# Use Case UC1: Sign Up

**Primary Actor:** Student

**Stakeholders and Interests:**

* Student: wants easily understandable questions, fast response, no system errors.

**Preconditions**:

* Student is not registered. (s: StartEvent.condition <- *CatchEvent* <- *FlowNode*)

**Success Guarantee (Postcondition):**

* Student is registered. (e: EndEvent.condition <- *ThrowEvent* <- *FlowNode*)

**Main Success Scenario (or Basic Flow):**

(s: StartEvent)

<s -> 1: SequenceFlow>

1. Student visits system home page. (1: Activity <- *FlowNode*)

<1 -> 2: SequenceFlow>

2. System shows home page with login form and sign up button. (2: Activity <- *FlowNode*)

<2 -> 3: SequenceFlow>

3. Student clicks sign up button. (3: Activity <- *FlowNode*)

<3 -> 4: SequenceFlow>

4. System shows sign up form. (4: Activity <- *FlowNode*)

<4 -> 5: SequenceFlow>

5. Student fills the form and clicks Submit button. (5: Activity <- *FlowNode*)

<5 -> 6:SequenceFlow>

6. System registers new student account and shows student’s home page. (6: Activity <- *FlowNode*)

<6 -> e: SequenceFlow>

(e: EndEvent)

**Extensions (or Alternative Flows):**

\*a. At any time, System fails: To support recovery, ensure all transaction sensitive state and events can be recovered from any step of the scenario. (\*a\_start: StartEvent)

<\*a -> 1: SequenceFlow>

1. Student restarts System and requests recovery of prior state. (\*a\_1: Activity <- *FlowNode*)

<\*a\_1 -> 2: SequenceFlow>

2. System reconstructs prior state. (\*a\_2: Activity <- *FlowNode*)

<\*a\_2 -> \*a\_2a: ConditionalSequenceFlow>

2a. System detects anomalies preventing recovery: (\*a\_2a: Activity <- *FlowNode*)

<2a -> \*a\_2a\_1: SequenceFlow>

1. System signals error to the Student. (\*a\_2a\_1: Activity <- *FlowNode*)

<\*a\_2a\_1 -> \*a\_2a\_2: SequenceFlow>

2. System records the error. (\*a\_2a\_2: Activity <- *FlowNode*)

<\*a\_2a\_1 -> \*a\_2a\_2: SequenceFlow>

3. System enters a clean state. (\*a\_2a\_3: Activity <- *FlowNode*)

<\*a\_2a\_1 -> \*a\_end: SequenceFlow>

*Dummy End Event* (\*a\_end: EndEvent)

5.

<5 -> 5a: ConditionalSequenceFlow>

5a. Student enters invalid form value.

<SequenceFlow>

1. System shows errors and request to Student to refill the form.

<SequenceFlow>

2. Student refills the form and clicks Submit button again.

<SequenceFlow>

6.

6.

<6 -> 6.a ConditionalSequenceFlow>

6a. System detects failure to communicate with server:

<SequenceFlow>

1. System signals error and rejects the request.

(e: EndEvent)

*PS: Generic Docx parser için öneriler: Doküman bir tree olacak. Node’lardan oluşan. Heading stillerine göre ve indention’lara göre hiyerarşi kurulacak. Noktalara dikkat edilecek. Her bir node’a ID verilecek ve önceki node ve sonraki node bilgileri tutulacak. Böylelikle araya node sokabileceğiz. Node’lara eğer : ile ayrılmış ise bu isim whitespace’lerden temizlenerek bir* ***key/label*** *olacak. Satırbaşından sonra ya da indention’dan sonra “/t\*. \*” (uygun bir reqex yaz) şeklinde bir yapı var ise bu da* ***key/label*** *olarak name’ sağdan soldan temizlenerek atılacak. Sonra gelen her bir cümle node olacak ve bağlı liste şeklinde birbirine bağlanacak.*

*PS: Model olarak Traceability Matrix üretebiliriz. Referansları bularak. Bir view olur Knowledge base’den*

*http://www.ibm.com/developerworks/rational/library/content/03July/1000/1164/1164\_fig5.jpg*

*http://www.ibm.com/developerworks/rational/library/835.html*

# Use Case UC2: Sign In

**Primary Actor:** Student

**Stakeholders and Interests:**

* Student: Wants simple user interface, fast response, no system errors.

