Iteration 4 Documentation

Team:

Gangs of Aggiesthan

Teammates roles:

Product Owner: Utkarsh Chanchlani

Scrum Masters:

Iteration 0	Ankit Garg
Iteration 1	Ranjith Tamil Selvan
Iteration 2	Harish Chigurupati
Iteration 3	Aswin Periyadan Kadinjapali
Iteration 4	Jatin Kamnani

Customer meeting details:

Date: 04/24/2019
Time: 5.30p.m.
Place: Rudder Tower, Room No.1005

Summary:

During the meeting, we summarized the project work we have done to our customer till previous iteration and what we are planning to implement in the next iteration. We shared information regarding SI Leader Session Management System and User Interfaces (UI) for Sessions. Students can view the sessions of SI leaders. Further, we discussed that for last iteration, we will be working on the interface where student is marking attendance for his registered session. Admin can view how many suggestions created by the leader for the course.
.
Additional Inputs from customer:

1. The customer said that he don't want the specific time for each student marking the attendance. Only the student marking the attendance for his registered session is of utmost importance for him.
2. Customer requested for user-friendly admin interface and we assured him to work on the same.
3. Customer also requested us to consider the fact that day-light saving time can change the session time for a particular course during Fall and Spring Semesters. Till now we are only considered the scheduled time of the session.
4. Customer wants us to add the term "Supplemental Instructions" below the logo.
5. Further, customer requested us to create a Comma-Separated (CSV) file for students, thus, generating a record about how many classes they attended as per time.
6. Moreover, customer wants SI leader also to make changes in the session if there is a discrepancy while student marking attendance. Currently, we have given this authority to admin only.

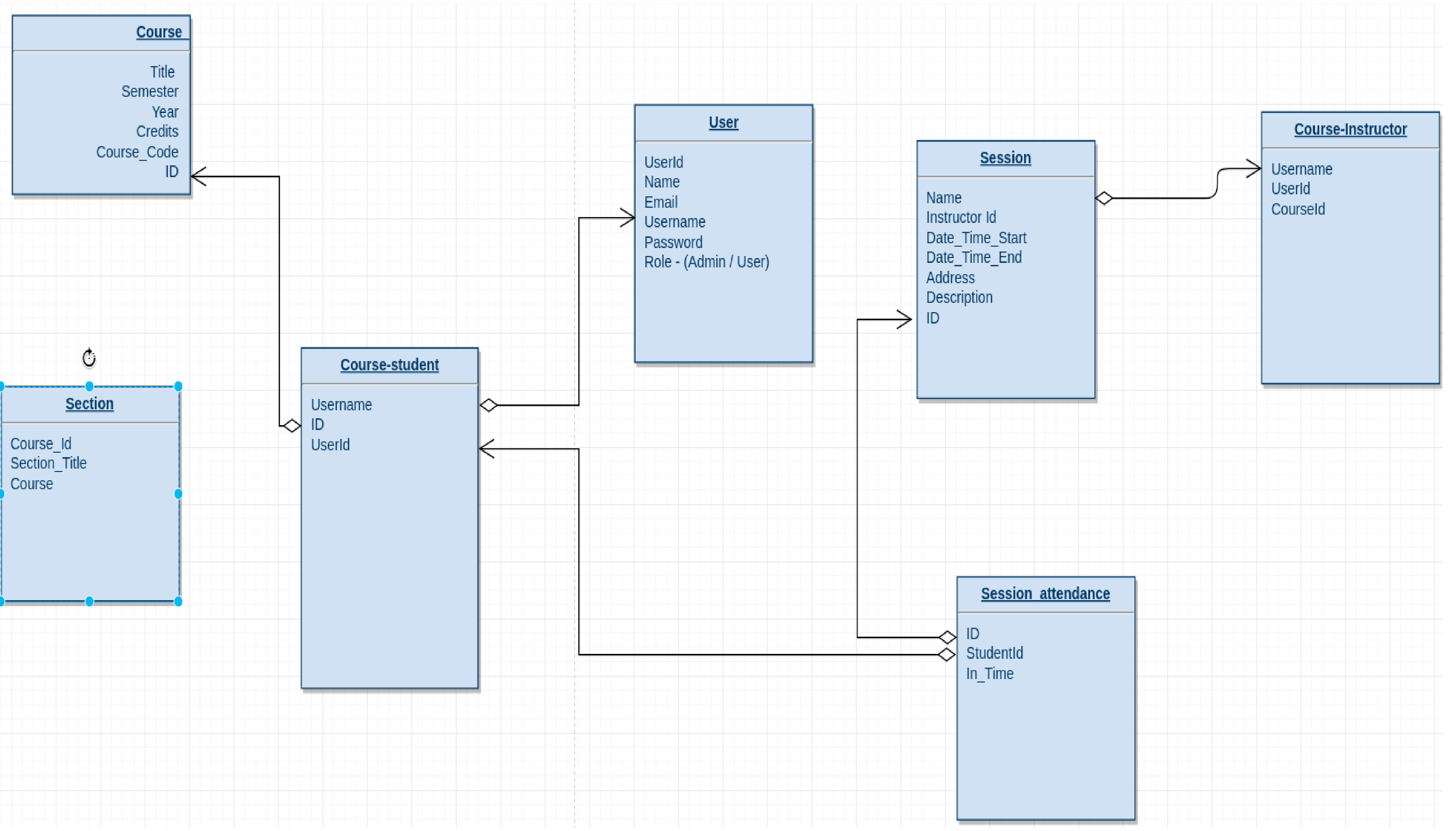
User Stories Implemented in Iteration 4:

Feature: Allowing a User (Student) to mark the attendance for the session he has registered for.

Sub Stories:

1. Getting all the sessions information for a particular student course (backend)
2. Student marking attendance for that course session (backend)
   a) Student marking attendance will be depicted with the color change on button press
3. Displaying Sessions for a student (front-end)
4. Marking attendance for an unattended present session (front-end)
5. Providing aggregation capabilities to admin and SI leaders to view number of sessions or attendance and make some specific changes accordingly.

Design UML Diagram



The Design diagram shows the relationships across the entities of the system, namely- User, Session, Course-student, Section, Course, Session attendance, Course-instructor. The attributes per entity are displayed in the UML diagram.

Each student/SI leader will be associated with a User object with role -user. Course-student is an object per student per course. This object is created by admin for valid student users, i.e., who have registered for that course. The course-student object will be associated with the User (student) once the student has registered with the SI system. As such, Course-student has two foreign-keys, namely Username and UserId, which are primary keys to user and course objects respectively. Each course can have multiple sections. An SI leader can create a session, which will be associated with a session object. As such, a session object will have a foreign key 'InstructorId' to the Course-instructor. Session attendance will be created per session, and will have foreign keys to session and course-student objects.

Pivotal Tracker:

https://www.pivotaltracker.com/n/projects/2318414

Github:

https://github.com/tamu-asc/ascss
Github Tag for Iteration 4 : https://github.com/tamu-asc/ascss/tree/Iteration4

Heroku:

https://tamu-ascss.herokuapp.com/

Auto-deployment:

We deploy the application automatically from the master branch. Master branch is our production branch and any code merged into that will be automatically deployed into the heroku production application. Manual deployments can be triggered via heroku console as well. Master branch is protected which means no one can commit directly into the master branch.

Grading Approach:

Since we are doing a non-legacy project and setting up the platform from scratch, and the user requirements are high, we are concentrating more on the behavior of the features. Since we focused more on the setup and implementation of the first-level features, we have not followed Test Driven Development.