# Getting Ready: The Meeting Scheduler Problem

Understand the meeting scheduler design and learn the questions to further simplify this problem.

**We'll cover the following**

* [Problem definition](https://www.educative.io/order-confirmation/stripe/subscription-buy?payment_intent=pi_3O1FFeKhXp6R50hI1xakUX3g&payment_intent_client_secret=pi_3O1FFeKhXp6R50hI1xakUX3g_secret_EDla1wT2fgsm34I6up6bRJM7s&transaction_id=1d4a8dd5-516d-4906-aa95-9e9effdf447d#Problem-definition)
* [Expectations from the interviewee](https://www.educative.io/order-confirmation/stripe/subscription-buy?payment_intent=pi_3O1FFeKhXp6R50hI1xakUX3g&payment_intent_client_secret=pi_3O1FFeKhXp6R50hI1xakUX3g_secret_EDla1wT2fgsm34I6up6bRJM7s&transaction_id=1d4a8dd5-516d-4906-aa95-9e9effdf447d#Expectations-from-the-interviewee)
  + [Room assignment](https://www.educative.io/order-confirmation/stripe/subscription-buy?payment_intent=pi_3O1FFeKhXp6R50hI1xakUX3g&payment_intent_client_secret=pi_3O1FFeKhXp6R50hI1xakUX3g_secret_EDla1wT2fgsm34I6up6bRJM7s&transaction_id=1d4a8dd5-516d-4906-aa95-9e9effdf447d#Room-assignment)
  + [Availability of attendees](https://www.educative.io/order-confirmation/stripe/subscription-buy?payment_intent=pi_3O1FFeKhXp6R50hI1xakUX3g&payment_intent_client_secret=pi_3O1FFeKhXp6R50hI1xakUX3g_secret_EDla1wT2fgsm34I6up6bRJM7s&transaction_id=1d4a8dd5-516d-4906-aa95-9e9effdf447d#Availability-of-attendees)
* [Design approach](https://www.educative.io/order-confirmation/stripe/subscription-buy?payment_intent=pi_3O1FFeKhXp6R50hI1xakUX3g&payment_intent_client_secret=pi_3O1FFeKhXp6R50hI1xakUX3g_secret_EDla1wT2fgsm34I6up6bRJM7s&transaction_id=1d4a8dd5-516d-4906-aa95-9e9effdf447d#Design-approach)
* [Design pattern](https://www.educative.io/order-confirmation/stripe/subscription-buy?payment_intent=pi_3O1FFeKhXp6R50hI1xakUX3g&payment_intent_client_secret=pi_3O1FFeKhXp6R50hI1xakUX3g_secret_EDla1wT2fgsm34I6up6bRJM7s&transaction_id=1d4a8dd5-516d-4906-aa95-9e9effdf447d#Design-pattern)

## Problem definition

A **meeting scheduler** software allows organizations to schedule and book meetings for a group of participants. The scheduler determines a meeting time and location depending on the availability of the participants. It ensures that most of the intended participants can effectively meet on the specified date and interval. The system allows users to book and cancel meetings. The invited participants promptly receive these notifications. The organizer can also add new participants to a meeting after the meeting is scheduled.

Schedule a meeting using a meeting scheduler

## Expectations from the interviewee

It is important to narrow down the components you will include in your design of a meeting scheduler. The following section provides an overview of some of the main expectations that the interviewer will want to hear you discuss in more detail during the interview.

### Room assignment

A meeting scheduler is responsible for assigning a meeting room to a scheduled meeting. Make sure to ask the following questions from the interviewer to figure out how this assignment works:

* How does the system determine available rooms?
* How important is the capacity of a room when assigning a room for a meeting?

### Availability of attendees

There are multiple attendees in a meeting, and all attendees have different schedules. You may ask the following questions to understand how the scheduler works:

* How does the system check the availability of the attendees?
* How does the system access the meeting information of all attendees?

## Design approach

We'll design a meeting scheduler using the bottom-up design approach. For this purpose, we will follow the steps below:

* Identify and design the smallest components first, like an interval and meeting room.
* Use these small components to design additional components, for example, a meeting and a calendar.
* Repeat the steps above until we design the scheduler, which is the main component of the system.

