**3.2 Instructor Functional Requirements**

* **Register**

Instructors should be registered to do other functions.

* **Authentication**

Instructor logs on from home page as well. Instructor can log on to the system with SSN number and password.

* **Create New Lecture**

Instructor adds courses he/she gives. Course name and class list is specified here.

* **Create Class**

Instructor creates class list of the course. Thus only the students who takes this course will be able to join this course’s exam.

* **Add Student**

This function is called after create class function. Adds student to class list of course.

* **Delete Student**

This function is called after create class function as well. Deletes student from class list if necessary like when student drops course or etc.

* **List Student**

Shows current class list of course.

* **Create Exam**

Creating exam with this function. Adding and deleting questions with other functions. Questions are updatable after publish but before exam start date and time.

* **Add Question**

Adds a new question and its answer to exam.

* **Delete Question**

Deletes a question from exam.

* **Update Exam**

Updates exam date, questions, or answers before examination date.

* **Update Question**

To update question itself or its answer.

* **Remove Exam**

This function is used for delete whole exam.

* **Remove Lecture**

To remove lecture when necessary.

* **List Student Results**

To list all of the results of students on the list for the lecture.

* **LogOff**

Instructors can logging off with this function as well.

**3.4 System Models**

**3.4.1 Instructor Scenarios**

## Scenario 1

|  |
| --- |
| **Scenario name:** Instructor Register |
| **Participant actor instances:** **Okan:**Instructor |
| **Flow of events:**   1. Firstly Okan enters [www.ES.com](http://www.ES.com) on browser. 2. ES welcomes Okan with main page. Okan sees register button below the login fields and clicks register. 3. He will be directed to register page after click for choose register as an instructor or student first. 4. After choosing register as an instructor he will be directed to instructor register fields to fill. 5. Okan fills field with information needed. 6. Okan clicks done. 7. Account is created. |

## Scenario 2

|  |
| --- |
| **Scenario name:** Instructor Login |
| **Participant actor instances:** **Okan:**Instructor |
| **Flow of events:**   1. Firstly Okan enters [www.ES.com](http://www.ES.com) on browser. 2. ES welcomes Okan with main page. Okan fills the login fields and clicks login. 3. If SSN and password matches with database he accesses his account. |

## Scenario 3

|  |
| --- |
| **Scenario name:** Create New Lecture |
| **Participant actor instances:** **Müge:**Instructor |
| **Flow of events:**   1. This event starts with Müge’s click to the Create a new lecture button. 2. Müge is directed to the create lecture page. 3. Müge fills the lecture informations. 4. Müge clicks done. 5. Lecture is added to the Müge’s database. |

## Scenario 4

|  |
| --- |
| **Scenario name:** Create Class(Empty) |
| **Participant actor instances:** **Müge:**Instructor |
| **Flow of events:**   1. This event starts with Müge’s click to the Create Class button. 2. Müge is directed to the Create Class page. 3. Müge clicks done without adding any student. 4. An empty class is created for lecture. |

## Scenario 5

|  |
| --- |
| **Scenario name:** Create Class |
| **Participant actor instances:** **Müge:**Instructor |
| **Flow of events:**   1. This event starts with Müge’s click to the Create Class button. 2. Müge is directed to the Create Class page. 3. Müge clicks add student button and can continue add or delete. 4. Müge clicks done. 5. A class list is constructed and setted to the database. |

## Scenario 6

|  |
| --- |
| **Scenario name:** Add student |
| **Participant actor instances:** **Müge:**Instructor |
| **Flow of events:**   1. This event starts with Müge’s click to the add Student button after opened Class page. 2. System shows to Müge fields to fill with student info. 3. Müge fills fields with the student information which is taking course. 4. Müge clicks done. 5. A new student added to class list and setted to the database. |

## Scenario 7

|  |
| --- |
| **Scenario name:** Delete student |
| **Participant actor instances:** **Müge:**Instructor |
| **Flow of events:**   1. This event starts with Müge’s click to the delete Student button after opened Class page. 2. Müge selects a student. 3. Müge clicks delete student button. 4. Student is deleted from class list , database updated. |

## Scenario 8

|  |
| --- |
| **Scenario name:** List student |
| **Participant actor instances:** **Müge:**Instructor |
| **Flow of events:**   1. This event starts with Müge’s click to the list Student button after opened Class page. 2. System shows to Müge class list. |

## Scenario 9

|  |
| --- |
| **Scenario name:** Create Exam |
| **Participant actor instances:** **Müge:**Instructor |
| **Flow of events:**   1. This flow starts with Müge’s click to the create exam button. 2. System shows to Müge fields to fill with exam info. 3. Müge fills fields with the exam information which is date, time, etc. 4. Then Müge adds questions. 5. Müge clicks done. |

## Scenario 10

|  |
| --- |
| **Scenario name:** Add Question |
| **Participant actor instances:** **Müge:**Instructor |
| **Flow of events:**   1. This event starts with Müge’s click to the add question button. 2. System directs Müge to question adding page. 3. Müge first selects a question type. 4. Müge adds question itself. 5. Müge adds answer(s). 6. Müge clicks done. 7. System adds question to the exam and database. |

