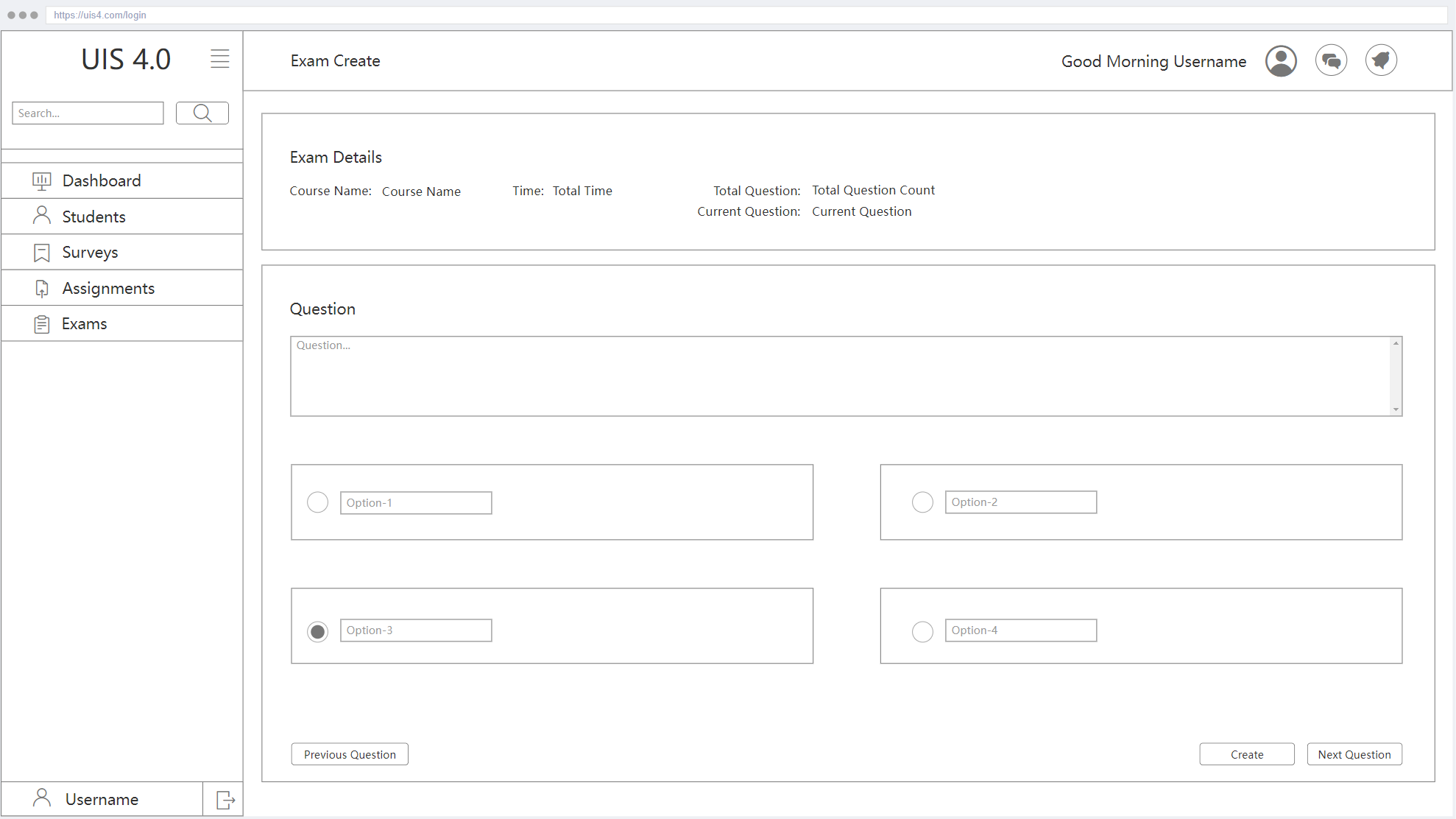
| **Name of Item** | Creating Survey |
| --- | --- |
| **Description of Purpose** | For Creating Survey |
| **Source of Input or Destination of Output** | Question input, Create button, Add Question button |
| **Relationships to Other Inputs / Outputs** | 1. The add question button is for adding extra questions into the survey. 2. This page creates a new survey with given questions if the create button is clicked and also redirects to the surveys page. |

#### 3.1.1.32 Survey List



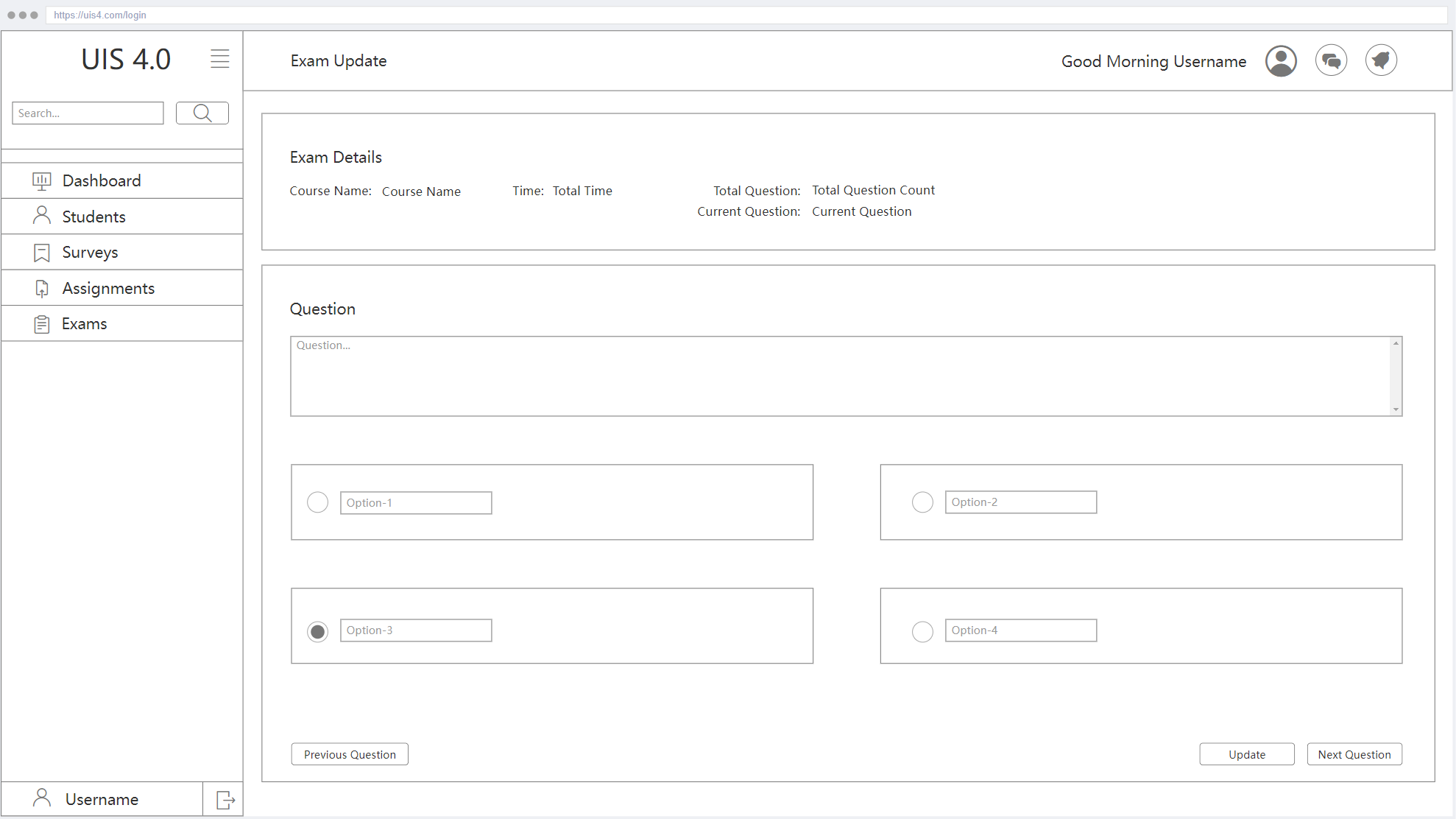
| **Name of Item** | Survey List |
| --- | --- |
| **Description of Purpose** | Viewing List of Surveys |
| **Source of Input or Destination of Output** | - |
| **Relationships to Other Inputs / Outputs** | - |

#### 3.1.1.33 Exam Create



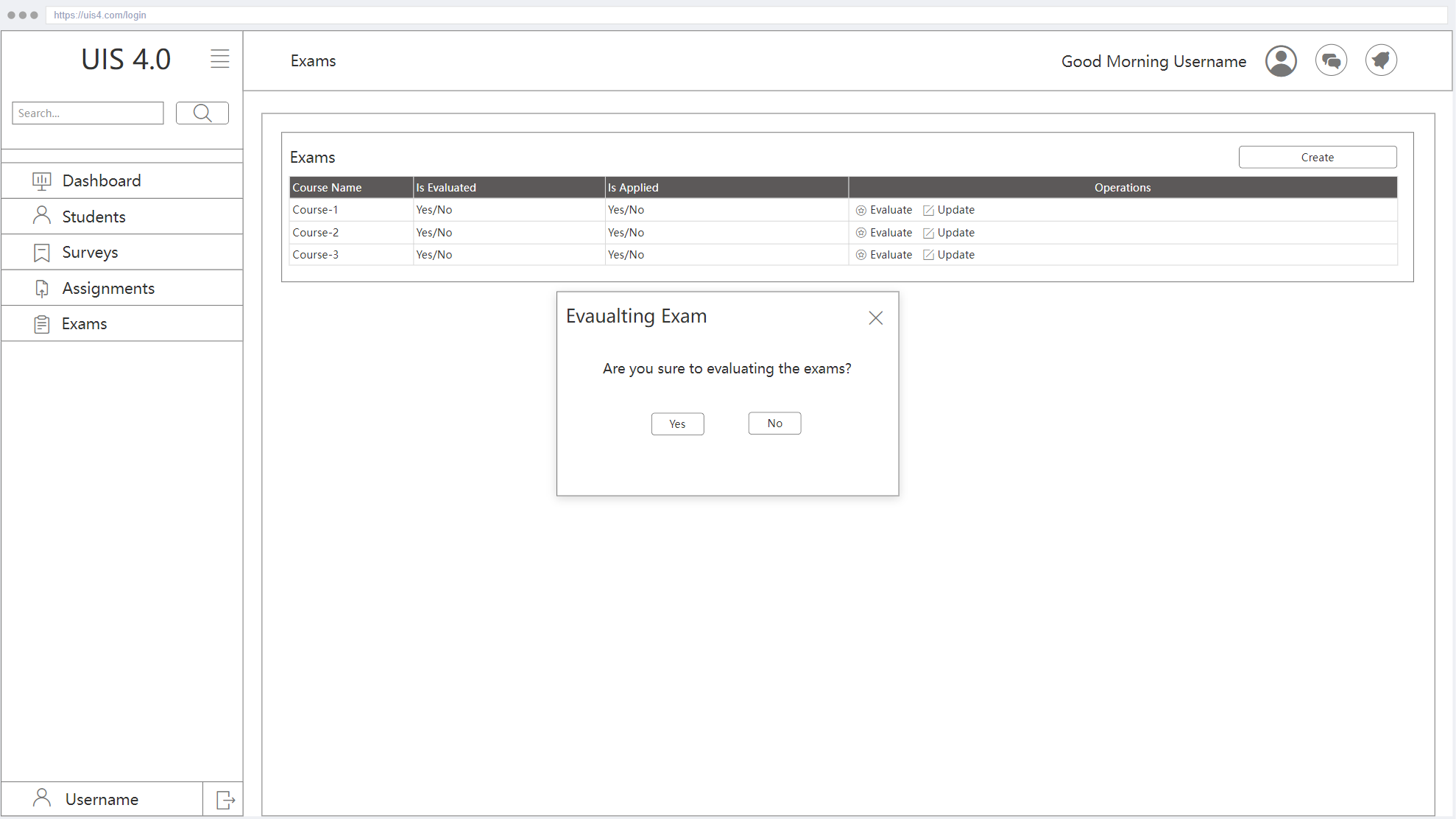
| **Name of Item** | Exam Create |
| --- | --- |
| **Description of Purpose** | For Creating Exam |
| **Source of Input or Destination of Output** | Question input, Option inputs, Previous Question button, Next Question button, Create button |
| **Relationships to Other Inputs / Outputs** | 1. This page redirects to the next question if the next question button is clicked, or to the previous question if the previous question button is clicked. 2. This page creates a new exam with given questions if the create button is clicked. |

#### 3.1.1.34 Exam Update



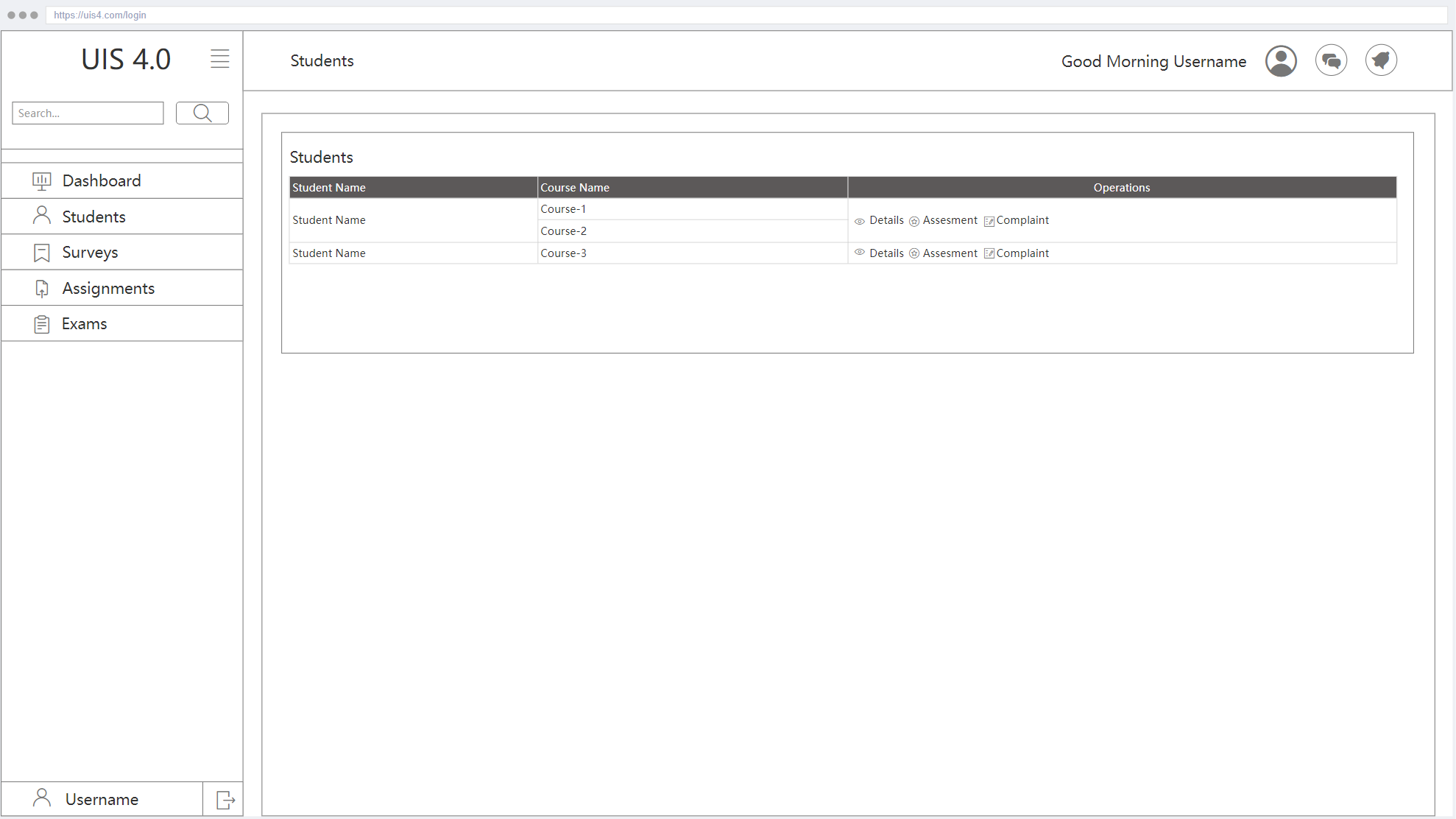
| **Name of Item** | Exam Update |
| --- | --- |
| **Description of Purpose** | For Updating Exam |
| **Source of Input or Destination of Output** | Question input, Option inputs, Previous Question button, Next Question button, Update button |
| **Relationships to Other Inputs / Outputs** | 1. This page redirects to the next question if the next question button is clicked, or to the previous question if the previous question button is clicked. 2. This page updates related exams with given questions if the update button is clicked. |