## Design pattern

It is always a good practice to discuss the design patterns that the meeting scheduler falls under, during the interview. Stating the design patterns will give the interviewer a positive impression and shows that the interviewee is well-versed in the advanced concepts of object-oriented design.

The following design pattern is used to design the meeting scheduler:

* Singleton design pattern

Let's explore the requirements of the meeting scheduler in the next lesson.

B

**Requirements for the Meeting Scheduler**

Learn about all requirements of the meeting scheduler.

**We'll cover the following**

* [Requirement collection](https://www.educative.io/order-confirmation/stripe/subscription-buy?payment_intent=pi_3O1FFeKhXp6R50hI1xakUX3g&payment_intent_client_secret=pi_3O1FFeKhXp6R50hI1xakUX3g_secret_EDla1wT2fgsm34I6up6bRJM7s&transaction_id=1d4a8dd5-516d-4906-aa95-9e9effdf447d#Requirement-collection)

In this lesson, we'll list the requirements of the meeting scheduler. This is a very crucial step, since requirements define the scope of a problem, so getting them right from the interviewer and understanding them well will make the design of the rest of the system smooth and easy.

We’ll use the notational convention to identify each requirement with a unique label "Rn", where "R" is short for Requirement and "n" is a natural number.

**Requirement collection**

The requirements for the meeting scheduler design problem are defined below:

**R1:**There should be an n*n* number of meeting rooms.

**R2:**Each meeting room should have a specific capacity to accommodate the desired number of people.

**R3:**If not reserved already, each meeting room should have the ability to be booked, along with setting an interval, a start time, and an end time for the meeting.

**R4:**A notification should be sent to all the people invited to the meeting.

**R5:** Users will receive an invite regardless of whether they are available at the interval or not. Users can respond to the invitation by either accepting or rejecting the invite.

**R6:**Each user should have access to a calendar that is used to track the date and time, as well as to schedule or cancel meetings.

We've identified our requirements for the problem, and in the next lesson, we will define different use cases for the meeting scheduler.

# Use Case Diagram for the Meeting Scheduler

Learn how to define use cases and create the corresponding use case diagram for the meeting scheduler problem.

