# Course Information Management System

## Group 2

David Chang
Linda Chowdhury
Oscar Fitinghoff
Patrik Parberg
Tomas Hansson

# Contents

# 5.5 Detailed Design

# Database

### Detailed database table definitions

**Table: User**

**Columns:**

| Key | Column name | Data type | Not null | Unique | Index | Extra |
|---|---|---|---|---|---|---|
| Primary | userID | integer | X | | | auto_increment |
| | username | varchar(50) | X | X | | |
| | password | varchar(64) | X | | | |
| | firstname | varchar(50) | X | | | |
| | lastname | varchar(50) | X | | | |

**Table: Course**

**Columns:**

| Key | Column name | Data type | Not null | Unique | Index | Extra |
|---|---|---|---|---|---|---|
| Primary | courseID | integer | | | | auto_increment |
| | courseCode | varchar(20) | X | X | | |
| Foreign | courseLeader | integer | | | | |
| | courseName | varchar(50) | X | | | |
| | startYear | integer | | | | |
| | startPeriod | smallint | | | | |
| | endYear | integer | | | | |
| | endPeriod | smallint | | | | |
| | credits | float | | | | |
| | description | text | X | | | |

**Table: Activity**

**Columns:**

| Key | Column name | Data type | Not null | Unique | Index | Extra |
|---|---|---|---|---|---|---|
| Primary | activityID | integer | X | | | auto_increment |
| Foreign | courseID | integer | X | | | |
| | title | varchar(100) | X | | | |
| | description | text | X | | | |
| | startTime | DateTime | X | | | |
| | endTime | DateTime | X | | | |

**Table: Deadline**

**Columns:**

| Key | Column name | Data type | Not null | Unique | Index | Extra |
|---|---|---|---|---|---|---|
| Primary | deadlineID | integer | X | | | auto_increment |
| Foreign | courseID | integer | X | | | |
| | headline | varchar(100) | X | | | |
| | content | text | X | | | |
| | time | DateTime | X | | | |

**Table: News**

**Columns:**

| Key | Column name | Data type | Not null | Unique | Index | Extra |
|---|---|---|---|---|---|---|
| Primary | newsID | integer | X | | | auto_increment |
| Foreign | courseID | integer | X | | | |
| Foreign | author | integer | X | | | |
| | headline | varchar(100) | X | | | |
| | content | text | X | | | |
| | time | DateTime | X | | | |

**Table: File**

**Columns:**

| Key | Column name | Data type | Not null | Unique | Index | Extra |
|---|---|---|---|---|---|---|
| Primary | fileID | integer | X | | | auto_increment |
| Foreign | courseID | integer | X | | | |
| | title | varchar(100) | X | | | |
| | description | text | X | | | |
| | filename | varchar(50) | X | | | |

**Table: Information Page**

**Columns:**

| Key | Column name | Data type | Not null | Unique | Index | Extra |
|---|---|---|---|---|---|---|
| Primary | pageID | integer | X | | | auto_increment |
| Foreign | courseID | integer | X | | | |
| | title | varchar(100) | X | | | |
| | content | text | X | | | |

**Table: Assignment**

**Columns:**

| Key | Column name | Data type | Not null | Unique | Index | Extra |
|---|---|---|---|---|---|---|
| Primary | assignmentID | integer | X | | | auto_increment |
| Foreign | courseID | integer | X | | | |
| | title | varchar(100) | X | | | |
| | description | text | X | | | |

**Table: InCourse**

**Columns:**

| Key | Column name | Data type | Not null | Unique | Index | Extra |
|---|---|---|---|---|---|---|
| Primary | userID | integer | X | | | |
| Primary | courseID | integer | X | | | |
| | status | enum | X | | | |

**Table: Result**

**Columns:**

| Key | Column name | Data type | Not null | Unique | Index | Extra |
|---|---|---|---|---|---|---|
| Primary | userID | integer | X | | | |
| Primary | assignmentID | integer | X | | | |
| | grade | enum | X | | | |

**Table: Privilege**

**Columns:**

| Key | Column name | Data type | Not null | Unique | Index | Extra |
|---|---|---|---|---|---|---|
| Primary | userID | integer | X | | | |
| | privilege | enum | X | | | |

# Classes

## Class Activity

### *Field belongToCourse*

The ID of the course that the activity belongs to.

**Type:** Integer
**Access level:** Private

### *Field description*

The description of the activity.

**Type:** String
**Access level:** Private

### *Field endDateTime*

The date and time that the activity ends.

**Type:** Date
**Access level:** Private

### *Field startDateTime*

The date and time that the activity starts.

**Type:** Date
**Access level:** Private

### *Field title*

The title of the activity.

**Type:** String
**Access level:** Private

### *Method GetBelongToCourse*

Retrieves the value of the belongToCourse field.

**Requirements:** 7.2
**Parameters:** N/A
**Return:** Integer (ID of the course that the activity belongs to)
**Data access:** N/A
**Pre-conditions:** N/A
**Validity Check:** N/A
**Post-conditions:** N/A
**Caller:** - ActivityController::Update
**Calls:** N/A

### *Method GetDescription*

Retrieves the value of the description field.

| | |
|---|---|
| **Requirements:** | 7.2 |
| **Parameters:** | N/A |
| **Return:** | String (description of the activity) |
| **Data access:** | N/A |
| **Pre-conditions:** | N/A |
| **Validity Check:** | N/A |
| **Post-conditions:** | N/A |
| **Caller:** | - ActivityController::Update |
| **Calls:** | N/A |

### *Method GetEndDateTime*

Retrieves the value of the endDateTime field.

| | |
|---|---|
| **Requirements:** | 7.2 |
| **Parameters:** | N/A |
| **Return:** | Date (Date and time for the end of the activity) |
| **Data access:** | N/A |
| **Pre-conditions:** | N/A |
| **Validity Check:** | N/A |
| **Post-conditions:** | N/A |
| **Caller:** | - ActivityController::Update<br>**-** Activity::SetStartDateTime |
| **Calls:** | N/A |

### *Method GetStartDateTime*

Retrieves the value of the startDateTime field.

| | |
|---|---|
| **Requirements:** | 7.2 |
| **Parameters:** | N/A |
| **Return:** | Date (Date and time for the start of the activity) |
| **Data access:** | N/A |
| **Pre-conditions:** | N/A |
| **Validity Check:** | N/A |
| **Post-conditions:** | N/A |
| **Caller:** | - ActivityController::Update<br>**-** Activity::SetEndDateTime |
| **Calls:** | N/A |

### *Method GetTitle*

Retrieves the value of the title field.

| | |
|---|---|
| **Requirements:** | 7.2 |
| **Parameters:** | N/A |
| **Return:** | String (title of the activity) |
| **Data access:** | N/A |
| **Pre-conditions:** | N/A |

**Validity Check:** N/A
**Post-conditions:** N/A
**Caller:** - ActivityController::Update
**Calls:** N/A

### *Method SetBelongToCourse*

Stores an integer in the belongToCourse field.

**Requirements:** 7.2
**Parameters:** Integer (ID of the course that the activity belongs to)
**Return:** Boolean (true if the ID was successfully stored and validated)
**Data access:** N/A
**Pre-conditions:** N/A
**Validity Check:** - The course ID is an integer and identifies a course that exists
**Post-conditions:** N/A
**Caller:** - ActivityController::Add
- ActivityController::Update
**Calls:** N/A

### *Method SetDescription*

Stores a String in the description field.

**Requirements:** 7.2
**Parameters:** String (description of the activity)
**Return:** Boolean (true if the description was successfully stored and validated)
**Data access:** N/A
**Pre-conditions:** N/A
**Validity Check:** - The description is a string no longer than 100 characters
**Post-conditions:** N/A
**Caller:** - ActivityController::Add
- ActivityController::Update
**Calls:** N/A

### *Method SetEndDateTime*

Stores a Date object in the endDateTime field.

**Requirements:** 7.2
**Parameters:** Timestamp (Date and time for the end of the activity)
**Return:** Boolean (true if the date and time of an activity was successfully stored and validated)
**Data access:** N/A
**Pre-conditions:** N/A
**Validity Check:** - The endDateTime is a Date object, representing date and time where the year has to be at least 1900, and if startDateTime is defined then the endDateTime has to occur after startDateTime.
**Post-conditions:** N/A
**Caller:** - ActivityController::Add
- ActivityController::Update
- Activity::SetStartDateTime
**Calls:** - Activity::SetStartDateTime

### *Method SetStartDateTime*

Stores a Date object in the startDateTime field.

| | |
|---|---|
| **Requirements:** | 7.2 |
| **Parameters:** | Date (Date and time for the start of the activity) |
| **Return:** | Boolean (true if the date and time of an activity was successfully stored and validated) |
| **Data access:** | N/A |
| **Pre-conditions:** | N/A |
| **Validity Check:** | - The startDateTime is a Date object, representing date and time where the year has to be at least 1900, and if endDateTime is defined and startDateTime is defined to occur after endDateTime, the endDateTime is changed into the same as startDateTime. |
| **Post-conditions:** | N/A |
| **Caller:** | - ActivityController::Add<br>- ActivityController::Update |
| **Calls:** | - Activity::GetEndDateTime<br>- Activity::SetEndDateTime |

### *Method SetTitle*

Stores a String in the title field.

| | |
|---|---|
| **Requirements:** | 7.2 |
| **Parameters:** | String (title of the activity) |
| **Return:** | Boolean (true if the title was successfully stored and validated) |
| **Data access:** | N/A |
| **Pre-conditions:** | N/A |
| **Validity Check:** | - The title is a string no longer than 50 characters |
| **Post-conditions:** | N/A |
| **Caller:** | - ActivityController::Add<br>- ActivityController::Update |
| **Calls:** | N/A |

## Class ActivityController

### *Method Add*

The method stores an activity in the database.

| | |
|---|---|
| **Requirements:** | 7.2 |
| **Parameters:** | **-** Activity (the activity to be stored) |
| **Return:** | Boolean (true if activity was successfully stored in the database) |
| **Data access:** | Inserts a row in the database table *Activity* |
| **Pre-conditions:** | - A connection to the database is established |
| **Validity Check:** | N/A |
| **Post-conditions:** | - One new row is inserted into the *Activity* table in the database |
| **Caller:** | - JSP page for adding a scheduled activity |
| **Calls:** | - Activity get methods |

### *Method Get*

The method gets an activity from the database.

| | |
|---|---|
| **Requirements:** | 7.2 and 7.3 |
| **Parameters:** | **-** Integer (the activity ID to be fetched) |
| **Return:** | Activity (the activity corresponding to the activity ID) |
| **Data access:** | Fetches a row in the database table *Activity* |
| **Pre-conditions:** | - A connection to the database is established |
| **Validity Check:** | N/A |
| **Post-conditions:** | N/A |
| **Caller:** | - JSP page for editing or viewing a scheduled activity |
| **Calls:** | - Activity set methods |

### *Method Remove*

The method removes an activity from the database.

| | |
|---|---|
| **Requirements:** | 7.2 |
| **Parameters:** | **-** Activity (the activity to be removed) |
| **Return:** | Boolean (true if activity was successfully removed from the database) |
| **Data access:** | Deletes a row in the database table *Activity* |
| **Pre-conditions:** | - A connection to the database is established<br>- The activity exists in the database |
| **Validity Check:** | N/A |
| **Post-conditions:** | - The activity's row is removed from the *Activity* table in the database |
| **Caller:** | - JSP page for removing a scheduled activity |
| **Calls:** | N/A |

### *Method Update*

The method updates an activity in the database.