#### 3.1.1.35 Exam Evaluating



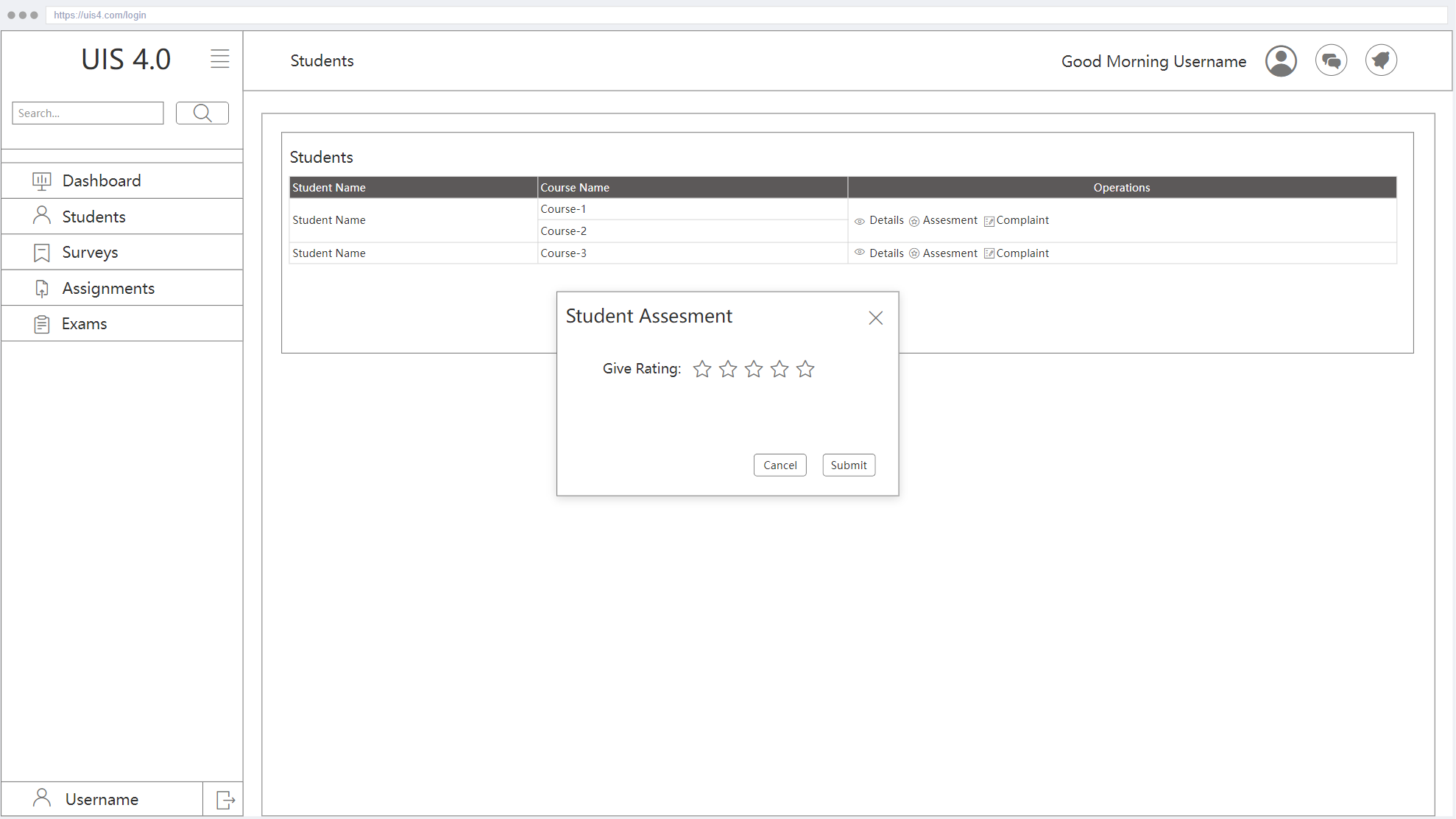
| **Name of Item** | Exam Evaluating |
| --- | --- |
| **Description of Purpose** | For Evaluating Exam |
| **Source of Input or Destination of Output** | This is a Confirm Window( Yes / No) |
| **Relationships to Other Inputs / Outputs** | If the option is ‘Yes’, the exam will be evaluated and the results can be seen, otherwise no action. |

#### 3.1.1.36 Student List



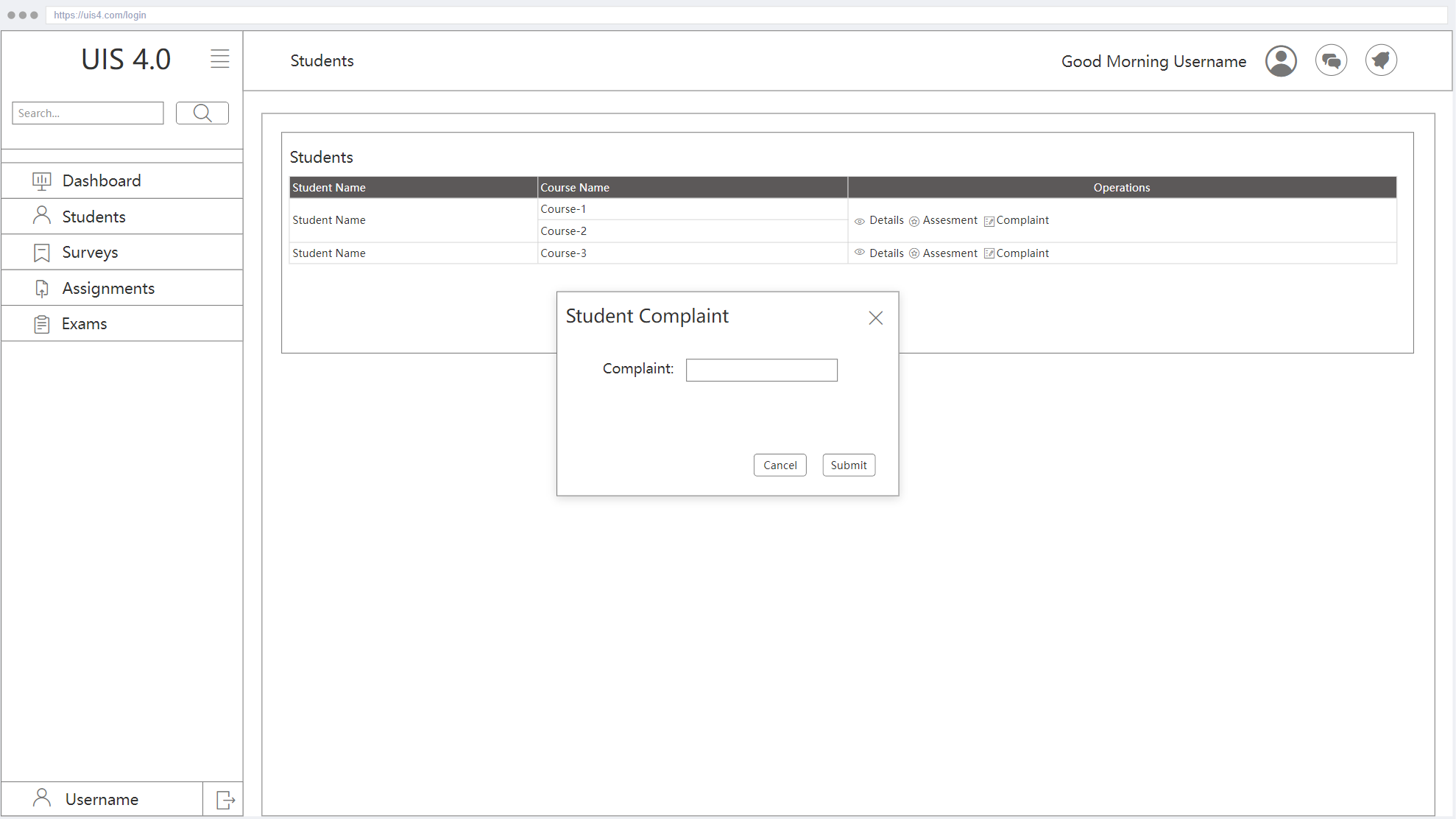
| **Name of Item** | Student List |
| --- | --- |
| **Description of Purpose** | View List of Students |
| **Source of Input or Destination of Output** | Details button, Assessment button, Complaint button |
| **Relationships to Other Inputs / Outputs** | 1. The detail button allows the instructor to view student information. 2. This page redirects to the student assessment page if the Assessment button is clicked. 3. This page redirects to the student critique page if the Complaint button is clicked. |

#### 3.1.1.37 Student Assessment



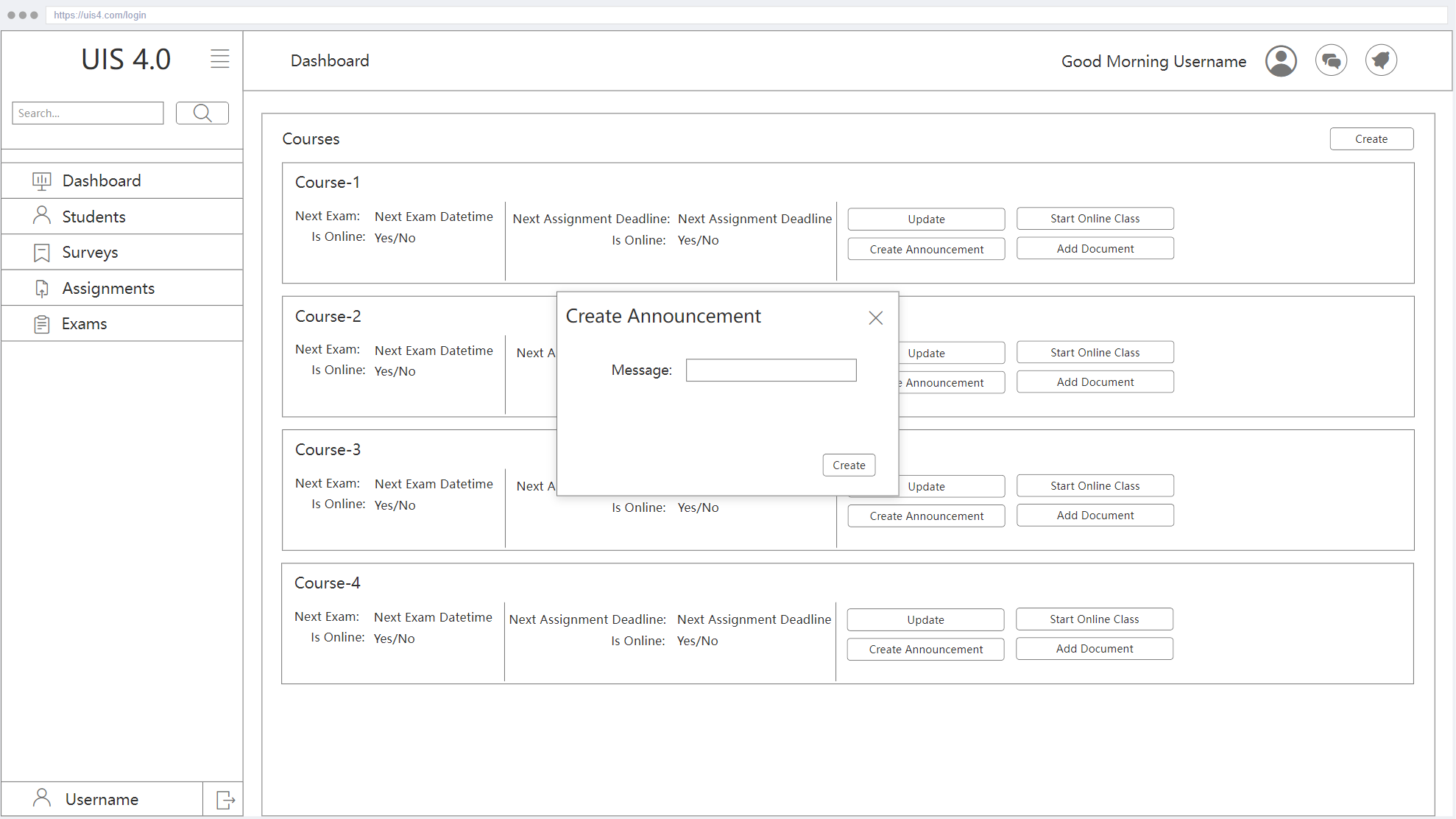
| **Name of Item** | Student Assessment |
| --- | --- |
| **Description of Purpose** | Creating Assessment for Student |
| **Source of Input or Destination of Output** | Assessment level input, Submit button |
| **Relationships to Other Inputs / Outputs** | This page redirects to the student's page after giving a chosen rating to the student if the submit button is clicked. |