**Preconditions:**

* Student is registered.

**Success Guarantee (Postcondition):**

* Student is logged in.

**Main Success Scenario (or Basic Flow):**

1. Student visits system home page.

2. System shows home page with login form and sign up button.

3. Student enters his/her username and password then click login button.

4. System shows Student’s home page.

**Extensions (or Alternative Flows):**

\*a. At any time, System fails: To support recovery, ensure all transaction sensitive state and events can be recovered from any step of the scenario.

1. Student restarts System and requests recovery of prior state.

2. System reconstructs prior state.

2a. System detects anomalies preventing recovery:

1. System signals error to the Student, records the error, and enters a clean state.

3a. Student enters invalid username or password.

1. System shows errors and request to Student to retry.

2. Student enters his/her username and password.

4a. System detects failure to communicate with server:

1. System signals error and rejects the request.

# Use Case UC3: File Download

**Primary Actor:** Student

**Stakeholders and Interests:**

* Student: Wants simple user interface, fast response, no system errors.

**Preconditions**: Student is identified and authenticated.

**Success Guarantee (Postcondition):** File is downloaded.

**Main Success Scenario (or Basic Flow):**

1. Student visits file download page.

2. System lists available files.

3. Student clicks the file that she/he is wanted to download.

4. System starts the download process.

**Extensions (or Alternative Flows):**

\*a. At any time, System fails: To support recovery, ensure all transaction sensitive state and events can be recovered from any step of the scenario.

1. Student restarts System and requests recovery of prior state.

2. System reconstructs prior state.

2a. System detects anomalies preventing recovery:

1. System signals error to the Student.

2. System records the error.

3. System enters a clean state.

4a. System detects failure to communicate with server:

1. System signals error and rejects the request.

# Use Case UC4: File Upload

**Primary Actor**: Student

**Stakeholders and Interests:**

* Student: Wants simple user interface, fast response, no system errors.

**Preconditions:** Student is identified and authenticated.

**Success Guarantee (Postcondition):** File is uploaded.

**Main Success Scenario (or Basic Flow):**

1. Student visits file upload page.

2. System opens file browser dialog.

3. Student chooses the file that she/he is wanted to upload.

4. System starts the upload process.

**Extensions (or Alternative Flows):**

\*a. At any time, System fails: To support recovery, ensure all transaction sensitive state and events can be recovered from any step of the scenario.

1. Student restarts System and requests recovery of prior state.

2. System reconstructs prior state.

2a. System detects anomalies preventing recovery:

1. System signals error to the Student, records the error, and enters a clean state.

3a. Invalid file:

1. System shows the error and returns the file upload page.

4a. System detects failure to communicate with server:

1. System signals error and rejects the request.

# Use Case UC5: Course Analysis Report

**Primary Actor**: Instructor

**Stakeholders and Interests:**

* Instructor: Wants easily understandable questions, fast response, no system errors.

**Preconditions**: Instructor is identified and authenticated.

**Success Guarantee (Postcondition):** Report is generated.

**Main Success Scenario (or Basic Flow):**

1. Instructor enters his/her profile page.

2. System shows Instructor’s profile page.

3. Instructor clicks My Courses button.

4. System lists Instructor’s courses.

5. Instructor selects the course that she/or is wanted to inspect.

6. System asks for report type.

7. Instructor selects report type.

8. System shows the analysis report.

**Extensions (or Alternative Flows):**

\*a. At any time, System fails: To support recovery, ensure all transaction sensitive state and events can be recovered from any step of the scenario.

