CS 4320 - Software Engineering I - Fall 2019

Assignment 2 - Requirements Analysis

Matt Hudson

[1] Identifying Users

There are four types of users (roles) in this system:

1) **Students.** These are the students enrolled in course(s) in the CS department. Students are the users who are submitting programming work to **TAs** and the **Instructor** for each course in which they are enrolled**.**

2) **TAs.** These are the teaching assistants who receive assignments submitted by **Students**. A TA collects the submissions from **students** in the course(s) they assist in.

3) **Instructor.** The Instructor manages the sections of their course(s), and has the ability to create and adjust settings for assignments. The Instructor also adds and removes **students** and **TAs** from the course(s) they teach. They also receive assignments, though collecting assignments typically is the responsibility of the **TAs.**

4) **Administrators**. Administrators create courses and assign **Instructors** to those courses and its sections. Administrators of the system should be trusted faculty or administrative staff in the CS department.

Note: Individual users in the system can be in different roles in different courses. i.e. one might be a **student** in CS4320 - Software Engineering I, but a **TA** in CS3530 - UNIX Operating Systems. It is also possible for one to be an **Instructor** and an **Administrator.**

[2] Identifying Activities (User Requirements)

For these user requirements, we are only concerned with an assignment **submission system**. That is to say, we are not concerned with keeping track of grades.

Students will:

- View directory information about *courses* in which they are enrolled:
    - Course Code and Title (i.e. CS4320 - Software Engineering I)
    - Course Instructor
    - Section
    - Course TA(s)
    - Contact Information for Instructor and TA(s)
    - Office Hours for Instructor and TA(s)
- View a list of *courses* in which they are enrolled as a students.
- Select a *course* from the list of *courses* in order to:
    - View a list of *assignments* created by the **Instructor**.
    - Construct *submissions* for each *assignment* by uploading file(s) containing their work.

- View their *submission status* for each assignment:
    - If the assignment has been submitted (Y/N).
    - If so, the *date/time* the assignment was submitted.

TAs will:

- View a list of *courses* in which they are a TA.
- Select a *course* from the list of *courses* in order to:
    - View a list of **students** in the course.
    - View a list of *assignments* created by the **Instructor.**
    - Select an *assignment* in order to:
        - View a list of *submission statuses* for each student:
            - If the assignment has been submitted (Y/N).
            - If so, the *date/time* the assignment was submitted.
        - Download a *collection* of submissions for the *assignment*.

Instructors will:

- View a list of *courses* in which they are an Instructor.
- Select a *course* from the list of *courses* in order to:
    - View a list of *sections* of the course
    - View a list of **students** in the course.
    - View a list of *assignments* in the course.
    - Set directory information for their course(s):
        - Contact Information for Instructor and TA(s)
        - Office Hours for Instructor and TA(s)
    - Create *assignments* for their course.
    - Adjust settings for *assignments*, including:
        - Due Date (Date/Time)
        - Whether late submissions are allowed or not
        - Assignment Description
        - Allowed file types for submissions (i.e. only allow .zip files)
    - Add and Remove **students** and **TAs** from the course.
    - Select an *assignment* in order to:
        - View a list of *submission statuses* for each student:
            - If the assignment has been submitted (Y/N).
            - If so, the *date/time* the assignment was submitted.
        - Download a *collection* of submissions for the *assignment*.

Administrators will:

- View a list of *courses* in the system.
- View a directory of *users* in the system.
- Create *courses* and *sections* of those courses.
- Assign a user as the **Instructor** for a course.
- Add and Remove **students** and **TAs** from a course if the **Instructor** is unavailable.

Any user will:

- Create a user account in the system by providing:
  - Their MU paw print as a unique username
  - A password
  - Their MU Email Address

[3] Identifying Entities, Attributes and Constraints

The following is a list of entities/attributes which relates to the Activities described in **[2]**.

Entities

Users - An entity to represent a user of the system. Its attributes include:

- Username
- Password (identified by its hash or salted hash, NOT plaintext)
- Email Address
- List of courses the user is enrolled as a **student** in (no courses designates the user as not a student)
- List of courses the user is a **TA** in (no courses designates the user as not a TA)
- List of courses the user is an **Instructor** for (no courses designates the user as not an Instructor)
- Whether the user is an **Administrator** or not

Course - An entity to represent each course offered by the CS Department. Its attributes include:

- Course Title
- Course Code (e.g. CS4320)
- Instructor (a **User**)
- TA(s)
- Contact Information for TAs and Instructor
- Office Hours for TAs and Instructor
- List of Sections
- List of Students
- List of Assignments

Assignment - An entity to represent an assignment created by the Instructor of a course. Its attributes include:

- Assignment Title
- Assignment Due Date/Time
- Description
- List of Submissions for the assignment
- Allowed filetypes for the assignment (blank specifies *any* file type)
- Course which this Assignment is for

Submission - An entity to represent a student's submission for an assignment. Its attributes include:

- Associated Course and Assignment the submission is for
- User (**Student**) who created the submission
- Date/Time of the submission
- Whether the submission is late (submitted after the Due Date/Time) or not
- File(s) submitted

The following describes each **process** and how entities/attributes relate to each:

Creating Courses and Sections

User (**Administrator)** creates a new Course entity. They specify the Title, Code, and List of Sections. This creates a new Course in the system.