#### 3.1.1.38 Student Critique



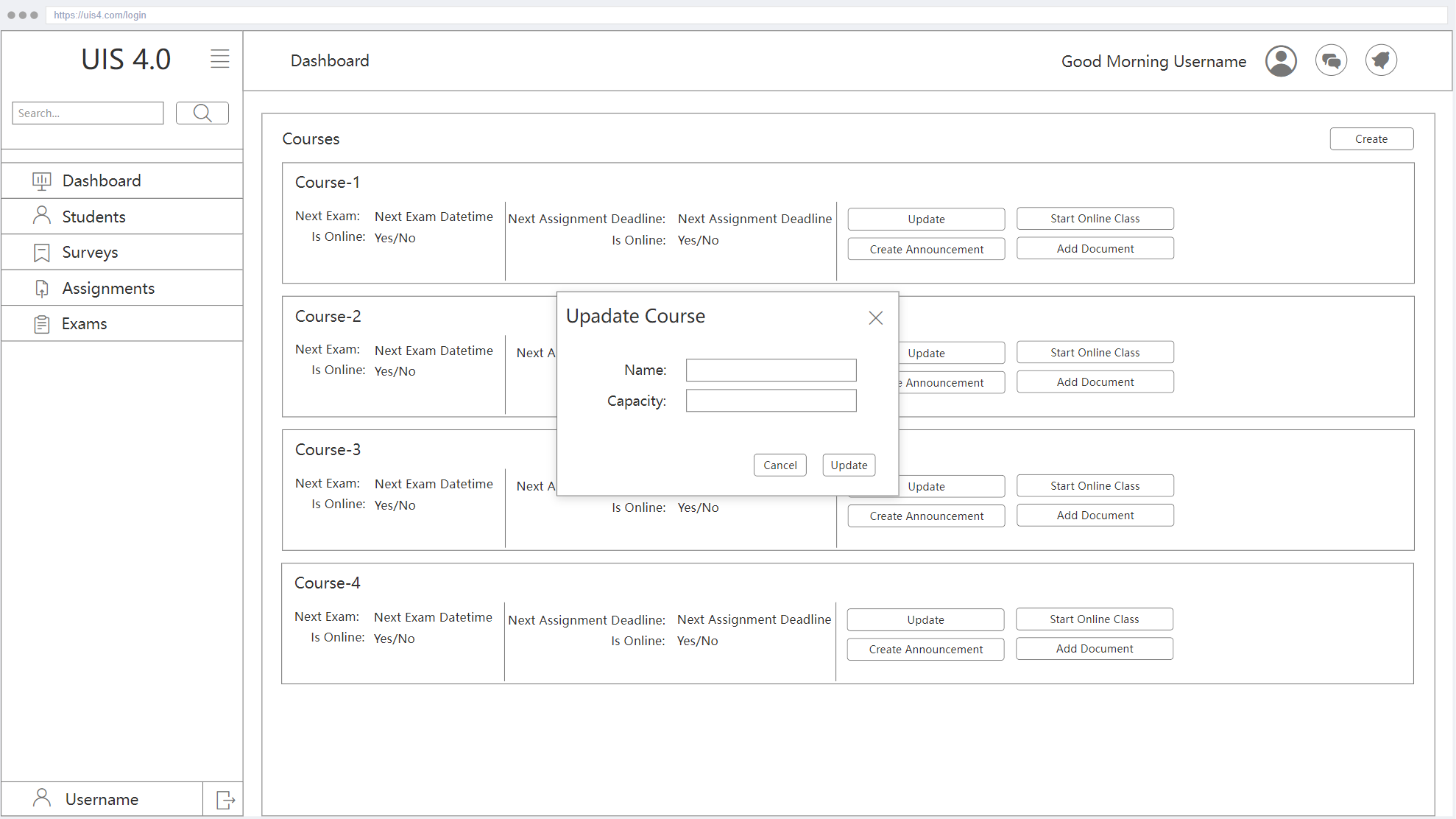
| **Name of Item** | Student Critique |
| --- | --- |
| **Description of Purpose** | Making Critique for Student |
| **Source of Input or Destination of Output** | Complaint input, Submit button |
| **Relationships to Other Inputs / Outputs** | This page redirects to the Students page after a written complaint if the submit button is clicked. |

#### 3.1.1.39 Making Announcement



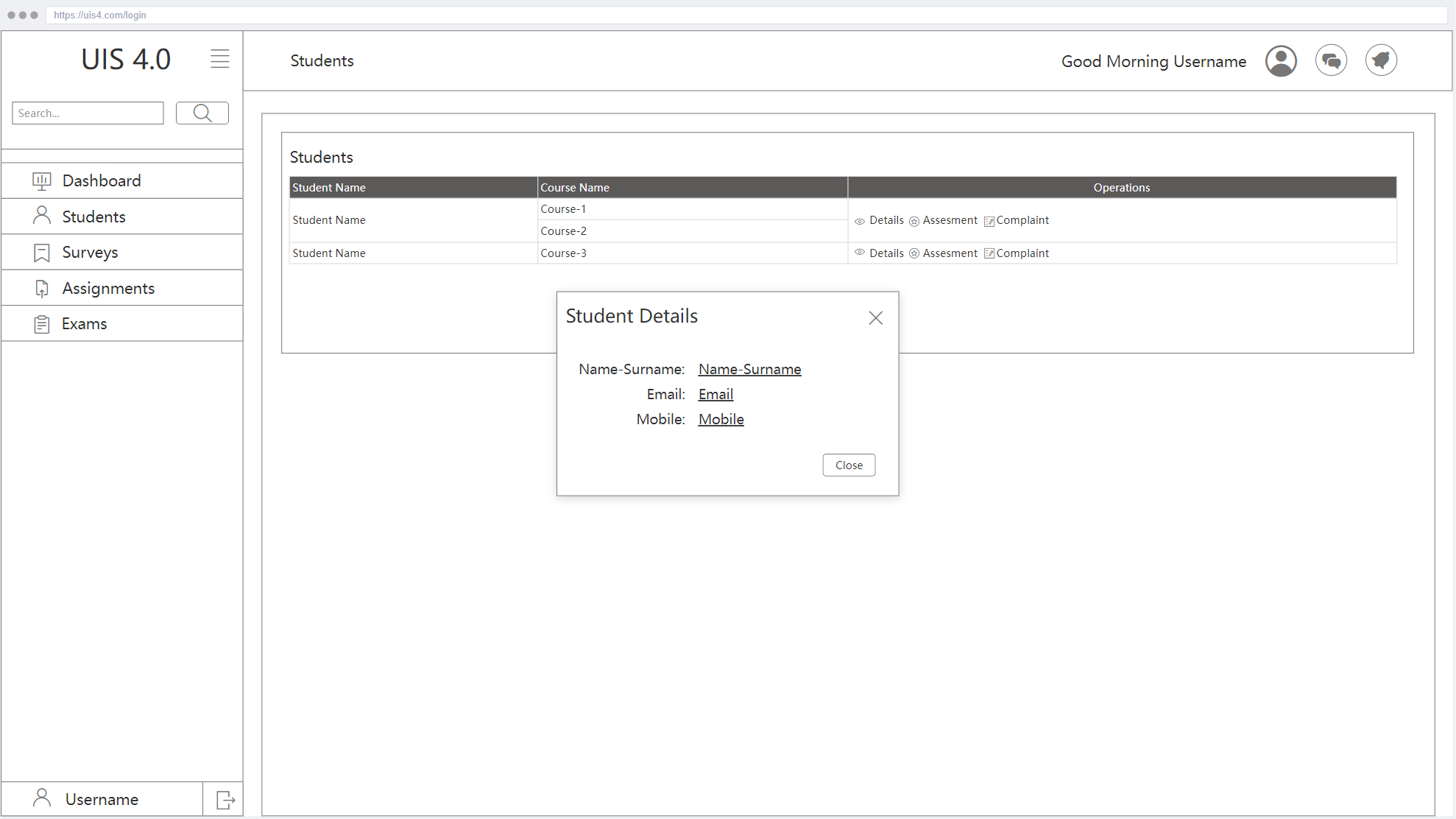
| **Name of Item** | Making Announcement |
| --- | --- |
| **Description of Purpose** | For Making Announcement |
| **Source of Input or Destination of Output** | Message input, Create button |
| **Relationships to Other Inputs / Outputs** | This page redirects to the Dashboard after creating the announcement with a given message if the create button is clicked. |

#### 3.1.1.40 Update Course



| **Name of Item** | Update Course |
| --- | --- |
| **Description of Purpose** | For Updating Course |
| **Source of Input or Destination of Output** | Name input, Capacity input, Update and Cancel buttons |
| **Relationships to Other Inputs / Outputs** | This page sends a request for updating the related course with a given name and capacity when the update button is clicked. |

#### 3.1.1.41 Student Details



| **Name of Item** | Student Details |
| --- | --- |
| **Description of Purpose** | For Viewing Student Details |
| **Source of Input or Destination of Output** | **Close** button |
| **Relationships to Other Inputs / Outputs** | To click the **Close** button, the *“Student Details”* modal will be closed. |

## 3.2. Functions

### 3.2.1. Login

| **Use Case ID** | The UIS 4.0 UC. 1 | |
| --- | --- | --- |
| **Description** | Users who have got a password via UIS system email while integration, can log in to the UIS 4.0 system. | |
| **Actors** | Students or Course Instructors | |
| **Preconditions** | Users must have created while integrating. | |
| **Post Conditions** | User logins to the system successfully | |
| **Main Success Scenario** | 1. User enters his credentials in the area provided for that.  2. User clicks the login button.  3. System checks if the entered login parameters are valid.  4. System creates a new session for the user. | |
| **Alternative Flow** | NA | |
| **Exceptional Flow** | ­If the user provides invalid login information, the system notifies the user and redirects him to the login section. | |
| **Rules** | **Name of the Business Rule** | **Result(s)** |
| NA | NA |

**3.2.1.1. Functional Requirements-1** The system shall control password and username properly.

**3.2.1.2. Functional Requirements-2** The system shall control password and username in a secure way.

**3.2.1.3. Functional Requirements-3** Whether the user logins to the system successfully or not the system shall inform the user by showing a message.

### 3.2.2. Forgot Password

| **Use Case ID** | The UIS 4.0 UC. 2 | |
| --- | --- | --- |
| **Description** | Users who have got a password via UIS system email while integration, can request password change code from the UIS 4.0 system. | |
| **Actors** | Students or Course Instructors | |
| **Preconditions** | User must be exist | |
| **Post Conditions** | User got password change code | |
| **Main Success Scenario** | 1. User enters his credentials in the area provided for that.  2. User clicks on the “Send” button.  3. System checks if the entered parameters are valid.  4. System sends password change codes to the user. | |
| **Alternative Flow** | NA | |
| **Exceptional Flow** | If user does not exist system will show default browser pop-up with exception | |
| **Rules** | **Name of the Business Rule** | **Result(s)** |
| NA | NA |

**3.2.2.1. Functional Requirements-1** The system shall control email properly.

**3.2.2.2. Functional Requirements-2** The system shall control email in a secure way.

**3.2.2.3. Functional Requirements-3** Whether the user exists and email sent successfully or not the system shall inform the user by showing a message.

### 3.2.3. Change Password

| **Use Case ID** | The UIS 4.0 UC. 3 | |
| --- | --- | --- |
| **Description** | Users who have got a password renew code via UIS system email, can use this code for password changing on the UIS 4.0 system. | |
| **Actors** | Students or Course Instructors | |
| **Preconditions** | User must have request password change code | |
| **Post Conditions** | Users password changed and redirected to login | |
| **Main Success Scenario** | 1. User enters his credentials in the area provided for that.  2. User clicks on the “Send” button.  3. System checks if the entered parameters are valid.  4. System changed the user's password and redirected to the login page. | |
| **Alternative Flow** | NA | |
| **Exceptional Flow** | If code is not valid or passwords are not same or passwords are not satisfy the requirements system shows default browser pop-up with exception | |
| **Rules** | **Name of the Business Rule** | **Result(s)** |
| NA | NA |

**3.2.3.1. Functional Requirements-1** The system shall control code and password properly.

**3.2.3.2. Functional Requirements-2** The system shall control code and password in a secure way.

**3.2.3.3. Functional Requirements-3** Whether the password changes or not the system shall inform the user by showing a message.

### 3.2.4. Making Course Registration

| **Use Case ID** | The UIS 4.0 UC. 4 | |
| --- | --- | --- |
| **Description** | Students can save courses that are added | |
| **Actors** | Students | |
| **Preconditions** | Student must have login, has been add at least one course, courses quota must have available | |
| **Post Conditions** | Students registered to the courses that are added | |
| **Main Success Scenario** | 1. The student clicks on the “Save” button.  2. System checks if the selected courses are available.  3. System adds the student to the course and redirects the student to the dashboard. | |
| **Alternative Flow** | NA | |
| **Exceptional Flow** | If one or more condition for registration is not satisfied, the system shows default browser pop-up with exception | |
| **Rules** | **Name of the Business Rule** | **Result(s)** |
| NA | NA |

**3.2.4.1. Functional Requirements-1** The system shall control the selected courses quota properly.

**3.2.4.2. Functional Requirements-2** The system shall control if the student has at least one course in a secure way.

**3.2.4.3. Functional Requirements-3** Whether the registration is successful or not the system shall inform the user by showing a message.

### 3.2.5. Add Course

| **Use Case ID** | The UIS 4.0 UC. 5 | |
| --- | --- | --- |
| **Description** | Student can add courses to their basket | |
| **Actors** | Student | |
| **Preconditions** | Student must have login, courses quota must have available | |
| **Post Conditions** | Course added to student and waiting for registration | |
| **Main Success Scenario** | 1. The student clicks on the “Add” button.  2. System checks if the selected course is available.  3. System add course to the student’s basket | |
| **Alternative Flow** | NA | |
| **Exceptional Flow** | If one or more condition for adding course to basket is not satisfied, system shows default browser pop-up with exception | |
| **Rules** | **Name of the Business Rule** | **Result(s)** |
| NA | NA |

**3.2.5.1. Functional Requirements-1** The system shall control selected course quota properly.

**3.2.5.2. Functional Requirements-2** The system shall control selected courses quota in a secure way.

**3.2.5.3. Functional Requirements-3** Whether the adding is successful or not the system shall inform the user by showing a message.

### 

### 3.2.6. Delete Course

| **Use Case ID** | The UIS 4.0 UC. 6 | |
| --- | --- | --- |
| **Description** | The student can delete courses they have added or had already registered. | |
| **Actors** | Student | |
| **Preconditions** | Student must have login | |
| **Post Conditions** | The relation between student and course removed | |
| **Main Success Scenario** | 1. The student clicks on the “Delete” button.  2. System delete course to the student’s basket or registration table | |
| **Alternative Flow** | NA | |
| **Exceptional Flow** | If one or more condition for deleting course is not satisfied, system shows default browser pop-up with exception | |
| **Rules** | **Name of the Business Rule** | **Result(s)** |
| NA | NA |

**3.2.6.1. Functional Requirements-1** The system shall control whether the user has already registered to course properly.

**3.2.6.2. Functional Requirements-2** The system shall control whether the user has already registered to course in a secure way.

**3.2.6.3. Functional Requirements-3** Whether the result is successful or not the system shall inform the user by showing a message.

### 3.2.7. Attending Class

| **Use Case ID** | The UIS 4.0 UC. 7 | |
| --- | --- | --- |
| **Description** | Student can access online class meetings or can sent attend request | |
| **Actors** | Student | |
| **Preconditions** | User must have login, have already registered for course | |
| **Post Conditions** | User redirected to meeting URL or send attending request. | |
| **Main Success Scenario** | 1-User clicks the “Attend Class” button. 2-If the next lecture is online, redirect to the meeting url in the new tab. 3-If not, sent request to the instructor for attending next class | |
| **Alternative Flow** | NA | |
| **Exceptional Flow** | If one or more condition for attending class is not satisfied, system shows default browser pop-up with exception | |
| **Rules** | **Name of the Business Rule** | **Result(s)** |
| NA | NA |

**3.2.7.1. Functional Requirements-1** The system shall control whether the selected course’s next class is online or not properly.

**3.2.7.2. Functional Requirements-2** The system shall control student registered for course

**3.2.7.3. Functional Requirements-3** Whether the attending request is successful or not the system shall inform the user by showing a message.

### 3.2.8. See Assignment Details

| **Use Case ID** | The UIS 4.0 UC. 8 | |
| --- | --- | --- |
| **Description** | Student can access assignment details | |
| **Actors** | Students | |
| **Preconditions** | User must have login, have already registered for a course, course have an assignment | |
| **Post Conditions** | Student view assignment details | |
| **Main Success Scenario** | 1-User clicks “Assignment Details” on the list of assignments 2-User redirected to assignment detail page | |
| **Alternative Flow** | NA | |
| **Exceptional Flow** | NA | |
| **Rules** | **Name of the Business Rule** | **Result(s)** |
| NA | NA |

**3.2.8.1. Functional Requirements-1** The system shall return details of assignments.

**3.2.8.2. Functional Requirements-2** The system shall return details of assignments in a secure way.

**3.2.8.3. Functional Requirements-3** Whether the result is successful or not the system shall inform the user by showing a message.