1. Instructor restarts System and requests recovery of prior state.

2. System reconstructs prior state.

2a. System detects anomalies preventing recovery:

1. System signals error to the Instructor, records the error, and enters a clean state.

\*b. System detects failure to communicate with server:

1. System signals error and rejects the request.

# Use Case UC14: Adding Lecture

**Primary Actor:** Instructor

**Stakeholders and Interests:**

* Instructor: Wants to add new lecture quickly.

**Preconditions**: Instructor is identified and authenticated.

**Success Guarantee (Postcondition):** Lecture is added.

**Main Success Scenario (or Basic Flow):**

1. Instructor clicks on Add Lecture button on his/her profile page.

2. System returns the information page about the new lecture.

3. Instructor fills out fields about the lecture then clicks on Submit button.

4. System adds the new lecture and saves changes.

**Extensions (or Alternative Flows):**

\*a. At any time, System fails: To support recovery, ensure all transaction sensitive state and events can be recovered from any step of the scenario.

1. Instructor restarts System and requests recovery of prior state.

2. System reconstructs prior state.

2a. System detects anomalies preventing recovery:

1. System signals error to the Instructor, records the error, and enters a clean state.

# Use Case UC15: Deleting Lecture

**Primary Actor:** Instructor

**Stakeholders and Interests:**

* Instructor: Wants to delete lecture quickly.

**Preconditions**: Instructor is identified and authenticated.

**Success Guarantee (Postcondition):** Lecture is deleted.

**Main Success Scenario (or Basic Flow):**

1. Instructor clicks on My Lectures button on his/her profile page.

2. System returns his/her lectures.

3. Instructor choses the lecture that he/she wants to delete.

4. System returns lecture’s information.

5. Instructor clicks on the Delete Lecture button.

6. System deletes the lecture and sends the message to all student who enrolls this lecture.

**Extensions (or Alternative Flows):**

\*a. At any time, System fails: To support recovery, ensure all transaction sensitive state and events can be recovered from any step of the scenario.

1. Instructor restarts System and requests recovery of prior state.

2. System reconstructs prior state.

2a. System detects anomalies preventing recovery:

1. System signals error to the Instructor, records the error, and enters a clean state.

# Use Case UC17: Lecture Listing

**Primary Actor:** Student, Instructors

**Stakeholders and Interests:**

* Student: Wants look up his/her lecture that he/she had chosen.
* Instructor: Wants look up his/her lecture

**Preconditions:** Student is identified and authenticated.

**Success Guarantee (Postcondition):** Lectures that students chose are listing well organized. Student has enrolled lectures that she/he wants. Student deletes his/her enrollment of lectures that he/she had chosen.

**Main Success Scenario (or Basic Flow):**

1. Student clicks my Lectures button.

2. System presents Lectures that was chosen by student.

3. Student views his/her lectures.

**Extensions (or Alternative Flows):**

\*a. At any time, System fails: To support recovery, ensure all transaction sensitive state and events can be recovered from any step of the scenario.

1. Student restarts System and requests recovery of prior state.

2. System reconstructs prior state.

2a. System detects anomalies preventing recovery:

1. System signals error to the Instructor.

2. System records the error.

3. System enters a clean state.

# Use Case UC18: Enrolling to a Lecture

**Primary Actor:** Student

**Stakeholders and Interests:**

* Student: Wants to enroll new lecture easily.

**Preconditions**: Student is identified and authenticated. There must be at least one lecture that student can select.

**Success Guarantee (Postcondition):** Student has enrolled lectures that she/he wants. Student deletes his/her enrollment of lectures that he/she had chosen.

**Main Success Scenario (or Basic Flow):**

1. Student clicks the Lecture Selection button.

2. System presents lectures that student can select.

3. Student selects a lecture.

4. System wants an enrollment key for the lecture from the student.

5. Student enters the keyword for the enrollment and clicks “enroll me” button.

**Extensions (or Alternative Flows):**

\*a. At any time, System fails: To support recovery, ensure all transaction sensitive state and events can be recovered from any step of the scenario.

