HW-6: Code 1 and Persistence

CSE 308- Section 2

10/13/15

*We don’t need no Documentation*

Dan Harel

Daniel Myrick

Safa Sattar

Jian Xie

[Overview:](#h.1fob9te) 2

[Assumptions:](#h.3znysh7) 2

[Functional Requirements:](#h.2et92p0) 3

[Other Requirements:](#h.tyjcwt) 8

[Use Case Diagram:](#h.3dy6vkm) 9

[Use Cases:](#h.1t3h5sf) 10

System Architecture: 17

Class Model: 19

Graphic User Interface and System Flow: 20

Languages, Technologies and Tools: 22

Dynamic Models: 23

[Contributions:](#h.4d34og8) 26

# Overview:

A testing center was created in Frey Hall during its recent renovation. The University has been looking for software to schedule use of the testing center. Currently, all such software is either proprietary or not designed for a University testing center. This requirements document is based on the University’s actual needs. Ideally, DoIT will obtain a deployable system from this course project (aside from some system integration issues).

The system serves two main purposes:

1. An instructor can submit a request to have their students take an online exam in the Testing Center. Instructors specify the duration of the exam and a time period during which students may take the exam.
2. A student on the roster for a course can schedule an appointment to take an exam for the course at any time during the time period specified by the instructor.

Note that the testing center (unlike a classroom) is not used by only one class at a time. Different students may concurrently take different exams for different classes.

# Assumptions:

Each course has a unique identifier. The system does not constrain the exact format of course identifiers, beyond assuming that they are short strings. At Stony Brook, course identifiers are 6-character strings, e.g., CSE308.

Each class has a unique identifier consisting of the course identifier, the section number, an underscore, and the term code. The system does not constrain the exact format of section numbers (which can be alphanumeric, despite being called “numbers”) or term codes, beyond assuming that they are short strings. At Stony Brook, term identifiers are 4-digit numbers; for example, Fall 2015 is 1158, Winter 2016 is 1161, Spring 2016 is 1164, Summer 2016 is 1166, and Fall 2016 is 1168. For example, the class identifier for CSE308 section 01 in Fall 2015 is CSE30801\_1158.

Each exam has a unique identifier consisting of the class identifier, “\_ex”, and the exam number. For example, the exam identifier for the second exam in CSE30801\_1158 is CSE30801\_1158\_ex2.

For each term, the testing center has specified hours, which specify a range of hours that the test center is open on each day of the week. At Stony Brook, the testing center’s hours in Fall 2015 are Monday-Friday, 8am10pm;Saturday and Sunday, closed. As exceptions to the standard hours, the testing center is closed during specified date ranges (e.g., Thanksgiving recess).

All appointment timeslots start on the hour or half-hour.

All seats in the testing center are equivalent. In other words, any exam can be taken at any seat. Seats are numbered (1,2, …), such that adjacent seats have consecutive numbers. This implies that seats with nonconsecutive numbers are not adjacent. Note that seats with consecutive numbers are not necessarily adjacent, e.g., the last seat in one row and the first seat in the next row, but we do not model the room layout in sufficient detail to take this into account.

In the production version of this system, user account management and authentication will be handled primarily through integration with the University’s Central Authentication Service (CAS). DoIT will handle that. For this project, we assume the system has access to a preexisting user account database that contains each user’s NetID, first name, last name, email address, hashed password, and roles (administrator, instructor, student). Note that a user may have multiple roles. Code that accesses the user account database should be encapsulated in one component, to facilitate replacing it with CAS client code.

# Functional Requirements:

**Administrators:**

1. Edit testing center information. This includes the:
   1. number of seats in the testing center. Stony Brook’s testing center currently has 64 seats.
   2. number of set-aside seats. A set-aside seat is a seat that cannot be reserved directly by students. It can be reserved only by an administrator, making an appointment on behalf of a student. Set-aside seats give administrators flexibility to deal with situations such as malfunctioning computers and students late for appointments due to emergencies.
   3. hours of the testing center for any future term.
   4. date ranges that the testing center is closed during the current term or a future term.
   5. time periods during the current or a future term when the testing center is reserved for a non-Stony Brook exam (e.g., MCAT). The entire testing center is reserved during these time periods.
   6. gap time. The gap time is a lower bound on how long a seat must be unassigned between appointments. The gap time may range from 0 to 30 minutes. For example, if a student has a 9am appointment for an 80-minute exam, and the gap time is 10 minutes, then another student may be assigned the same seat for a 9:30am appointment. If the gap time is 30 minutes, then the seat cannot be assigned to another student until the 10am timeslot (recall that all timeslots start on the hour or half-hour).Changes to the gap time affect subsequent appointment requests but do not affect existing appointments.
   7. reminder interval. For each appointment to take an exam, the system sends the student a reminder by email. The reminder interval specifies the number of minutes before the appointment when the reminder message is sent.
2. Import data. The user specifies a term and the name of a folder in the server’s filesystem that contains files with course registration data for that term. The system discards existing data for that term and imports new data for that term from those files. The files should be named user.csv, class.csv, and roster.csv. They are in comma separated value (csv) format. Each file contains a header line with column names (the system can treat this line as a comment) followed by data lines. Each data line in user.csv has the format: FirstName, LastName, NetID, Email. Each data line in class.csv has the format: ClassID, Subject, CatalogNumber, Section, InstructorNetID. Each data line in roster.csv has the format: NetID, ClassID. ClassID is a unique identifier for a section of a course in the context of a specified term. Subject and CatalogNumber together form the course identifier; for example, for CSE 308, the Subject is “CSE”, and the Catalog Number is “308”. The format of the ClassID in these datafiles is determined by the data export utility and is not necessarily the same as the more readable format for class identifiers described in Section 2. Although ClassID in the sample data incorporates the term identifier, this is not guaranteed, Therefore, the ClassID needs to be paired with the term identifier to create a key guaranteed to be unique across terms.