| | |
|---|---|
| **Requirements:** | 7.2 |
| **Parameters:** | **-** Activity (the activity to be updated) |
| **Return:** | Boolean (true if activity was successfully updated in the database) |
| **Data access:** | Update a row in the database table *Activity* |
| **Pre-conditions:** | - A connection to the database is established<br>- The activity exists in the database |
| **Validity Check:** | N/A |
| **Post-conditions:** | - One row is updated in the *Activity* table in the database |
| **Caller:** | - JSP page for updating a scheduled activity |
| **Calls:** | - Activity get methods |

## Class Assignment

### *Field belongToCourse*

The ID of the course that the assignment belongs to.

| | |
|---|---|
| **Type:** | Integer |
| **Access level:** | Private |

### *Field deadline*
The deadline of the assignment.

**Type:** Deadline
**Access level:** Private

### *Field description*
The description of the assignment.

**Type:** String
**Access level:** Private

### *Field title*
The title of the assignment.

**Type:** String
**Access level:** Private

### *Method GetBelongToCourse*
Retrieves the value of the belongToCourse field.

**Requirements:** 10.1
**Parameters:** N/A
**Return:** String (name of the course that the activity belongs to)
**Data access:** N/A
**Pre-conditions:** N/A
**Validity Check:** N/A
**Post-conditions:** N/A
**Caller:** - AssignmentController::Update
**Calls:** N/A

### *Method GetDeadline*
Retrieves the Deadline object in the deadline field.

**Requirements:** 10.1
**Parameters:** N/A
**Return:** Deadline (the deadline of the activity)
**Data access:** N/A
**Pre-conditions:** N/A
**Validity Check:** N/A
**Post-conditions:** N/A
**Caller:** - AssignmentController::Update
**Calls:** N/A

### *Method GetDescription*
Retrieves the value of the description field.

**Requirements:** 10.1
**Parameters:** N/A

| | |
|---|---|
| **Return:** | String (description of the assignment) |
| **Data access:** | N/A |
| **Pre-conditions:** | N/A |
| **Validity Check:** | N/A |
| **Post-conditions:** | N/A |
| **Caller:** | - AssignmentController::Update |
| **Calls:** | N/A |

### *Method GetTitle*

Retrieves the value of the title field.

| | |
|---|---|
| **Requirements:** | 10.1 |
| **Parameters:** | N/A |
| **Return:** | String (title of the assignment) |
| **Data access:** | N/A |
| **Pre-conditions:** | N/A |
| **Validity Check:** | N/A |
| **Post-conditions:** | N/A |
| **Caller:** | - AssignmentController::Update |
| **Calls:** | N/A |

### *Method SetBelongToCourse*

Stores an integer in the belongToCourse field.

| | |
|---|---|
| **Requirements:** | 7.2 |
| **Parameters:** | Integer (ID of the course that the assignment belongs to) |
| **Return:** | Boolean (true if the ID was successfully stored and validated) |
| **Data access:** | N/A |
| **Pre-conditions:** | N/A |
| **Validity Check:** | - The course ID is an integer and identifies a course that exists |
| **Post-conditions:** | N/A |
| **Caller:** | - AssignmentController::Add<br>- AssignmentController::Update |
| **Calls:** | N/A |

### *Method SetDeadline*

Stores a Deadline object in the deadline field.

| | |
|---|---|
| **Requirements:** | 10.1 |
| **Parameters:** | Deadline (the deadline of the activity) |
| **Return:** | Boolean (true if the deadline was successfully stored and validated) |
| **Data access:** | N/A |
| **Pre-conditions:** | N/A |
| **Validity Check:** | The deadline is a Deadline object. |
| **Post-conditions:** | N/A |
| **Caller:** | - AssignmentController::Update |
| **Calls:** | N/A |

### *Method SetDescription*

Stores a String in the description field.

| | |
|---|---|
| **Requirements:** | 7.2 |
| **Parameters:** | String (description of the assignment) |
| **Return:** | Boolean (true if the description was successfully stored and validated) |
| **Data access:** | N/A |
| **Pre-conditions:** | N/A |
| **Validity Check:** | - The description is a string no longer than 100 characters |
| **Post-conditions:** | N/A |
| **Caller:** | - AssignmentController::Add<br>- AssignmentController::Update |
| **Calls:** | N/A |

### *Method SetTitle*

Stores a String in the title field.

| | |
|---|---|
| **Requirements:** | 7.2 |
| **Parameters:** | String (title of the assignment) |
| **Return:** | Boolean (true if the title was successfully stored and validated) |
| **Data access:** | N/A |
| **Pre-conditions:** | N/A |
| **Validity Check:** | - The title is a string no longer than 50 characters |
| **Post-conditions:** | N/A |
| **Caller:** | - AssignmentController::Add<br>- AssignmentController::Update |
| **Calls:** | N/A |

## Class AssignmentController

### *Method Add*

The method stores an assignment in the database.

| | |
|---|---|
| **Requirements:** | 10.1 |
| **Parameters:** | **-** Assignment (the assignment to be stored) |
| **Return:** | Boolean (true if assignment was successfully stored in the database) |
| **Data access:** | Inserts a row in the database table *Assignment* |
| **Pre-conditions:** | - A connection to the database is established |
| **Validity Check:** | N/A |
| **Post-conditions:** | - One new row is inserted into the *Assignment* table in the database |
| **Caller:** | - JSP page for adding an assignment |
| **Calls:** | - Assignment get methods |

### *Method Get*

The method gets an assignment from the database.

| | |
|---|---|
| **Requirements:** | 10.1 and 10.2 |
| **Parameters:** | **-** Integer (the assignment ID to be fetched) |
| **Return:** | Assignment (the assignment corresponding to the assignment ID) |
| **Data access:** | Fetches a row in the database table *Assignment* |
| **Pre-conditions:** | - A connection to the database is established |

| | |
|---|---|
| **Validity Check:** | N/A |
| **Post-conditions:** | N/A |
| **Caller:** | - JSP page for editing an assignment |
| **Calls:** | - Assignment set methods |

### *Method GetAssignmentByCourse*

The method generates a list of all assignments for a specific course.

| | |
|---|---|
| **Requirements:** | 10.1 |
| **Parameters:** | **-** Course (the course for which assignments shall be retrieved) |
| **Return:** | ArrayList (containing all assignments for the course) |
| **Data access:** | Retrieve rows from the database table *Assignment* |
| **Pre-conditions:** | - A connection to the database is established<br>- The course exists in the database |
| **Validity Check:** | N/A |
| **Post-conditions:** | N/A |
| **Caller:** | - JSP page for displaying assignments for a course<br>- JSP page for displaying a list of existing assignments |
| **Calls:** | - Assignment set methods |

### *Method Remove*

The method removes an assignment from the database. It also removes all results associated to the assignment.

| | |
|---|---|
| **Requirements:** | 10.1 |
| **Parameters:** | **-** Assignment (the assignment to be removed) |
| **Return:** | Boolean (true if assignment was successfully removed from the database) |
| **Data access:** | Deletes a row in the database table *Assignment* and all rows in the database table *Result* that are associated to the deleted row in *Assignment* |
| **Pre-conditions:** | - A connection to the database is established<br>- The assignment exists in the database |
| **Validity Check:** | N/A |
| **Post-conditions:** | - The assignment's row is removed from the *Assignment* table in the database and all rows in *Result* that were associated to that row has been removed |
| **Caller:** | - JSP page for removing an assignment |
| **Calls:** | N/A |

### *Method Update*

The method updates an assignment in the database.

| | |
|---|---|
| **Requirements:** | 10.1 |
| **Parameters:** | **-** Assignment (the assignment to be updated) |
| **Return:** | Boolean (true if assignment was successfully updated in the database) |
| **Data access:** | Update a row in the database table *Assignment* |
| **Pre-conditions:** | - A connection to the database is established<br>- The assignment exists in the database |
| **Validity Check:** | N/A |

**Post-conditions:** - One row is updated in the *Assignment* table in the database
**Caller:** - JSP page for updating an assignment
**Calls:** - Assignment get methods

## Class Cache

### *Field cache*

Stores the cached objects and the keys needed to retrieve them.

**Type:** HashMap<Object, BaseObject>
**Access level:** Private

### *Field timeout*

The current cache timeout value.

**Type:** Integer
**Access level:** Private

### *Method Add*

Adds an object to the object cache for later retrieval.

**Requirements:** 14.1-14.3
**Parameters:** **-** Object (The key used to later retrieve the object)
- BaseObject (the object to cache)
**Return:** void
**Data access:** N/A
**Pre-conditions:** N/A
**Validity Check:** N/A
**Post-conditions:** The object has been added to the object cache.
**Caller:** - Any page or object that loads BaseObject type objects.
**Calls:** N/A

### *Method Get*

Retrieves a previously cached object.

**Requirements:** 14.1-14.3
**Parameters:** **-** Object (The key to which the object is mapped)
**Return:** Object (Object mapped to the key, or null if no object found)
**Data access:** N/A
**Pre-conditions:** N/A
**Validity Check:** N/A
**Post-conditions:** An object reference is returned, or null if there was no object found.
**Caller:** - Any page or object that loads a BaseObject type object.
**Calls:** N/A

### *Method GetCacheTimeout*

Gets the amount of time before cached objects become invalid.

**Requirements:** 14.1-14.3

**Parameters:** N/A
**Return:** Integer (representing the amount of minutes before cache items are considered invalid)
**Data access:** N/A
**Pre-conditions:** N/A
**Validity Check:** N/A
**Post-conditions:** N/A
**Caller:** N/A
**Calls:** N/A

### *Method Remove*

Removes an object from the object cache.

**Requirements:** 14.1-14.3
**Parameters:** **-** Key
**Return:** Boolean (true if the item was found and removed from the cache)
**Data access:** N/A
**Pre-conditions:** N/A
**Validity Check:** N/A
**Post-conditions:** The object mapped to the specified key can no longer be retrieved.
**Caller:** - Any controller object.
**Calls:** N/A

### *Method SetCacheTimeout*

Sets the amount of minutes before cached items are considered invalid.

**Requirements:** 14.1-14.3
**Parameters:** - Integer (representing the amount of minutes before cache items are considered invalid)
**Return:** void
**Data access:** N/A
**Pre-conditions:** N/A
**Validity Check:** N/A
**Post-conditions:** The cache timeout time has been updated.
**Caller:** N/A
**Calls:** N/A

## Class Course

### *Field assignments*

A collection of Assignment objects representing assignments of this course.

**Type:** ArrayList<Assignment>
**Access level:** Private

### *Field courseCode*

The course code of the course.

**Type:** String

**Access level:** Private

### *Field deadlines*

A collection of Deadline objects representing deadlines of this course.

**Type:** ArrayList<DeadLine>
**Access level:** Private

### *Field files*

A collection of File objects representing files of this course.

**Type:** ArrayList<File>
**Access level:** Private

### *Field informationPages*

A collection of InformationPage objects representing information pages of this course.

**Type:** ArrayList<InformationPage>
**Access level:** Private

### *Field news*

A collection of News objects representing news of this course.

**Type:** ArrayList<News>
**Access level:** Private

### *Field results*

A collection of Result objects representing results of this course.

**Type:** ArrayList<Result>
**Access level:** Private

### *Field users*

A collection of User objects representing the students registered for this course.

**Type:** ArrayList<User>
**Access level:** Private

### *Method AddAssignment*

Stores an Assignment object in the assignments ArrayList.

**Requirements:** 10.1
**Parameters:** Assignment (the assignment that's added to the course)
**Return:** Boolean (true if the Assignment object is successfully stored and validated)
**Data access:** N/A
**Pre-conditions:** N/A
**Validity Check:** The input has to be an Assignment object.
**Post-conditions:** N/A

**Caller:** N/A
**Calls:** N/A

### *Method AddDeadline*

Stores a deadline object in the deadlines ArrayList.