1. Student restarts System and requests recovery of prior state.

2. System reconstructs prior state.

2a. System detects anomalies preventing recovery:

1. System signals error to the Instructor, records the error, and enters a clean state.

2a. If there is no lecture to select, system gives a message to the student.

5a. If student enters a wrong enrollment key, system gives an error message.

1. System returns to step 4.

# Use Case UC19: Unenrolling a Lecture

**Primary Actor:** Student

**Stakeholders and Interests:**

* Student: Wants delete his/her enrollment of a lecture.

**Preconditions**: Student is identified and authenticated.

**Success Guarantee (Postcondition):** Student deletes his/her enrollment of lectures that he/she had chosen.

**Main Success Scenario (or Basic Flow):**

1. Student clicks My Lectures button.

2. System presents lectures that was chosen by student.

3. Student selects lecture that she/he wants to delete.

4. System removes student from student list of this lecture and takes out that lecture from students lecture list.

**Extensions (or Alternative Flows):**

\*a. At any time, System fails: To support recovery, ensure all transaction sensitive state and events can be recovered from any step of the scenario.

1. Student restarts System and requests recovery of prior state.

2. System reconstructs prior state.

2a. System detects anomalies preventing recovery:

1. System signals error to the Instructor, records the error, and enters a clean state.

3a. If there is no lecture student’s list system gives error message to the student.

# Use Case UC6: New Forum Post

**Primary Actor**: Student

**Stakeholders and Interests:**

* Student: Wants to create a new post easily.

**Preconditions:** Student is identified and authenticated.

**Success Guarantee (Postcondition**): Post is saved.

**Main Success Scenario (or Basic Flow):**

1. Student clicks on New Post button to post a new topic to the topic page.

2. System returns topic editor page.

3. Student fills out the required fields.

4. System checks the fields.

5. Student clicks on the Post button.

6. System records transaction and.

**Extensions (or Alternative Flows):**

\*a. At any time, System fails: To support recovery, ensure all transaction sensitive state and events can be recovered from any step of the scenario.

1. Student restarts System and requests recovery of prior state.

2. System reconstructs prior state.

2a. System detects anomalies preventing recovery:

1. System signals error to the Student, records the error, and enters a clean state.

# Use Case UC7: Reply Forum Post

**Primary Actor:** Student

**Stakeholders and Interests:**

* Student: Wants to reply a post easily.

**Preconditions:** Student is identified and authenticated.

**Success Guarantee (Postcondition):** Post is saved.

**Main Success Scenario (or Basic Flow):**

1. Student clicks on a topic to view it.

2. System returns a page which contains view of topic.

3. Student clicks on Reply button to post an entry to the topic.

4. System returns entry editor page.

5. Student fills out fields.

6. System checks out the fields.

7. Student clicks on the Post button.

8. System records transaction.

**Extensions (or Alternative Flows):**

\*a. At any time, System fails: To support recovery, ensure all transaction sensitive state and events can be recovered from any step of the scenario.

1. Student restarts System and requests recovery of prior state.

2. System reconstructs prior state.

2a. System detects anomalies preventing recovery:

1. System signals error to the Student

2. System records the error

3. System enters a clean state.

# Use Case UC8: Edit Forum Post

**Primary Actor:** Student

**Stakeholders and Interests:**

* Student: Wants to edit his/her post easily.

**Preconditions**: Student is identified and authenticated.

**Success Guarantee (Postconditions):** Post is saved.

**Main Success Scenario (or Basic Flow):**

1. Student clicks on a topic to view it.
2. System returns a page which contains view of topic.
3. Student clicks on Edit button to edit topic if he/she posted before.
4. System returns page of the topic editor.
5. Student update fields.
6. System checks out the fields.
7. Student clicks on the Save Changes button.
8. System saves changes.