### 3.2.9. Submit Assignment

| **Use Case ID** | The UIS 4.0 UC. 9 | |
| --- | --- | --- |
| **Description** | Student can submit assignment | |
| **Actors** | Students | |
| **Preconditions** | User must have login, have already registered for a course, course have an assignment | |
| **Post Conditions** | Student submit assignment | |
| **Main Success Scenario** | 1-User clicks “Submit” on the list of assignments 2-User select file from default OS pop-up. 3-System check deadline has pass, or student registered for course  4-Document upload | |
| **Alternative Flow** | NA | |
| **Exceptional Flow** | If one or more condition for attending class is not satisfied, system shows default browser pop-up with exception | |
| **Rules** | **Name of the Business Rule** | **Result(s)** |
| NA | NA |

**3.2.9.1. Functional Requirements-1** The system shall check the assignments deadline.

**3.2.9.2. Functional Requirements-2** The system shall check if the student that tries to submit assignments is registered for the course.

**3.2.9.3. Functional Requirements-3** Whether the result is successful or not the system shall inform the user by showing a message.

### 3.2.10. Contacting with Course Instructor

| **Use Case ID** | The UIS 4.0 UC. 10 | |
| --- | --- | --- |
| **Description** | Student can chat with course instructors | |
| **Actors** | Students | |
| **Preconditions** | User must have login, have already registered for a course | |
| **Post Conditions** | Student redirected to the message page with filled receiver who is course instructor | |
| **Main Success Scenario** | 1-User clicks message icon which near to instructor name in dashboard 2-System redirect user to messages page | |
| **Alternative Flow** | NA | |
| **Exceptional Flow** | If one or more condition for attending class is not satisfied, system shows default browser pop-up with exception | |
| **Rules** | **Name of the Business Rule** | **Result(s)** |
| NA | NA |

**3.2.10.1. Functional Requirements-1** The system shall check if the user is logged in.

**3.2.10.2. Functional Requirements-2** The system shall check if the student that tries to contact the instructor is registered for the course.

**3.2.10.3. Functional Requirements-3** Whether the result is successful or not the system shall inform the user by showing a message.

### 3.2.11. Viewing Course Documents

| **Use Case ID** | The UIS 4.0 UC. 11 | |
| --- | --- | --- |
| **Description** | Student can view course documents | |
| **Actors** | Students | |
| **Preconditions** | User must have login, have already registered for a course | |
| **Post Conditions** | Student redirected to course documents page | |
| **Main Success Scenario** | 1-User clicks “Documents” which on the course section in dashboard 2-System redirect user to course documents page | |
| **Alternative Flow** | NA | |
| **Exceptional Flow** | If one or more condition for attending class is not satisfied, system shows default browser pop-up with exception | |
| **Rules** | **Name of the Business Rule** | **Result(s)** |
| NA | NA |

**3.2.11.1. Functional Requirements-1** The system shall check if the user is logged in.

**3.2.11.2. Functional Requirements-2** The system shall check if the student is registered for any course.

**3.2.11.3. Functional Requirements-3** Whether the result is successful or not the system shall inform the user by showing a message.

### 

### 

### 

### 3.2.12. Downloading Course Documents

| **Use Case ID** | The UIS 4.0 UC. 12 | |
| --- | --- | --- |
| **Description** | Student can download course documents | |
| **Actors** | Students | |
| **Preconditions** | User must have login, have already registered for a course | |
| **Post Conditions** | Student download document. | |
| **Main Success Scenario** | 1-User clicks “Download” which in the document list 2-System allow user to download course document | |
| **Alternative Flow** | NA | |
| **Exceptional Flow** | If one or more condition for attending class is not satisfied, system shows default browser pop-up with exception | |
| **Rules** | **Name of the Business Rule** | **Result(s)** |
| NA | NA |

**3.2.12.1. Functional Requirements-1** The system shall check if the user is logged in.

**3.2.12.2. Functional Requirements-2** The system shall check if the student is registered for the course.

**3.2.12.3. Functional Requirements-3** Whether the result is successful or not the system shall inform the user by showing a message.

### 3.2.13. Attending Exam

| **Use Case ID** | The UIS 4.0 UC. 13 | |
| --- | --- | --- |
| **Description** | Student can attend exam | |
| **Actors** | Students | |
| **Preconditions** | User must have login, have already registered for a course | |
| **Post Conditions** | Student attend exam | |
| **Main Success Scenario** | 1-User clicks “Start” which is in the exam details page. 2-System check user registered for course and exam start date is passed. | |
| **Alternative Flow** | NA | |
| **Exceptional Flow** | If one or more condition for attending class is not satisfied, system shows default browser pop-up with exception | |
| **Rules** | **Name of the Business Rule** | **Result(s)** |
| NA | NA |

**3.2.13.1. Functional Requirements-1**

The system shall check if the student that tries to attend the exam is registered for the course.

**3.2.13.2. Functional Requirements-2**

The system shall check whether the exam start date is passed or not.

**3.2.13.3. Functional Requirements-3** Whether the result is successful or not the system shall inform the user by showing a message.

### 3.2.14. In Exam Giving Answers

| **Use Case ID** | The UIS 4.0 UC. 14 | |
| --- | --- | --- |
| **Description** | Student can give answers to question in exam | |
| **Actors** | Students | |
| **Preconditions** | User must have login, have already registered for a course, have already attend exam | |
| **Post Conditions** | Student give answer | |
| **Main Success Scenario** | 1-User clicks option. 2-System check user registered for course and exam start date is passed. | |
| **Alternative Flow** | NA | |
| **Exceptional Flow** | If one or more condition for attending class is not satisfied, system shows default browser pop-up with exception | |
| **Rules** | **Name of the Business Rule** | **Result(s)** |
| NA | NA |

**3.2.14.1. Functional Requirements-1**

The system shall check if the student that tries to attend the exam is registered for the course.

**3.2.14.2. Functional Requirements-2**

The system shall check whether the exam start date is passed or not.

**3.2.14.3. Functional Requirements-3** Whether the result is successful or not the system shall inform the user by showing a message.

### 3.2.15. In Exam Getting Previous Question

| **Use Case ID** | The UIS 4.0 UC. 15 | |
| --- | --- | --- |
| **Description** | Student can return previous question | |
| **Actors** | Students | |
| **Preconditions** | User must have login, have already registered for a course, have already attend exam, has already answer first question | |
| **Post Conditions** | Student view previous question | |
| **Main Success Scenario** | 1-User clicks “Previous Question”. 2-System checks if there is a previous question. | |
| **Alternative Flow** | NA | |
| **Exceptional Flow** | If one or more condition for attending class is not satisfied, system shows default browser pop-up with exception | |
| **Rules** | **Name of the Business Rule** | **Result(s)** |
| NA | NA |

**3.2.15.1. Functional Requirements-1**

The system shall check if there is a previous question.

**3.2.15.2. Functional Requirements-2** The system shall check if the student is registered for the course.

**3.2.15.3. Functional Requirements-3** Whether the result is successful or not the system shall inform the user by showing a message.

### 3.2.16. In Exam Getting Next Question

| **Use Case ID** | The UIS 4.0 UC. 16 | |
| --- | --- | --- |
| **Description** | Student can move next question | |
| **Actors** | Students | |
| **Preconditions** | User must have login, have already registered for a course, have already attend exam, has not in the last question | |
| **Post Conditions** | Student view next question | |
| **Main Success Scenario** | 1-User clicks “Next Question”. 2-System check if there is a next question. | |
| **Alternative Flow** | NA | |
| **Exceptional Flow** | If one or more condition for attending class is not satisfied, system shows default browser pop-up with exception | |
| **Rules** | **Name of the Business Rule** | **Result(s)** |
| NA | NA |

**3.2.16.1. Functional Requirements-1**

The system shall check if there is a next question.

**3.2.16.2. Functional Requirements-2** The system shall check if the student is registered for the course.

**3.2.16.3. Functional Requirements-3** Whether the result is successful or not the system shall inform the user by showing a message.

### 3.2.17. Finishing Exam

| **Use Case ID** | The UIS 4.0 UC. 17 | |
| --- | --- | --- |
| **Description** | Student can finish exam | |
| **Actors** | Students | |
| **Preconditions** | User must have login, have already registered for a course, have already attended exam, has view the last question | |
| **Post Conditions** | Student complete exam | |
| **Main Success Scenario** | 1-User clicks “Finish Exam”. 2-System close users session. | |
| **Alternative Flow** | NA | |
| **Exceptional Flow** | If one or more condition for attending class is not satisfied, system shows default browser pop-up with exception | |
| **Rules** | **Name of the Business Rule** | **Result(s)** |
| NA | NA |

**3.2.17.1. Functional Requirements-1** The system shall check if the student is registered for the course.

**3.2.17.2. Functional Requirements-2** The system shall check if the student has attended the exam.

**3.2.17.3. Functional Requirements-3** Whether the result is successful or not the system shall inform the user by showing a message.

### 3.2.18. Viewing Exam Result

| **Use Case ID** | The UIS 4.0 UC. 18 | |
| --- | --- | --- |
| **Description** | Student can view exam results | |
| **Actors** | Students | |
| **Preconditions** | User must have login, have already registered for a course, have already attend exam, instructor has been evaluate exams | |
| **Post Conditions** | Student view exam result | |
| **Main Success Scenario** | 1-User clicks “Exams” on navigation. 2-System load exams information. | |
| **Alternative Flow** | NA | |
| **Exceptional Flow** | If one or more condition for attending class is not satisfied, system shows default browser pop-up with exception | |
| **Rules** | **Name of the Business Rule** | **Result(s)** |
| NA | NA |

**3.2.18.1. Functional Requirements-1** The system shall check if the student is registered for any course.

**3.2.18.2. Functional Requirements-2** The system shall check if the student is logged in.

**3.2.18.3. Functional Requirements-3** Whether the result is successful or not the system shall inform the user by showing a message.

### 

### 

### 3.2.19. Viewing Note List

| **Use Case ID** | The UIS 4.0 UC. 19 | |
| --- | --- | --- |
| **Description** | Student can view note list | |
| **Actors** | Students | |
| **Preconditions** | User must have login | |
| **Post Conditions** | Student view list of notes | |
| **Main Success Scenario** | 1-User clicks “Notes” on navigation. 2-System load notes. | |
| **Alternative Flow** | NA | |
| **Exceptional Flow** | If one or more condition for attending class is not satisfied, system shows default browser pop-up with exception | |
| **Rules** | **Name of the Business Rule** | **Result(s)** |
| NA | NA |

**3.2.19.1. Functional Requirements-1** The system shall check if the student has any notes.

**3.2.19.2. Functional Requirements-2** The system shall check if the student is logged in.

**3.2.19.3. Functional Requirements-3** Whether the result is successful or not the system shall inform the user by showing a message.

### 3.2.20. Adding New Note

| **Use Case ID** | The UIS 4.0 UC. 20 | |
| --- | --- | --- |
| **Description** | Student can add notes | |
| **Actors** | Students | |
| **Preconditions** | User must have login | |
| **Post Conditions** | Student view note detail page | |
| **Main Success Scenario** | 1-User clicks the “Add Notes” button in the notes page. 2-System creates new notes and redirects the user to note details. | |
| **Alternative Flow** | NA | |
| **Exceptional Flow** | If one or more condition for attending class is not satisfied, system shows default browser pop-up with exception | |
| **Rules** | **Name of the Business Rule** | **Result(s)** |
| NA | NA |

**3.2.20.1. Functional Requirements-1** The system shall check note length is not greater than expected.

**3.2.20.2. Functional Requirements-2** The system shall check if the student is logged in.

**3.2.20.3. Functional Requirements-3** Whether the result is successful or not the system shall inform the user by showing a message.

### 3.2.21. Updating Existing Note

| **Use Case ID** | The UIS 4.0 UC. 21 | |
| --- | --- | --- |
| **Description** | Student can update notes | |
| **Actors** | Students | |
| **Preconditions** | User must have login | |
| **Post Conditions** | Student updates notes | |
| **Main Success Scenario** | 1-User clicks the “Update” button in the notes page. 2-System redirects the user to note details. | |
| **Alternative Flow** | NA | |
| **Exceptional Flow** | If one or more condition for attending class is not satisfied, system shows default browser pop-up with exception | |
| **Rules** | **Name of the Business Rule** | **Result(s)** |
| NA | NA |

**3.2.21.1. Functional Requirements-1** The system shall check that the student's notes length is not greater than expected.

**3.2.21.2. Functional Requirements-2** The system shall check if the student is logged in.

**3.2.21.3. Functional Requirements-3** Whether the result is successful or not the system shall inform the user by showing a message.

### 3.2.22. Deleting Existing Note

| **Use Case ID** | The UIS 4.0 UC. 22 | |
| --- | --- | --- |
| **Description** | Student can delete notes | |
| **Actors** | Students | |
| **Preconditions** | User must have login | |
| **Post Conditions** | Student delete notes | |
| **Main Success Scenario** | 1-User clicks the “Delete” button in the notes page. 2-System redirect deletes notes. | |
| **Alternative Flow** | NA | |
| **Exceptional Flow** | If one or more condition for attending class is not satisfied, system shows default browser pop-up with exception | |
| **Rules** | **Name of the Business Rule** | **Result(s)** |
| NA | NA |

**3.2.22.1. Functional Requirements-1** The system shall check if the student that tries to delete is the owner of the notes.

**3.2.22.2. Functional Requirements-2** The system shall check if the student is logged in.

**3.2.22.3. Functional Requirements-3** Whether the result is successful or not the system shall inform the user by showing a message.

### 3.2.23. Viewing Surveys

| **Use Case ID** | The UIS 4.0 UC. 23 | |
| --- | --- | --- |
| **Description** | Student can view surveys | |
| **Actors** | Students | |
| **Preconditions** | User must have login | |
| **Post Conditions** | Student view surveys | |
| **Main Success Scenario** | 1-User clicks the notification button in the header.  2-User click survey notifications 2-System load survey notifications | |
| **Alternative Flow** | NA | |
| **Exceptional Flow** | If one or more condition for attending class is not satisfied, system shows default browser pop-up with exception | |
| **Rules** | **Name of the Business Rule** | **Result(s)** |
| NA | NA |

**3.2.23.1. Functional Requirements-1** The system shall check if the student has any survey.

**3.2.23.2. Functional Requirements-2** The system shall check if the student is logged in.

**3.2.23.3. Functional Requirements-3** Whether the result is successful or not the system shall inform the user by showing a message.

### 

### 

### 

### 3.2.24. Attending Surveys

| **Use Case ID** | The UIS 4.0 UC. 24 | |
| --- | --- | --- |
| **Description** | Student can attend survey | |
| **Actors** | Students | |
| **Preconditions** | User must have login | |
| **Post Conditions** | Student view survey | |
| **Main Success Scenario** | 1-User click the notification button in the header.  2-User click survey notifications 3-User click the survey that want to attend  4-System load survey details | |
| **Alternative Flow** | NA | |
| **Exceptional Flow** | If one or more condition for attending class is not satisfied, system shows default browser pop-up with exception | |
| **Rules** | **Name of the Business Rule** | **Result(s)** |
| NA | NA |

**3.2.24.1. Functional Requirements-1** The system shall check if the student has the right to attend the survey.

**3.2.24.2. Functional Requirements-2** The system shall check if the student is logged in.

**3.2.24.3. Functional Requirements-3** Whether the result is successful or not the system shall inform the user by showing a message.