**Requirements:** 8.1
**Parameters:** Deadline (the deadline that's added to the course)
**Return:** Boolean (true if the Deadline object is successfully stored and validated)
**Data access:** N/A
**Pre-conditions:** N/A
**Validity Check:** The input has to be a Deadline object.
**Post-conditions:** N/A
**Caller:** N/A
**Calls:** N/A

### *Method AddFile*

Stores a File object in the files ArrayList.

**Requirements:** 9.1
**Parameters:** File (the file that's added to the course)
**Return:** Boolean (true if the Files object is successfully stored and validated)
**Data access:** N/A
**Pre-conditions:** N/A
**Validity Check:** The input has to be a File object.
**Post-conditions:** N/A
**Caller:** N/A
**Calls:** N/A

### *Method AddInformationPage*

Stores an InformationPage object in the InformationPage ArrayList.

**Requirements:** 6.1
**Parameters:** InformationPage (the InformationPage that's added to the course)
**Return:** Boolean (true if the InformationPage object is successfully stored and validated)
**Data access:** N/A
**Pre-conditions:** N/A
**Validity Check:** The input has to be an InformationPage object.
**Post-conditions:** N/A
**Caller:** N/A
**Calls:** N/A

### *Method AddNews*

Stores a News object in the news ArrayList.

**Requirements:** 5.1
**Parameters:** News (the news that's added to the course)
**Return:** Boolean (true if the News object is successfully stored and validated)

**Data access:** N/A
**Pre-conditions:** N/A
**Validity Check:** The input has to be a News object.
**Post-conditions:** N/A
**Caller:** N/A
**Calls:** N/A

### *Method AddResult*

Stores a Result object in the results ArrayList.

**Requirements:** 11.1
**Parameters:** Result (the result that's added to the course)
**Return:** Boolean (true if the Result object is successfully stored and validated)
**Data access:** N/A
**Pre-conditions:** N/A
**Validity Check:** The input has to be a Result object.
**Post-conditions:** N/A
**Caller:** N/A
**Calls:** N/A

### *Method AddUser*

Stores a User object in the users ArrayList. Register a student for the course.

**Requirements:** 12.2
**Parameters:** User (the student that's registered for the course)
**Return:** Boolean (true if the User object is successfully stored and validated)
**Data access:** N/A
**Pre-conditions:** N/A
**Validity Check:** The input is a User object. The user isn't already registered for the course.
**Post-conditions:** N/A
**Caller:** N/A
**Calls:** N/A

### *Method GetAllAssignments*

Retrieves all assignments for this course.

**Requirements:** 10.1-10.2
**Parameters:** N/A
**Return:** ArrayList<Deadline> (all deadlines for this course)
**Data access:** N/A
**Pre-conditions:** N/A
**Validity Check:** N/A
**Post-conditions:** N/A
**Caller:** N/A
**Calls:** N/A

### *Method GetAllDeadlines*

Retrieves all deadlines for this course.

**Requirements:** 8.1-8.3
**Parameters:** N/A
**Return:** ArrayList<Deadline> (all deadlines for this course)
**Data access:** N/A
**Pre-conditions:** N/A
**Validity Check:** N/A
**Post-conditions:** N/A
**Caller:** N/A
**Calls:** N/A

### *Method GetAllFiles*

Retrieves all files for this course.

**Requirements:** 9.1-9.2
**Parameters:** N/A
**Return:** ArrayList<File> (all files for this course)
**Data access:** N/A
**Pre-conditions:** N/A
**Validity Check:** N/A
**Post-conditions:** N/A
**Caller:** N/A
**Calls:** N/A

### *Method GetAllInformationPages*

Retrieves all information pages for this course.

**Requirements:** 6.1-6.2
**Parameters:** N/A
**Return:** ArrayList<InformationPages> (all information pages for this course)
**Data access:** N/A
**Pre-conditions:** N/A
**Validity Check:** N/A
**Post-conditions:** N/A
**Caller:** N/A
**Calls:** N/A

### *Method GetAllNews*

Retrieves all news for this course.

**Requirements:** 5.1-5.2
**Parameters:** N/A
**Return:** ArrayList<News> (all the news for this course)
**Data access:** N/A
**Pre-conditions:** N/A
**Validity Check:** N/A
**Post-conditions:** N/A
**Caller:** N/A
**Calls:** N/A

### *Method GetAllResults*

Retrieves all results for this course.

| | |
|---|---|
| **Requirements:** | 11.1-11.2 |
| **Parameters:** | N/A |
| **Return:** | ArrayList<Results> (all results for this course) |
| **Data access:** | N/A |
| **Pre-conditions:** | N/A |
| **Validity Check:** | N/A |
| **Post-conditions:** | N/A |
| **Caller:** | N/A |
| **Calls:** | N/A |

### *Method GetAllUsers*

Retrieves all students that are registered for this course.

| | |
|---|---|
| **Requirements:** | 12.1 |
| **Parameters:** | N/A |
| **Return:** | ArrayList<Users> (all students registered for this course) |
| **Data access:** | N/A |
| **Pre-conditions:** | N/A |
| **Validity Check:** | N/A |
| **Post-conditions:** | N/A |
| **Caller:** | N/A |
| **Calls:** | N/A |

### *Method GetCourseCode*

Retrieves the value of the courseCode field.

| | |
|---|---|
| **Requirements:** | N/A |
| **Parameters:** | N/A |
| **Return:** | String (the course code for this course) |
| **Data access:** | N/A |
| **Pre-conditions:** | N/A |
| **Validity Check:** | N/A |
| **Post-conditions:** | N/A |
| **Caller:** | N/A |
| **Calls:** | N/A |

### *Method SetCourseCode*

Updates the value of the course code field.

| | |
|---|---|
| **Requirements:** | 13.3 |
| **Parameters:** | String (the course code for this course) |
| **Return:** | N/A |
| **Data access:** | N/A |
| **Pre-conditions:** | N/A |
| **Validity Check:** | N/A |
| **Post-conditions:** | N/A |

**Caller:** N/A
**Calls:** N/A

## Class CourseController

### *Method Add*

The method stores a course in the database.

**Requirements:** 13.3
**Parameters:** **-** Course (the course to be stored)
**Return:** Boolean (true if course was successfully stored in the database)
**Data access:** Inserts a row in the database table *Course*
**Pre-conditions:** - A connection to the database is established
**Validity Check:** N/A
**Post-conditions:** - One new row is inserted into the *Course* table in the database
**Caller:** - JSP page for adding a course
**Calls:** - Course get methods

### *Method GetDescription*

The method gets a course description from the database.

**Requirements:** 4.1 and 4.2
**Parameters:** **-** Integer (the course ID of the course to be fetched)
**Return:** Course (the course corresponding to the course ID)
**Data access:** Fetches a row in the database table *Course*
**Pre-conditions:** - A connection to the database is established
**Validity Check:** N/A
**Post-conditions:** N/A
**Caller:** - JSP page for editing or viewing a course description
**Calls:** - Course set methods

### *Method GetDescriptionByCourseCode*

The method gets a course description from the database using a course code. The course code has to match exactly.

**Requirements:** 4.1
**Parameters:** **-** String (the course code of the course to be fetched)
**Return:** Course (the course corresponding to the course code)
**Data access:** Fetches a row in the database table *Course*
**Pre-conditions:** - A connection to the database is established
**Validity Check:** N/A
**Post-conditions:** N/A
**Caller:** - JSP page for editing user privileges for a course
**Calls:** - Course set methods

### *Method Update*

The method updates a course stored in the database.

**Requirements:** 13.3

| | |
|---|---|
| **Parameters:** | **-** Course (the course to be stored) |
| **Return:** | Boolean (true if course was successfully stored in the database) |
| **Data access:** | Update a row in the database table *Course* |
| **Pre-conditions:** | - A connection to the database is established |
| **Validity Check:** | N/A |
| **Post-conditions:** | - One row is updated in the *Course* table in the database |
| **Caller:** | - JSP page for editing a course |
| **Calls:** | - Course get methods |

### *Method UpdateDescription*

The method updates a course with a course description in the database.

| | |
|---|---|
| **Requirements:** | 4.1 and 13.3 |
| **Parameters:** | **-** Course (the course to be updated) |
| **Return:** | Boolean (true if course description was successfully updated in the database) |
| **Data access:** | Update a row in the database table *Course* |
| **Pre-conditions:** | - A connection to the database is established<br>- The course exists in the database |
| **Validity Check:** | N/A |
| **Post-conditions:** | - One row is updated in the *Course* table in the database |
| **Caller:** | - JSP page for updating a course description |
| **Calls:** | - Course get methods |

### *Method Apply*

The method applies a user for a course in the database. This is done by inserting a row in table *InCourse* with status field APPLYING.

| | |
|---|---|
| **Requirements:** | 12.3 |
| **Parameters:** | **-** Course (the course to which the user is applying for)<br>- User (the user to apply) |
| **Return:** | Boolean (true if the user was successfully applied for the course in the database) |
| **Data access:** | Update a row in the database table *InCourse* |
| **Pre-conditions:** | - A connection to the database is established<br>- The course exists in the database<br>- The user exists in the database |
| **Validity Check:** | N/A |
| **Post-conditions:** | - One row is inserted in the *InCourse* table in the database |
| **Caller:** | - JSP page for applying for a course |
| **Calls:** | N/A |

### *Method Register*

The method registers a user for a course in the database. This is done by updating the status field in table *InCourse* to REGISTERED, from the previous state APPLYING.

| | |
|---|---|
| **Requirements:** | 12.2 |
| **Parameters:** | **-** Course (the course to which the user shall be registered for)<br>- User (the user to register) |

| | |
|---|---|
| **Return:** | Boolean (true if the user was successfully registered for the course in the database) |
| **Data access:** | Update a row in the database table *InCourse* |
| **Pre-conditions:** | - A connection to the database is established<br>- The course exists in the database<br>- The user exists in the database<br>- The user has applied for the course |
| **Validity Check:** | N/A |
| **Post-conditions:** | - One row is updated in the *InCourse* table in the database |
| **Caller:** | - JSP page for updating a course description |
| **Calls:** | N/A |

### *Method Unregister*

The method unregisters a user from a course in the database. This is done deleting the row in table *InCourse* corresponding to the given course and user.

| | |
|---|---|
| **Requirements:** | 12.4 |
| **Parameters:** | **-** Course (the course to which the user shall be unregistered from)<br>- User (the user to unregister) |
| **Return:** | Boolean (true if the user was successfully unregistered from the course in the database) |
| **Data access:** | Delete a row in the database table *InCourse* |
| **Pre-conditions:** | - A connection to the database is established<br>- The course exists in the database<br>- The user exists in the database<br>- The user has been registered for the course |
| **Validity Check:** | N/A |
| **Post-conditions:** | - One row is deleted in the *InCourse* table in the database |
| **Caller:** | - JSP page for updating a course description |
| **Calls:** | N/A |

## Class Deadline

### *Field belongToCourse*

The ID of the course that the deadline belongs to.

| | |
|---|---|
| **Type:** | Integer |
| **Access level:** | Private |

### *Field deadlineDateTime*

The date and time that of the deadline.

| | |
|---|---|
| **Type:** | Date |
| **Access level:** | Private |

### *Field description*

The description of the deadline.

