

Scheduling System for Math Department Resources

Matthew Mooney

Brandon Rozzi

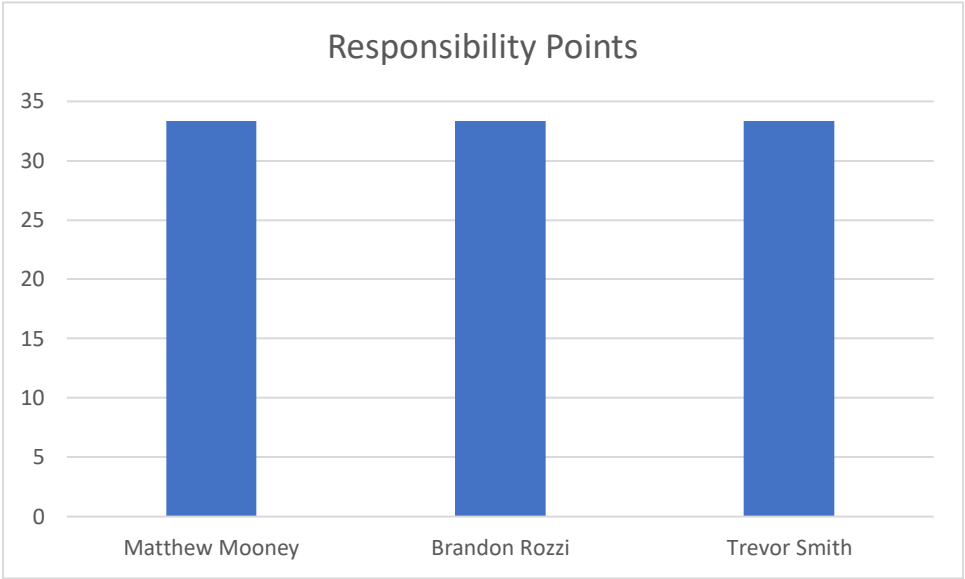
Trevor Smith

Individual Contribution Breakdown

Table 1: Responsibility Matrix

Allocation by Team Member		Team Member Name		
		Matthew Mooney	Brandon Rozzi	Trevor Smith
Responsibility levels	Project Management (8 points)	33%	33%	33%
	Sec.1: Customer Statement of Requirements (8 points)	33%	33%	33%
	Sec.2: System Requirements (8 points)	33%	33%	33%
	Sec.3: Functional Requirements Specification (8 points)	33%	33%	33%
	Sec.4: User Interface Specs (8 points)	33%	33%	33%
	Sec.5: Domain Analysis (8 points)	33%	33%	33%
	Sec.6: Interaction Diagrams (8 points)	33%	33%	33%
	Sec 7: Class Diagrams and Interface Specification (8 points)	33%	33%	33%
	Sec 8: System Architecture and System Design (9 points)	33%	33%	33%
	Sec 9: Algorithms (9 points)	33%	33%	33%
	Sec 10: User Interface Design and Implementation (8 points)	33%	33%	33%
	Sec 11: Design of Tests (9 points)	33%	33%	33%
	Sec 12: History of Work (9 points)	33%	33%	33%

Graph 1: Responsibility Chart



Contents

Individual Contribution Breakdown	2
Contents	4
Customer Statement of Requirements	7
Glossary of Terms.....	10
System Requirements	10
Functional Requirements	11
Nonfunctional Requirements	11
User Interface Layout	13
Functional Requirements Specification.....	13
Stakeholders.....	13
Room Reservation Actors and Goals	14
Tutor System Actors and Goals	14
Use Case Descriptions	15
Room Scheduling.....	15
Approval	15
View Room Analytics.....	17
Obtain Suggested Schedule	18
Traceability Matrix	19
Detailed Use Case Description	19
Use Case: Scheduling a Room.....	19
Stakeholders and Interests:	20
Preconditions:	20
Successful Operation:	20
Exception Cases:	20
Post conditions:	21
Stakeholders and Interests:	21
Preconditions:	21
Successful Operation:	21
Exception Scenarios:	21
Post conditions:	21
Stakeholders and Interests:	22
Preconditions:	22

Successful Operation:	22
Exception Scenarios:	22
Sequence Diagrams	23
Room Scheduling	24
User-Effort Estimation	26
Administrator Approval	26
User-Effort Estimation	27
Viewing Tutors	27
User-Effort Estimation	28
Room Unavailability	29
User-Effort Estimation	30
Suggested Schedule	30
User-Effort Estimation	31
Viewing Analytics	31
User-Effort Estimation	32
Room Scheduling Domain Model	32
Concept Definitions	32
Association Definitions	33
Attribute Definitions	33
Traceability Matrix	33
Operation Contracts	34
Tutor Scheduling System Domain Model	35
Concept Definitions	35
Association Definitions	35
Attribute Definitions	35
Traceability Matrix	35
Operation Contracts	36
Interaction Diagrams	36
Approval	36
Room Scheduling	37
Adding/Editing Tutors	38
Class Diagram and Interface Specification	40
Room Scheduling System	40

Data Types and Operations	40
Traceability Matrix	42
Tutor Scheduling System	43
Data Types and Operations	43
Traceability Matrix	44
System Architecture and System Design	44
Architectural Styles	44
Identifying Subsystems	45
Subsystems to Hardware	47
Persistent Data Storage	47
Network Protocol	47
Global Control Flow	48
Hardware Requirements	48
Algorithms and Data Structures	48
Algorithms	48
User Interface Design and Implementation	48
Design of Tests	49
Project Management	50
Merging Contribution Conflicts	50
Project Coordination and Progress Report	50
Plan of Work	50
History of Work	50
Key Accomplishments	50
Future Plans	51
Breakdown of Responsibilities	52
References	51

Customer Statement of Requirements

The mathematics department requires a system that will allow specific rooms to be reserved as well as a system to aid in tracking Math Assistance Center (MAC) tutors' availability. Presently the department houses three binders for room scheduling. These binders correspond to one to two conference rooms or computer labs. Within the binder is pages upon pages of spreadsheets displaying the available time blocks for the respective room on a day by day basis. When a room reservation is desired, the faculty member must identify the binder, page and time block that corresponds to the reservation they would like to place. The faculty member then reserves the room by filling the time block in with their name, course number, or other identifying information to signify that they have reserved the room. Reservation is restricted to faculty members of the math department only unless a special request is made by a faculty or organization external to the math department.

The proposed solution is one that would automate this process. The system will check and verify that a faculty member or student is currently enrolled at Youngstown State to allow a room reservation. The proposed system will allow for faculty to be able to go online and make reservations from anywhere and at any time. A faculty member is expected to be able to utilize the system to view availability for the conference rooms and computer labs. When they have identified the room and time they desire they should be able to schedule the room immediately. Reoccurring scheduling is an option for cases where the same time block is required for several occasions. The time block is removed from the availability pool once a reservation has been made. Once the Faculty schedules the room, an email notification should be issued as a confirmation that the room was successfully scheduled with the date, time, and room.

The system should allow for users who are not faculty of the mathematics department. Although non-faculty members of the mathematics department can make requests, mathematics

faculty have priority over scheduling rooms and labs. These users can see available rooms and request a reservation. A reservation request should contain a brief reason for the reservation. Once the request has been placed, the department's Administrative assistant must approve the request for the reservation to be scheduled. When a request is pending for a time block the time block is removed from the availability pool. If the request is denied the time block is returned to the availability pool. If the request is approved, the reservation is scheduled, and a notification should be sent to the request as a confirmation that the reservation was scheduled. There are no RSVP requirements stipulated on room reservation. A room reservation can be made up until the time block begins. After a reservation is made a faculty member or student will be able to view their reservation. If a reservation must be changed or canceled the owner of the reservation will be able to delete it. Once a reservation gets deleted the time block should be added back to the available rooms for other people to be able to schedule the room.

There may be such a time at which a room may be closed for maintenance, holiday or other reason. Under such circumstances the administrative assistant can remove time blocks from the availability pool. If a reservation has been previously scheduled for a time in which the administrative assistant needs to close a room, the reservation is lost. The owner of that reservation receives a notification that their reservation has been cancelled with a brief reason for the cancelation.

The room reservation system will also provide basic data analytics. The administrator of the system will be able to view various statistics for several categories. It will be able to collect data on how often a specific room type is requested and scheduled. The system will also collect data on what are the busy days of the week that rooms are scheduled. This analytical system will

provide the department with data that will better help them to adjust room availability and type of rooms that are needed.

The MAC coordinator currently utilizes excel spreadsheets to track the availability of the employees. This spreadsheet contains charts of each day of the week. The chart columns represent a block of time. Rows correspond to a tutor and the cells are populated based on whether the tutor is available at the respective time slot. Another spreadsheet exists to track the classes for which a tutor can provide tutoring. Using these two spreadsheets, the MAC coordinator builds a schedule and enters it into the online scheduling system When Do I Work. This system is used to distribute the schedule to the MAC personnel to know when they are scheduled to tutor in the MAC.

The proposed system will provide analysis of tutor availability to streamline the schedule creation process. At the beginning of the semester tutors submit their availability to the MAC coordinator who will enter the availability into the system along with the capable classes. The coordinator maintains the ability to modify availability in the case that a tutor must change their availability. Once the availability has been entered, the coordinator can view available tutors by day, time, or by classes. This will allow the coordinator to clearly see which tutor is available at a specific time and for specific classes to build a schedule accordingly.

The system will take the information provided by the tutors to help provide a suggested schedule for the MAC. This schedule will insure that every subject offered at the MAC always has a tutor available. A student should be able to go to the MAC at any time and be able to find a tutor for their specific class. With the analytics of the system, if the MAC needs more tutors at a specific time and specific class the coordinator will be able to see what tutors meet the requirements to fill the spot. The system should attempt to create a schedule that spreads the

work hours out among the various tutors. This will prevent any one tutor from getting too many or too few hours. The MAC coordinator will be able look over the proposed schedule that was given by the system. Then the MAC coordinator can then take this schedule into consideration when he or she is making the final schedule for the semester.

Glossary of Terms

TERM	DISCRIPTION
Availability Poll	Show the available room times for a given class room
Database	A collection of stored related data
Data Analytics	Process of examining data sets to draw conclusions about the information they contain
Drop-down menu	A list of items that appear when clicking on a button or text selection.
Reoccurring scheduling	An option for cases where the same time block is required for several occasions
Sequence Diagram	An interaction diagram that shows how processes with one another and in what order
System Admin	Person who is responsible for the upkeep, configuration, and reliable operation of the system
Time Block	Shows the time of a specific date Ex: 10-11, 11-12
Use Case	a set of actions (use cases) that some system or should or Can perform in collaboration with the
User Interface	How the user and a computer system interact

System Requirements

Functional Requirements

REQ-ID	Priority Weight	Description
REQ-1	High	Faculty member or student can view if a room is available or scheduled.
REQ-2	Med	User can schedule a reoccurring reservation of a conference room or computer lab.
REQ-3	High	When a student or non-math department faculty requests a conference room or computer lab it will get sent to the departments Administrative Assistant for approval or disapproval.
REQ-4	High	If a conference room or computer lab is under maintenance, holiday, or other reason and it has been reserved. Then the owner of the reservation will be notified with a description of the reason for cancelation.
REQ-5	Low	A user can view and cancel their existing reservations.
REQ-6	Low	Will give the administrator data analytics for which specific days and rooms get scheduled more often. This will help them to adjust room availability.
REQ-7	High	The administrator can view tutors by their availability and classes they can tutor.
REQ-8	Med	The system will give a suggested schedule to help administration have a place to start when creating the MAC schedule.
REQ-9	High	Approve users to faculty status.

Nonfunctional Requirements

REQ-ID	Priority Weight	Description
--------	-----------------	-------------

REQ-10	High	Sustainability – The system administrator must be able to add additional rooms without modifications to the existing system.
REQ-11	High	Usability – Users should be able to view room availability without signing in.
REQ-12	Low	Usability – The user should be able to reserve a room in as little as three clicks.
REQ-13	High	UX – Room time slots will update in real time with availability.
REQ-14	Med	Scalability – The system should be able to be used for any feasible number of rooms.
REQ-15	High	Reusability – The system should be able to work for any other department that could use it. As well as being able to work every semester.
REQ-16	High	Availability – The system should be available twenty-four hours a day, seven days a week.

