

Library Room Reservation App

This document illustrates user stories, use cases, and user acceptance tests in the context of a library room reservation application.

The example illustrates the format for use cases well, but describes a system that would be impractical to implement in a single quarter.

User Stories

- As a student working in the library, I would like to be able to reserve a library room for my weekly meetings so that I will have a consistent weekly pattern.
- As a student who has made a recurring reservation, I would like to be able to log in and edit my meeting's details so that I can adjust as other requirements come up throughout the quarter and so that I can confirm the information is entered correctly.
- As a student studying for an exam, I would like to see if there is a room available for study RIGHT NOW so that I don't have to walk over to the library unless the room is available.
- As a librarian, I would like to be able to enter new rooms into the system so that students can sign up for them.
- As a librarian, I would like to block off rooms for special events so that students don't have any surprises.
- As a librarian, I would like to monitor room use to get a sense for the utilization of my rooms and to secure funding for library expansion if necessary.

[Insert diagram showing all user stories]

Stakeholders

- Librarians
- Students making reservations
- Other students in the group whose reservations are made

“Feature requests”

- Integration with class schedule system. This would be great for faculty, too!
- Allow scheduling fractions of an hour (to the minute?)
- Live update of options during entry based on availability

Use Case

Use Case 1 - Login

User Story

As a student working in the library, I would like to be able to login so I can reserve a room.

Preconditions

The student's username and login credentials are stored in the system.

Postconditions

The student is logged in.

Use Case 2 - Student Weekly Room Reservation

User Story

As a student working in the library, I would like to be able to reserve a library room for my weekly meetings so that I will have a consistent weekly pattern.

Goal

The student would like to log in, find a time that works for their entire team, and reserve that time for the remainder of the quarter/semester. They would like an email confirmation and the ability to log in and edit the reservation later on (as part of a different Use Case).

Actor

- The student making the reservation

Preconditions

- The student has an active school account with integrated login and status testing
- The room to be reserved has been entered into the system
- There is a time that can be registered weekly available still

Scenario 1.1 - Sunny-Day Path - Weekly first

1. The student logs in to the system using their school account on their laptop. The student's account is checked, confirming they have an active status.
2. The system displays a schedule showing the number of rooms available each hour of the day for the next week. It also displays a list of all the rooms and allows the student to select options including recurrent scheduling.
3. The student selects a time.
4. The student selects recurrent scheduling. Options of daily, weekly, bi-weekly, and monthly reoccurrence are presented, along with start and stop dates.
5. The student selects a weekly reoccurrence.
6. The student clicks on the stop date calendar icon. A calendar is displayed.
7. The student selects a stop date at the end of the quarter (manually checking school's schedule on a separate site). The date is displayed in the main window.

8. The student selects a room.
9. The student submits the request. The system confirms that the room is available at the selected time.
10. A summary screen displays the room name, day(s) of week, time, and stop and start dates.
11. The student finalizes the reservation request. The system reserves the room and displays a final confirmation screen.

Scenario 1.2 – Rainy Day Path – Non-current student

1. The student logs in to the system using their school account on their laptop. The system confirms the student's log in is valid, but finds the student does not have an active status.
2. The system displays "Your status is not active" along with contact information for whom to contact about this problem. It also includes a prompt for the librarian on how to change this information (since the librarian would have a challenging time reproducing this error message by logging in themselves).

Scenario 1.3 – Rainy Day Path – Hard to find room

1. The student logs in to the system using their school account on their laptop. The student's account is checked, confirming they have an active status.
2. The system displays a schedule showing the number of rooms available each hour of the day for the next week. It also displays a list of all the rooms and allows the student to select options including recurrent scheduling.
3. The student selects a time.
4. The student selects recurrent scheduling. Options of daily, weekly, bi-weekly, and monthly reoccurrence are presented, along with start and stop dates.
5. The student selects a weekly reoccurrence.
6. The student clicks on the stop date calendar icon. A calendar is displayed.
7. The student selects a stop date at the end of the quarter (manually checking school's schedule on a separate site). The date is displayed in the main window.
8. The student selects a room.
9. The student clicks the "submit" button. The system finds that there is a conflict with the schedule on a few dates. It displays these dates to the student.
10. Consulting with the team, the student selects a different hour of the week.
11. The student clicks the "submit" button again. The system confirms there are no conflicts this time.
12. A summary screen displays the room name, day(s) of week, time, and stop and start dates.
13. The student clicks "reserve" button to reserve the room. A final confirmation screen is displayed with a summary of the reservation.

Postconditions

- The room is marked as reserved for the specific hours named on the schedule.

Other use cases

Wow! I haven't even considered the librarian's use cases yet! I don't even have any user stories for the librarian adding any rooms to the system!

Acceptance Test Plans

Scenario 2.1 – Sunny-Day Path

Objectives

Confirm the ability to select a schedule when the schedule is mostly full.

Preconditions

- The testing setup populates the system with:
 - One of the student developer’s accounts, activated in a reduced-privilege mode
- A set of existing reservations for fake student accounts -- There is a time that can be registered weekly available still for one day, but not for another.
- A set of rooms matching MSOE’s setup has been entered into the system

Procedure

Follow Scenario 1.1, selecting the day that works.

Expected result

All the system responses described in Scenario 1.1.

Scenario 2.3 – Rainy Day Path – Hard to find room

Objectives

Confirm the ability to select a schedule when the schedule is mostly full.

Preconditions

Same as Scenario 1.4

Procedure

Follow Scenario 1.1, selecting the day that does not work first.

Expected result

All the system responses described in Scenario 1.4.

How many acceptance test plans?

It seems that thorough testing would require at least 10 scenarios for each user story, and about twice the number of user stories I have so far (for about 15 total), so about 150 acceptance test plans. (Wow!)

Concluding reflection

Having worked through this exercise over the course of two or three hours, I can see the value of the exercise before writing code. It helps me think through the details of the program before doing any design at all. The code I have described will take some hundreds of hours to write, and in a couple of hours, I have gotten a much better idea of what it would cost.