## Scenario 11

|  |
| --- |
| **Scenario name:** Delete Question |
| **Participant actor instances:** **Müge:**Instructor |
| **Flow of events:**   1. This event starts with Müge’s click to the delete question button. 2. System directs Müge to exam page. 3. Müge first selects a question. 4. Müge clicks delete button. 5. System deletes question from the exam and database. |

## Scenario 12

|  |
| --- |
| **Scenario name:** Update Exam |
| **Participant actor instances:** **Müge:**Instructor |
| **Flow of events:**   1. This event starts with Müge’s click to the update exam button. 2. System directs Müge to exam page. 3. Müge updates exam informations (date, time etc.). 4. Müge clikcs done. 5. System updates exam information and database. |

## Scenario 13

|  |
| --- |
| **Scenario name:** Update Question |
| **Participant actor instances:** **Müge:**Instructor |
| **Flow of events:**   1. This event starts after Müge’s selecting a question. 2. System opens Müge to question page. 3. Müge clicks selects update question. 4. Müge updates question or answer(s). 5. Müge clicks done. 6. System updates question and database. |

## Scenario 14

|  |
| --- |
| **Scenario name:** Remove Exam |
| **Participant actor instances:** **Müge:**Instructor |
| **Flow of events:**   1. This event starts after Müge clicks remove exam button. 2. System deletes exam from lecture and database. |

## Scenario 15

|  |
| --- |
| **Scenario name:** Remove Lecture |
| **Participant actor instances:** **Müge:**Instructor |
| **Flow of events:**   1. Flow starts when Müge clicks to the remove lecture button. 2. System removes lecture from Müge’s account and database. |

## Scenario 16

|  |
| --- |
| **Scenario name:** List students results |
| **Participant actor instances:** **Müge:**Instructor |
| **Flow of events:**   1. Flow starts with Müge’s click to show results button. 2. System opens the student grade list for the lecture to Müge. |

## Scenario 17

|  |
| --- |
| **Scenario name:** InstructorLogoff |
| **Participant actor instances:** **Müge:**Instructor |
| **Flow of events:**   1. Flow starts with Müge’s click to log off button. 2. System directs Müge to the main page. 3. Müge logged off. |

**3.4.2 Instructor Use Cases**

## Use Case 1

|  |
| --- |
| **Use Case Name:** Create New Lecture |
| **Participant actor instances:** Instructor |
| **Flow of events:**   1. This event starts with Instructor’s click to the Create a new lecture button. 2. Instructor is directed to the create lecture page. 3. Instructor fills the lecture informations. 4. Instructor clicks done. 5. Lecture is added to the database. |
| **Entry Condition:** User should be logged on as instructor. |
| **Exit Condition:** Lecture is created. Or cancelled if user left page without click done |
| **Alternative flow:**  **4a.** Instructor lefts page without click done.   1. System cancelled function automatically. |
| **Quality Requirement:**  **1.**  System response to the actor arrives in a few seconds. |

## Use Case 2

|  |
| --- |
| **Use Case Name:** Create Class |
| **Participant actor instances:** Instructor |
| **Flow of events:**   1. This event starts with Instructor’s click to the Create Class button. 2. Instructor is directed to the Create Class page. 3. Instructor clicks add student button and can continue add or delete. 4. Instructor clicks done 5. A class list is constructed and setted to the database. |
| **Entry Condition:** User should be logged on as instructor. And also should already created a lecture. |
| **Exit Condition:** A class is created. Or alternatively empty one is created. |
| **Alternative flow:**  **3a.** Instructor clicks done.  **1.**System created an empty class list. |
| **Quality Requirement:**  **1.** System response to the actor arrives in a few seconds. |

## Use Case 3

|  |
| --- |
| **Use Case Name:** Add student |
| **Participant actor instances:** Instructor |
| **Flow of events:**   1. This event starts with Instructor’s click to the add Student button after opened Class page. 2. System shows to Instructor fields to fill with student info. 3. Instructor fills fields with the student information which is taking course. 4. Instructor clicks done. 5. A new student added to class list and setted to the database. |
| **Entry Condition:** User should be logged on as instructor. And also should already created a lecture and class. |
| **Exit Condition:** A student is added to class list. |
| **Quality Requirement:**  **1.** System response to the actor arrives in a few seconds. |

## Use Case 4

|  |
| --- |
| **Use Case Name:** Delete student |
| **Participant actor instances:** Instructor |
| **Flow of events:**   1. This event starts with Instructor’s click to the delete Student button after opened Class page. 2. Instructor selects a student. 3. Instructor clicks delete student button. 4. Student is deleted from class list , database updated. |
| **Entry Condition:** User should be logged on as instructor. And also should already created a lecture and class. |
| **Exit Condition:** Selected student(s) is deleted from class list. |
| **Quality Requirement:**  **1.** System response to the actor arrives in a few seconds. |