### 3.2.25. Filling Out Surveys

| **Use Case ID** | The UIS 4.0 UC. 25 | |
| --- | --- | --- |
| **Description** | Student can filling out survey | |
| **Actors** | Students | |
| **Preconditions** | User must have login, view survey details | |
| **Post Conditions** | Student submit survey | |
| **Main Success Scenario** | 1-User gives ratings  2-User click “Submit Survey” 3-System saves survey | |
| **Alternative Flow** | NA | |
| **Exceptional Flow** | If one or more condition for attending class is not satisfied, system shows default browser pop-up with exception | |
| **Rules** | **Name of the Business Rule** | **Result(s)** |
| NA | NA |

**3.2.25.1. Functional Requirements-1**

The system shall check if all the questions are rated.

**3.2.25.2. Functional Requirements-2** The system shall check if the student is logged in.

**3.2.25.3. Functional Requirements-3** Whether the result is successful or not the system shall inform the user by showing a message.

### 3.2.26. Viewing Communities

| **Use Case ID** | The UIS 4.0 UC. 26 | |
| --- | --- | --- |
| **Description** | Student can view communities | |
| **Actors** | Students | |
| **Preconditions** | User must have login | |
| **Post Conditions** | Student view communities | |
| **Main Success Scenario** | 1-User click “Community Management”  2-System load communities | |
| **Alternative Flow** | NA | |
| **Exceptional Flow** | If one or more condition for attending class is not satisfied, system shows default browser pop-up with exception | |
| **Rules** | **Name of the Business Rule** | **Result(s)** |
| NA | NA |

**3.2.26.1. Functional Requirements-1** The system shall check if there are any communities.

**3.2.26.2. Functional Requirements-2** The system shall check if the student is logged in.

**3.2.26.3. Functional Requirements-3** Whether the result is successful or not the system shall inform the user by showing a message.

### 

### 

### 3.2.27. Joining Communities

| **Use Case ID** | The UIS 4.0 UC. 27 | |
| --- | --- | --- |
| **Description** | Student can join communities | |
| **Actors** | Students | |
| **Preconditions** | User must have login | |
| **Post Conditions** | Student joined communities | |
| **Main Success Scenario** | 1-User click “Join” button  2-System enrolled user to community | |
| **Alternative Flow** | NA | |
| **Exceptional Flow** | If one or more condition for attending class is not satisfied, system shows default browser pop-up with exception | |
| **Rules** | **Name of the Business Rule** | **Result(s)** |
| NA | NA |

**3.2.27.1. Functional Requirements-1** The system shall check if the community exists.

**3.2.27.2. Functional Requirements-2** The system shall check if the student is logged in.

**3.2.27.3. Functional Requirements-3** Whether the result is successful or not the system shall inform the user by showing a message.

### 3.2.28. Viewing Communities Messages

| **Use Case ID** | The UIS 4.0 UC. 28 | |
| --- | --- | --- |
| **Description** | Student can view community messages | |
| **Actors** | Students | |
| **Preconditions** | User must have login | |
| **Post Conditions** | Student view community messages | |
| **Main Success Scenario** | 1-User click the message button in the header.  2-User click community messages 2-System load community messages | |
| **Alternative Flow** | NA | |
| **Exceptional Flow** | If one or more condition for attending class is not satisfied, system shows default browser pop-up with exception | |
| **Rules** | **Name of the Business Rule** | **Result(s)** |
| NA | NA |

**3.2.28.1. Functional Requirements-1** The system shall check if the student has any community message.

**3.2.28.2. Functional Requirements-2** The system shall check if the student is logged in.

**3.2.28.3. Functional Requirements-3** Whether the result is successful or not the system shall inform the user by showing a message.

### 3.2.29. Leaving Communities

| **Use Case ID** | The UIS 4.0 UC. 29 | |
| --- | --- | --- |
| **Description** | Student can leave communities | |
| **Actors** | Students | |
| **Preconditions** | User must have login, enrolled community | |
| **Post Conditions** | Student leaved communities | |
| **Main Success Scenario** | 1-User click “Leave” button  2-System remove user from community | |
| **Alternative Flow** | NA | |
| **Exceptional Flow** | If one or more condition for attending class is not satisfied, system shows default browser pop-up with exception | |
| **Rules** | **Name of the Business Rule** | **Result(s)** |
| NA | NA |

**3.2.29.1. Functional Requirements-1** The system shall check if the student has enrolled to the community.

**3.2.29.2. Functional Requirements-2** The system shall check if the student is logged in.

**3.2.29.3. Functional Requirements-3** Whether the result is successful or not the system shall inform the user by showing a message.

### 3.2.30. Changing Personal Informations

| **Use Case ID** | The UIS 4.0 UC. 30 | |
| --- | --- | --- |
| **Description** | Student can change personal informations | |
| **Actors** | Students | |
| **Preconditions** | User must have login | |
| **Post Conditions** | Student personal informations updated | |
| **Main Success Scenario** | 1-User filled out inputs wants to change 2-User click update button. 3-System updates personal information | |
| **Alternative Flow** | NA | |
| **Exceptional Flow** | If one or more condition for attending class is not satisfied, system shows default browser pop-up with exception | |
| **Rules** | **Name of the Business Rule** | **Result(s)** |
| NA | NA |

**3.2.30.1. Functional Requirements-1** The system shall check if the student has given information in the expected format.

**3.2.30.2. Functional Requirements-2** The system shall check if the student is logged in.

**3.2.30.3. Functional Requirements-3** Whether the result is successful or not the system shall inform the user by showing a message.

### 3.2.31. Downloading Educational Informations

| **Use Case ID** | The UIS 4.0 UC. 31 | |
| --- | --- | --- |
| **Description** | Student can download education informations | |
| **Actors** | Students | |
| **Preconditions** | User must have login | |
| **Post Conditions** | Student download education information | |
| **Main Success Scenario** | 1-User click “Download Transcript” button  2-System allow user to download transcript | |
| **Alternative Flow** | NA | |
| **Exceptional Flow** | If one or more condition for attending class is not satisfied, system shows default browser pop-up with exception | |
| **Rules** | **Name of the Business Rule** | **Result(s)** |
| NA | NA |

**3.2.31.1. Functional Requirements-1** The system shall check if the student has a transcript.

**3.2.31.2. Functional Requirements-2** The system shall check if the student is logged in.

**3.2.31.3. Functional Requirements-3** Whether the result is successful or not the system shall inform the user by showing a message.

### 3.2.32. Viewing Notifications

| **Use Case ID** | The UIS 4.0 UC. 32 | |
| --- | --- | --- |
| **Description** | Student can view notifications | |
| **Actors** | Students | |
| **Preconditions** | User must have login | |
| **Post Conditions** | Student view notifications | |
| **Main Success Scenario** | 1-User click the notifications button in the header.  2-User click personal notifications 2-System load personal notifications | |
| **Alternative Flow** | NA | |
| **Exceptional Flow** | If one or more condition for attending class is not satisfied, system shows default browser pop-up with exception | |
| **Rules** | **Name of the Business Rule** | **Result(s)** |
| NA | NA |

**3.2.32.1. Functional Requirements-1** The system shall check if the student has any notifications.

**3.2.32.2. Functional Requirements-2** The system shall check if the student is logged in.

**3.2.32.3. Functional Requirements-3** Whether the result is successful or not the system shall inform the user by showing a message.

### 3.2.33. Entering Excuse Report

| **Use Case ID** | The UIS 4.0 UC. 33 | |
| --- | --- | --- |
| **Description** | Student can upload excuse report | |
| **Actors** | Students | |
| **Preconditions** | User must have login | |
| **Post Conditions** | Student upload report. | |
| **Main Success Scenario** | 1-User clicks “Upload Excuse Report” which in the profile 2-System allow user to upload excuse report | |
| **Alternative Flow** | NA | |
| **Exceptional Flow** | If one or more condition for attending class is not satisfied, system shows default browser pop-up with exception | |
| **Rules** | **Name of the Business Rule** | **Result(s)** |
| NA | NA |

**3.2.33.1. Functional Requirements-1** The system shall check if the student has selected any file.

**3.2.33.2. Functional Requirements-2** The system shall check if the student is logged in.

**3.2.33.3. Functional Requirements-3** Whether the result is successful or not the system shall inform the user by showing a message.

### 3.2.34. Entering Permission Documents

| **Use Case ID** | The UIS 4.0 UC. 34 | |
| --- | --- | --- |
| **Description** | Student can upload permission document | |
| **Actors** | Students | |
| **Preconditions** | User must have login | |
| **Post Conditions** | Student upload permission document | |
| **Main Success Scenario** | 1-User clicks “Upload Permission Document” which in the profile 2-System allow user to upload permission document | |
| **Alternative Flow** | NA | |
| **Exceptional Flow** | If one or more condition for attending class is not satisfied, system shows default browser pop-up with exception | |
| **Rules** | **Name of the Business Rule** | **Result(s)** |
| NA | NA |

**3.2.34.1. Functional Requirements-1** The system shall check if the student has selected any file.

**3.2.34.2. Functional Requirements-2** The system shall check if the student is logged in.

**3.2.34.3. Functional Requirements-3** Whether the result is successful or not the system shall inform the user by showing a message.

### 3.2.35. Viewing Their Courses

| **Use Case ID** | The UIS 4.0 UC. 35 | |
| --- | --- | --- |
| **Description** | Instructors can view their course | |
| **Actors** | Instructor | |
| **Preconditions** | User must have login | |
| **Post Conditions** | Instructor view their course | |
| **Main Success Scenario** | 1-User clicks “Dashboard” which in the navigation 2-System loads their courses. | |
| **Alternative Flow** | NA | |
| **Exceptional Flow** | If one or more condition for attending class is not satisfied, system shows default browser pop-up with exception | |
| **Rules** | **Name of the Business Rule** | **Result(s)** |
| NA | NA |

**3.2.35.1. Functional Requirements-1** The system shall check if the instructor has any course.

**3.2.35.2. Functional Requirements-2** The system shall check if the instructor is logged in.

**3.2.35.3. Functional Requirements-3** Whether the result is successful or not the system shall inform the user by showing a message.

### 3.2.36. Creating Course

| **Use Case ID** | The UIS 4.0 UC. 36 | |
| --- | --- | --- |
| **Description** | Instructors can create course | |
| **Actors** | Instructor | |
| **Preconditions** | User must have login | |
| **Post Conditions** | Instructor create course | |
| **Main Success Scenario** | 1-User clicks “Dashboard” which in the navigation 2-User clicks “Create” button  3-User fills inputs  4-System create new course | |
| **Alternative Flow** | NA | |
| **Exceptional Flow** | If one or more condition for attending class is not satisfied, system shows default browser pop-up with exception | |
| **Rules** | **Name of the Business Rule** | **Result(s)** |
| NA | NA |

**3.2.36.1. Functional Requirements-1** The system shall check if the instructor has given required information.

**3.2.36.2. Functional Requirements-2** The system shall check if the instructor is logged in.

**3.2.36.3. Functional Requirements-3** Whether the result is successful or not the system shall inform the user by showing a message.

### 3.2.37. Updating Course

| **Use Case ID** | The UIS 4.0 UC. 37 | |
| --- | --- | --- |
| **Description** | Instructors can update course | |
| **Actors** | Instructor | |
| **Preconditions** | User must have login | |
| **Post Conditions** | Instructor update course | |
| **Main Success Scenario** | 1-User clicks “Dashboard” which is in the navigation 2-User clicks the “Update” button to update the course section.  3-User fills inputs  4-System updates course | |
| **Alternative Flow** | NA | |
| **Exceptional Flow** | If one or more condition for attending class is not satisfied, system shows default browser pop-up with exception | |
| **Rules** | **Name of the Business Rule** | **Result(s)** |
| NA | NA |

**3.2.37.1. Functional Requirements-1** The system shall check if the instructor is the owner of the course.

**3.2.37.2. Functional Requirements-2** The system shall check if the instructor is logged in.

**3.2.37.3. Functional Requirements-3** Whether the result is successful or not the system shall inform the user by showing a message.

### 3.2.38. Creating Online Class

| **Use Case ID** | The UIS 4.0 UC. 38 | |
| --- | --- | --- |
| **Description** | Instructors can start online class | |
| **Actors** | Instructor | |
| **Preconditions** | User must have login | |
| **Post Conditions** | Instructor planned online class | |
| **Main Success Scenario** | 1-User clicks “Dashboard” which in the navigation 2-User clicks the “Update” button to update the course section.  3-User fills inputs  4-System creates new online class course | |
| **Alternative Flow** | NA | |
| **Exceptional Flow** | If one or more condition for attending class is not satisfied, system shows default browser pop-up with exception | |
| **Rules** | **Name of the Business Rule** | **Result(s)** |
| NA | NA |

**3.2.38.1. Functional Requirements-1** The system shall check if the instructor has given a valid URL.

**3.2.38.2. Functional Requirements-2** The system shall check if the instructor is logged in.

**3.2.38.3. Functional Requirements-3** Whether the result is successful or not the system shall inform the user by showing a message.

### 3.2.39. Publishing Assignments

| **Use Case ID** | The UIS 4.0 UC. 39 | |
| --- | --- | --- |
| **Description** | Instructors can update course | |
| **Actors** | Instructor | |
| **Preconditions** | User must have login | |
| **Post Conditions** | Instructor create assignments | |
| **Main Success Scenario** | 1-User clicks “Create” which in assignment list page  2-User fills inputs 3-User clicks “Submit” button  4-System creates new assignments | |
| **Alternative Flow** | NA | |
| **Exceptional Flow** | If one or more condition for attending class is not satisfied, system shows default browser pop-up with exception | |
| **Rules** | **Name of the Business Rule** | **Result(s)** |
| NA | NA |

**3.2.39.1. Functional Requirements-1**

The system shall check inputs are filled.

**3.2.39.2. Functional Requirements-2**

The system shall check that the deadline is after the current day.

**3.2.39.3. Functional Requirements-3** Whether the result is successful or not the system shall inform the user by showing a message.

### 3.2.40. Viewing Submitted Assignments

| **Use Case ID** | The UIS 4.0 UC. 40 | |
| --- | --- | --- |
| **Description** | Instructors can view submitted assignments | |
| **Actors** | Instructor | |
| **Preconditions** | User must have login, user has been defined assignments | |
| **Post Conditions** | Instructor view submitted assignment | |
| **Main Success Scenario** | 1-User clicks “Detail” which in assignment list page  2-System load details of submitted assignments | |
| **Alternative Flow** | NA | |
| **Exceptional Flow** | If one or more condition for attending class is not satisfied, system shows default browser pop-up with exception | |
| **Rules** | **Name of the Business Rule** | **Result(s)** |
| NA | NA |

**3.2.40.1. Functional Requirements-1** The system shall check if the instructor has already defined any assignment.

**3.2.40.2. Functional Requirements-2** The system shall check if the instructor is logged in.

**3.2.40.3. Functional Requirements-3** Whether the result is successful or not the system shall inform the user by showing a message.