**Superfluous Appointments**. If a student has an appointment for a course exam, and does not appear on the updated roster for the course, then the system marks the appointment as superfluous, releases the assigned seat, and sends email notifications of this to the student, instructor, and administrator. Conversely, if the database contains a superfluous appointment for a course exam, and the associated student appears on the updated roster for the course, then the system reinstates the appointment if possible (i.e., if a seat is available), and in any case sends email notifications to the student, instructor, and administrator. The main motivation for the latter requirement is to easily undo the damage if a bundled data import incorrectly causes appointments to be marked as superfluous.

1. Display the utilization of the testing center for each day in any specified date range. For a past or current day d , the system shows the actual utilization, defined by:

where *Appts(d)* is the set of appointments on day *d* , *duration(a)* is the duration of appointment a including gap time after the appointment rounded up to the beginning of the next timeslot, *N\_seat* is the number of seats in the testing center, and *OpenTime(d)* is the duration that the testing center is open on day *d* . For a future day *d* , the system shows the expected utilization, defined by:

where *Exams(d)* is the set of exams whose date range includes *d* , *duration(e)* is the duration of exam *e*, *gap* is the gap time, *N\_student(e)* is the number of students who need to take exam *e* , *N\_appt(e)* is the number of existing appointments to take exam *e* , and *N\_day(e)* is the number of days in the date range for exam *e*. For simplicity, this definition ignores the start and end times of the exam.

1. Approve or deny an exam scheduling request for the current or a future term.

**Schedulability**. A request can be approved only if it is schedulable. A request is schedulable if, after it is added to the schedule, the testing center has enough seats for all students to take all of their scheduled exams during the specified time periods, taking into account both existing appointments and necessary future appointments. Note that schedulability varies over time, even if nothing else changes. For example, suppose an instructor schedules exam1 over a 3-day period, 200 students needs to take it, and at most 100 students can take it per day, based on the number of seats, the testing center hours, and the exam duration. Suppose no other exams are scheduled. Suppose another instructor wants to schedule exam2 on the last day of that 3-day period, 100 students need to take it, and at most 100 students can take it per day. If the request for exam2 is submitted before the 3-day period starts and before any students make appointments for exam1, then the request is schedulable, because it is possible for 200 students to take exam1 on the first 2 days of the 3-day period, and 100 students to take exam2 on the last day of that period. However, if the request is submitted partway into the 3-day period, and one or more timeslots have gone unused, then the request is not schedulable.

**Utilization**. This schedulability test does not ensure any scheduling flexibility for students: the last student to make an appointment may have only one timeslot to choose from. To help the administrator decide whether students will have sufficient scheduling flexibility, the system displays, for each day in the date range of the request, the utilization of the testing center based on existing reservations and the resulting utilization if the current request is approved.

1. Make an appointment on behalf of a student in the current term. The administrator indicates whether the seat being reserved is a set-aside seat, being used to deal with an exceptional situation, or a regular seat, being reserved on behalf of a walkin, i.e., a student who happens to comes to the test center to take an exam without an appointment. The system performs the same validity checks as when a student makes an appointment, except that, if a set-aside seat is being reserved, the availability test (item e) is modified to check whether a set-aside seat is available.
2. View appointments. The system displays all appointments and the number of available seats at the current time or a specified other time.
3. Cancel or modify any appointment in the system.
4. Check-in a student for an appointment. The system records that the student kept the appointment and displays the student’s seat assignment. (Ideally, students would check in by swiping their ID card, but that is beyond the scope of this course project.)
5. Generate reports. For each report, a specified term can be a past, current, or future term. For web-based systems, each report can simply be displayed in the browser, and the browser’s Save or Print function can be used to save it to a file. Non-web-based systems should also provide the ability to save a report to a file. Plain text format is acceptable, although better formatting (e.g., HTML) is preferable.
   1. For each day in a specified term, report the number of student appointments on that day.
   2. For each week in a specified term, report the number of student appointments that week and the course identifiers of courses associated with any of those appointments.
   3. For a specified term, report the courses that used the testing center in that term.
   4. For a specified range of terms, report the total number of student appointments in each term.

**Instructors:**

1. Request to schedule an exam in the testing center in the current or a future term.
   1. If the exam is for a course, the instructor provides the course identifier, section number, term, duration of the exam; the start date and start time, and the end date and end time For example, if the start date and time are August 25 at noon, and the end date and time are Aug 28 at 5pm, then all appointments to take that exam must be completely within that time period.
   2. If the exam is ad hoc (i.e., not for a course), for example a placement exam or language proficiency exam, the instructor provides the name of the exam, the duration, a list of students taking the exam, the start date and start time, and the end date and end time. The list of students is entered into a text box. Each line should have the format: NetID, LastName, FirstName.

The system rejects unschedulable requests. If the request is schedulable, the system displays, for each day in the date range of the request, the utilization of the testing center based on existing reservations and the resulting utilization if the current request and all pending requests to schedule exams are approved. The instructor can then choose whether to cancel or submit the request.

1. See a list of his or her exam scheduling requests in the current and future terms. The system should display some details of each request, including the current status (pending, approved, or denied) and, for approved requests, how many students should take the exam, and how many students have taken the exam.
2. Cancel a pending exam scheduling request.
3. See appointment and attendance details for a specified exam, including each student’s appointment time (if any), assigned seat, and whether the student showed up for the appointment.