**Extensions (or Alternative Flows):**

\*a. At any time, System fails:

To support recovery, ensure all transaction sensitive state and events can be recovered from any step of the scenario.

* 1. Student restarts System and requests recovery of prior state.
  2. System reconstructs prior state.

2a. System detects anomalies preventing recovery:

* + 1. System signals error to the Student, records the error, and enters a clean state.

# Use Case UC9: Complain User

**Primary Actor:** Student

**Stakeholders and Interests:**

* Student: Wants to complain a user.

**Preconditions**: Student is identified and authenticated.

**Success Guarantee (Postconditions):** Administrator is informed about the complain.

**Main Success Scenario (or Basic Flow):**

1. Student clicks on a topic to view it.
2. System returns a page which contains view of topic.
3. Student can complain another user that wrote a message before.
4. Student clicks on complain button.
5. System wants the reason of this complaints.
6. Student write the reason for this complaints.
7. System send this message to the administrator.

**Extensions (or Alternative Flows):**

\*a. At any time, System fails:

To support recovery, ensure all transaction sensitive state and events can be recovered from any step of the scenario.

* 1. Student restarts System and requests recovery of prior state.
  2. System reconstructs prior state.

2a. System detects anomalies preventing recovery:

* + 1. System signals error to the Student, records the error, and enters a clean state.

# Use Case UC10: Hanging-out in Forum

**Primary Actor:** Student

**Stakeholders and Interests:**

* Student: Wants to use forum, read topics and navigate between pages easily.

**Preconditions**: Student is identified and authenticated.

**Success Guarantee (Postconditions):** Forum is viewed.

**Main Success Scenario (or Basic Flow):**

1. Student clicks on Home Page button.
2. System returns the content of Home Page.
3. Student view the content.
4. Student clicks out on log-out button
5. System save changes
6. Student exists.

**Extensions (or Alternative Flows):**

\*a. At any time, System fails:

To support recovery, ensure all transaction sensitive state and events can be recovered from any step of the scenario.

* 1. Student restarts System and requests recovery of prior state.
  2. System reconstructs prior state.

2a. System detects anomalies preventing recovery:

* + 1. System signals error to the Student, records the error, and enters a clean state.

# Use Case UC11: Create Announcement

**Primary Actor:** Instructor

**Stakeholders and Interests:**

* Instructor: Wants to add announcements quickly and easily.
* Student: Wants to be informed immediately when new … announced.

**Preconditions**: Instructor is identified and authenticated.

**Success Guarantee (Postconditions):** Announcement is created and saved.

**Main Success Scenario (or Basic Flow):**

1. Instructor clicks on My Profile button.
2. System returns the profile page.
3. Instructor clicks on Announcements button.
4. System returns the page that contains announcements.
5. Instructor clicks on New Announcement button.
6. System returns an editor to create announcement.
7. Instructor fills out the required fields and choose the class.
8. System checks fields.
9. Instructor clicks on Finish button.
10. System save changes, send massages to students who take that lesson.
11. Instructor closes the page and logs out.

**Extensions (or Alternative Flows):**

\*a. At any time, System fails:

To support recovery, ensure all transaction sensitive state and events can be recovered from any step of the scenario.

* 1. Instructor restarts System and requests recovery of prior state.
  2. System reconstructs prior state.

2a. System detects anomalies preventing recovery:

* + 1. System signals error to the Instructor, records the error, and enters a clean state.

7a. Instructor clicks on Cancel button.

1. System ignores the announcement.
2. Instructor logs out.

8a. System detects missing field(s) or incorrect information(s).

1. System pops up an alert to instructor.
2. Instructor makes corrections about those fields.
3. System checks fields.

# Use Case UC12: Delete Announcement

**Primary Actor:** Instructor

**Stakeholders and Interests:**

* Instructor: Wants to delete announcements fastly and easily.