**We'll cover the following**

* [System](https://www.educative.io/order-confirmation/stripe/subscription-buy?payment_intent=pi_3O1FFeKhXp6R50hI1xakUX3g&payment_intent_client_secret=pi_3O1FFeKhXp6R50hI1xakUX3g_secret_EDla1wT2fgsm34I6up6bRJM7s&transaction_id=1d4a8dd5-516d-4906-aa95-9e9effdf447d#System)
* [Actors](https://www.educative.io/order-confirmation/stripe/subscription-buy?payment_intent=pi_3O1FFeKhXp6R50hI1xakUX3g&payment_intent_client_secret=pi_3O1FFeKhXp6R50hI1xakUX3g_secret_EDla1wT2fgsm34I6up6bRJM7s&transaction_id=1d4a8dd5-516d-4906-aa95-9e9effdf447d#Actors)
  + [Primary actors](https://www.educative.io/order-confirmation/stripe/subscription-buy?payment_intent=pi_3O1FFeKhXp6R50hI1xakUX3g&payment_intent_client_secret=pi_3O1FFeKhXp6R50hI1xakUX3g_secret_EDla1wT2fgsm34I6up6bRJM7s&transaction_id=1d4a8dd5-516d-4906-aa95-9e9effdf447d#Primary-actors)
  + [Secondary actors](https://www.educative.io/order-confirmation/stripe/subscription-buy?payment_intent=pi_3O1FFeKhXp6R50hI1xakUX3g&payment_intent_client_secret=pi_3O1FFeKhXp6R50hI1xakUX3g_secret_EDla1wT2fgsm34I6up6bRJM7s&transaction_id=1d4a8dd5-516d-4906-aa95-9e9effdf447d#Secondary-actors)
* [Use cases](https://www.educative.io/order-confirmation/stripe/subscription-buy?payment_intent=pi_3O1FFeKhXp6R50hI1xakUX3g&payment_intent_client_secret=pi_3O1FFeKhXp6R50hI1xakUX3g_secret_EDla1wT2fgsm34I6up6bRJM7s&transaction_id=1d4a8dd5-516d-4906-aa95-9e9effdf447d#Use-cases)
  + [Scheduler](https://www.educative.io/order-confirmation/stripe/subscription-buy?payment_intent=pi_3O1FFeKhXp6R50hI1xakUX3g&payment_intent_client_secret=pi_3O1FFeKhXp6R50hI1xakUX3g_secret_EDla1wT2fgsm34I6up6bRJM7s&transaction_id=1d4a8dd5-516d-4906-aa95-9e9effdf447d#Scheduler)
  + [User](https://www.educative.io/order-confirmation/stripe/subscription-buy?payment_intent=pi_3O1FFeKhXp6R50hI1xakUX3g&payment_intent_client_secret=pi_3O1FFeKhXp6R50hI1xakUX3g_secret_EDla1wT2fgsm34I6up6bRJM7s&transaction_id=1d4a8dd5-516d-4906-aa95-9e9effdf447d#User)
  + [System](https://www.educative.io/order-confirmation/stripe/subscription-buy?payment_intent=pi_3O1FFeKhXp6R50hI1xakUX3g&payment_intent_client_secret=pi_3O1FFeKhXp6R50hI1xakUX3g_secret_EDla1wT2fgsm34I6up6bRJM7s&transaction_id=1d4a8dd5-516d-4906-aa95-9e9effdf447d#System)
* [Relationships](https://www.educative.io/order-confirmation/stripe/subscription-buy?payment_intent=pi_3O1FFeKhXp6R50hI1xakUX3g&payment_intent_client_secret=pi_3O1FFeKhXp6R50hI1xakUX3g_secret_EDla1wT2fgsm34I6up6bRJM7s&transaction_id=1d4a8dd5-516d-4906-aa95-9e9effdf447d#Relationships)
  + [Associations](https://www.educative.io/order-confirmation/stripe/subscription-buy?payment_intent=pi_3O1FFeKhXp6R50hI1xakUX3g&payment_intent_client_secret=pi_3O1FFeKhXp6R50hI1xakUX3g_secret_EDla1wT2fgsm34I6up6bRJM7s&transaction_id=1d4a8dd5-516d-4906-aa95-9e9effdf447d#Associations)
  + [Include](https://www.educative.io/order-confirmation/stripe/subscription-buy?payment_intent=pi_3O1FFeKhXp6R50hI1xakUX3g&payment_intent_client_secret=pi_3O1FFeKhXp6R50hI1xakUX3g_secret_EDla1wT2fgsm34I6up6bRJM7s&transaction_id=1d4a8dd5-516d-4906-aa95-9e9effdf447d#Include)
* [Use case diagram](https://www.educative.io/order-confirmation/stripe/subscription-buy?payment_intent=pi_3O1FFeKhXp6R50hI1xakUX3g&payment_intent_client_secret=pi_3O1FFeKhXp6R50hI1xakUX3g_secret_EDla1wT2fgsm34I6up6bRJM7s&transaction_id=1d4a8dd5-516d-4906-aa95-9e9effdf447d#Use-case-diagram)

Let's build the use case diagram of the meeting scheduler and understand the relationship between its different components.

First, we'll define the different elements of the meeting scheduler system, followed by the complete use case diagram of the system.

## System

Our system is a "meeting scheduler."

## Actors

Now, we'll define the main actors of the meeting scheduler.

### Primary actors

* **Scheduler:**This actor can schedule and cancel meetings and book and release meeting rooms.
* **User:** This actor can accept and reject invitations and will decide their presence for the meetings.

### Secondary actors

* **System:** This is responsible for sending out notifications regarding any new meetings or cancellations.

## Use cases

In this section, we'll define the use cases for the meeting scheduler. We have listed the use cases according to their respective interactions with a particular actor.

### Scheduler

* **Schedule/Cancel meeting:**To schedule a new meeting or cancel an existing one
* **Book/Release room:** To book a room for a meeting or release it

### User

* **Attend meeting:** To attend a meeting
* **Accept/Reject meeting:**To accept or reject a meeting invitation

### System

* **Send invite notification:**To send a notification of any new meeting invitations
* **Send cancelation notification:**To send a notification of any meeting cancellations

## Relationships

We describe the relationships between and among actors and their use cases in this section.