| | |
|---|---|
| **Type:** | String |

**Access level:** Private

### *Field title*
The title of the deadline.

**Type:** String
**Access level:** Private

### *Method GetBelongToCourse*
Retrieves the value of the belongToCourse field.

**Requirements:** 8.1-8.3
**Parameters:** N/A
**Return:** Integer (ID of the course that the deadline belongs to)
**Data access:** N/A
**Pre-conditions:** N/A
**Validity Check:** N/A
**Post-conditions:** N/A
**Caller:** - DeadlineController::Update
**-** DeadlineController::getCourseDeadlines
- DeadlineController::getUserDeadlines
**Calls:** N/A

### *Method GetDeadlineDateTime*
Retrieves the value of the deadlineDateTime field.

**Requirements:** 8.1-8.3
**Parameters:** N/A
**Return:** Date (Date and time for when the deadline expire)
**Data access:** N/A
**Pre-conditions:** N/A
**Validity Check:** N/A
**Post-conditions:** N/A
**Caller:** - DeadlineController::Update
**-** DeadlineController::getCourseDeadlines
- DeadlineController::getUserDeadlines
**Calls:** N/A

### *Method GetDescription*
Retrieves the value of the description field.

**Requirements:** 8.1-8.3
**Parameters:** N/A
**Return:** String (description of the deadline)
**Data access:** N/A
**Pre-conditions:** N/A
**Validity Check:** N/A
**Post-conditions:** N/A
**Caller:** - DeadlineController::Update
**-** DeadlineController::getCourseDeadlines

| | |
|---|---|
| | - DeadlineController::getUserDeadlines |
| **Calls:** | N/A |

### *Method GetTitle*

Retrieves the value of the title field.

| | |
|---|---|
| **Requirements:** | 8.1-8.3 |
| **Parameters:** | N/A |
| **Return:** | String (title of the deadline) |
| **Data access:** | N/A |
| **Pre-conditions:** | N/A |
| **Validity Check:** | N/A |
| **Post-conditions:** | N/A |
| **Caller:** | - DeadlineController::Update<br>- DeadlineController::getCourseDeadlines<br>- DeadlineController::getUserDeadlines |
| **Calls:** | N/A |

### *Method SetBelongToCourse*

Stores an integer in the belongToCourse field.

| | |
|---|---|
| **Requirements:** | 8.1-8.3 |
| **Parameters:** | Integer (ID of the course that the deadline belongs to) |
| **Return:** | Boolean (true if the ID was successfully stored and validated) |
| **Data access:** | N/A |
| **Pre-conditions:** | N/A |
| **Validity Check:** | - The course ID is an integer and identifies a course that exists |
| **Post-conditions:** | N/A |
| **Caller:** | - DeadlineController::Add<br>- DeadlineController::Update |
| **Calls:** | N/A |

### *Method SetDeadlineDateTime*

Stores a Date object in the deadlineDateTime field.

| | |
|---|---|
| **Requirements:** | 8.1-8.3 |
| **Parameters:** | Date (Date and time for when the deadline expire) |
| **Return:** | Boolean (true if the date and time of a deadline was successfully stored and validated) |
| **Data access:** | N/A |
| **Pre-conditions:** | N/A |
| **Validity Check:** | The input has to be a Date object and the year has to be at least 1900. |
| **Post-conditions:** | N/A |
| **Caller:** | - DeadlineController::Update<br>- DeadlineController::Add |
| **Calls:** | N/A |

### *Method SetDescription*

Stores a String in the description field.

| | |
|---|---|
| **Requirements:** | 8.1-8.3 |
| **Parameters:** | String (description of the deadline) |
| **Return:** | Boolean (true if the description was successfully stored and validated) |
| **Data access:** | N/A |
| **Pre-conditions:** | N/A |
| **Validity Check:** | - The description is a string no longer than 100 characters |
| **Post-conditions:** | N/A |
| **Caller:** | - DeadlineController::Add<br>- DeadlineController::Update |
| **Calls:** | N/A |

### *Method SetTitle*

Stores a String in the title field.

| | |
|---|---|
| **Requirements:** | 8.1-8.3 |
| **Parameters:** | String (title of the deadline) |
| **Return:** | Boolean (true if the title was successfully stored and validated) |
| **Data access:** | N/A |
| **Pre-conditions:** | N/A |
| **Validity Check:** | - The title is a string no longer than 50 characters |
| **Post-conditions:** | N/A |
| **Caller:** | - DeadlineController::Add<br>- DeadlineController::Update |
| **Calls:** | N/A |

## Class DeadlineController

### *Method Add*

The method stores a deadline in the database.

| | |
|---|---|
| **Requirements:** | 8.1 |
| **Parameters:** | **-** Deadline (the deadline to be stored) |
| **Return:** | Boolean (true if deadline was successfully stored in the database) |
| **Data access:** | Inserts a row in the database table *Deadline* |
| **Pre-conditions:** | - A connection to the database is established |
| **Validity Check:** | N/A |
| **Post-conditions:** | - One new row is inserted into the *Deadline* table in the database |
| **Caller:** | - JSP page for adding a deadline |
| **Calls:** | - Deadline get methods |

### *Method Get*

The method gets a deadline from the database.

| | |
|---|---|
| **Requirements:** | 8.1 and 8.2 |
| **Parameters:** | **-** Integer (the deadline ID of the deadline to be fetched) |
| **Return:** | Deadline (the deadline corresponding to the deadline ID) |
| **Data access:** | Fetches a row in the database table *Deadline* |
| **Pre-conditions:** | - A connection to the database is established |
| **Validity Check:** | N/A |

| | |
|---|---|
| **Post-conditions:** | N/A |
| **Caller:** | - JSP page for editing or viewing a deadline |
| **Calls:** | - Deadline set methods |

### *Method GetDeadlinesByCourse*

The method generates a list of all deadlines for a specific course.

| | |
|---|---|
| **Requirements:** | 2.1 and 2.2 |
| **Parameters:** | **-** Course (the course for which deadlines shall be retrieved) |
| **Return:** | ArrayList (containing all deadlines for the course) |
| **Data access:** | Retrieve rows from the database table *Deadline* |
| **Pre-conditions:** | - A connection to the database is established<br>- The deadline exists in the database |
| **Validity Check:** | N/A |
| **Post-conditions:** | N/A |
| **Caller:** | JSP page for displaying deadlines for a course |
| **Calls:** | - Deadline set methods |

### *Method GetDeadlinesByUser*

The method generates a list of all deadlines for courses that a specific user is registered for.

| | |
|---|---|
| **Requirements:** | 8.3 |
| **Parameters:** | **-** User (the user for which deadlines shall be retrieved) |
| **Return:** | ArrayList (containing all deadlines for the user's courses) |
| **Data access:** | Retrieves row from the database table *Deadline* |
| **Pre-conditions:** | - A connection to the database is established<br>- The user exists in the database |
| **Validity Check:** | N/A |
| **Post-conditions:** | N/A |
| **Caller:** | JSP page for displaying deadlines for a user |
| **Calls:** | - Deadline set methods |

### *Method Remove*

The method removes a deadline from the database.

| | |
|---|---|
| **Requirements:** | 8.1 |
| **Parameters:** | **-** Deadline (the deadline to be removed) |
| **Return:** | Boolean (true if deadline was successfully removed from the database) |
| **Data access:** | Deletes a row in the database table *Deadline* |
| **Pre-conditions:** | - A connection to the database is established<br>- The deadline exists in the database |
| **Validity Check:** | N/A |
| **Post-conditions:** | - The deadline's row is removed from the *Deadline* table in the database |
| **Caller:** | - JSP page for removing a deadline |
| **Calls:** | N/A |

### *Method Update*

The method updates a deadline in the database.

**Requirements:** 8.1
**Parameters:** **-** Deadline (the deadline to be updated)
**Return:** Boolean (true if deadline was successfully updated in the database)
**Data access:** Update a row in the database table *Deadline*
**Pre-conditions:** - A connection to the database is established
- The deadline exists in the database
**Validity Check:** N/A
**Post-conditions:** - One row is updated in the *Deadline* table in the database
**Caller:** - JSP page for updating a deadline
**Calls:** - Deadline get methods

## Class File

### *Field belongToCourse*

The ID of the course that the file belongs to.

**Type:** Integer
**Access level:** Private

### *Field description*

The description of the file.

**Type:** String
**Access level:** Private

### *Field filename*

The name of the file.

**Type:** String
**Access level:** Private

### *Field title*

The title of the file.

**Type:** String
**Access level:** Private

### *Method GetBelongToCourse*

Retrieves the value of the belongToCourse field.

**Requirements:** 9.1-9.2
**Parameters:** N/A
**Return:** Integer (ID of the course that the file belongs to)
**Data access:** N/A
**Pre-conditions:** N/A
**Validity Check:** N/A
**Post-conditions:** N/A
**Caller:** - FileController::Update
**Calls:** N/A

### *Method GetDescription*

Retrieves the value of the description field.

| | |
|---|---|
| **Requirements:** | 9.1-9.2 |
| **Parameters:** | N/A |
| **Return:** | String (description of the file) |
| **Data access:** | N/A |
| **Pre-conditions:** | N/A |
| **Validity Check:** | N/A |
| **Post-conditions:** | N/A |
| **Caller:** | - FileController::Update |
| **Calls:** | N/A |

### *Method GetFilename*

Retrieves the value of the filename field.

| | |
|---|---|
| **Requirements:** | 9.1-9.2 |
| **Parameters:** | N/A |
| **Return:** | String (name of the file) |
| **Data access:** | N/A |
| **Pre-conditions:** | N/A |
| **Validity Check:** | N/A |
| **Post-conditions:** | N/A |
| **Caller:** | - FileController::Update |
| **Calls:** | N/A |

### *Method GetTitle*

Retrieves the value of the title field.

| | |
|---|---|
| **Requirements:** | 9.1-9.2 |
| **Parameters:** | N/A |
| **Return:** | String (title of the file) |
| **Data access:** | N/A |
| **Pre-conditions:** | N/A |
| **Validity Check:** | N/A |
| **Post-conditions:** | N/A |
| **Caller:** | - FileController::Update |
| **Calls:** | N/A |

### *Method SetBelongToCourse*

Stores an integer in the belongToCourse field.

| | |
|---|---|
| **Requirements:** | 9.1-9.2 |
| **Parameters:** | Integer (ID of the course that the file belongs to) |
| **Return:** | Boolean (true if the ID was successfully stored and validated) |
| **Data access:** | N/A |
| **Pre-conditions:** | N/A |
| **Validity Check:** | - The course ID is an integer and identifies a course that exists |
| **Post-conditions:** | N/A |

| | |
|---|---|
| **Caller:** | - FileController::Add<br>- FileController::Update |
| **Calls:** | N/A |

### *Method SetDescription*

Stores a String in the description field.

| | |
|---|---|
| **Requirements:** | 9.1-9.2 |
| **Parameters:** | String (description of the file) |
| **Return:** | Boolean (true if the description was successfully stored and validated) |
| **Data access:** | N/A |
| **Pre-conditions:** | N/A |
| **Validity Check:** | - The description is a string no longer than 100 characters |
| **Post-conditions:** | N/A |
| **Caller:** | - FileController::Add<br>- FileController::Update |
| **Calls:** | N/A |

### *Method SetFilename*

Stores a String in the filename field.

| | |
|---|---|
| **Requirements:** | 9.1-9.2 |
| **Parameters:** | String (name of the file) |
| **Return:** | Boolean (true if the name was successfully stored and validated) |
| **Data access:** | N/A |
| **Pre-conditions:** | N/A |
| **Validity Check:** | - The name is a string no longer than 50 characters |
| **Post-conditions:** | N/A |
| **Caller:** | - FileController::Add<br>- FileController::Update |
| **Calls:** | N/A |