### 

### 

### 

### 3.2.41. Evaluating Submitted Assignments

| **Use Case ID** | The UIS 4.0 UC. 41 | |
| --- | --- | --- |
| **Description** | Instructors can evaluate assignments | |
| **Actors** | Instructor | |
| **Preconditions** | User must have login, user has open details of submitted assignment | |
| **Post Conditions** | Instructor evaluate assignment | |
| **Main Success Scenario** | 1-User fills grade input  2-User clicks “Evaluate”  3-System saves grade | |
| **Alternative Flow** | NA | |
| **Exceptional Flow** | If one or more condition for attending class is not satisfied, system shows default browser pop-up with exception | |
| **Rules** | **Name of the Business Rule** | **Result(s)** |
| NA | NA |

**3.2.41.1. Functional Requirements-1** The system shall check if the instructor has already defined any assignment.

**3.2.41.2. Functional Requirements-2** The system shall check if the instructor is logged in.

**3.2.41.3. Functional Requirements-3** Whether the result is successful or not the system shall inform the user by showing a message.

### 3.2.42. Creating Survey

| **Use Case ID** | The UIS 4.0 UC. 42 | |
| --- | --- | --- |
| **Description** | Instructors can create survey | |
| **Actors** | Instructor | |
| **Preconditions** | User must have login | |
| **Post Conditions** | Instructor create survey | |
| **Main Success Scenario** | 1-User clicks “Create” button in survey list  2-User clicks “Add Question”  3-User fill questions.  4-User clicks “Create” button  5-System creates survey | |
| **Alternative Flow** | NA | |
| **Exceptional Flow** | If one or more condition for attending class is not satisfied, system shows default browser pop-up with exception | |
| **Rules** | **Name of the Business Rule** | **Result(s)** |
| NA | NA |

**3.2.42.1. Functional Requirements-1**

The system shall check all the question has been filled

**3.2.42.2. Functional Requirements-2** The system shall check if the instructor is logged in.

**3.2.42.3. Functional Requirements-3** Whether the result is successful or not the system shall inform the user by showing a message.

### 3.2.43. Viewing Result of Survey

| **Use Case ID** | The UIS 4.0 UC. 43 | |
| --- | --- | --- |
| **Description** | Instructors can view survey results | |
| **Actors** | Instructor | |
| **Preconditions** | User must have login | |
| **Post Conditions** | Instructor view survey results | |
| **Main Success Scenario** | 1-User clicks “Surveys” button in navigation  2-System load surveys | |
| **Alternative Flow** | NA | |
| **Exceptional Flow** | If one or more condition for attending class is not satisfied, system shows default browser pop-up with exception | |
| **Rules** | **Name of the Business Rule** | **Result(s)** |
| NA | NA |

**3.2.43.1. Functional Requirements-1** The system shall check if the instructor has already defined any survey.

**3.2.43.2. Functional Requirements-2** The system shall check if the instructor is logged in.

**3.2.43.3. Functional Requirements-3** Whether the result is successful or not the system shall inform the user by showing a message.

### 3.2.44. Creating Exam

| **Use Case ID** | The UIS 4.0 UC. 44 | |
| --- | --- | --- |
| **Description** | Instructors can create exam | |
| **Actors** | Instructor | |
| **Preconditions** | User must have login, user has course | |
| **Post Conditions** | Instructor create exam | |
| **Main Success Scenario** | 1-User clicks “Exams” button in navigation  2-User clicks “Create” button in exam list  3-User fills question  4-User clicks “Create” button  5-System creates new exam | |
| **Alternative Flow** | NA | |
| **Exceptional Flow** | If one or more condition for attending class is not satisfied, system shows default browser pop-up with exception | |
| **Rules** | **Name of the Business Rule** | **Result(s)** |
| NA | NA |

**3.2.44.1. Functional Requirements-1** The system shall check if the instructor has already defined any course.

**3.2.44.2. Functional Requirements-2** The system shall check if the instructor is logged in.

**3.2.44.3. Functional Requirements-3** Whether the result is successful or not the system shall inform the user by showing a message.

### 3.2.45. Updating Exam

| **Use Case ID** | The UIS 4.0 UC. 45 | |
| --- | --- | --- |
| **Description** | Instructors can update exam | |
| **Actors** | Instructor | |
| **Preconditions** | User must have login, user must have course, user must have exam | |
| **Post Conditions** | Instructor updates exam | |
| **Main Success Scenario** | 1-User clicks “Exams” button in navigation  2-User clicks “Update” button in exam list  3-User change question or options  4-User clicks “Update” button  5-System creates new exam | |
| **Alternative Flow** | NA | |
| **Exceptional Flow** | If one or more condition for attending class is not satisfied, system shows default browser pop-up with exception | |
| **Rules** | **Name of the Business Rule** | **Result(s)** |
| NA | NA |

**3.2.45.1. Functional Requirements-1** The system shall check if the instructor is the owner of the exam.

**3.2.45.2. Functional Requirements-2** The system shall check if the instructor is logged in.

**3.2.45.3. Functional Requirements-3** Whether the result is successful or not the system shall inform the user by showing a message.

### 3.2.46. Uploading Notes

| **Use Case ID** | The UIS 4.0 UC. 46 | |
| --- | --- | --- |
| **Description** | Instructors can upload documents | |
| **Actors** | Instructor | |
| **Preconditions** | User must have login, user must have course | |
| **Post Conditions** | Instructor uploads documents | |
| **Main Success Scenario** | 1-User clicks “Add Documents” button in dashboard in course that user wants to add document  2-User selects document from OS default file picker pop-up  3-System uploads document | |
| **Alternative Flow** | NA | |
| **Exceptional Flow** | If one or more condition for attending class is not satisfied, system shows default browser pop-up with exception | |
| **Rules** | **Name of the Business Rule** | **Result(s)** |
| NA | NA |

**3.2.46.1. Functional Requirements-1** The system shall check if the instructor is the owner of course.

**3.2.46.2. Functional Requirements-2** The system shall check if the instructor is logged in.

**3.2.46.3. Functional Requirements-3** Whether the result is successful or not the system shall inform the user by showing a message.

### 3.2.47. Viewing Student List

| **Use Case ID** | The UIS 4.0 UC. 47 | |
| --- | --- | --- |
| **Description** | Instructors can view student list | |
| **Actors** | Instructor | |
| **Preconditions** | User must have login | |
| **Post Conditions** | Instructor view student list | |
| **Main Success Scenario** | 1-User clicks “Students” button in navigation  2-System loads student informations | |
| **Alternative Flow** | NA | |
| **Exceptional Flow** | If one or more condition for attending class is not satisfied, system shows default browser pop-up with exception | |
| **Rules** | **Name of the Business Rule** | **Result(s)** |
| NA | NA |

**3.2.47.1. Functional Requirements-1** The system shall check if the instructor is the owner of any course.

**3.2.47.2. Functional Requirements-2** The system shall check if the instructor is logged in.

**3.2.47.3. Functional Requirements-3** Whether the result is successful or not the system shall inform the user by showing a message.

### 3.2.48. Making Student Assessment

| **Use Case ID** | The UIS 4.0 UC. 48 | |
| --- | --- | --- |
| **Description** | Instructors can make student assessment | |
| **Actors** | Instructor | |
| **Preconditions** | User must have login, open student list | |
| **Post Conditions** | Instructor assessment a student | |
| **Main Success Scenario** | 1-User clicks “Students” button in navigation  2-User clicks “Assessment” button in navigation  3-User gives rating  4-User clicks “Submit” button  5-System saves student assessment | |
| **Alternative Flow** | NA | |
| **Exceptional Flow** | If one or more condition for attending class is not satisfied, system shows default browser pop-up with exception | |
| **Rules** | **Name of the Business Rule** | **Result(s)** |
| NA | NA |

**3.2.48.1. Functional Requirements-1** The system shall check if the instructor is the owner of any course.

**3.2.48.2. Functional Requirements-2** The system shall check if the instructor is logged in.

**3.2.48.3. Functional Requirements-3** Whether the result is successful or not the system shall inform the user by showing a message.

### 3.2.49. Making Student Critique

| **Use Case ID** | The UIS 4.0 UC. 49 | |
| --- | --- | --- |
| **Description** | Instructors can make student critique | |
| **Actors** | Instructor | |
| **Preconditions** | User must have login, open student list | |
| **Post Conditions** | Instructor make an critique to student | |
| **Main Success Scenario** | 1-User clicks “Students” button in navigation  2-User clicks “Critique” button in navigation  3-User gives input  4-User clicks “Submit” button  5-System saves student critique | |
| **Alternative Flow** | NA | |
| **Exceptional Flow** | If one or more condition for attending class is not satisfied, system shows default browser pop-up with exception | |
| **Rules** | **Name of the Business Rule** | **Result(s)** |
| NA | NA |

**3.2.49.1. Functional Requirements-1** The system shall check if the instructor is the owner of the course that the student is enrolled in.

**3.2.49.2. Functional Requirements-2** The system shall check if the instructor is logged in.

**3.2.49.3. Functional Requirements-3** Whether the result is successful or not the system shall inform the user by showing a message.

### 3.2.50. Viewing Student Contact Informations

| **Use Case ID** | The UIS 4.0 UC. 50 | |
| --- | --- | --- |
| **Description** | Instructors can view student contact information | |
| **Actors** | Instructor | |
| **Preconditions** | User must have login, open student list | |
| **Post Conditions** | Instructor view student contact information | |
| **Main Success Scenario** | 1-User clicks “Students” button in navigation  2-User clicks “Details” button in navigation  3-System loads student contact information | |
| **Alternative Flow** | NA | |
| **Exceptional Flow** | If one or more condition for attending class is not satisfied, system shows default browser pop-up with exception | |
| **Rules** | **Name of the Business Rule** | **Result(s)** |
| NA | NA |

**3.2.50.1. Functional Requirements-1** The system shall check if the instructor is the owner of the course that the student is enrolled in.

**3.2.50.2. Functional Requirements-2** The system shall check if the instructor is logged in.

**3.2.50.3. Functional Requirements-3** Whether the result is successful or not the system shall inform the user by showing a message.

### 3.2.51. Sending Announcement Notification

| **Use Case ID** | The UIS 4.0 UC. 51 | |
| --- | --- | --- |
| **Description** | Instructors can send announcement | |
| **Actors** | Instructor | |
| **Preconditions** | User must have login, user must have course | |
| **Post Conditions** | Instructor send announcement | |
| **Main Success Scenario** | 1-User clicks “Create Announcement” button in dashboard in course that user wants to make announcement  2-User fills input  3-System create announcement | |
| **Alternative Flow** | NA | |
| **Exceptional Flow** | If one or more condition for attending class is not satisfied, system shows default browser pop-up with exception | |
| **Rules** | **Name of the Business Rule** | **Result(s)** |
| NA | NA |

**3.2.51.1. Functional Requirements-1** The system shall check if the instructor is the owner of the course.

**3.2.51.2. Functional Requirements-2** The system shall check if the instructor is logged in.

**3.2.51.3. Functional Requirements-3** Whether the result is successful or not the system shall inform the user by showing a message.

## 3.3. Performance Requirements

## Requirements 1: The system shall load any page in less than 1 seconds. Requirements 2: The system shall return response when course adding in less than 1 second. Requirements 3: The system shall have at least %99.99 uptime. Requirements 4: The system shall not have any problem with screens that are 480px or 576px or 768px width. Requirements 5: The system shall try to reconnect the websocket. Requirements 6: The system shall establish the connection between websocket and itself in at most 1 second.

## 3.4. Logical Database Requirements

* **University information:** Id, name
  + **Frequency of use:** Every use
  + **Accessing capabilities:** University system’s API token, required information to reach university's own system
  + **Entities and relationship:** UIS 4.0 and university’s own system
  + **Data retention:** As long as the university uses the UIS 4.0
* **User information:** Id, e-mail, password, password salt, mobile phone, is student, is instructor, name , surname, required information fetched from university's database
  + **Frequency of use:** Every use
  + **Accessing capabilities:** University system’s API token, required information to reach university's own user information database
  + **Entities and relationship:** UIS 4.0, messages, notes, community sub-tables, exams, exam sub-tables, courses, course sub-tables, assessments, complaint, assignments, assignment sub-table, lesson attend request, patient reports, permission reports, survey sub-table and university’s database
  + **Data retention:** As long as the user exists in the university’s and UIS 4.0’s database
* **Message information:** Id, messages, sender, receiver, date & time
  + **Frequency of use:** At least 5 day a week
  + **Accessing capabilities:** None
  + **Entities and relationship:** UIS 4.0 and users (sender and receiver)
  + **Data retention:** As long as the user exists in the university’s and UIS 4.0’s database
* **Document information:** Id, name, documents, date & time
  + **Frequency of use:** At least 1 day a month
  + **Accessing capabilities:** None
  + **Entities and relationship:** UIS 4.0, user, courses
  + **Data retention:** As long as the user exists in the university’s and UIS 4.0’s database
* **Note information:** Id, name, notes, date & time
  + **Frequency of use:** At least 1 day a week
  + **Accessing capabilities:** None
  + **Entities and relationship:** UIS 4.0 and user, courses
  + **Data retention:** As long as the user exists in the university’s and UIS 4.0’s database
* **Community information:** Id, name, date & time
  + **Frequency of use:** At least 1 day a month
  + **Accessing capabilities:** None
  + **Entities and relationship:** UIS 4.0, user and university’s own system, sub-tables
  + **Data retention:** As long as the user and the community exists in the university’s and UIS 4.0’s database
* **Notification information:** Id, name, message, date & time
  + **Frequency of use:** At least 1 day a week
  + **Accessing capabilities:** None
  + **Entities and relationship:** UIS 4.0, user and university’s own system
  + **Data retention:** As long as the notification exists in the university’s database
* **Coursess:** Id, meeting url, date & time, required information fetched from course application
  + **Frequency of use:** At least 1 day a week
  + **Accessing capabilities:** None
  + **Entities and relationship:** UIS 4.0, users, online meeting application, coursess sub-tables
  + **Data retention:** Until semester end
* **Assignments:** Id, document url, deadline and related informations
  + **Frequency of use:** At most 1 day a term (3 - 6 months)
  + **Accessing capabilities:** None
  + **Entities and relationship:** UIS 4.0, users, coursess
  + **Data retention:** As long as the semester
* **Exams:** Id, start time and related informations
  + **Frequency of use:** At most 1 day a term (3 - 6 months)
  + **Accessing capabilities:** None
  + **Entities and relationship:** UIS 4.0, courses, exam sub-tables
  + **Data retention:** Until semester end
* **Surveys:** Id, course id and related informations
  + **Frequency of use:** At most 1 day a term (3 - 6 months)
  + **Accessing capabilities:** None
  + **Entities and relationship:** UIS 4.0, course, survey sub-tables
  + **Data retention:** Until semester end
* **Assessment:** Id, rate and related informations
  + **Frequency of use:** At most 1 day a term (3 - 6 months)
  + **Accessing capabilities:** None
  + **Entities and relationship:** UIS 4.0, users
  + **Data retention:** As long as the user exists in the university’s and UIS 4.0’s database
* **Complaint:** Id, rate and related informations
  + **Frequency of use:** At most 1 day a term (3 - 6 months)
  + **Accessing capabilities:** None
  + **Entities and relationship:** UIS 4.0, users
  + **Data retention:** As long as the user exists in the university’s and UIS 4.0’s database

## 3.5. Design Constraints

1. The design of the applications will be prepared according to the Microsoft Fluent Design Principles[2].
2. The design should have two versions: mobile view and desktop view.
3. The design should be suitable to Laws Of UX[3] principles.
4. The design should use the Ant Design[4] component.