### Associations

The table below shows the association relationship between actors and their use cases.

|  |  |  |
| --- | --- | --- |
| **Scheduler** | **User** | **System** |
| Schedule/Cancel meeting | Attend meeting | Send invite notification |
| Book/Release room | Accept/Reject meeting | Send cancelation notification |

### Include

* The "Schedule meeting" use case has an include relationship with "Book room," since a room also needs to be booked when a meeting is scheduled.
* The "Schedule meeting" use case also has an include relationship with "Send invite notification," since, whenever a new meeting is scheduled, an invite is sent to all participant's users.
* The "Cancel meeting" use case has an include relationship with the "Release room" use case, since the scheduled room needs to be set free whenever any meeting is canceled.
* The "Cancel meeting" use case has an include relationship with the "Send cancelation notification" use case, since, whenever a meeting is canceled, a cancelation notice is sent to all participant's users.
* The "Cancel meeting" use case also has an include relationship with the "Remove meeting from calendar" use case, since, once a meeting has been canceled, the interval time of all participant's users needs to be set free.

## Use case diagram

A diagram of a meeting

Description automatically generated

The use case diagram of the meeting scheduler

In the next lesson, we'll discuss the class diagram with a detailed explanation of all classes and their relationship with each other.

# Class Diagram for the Meeting Scheduler

Learn to create a class diagram for a meeting scheduler using the bottom-up approach.