User Interface Layout

The diagram illustrates a user interface for scheduling a room. It includes a 'DATE' dropdown menu and a table titled 'Room Times'. The table has columns for time slots (9-10, 10-11, 11-12, ..., 8-9) and rows for rooms (Room 1, Room 2, Room 3, Room 4, Room 5, ...). The time slots are represented by boxes, and the rooms are listed in a column on the left.

Room#	9-10	10-11	11-12	...	8-9
Room 1	9-10	10-11	11-12	...	8-9
Room 2	9-10	10-11	11-12	...	8-9
Room 3	9-10	10-11	11-12	...	8-9
Room 4	9-10	10-11	11-12	...	8-9
Room 5	9-10	10-11	11-12	...	8-9
...					

Figure 1: Scheduling a room

Figure 1 displays an illustration of the planned layout of the user interface for scheduling a room. A user selects a date from a drop-down menu and a list of rooms with their available time blocks for that date are displayed. When a time block is reserved, that time block is hidden from the user's view.

Functional Requirements Specification

Stakeholders

The stakeholders for the system are the students who will use the system, the Math department of YSU who will act as the administrators of the system, and the Computer Science

department of YSU which was tasked with its creation. Trevor, Matt, and Brandon are also stakeholders since they are acting as developers of the system.

Room Reservation Actors and Goals

Actor	Type	Goal	Description
User	Initiating	To request a room reservation.	A general user that must have a request accepted to reserve a room.
PowerUser (Faculty)	Initiating	To reserve a room.	Math department faculty who can reserve a room without permission.
Administrator	Initiating	To accept or deny room requests as well as decide when rooms are unavailable.	User who can directly change room availability as well as handle room reservation requests.
Room Database	Participating		Stores information about rooms and their availability

Tutor System Actors and Goals

Actor	Type	Goal	Description
Administrator	Initiating	View tutor availability and suggested schedule.	The user in charge of the MAC scheduling.
Tutor Database	Participating		Stores tutor information and availability.

Use Case Descriptions

Room Scheduling

A registered user logs into the system and is taken to view availability (REQ-1). The user then has the option to schedule a room reservation (REQ-2) and manage their existing reservations (REQ-5). A reservation can be placed reoccurring on a given time slot (REQ-2). If the user is not a Math Department faculty member, the request will be sent to the administrator for approval (REQ-3). If no approval is required or the request is approved, the room is then scheduled.

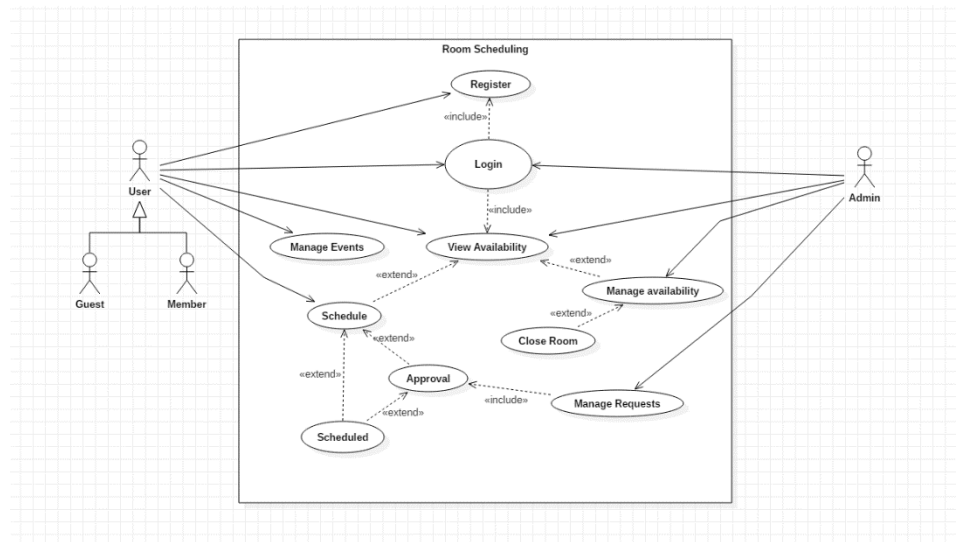


Figure 2: Room Scheduling Use Case Diagram

Approval

The administrator logs into the system and can choose between viewing available room reservation requests and available faculty user access requests. When viewing the pending room requests the administrator will be able to view each requests information and either approve or deny the request (REQ-3). When viewing the faculty user requests the administrator will be given the requester's information and can then either approve or deny them accordingly. (REQ-10).

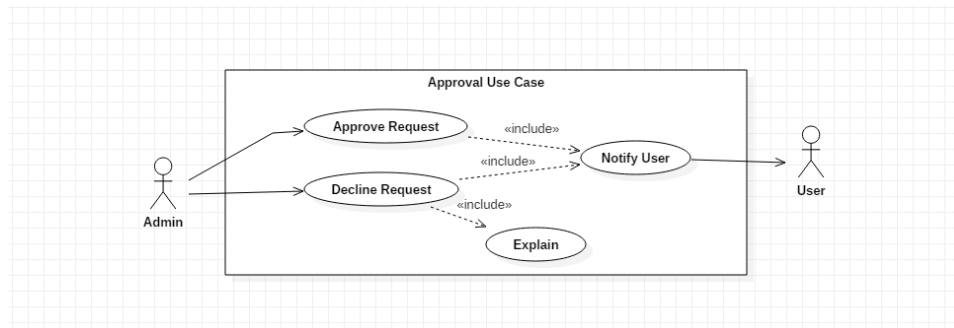


Figure 3: Approval Use Case Diagram

Room Unavailability

When the administrator logs into the system, they will have to option to make a room unavailable for reservation (REQ-4). If the room is already scheduled, but needs to be closed for some reason, the admin will be able to send an email alerting the owner of the reservation of the cancellation.

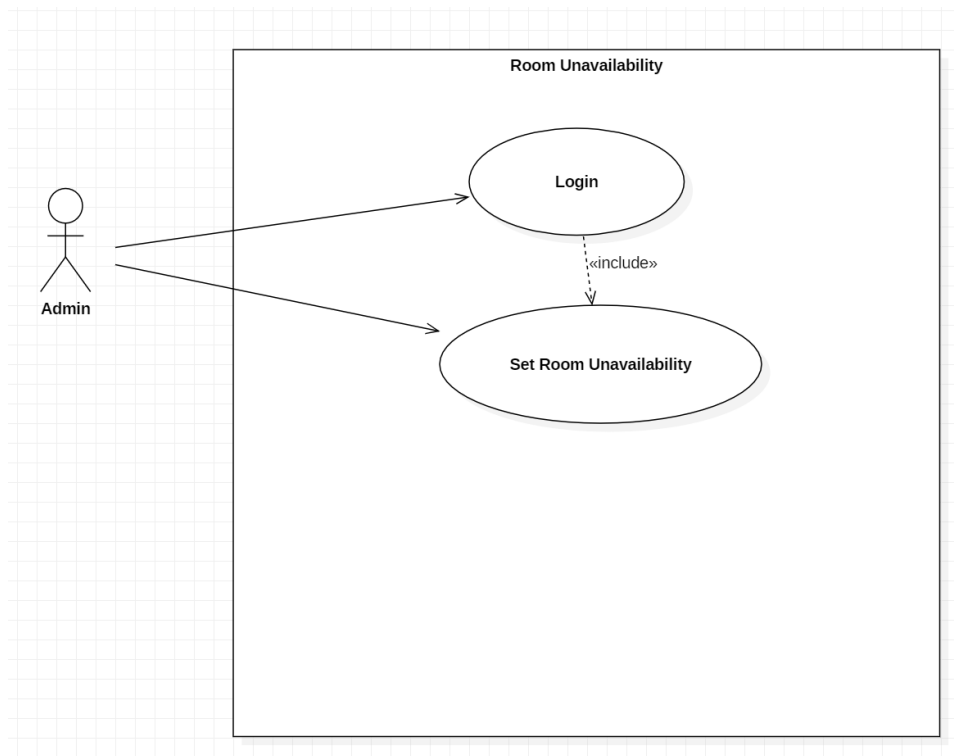


Figure 4: Room Unavailability Use Case Diagram

View Room Analytics

The scheduling system admin logs in to the system. The admin has the option to view analytic data regarding room utilization (REQ-6)

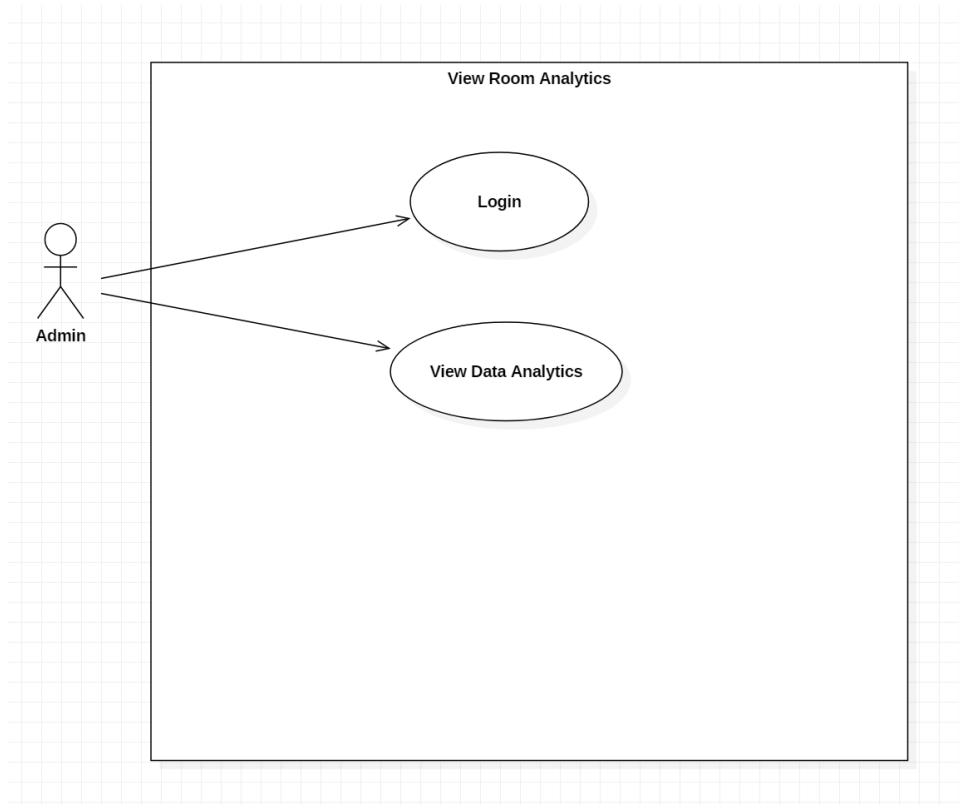


Figure 5: View Room Analytics Use Case Diagram

Updating Tutors

The Administrator will have a page to view a list of the tutors that are on staff at the MAC (REQ-8). The page will have to name of each tutor alongside with the classes they can tutor and their availability. The administrator will be able to click an edit button by each of the tutors to be able to delete or change the availability of a tutor. Administrator will also be able to add a new tutor to the list.