### *Method SetTitle*

Stores a String in the title field.

| | |
|---|---|
| **Requirements:** | 9.1-9.2 |
| **Parameters:** | String (title of the file) |
| **Return:** | Boolean (true if the title was successfully stored and validated) |
| **Data access:** | N/A |
| **Pre-conditions:** | N/A |
| **Validity Check:** | - The title is a string no longer than 50 characters |
| **Post-conditions:** | N/A |
| **Caller:** | - FileController::Add<br>- FileController::Update |
| **Calls:** | N/A |

## Class FileController

### *Method Add*

The method stores a file on the file system and its metadata in the database.

| | |
|---|---|
| **Requirements:** | 9.1 |
| **Parameters:** | **-** File (the file object to be stored) |
| **Return:** | Boolean (true if file was successfully stored in the database) |
| **Data access:** | Inserts a row in the database table *File* |
| **Pre-conditions:** | - A connection to the database is established |
| **Validity Check:** | N/A |
| **Post-conditions:** | - One new row is inserted into the *File* table in the database and a new file is saved on the file system |
| **Caller:** | - JSP page for adding a file |
| **Calls:** | - File get methods |

### *Method Get*

The method gets a file from the database.

| | |
|---|---|
| **Requirements:** | 9.1 |
| **Parameters:** | **-** Integer (the file ID of the file to be fetched) |
| **Return:** | File (the file corresponding to the file ID) |
| **Data access:** | Fetches a row in the database table *File* |
| **Pre-conditions:** | - A connection to the database is established |
| **Validity Check:** | N/A |
| **Post-conditions:** | N/A |
| **Caller:** | - JSP page for editing or viewing a file |
| **Calls:** | - File set methods |

### *Method GetFileByCourse*

The method generates a list of all files for a specific course.

| | |
|---|---|
| **Requirements:** | 9.1 |
| **Parameters:** | **-** Course (the course for which files shall be retrieved) |
| **Return:** | ArrayList (containing all files for the course) |
| **Data access:** | Retrieve rows from the database table *File* |
| **Pre-conditions:** | - A connection to the database is established<br>- The course exists in the database |
| **Validity Check:** | N/A |
| **Post-conditions:** | N/A |
| **Caller:** | - JSP page for displaying files for a course<br>- JSP page for displaying a list of existing files |
| **Calls:** | - File set methods |

### *Method Remove*

The method removes a file from the file system and its metadata from the database.

| | |
|---|---|
| **Requirements:** | 9.1 |
| **Parameters:** | **-** File (the file object to be removed) |

| | |
|---|---|
| **Return:** | Boolean (true if file was successfully removed from the database) |
| **Data access:** | Deletes a row in the database table *File* and a file from the file system |
| **Pre-conditions:** | - A connection to the database is established<br>- The file exists on the file system and its metadata exists in the database |
| **Validity Check:** | N/A |
| **Post-conditions:** | - The file's row is removed from the *File* table in the database and the file is removed from the file system |
| **Caller:** | - JSP page for removing a file |
| **Calls:** | N/A |

### *Method Update*

The method updates a file on the file system and its metadata in the database.

| | |
|---|---|
| **Requirements:** | 9.1 |
| **Parameters:** | **-** File (the file object to be updated) |
| **Return:** | Boolean (true if file was successfully updated in the database) |
| **Data access:** | Update a row in the database table *File* |
| **Pre-conditions:** | - A connection to the database is established<br>- The file exists in the database |
| **Validity Check:** | N/A |
| **Post-conditions:** | - One row is updated in the *File* table in the database |
| **Caller:** | - JSP page for updating a file |
| **Calls:** | - File get methods |

## Class InformationPage

### *Field belongToCourse*

The ID of the course that the information page belongs to.

| | |
|---|---|
| **Type:** | Integer |
| **Access level:** | Private |

### *Field content*

The content of the information page.

| | |
|---|---|
| **Type:** | String |
| **Access level:** | Private |

### *Field title*

The title of the information page.

| | |
|---|---|
| **Type:** | String |
| **Access level:** | Private |

### *Method GetBelongToCourse*

Retrieves the value of the belongToCourse field.

| | |
|---|---|
| **Requirements:** | 6.1-6.2 |

**Parameters:** N/A
**Return:** Integer (ID of the course that the information page belongs to)
**Data access:** N/A
**Pre-conditions:** N/A
**Validity Check:** N/A
**Post-conditions:** N/A
**Caller:** - InformationPageController::Update
**Calls:** N/A

### *Method GetContent*

Retrieves the value of the content field.

**Requirements:** 6.1-6.2
**Parameters:** N/A
**Return:** String (content of the information page)
**Data access:** N/A
**Pre-conditions:** N/A
**Validity Check:** N/A
**Post-conditions:** N/A
**Caller:** - InformationPageController::Update
**Calls:** N/A

### *Method GetTitle*

Retrieves the value of the title field.

**Requirements:** 6.1-6.2
**Parameters:** N/A
**Return:** String (title of the information page)
**Data access:** N/A
**Pre-conditions:** N/A
**Validity Check:** N/A
**Post-conditions:** N/A
**Caller:** - InformationPageController::Update
**Calls:** N/A

### *Method SetBelongToCourse*

Stores an integer in the belongToCourse field.

**Requirements:** 6.1-6.2
**Parameters:** Integer (ID of the course that the information page belongs to)
**Return:** Boolean (true if the ID was successfully stored and validated)
**Data access:** N/A
**Pre-conditions:** N/A
**Validity Check:** - The course ID is an integer and identifies a course that exists
**Post-conditions:** N/A
**Caller:** - InformationPageController::Add
- InformationPageController::Update
**Calls:** N/A

### *Method SetContent*

Stores a String in the content field.

| | |
|---|---|
| **Requirements:** | 6.1-6.2 |
| **Parameters:** | String (content of the information page) |
| **Return:** | Boolean (true if the content was successfully stored and validated) |
| **Data access:** | N/A |
| **Pre-conditions:** | N/A |
| **Validity Check:** | - The content is a string |
| **Post-conditions:** | N/A |
| **Caller:** | - FileController::Add<br>- FileController::Update |
| **Calls:** | N/A |

### *Method SetTitle*

Stores a String in the title field.

| | |
|---|---|
| **Requirements:** | 6.1-6.2 |
| **Parameters:** | String (title of the information page) |
| **Return:** | Boolean (true if the title was successfully stored and validated) |
| **Data access:** | N/A |
| **Pre-conditions:** | N/A |
| **Validity Check:** | - The title is a string no longer than 50 characters |
| **Post-conditions:** | N/A |
| **Caller:** | - InformationPageController::Add<br>- InformationPageController::Update |
| **Calls:** | N/A |

## Class InformationPageController

### *Method Add*

The method stores an information page in the database.

| | |
|---|---|
| **Requirements:** | 6.1 |
| **Parameters:** | **-** InformationPage (the information page to be stored) |
| **Return:** | Boolean (true if information page was successfully stored in the database) |
| **Data access:** | Inserts a row in the database table *InformationPage* |
| **Pre-conditions:** | - A connection to the database is established |
| **Validity Check:** | N/A |
| **Post-conditions:** | - One new row is inserted into the *InformationPage* table in the database |
| **Caller:** | - JSP page for adding an information page |
| **Calls:** | - InformationPage get methods |

### *Method Get*

The method gets an information page from the database.

| | |
|---|---|
| **Requirements:** | 6.1 and 6.2 |

| | |
|---|---|
| **Parameters:** | **-** Integer (the information page ID of the information page to be fetched) |
| **Return:** | InformationPage (the information page corresponding to the file ID) |
| **Data access:** | Fetches a row in the database table *InformationPage* |
| **Pre-conditions:** | - A connection to the database is established |
| **Validity Check:** | N/A |
| **Post-conditions:** | N/A |
| **Caller:** | - JSP page for editing or viewing an information page |
| **Calls:** | - InformationPage set methods |

### *Method GetInformationPageByCourse*

The method generates a list of all information pages for a specific course.

| | |
|---|---|
| **Requirements:** | 6.1 and 6.2 |
| **Parameters:** | **-** Course (the course for which information pages shall be retrieved) |
| **Return:** | ArrayList (containing all information pages for the course) |
| **Data access:** | Retrieve rows from the database table *InformationPage* |
| **Pre-conditions:** | - A connection to the database is established<br>- The course exists in the database |
| **Validity Check:** | N/A |
| **Post-conditions:** | N/A |
| **Caller:** | - JSP page for displaying a course website<br>- JSP page for displaying a list of existing information pages |
| **Calls:** | - InformationPage set methods |

### *Method Remove*

The method removes an information page from the database.

| | |
|---|---|
| **Requirements:** | 6.1 |
| **Parameters:** | **-** InformationPage (the information page to be removed) |
| **Return:** | Boolean (true if information page was successfully removed from the database) |
| **Data access:** | Deletes a row in the database table *InformationPage* |
| **Pre-conditions:** | - A connection to the database is established<br>- The information page exists in the database |
| **Validity Check:** | N/A |
| **Post-conditions:** | - The information page's row is removed from the *InformationPage* table in the database |
| **Caller:** | - JSP page for removing an information page |
| **Calls:** | N/A |

### *Method Update*

The method updates an information page in the database.

| | |
|---|---|
| **Requirements:** | 6.1 |
| **Parameters:** | **-** InformationPage (the information page to be updated) |
| **Return:** | Boolean (true if information page was successfully updated in the database) |
| **Data access:** | Update a row in the database table *InformationPage* |
| **Pre-conditions:** | - A connection to the database is established |

| | |
|---|---|
| | - The information page exists in the database |
| **Validity Check:** | N/A |
| **Post-conditions:** | - One row is updated in the *InformationPage* table in the database |
| **Caller:** | - JSP page for updating an information page |
| **Calls:** | - InformationPage get methods |

## Class News

### *Field belongToCourse*

The ID of the course that the news belongs to.

| | |
|---|---|
| **Type:** | Integer |
| **Access level:** | Private |

### *Field content*

The content of the news.

| | |
|---|---|
| **Type:** | String |
| **Access level:** | Private |

### *Field headline*

The headline of the news.

| | |
|---|---|
| **Type:** | String |
| **Access level:** | Private |

### *Method GetBelongToCourse*

Retrieves the value of the belongToCourse field.

| | |
|---|---|
| **Requirements:** | 5.1-5.2 |
| **Parameters:** | N/A |
| **Return:** | Integer (ID of the course that the news belongs to) |
| **Data access:** | N/A |
| **Pre-conditions:** | N/A |
| **Validity Check:** | N/A |
| **Post-conditions:** | N/A |
| **Caller:** | - NewsController::Update<br>- NewsController::GetCourseNews<br>- NewsController::GetUserNews |
| **Calls:** | N/A |

### *Method GetContent*

Retrieves the value of the content field.

| | |
|---|---|
| **Requirements:** | 5.1-5.2 |
| **Parameters:** | N/A |
| **Return:** | String (content of the news) |
| **Data access:** | N/A |
| **Pre-conditions:** | N/A |

| | |
|---|---|
| **Validity Check:** | N/A |
| **Post-conditions:** | N/A |
| **Caller:** | - NewsController::Update<br>- NewsController::GetCourseNews<br>- NewsController::GetUserNews |
| **Calls:** | N/A |

### *Method GetHeadline*

Retrieves the value of the headline field.

| | |
|---|---|
| **Requirements:** | 5.1-5.2 |
| **Parameters:** | N/A |
| **Return:** | String (headline of the news) |
| **Data access:** | N/A |
| **Pre-conditions:** | N/A |
| **Validity Check:** | N/A |
| **Post-conditions:** | N/A |
| **Caller:** | - NewsController::Update<br>- NewsController::GetCourseNews<br>- NewsController::GetUserNews |
| **Calls:** | N/A |