**We'll cover the following**

* [Components of a meeting scheduler](https://www.educative.io/order-confirmation/stripe/subscription-buy?payment_intent=pi_3O1FFeKhXp6R50hI1xakUX3g&payment_intent_client_secret=pi_3O1FFeKhXp6R50hI1xakUX3g_secret_EDla1wT2fgsm34I6up6bRJM7s&transaction_id=1d4a8dd5-516d-4906-aa95-9e9effdf447d#Components-of-a-meeting-scheduler)
  + [User](https://www.educative.io/order-confirmation/stripe/subscription-buy?payment_intent=pi_3O1FFeKhXp6R50hI1xakUX3g&payment_intent_client_secret=pi_3O1FFeKhXp6R50hI1xakUX3g_secret_EDla1wT2fgsm34I6up6bRJM7s&transaction_id=1d4a8dd5-516d-4906-aa95-9e9effdf447d#User)
  + [Interval](https://www.educative.io/order-confirmation/stripe/subscription-buy?payment_intent=pi_3O1FFeKhXp6R50hI1xakUX3g&payment_intent_client_secret=pi_3O1FFeKhXp6R50hI1xakUX3g_secret_EDla1wT2fgsm34I6up6bRJM7s&transaction_id=1d4a8dd5-516d-4906-aa95-9e9effdf447d#Interval)
  + [Meeting room](https://www.educative.io/order-confirmation/stripe/subscription-buy?payment_intent=pi_3O1FFeKhXp6R50hI1xakUX3g&payment_intent_client_secret=pi_3O1FFeKhXp6R50hI1xakUX3g_secret_EDla1wT2fgsm34I6up6bRJM7s&transaction_id=1d4a8dd5-516d-4906-aa95-9e9effdf447d#Meeting-room)
  + [Meeting](https://www.educative.io/order-confirmation/stripe/subscription-buy?payment_intent=pi_3O1FFeKhXp6R50hI1xakUX3g&payment_intent_client_secret=pi_3O1FFeKhXp6R50hI1xakUX3g_secret_EDla1wT2fgsm34I6up6bRJM7s&transaction_id=1d4a8dd5-516d-4906-aa95-9e9effdf447d#Meeting)
  + [Calendar](https://www.educative.io/order-confirmation/stripe/subscription-buy?payment_intent=pi_3O1FFeKhXp6R50hI1xakUX3g&payment_intent_client_secret=pi_3O1FFeKhXp6R50hI1xakUX3g_secret_EDla1wT2fgsm34I6up6bRJM7s&transaction_id=1d4a8dd5-516d-4906-aa95-9e9effdf447d#Calendar)
  + [Meeting scheduler](https://www.educative.io/order-confirmation/stripe/subscription-buy?payment_intent=pi_3O1FFeKhXp6R50hI1xakUX3g&payment_intent_client_secret=pi_3O1FFeKhXp6R50hI1xakUX3g_secret_EDla1wT2fgsm34I6up6bRJM7s&transaction_id=1d4a8dd5-516d-4906-aa95-9e9effdf447d#Meeting-scheduler)
  + [Notification](https://www.educative.io/order-confirmation/stripe/subscription-buy?payment_intent=pi_3O1FFeKhXp6R50hI1xakUX3g&payment_intent_client_secret=pi_3O1FFeKhXp6R50hI1xakUX3g_secret_EDla1wT2fgsm34I6up6bRJM7s&transaction_id=1d4a8dd5-516d-4906-aa95-9e9effdf447d#Notification)
* [Relationship between the classes](https://www.educative.io/order-confirmation/stripe/subscription-buy?payment_intent=pi_3O1FFeKhXp6R50hI1xakUX3g&payment_intent_client_secret=pi_3O1FFeKhXp6R50hI1xakUX3g_secret_EDla1wT2fgsm34I6up6bRJM7s&transaction_id=1d4a8dd5-516d-4906-aa95-9e9effdf447d#Relationship-between-the-classes)
  + [Association](https://www.educative.io/order-confirmation/stripe/subscription-buy?payment_intent=pi_3O1FFeKhXp6R50hI1xakUX3g&payment_intent_client_secret=pi_3O1FFeKhXp6R50hI1xakUX3g_secret_EDla1wT2fgsm34I6up6bRJM7s&transaction_id=1d4a8dd5-516d-4906-aa95-9e9effdf447d#Association)
  + [Composition](https://www.educative.io/order-confirmation/stripe/subscription-buy?payment_intent=pi_3O1FFeKhXp6R50hI1xakUX3g&payment_intent_client_secret=pi_3O1FFeKhXp6R50hI1xakUX3g_secret_EDla1wT2fgsm34I6up6bRJM7s&transaction_id=1d4a8dd5-516d-4906-aa95-9e9effdf447d#Composition)
  + [Aggregation](https://www.educative.io/order-confirmation/stripe/subscription-buy?payment_intent=pi_3O1FFeKhXp6R50hI1xakUX3g&payment_intent_client_secret=pi_3O1FFeKhXp6R50hI1xakUX3g_secret_EDla1wT2fgsm34I6up6bRJM7s&transaction_id=1d4a8dd5-516d-4906-aa95-9e9effdf447d#Aggregation)
* [Class diagram of the meeting scheduler](https://www.educative.io/order-confirmation/stripe/subscription-buy?payment_intent=pi_3O1FFeKhXp6R50hI1xakUX3g&payment_intent_client_secret=pi_3O1FFeKhXp6R50hI1xakUX3g_secret_EDla1wT2fgsm34I6up6bRJM7s&transaction_id=1d4a8dd5-516d-4906-aa95-9e9effdf447d#Class-diagram-of-the-meeting-scheduler)
* [Design pattern](https://www.educative.io/order-confirmation/stripe/subscription-buy?payment_intent=pi_3O1FFeKhXp6R50hI1xakUX3g&payment_intent_client_secret=pi_3O1FFeKhXp6R50hI1xakUX3g_secret_EDla1wT2fgsm34I6up6bRJM7s&transaction_id=1d4a8dd5-516d-4906-aa95-9e9effdf447d#Design-pattern)

In this lesson, we'll identify and design the classes, abstract classes, and interfaces based on the requirements that we have previously gathered from the interviewer in a meeting scheduler.

## Components of a meeting scheduler

As mentioned earlier, we'll design the meeting scheduler using a bottom-up approach.

### User[**#**](https://www.educative.io/order-confirmation/stripe/subscription-buy?payment_intent=pi_3O1FFeKhXp6R50hI1xakUX3g&payment_intent_client_secret=pi_3O1FFeKhXp6R50hI1xakUX3g_secret_EDla1wT2fgsm34I6up6bRJM7s&transaction_id=1d4a8dd5-516d-4906-aa95-9e9effdf447d#User)

The Userclass is responsible for representing the personal information of a user such as their name, email, and can also accept or reject an invitation to a meeting.