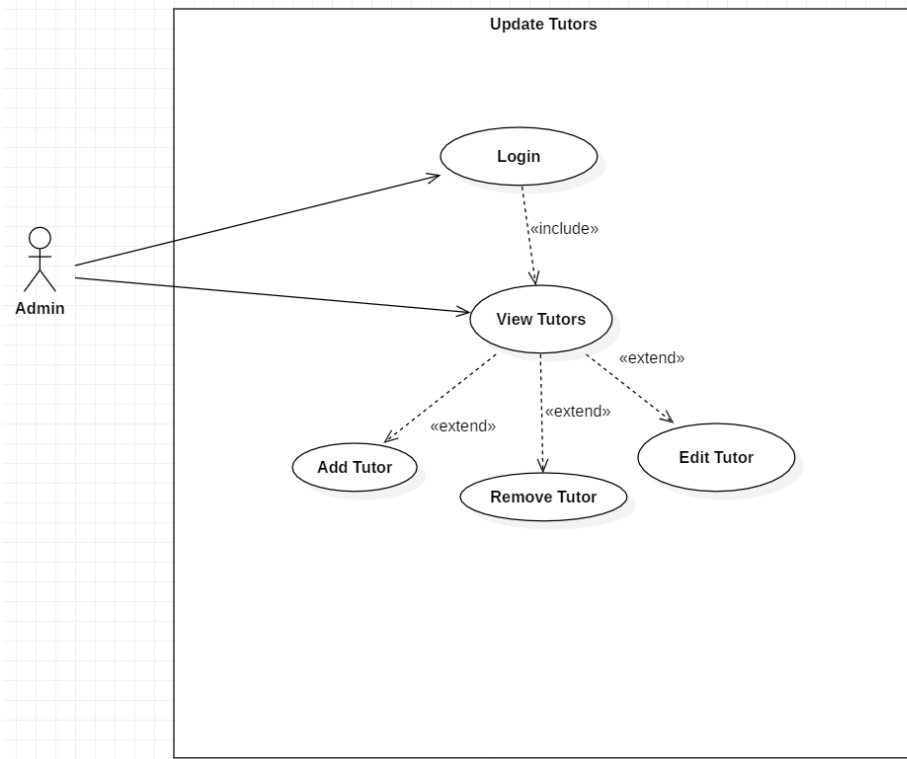


Figure 6: Update Tutors Use Case Diagram

Obtain Suggested Schedule

The MAC admin logs in to the system. The admin then has the option to view a suggested schedule generated by the system (REQ-8).

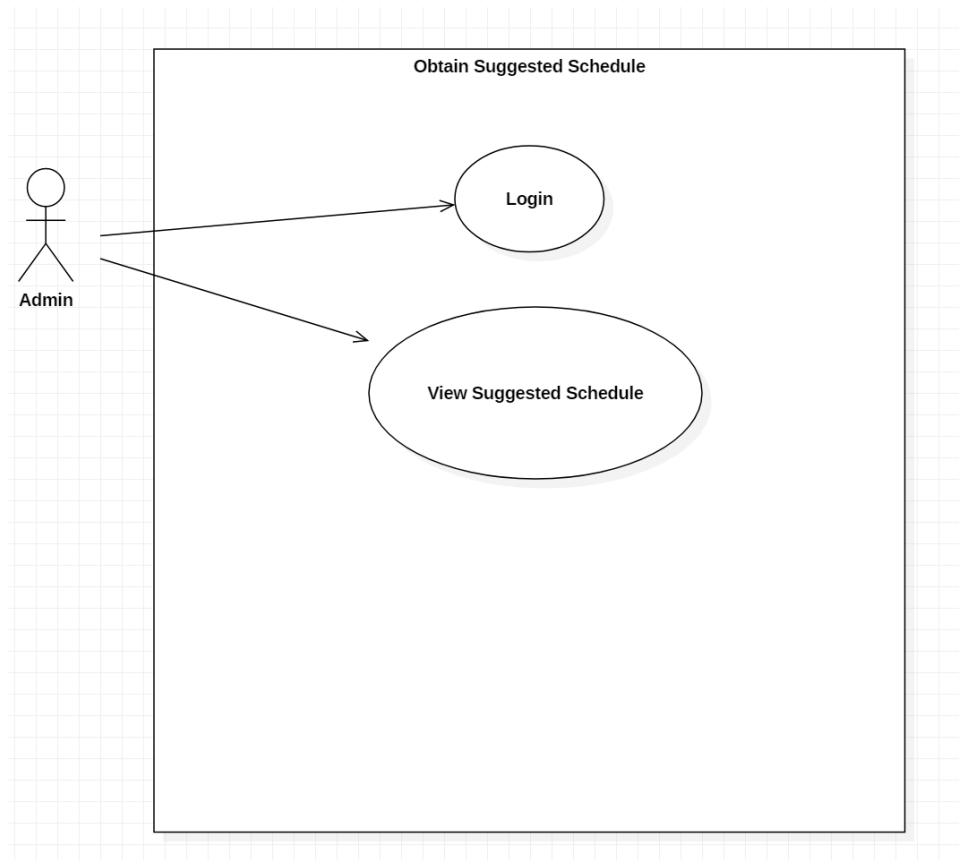


Figure 7: Obtain Suggested Schedule Use Case Diagram

Traceability Matrix

Use Case	Priority	REQs Met
Scheduling a Room	1	1, 2, 3, 5
Approval	2	3, 9
Viewing Tutors	3	7
Room Unavailability	4	4
Obtaining a Suggested Schedule	5	8
Viewing Room Analytics	6	6

Detailed Use Case Description

Use Case: Scheduling a Room

Primary Actors: User

Participating Actors: Admin

Stakeholders and Interests:

- User wants to schedule a room for an event and receive confirmation that the room was scheduled. The process should be convenient and simple.
- The admin wants faculty to be able to schedule a room without approval while non-department faculty must receive approval

Preconditions:

- User successfully authenticates into system

Successful Operation:

1. The user views available rooms and time blocks
2. User selects the time block for the room that they desire
3. User completes a form with information regarding their event. This form details reoccurrence, start and end date of reoccurrence, and a description of the event.
4. The system returns confirmation that the reservation was placed.
5. Non-department faculty continue the use case for approval. The reservation is scheduled and the use case ends.
6. The admin approves the request
7. The user receives confirmation that the reservation has been scheduled and use case ends.

Exception Cases:

- A user completes the reservation form for a room that is already reserved. Reservation is not scheduled.
- The admin declines the reservation request. User receives notification of the decline with a brief reason. The reservation is not scheduled.

Post conditions:

- User has confirmation that their meeting was scheduled

Use Case: Approval

Primary Actors: Admin

Participating Actors: User

Stakeholders and Interests:

- Admin wants to be able to view all pending requests, quickly view details about the requests and easily approve or decline the requests.
- User wants to request a room reservation or department faculty account status and receive notifications regarding whether their requests have been approved or declined

Preconditions:

- Admin successfully authenticates into the scheduling system

Successful Operation:

1. The admin views the pending requests
2. Admin selects a pending request and reviews the request details
3. Admin approves the request
4. User receives a notification that the request was approved and use case ends.

Exception Scenarios:

- Admin declines a pending request. Admin provides a reason for the decline and the user receives a notification containing notice of decline and the reason provided.

Post conditions:

- Room is scheduled, or account rights are increased to faculty status

Use Case: Updating Tutors

Primary Actors: MAC Coordinator

Stakeholders and Interests:

- The MAC Coordinator wants to be able to view tutor availability and easily modify or add tutors to reflect changes in availability

Preconditions:

- The MAC Coordinator successfully authenticates into the tutor availability system

Successful Operation:

1. MAC Coordinator views a list of tutors and their availability
2. MAC Coordinator selects add to add a new tutor or selects a tutor to edit. If the option to delete a tutor is chosen, the use case ends.
3. A form is presented containing the availability of the tutor which the MAC Coordinator populates with the tutor information. When editing a tutor this form is prepopulated with the existing information which can be changed.
4. The MAC Coordinator submits the form
5. The use case ends

Exception Scenarios:

- A tutor is added that already exists, existing tutor is updated.