### *Method SetBelongToCourse*

Stores an integer in the belongToCourse field.

| | |
|---|---|
| **Requirements:** | 5.1-5.2 |
| **Parameters:** | Integer (ID of the course that the news belongs to) |
| **Return:** | Boolean (true if the ID was successfully stored and validated) |
| **Data access:** | N/A |
| **Pre-conditions:** | N/A |
| **Validity Check:** | - The course ID is an integer and identifies a course that exists |
| **Post-conditions:** | N/A |
| **Caller:** | - NewsController::Add<br>- NewsController::Update |
| **Calls:** | N/A |

### *Method SetContent*

Stores a String in the content field.

| | |
|---|---|
| **Requirements:** | 5.1-5.2 |
| **Parameters:** | String (content of the news) |
| **Return:** | Boolean (true if the content was successfully stored and validated) |
| **Data access:** | N/A |
| **Pre-conditions:** | N/A |
| **Validity Check:** | - The description is a string |
| **Post-conditions:** | N/A |
| **Caller:** | - NewsController::Add<br>- NewsController::Update |
| **Calls:** | N/A |

### *Method SetHeadline*

Stores a String in the headline field.

| | |
|---|---|
| **Requirements:** | 5.1-5.2 |
| **Parameters:** | String (headline of the news) |
| **Return:** | Boolean (true if the headline was successfully stored and validated) |
| **Data access:** | N/A |
| **Pre-conditions:** | N/A |
| **Validity Check:** | - The headline is a string no longer than 50 characters |
| **Post-conditions:** | N/A |
| **Caller:** | - NewsController::Add<br>- NewsController::Update |
| **Calls:** | N/A |

## Class NewsController

### *Method Add*

The method stores a news post in the database.

| | |
|---|---|
| **Requirements:** | 5.1 |
| **Parameters:** | **-** News (the news post to be stored) |
| **Return:** | Boolean (true if news post was successfully stored in the database) |
| **Data access:** | Inserts a row in the database table *News* |
| **Pre-conditions:** | - A connection to the database is established |
| **Validity Check:** | N/A |
| **Post-conditions:** | - One new row is inserted into the *News* table in the database |
| **Caller:** | - JSP page for adding a news post |
| **Calls:** | - News get methods |

### *Method Get*

The method gets a news post from the database.

| | |
|---|---|
| **Requirements:** | 5.1 |
| **Parameters:** | **-** Integer (the news ID of the news to be fetched) |
| **Return:** | News (the news post corresponding to the news ID) |
| **Data access:** | Fetches a row in the database table *News* |
| **Pre-conditions:** | - A connection to the database is established |
| **Validity Check:** | N/A |
| **Post-conditions:** | N/A |
| **Caller:** | - JSP page for editing or viewing a news post |
| **Calls:** | - News set methods |

### *Method GetNewsByCourse*

The method generates a list of all news for a specific course.

| | |
|---|---|
| **Requirements:** | 5.2 |
| **Parameters:** | **-** Course (the course for which news shall be retrieved) |
| **Return:** | ArrayList (containing all news for the course) |
| **Data access:** | Retrieve rows from the database table *News* |

| | |
|---|---|
| **Pre-conditions:** | - A connection to the database is established<br>- The course exists in the database |
| **Validity Check:** | N/A |
| **Post-conditions:** | N/A |
| **Caller:** | - JSP page for displaying news for a course |
| **Calls:** | - News set methods |

### *Method GetNewsByUser*

The method generates a list of all news for courses that a specific user is registered for.

| | |
|---|---|
| **Requirements:** | 2.1, 2.2 and 2.3 |
| **Parameters:** | **-** User (the user for which news shall be retrieved) |
| **Return:** | ArrayList (containing all news for the user's courses) |
| **Data access:** | Retrieves row from the database table *News* |
| **Pre-conditions:** | - A connection to the database is established<br>- The user exists in the database |
| **Validity Check:** | N/A |
| **Post-conditions:** | N/A |
| **Caller:** | - JSP page for displaying news for a user |
| **Calls:** | - News get methods |

### *Method Remove*

The method removes a news post from the database.

| | |
|---|---|
| **Requirements:** | 5.1 |
| **Parameters:** | **-** News (the news post to be removed) |
| **Return:** | Boolean (true if news post was successfully removed from the database) |
| **Data access:** | Deletes a row in the database table *News* |
| **Pre-conditions:** | - A connection to the database is established<br>- The news post exists in the database |
| **Validity Check:** | N/A |
| **Post-conditions:** | - The news post's row is removed from the *News* table in the database |
| **Caller:** | - JSP page for removing a news post |
| **Calls:** | N/A |

### *Method Update*

The method updates a news post in the database.

| | |
|---|---|
| **Requirements:** | 5.1 |
| **Parameters:** | **-** News (the news post to be updated) |
| **Return:** | Boolean (true if news post was successfully updated in the database) |
| **Data access:** | Update a row in the database table *News* |
| **Pre-conditions:** | - A connection to the database is established<br>- The news post exists in the database |
| **Validity Check:** | N/A |
| **Post-conditions:** | - One row is updated in the *News* table in the database |
| **Caller:** | - JSP page for updating a news post |
| **Calls:** | - News set methods |

## Class PrivilegeController

### *Method AssignCourseAssistant*

The method assigns a user the course assistant privilege in the database by updating the status field of the *InCourse* table to ASSISTANT.

| | |
|---|---|
| **Requirements:** | 13.1 |
| **Parameters:** | **-** User (the user to assign the privilege to)<br>- Course (the course to assign the privilege to) |
| **Return:** | Boolean (true if privilege was successfully assigned in the database) |
| **Data access:** | Update a row in the database table *InCourse* |
| **Pre-conditions:** | - A connection to the database is established<br>- The user exists in the database<br>- The course exists in the database |
| **Validity Check:** | N/A |
| **Post-conditions:** | - One row is updated or inserted in the *InCourse* table in the database with value of the status field set to ASSISTANT |
| **Caller:** | - JSP page for assigning the course assistant privilege |
| **Calls:** | N/A |

### *Method AssignCourseLeader*

The method assigns a user the course leader privilege in the database by updating the status field of the *InCourse* table to LEADER.

| | |
|---|---|
| **Requirements:** | 13.1 |
| **Parameters:** | **-** User (the user to assign the privilege to)<br>- Course (the course to assign the privilege to) |
| **Return:** | Boolean (true if privilege was successfully assigned in the database) |
| **Data access:** | Update a row in the database table *InCourse* |
| **Pre-conditions:** | - A connection to the database is established<br>- The user exists in the database<br>- The course exists in the database |
| **Validity Check:** | N/A |
| **Post-conditions:** | - One row is updated or inserted in the *InCourse* table in the database with value of the status field set to LEADER |
| **Caller:** | - JSP page for assigning the course leader privilege |
| **Calls:** | N/A |

### *Method AssignSysAdmin*

The method assigns a user the system administrator privilege in the database.

| | |
|---|---|
| **Requirements:** | 13.1 |
| **Parameters:** | **-** User (the user to assign the privilege to) |
| **Return:** | Boolean (true if privilege was successfully assigned in the database) |
| **Data access:** | Update a row in the database table *Privilege* |
| **Pre-conditions:** | - A connection to the database is established<br>- The user exists in the database |
| **Validity Check:** | N/A |
| **Post-conditions:** | - One row is inserted in the *Privilege* table in the database |

| | |
|---|---|
| **Caller:** | - JSP page for assigning the system administrator privilege |
| **Calls:** | N/A |

### *Method RevokeCourseAssistant*

The method revokes the course assistant privilege for a user in the database by removing the row for the user and course in the *InCourse* database table.

| | |
|---|---|
| **Requirements:** | 13.1 |
| **Parameters:** | **-** User (the user to revoke the privilege from)<br>- Course (the course to revoke the privilege from) |
| **Return:** | Boolean (true if privilege was successfully revoked in the database) |
| **Data access:** | Delete a row in the database table *InCourse* |
| **Pre-conditions:** | - A connection to the database is established<br>- The user exists in the database<br>- The course exists in the database |
| **Validity Check:** | N/A |
| **Post-conditions:** | - The user is no longer course assistant for the specified course |
| **Caller:** | - JSP page for revoking the course assistant privilege |
| **Calls:** | N/A |

### *Method RevokeCourseLeader*

The method revokes the course assistant privilege for a user in the database by removing the row for the user and course in the *InCourse* database table.

| | |
|---|---|
| **Requirements:** | 13.1 |
| **Parameters:** | **-** User (the user to revoke the privilege from)<br>- Course (the course to revoke the privilege from) |
| **Return:** | Boolean (true if privilege was successfully revoked in the database) |
| **Data access:** | Delete a row in the database table *InCourse* |
| **Pre-conditions:** | - A connection to the database is established<br>- The user exists in the database<br>- The course exists in the database |
| **Validity Check:** | N/A |
| **Post-conditions:** | - The user is no longer course leader for the specified course |
| **Caller:** | - JSP page for revoking the course leader privilege |
| **Calls:** | N/A |

### *Method RevokeSysAdmin*

The method revokes a user the system administrator privilege in the database.

| | |
|---|---|
| **Requirements:** | 13.1 |
| **Parameters:** | **-** User (the user to revoke the privilege from) |
| **Return:** | Boolean (true if privilege was successfully revoked in the database) |
| **Data access:** | Delete a row in the database table *Privilege* |
| **Pre-conditions:** | - A connection to the database is established<br>- The user exists in the database |
| **Validity Check:** | N/A |
| **Post-conditions:** | - The user is no longer system administrator |
| **Caller:** | - JSP page for revoking the system administrator privilege |
| **Calls:** | N/A |

## Class Result

### *Field belongToAssignment*

The assignment the result is registered for.

**Type:** Assignment
**Access level:** Private

### *Field belongToCourse*

The ID of the course that the result belongs to.

**Type:** Integer
**Access level:** Private

### *Field grade*

The grade of the assignment.

**Type:** String
**Access level:** Private

### *Field user*

The user that the result belongs to.

**Type:** User
**Access level:** Private

### *Method GetBelongToAssignment*

Retrieves the value of the belongToAssignment field.

**Requirements:** 11.1-11.2
**Parameters:** N/A
**Return:** Assignment (assignment that the result belongs to)
**Data access:** N/A
**Pre-conditions:** N/A
**Validity Check:** N/A
**Post-conditions:** N/A
**Caller:** - ResultController::Update
**Calls:** N/A

### *Method GetBelongToCourse*

Retrieves the value of the belongToCourse field.

**Requirements:** 11.1-11.2
**Parameters:** N/A
**Return:** String (name of the course that the result belongs to)
**Data access:** N/A
**Pre-conditions:** N/A
**Validity Check:** N/A
**Post-conditions:** N/A

**Caller:** - ResultController::Update
**Calls:** N/A

### *Method GetGrade*

Retrieves the value in the grade field.

**Requirements:** 11.1-11.2
**Parameters:** N/A
**Return:** String (grade of an assignment for a specific course)
**Data access:** N/A
**Pre-conditions:** N/A
**Validity Check:** N/A
**Post-conditions:** N/A
**Caller:** - AssignmentController::Update
**Calls:** N/A

### *Method GetUser*

Retrieves the value of the user field.

**Requirements:** 11.1-11.2
**Parameters:** N/A
**Return:** User (user that the result belongs to)
**Data access:** N/A
**Pre-conditions:** N/A
**Validity Check:** N/A
**Post-conditions:** N/A
**Caller:** - ResultController::Update
**Calls:** N/A

### *Method SetBelongToAssignment*

Stores an Assignment object in the belongToAssignment field.