## Use Case 5

|  |
| --- |
| **Use Case Name:** List student |
| **Participant actor instances:** Instructor |
| **Flow of events:**   1. This event starts with Instructor’s click to the list Student button after opened Class page. 2. System shows to Instructor class list. |
| **Entry Condition:** User should be logged on as instructor. And also should already created a lecture and class. |
| **Quality Requirement:**   1. System response to the actor arrives in a few seconds. 2. List is showed in alphabetic order. |
| **Exit Condition:** A student is added to class list. |

## Use Case 6

|  |
| --- |
| **Use Case Name:** Create Exam |
| **Participant actor instances:** Instructor |
| **Flow of events:**   1. This flow starts with Instructor’s click to the create exam button. 2. System shows to Instructor fields to fill with exam info. 3. Instructor fills fields with the exam information which is date, time, etc. 4. Then Instructor adds questions. 5. Instructor clicks done. |
| **Entry Condition:** User should already created a lecture. |
| **Exit Condition:** An exam is created. |
| **Quality Requirement:**   1. System response to the actor arrives in a few seconds. 2. Instructor can select day from a calendar to make his job easy. |

## Use Case 7

|  |
| --- |
| **Use Case Name:** Add Question |
| **Participant actor instances:** Instructor |
| **Flow of events:**   1. This event starts with Instructor’s click to the add question button. 2. System directs Instructor to question adding page. 3. Instructor first selects a question type. 4. Instructor adds question itself. 5. Instructor adds answer(s). 6. Instructor clicks done. 7. System adds question to the exam and database. |
| **Entry Condition:** User should be created an exam. |
| **Exit Condition:** A question is added to exam. |
| **Alternative flow:**  **6a.** Instructor doesn’t click done and left page.  **1.**System cancelled add student function. |
| **Quality Requirement:**  **1.** System response to the actor arrives in a few seconds. |

## Use Case 8

|  |
| --- |
| **Use Case Name:** Delete Question |
| **Participant actor instances:** Instructor |
| **Flow of events:**   1. This event starts with Instructor’s click to the delete question button. 2. System directs Instructor to exam page. 3. Instructor first selects a question. 4. Instructor clicks delete button. 5. System deletes question from the exam and database. |
| **Entry Condition:** User should be created an exam. |
| **Exit Condition:** A question is deleted from exam. |
| **Quality Requirement:**  **1.** System response to the actor arrives in a few seconds. |

## Use Case 9

|  |
| --- |
| **Use Case name:** Update Exam |
| **Participant actor instances:** Instructor |
| **Flow of events:**   1. This event starts with Instructor’s click to the update exam button. 2. System directs Instructor to exam page. 3. Instructor updates exam informations (date, time etc.). 4. Instructor clicks done. 5. System updates exam information and database. |
| **Entry Condition:** User should be created an exam. |
| **Exit Condition:** Exam is updated. |
| **Quality Requirement:**  **1.** System response to the actor arrives in a few seconds. |

## Use Case 10

|  |
| --- |
| **Use Case Name:** Update Question |
| **Participant actor instances:** Instructor |
| **Flow of events:**   1. This event starts after Instructor’s selecting a question. 2. System opens Instructor to question page. 3. Instructor clicks update question. 4. Instructor updates question or answer(s). 5. Instructor clicks done. 6. System updates question and database. |
| **Entry Condition:** User should be created an exam. |
| **Exit Condition:** Question is updated. |
| **Quality Requirement:**  **1**. System response to the actor arrives in a few seconds. |

## Use Case 11

|  |
| --- |
| **Use Case Name:** Remove Exam |
| **Participant actor instances:** Instructor |
| **Flow of events:**   1. This event starts after Instructor clicks remove exam button. 2. System deletes exam from lecture and database. |
| **Entry Condition:** User should be created an exam. |
| **Exit Condition:** Exam is removed. |
| **Quality Requirement:**  **1.** System response to the actor arrives in a few seconds. |

## Use Case 12

|  |
| --- |
| **Use Case Name:** Remove Lecture |
| **Participant actor instances:** Instructor |
| **Flow of events:**   1. Flow starts when Instructor clicks to the remove lecture button. 2. System removes lecture from Instructor’s account and database. |
| **Entry Condition:** User should be created a lecture. |
| **Exit Condition:** Lecture is removed ,database changed. |
| **Quality Requirement:**  **1.** System response to the actor arrives in a few seconds. |

## Use Case 13

|  |
| --- |
| **Use Case Name:** List students results |
| **Participant actor instances:** Instructor |
| **Flow of events:**   1. Flow starts with Instructor’s click to show results button. 2. System opens the student grade list for the lecture to Instructor. |
| **Entry Condition:** Instructor should be logged in. |
| **Exit Condition:** Instructor has seen grade list of the exam. |
| **Quality Requirement:**  **1.** System response to the actor arrives in a few seconds. |