## 3.6. Software System Attributes

The following items provide a partial list of system attributes that can serve as requirements that should be objectively verified.

### 3.6.1. Reliability

1. The system should store passwords as a result of a hash function which is calculated by users password and user unique password salt keyword.
2. The system should store sensitive information anonymously.

### 3.6.2. Availability

The system should have at least %99.99 uptime ratio in registration and course add-delete weeks.

### 3.6.3. Security

1. The system should be developed by considering secure programming principles which are defined in *Building Secure Software: How to Avoid Security Problems the Right Way. Maddison-Wesley Professional p. 528* [5]
2. The system should have authentication for operations and should be developed as noted on the *“What is ‘Authentication’”* by *The Economic Times* [6].

### 3.6.4. Maintainability

1. The system should be developed as microservices.
2. The system should be developed by using NPM and Nu-get Packages that will allow catching up the new versions.

### 3.6.5. Portability

The system should be deployed to cloud services and while changing the cloud service provider, the system should not be affected.

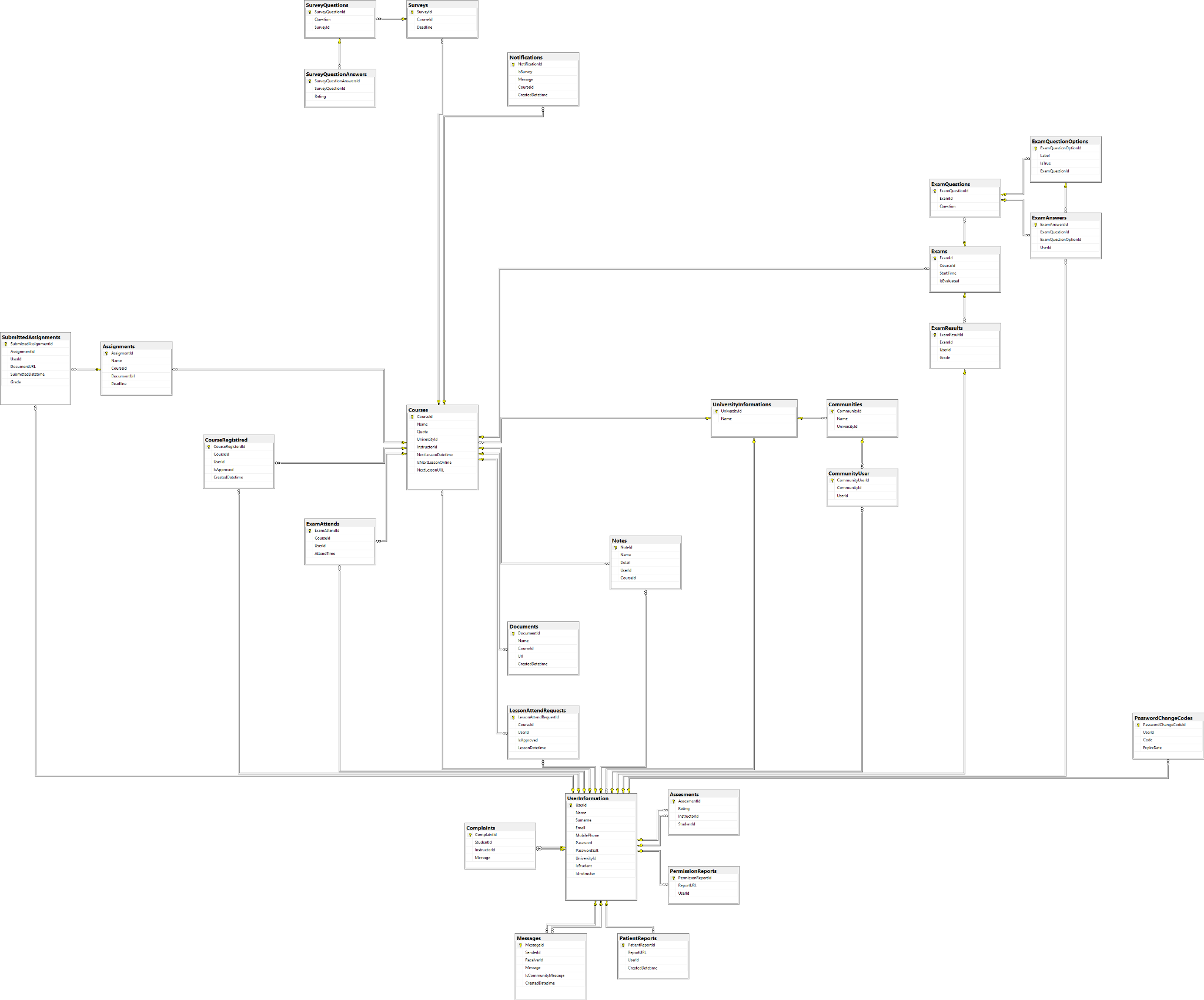
4. Appendices

## 4.1. Appendix A

### 4.1.1. Data Dictionary

| **Table Name** | **Keys** | **Field Name** | **Data Type and Length** | **Constraint** | **Description** |
| --- | --- | --- | --- | --- | --- |
| Assessments | PK | AssesmentId | int(10) | Not Null | Assessment id, Auto generated |
|  | Rating | int(10) | Not Null | Assessment Rating |
| FK | InstructorId | int(10) | Not Null | Instructor that gave assessment |
| FK | StudentId | int(10) | Not Null | Student that has being assessment |
| Assignments | PK | AssignmentId | int(10) | Not Null | Assignment id, Auto generated |
|  | Name | nvarchar(50) | Not Null | Assessment name |
| FK | CourseId | int(10) | Not Null | Course id of given assignment |
|  | DocumentUrl | nvarchar(MAX) | Not Null | Url of added document |
|  | Deadline | datetime(3) | Not Null | Deadline date & time of assignment |
| Communities | PK | CommunityId | int(10) | Not Null | Community id, Auto generated |
|  | Name | nvarchar(MAX) | Not Null | Name of the community |
| FK | UniversityId | int(10) | Not Null | University that community related |
| Community User | PK | CommunityUserId | int(10) | Not Null | Id of correspond to relation between community and student |
| FK | CommunityId | int(10) | Not Null | Community that student join |
| FK | UserId | int(10) | Not Null | Student that joined community |
| Complaints  Complaints | PK | ComplaintId | int(10) | Not Null | Complaint id, Auto generated |
| FK | StudentId | int(10) | Not Null | Student that has being complaint |
| FK | InstructorId | int(10) | Not Null | Instructor that gave assessment |
|  | Message | nvarchar(MAX) | Not Null | Given complaint message |
| Course Registered | PK | CourseRegisterdId | int(10) | Not Null | Id of correspond to relation between course and student |
| FK | CourseId | int(10) | Not Null | Course that student enrolled |
| FK | UserId | int(10) | Not Null | Student that enrolled to the course |
|  | IsApproved | bit | Not Null | Approval status of course registration |
|  | CreatedDatetime | datetime(3) | Not Null | Students course select date & time |
| Courses | PK | CourseId | int(10) | Not Null | Course id, Auto generated |
|  | Name | nvarchar(MAX) | Not Null | Name of course |
|  | Quota | int(10) | Not Null | Quota of course |
| FK | UniversityId | int(10) | Not Null | University that course related |
| FK | InstructorId | int(10) | Not Null | Instructor that give course |
|  | NextLessonDatetime | datetime(3) | Not Null | Date & time of courses next lesson |
|  | IsNextLessonOnline | bit | Not Null | Information of next course is online or not |
|  | NextLessonURL | nvarchar(MAX) | Null | If next lesson is online meeting url of that lesson |
| Documents | PK | DocumentId | int(10) | Not Null | Document id, Auto generated |
|  | Name | nvarchar(50) | Not Null | Name of the document |
| FK | CourseId | int(10) | Not Null | Course that document related |
|  | Url | nvarchar(MAX) | Not Null | Url of document can download |
|  | CreatedDatetime | datetime(3) | Not Null | Date & time of the document created |
| Exam Answers | PK | ExamAnswersId | int(10) | Not Null | Exam answer id, Auto generated |
| FK | ExamQuestionId | int(10) | Not Null | Question that answer given |
| FK | ExamQuestionOptionId | int(10) | Not Null | Option that chosed |
| FK | UserId | int(10) | Not Null | Student that give answer |
| Exam Attends  Exam Attends | PK | ExamAttendId | int(10) | Not Null | Exam attend id, Auto generated |
| FK | CourseId | int(10) | Not Null | Course that student attend |
| FK | UserId | int(10) | Not Null | Student that attend to exam |
|  | AttendTime | datetime(3) | Not Null | Date & time that student attend |
| Exam Question Options | PK | ExamQuestionOptionId | int(10) | Not Null | Exams Question Option id, Auto generated |
|  | Label | nvarchar(MAX) | Not Null | Content of option |
|  | IsTrue | bit | Not Null | Correctness status of option |
| FK | ExamQuestionId | int(10) | Not Null | Option that question related |
| Exam Questions | PK | ExamQuestionId | int(10) | Not Null | Exam Question id, Auto generated |
| FK | ExamId | int(10) | Not Null | Exam that question related |
|  | Question | nvarchar(MAX) | Not Null | Content of question |
| Exam Results | PK | ExamResultId | int(10) | Not Null | Exam Result id, Auto generated |
| FK | ExamId | int(10) | Not Null | Exam that result related |
|  | UserId | int(10) | Not Null | Student that result related |
|  | Grade | int(10) | Not Null | Grade of student got |
| Exams | PK | ExamId | int(10) | Not Null | Exam id, Auto generated |
| FK | CourseId | int(10) | Not Null | Course that exam related |
|  | StartTime | datetime(3) | Not Null | Date & time that exam will start |
|  | IsEvaluated | bit | Not Null | Evaluation status of exam |
| Lesson Attend Requests | PK | LessonAttendRequestId | int(10) | Not Null | Lesson Attend Request id, Auto generated |
| FK | CourseId | int(10) | Not Null | Course that request sent |
| FK | UserId | int(10) | Not Null | Student that send request |
|  | IsApproved | bit | Not Null | Approval status of lesson attend |
|  | LessonDatetime | datetime(3) | Not Null | Lessons date & time |
| Messages  Messages | PK | MessageId | int(10) | Not Null | Message id, Auto generated |
| FK | SenderId | int(10) | Not Null | User that message sent |
| FK | ReceiverId | int(10) | Not Null | User that message receiver |
|  | Message | nvarchar(MAX) | Not Null | The message written by the user |
|  | IsCommunityMessage | bit | Not Null | Status of that message sent by community |
|  | CreatedDatetime | datetime(3) | Not Null | Date & time of the message created |
| Notes | PK | NoteId | int(10) | Not Null | Note id, Auto generated |
|  | Name | nvarchar(50) | Not Null | Name of the note |
|  | Detail | nvarchar(MAX) | Null | The note written by the user |
| FK | UserId | int(10) | Not Null | Student that owner of note |
| FK | CourseId | int(10) | Not Null | Course that related with note |
| Notifications | PK | NotificationId | int(10) | Not Null | Notification id, Auto generated |
|  | IsSurvey | bit | Not Null | Status of that notification sent as survey |
|  | Message | nvarchar(MAX) | Not Null | The notification written by the user |
| FK | CourseId | int(10) | Not Null | Course that notification content related |
|  | CreatedDatetime | datetime(3) | Not Null | Date & time of the notification created |
| Password Change Codes | PK | PasswordChangeCodeId | int(10) | Not Null | Password Change Code id, Auto generated |
| FK | UserId | int(10) | Not Null | User that request the code |
|  | Code | int(10) | Not Null | Code that need to giving when password change |
|  | ExpireDate | datetime(3) | Not Null | Expire date of code |
| Patient Reports | PK | PatientReportId | int(10) | Not Null | Patient Report id, Auto generated |
|  | ReportURL | nvarchar(MAX) | Not Null | Report url that can be accessed |
| FK | UserId | int(10) | Not Null | Student that upload report |
|  | CreatedDatetime | datetime(3) | Not Null | Date & time that report uploaded |
| Permission Reports  Permission Reports | PK | PermissonReportId | int(10) | Not Null | Permission Report id, Auto generated |
|  | ReportURL | nvarchar(MAX) | Not Null | Report url that can be accessed |
| FK | UserId | int(10) | Not Null | Student that upload report |
| Submitted Assignments | PK | SubmittedAssignmentId | int(10) | Not Null | Submitted Assignment id, Auto generated |
| FK | AssignmentId | int(10) | Not Null | Assignment that submitted |
| FK | UserId | int(10) | Not Null | Student that submit assignment |
|  | DocumentURL | nvarchar(MAX) | Not Null | Submitted assignment url that can be accessed |
|  | SubmittedDatetime | datetime(3) | Not Null | Date & time that assignment submitted |
|  | Grade | int(10) | Not Null | Grade of student got |
| Survey Question Answers | PK | SurveyQuestionAnswersId | int(10) | Not Null | Survey Question Answers id, Auto generated |
| FK | SurveyQuestionId | int(10) | Not Null | Survey Question that answer given |
|  | Rating | int(10) | Not Null | Rating that given to the question |
| Survey Questions | PK | SurveyQuestionId | int(10) | Not Null | Survey Question id, Auto generated |
|  | Question | nvarchar(MAX) | Not Null | Content of survey question |
| FK | SurveyId | int(10) | Not Null | Survey that question related |
| Surveys | PK | SurveyId | int(10) | Not Null | Survey id, Auto generated |
| FK | CourseId | int(10) | Not Null | Course that survey related |
|  | Deadline | datetime(3) | Not Null | Deadline date & time of survey |
| University Informations | PK | UniversityId | int(10) | Not Null | University id, Auto generated |
|  | Name | nvarchar(50) | Not Null | Name of the University |
| User Information  User Information | PK | UserId | int(10) | Not Null | User id, Auto generated |
|  | Name | nvarchar(50) | Not Null | Name of user |
|  | Surname | nvarchar(50) | Not Null | Surname of user |
|  | Email | nvarchar(50) | Not Null | Any user email |
|  | MobilePhone | nvarchar(50) | Not Null | Mobile phone of user |
|  | Password | nvarchar(MAX) | Not Null | Login password for user |
|  | PasswordSalt | nvarchar(50) | Not Null | Password salt that password will encrypted with |
| FK | UniversityId | int(10) | Not Null | University that user related |
|  | IsStudent | bit | Not Null | Status of users being student |
|  | IsInstructor | bit | Not Null | Status of users being instructor |

### 4.1.2. Entity Relationship Diagram

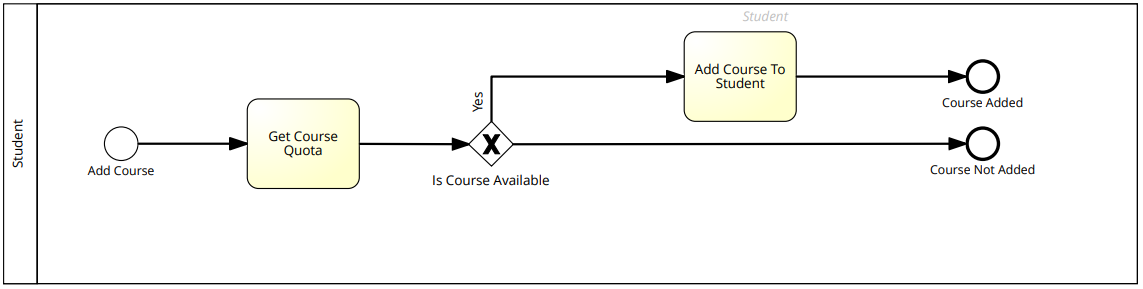


\*Full View of Entity Relationship Diagram[7]