**Requirements:** 11.1-11.2
**Parameters:** Assignment (Assignment the result belongs to)
**Return:** Boolean (true if the assignment was successfully stored and validated)
**Data access:** N/A
**Pre-conditions:** N/A
**Validity Check:** - The assignment is an Assignment object and identifies an existing assignment for the specified course in the belongToCourse field.
**Post-conditions:** N/A
**Caller:** - ResultController::Add
- ResultController::Update
**Calls:** N/A

### *Method SetBelongToCourse*

Stores an integer in the belongToCourse field.

**Requirements:** 11.1-11.2
**Parameters:** Integer (ID of the course that the result belongs to)

| | |
|---|---|
| **Return:** | Boolean (true if the ID was successfully stored and validated) |
| **Data access:** | N/A |
| **Pre-conditions:** | N/A |
| **Validity Check:** | - The course ID is an integer and identifies a course that exists |
| **Post-conditions:** | N/A |
| **Caller:** | - ResultController::Add<br>- ResultController::Update |
| **Calls:** | N/A |

### *Method SetGrade*

Stores a String in the grade field.

| | |
|---|---|
| **Requirements:** | 11.1-11.2 |
| **Parameters:** | String (the grade of a specified assignment for a specified course) |
| **Return:** | Boolean (true if the grade was successfully stored and validated) |
| **Data access:** | N/A |
| **Pre-conditions:** | N/A |
| **Validity Check:** | - The grade has to be one of the letters A-F, the letter P or the string "Fx" |
| **Post-conditions:** | N/A |
| **Caller:** | - AssignmentController::Add<br>- AssignmentController::Update |
| **Calls:** | N/A |

### *Method SetUser*

Stores a User object in the user field.

| | |
|---|---|
| **Requirements:** | 11.1-11.2 |
| **Parameters:** | User (user that the result belongs to) |
| **Return:** | Boolean (true if the user was successfully stored and validated) |
| **Data access:** | N/A |
| **Pre-conditions:** | N/A |
| **Validity Check:** | - The input is a User object and identifies a user that exists |
| **Post-conditions:** | N/A |
| **Caller:** | - ResultController::Add<br>- ResultController::Update |
| **Calls:** | N/A |

## Class ResultController

### *Method GetResultByUser*

The method gets all results for a user from the database.

| | |
|---|---|
| **Requirements:** | 4.1 |
| **Parameters:** | **-** Integer (the user ID of the user whose results are to be fetched) |
| **Return:** | ArrayList<Result> (the results corresponding to the user) |
| **Data access:** | Fetches rows from the database table *Result* |
| **Pre-conditions:** | - A connection to the database is established |
| **Validity Check:** | N/A |

**Post-conditions:** N/A
**Caller:** - JSP page for viewing results for a user
**Calls:** - Result set methods

### *Method Update*

The method updates a user's result for an assignment in the database. If no result previously exists for the user, a new result is created. If no result is given, any previous result is deleted.

**Requirements:** 11.1
**Parameters:** **-** Result (the result to be saved, contains an assignment, a user and a grade)
**Return:** Boolean (true if the result for the assignment was successfully changed in the database)
**Data access:** Update, inserted or deleted a row in the database table *Result*
**Pre-conditions:** - A connection to the database is established
**Validity Check:** N/A
**Post-conditions:** One row is updated, inserted or deleted in the *Result* table in the database
**Caller:** - JSP page for updating a result for an assignment
**Calls:** - Result get methods

## Class Session

### *Method Authenticate*

Authenticates a user with the system.

**Requirements:** 1.1
**Parameters:** - String (username)
- String (password)
**Return:** Boolean (true if authentication was successful)
**Data access:** Retrieves values from database table *User*
**Pre-conditions:** - User is not logged in
- A connection to the database is established
**Validity Check:** N/A
**Post-conditions:** - The user state in the current session is set to logged in
**Caller:** - Any page that allows the user to log in.
**Calls:** - UserController::GetUserByUsername
- Cache::Add
- Cache::Get

### *Method Logout*

Logs out a user from the system.

**Requirements:** 1.1
**Parameters:** N/A
**Return:** Void
**Data access:** N/A
**Pre-conditions:** - User is logged in.
**Validity Check:** N/A

**Post-conditions:** - The user state in the current session is set to logged out
- Any temporarily saved user data is removed.
**Caller:** - Any page.
**Calls:** N/A

# Class Schedule

### *Field activities*

**Type:** ArrayList<Activity>
**Access level:** Private

### *Field belongToCourse*

The ID of the course that the schedule belongs to.

**Type:** Integer
**Access level:** Private

### *Method AddActivity*

The method adds an Activity object in the activities ArrayList.

**Requirements:** 7.1-7.5
**Parameters:** **-** Activity (activity to be added in the schedule that belongs to the specified course)
**Return:** Boolean (true if activity was successfully added and validated)
**Data access:** N/A
**Pre-conditions:** N/A
**Validity Check:** - The activity has to be an existing Activity object belonging to the course specified in the belongToCourse field.
**Post-conditions:** N/A
**Caller:** N/A
**Calls:** N/A

### *Method GetAllActivities*

The method retrieves all activities

**Requirements:** 7.1-7.5
**Parameters:** N/A
**Return:** ArrayList<Activity> (All the activities for a course)
**Data access:** N/A
**Pre-conditions:** N/A
**Validity Check:** N/A
**Post-conditions:** N/A
**Caller:** - ScheduleController::GetCourseSchedule
- ScheduleController::GetUserSchedule
**Calls:** N/A

### *Method GetBelongToCourse*

Retrieves the value of the belongToCourse field.

| | |
|---|---|
| **Requirements:** | 7.1-7.5 |
| **Parameters:** | N/A |
| **Return:** | Integer (ID of the course that the schedule belongs to) |
| **Data access:** | N/A |
| **Pre-conditions:** | N/A |
| **Validity Check:** | N/A |
| **Post-conditions:** | N/A |
| **Caller:** | - ScheduleController::GetCourseSchedule<br>- ScheduleController::GetUserSchedule |
| **Calls:** | N/A |

### *Method SetBelongToCourse*

Stores an integer in the belongToCourse field.

| | |
|---|---|
| **Requirements:** | 7.1-7.5 |
| **Parameters:** | Integer (ID of the course that the schedule belongs to) |
| **Return:** | Boolean (true if course ID was successfully added and validated) |
| **Data access:** | N/A |
| **Pre-conditions:** | N/A |
| **Validity Check:** | The course ID is an integer and identifies a course that exists |
| **Post-conditions:** | N/A |
| **Caller:** | N/A |
| **Calls:** | N/A |

## Class ScheduleController

### *Method Export*

The method exports a schedule into the iCalendar format.

| | |
|---|---|
| **Requirements:** | 7.5, 13.3 |
| **Parameters:** | N/A |
| **Return:** | String (representing schedule in iCalendar format) |
| **Data access:** | N/A |
| **Pre-conditions:** | N/A |
| **Validity Check:** | N/A |
| **Post-conditions:** | - Scheduled activities are represented in iCalendar format |
| **Caller:** | - JSP page for viewing a schedule |
| **Calls:** | - Activity get methods. |

### *Method GetScheduleByCourse*

The method generates a list of all activities for a specific course.

| | |
|---|---|
| **Requirements:** | 7.3 |
| **Parameters:** | **-** Course (the course for which news shall be retrieved) |
| **Return:** | ArrayList (containing all scheduled activities for the course) |
| **Data access:** | Retrieve rows from the database table *News* |
| **Pre-conditions:** | - A connection to the database is established<br>- The course exists in the database |
| **Validity Check:** | N/A |

| | |
|---|---|
| **Post-conditions:** | N/A |
| **Caller:** | JSP page for displaying a schedule for a course |
| **Calls:** | - Activity get methods |

### *Method GetScheduleByUser*

The method generates a list of all scheduled activities for courses that a specific user is registered for.

| | |
|---|---|
| **Requirements:** | 2.1 and 2.2 |
| **Parameters:** | **-** User (the user for which a schedule shall be retrieved) |
| **Return:** | ArrayList (containing all scheduled activities for the user's courses) |
| **Data access:** | Retrieves row from the database table *News* |
| **Pre-conditions:** | - A connection to the database is established<br>- The user exists in the database |
| **Validity Check:** | N/A |
| **Post-conditions:** | N/A |
| **Caller:** | JSP page for displaying a schedule for a user |
| **Calls:** | - Activity get methods |

### *Method Import*

The method imports a schedule and stores it in the database.

| | |
|---|---|
| **Requirements:** | 7.1 |
| **Parameters:** | **-** Schedule (represented in iCalendar format) |
| **Return:** | Boolean (true if schedule was successfully stored in the database) |
| **Data access:** | Inserts new rows into the *Activity* database table. |
| **Pre-conditions:** | - A connection to the database is established |
| **Validity Check:** | N/A |
| **Post-conditions:** | - Scheduled activities are inserted into the *Activity* table in the database |
| **Caller:** | - JSP page for importing a schedule |
| **Calls:** | - N/A |

### *Method RemoveScheduleByCourse*

The method removes all scheduled activities belonging to a course.

| | |
|---|---|
| **Requirements:** | 7.1, 13.3 |
| **Parameters:** | String (containing course code) |
| **Return:** | Void |
| **Data access:** | All rows in table *Activity* containing activities for the course are removed. |
| **Pre-conditions:** | - A connection to the database is established |
| **Validity Check:** | N/A |
| **Post-conditions:** | - There are no activities in the database table *Activity* for the course |
| **Caller:** | - JSP page for removing a schedule |
| **Calls:** | N/A |

## Class User

### *Field firstname*
The firstname of a user.

**Type:** String
**Access level:** Private

### *Field lastname*
The lastname of a user.

**Type:** String
**Access level:** Private

### *Field password*
The password of a user.

**Type:** String
**Access level:** Private

### *Field privileges*
The privileges of a user. If true then the user is has the privilege “System Administrator”.

**Type:** boolean
**Access level:** Private

### *Field username*
The username of a user.

**Type:** String
**Access level:** Private

### *Method GetFirstName*
The method retrieves a first name.

**Requirements:** 13.4
**Parameters:** N/A
**Return:** String (first name of the user)
**Data access:** N/A
**Pre-conditions:** N/A
**Validity Check:** N/A
**Post-conditions:** N/A
**Caller:** - UserController::Update
**Calls:** N/A

### *Method GetLastName*
The method retrieves a last name.

**Requirements:** 13.4

**Parameters:** N/A
**Return:** String (Last name of the user)
**Data access:** N/A
**Pre-conditions:** N/A
**Validity Check:** N/A
**Post-conditions:** N/A
**Caller:** - UserController::Update
**Calls:** N/A

### *Method GetPassword*

The method retrieves a password.

**Requirements:** 13.4
**Parameters:** N/A
**Return:** Char (password of the user)
**Data access:** N/A
**Pre-conditions:** N/A
**Validity Check:** N/A
**Post-conditions:** N/A
**Caller:** - UserController::Update
**Calls:** N/A

### *Method GetPrivilege*

The method retrieves the privilege for a user.

**Requirements:** 13.1-13.2
**Parameters:** N/A
**Return:** Boolean (true if the user is a system administrator)
**Data access:** N/A
**Pre-conditions:** N/A
**Validity Check:** N/A
**Post-conditions:** N/A
**Caller:** - UserController::Update
**Calls:** N/A

### *Method GetUsername*

The method retrieves a username.