Assigning users to a course as the Instructor

User (**Administrator)** creates a new Course entity or updates an existing Course entity. They specify the Instructor by specifying the appropriate **User** identified by their pawprint**.** This process also updates the appropriate **User** by adding the course to the list of courses they instruct.

Adding and Removing Students/TAs from a course

User (**Administrator** or the associated **Instructor** of the course**)** updates an existing Course entity. They specify the Course, the user, and whether to Add or Remove the user from the course. This updates the **Course**'s List of Students or List of TAs, and the **User** by adding/removing the course from the List of Courses the user is a student or TA in.

Viewing a list of courses in the system

User (**Administrator)** queries for a list of courses. The system returns a list of created Courses and their attributes.

Viewing a directory of users in the system

User (**Administrator)** queries for a list of users. The system returns a list of Users and their attributes. The system should not return the Password attribute unless specified for security purposes.

Viewing Directory Information about a course

User (**Student)** queries for directory information about a course they are enrolled in. The system returns information about the course, including Course Title, Course Code, Instructor, TA(s), Contact Information, and Office Hours.

Viewing a list of courses enrolled as <role>

User (**Student, TA, Instructor)** queries for a list of courses in which they enrolled in as a specific role (Student, TA, or Instructor). The system returns the list of courses from their **User** entity.

Selecting a course from the list of courses

User (**Student, TA, Instructor)** selects a course from the list of courses they are associated with. In this context, 'associated with' means the User is a Student, TA or Instructor in the course. The system changes the currently selected Course to this Course to perform specific actions on the course according to the User's role in the particular course.

Viewing a list of assignments for a course

User (**Student, TA, Instructor**) queries for a list of assignments in the currently selected course. The system returns a list of Assignments and their attributes. The system returns an error if no Course is currently selected.

Selecting an assignment from the list of assignments

User (**Student, TA, Instructor**) selects an assignment from the list of assignments in the currently selected course. The system changes the currently selected Assignment to this Assignment to perform specific actions on the assignment according to the User's role in the currently selected course.

Submitting work

User (**Student)** first selects a Course. They then select an Assignment from the list of Assignments. The Student creates a Submission by uploading file(s). The Submission is created and the Submission Date/Time is set to the moment the Submission entity is created. The User is notified the submission was successful, or if an error was raised in the process.

Viewing submission status for an assignment as a student

User (**Student)** selects a Course and Assignment. The Student queries their submission status for the Assignment. The system returns whether the Student has submitted for the particular assignment.

Viewing a list of students in a course

User (**TA, Instructor)** queries for a list of students in the currently selected Course. The system returns a list of Users who are a Student in the course and their attributes. The system should only show the User's Username and Email Address for security purposes.

Viewing a list of submission statuses for an assignment in a course

User (**TA, Instructor**) selects a Course and Assignment. The User queries for a list of submission statuses for the selected Assignment. The system returns entries for each enrolled Students comprised of the Student's Username, whether they have submitted the assignment, and if so, the Date/Time the Submission was created.

Downloading a collection of submissions for a particular assignment

User (**TA, Instructor**) selects a Course and Assignment. The User queries for a collection of submissions for the selected Assignment. The system returns an archived .zip file containing each Submission's file(s). Each Submission's file(s) will be archived into a .zip file named <Username>.zip.

Setting Directory Information for a course

User (**Instructor**) selects a Course. The Instructor specifies the Contact Information and Office Hours for the Course. The system updates the Course attributes with the new provided information.

Creating an Assignment

User (**Instructor**) selects a Course. The Instructor creates an Assignment by specifying the Name, Due Date/Time, Description, and allowed file extensions. The system creates a new Assignment and sets its attributes with the provided information. The Assignment is associated with the currently selected course.

Adjusting Assignment Settings

User (**Instructor**) selects a Course and Assignment. The Instructor updates an Assignment by specifying a new Name, Due Date/Time, Description, allowed file extensions, or a combination of these attributes. The system updates the specified attributes to the provided information.

Constraints

- The system cannot accept an infinite number of file submissions. In other words, the system must manage its finite internal hard drive storage and will require sufficient space.
- The system must have consistent uptime to enable submissions and queries.

[4] System Requirements

The following are requirements of the system in terms of hardware and design.

Hardware

The system requires:

- A web server such as Apache2 or a similar system to handle requests and replies when users interact with the system.
- A relational database system such as MySQL or a similar system to store information about Users, Courses, Assignments and Submissions.
- A server with sufficient processing power, memory, storage, and network connectivity to run the web server and database server.
- Sufficient additional hard drive space if and when necessary.
- Backup storage and configuration in the case of an unseen accident resulting in data loss.

Design

In terms of components of the system, the system requires a MVC (Model-View-Controller) architecture or similar such design. This design can be implemented using a LAMP or MEAN stack on a server, either physically or on the cloud such as an Amazon EC2 instance. In terms of the MVC design:

Model - A relational database system such as MySQL is needed to host a database for the system, containing tables for Users, Courses, Assignments and Submissions.

View - The view requires a web application and/or mobile application run by Apache2 or another such web server. Users need to interact through a web browser and/or a mobile app to see results and interact with the underlying model.

Controller - PHP or another such scripting language is necessary to perform operations on the Model and construct Views for users of the system.

In terms of Front-End and Back-End, the **View** constitutes the Front-End, whereas the **Model** and **Controller** constitutes the Back-End.