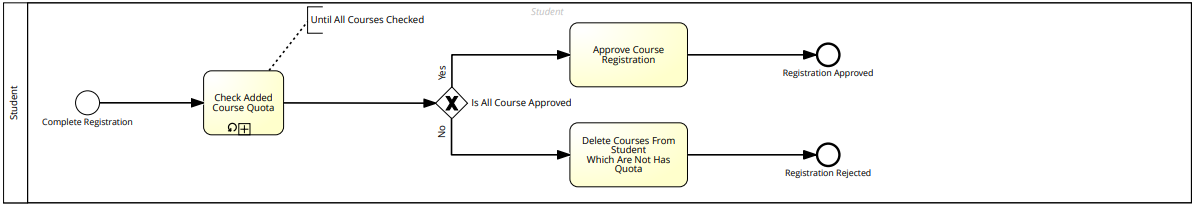
## 4.2. Appendix B

### 4.2.1. Process Models of the Designed To-be System[8]

#### 4.2.1.1. Add Course

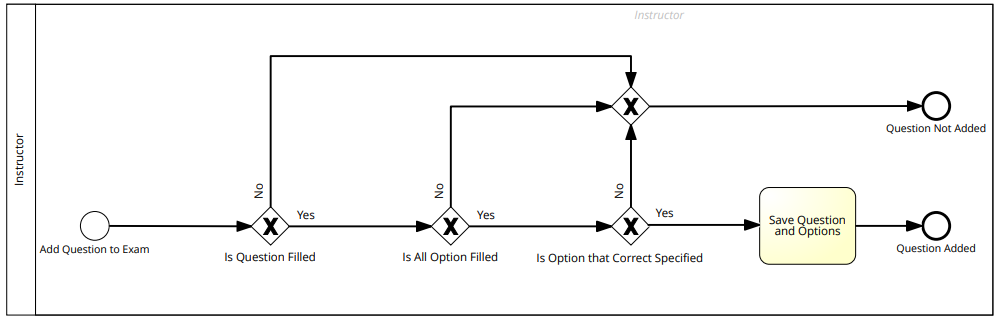


#### 4.2.1.2. Course Registration



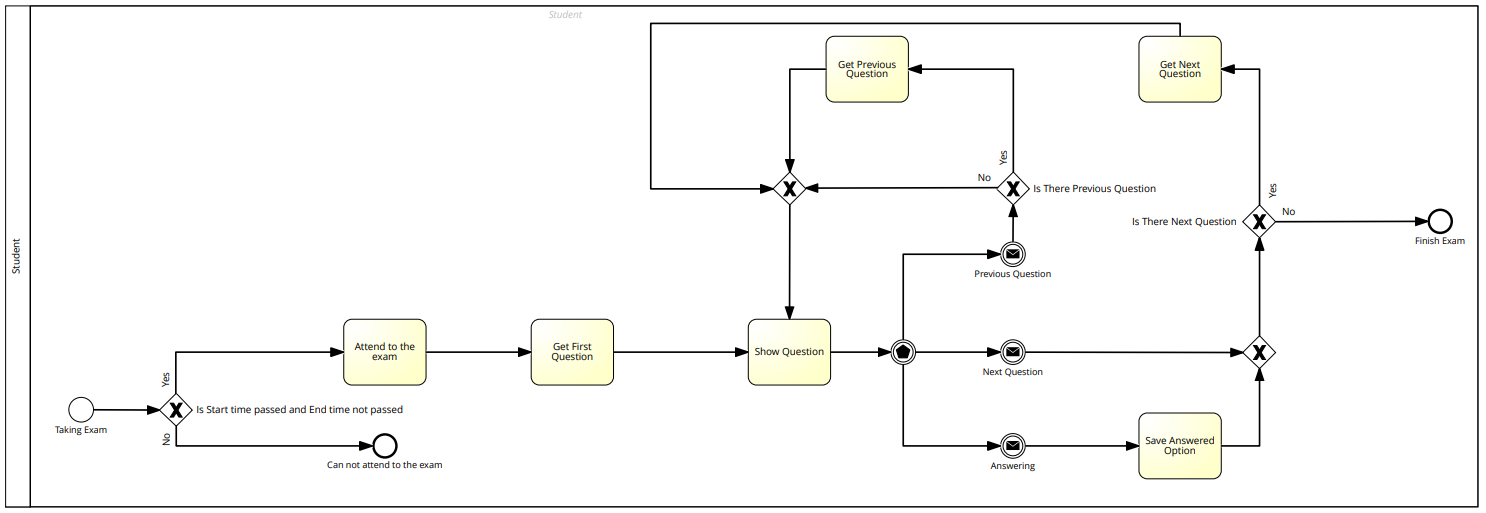
#### 4.2.1.2.1. Check Added Course Quota

#### 4.2.1.3. Add Question To Exam



#### 4.2.1.4. Create Exam

#### 4.2.1.5. Taking Exam

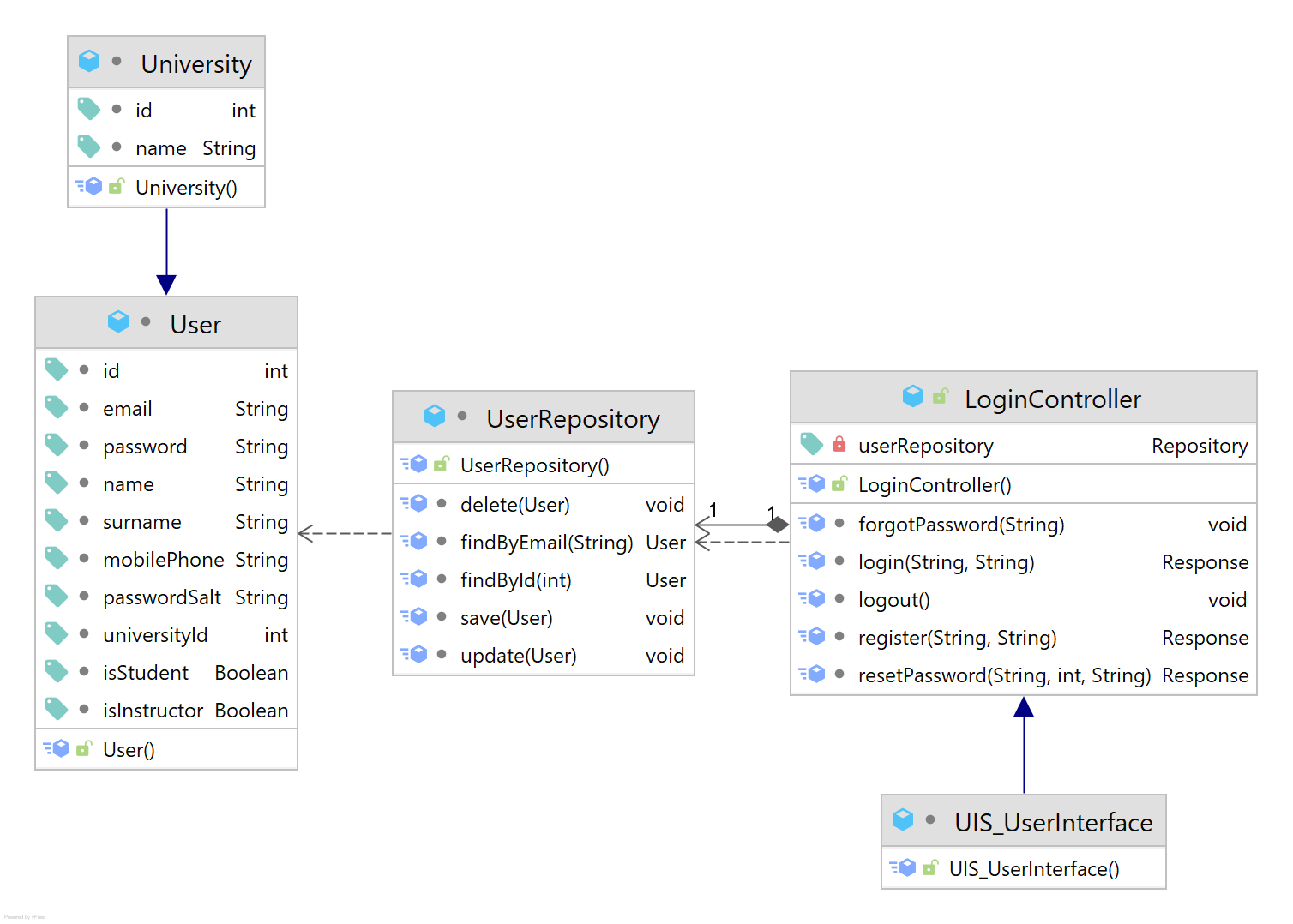


#### 4.2.1.6. Survey Filling Out

## 4.3. Appendix C

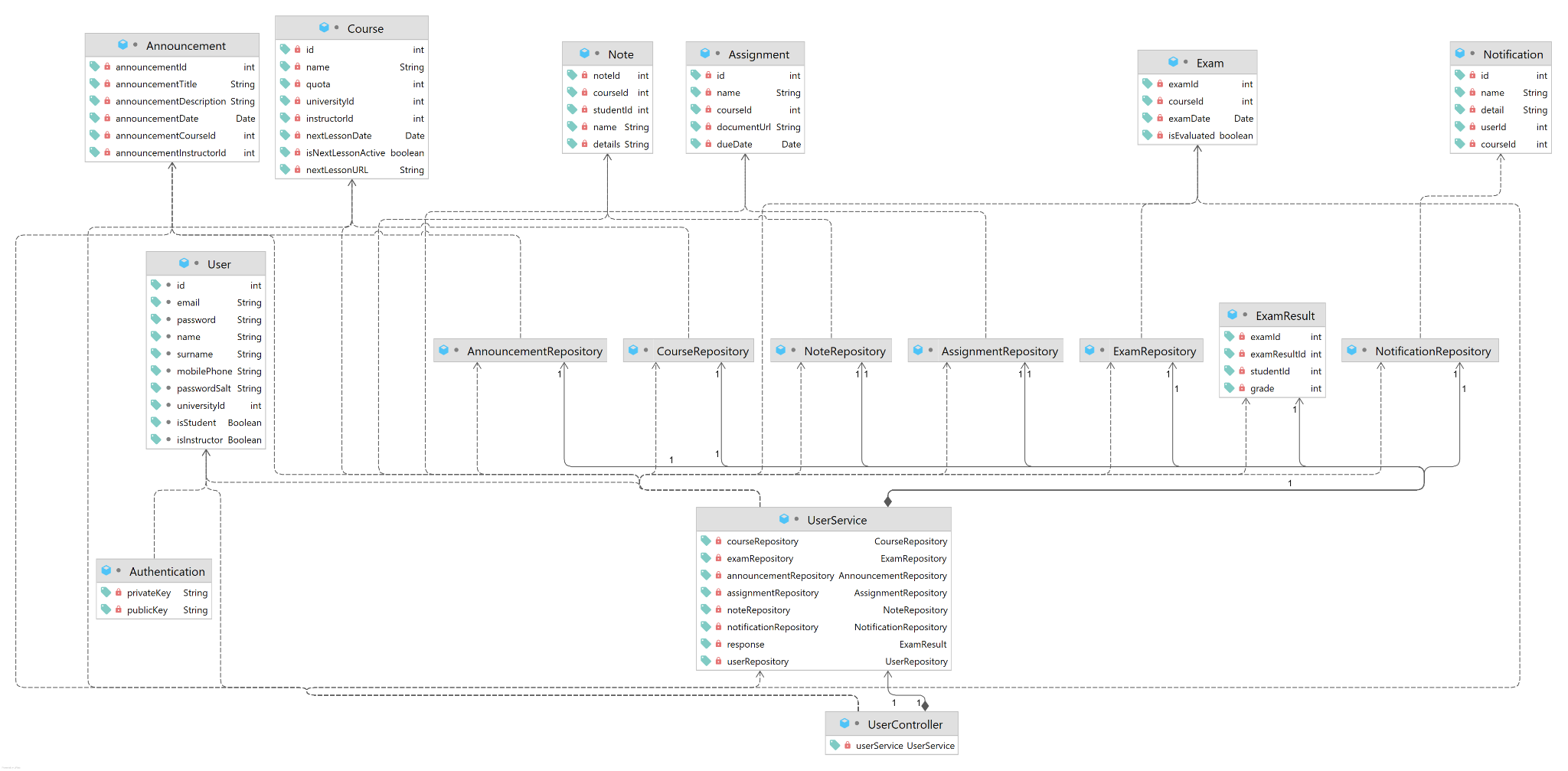
### 4.3.1 Class Diagrams

#### 4.3.1.1 Login Class diagram



#### 

#### 4.3.1.2 System Class Diagram



#### 

5. Lessons Learned Report

**Emre Hancı:** While preparing this document, I learned which contents should be covered in SRS and Vision and Scope documents. I am learning to use Signavio for creating Business Process Models. I gained experience in developing mock screens with Mockplus, also I gained experience in projects business domain while specifying the requirements. I learned how to export Entity Relationship Diagrams and Data Dictionaries via SQL Management Studio.

**Eren Kumru:** During this project I have learned how to and what to write in the Vision and Scope document and Software Requirements Specifications document. I have also learned the importance of some of the soft skills such as time management, researching, teamwork, communication, and leadership. I also realized both the difficulty and the importance of creating various project documents and improved myself on both ideation and documentation phases. I gained experience on managing a team, managing a project process, and making presentations. I discovered different aspects of my leadership abilities I had never explored before. After realizing that, I think I can say that I could have been more impactful as a leader. After learning all of these, I have more confidence in myself both as a person and as a leader.

**Mücahit Veli Cumart:** During the completion of this project, I gained experience in many subjects that are used in the field of software requirements and need to be known. I learned what stages are passed in determining the requirements of a project and these stages.I met mock screens and learned its basics by doing research on it.

I improved myself by learning new information about how use cases are created, data dictionaries, and diagrams (E-R, System Class Diagram,Use Case Diagram). In addition, I gained experience in group work and time management in a project and how to work in sync with teammates.

**Muhammed Fidan:** In general, preparing a SRS document allowed me to see a project's overview in a better way. This especially can help us in the future with big projects when we start our business life. Having such a document can be used in the foundation and continuation of the project.

Even though I was familiar with most of the SRS document contents I've learned the importance of mock-up screens. That impressed me the most and preparing these screens beforehand can especially help for front-end development. Which usually takes a lot of time to develop.

**Mehmet Yıldız:** At first, we wrote the Vision and Scope document. While writing that document, I learned how to ensure that mutual decisions and considerations are made between the company, the software team, and the customer. In this way, before the design and development phases of the software started, we laid out the steps and requirements of the project in these documents simply and clearly. This has been a very informative study for me. While preparing the Software Requirements Specifications document, I learned how to detail all the requirements of a project step by step. I learned the importance of the use cases used in the project and the benefits of these use cases in the design and development process of the project. I also understood the importance of mock designs in the project. I saw the importance of the concepts of data dictionary and entity relations in the project preparation phase, the usefulness of diagrams in starting the project design, etc. Therefore, while creating SRS documents, I discovered that the skeleton structure of the project was created, it has a universal language and allows the projects to be developed robustly.