**Students:**

1. Make an appointment to take an exam. The system ensures:
   1. the student is enrolled in the course in the current term or is on the list of students who need to take the ad hoc exam.
   2. the student does not have an existing appointment for the same exam.
   3. the student does not have an appointment for a different exam in an overlapping timeslot.
   4. the appointment is entirely between the start datetime and end datetime of the exam.
   5. a non-set-aside seat is available at the requested time, taking into account the gap time between appointments.

If these conditions are satisfied, the system assigns the student an available seat, trying to avoid placing students taking the same exam in adjacent seats. For simplicity, the system just tries to avoid placing students taking the same exam in consecutively numbered seats, ignoring the fact that some consecutively numbered seats are not adjacent.

1. Cancel an appointment. A student may cancel an appointment any time prior to 24 hours before the appointment, but not after that.
2. See a list of his or her appointments in a specified past, current, or future term.
3. Appointment reminder. For each appointment to take an exam, the system sends the student a reminder by email. The reminder interval specifies the number of minutes before the appointment when the reminder message is sent. If the appointment is made less than the reminder interval before the appointment time, no reminder is sent.

# Other Requirements:

**Authentication:** All access to the system requires authentication with a password.

**Network Security:** Communication is secured using HTTPS/SSL.

**Application Security:** The application should have reasonable protection against common vulnerabilities such as listed in the OWASP Top 10 ( https://www.owasp.org/index.php/Top10 ). The application should not be vulnerable to URL replay attacks. The demo must include a test case that involves a URL replay attack. For example, an administrator logs in, navigates to the “View appointments” page, copies the URL, pastes it in another browser, and tries to view appointments in the second browser.

**FERPA compliance.** The system should be FERPA compliant. This implies that students can see only their own information. They cannot see any information about other students.

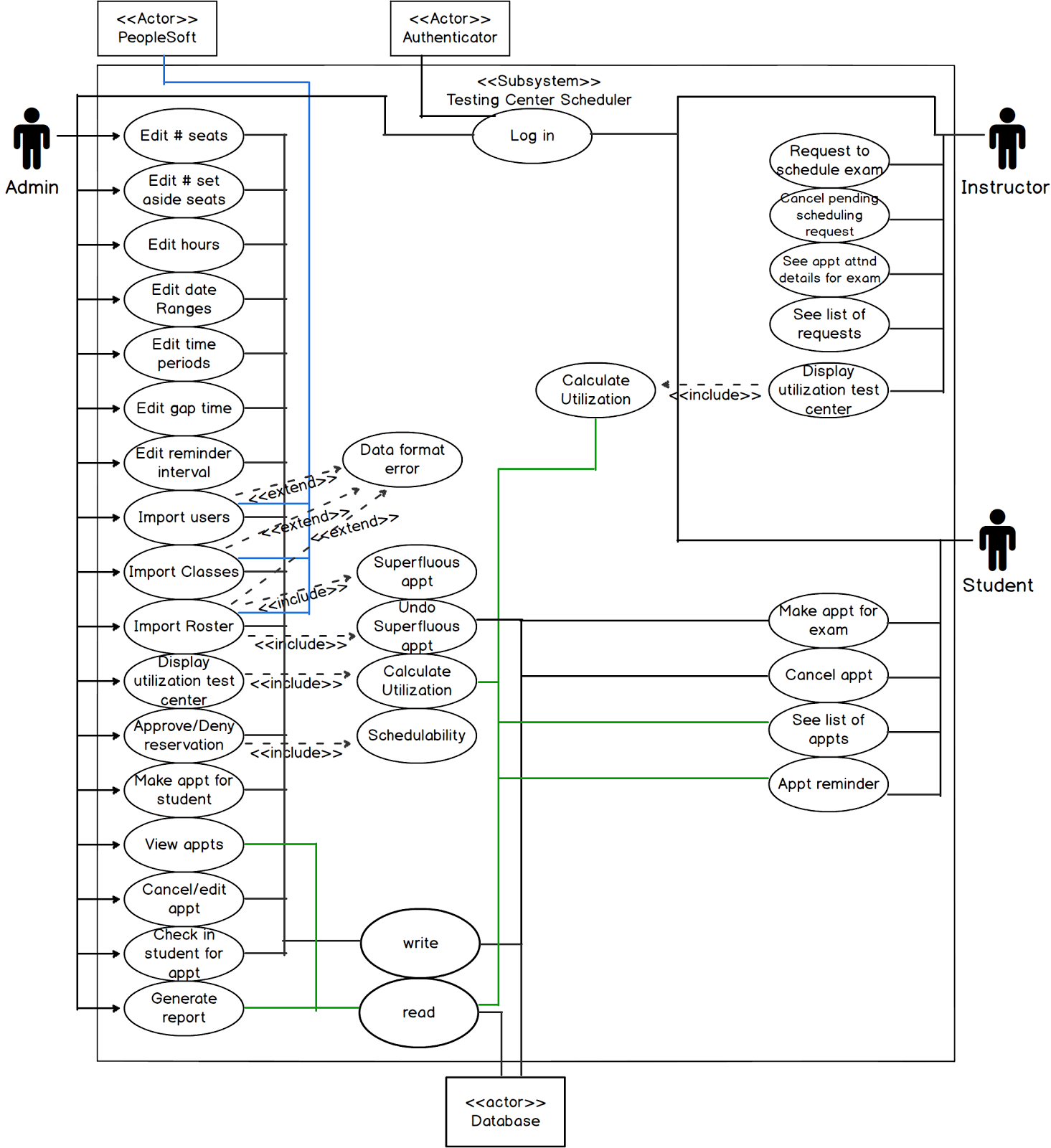
**Concurrency:** [20150831: modified the following sentence to include all kinds of users.] Multiple users on different computers may use the system concurrently. Synchronization is used to ensure sensible behavior. The system may prevent the same user from having multiple concurrent login sessions; if it does not prevent this, it should ensure sensible behavior.