**Preconditions**: Instructor is identified and authenticated.

**Success Guarantee (Postconditions):** Announcement is deleted.

**Main Success Scenario (or Basic Flow):**

1. Instructor clicks on My Profile button.
2. System returns the profile page.
3. Instructor clicks on Announcements button.
4. System returns the page that contains announcements.
5. Instructor clicks on an announcement that was created by instructor before.
6. System returns the content of announcement.
7. Instructor clicks on Delete button.
8. System deletes announcement.
9. Instructor logs out.
10. System save changes.

**Extensions (or Alternative Flows):**

\*a. At any time, System fails:

To support recovery, ensure all transaction sensitive state and events can be recovered from any step of the scenario.

* 1. Instructor restarts System and requests recovery of prior state.
  2. System reconstructs prior state.

2a. System detects anomalies preventing recovery:

* + 1. System signals error to the Instructor, records the error, and enters a clean state.

# Use Case UC13: Update Announcement

**Primary Actor:** Instructor

**Stakeholders and Interests:**

* Instructor: Wants to update announcements fastly and easily.

**Preconditions**: Instructor is identified and authenticated.

**Success Guarantee (Postconditions):** Announcement is updated.

**Main Success Scenario (or Basic Flow):**

1. Instructor clicks on My Profile button.
2. System returns the profile page.
3. Instructor clicks on Announcements button.
4. System returns the page that contains announcements.
5. Instructor clicks on an announcement that was created by instructor before.
6. System returns the content of announcement.
7. Instructor edits fields.
8. System checks fields.
9. Instructor clicks on save changes button, closes the page and logs out.
10. System save changes.

**Extensions (or Alternative Flows):**

\*a. At any time, System fails:

To support recovery, ensure all transaction sensitive state and events can be recovered from any step of the scenario.

* 1. Instructor restarts System and requests recovery of prior state.
  2. System reconstructs prior state.

2a. System detects anomalies preventing recovery:

* + 1. System signals error to the Instructor, records the error, and enters a clean state.

# Use Case UC16: Sending Message

**Primary Actors:** Instructor, Student

**Stakeholders and Interests:**

* Instructor: Wants send message quickly.
* Student: Wants send message quickly.

**Preconditions**: Instructor/Student is identified and authenticated.

**Success Guarantee (Postconditions):** Message sent.

**Main Success Scenario (or Basic Flow):**

1. Instructor/Student clicks on My Messages button.
2. System returns his/her message’s page.
3. Instructor/Student search for user that he/she wants to send message.
4. System shows result.
5. Instructor/Student chooses user from results.
6. System returns new message page.
7. Instructor/Student writes the message then clicks on Send button.
8. System sends the message.

**Extensions (or Alternative Flows):**

\*a. At any time, System fails:

To support recovery, ensure all transaction sensitive state and events can be recovered from any step of the scenario.

* 1. Instructor/Student restarts System and requests recovery of prior state.
  2. System reconstructs prior state.

2a. System detects anomalies preventing recovery:

* + 1. System signals error to the Instructor/Student, records the error, and enters a clean state.

2a. System shows unread incoming messages to user.

1. Instructor/Student clicks on the message that he/she wants to read.
2. System shows the message.
3. Instructor/Student reads the message and clicks on Close button.
4. System returns messages page.

# Use Case UC20: Solving an Exam/Quiz

**Primary Actor:** Student

**Stakeholders and Interests:**

* Student: Wants to solve all questions in exam without server connection problems.

**Preconditions**: Student is identified and authenticated. Exam/Quiz is not exceed deadline.

**Success Guarantee (Postconditions):** Exam/Quiz is solved by student and answers are saved.

**Main Success Scenario (or Basic Flow):**

1. Student click calendar cell that has exam/quiz event.
2. System presents exam/quiz stage of LMS and starts exam timer.
3. Student gives answers to questions non-sequential or sequential.

3rd step looping until exam timer finishes.

1. System finished exam timer and blocks student to changing on the answers.
2. System routing the student to main page.

**Extensions (or Alternative Flows):**

\*a. At any time, System fails:

To support recovery, ensure all transaction sensitive state and events can be recovered from any step of the scenario.