Sequence Diagrams

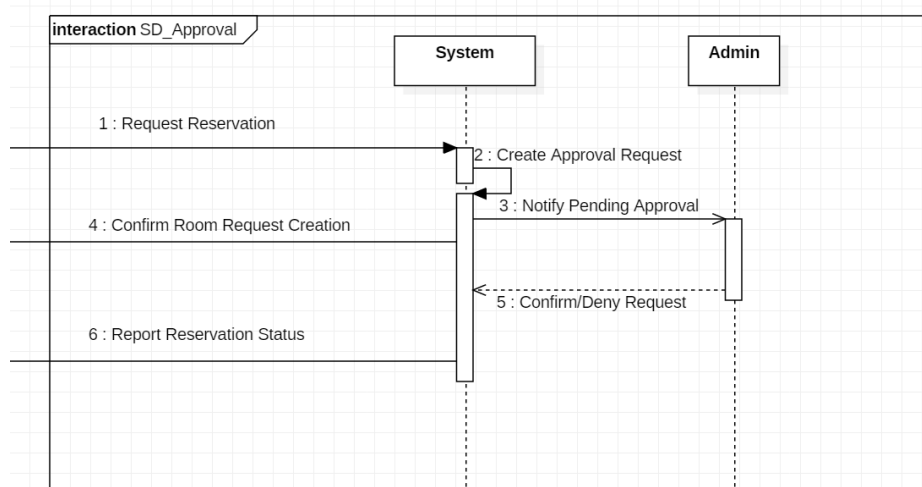


Figure 8: Approval Sequence Diagram

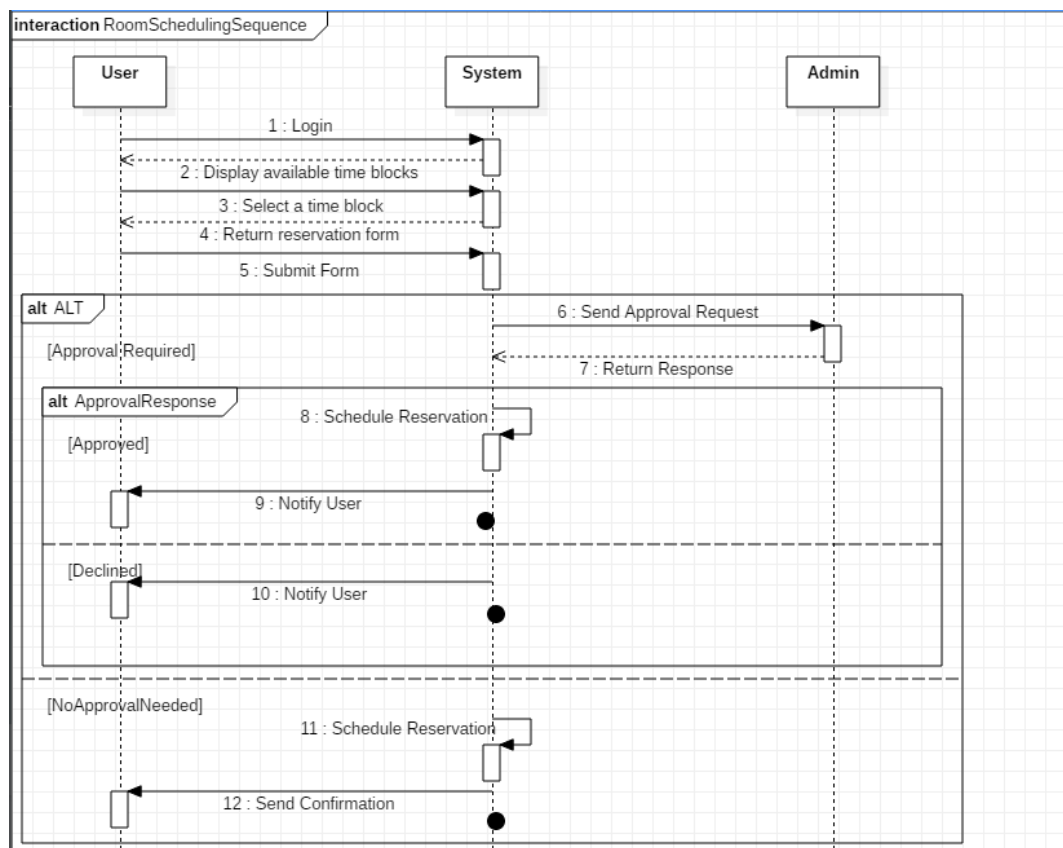


Figure 9: Room Scheduling Sequence

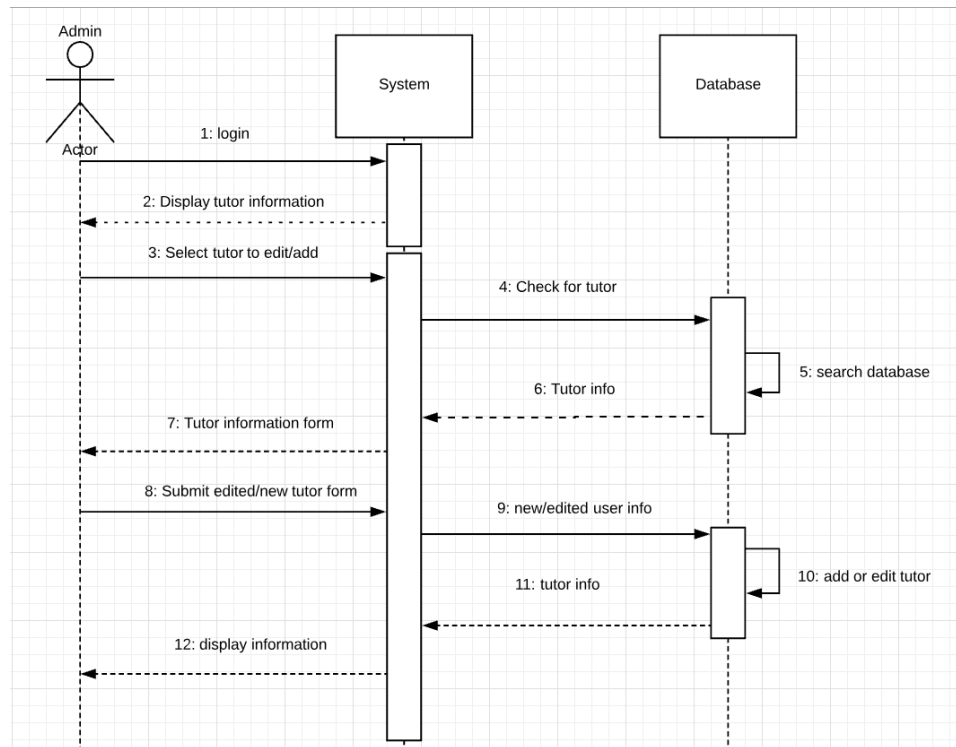


Figure 10: Adding/Editing tutors

User Interface Specification

Room Scheduling

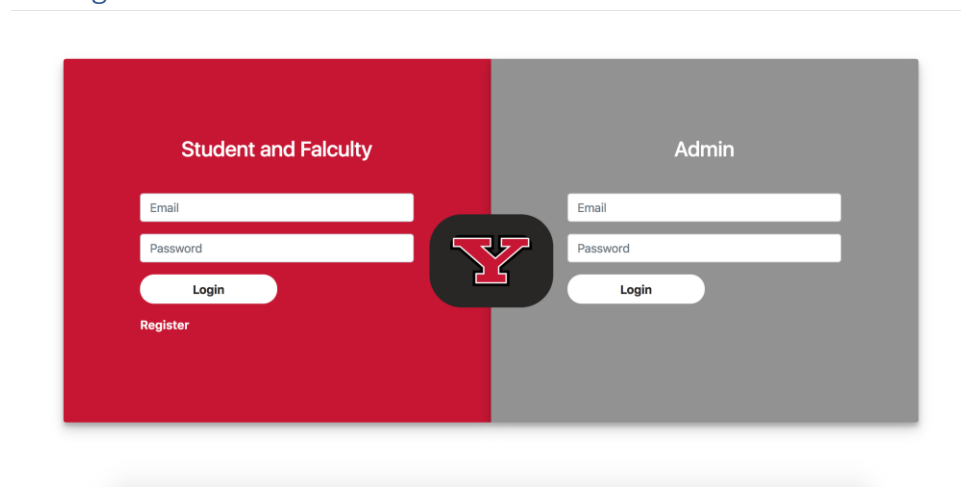


Figure 11: Log in

Figure 12: Choosing a room.

Figure 13: Submitting a room request.

Once a user logs into the system (**Figure 11**) they will be taken to the home page. The home page is where the room availability will be shown as seen in **Figure 12**. The user will then go to the top left of the page below the navigational bar and enter a date to view the room availability. Once this is done, a list of the room numbers and the times available for those rooms will be shown. The user will then click a time slot that they want and hit submit. If the user is faculty of the math department then the reservation will be automatically be accepted and

scheduled. However, if the user is anyone else the request will have to be submitted to an administrator for approval. If this is the case, the user will be taken to the page shown in **Figure 13**. The user will then have to fill out basic information about their room reservation. Which includes if the reservation is recurring and the reason for the reservation. Once these fields are completed the user will hit submit and the request will be sent to an administrator.

User-Effort Estimation

The login page will require three clicks. To select a room the user must select a date, the time slots they want, and hit the submit button which is all together at least three clicks. Then when filling out the reservation information the user will have to select if it is a reoccurring reservation, dates of reoccurring reservation, reason for reservation, and the submit button. This is a total of five clicks. If the user has a non-reoccurring reservation this process can require as few as eight clicks, if it is a reoccurring reservation then it will take a minimum of eleven clicks. The number of keystrokes for this use case of the system is small because only basic user input is required.

Administrator Approval

Name	Date	Time	Room Number	Details
Brandon Rozzi	11/18/2018	11:00-12:00	556	Details
<div> <div>This is the reason</div> <div>If the request is declined please give a reason</div> <div> <input type="button" value="✓ Approve"/> <input type="button" value="✗ Decline"/> </div> </div>				
Trevor Smith	11/20/2018	11:00-12:00	555	Details

Figure 14: administrator approval

Once a request is submitted the administrator will be able to view all the pending room requests shown in **Figure 14**. The requests will show the date of the room reservation and if it is recurring or not. Then below the date information, will be the student's full name and YSU id number. Below that will be the reason the student wants to reserve the room. The administrator will be able to look over all the information and choose whether the reservation can be approved. If the reservation can be approved, then the administrator can click the approved button and the reservation is booked. However, if it cannot be approved then the administrator will type a brief description of the reason why the request got declined. Then hit the disapprove button and the student will be notified by email of the reason for disapproval of the room reservation.

User-Effort Estimation

The admin will have to log into the system which only take about three clicks. Then from the home page must select the pending reservations tab. From there either must click of approve or disapprove. If the reservation is not approved, then the admin must type in why it was not approved. This task can take as little as 5 click to be completed and only a small number of keystrokes.

Viewing Tutors

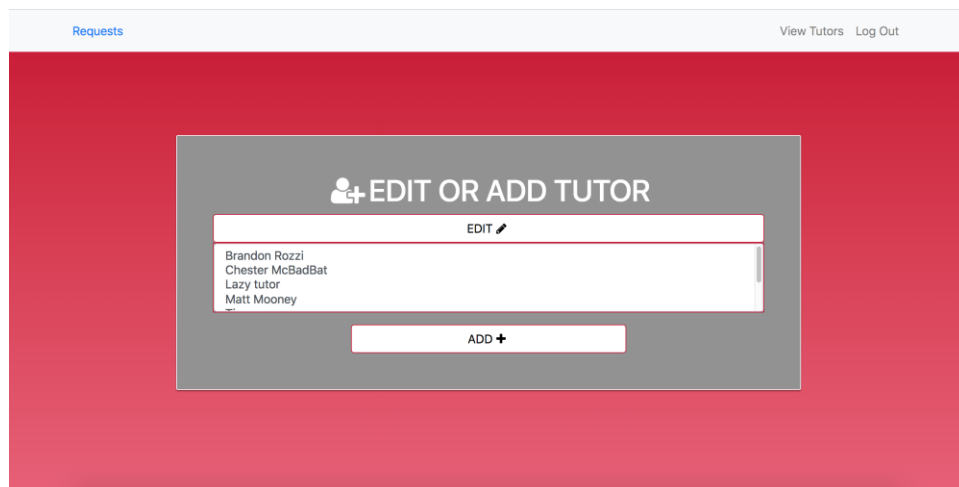


Figure 15: viewing tutors

Figure 16: editing and added a new tutor

When the administrator is logged in, at the top of the page there is a tab for the admin to view the list of tutors on staff at the MAC. This page shown in **Figure 15** will list the current roster of tutors. Along each of the tutor's names is a button for the admin to be able to edit the tutor's information. Also, at the bottom of the page there is an add button that will allow the admin to add a new tutor to the roster. When the admin clicks on the edit button, they are taken to the page shown in **Figure 16**. The admin can then change any of the information such as name of tutor, classes they can tutor, and the tutor's availability. When the admin presses the add button, they are taken to the same page shown in **Figure 16**. Except this time all the fields are empty, and the admin will enter all the tutor's information fields. Once all the fields are filled out then the admin will hit the save and submit button and the tutor will be added to the list.

User-Effort Estimation

The admin will have to click on the *view tutors* tab in the menu bar. Then they must click the add button to add a tutor to the list. From there they must fill in the input fields about the tutor. Depending on how many classes the tutor can teach and their availability the amount of clicks this task takes to complete can be as low as fourteen.

Room Unavailability

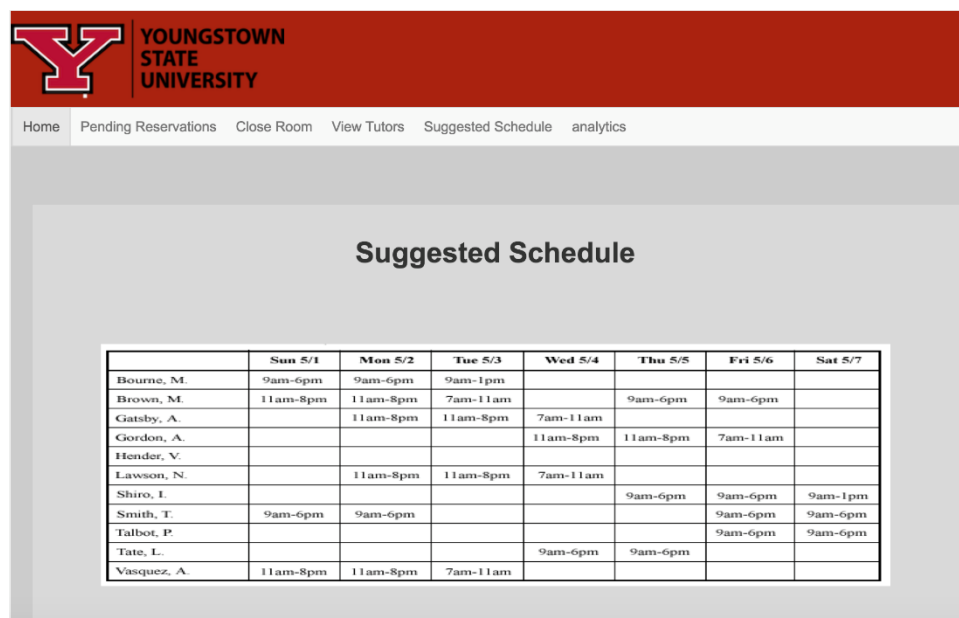
Figure 17: Closing a room

If a room is under maintenance, being used for a class, or the building is closed for a holiday then the admin needs to be able to block off that room from being able to be reserved. To do this the admin will go to the menu bar at the top of the page and click on the *close room* tab. The admin will be taken to the page shown in **Figure 17**. At the top of the page there is a field to put the date to bring up all the times for a given room. The admin can then choose one-time slot to make the room unavailable for only one day, specific hour, or can choose a room and make it closed for an extended period. The admin will pick a room and time and then if the room needs to be closed for more than just one day, they will fill out the field in the bottom left corner. This field allows the admin to enter a start date and end date for a room being closed. Once this is done the admin will fill out a reason for the room being closed and then click submit. Once the admin clicks submit the room becomes closed and is no longer able to be reserved. The system will also send out an email to any owner of a reservation where the room is no longer available to inform them.