**Client-Server Architecture:** Clients may run concurrently on different hosts. All clients access the same database.

**Programming language and Web development framework**. The system may be implemented using any programming language and Web development framework. Stony Brook DoIT’s production environment uses Microsoft’s ASP.NET framework. Teams are encouraged to use that framework, if they are interested in seeing their system go into production use.

**User friendliness.** The system should have reasonable response time, display confirmation messages when requests have been processed successfully, and display informative error messages when a request cannot be completed or an error occurs.

# Use Case Diagram:



# Use Cases:

**Edit # of seats in testing center**  
Primary Actor: Admin  
Precondition: The physical # of seats changes, admin is logged in  
Primary Flow: 1. Admin edits # of total seats in testing center;   
 2. Admin submits form to save changes  
Success: Admin receives confirmation  
  
**Edit # of set aside seats**  
Primary Actor: Admin  
Precondition: Admin is logged in  
Primary Flow: 1. Admin edits # of total seats to set aside in testing

center;

2. Admin submits form to save changes  
Success: Admin receives confirmation  
  
**Edit hours for future term**  
Primary Actor: Admin  
Precondition: Admin is logged in  
Primary Flow: 1. Admin edits # of hours available in testing center;   
 2. Admin submits form to save changes  
Success: Admin receives confirmation  
  
**Edit date ranges that test center is closed**  
Primary Actor: Admin  
Precondition: Admin is logged in  
Primary Flow: 1. Admin edits date ranges testing center is closed;  
 2. Admin submits form to save changes  
Success: Admin receives confirmation

**Edit time periods (current or future terms) that test center is reserved for non-StonyBrook exams**Primary Actor: Admin  
Precondition: Admin is logged in; time period is available for all

seats

Primary Flow: 1. Admin edits time periods that test center is reserved

For non-StonyBrook exams;

2. Admin submits form to save changes

3. Admin will not be able to reserve more seats during  
 this time period without cancelling this reservation.

Success: Admin receives confirmation.

**Edit gap time**  
Primary Actor: Admin  
Precondition: Admin is logged in  
Primary Flow: 1. Admin edits field for Gap time;  
 2. Admin submits form to save changes  
Success: Admin receives confirmation  
  
**Edit reminder interval**  
Primary Actor: Admin  
Precondition: Admin is logged in  
Primary Flow: 1. Admin edits field for Reminder Interval;  
 2. Admin submits form to save changes  
Success: Admin receives confirmation

**Import Users**  
Primary Actor: Admin  
Precondition: Admin is logged in  
Primary Flow: 1. Admin requests to update users  
 2. Admin submits form to save changes  
Success: Admin receives confirmation, Display updated users  
  
**Import Classes**  
Primary Actor: Admin  
Precondition: Admin is logged in  
Primary Flow: 1. Admin requests to update classes  
 2. Admin submits form to save changes  
Success: Admin receives confirmation, Display updated classes  
  
**Import Roster**  
Primary Actor: Admin  
Precondition: Admin is logged in  
Primary Flow: 1. Admin requests to update roster  
 2. Admin submits form to save changes  
Success: Admin receives confirmation, Display updated roster

**Superfluous appointment**  
Precondition: Student has an appointment for a course exam  
Trigger: Roster is updated; student is no longer registered for

course

Primary Flow: 1. System marks the appointment as superfluous;  
 2. System releases the assigned seat;  
 3. System sends email notifications to the student,

instructor and administrator

Success: Student is released from the appointment

**Undo Superfluous appointment**  
Precondition: Database contains a superfluous appointment for the

associated student and course exam

Trigger: Roster is updated; Student appears on the

updated roster for the course

Primary Flow: 1. System reinstates the appointment if seat is

available;  
 2. System notifies student, instructor and admin of the

change

Success: Student is re-added to the exam and is notified.  
  
**Display utilization of testing center**Primary Actor: Admin  
Precondition: Admin is logged in  
Primary Flow: 1. Admin requests to see utilization of testing center

within a specified date range

2. System calculates utilization for the date range

Success: System displays utilization of the testing center  
  
**Approve or deny reservation**

Primary Actor: Admin

Precondition: Admin is logged in. A scheduling request was made.  
Trigger: A request is made to create an appointment at the testing

center

Primary Flow: 1. System checks whether there are enough seats for all

students to take all of their scheduled exams during the

specified time periods

Success: Admin approves or denies a reservation

**Make an appointment on behalf of a student**  
Primary Actor: Admin  
Precondition: Admin is logged in; time period is available for all

seats  
Primary Flow: 1. Admin enters student's NetID, seat number, time slot  
 2. Admin indicates whether the seat being reserved is a

set-aside seat or a regular seat;

3. Admin submits form to save changes

Success: Admin receives confirmation, Student receives

confirmation

**View appointments**  
Primary Actor: Admin  
Precondition: Admin is logged in  
Primary Flow: 1. Admin requests to view appointments within a specified

time

2. System collects appointment information for specified time

Success: Appointments and the number of available seats at the

specified time are displayed on screen

**Cancel or edit an appointment**  
Primary Actor: Admin  
Precondition: Admin is logged in  
Primary Flow: 1. Admin requests to view appointments within a specified

time;