**Requirements:** 13.4
**Parameters:** N/A
**Return:** String (username of the user)
**Data access:** N/A
**Pre-conditions:** N/A
**Validity Check:** N/A
**Post-conditions:** N/A
**Caller:** - UserController::Update
**Calls:** N/A

### *Method SetFirstName*

The method sets a first name.

| | |
|---|---|
| **Requirements:** | 13.4 |
| **Parameters:** | **-** Firstname (the first name to be stored) |
| **Return:** | Boolean (true if first name was successfully stored in the database) |
| **Data access:** | N/A |
| **Pre-conditions:** | N/A |
| **Validity Check:** | - The username is a string |
| **Post-conditions:** | N/A |
| **Caller:** | - UserController::Update<br>**-** UserController::Add |
| **Calls:** | N/A |

### *Method SetLastName*

The method sets a last name.

| | |
|---|---|
| **Requirements:** | 13.4 |
| **Parameters:** | **-** Lastname (the last name to be stored) |
| **Return:** | Boolean (true if last name was successfully stored in the database) |
| **Data access:** | N/A |
| **Pre-conditions:** | N/A |
| **Validity Check:** | - The lastname is a string |
| **Post-conditions:** | N/A |
| **Caller:** | - UserController::Update<br>**-** UserController::Add |
| **Calls:** | N/A |

### *Method SetPassword*

The method sets a password.

| | |
|---|---|
| **Requirements:** | 13.4 |
| **Parameters:** | **-** Password (the password to be stored) |
| **Return:** | Boolean (true if password was successfully stored in the database) |
| **Data access:** | N/A |
| **Pre-conditions:** | N/A |
| **Validity Check:** | - The password is at least 5 chars |
| **Post-conditions:** | N/A |
| **Caller:** | - UserController::Update<br>**-** UserController::Add |
| **Calls:** | N/A |

### *Method SetPrivilege*

Stores a Boolean in the privilege field.

| | |
|---|---|
| **Requirements:** | 13.1-13.2 |
| **Parameters:** | Boolean (true if the user is a system administrator) |
| **Return:** | Boolean (true if the privilege is successfully stored and validated) |
| **Data access:** | N/A |
| **Pre-conditions:** | N/A |
| **Validity Check:** | The privilege has to be a Boolean. |
| **Post-conditions:** | N/A |

**Caller:** - UserController::Add
- UserController::Update
**Calls:** N/A

### *Method SetUsername*

The method sets a username.

**Requirements:** 13.4
**Parameters:** **-** Username (the username to be stored)
**Return:** Boolean (true if username was successfully stored in the database)
**Data access:** N/A
**Pre-conditions:** N/A
**Validity Check:** - The username is a string and is at least 5 chars long
**Post-conditions:** N/A
**Caller:** - UserController::Update
**-** UserController::Add
**Calls:** N/A

## Class UserController

### *Method Add*

The method stores a user in the database.

**Requirements:** 13.4
**Parameters:** **-** User (the user to be stored)
**Return:** Boolean (true if user was successfully stored in the database)
**Data access:** Inserts a row in the database table *User*
**Pre-conditions:** - A connection to the database is established
**Validity Check:** N/A
**Post-conditions:** - One new row is inserted into the *User* table in the database
**Caller:** - JSP page for adding a user
**Calls:** - User get methods

### *Method Get*

The method gets a user from the database.

**Requirements:** 13.4
**Parameters:** **-** Integer (the user ID of the user to be fetched)
**Return:** User (the user corresponding to the user ID)
**Data access:** Fetches a row in the database table *User*
**Pre-conditions:** - A connection to the database is established
**Validity Check:** N/A
**Post-conditions:** N/A
**Caller:** - JSP page for editing or viewing a user
**Calls:** - User set methods

### *Method GetUserByUsername*

The method gets a user from the database using a username. The username has to match exactly.

| | |
|---|---|
| **Requirements:** | 1.1 |
| **Parameters:** | **-** String (the username of the user to be fetched) |
| **Return:** | User (the user corresponding to the username) |
| **Data access:** | Fetches a row in the database table *User* |
| **Pre-conditions:** | - A connection to the database is established |
| **Validity Check:** | N/A |
| **Post-conditions:** | N/A |
| **Caller:** | - Session::Authenticate |
| **Calls:** | - User set methods |

### *Method Remove*

The method removes a user from the database.

| | |
|---|---|
| **Requirements:** | 13.4 |
| **Parameters:** | **-** User (the user to be removed) |
| **Return:** | Boolean (true if user was successfully removed from the database) |
| **Data access:** | Deletes a row in the database table *User* |
| **Pre-conditions:** | - A connection to the database is established<br>- The user exists in the database |
| **Validity Check:** | N/A |
| **Post-conditions:** | - The user's row is removed from the *User* table in the database |
| **Caller:** | - JSP page for removing a user |
| **Calls:** | N/A |

### *Method Update*

The method updates a user in the database.

| | |
|---|---|
| **Requirements:** | 13.4 |
| **Parameters:** | **-** User (the user to be updated) |
| **Return:** | Boolean (true if user was successfully updated in the database) |
| **Data access:** | Update a row in the database table *User* |
| **Pre-conditions:** | - A connection to the database is established<br>- The user exists in the database |
| **Validity Check:** | N/A |
| **Post-conditions:** | - One row is updated in the *User* table in the database |
| **Caller:** | - JSP page for updating a user |
| **Calls:** | - User get methods |

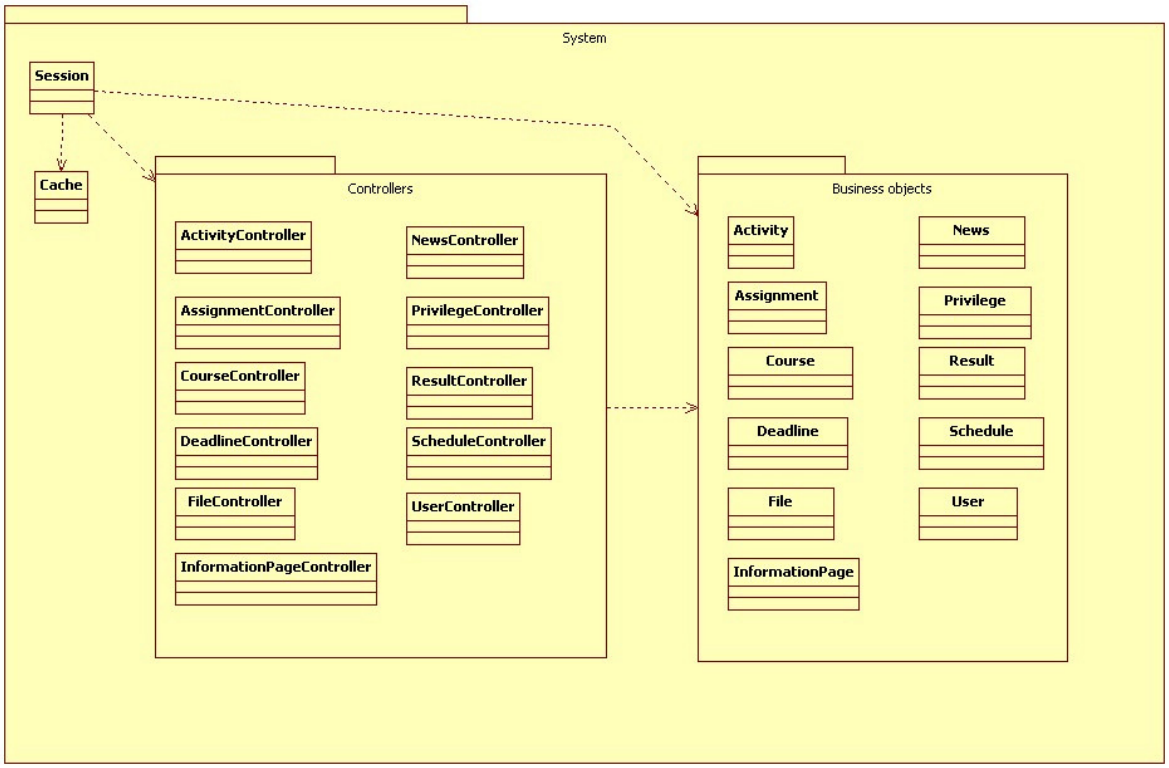
## Implementation Index of Functional Requirements

| Requirement | Implemented in |
|---|---|
| 1.1 | Session::Authenticate |
| | Session::Logout |
| 2.1 | DeadlineController::GetDeadlinesByUser |
| | NewsController::GetNewsByUser |
| | ScheduleController::GetScheduleByUser |
| | ResultController::GetResultByUser |
| 2.2 | Display implemented in JSP page |

| | |
|---|---|
| | DeadlineController::GetDeadlinesByUser |
| | NewsController::GetNewsByUser |
| | ScheduleController::GetScheduleByUser |
| 2.3 | NewsController::GetNewsByUser |
| 3.1 | JSP Page for the creation guide |
| 4.1 | CourseController::UpdateDescription |
| 4.2 | CourseController::GetDescription |
| 5.1 | NewsController::Add |
| | NewsController::GetNewsByCourse |
| | NewsController::Update |
| | NewsController::Remove |
| 5.2 | NewsController::GetNewsByCourse |
| 6.1 | InformationPageController::Add |
| | InformationPageController::Update |
| | InformationPageController::Remove |
| | InformationPageController::GetInformationPageByCourse |
| 6.2 | InformationPageController::Get |
| 7.1 | ScheduleController::Import |
| 7.1 | ScheduleController::RemoveScheduleByCourse |
| 7.2 | ActivityController::Add |
| | ActivityController::Update |
| | ActivityController::Remove |
| 7.3 | ScheduleController::GetScheduleByCourse |
| | ActivityController::Get |
| 7.4 | ScheduleController::GetScheduleByUser |
| 7.5 | ScheduleController::Export |
| 8.1 | DeadlineController::Add |
| | DeadlineController::Update |
| | DeadlineController::Remove |
| 8.2 | DeadlineController::Get |
| 8.3 | DeadlineController::GetDeadlinesByUser |
| 9.1 | FileController::Add |
| | FileController::Update |
| | FileController::Remove |
| 9.2 | FileController::Get |
| | Course::GetAllUsers |
| 10.1 | AssignmentController::Add |
| | AssignmentController::Update |
| | AssignmentController::Remove |
| 10.2 | AssignmentController::GetAssignmentByCourse |
| 11.1 | ResultController::Add |
| 11.2 | ResultController::GetResultByUser |
| 12.1 | Course::GetAllUsers |
| 12.2 | CourseController::Register |
| | Course::AddUser |
| 12.3 | CourseController::Apply |

| | |
|---|---|
| 12.4 | CourseController::Unregister |
| 13.1 | PrivilegeController::AssignCourseLeader |
| | PrivilegeController::AssignCourseAssistant |
| | PrivilegeController::AssignSysAdmin |
| 13.2 | PrivilegeController::AssignCourseAssistant |
| | PrivilegeController::RevokeCourseAssistant |
| 13.3 | CourseController::Add |
| | CourseController::Update |
| 13.4 | UserController::Add |
| | UserController::Update |
| | UserController::Remove |
| 14.1 | Cache::Add |
| | Cache::Get |
| 14.2 | Cache::Add |
| | Cache::Get |
| 14.3 | Cache::Add |
| | Cache::Get |

# 5.6 Package diagram




The system can be roughly divided into two packages, one being the controller package. These classes handle loading and saving of information to the database. The other package is the business object package, where the classes represent relevant domain entities such as news, assignments etc. These objects encapsulate all the data of the related entities and are also responsible for all data validation.