User-Effort Estimation

The admin will have to click on the close room tab in the menu bar at the top of the page. Then select a date to view rooms and select the times the room will be closed. They can also choose to close a room for multiple days which means they would have to select the dates for the closure. After selecting the dates, the admin will add a brief description of why the room is closed and click submit this will keep the number of keystrokes to a small number. The number clicks will be different depending on how many time slots need to be selected. Assuming only one-time slot needs to be selected this use case can be done in seven clicks.

Suggested Schedule



	Sun 5/1	Mon 5/2	Tue 5/3	Wed 5/4	Thu 5/5	Fri 5/6	Sat 5/7
Bourne, M.	9am-6pm	9am-6pm	9am-1pm				
Brown, M.	11am-8pm	11am-8pm	7am-11am		9am-6pm	9am-6pm	
Gatsby, A.		11am-8pm	11am-8pm	7am-11am			
Gordon, A.				11am-8pm	11am-8pm	7am-11am	
Hender, V.							
Lawson, N.		11am-8pm	11am-8pm	7am-11am			
Shiro, I.					9am-6pm	9am-6pm	9am-1pm
Smith, T.	9am-6pm	9am-6pm				9am-6pm	9am-6pm
Talbot, P.						9am-6pm	9am-6pm
Tate, L.				9am-6pm	9am-6pm		
Vasquez, A.	11am-8pm	11am-8pm	7am-11am				

Figure 18: suggested schedule

An admin can also view a suggested schedule that the system provides from the information about the tutors. The admin will go to the menu bar at the top of the page and select on the *suggested schedule* tab. This will then take them to the page shown in **Figure 18** where they can view the suggested schedule.

User-Effort Estimation

This use case will take only one click from when the admin is signed in. The user must click on the view schedule tab in the menu bar. For this use case no keystrokes are required or needed.

Viewing Analytics

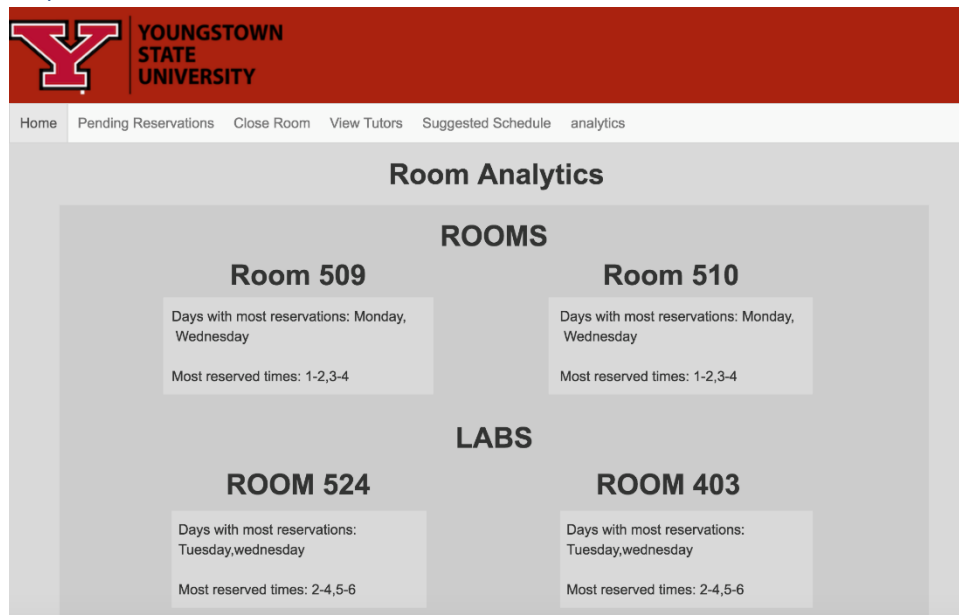


Figure 19: analytics

The admin will be able to go menu bar at the top and select on the *analytics tab*. This will then take the admin to the page shown in **Figure 19**. From this page they will be able to view what days each room gets reserved the most. Also, will be able to see at what times are the busy times for reservations. With the data from this page the admin will be able to see if they need to open additional amount of rooms at a given time.

User-Effort Estimation

This use case will take only one click from when the admin is signed in. The user must click on the analytics tab in the menu bar. For this use case no keystrokes are required or needed.

Room Scheduling Domain Model

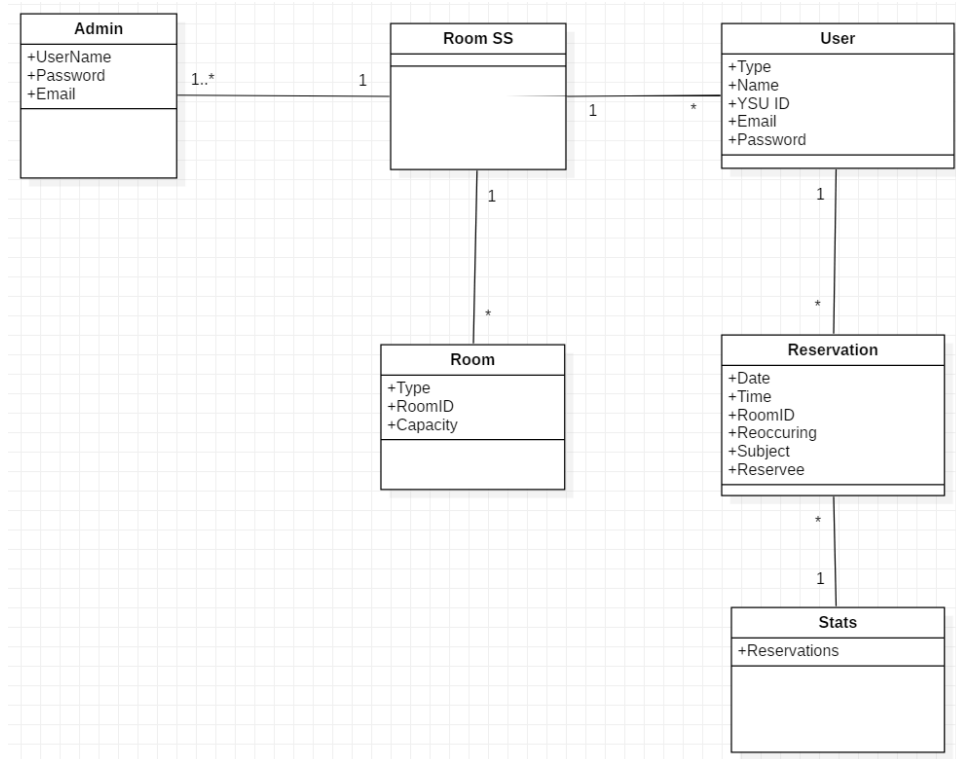


Figure 20: Domain Model (RSS)

Concept Definitions

- **DC1: Room Scheduling System:** This is the internal representation of the system which processes and schedules reservation requests as well as notifying users and sending request to the admin.
- **DC2: Users:** The students, faculty and other external actors which will interact with the system
- **DC3: Admin:** The Math Department Administrator who will manage room availability and request approvals within the system.
- **DC4: Room:** A room represents a physical room within the Math Department which can be scheduled with a reservation.
- **DC5: Reservation:** A scheduled block of time in which a user has reserved a specific room for an event.
- **DC6: Stats:** Statistical representation of the frequency at which reservations are placed.

Association Definitions

- Room Scheduling System to Admin: One room scheduling system has at least one admin
- Room Scheduling System to Rooms: One room scheduling system has many rooms
- Room Scheduling System to Users: One room scheduling system contains many users
- Users to Reservation: One user has many reservations
- Reservation to Stats: One stat analyzes many reservations

Attribute Definitions

- Admin
 - Username: To identify the person trying to access the system
 - Password: To allow access to the admin to enter the system
 - Email: Email address used for to contact the admin
- User
 - Type: To identify if user is a student or faculty outside of the math department
 - Name: Name of the user
 - YSU ID: unique ID for the user in the format of Y00765690
 - Email: Email address used to login in and contact the user
- Room
 - Type: if the room is a conference room or computer lab
 - Room ID: unique id to identity each room
 - Capacity: the number of people a room can hold
- Reservation
 - Date: the date a room is reserved for
 - Time: the specific hour that a room is reserved for
 - Room ID: unique id of the room being reserved
 - Reoccurring: true if the room is reserved for multiple days, false otherwise.
 - Subject: Title for the event
 - Reservee: The user who made the reservation
- Stats
 - Reservations: Reservation objects to get analytics from

Traceability Matrix

Use Case	Priority	Domain Concept
Scheduling a Room	1	1,2,3,4,5,6
Approval	2	3
Viewing Tutors	3	7,8,10
Room Unavailability	4	3

Obtaining a Suggested Schedule	5	7,8,9
Viewing Room Analytics	6	6

Operation Contracts

<p>Operation: SubmitReservationRequest()</p> <p>Cross References: Use Case - Room Scheduling</p> <p>Preconditions: The user is logged in and has filled out the form correctly.</p> <p>Post conditions: The request has been submitted</p>
<p>Operation: SendApprovalRequest(Reservation: Reservation)</p> <p>Cross References: Use Case - Room Scheduling</p> <p>Preconditions: A reservation request has been submitted and requires approval</p> <p>Post conditions: The reservation request has been sent to the Admin for approval</p>
<p>Operation: ScheduleReservation(Reservation: Reservation)</p> <p>Cross References: Use Case - Room Scheduling</p> <p>Precondition: A reservation has been approved or a reservation request submitted that did not require approval</p> <p>Post condition: The reservation is scheduled, and the time block is removed from the availability pool</p>
<p>Operation: NotifyUser(User: User, Reservation: Reservation)</p> <p>Cross References: Use Case – Room Scheduling</p> <p>Preconditions: A request has been accepted and reservation scheduled, or a request has been denied</p> <p>Post conditions: User is notified of the change</p>
<p>Operation: ViewApprovals()</p> <p>Cross References: Use Case: Approval</p> <p>Preconditions: Admin successfully authenticates into the scheduling system.</p> <p>Post conditions: -Admin views list of pending requests for room reservation and user faculty status.</p>
<p>Operation: ApproveRequest(Reservation: Reservation)</p> <p>Cross References: Use Case: Approval</p> <p>Preconditions: Admin successfully views pending requests and selects a request.</p> <p>Post conditions: Request is approved, notifying the user.</p>
<p>Operation: DenyRequest(Reservation: Reservation)</p> <p>Cross References: Use Case: Approval</p> <p>Preconditions: Admin successfully views pending requests and selects a request.</p> <p>Post conditions: Request is denied, notifying the user.</p>
<p>Operation: ApproveRequest(User: User)</p> <p>Cross References: Use Case: Approval</p> <p>Preconditions: Admin successfully views pending requests and selects a request.</p> <p>Post conditions: Request is approved, notifying the user.</p>
<p>Operation: DenyRequest(User: User)</p> <p>Cross References: Use Case: Approval</p> <p>Preconditions: Admin successfully views pending requests and selects a request.</p> <p>Post conditions: Request is denied, notifying the user.</p>

Tutor Scheduling System Domain Model

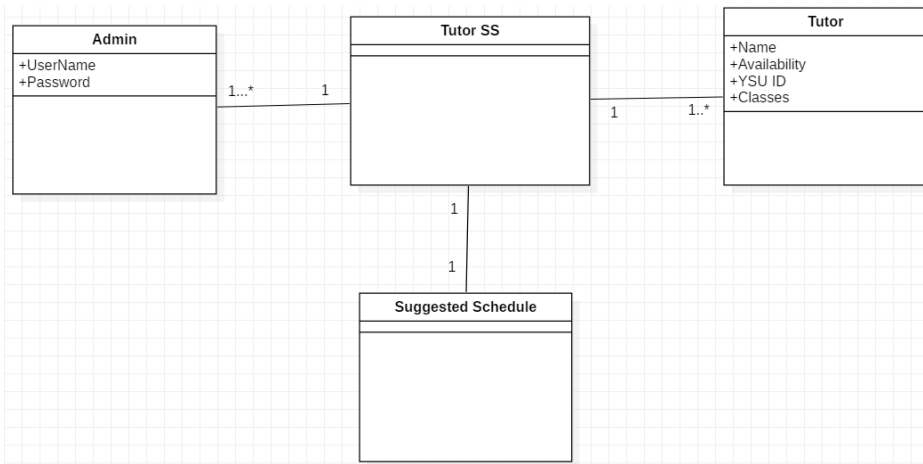


Figure 21: Domain Model (TSS)