2. Admin selects an appointment;  
 3. Admin edits or cancels this appointment;  
 4. Admin submits form to save changes

Success: Admin receives confirmation of changes  
  
**Check in a student for an appointment**

Primary Actor: Admin

Precondition: Admin is logged in; the student is registered for an

Appointment

Primary Flow: 1. System records that the student kept the appointment

Success: Student is checked in, seat assignment is displayed

**Generate report for each day in a specified term**  
Primary Actor: Admin  
Precondition: Admin is logged in  
Primary Flow: 1. Admin requests report for each day in a specified

term;  
 2. Admin specifies whether to save or print the report to

a file

Success: System generates report for the admin

**Request to schedule an exam**  
Primary Actor: Instructor  
Precondition: Instructor is logged in  
Primary Flow: 1. Instructor requests to schedule an exam for a current

or future term;

2. For a course, the instructor provides the course

identifier, section number, term. For an ad hoc exam

instructor specifies the name of the exam.

3. Instructor specifies the duration of the exam, start

date and start time and the end date and end time;

4. If the request is schedulable, system displays for each day in the date range of the request, the utilization of the testing center based on existing reservations and the resulting utilization if the current  
request and all pending requests to schedule exams are approved;  
5. Else, instructor is prompted to select another time period;  
6. Instructor chooses to cancel or submit the request

Success: On submit, instructor receives confirmation, on cancel,

instructor encounters a dialog box

Instructor's scheduling requests and testing center

schedule are updated

**See list of scheduling requests in prior, current, future terms**  
Primary Actor: Instructor  
Precondition: Instructor is logged in  
Primary Flow: 1. Instructor requests to see list of scheduling requests  
Success: System displays total requests made by the instructor  
  
**Cancel pending scheduling request**  
Primary Actor: Instructor  
Precondition: Instructor is logged in  
Primary Flow: 1. Instructor requests to see list of scheduling

requests;  
2. Instructor requests to cancel a pending request

Success: Instructor receives confirmation, students enrolled in

the course are notified Scheduling requests and testing center schedule are updated

**See appointment and attendance details for a specific exam**  
Primary Actor: Instructor  
Precondition: Instructor is logged in  
Primary Flow: 1. Instructor requests to see appointment and attendance

details for a specific exam

2. Appointment and attendance records for exam are

compiled

Success: Compiled records for specified exam are displayed to the

instructor

**Make an appointment to take an exam**  
Primary Actor: Student  
Precondition: Student is logged in, student is enrolled in course or is

on the list of students who need to take the ad hoc exam;

the student does not have an existing appointment for the

same exam, the student does not have an appointment for a

different exam in an overlapping time-slot, the appointment is entirely between the start date-time and end date-time of the exam, a non-set aside seat is available at the requested time

Primary Flow: 1. Student fills out a form to make an appointment;

this includes date-time and course information or the

ad-hoc selection;

2. student submits form

Success: System returns seat number and testing information to

student as confirmation

**Cancel appointment**  
Primary Actor: Student  
Precondition: Student is logged in  
Primary Flow: 1. Student views all appointments;

2. student selects appointment to cancel

Alternative Flow: 2. If it is within 24 hours before the exam, system does

not allow the student to cancel

Success: Student's appointments are updated  
  
**See a list of his/her appointments**  
Primary Actor: Student  
Precondition: Student is logged in  
Primary Flow: 1. Student requests to view all appointments

2. List of appointments is compiled

Success: Student’s appointments are displayed

**Appointment reminder**

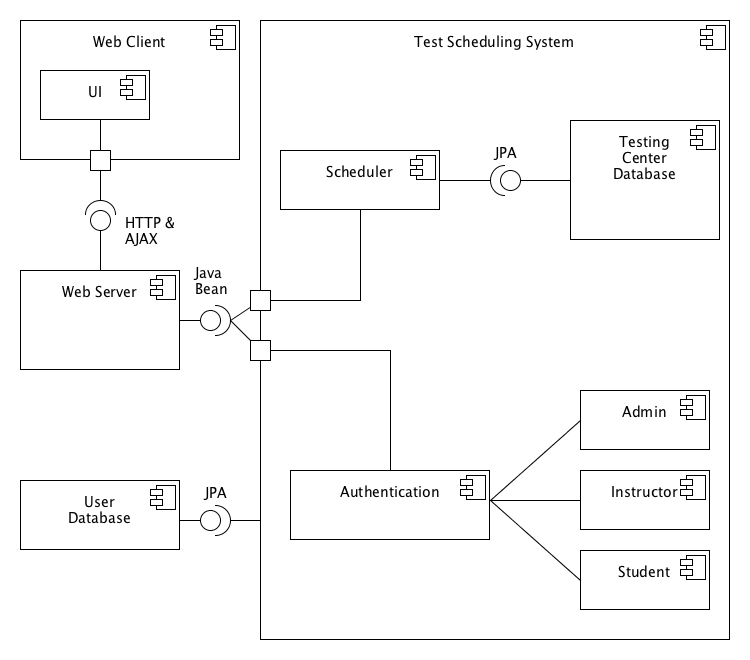
Primary Actor: Student

Precondition: Student has made an appointment before reminder interval  
Trigger: Correct time interval is reached  
Primary Flow: 1. System checks to see if student still has appointment

2. System sends the student a reminder by email

Success: Student receives email

# System Architecture:



**Web Client**

The web application that the user interacts with. Communicates with the Web Server.