* 1. Student restarts System and requests recovery of prior state.
  2. System reconstructs prior state.

2a. System detects anomalies preventing recovery:

* + 1. System signals error to the Instructor, records the error, and enters a clean state.

# Use Case UC21: Displaying an Announcement

**Primary Actor:** Student

**Stakeholders and Interests:**

* Student: Wants to display announcement quickly.

**Preconditions**: Student is identified and authenticated.

**Success Guarantee (Postconditions):** Announcement is displayed.

**Main Success Scenario (or Basic Flow):**

1. Student clicks the Announcements button.
2. System display all announcements.
3. Student clicks an announcement that he/she wants to display.
4. System displays to selected announcement.

**Extensions (or Alternative Flows):**

\*a. At any time, System fails:

To support recovery, ensure all transaction sensitive state and events can be recovered from any step of the scenario.

* 1. Student restarts System and requests recovery of prior state.
  2. System reconstructs prior state.

2a. System detects anomalies preventing recovery:

* + 1. System signals error to the Instructor, records the error, and enters a clean state.

# Use Case UC22: Displaying Grades

**Primary Actor:** Student

**Stakeholders and Interests:**

* Student: Wants to display own grades quickly.

**Preconditions**: Student is identified and authenticated.

**Success Guarantee (Postconditions):** Grades are displayed.

**Main Success Scenario (or Basic Flow):**

1. Student clicks My Grades button.
2. System lists his/her grades for all lectures that he/she is enrolled.

**Extensions (or Alternative Flows):**

\*a. At any time, System fails:

To support recovery, ensure all transaction sensitive state and events can be recovered from any step of the scenario.

* 1. Student restarts System and requests recovery of prior state.
  2. System reconstructs prior state.

2a. System detects anomalies preventing recovery:

* + 1. System signals error to the Instructor, records the error, and enters a clean state.

# Use Case UC23: Displaying Calendar

**Primary Actor:** Student

**Stakeholders and Interests:**

* Student: Wants to display calendar easily.

**Preconditions**: Student is identified and authenticated.

**Success Guarantee (Postconditions):** Calendar and events are displayed.

**Main Success Scenario (or Basic Flow):**

1. Student clicks My Calendar button.
2. System returns his/her calendar page.
3. Student selects specific day on calendar.
4. System display events for that day.

**Extensions (or Alternative Flows):**

\*a. At any time, System fails:

To support recovery, ensure all transaction sensitive state and events can be recovered from any step of the scenario.

* 1. Student restarts System and requests recovery of prior state.
  2. System reconstructs prior state.

2a. System detects anomalies preventing recovery:

* + 1. System signals error to the Instructor, records the error, and enters a clean state.

# Use Case UC24: Adding Event to Calendar

**Primary Actor:** Instructor

**Stakeholders and Interests:**

* Instructor: Wants to add an event to calendar easily.

**Preconditions**: Instructor is identified and authenticated.

**Success Guarantee (Postconditions):** Event is added.

**Main Success Scenario (or Basic Flow):**

1. Instructor clicks My Calendar button on his/her profile page.
2. System displays his/her calendar.
3. Instructor selects a day from calendar.
4. System displays events for that day.
5. Instructor clicks Add Event button.
6. System displays Add Event page.
7. Instructor writes event’s detail and clicks Save button.
8. System saves event.

**Extensions (or Alternative Flows):**

\*a. At any time, System fails:

To support recovery, ensure all transaction sensitive state and events can be recovered from any step of the scenario.

* 1. Instructor restarts System and requests recovery of prior state.
  2. System reconstructs prior state.

2a. System detects anomalies preventing recovery:

* + 1. System signals error to the Instructor, records the error, and enters a clean state.

7a. If event’s detail is empty, system shows an error.

# Use Case UC25: Deleting Event from Calendar

**Primary Actor:** Instructor

**Stakeholders and Interests:**

* Instructor: Wants to delete an event from calendar easily.