Concept Definitions

- DC7: Tutor Scheduling System: This is the internal representation of the system which is responsible for storing tutor information as well as generating the suggested schedule.
- DC8: Admin: The MAC Coordinator who will manage and interact with the Tutor Scheduling System
- DC9: Suggested Schedule: A suggested tutor work schedule built from analyzing tutor availability
- DC10: Tutors: The MAC personnel who provide tutoring to the students

Association Definitions

- Tutor Scheduling System to Admin: One tutor scheduling system has at least one admin
- Tutor Scheduling System to Tutor: One tutor scheduling system has many tutors
- Tutor scheduling system to Suggested Schedule: One tutor scheduling system has many suggested schedule

Attribute Definitions

- Admin
 - Username: To identify the person trying to access the system
 - Password: To allow access to the admin to enter the system
- Tutors
 - Name: Name of a specific tutor
 - Availability: Hours they can work each week
 - YSU ID: unique ID for the user in the format of Y00765690
 - Classes: class subjects that a tutor can teach

Traceability Matrix

Use Case	Priority	Domain Concept
Scheduling a Room	1	N/A

Approval	2	N/A
Viewing Tutors	3	7,8,10
Room Unavailability	4	N/A
Obtaining a Suggested Schedule	5	7,8,9
Viewing Room Analytics	6	N/A

Operation Contracts

Operation: SelectTutor(tutor) Cross References: Updating Tutors Preconditions: Admin has logged in and is viewing list of tutors Post conditions: - A new/updated information about tutor is entered
Operation: checkTutor(tutor) Cross References: Updating Tutors Preconditions: A database with the tutors exists Post conditions: Tutor information is sent back to admin
Operation: submitTutorInfo() Cross References: Updating Tutors Preconditions: Admin has the tutor information form Post conditions: The new tutor information has been submitted
Operation: updateTutor(tutorInfo) Cross Reference: Updating Tutors Preconditions: - A database with tutor information exists - Edited tutor information is submitted Postconditions: Tutor information is updated in the database

Interaction Diagrams

We used a Publisher-Subscriber design pattern for our overall system design. This made sense due to our system revolving around altering and receiving data from a database. Each interaction involves creating requests that are sent to the Cloud and handled by a Lambda function.

Approval

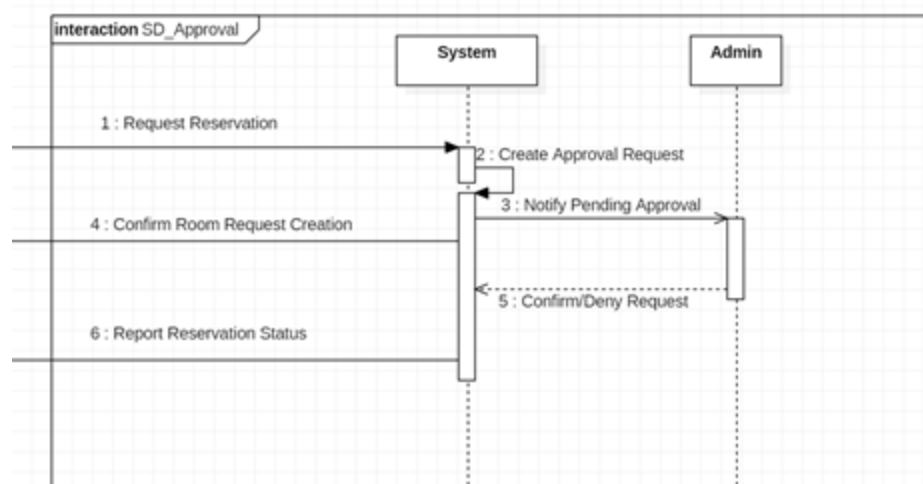


Figure 22: Approval Sequence Diagram

The Approval Use Case has two main contributors, and one outside actor who starts the flow of events. The design principle of the System actor is the Expert Doer Principle. The System actor must be notified of an outside user wanting to reserve something, create the request, and then notify the Admin of the pending request. Since the System is tasked with both communication and computation responsibilities, using Expert Doer Principle for the System makes sense. The Admin's design principle is High Cohesion since the Admin shouldn't need to perform any computations. The Admin should only need to communicate with the system that the request has been approved or denied.

Room Scheduling

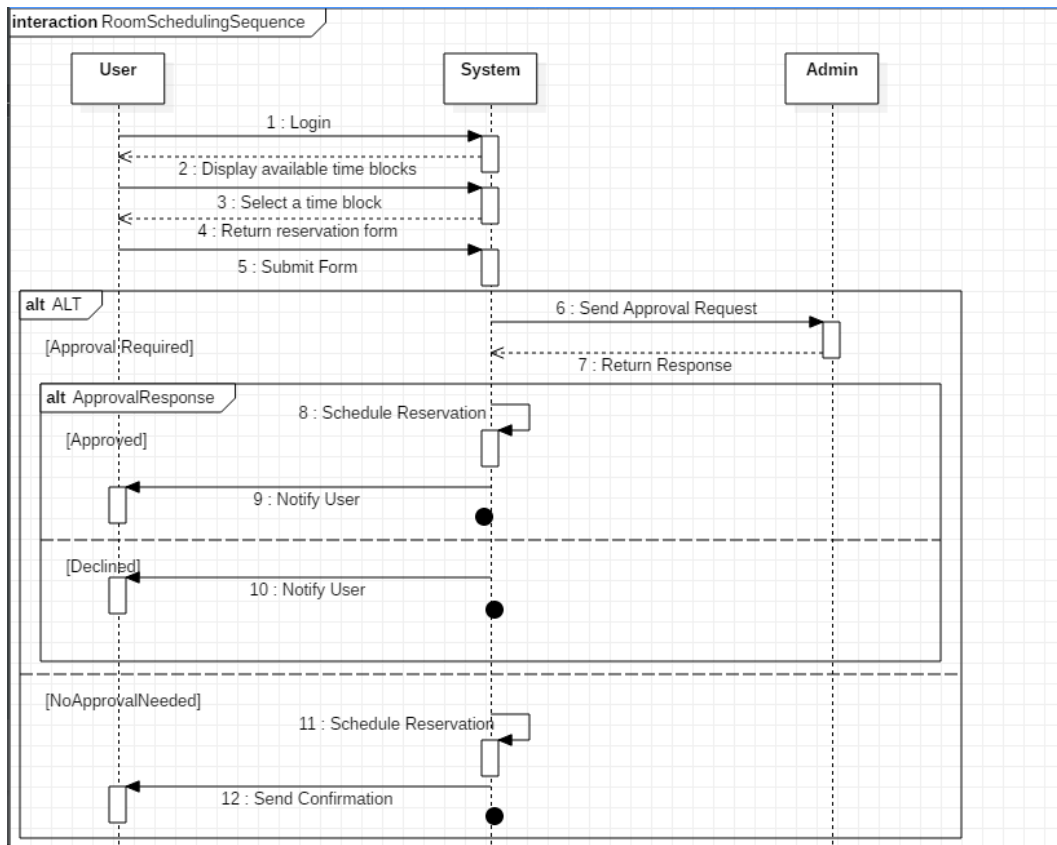


Figure 23: Room Scheduling Sequence Diagram

The room scheduling use case has two main contributors and an outside actor who starts the flow of events. The System actor must be notified of an outside user wanting to reserve a room, create the request, send the request, schedule the event and notify the end user. Since the System is tasked with both communication and computation responsibilities, using Expert Doer Principle for the System makes sense. The Admin's design principle is High Cohesion since the Admin shouldn't need to perform any computations. The Admin should only need to communicate with the system that the request has been approved or denied.

[Adding/Editing Tutors](#)

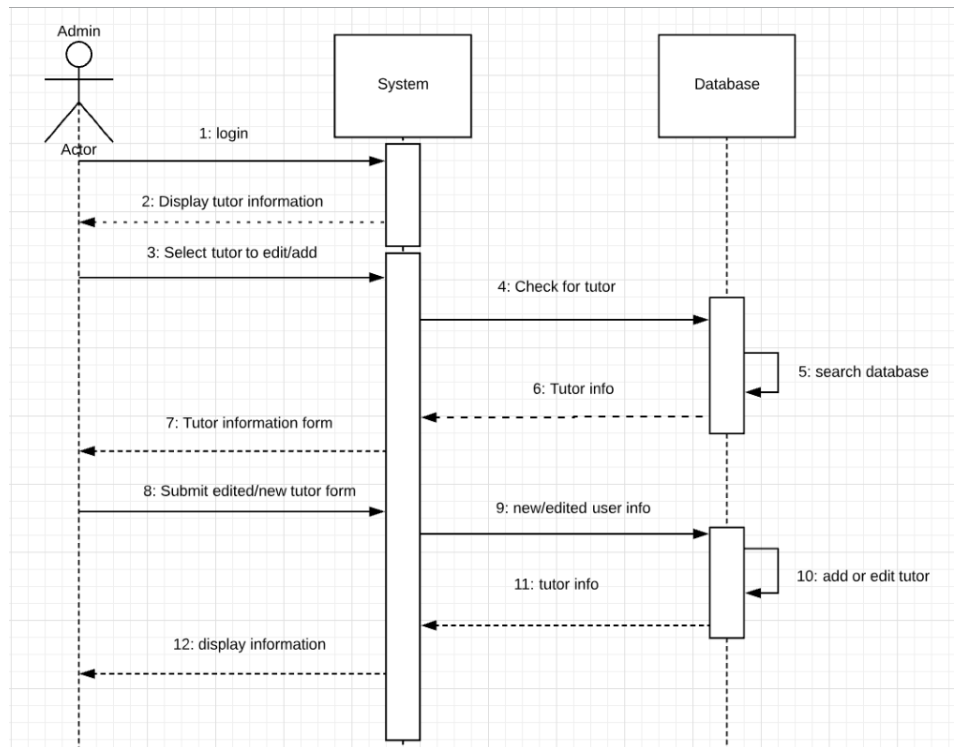


Figure 24: Adding/Editing Tutors Sequence Diagram

The Adding/Editing Tutors Use Case has one main contributor who starts the flow of events. The admin actor must login and select a tutor to edit/add, give information about the tutor, and submit the form. The design principle of the System actor is the Expert Doer Principle. Since the system actor does not need to do any computations then the admin's design principle is of high cohesion. The admin only needs to communicate with the system to add or edit a tutor.