**UI**

User Interface. Displays information that is fetched from the server. Provides an interface for the user to request or modify information.

**Web Server**

Sends and receives information to and from the client. Provides a bridge between the client and the system.

**Test Scheduling System**

The entire system that manages the testing center availability and scheduling, as well as storing and retrieving the information in a database.

**Scheduler**

Main logic for the scheduler. Determines which requests are possible. Communicates with the database.

**Testing Center Database**

Stores information on exams, exam schedules, and seating.

**Authentication**

Authenticates users. Accesses user database. Will later be replaced by the Stony Brook Central Authentication Service (CAS).

**Admin**

Testing Center administrator. Represents a user with the most privileges.

**Instructor**

Course instructor. Has the ability to request an exam and see exam seating.

**Student**

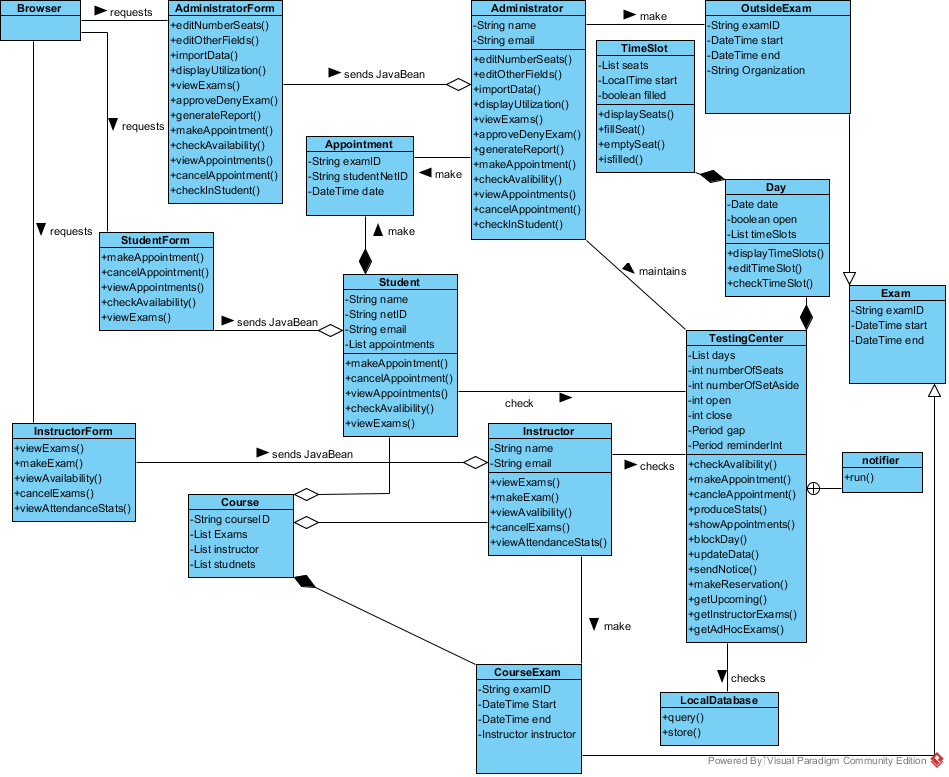
Student. Has the ability to enroll in an exam.