**Preconditions**: Instructor is identified and authenticated. There is must be an event in instructor’s calendar that was added by this instructor.

**Success Guarantee (Postconditions):** Event is deleted.

**Main Success Scenario (or Basic Flow):**

1. Instructor clicks My Calendar button on his/her profile page.
2. System displays his/her calendar.
3. Instructor selects a day from calendar.
4. System displays events for that day.
5. Instructor clicks the event that he/she wants to delete.
6. System displays the event.
7. Instructor clicks Delete Event button.
8. System deletes event.

**Extensions (or Alternative Flows):**

\*a. At any time, System fails:

To support recovery, ensure all transaction sensitive state and events can be recovered from any step of the scenario.

* 1. Instructor restarts System and requests recovery of prior state.
  2. System reconstructs prior state.

2a. System detects anomalies preventing recovery:

* + 1. System signals error to the Instructor, records the error, and enters a clean state.

# Use Case UC26: Entering Grades

**Primary Actor:** Instructor

**Stakeholders and Interests:**

* Instructor: Wants to enter grades easily.

**Preconditions**: Instructor is identified and authenticated.

**Success Guarantee (Postconditions):** Grades are saved.

**Main Success Scenario (or Basic Flow):**

1. Instructor clicks Grades button on his/her profile page.
2. System returns grades page.
3. Instructor selects a specific course from the courses list.
4. System shows student list and grades labels.
5. Instructor enters grade for students one by one and clicks Save button.
6. System saves grades.

**Extensions (or Alternative Flows):**

\*a. At any time, System fails:

To support recovery, ensure all transaction sensitive state and events can be recovered from any step of the scenario.

* 1. Instructor restarts System and requests recovery of prior state.
  2. System reconstructs prior state.

2a. System detects anomalies preventing recovery:

* + 1. System signals error to the Instructor, records the error, and enters a clean state.

# Use Case UC27: Adding an Exam/Quiz

**Primary Actor:** Instructor

**Stakeholders and Interests:**

* Instructor: Wants to add new exam easily.

**Preconditions**: Instructor is identified and authenticated.

**Success Guarantee (Postconditions):** New exam is created.

**Main Success Scenario (or Basic Flow):**

1. Instructor clicks Exams button on his/her profile page.
2. System returns exams page.
3. Instructor clicks New Exam Button.
4. System returns new exam form.
5. Instructor enters exam name and selects lecture.
6. System shows exam questions page.
7. Instructor writes exam questions and clicks Save button.
8. System saves exam and inform students about this exam.

**Extensions (or Alternative Flows):**

\*a. At any time, System fails:

To support recovery, ensure all transaction sensitive state and events can be recovered from any step of the scenario.

* 1. Instructor restarts System and requests recovery of prior state.
  2. System reconstructs prior state.

2a. System detects anomalies preventing recovery:

* + 1. System signals error to the Instructor, records the error, and enters a clean state.

# Use Case UC28: Exam Assessment

**Primary Actor:** Instructor

**Stakeholders and Interests:**

* Instructor: Wants to evaluate exam easily.

**Preconditions**: Instructor is identified and authenticated.

**Success Guarantee (Postcondition):** Exam is evaluated.

**Main Success Scenario (or Basic Flow):**

1. Instructor clicks Exams button on his/her profile page.

2. System returns exams page.

3. Instructor selects an exam that he/she wants to evaluate.

4. System shows student list that are solved this exam.

5. Instructor selects a student.

6. System shows this student’s answer this grade label.

7. Instructor enters grade for each answer and Save button.

8. System saves grades.

**Extensions (or Alternative Flows):**

\*a. At any time, System fails: To support recovery, ensure all transaction sensitive state and events can be recovered from any step of the scenario.

1. Instructor restarts System and requests recovery of prior state.

2. System reconstructs prior state.

2a. System detects anomalies preventing recovery:

1. System signals error to the Instructor.

2. System records the error.

3. System enters a clean state.