Class Diagram and Interface Specification

Room Scheduling System

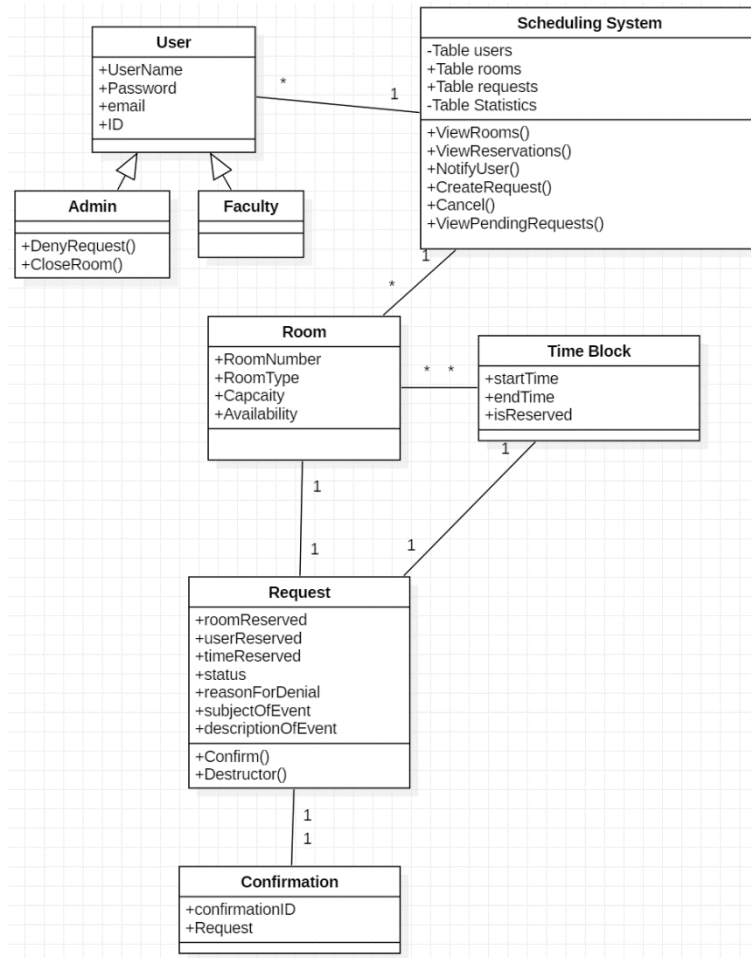


Figure 25: Room Scheduling System Class Diagram

Data Types and Operations

Scheduling System

- Table users;
- Table rooms;
- Table requests;
- Table Statistics;
- Room[] ViewRooms() - Returns an array of all rooms.
- Request[] ViewReservations() - returns an array of all requests made by the user.
- Void NotifyUser(Confirmation) - notifies user of their request being approved/denied and why.
- Void CreateRequest(user, room, TimeBlock) - Creates request.
- Void Cancel(Request) - cancels the request.
- Void ViewPendingRequests() - shows the admin all requests that are pending.

User

- Private String UserName - The user's login identification
- Private String Password - The user's login password
- Private String email - the user's email address
- Private String ID – the users ID, typically their Y number.

Admin (extend User)

- Void ApproveRequest(Request) - Approves the specified requests and notifies the user
- Void DenyRequest(Request) - Denies the selected request, informing the user it was denied and for what reason.
- Void AdminReserveRoom(Room, DATETIME, DATETIME) - Reserves the specified room from the two inputted times. (reserves the room so that it is "closed" to others)

Faculty (extend User)

-

Room

- int RoomNumber - The number of the room, first digit details floor it is on.
- String RoomType – The type of room, whether it is a lab or a conference room
- Int Capacity – total number of people that the room can hold.
- TimeBlock[] Availability – An array of the rooms availability, time slots are removed when the room is in use.

Request

- Room roomReserved – The room that the Request is for.
- User userReserved - The user who requested the room.
- TimeBlock timeReserved - The TimeBlock that the request is for.
- Reoccurring – integer 0-6 for what day of the week it is reoccurring, negative value means not reoccurring.
- Int status - The current status of the request. 0 = pending, -1 = denied, 1 = approved.
- String reasonForDenial - The reason the admin gave for denying the room.
- String subjectOfEvent - The name of the event the reservation is for.
- String descriptionOfEvent - A detailed description of the event.

Confirmation

- Unsigned int confirmationId - The identification of the confirmation
- Request* Request - The request linked to the confirmation.

Time Block

- DATETIME startTime – Start time of the reservation of the room.
- DATETIME endTime - End time of the reservation of the room.

Statistic

- Room analyzedRoom – The room being analyzed by the statistic
- Date DayofWeek – The day being analyzed by the stat
- TimeBlock highestUtil – The timeblock with the highest level of utilization
- TimeBlock lowestUtil – The timeblock with the lowest level of utilization
- Map<TimeBlock, float> avgVisitsHourly – A map of average visits during a given timeblock

Traceability Matrix

Domain Concept	Classes	Explanation
User	User	A user represents the concepts of the person who will be utilizing the service.
User	Faculty	An extended user who is automatically approved for room requests.
Admin	Admin	Admin is an extension of a user who has management abilities.
Room	Room	A room is the representation of the concept of a physical room that will be reserved.
Reservation	Request	A request represents a request for a reservation, a request contains all details of a reservation. An approved request represents a scheduled reservation.
Reservation	Confirmation	When a request is created, approved, declined, or cancelled, an associated confirmation must be generated to notify a user.
Room SS	Scheduling System	The room scheduling system represents the system which will be responsible for completing operations and communicating with the user.
Stats	Statistic	A statistic contains analytical data that the admin can use to understand room utilization.

Tutor Scheduling System

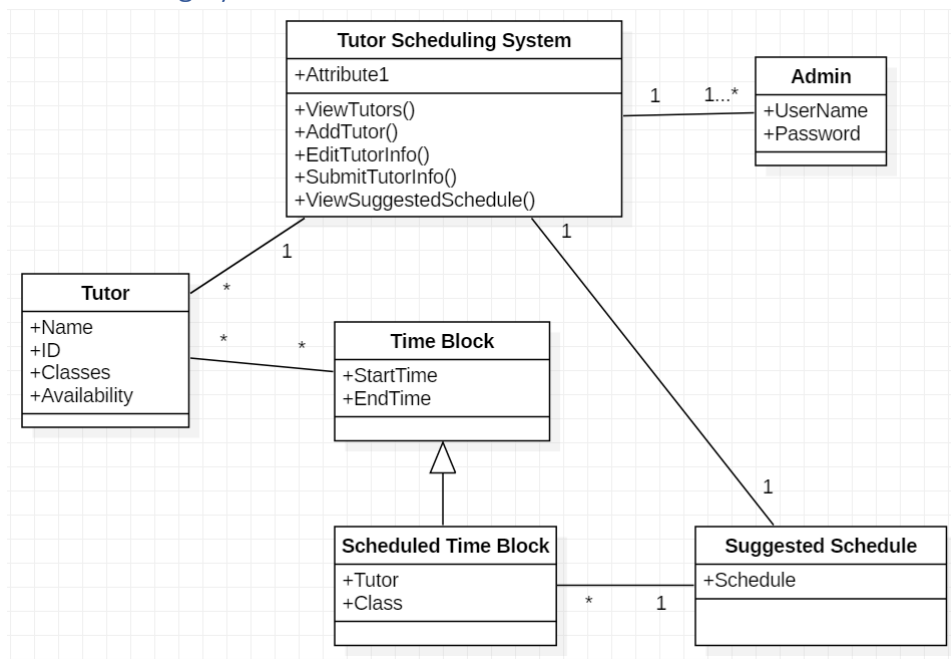


Figure 26: Tutor Scheduling System Class Diagram

Data Types and Operations

Tutor Scheduling System

- Table Tutors
- `tutor[] ViewTutors()` - return a list of tutors
- `form AddTutor()` - add a tutor
- `form EditTutorInfo(tutor)` - edit information about a tutor
- `void SubmitTutorInfo()` – submits a tutor form and adds the info to the tutor table
- `SuggestedSchedule ViewSuggestedSchedule()` – Returns a suggested schedule for the admin to view

Tutor

- String name – Tutors full name.
- String ID – Tutors YSU ID.
- String Classes – The specific classes a tutor can teach.
- `TimeBlock[] availability` - An array of the hours a tutor can work.

Admin

- String Username - Admins login id.
- String Password - Admins password to log in.

Time Block

- `DATETIME startTime` - The time a tutor can start work.

- DATETIME endTime - The time a tutor has to leave.

Scheduled Time Block

- Tutor Tutor
- String Class

Suggested Schedule

- ScheduleTimeBlock[] Schedule

Traceability Matrix

Domain Concept	Class	Explanation
Admin	Admin	The admin class represents the administrator who will interact with the system.
Tutor SS	Tutor Scheduling System	This represents the system which the admin will complete the operations and communicate to the admin.
Tutor	Tutor	Represents a tutor who works at the MAC.
Tutor	Time Block	A tutor has a time block that represents a period in which they are available.
Suggested Schedule	SuggestedSchedule	A suggested schedule is a data type that the tutor scheduling system will generate for the admin to use in scheduling tutors.
Suggested Schedule	ScheduledTimeBlock	A scheduledtimbeblock is an extended timeBlock that is used to represent a period in which the tutor is scheduled to work.

System Architecture and System Design

Architectural Styles

Client-Server:

- Service must be accessible from anywhere.
- Some level of security required.
- Server needed for any and all decision making.
- Allows for user to pull information at runtime so the information shown is always up to date.
- Can easily change design to an n-tiered approach if necessary.
- Client-Server architecture is easily scalable.

Identifying Subsystems

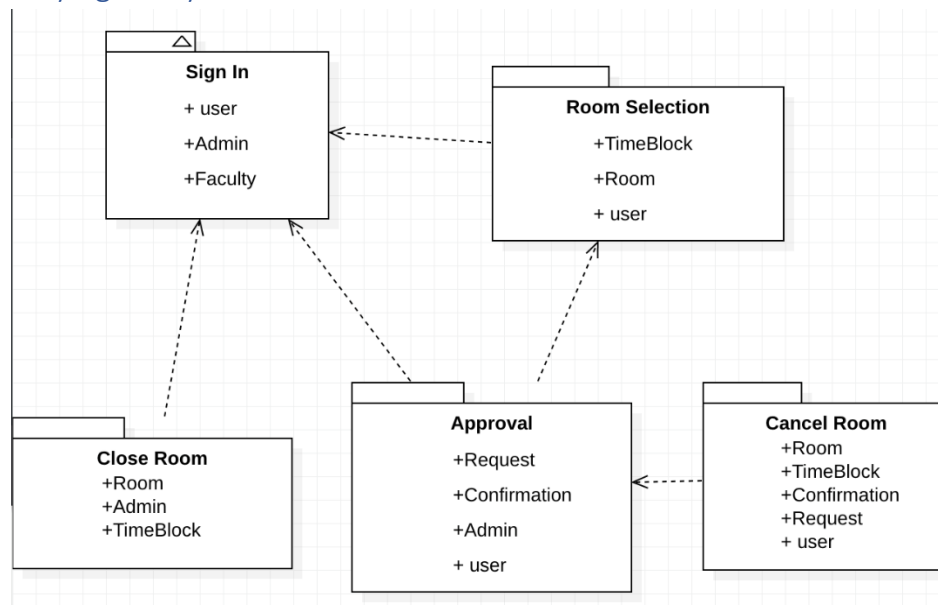


Figure 27: Room Scheduling Package Diagram

Sign In Package

- Subsystem to authenticate type of user.
- User Class - Public
- Admin Class – Public
- TimeBlock Class – Public

Room Selection Package

- Subsystem for a user or faculty member to select a room to reserve.
- Room Selection Package is dependent on the sign in package on whether a user or faculty member has signed in.
- User Class - Public
- TimeBlock Class - Public
- Room – Public

Approval Package

- Subsystem to get the admin approval and confirm a room reservation.
- Dependent on the sign in system. Only admin has access.
- Dependent on the Room Selection Package. Approval Needs a user, room, and timeblock for the Room Selection Package.
- Request class - Public
- Confirmation Class – Public
- Admin Class – Public
- User Class – Public

Cancel Room Package

- Subsystem for a user or faculty member to cancel a room reservation.
- Cancel Room Package is dependent on Approval. An accepted approval needs to happen before a user can cancel a reservation.
- Room – Public
- TimeBlock – Public
- Confirmation – Public
- Request – Public
- User- Public

Close Room Package

- Subsystem for the admin to close a room so it cannot be reserved
- Close Room is depending on whether an Admin, user, or faculty member signs into the system.
- Room Class – Public
- Admin Class- Public
- TimeBlock Class – Public