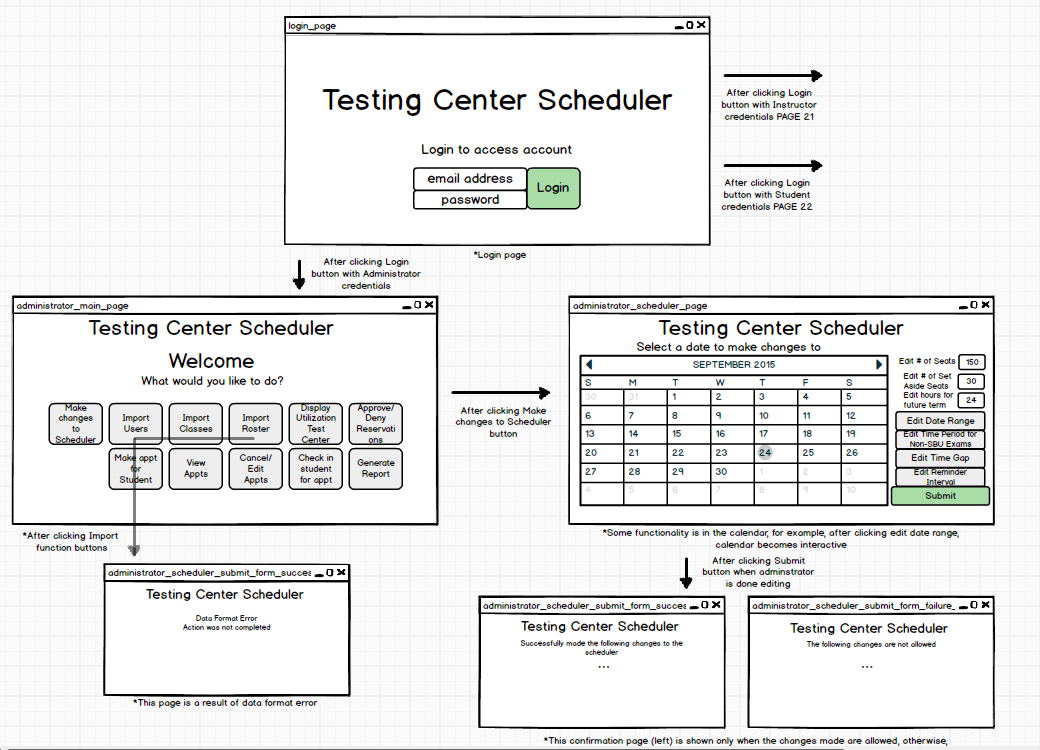
**User Database**

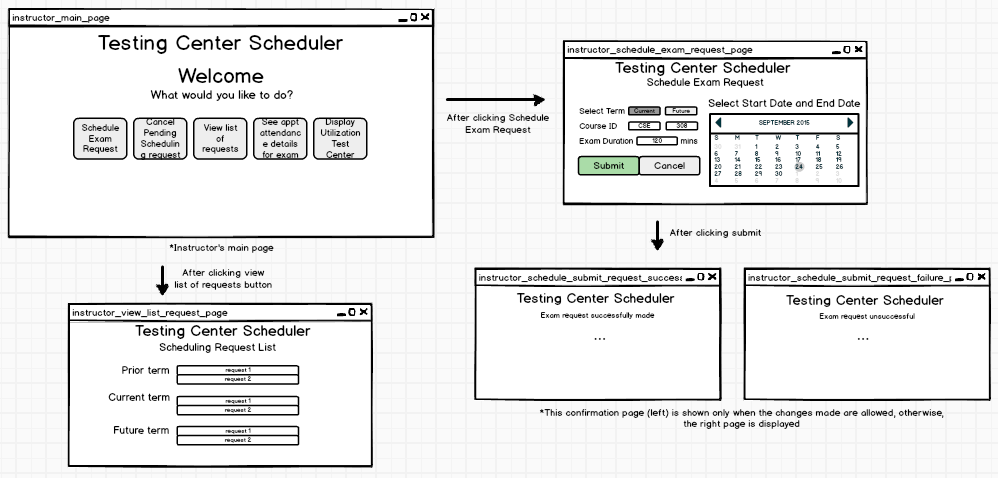
Database that stores user roles and permissions.

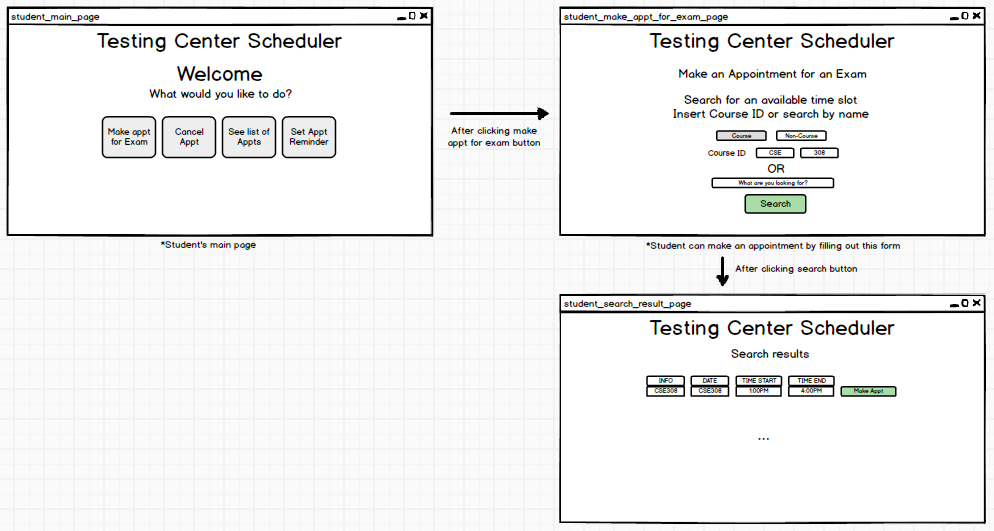
# Object Model (Class Model):