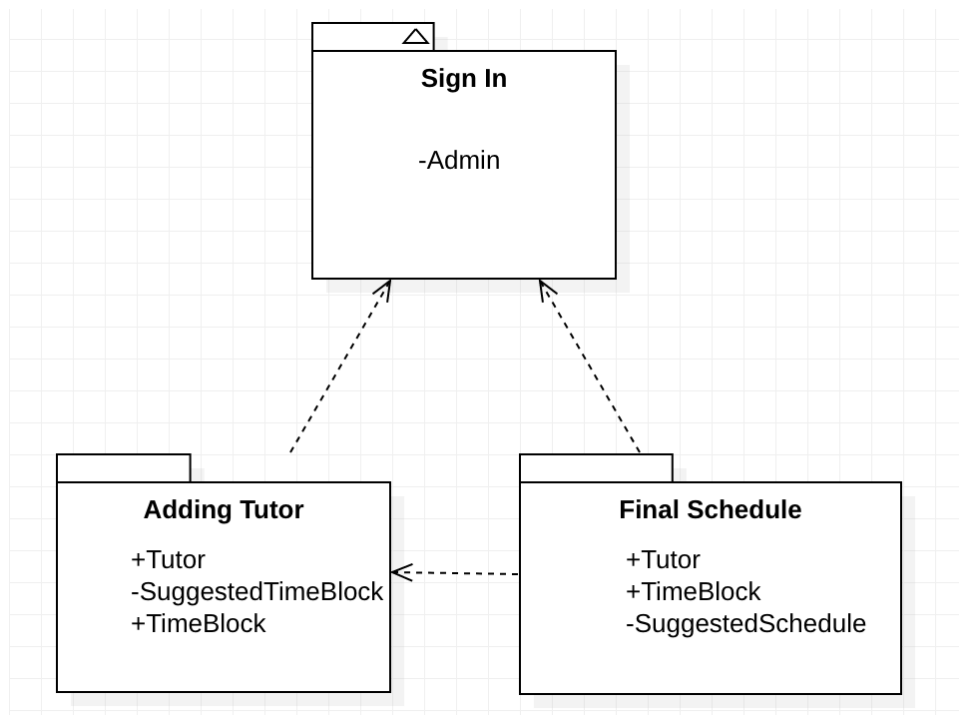


Figure 28: Suggested Schedule Package Diagram

Sign In Package

- Subsystem to authenticate the admin's credentials.
- Admin – Private

Adding Tutor

- Subsystem for the admin to add a tutor for the suggested schedule
- Dependent on the sign in Package for an admin to log in.
- Tutor Class – Public
- SegestedTimeBlock Class – Private
- TimeBlock – Public

Final Schedule

- Subsystem that gives the admin a suggested work schedule for the tutors.
- Dependent on sign in Package for an admin to log in.
- Dependent on Adding Tutor Package. Tutor Need to be added for a suggested schedule to be made
- Tutor Class – Public
- TimeBlock Class – Public
- SuggestedSchedule Class - Public

Subsystems to Hardware

Client Machine:

- Client Browser will pull html files containing dynamic content
- Client Browser will execute JavaScript to send an http request to the API and acquire the dynamic content
- Client Browser will display the content

Server:

- AWS S3 will store the html and JavaScript which will be pulled by the user's browser
- RESTful API (AWS API Gateway) will catch http requests and trigger server functions (Lambda).
- AWS Lambda will communicate with the database, complete operations and return dynamic content
- Database (DynamoDB) will store data to which will be pulled and edited by Lambda

Persistent Data Storage

AWS S3:

- Will be responsible for storing flat html and JavaScript files

DynamoDB:

NoSql database which will store class data

Network Protocol

HTTP:

- User's Browser will use the HTTP protocol to request webpages from S3
- HTTP requests will be sent to the API to request dynamic content and or perform operations.

Global Control Flow

- The system is event driven. Events occur when a webpage is loaded and when a user performs an operation. These events result in API calls which executes a Lambda function.
- The Room Scheduling System will require real time updates. This will operate in a periodic fashion, updating multiple times a minute as well as on event triggers.
- The system will operate on multiple threads. If two users perform the same operation at the same time, multiple independent Lambda functions are initiated. This allows for infinite scaling. The threads are independent of each other, with no synchronization.

Hardware Requirements

- Client internet connection with a minimum bandwidth of 512 Kbps.
- Windows XP SP 2 or higher
- 128MB Minimum of RAM
- Pentium 4 Processor or higher

Algorithms and Data Structures

Algorithms

The tutor scheduling system requires an algorithm to generate the suggested schedule. The algorithm assumes that the MAC is open Monday-Friday and 8-8 each day. The algorithm will attempt to schedule a tutor to cover each class for every hour the MAC is open. It accomplishes this by finding all the tutors who are available to teach a given class on a given day. It will then loop through that subset of tutors and schedule them for the class at the first time they are available for the day if possible. If that time is already filled it will look for the tutors next available time. Once a tutor has been scheduled for one class at a given time, the algorithm will also examine the other classes that the tutor can tutor and attempt to schedule them to cover additional classes at the same time. The algorithm will balance the number of hours a tutor tutors a class by signaling the loop to exit once one tutor runs out of availability. The loop will exit once the class is fully scheduled for the day or the tutors that can teach the class have no more availability.

User Interface Design and Implementation

As both the room scheduling system and tutor scheduling system are web-based the team decided to create the user interface using Bootstrap. As Bootstrap also contains libraries for Javascript, it works well with our backend systems which also run on Javascript. Our initial mockups are functionally similar to the end result, with only a few minor differences made to make the website even easier to use.

Ease-of-use was a key concept at the forefront of the User Interface design. Our goal was to allow the user to request a room in as few clicks as possible. Currently, once a user has logged in, they can request a room in as little as three clicks. This number changes based on the users need, such as changing the date of the request or requesting multiple reservations at a time, but the overall ease-of-use remains at an ideal level.

The tutor scheduling system was also designed with ease-of-use in mind. The team wanted to ensure that the administrator for the system would be able to figure out the system quickly and then use it efficiently. This was accomplished by using forms to add or edit tutors. The form has labeled fields for all the information the system needs about an individual tutor and simply needs to be filled out. When the administrator needs to edit a tutor, the fields of the form are automatically filled with the information that was previously entered so the administrator only needs to input the changes needed.

Design of Tests

The functionality of the room scheduling system mostly depends on inserting, retrieving, and editing various tables in a database. As such, the tests for each individual component will be similar with the only differences being what data is being affected in the database. These tests can be broken down into three different categories: Inserting, Retrieving, and Editing. However, retrieving data will be done in both the inserting and retrieving tests for each type of data so it will not require a test.

- Inserting Data Test
 - The test will insert a test record into the system for each type of data (user information, room requests, etc.)
 - It will then check to see if the test data is available inside the database by checking the relevant table for the information. If the test finds a row inside the table that has all the test data, then the test has passed.
 - After the test has passed, the test information will be deleted from the database.
- Editing Data Test
 - This test will retrieve the last record in the database and will save the information.
 - The test will then edit the information in the last record to be the desired test information.
 - To check whether the test has passed or failed, the data of the last record will be retrieved again and checked against the test data. If they are the same, the test has passed.
 - Finally, once the test has passed or failed, the data in the last table will be restored to its original value so that it does not affect the actual database.

The functionality of the tutor scheduling system also depends on the tests above, as such it will use similar methods of testing. On top of the above tests, it also needs a test for the creation of the suggested schedule to test the algorithm.

- Suggested Schedule Creation
 - This test will be done on a small scale, as it is near impossible to accurately predict the sophisticated algorithm on a large scale.
 - The test will be run on a small sample of tutors and then compared with a manually entered schedule to see if they are equivalent.

While the tests described above ensure that our database is working correctly, the functionality of the website itself also needs to be tested. The team plans to do real user tests of the system as well to ensure that it is working as expected. Between the automated testing of the database and the manual testing of the webpages, the system should have a high-degree of test coverage.

Project Management

Merging Contribution Conflicts

- The team worked in an online document simultaneously while discussing design elements.
- Each team member worked together on the class diagrams and package diagrams for both the scheduling system and suggested tutoring schedule, so all team members will be on the same page when implementing the system begins.

Project Coordination and Progress Report

- Implementation has not begun.
- The team plans to begin implementation in the next few days to keep on schedule.

Plan of Work

We are currently building our prototype which we will use to pivot to our first demo. The demo is our top priority now that we have most of the design for the project completed. While we are not on track for the prototype's original deadline, we do not believe it should affect our demo's deadline since the majority of the prototype should be able to be used for demo purposes. We are currently on track with our reports and should have no issues getting each of them completed by their due dates.

History of Work

Our history of work has evolved just as our plan of work had planned for. We did not miss a class deadline for any report or demo that was due. We spent a total of three weeks planning and designing the system. During this time, we were able to complete report one and two, and start working on our prototype for the first demo. We were able to figure out the web hosting we were going to use and were able to set up all of databases by the deadline. We were not able to come up with a prototype by our original timeline of the 31st of October. The demo was finished on the 10th of November and we were ready to present the working demo for the class. After presenting the first demo we began work on the remaining features of the system. Our largest obstacle was the suggested schedule algorithm we had to design. Once that was done, we focused on finishing up the Lambda functions so the front-end could be completed in time for demo 2.

Key Accomplishments

- Successfully designed a system from start to finish.
- Successfully set up Databases for the system.
- Successfully deployed an application on cloud-based services.
- Developed a working prototype of the system for the demos.
- Development of an algorithm for the suggested tutor schedule.

Future Plans

Our original plan for the system included in-depth analytics for the room reservation system. While the team hopes to get simple analytics done in time for demo 2, it is unlikely that we can complete a more in-depth analytical system. The system could also be expanded to work with more rooms or even additional buildings. To make the system more secure the API should also be updated to implement the Cognito User Pool Authorizer to prevent unauthorized access.

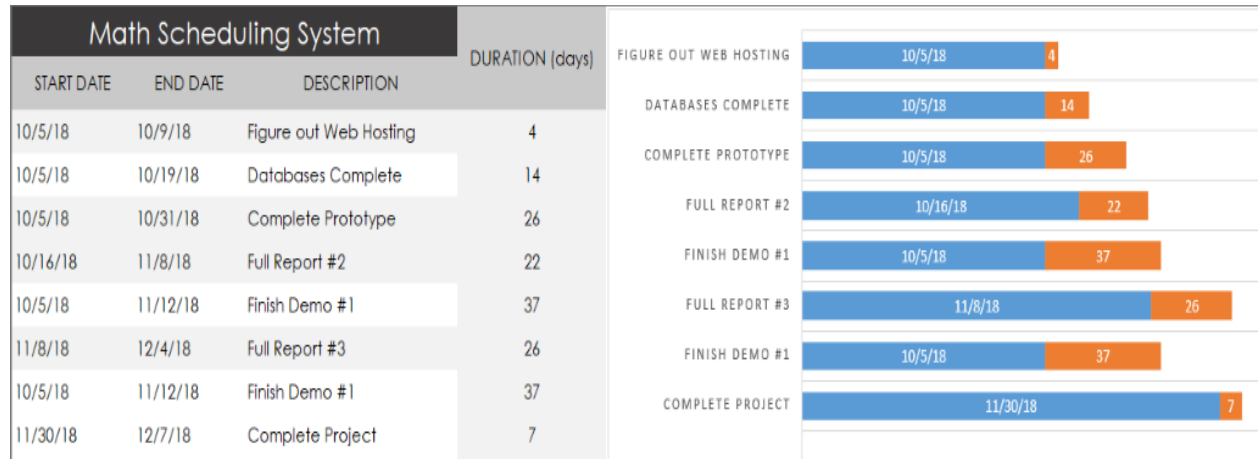


Figure 29: Gantt Chart for Scheduling System

References

AWS Lambda Developer Guide: <https://docs.aws.amazon.com/lambda/latest/dg/welcome.html>

AWS API Gateway Developer Guide: <https://docs.aws.amazon.com/apigateway/latest/developerguide/welcome.html>

AWS Cognito Developer Guide: <https://docs.aws.amazon.com/cognito/latest/developerguide/what-is-amazon-cognito.html>

AWS Cognito SDK: <https://github.com/amazon-archives/amazon-cognito-identity-js>

AWS SDK: <https://docs.aws.amazon.com/AWSJavaScriptSDK/latest/>

Bootstrap: <https://getbootstrap.com>

Sublime Text: <https://www.sublimetext.com>

Brackets: <http://brackets.io>

UUID NPM Package: <https://www.npmjs.com/package/uuid>