# Graphical User Interface and System Flow:









# Languages, Technologies and Tools:

Use Case Diagram: Balsamiq

Component Diagram: UMLet

Class Diagram: Visual Paradigm

GUI layout: Balsamiq

Dynamic Diagrams: UMLet/Visual Paradigm

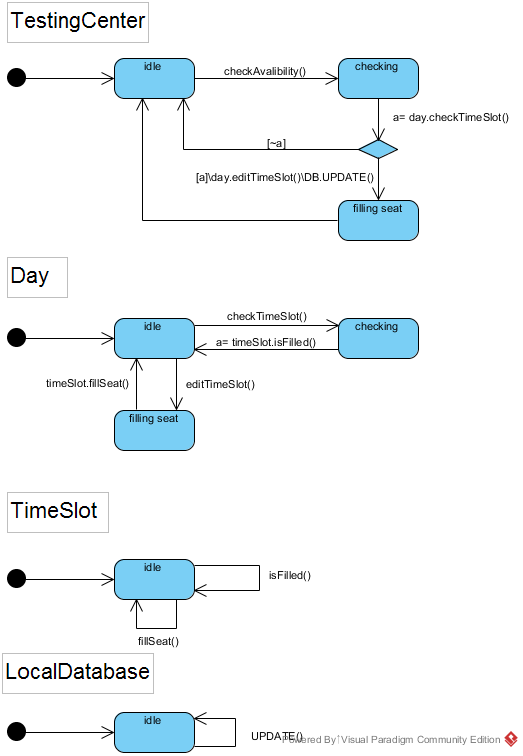
Back End Services: Java

Front End Services(UI): HTML5, JavaScript, CSS3, Bootstrap, Ajax,

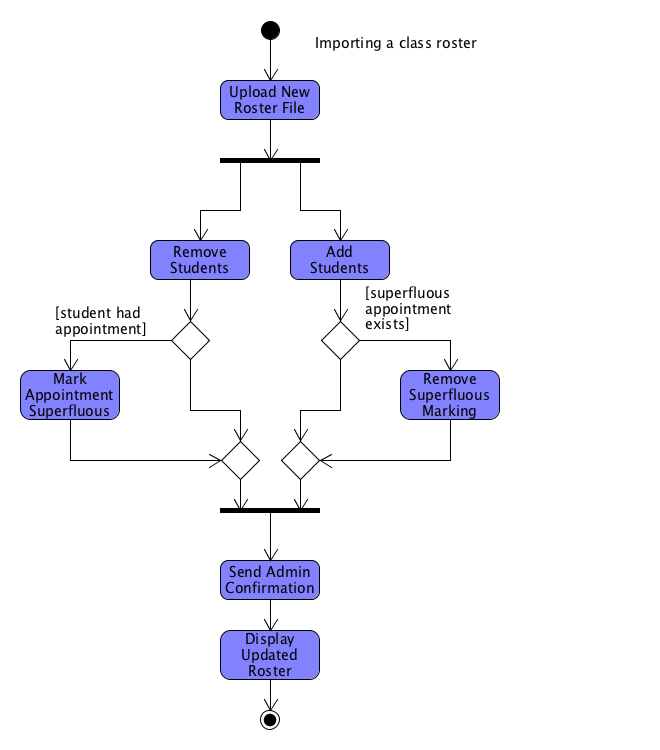
Persistence: JPA

Database: MySQL

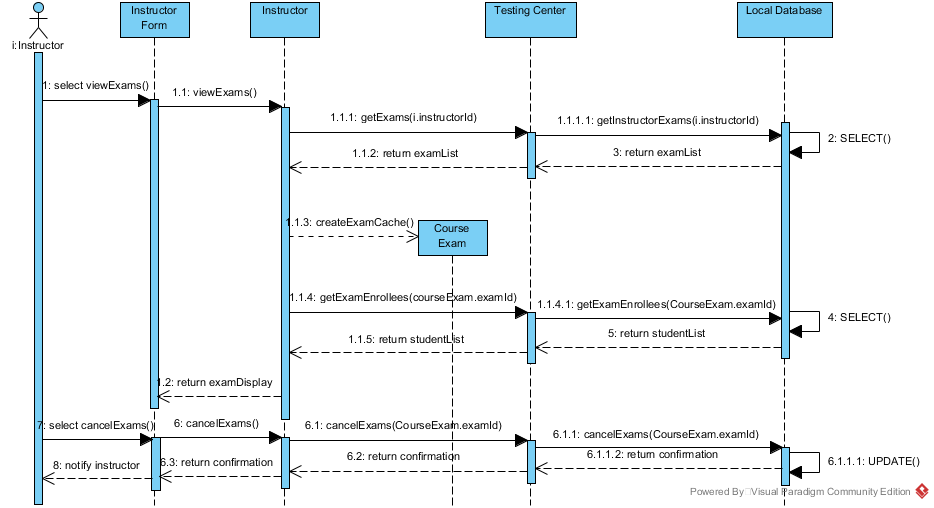
# Dynamic Models:



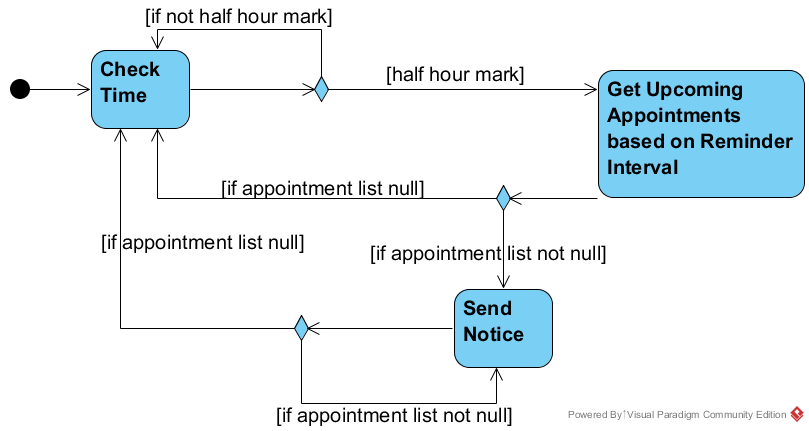
State Machine for Testing Center checking availability and making an appointment



Activity Diagram for updating the class roster and handling superfluous appointments



3 Sequence Diagram for an instructor canceling an exam. This includes him first viewing the exams and the exams being cached locally as objects.



Activity Diagram within the TestingCenter class which shows the process of sending reminder notices to students who have exams the next day

# Contributions:

**Dan Harel: HW2:**

Writing and Editing of Use Cases

Creation of Use Case Diagram

**HW3:**

System Architecture section

**HW4:**

Superfluous Appointment Activity Diagram

**Daniel Myrick: HW2:**

Copied and reformatted the Overview and Requirements from the provided materials.

Writing and Editing of Use Cases

**HW3:**

Back End classes of class diagram

**HW4:**

Test Center State Machine Diagram

**Safa Sattar: HW2:**

Writing, Scribing and Editing of Use Cases

Creation of Use Case Diagram

**HW3:**

GUI Design

Front End classes of class diagram

**HW4:**

View and Cancel Exam Sequence Diagram

**Jian Xie: HW2:**

Editing of Use Cases

Creation and Scribing of Use Case Diagram

**HW3:**

GUI Design

GUI mock-up

**HW4:**

Send Email Reminder Activity Diagram