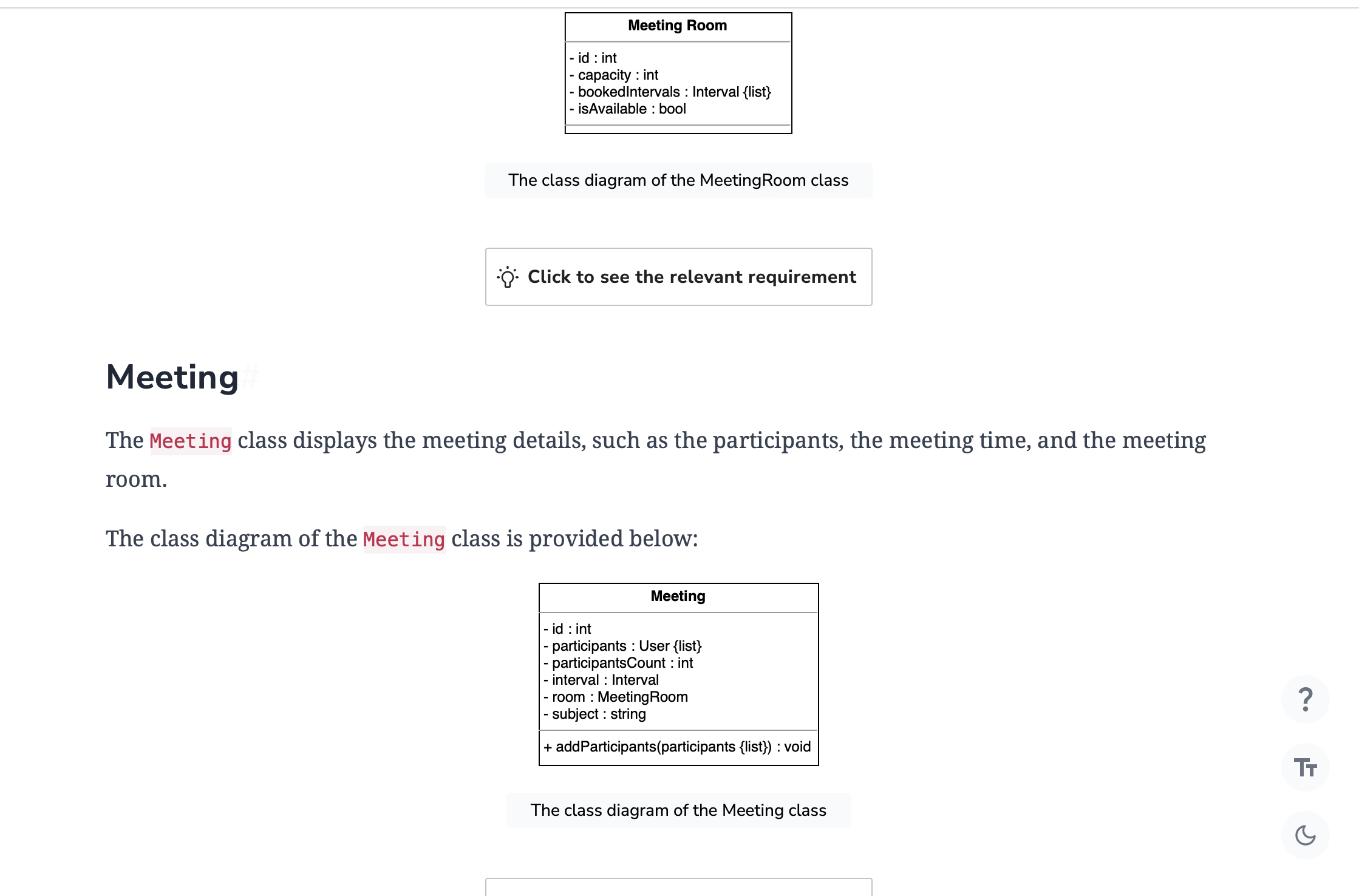
The class definition is shown below:

A screenshot of a computer

Description automatically generated

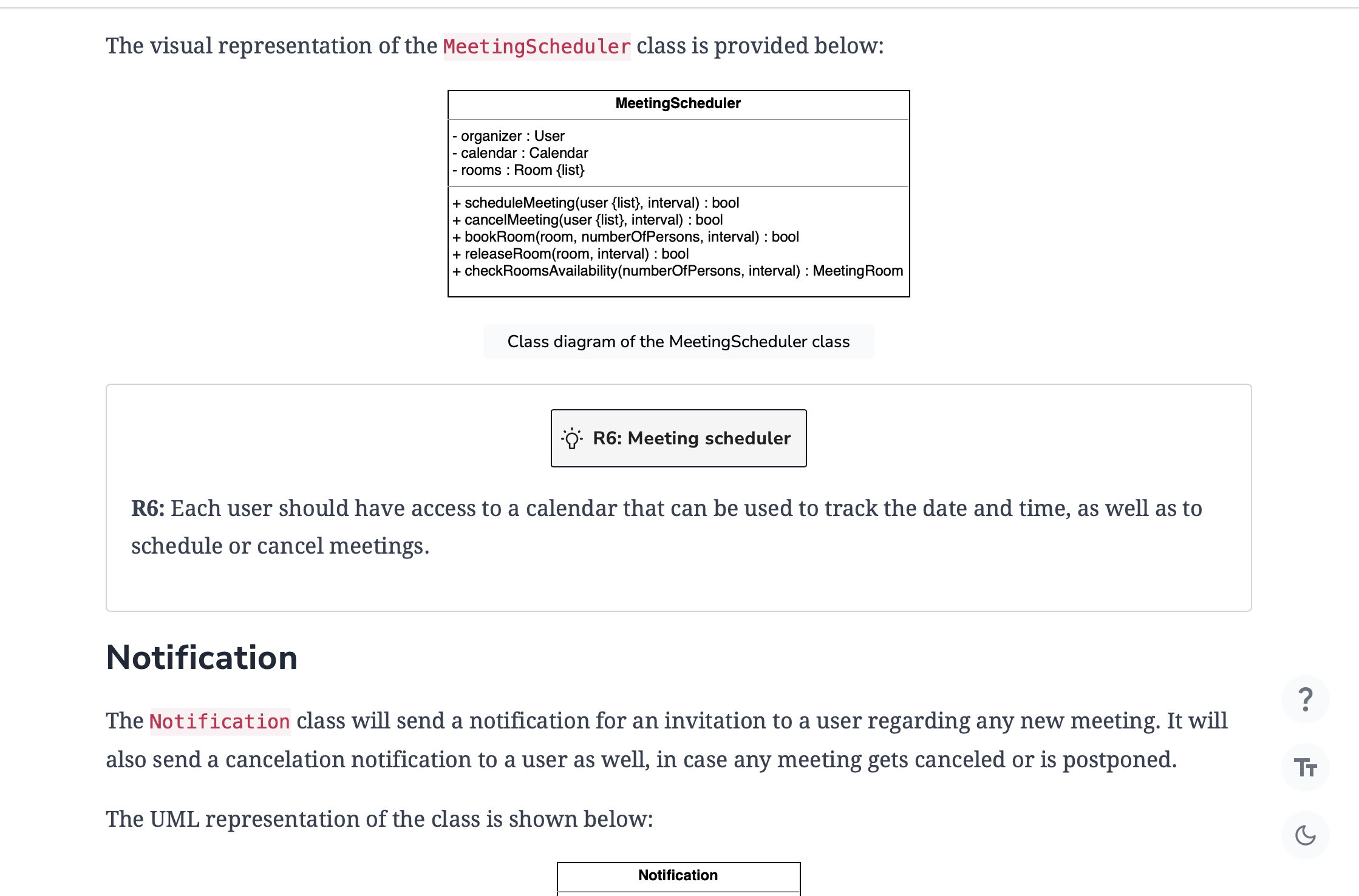
The MeetingRoom class contains the details of any particular room, such as its capacity and a status, to identify whether it is currently available. It also contains a list of intervals to keep track of when the room is booked for a meeting.

The class diagram of the MeetingRoom class is provided below:



A screenshot of a meeting schedule

Description automatically generated

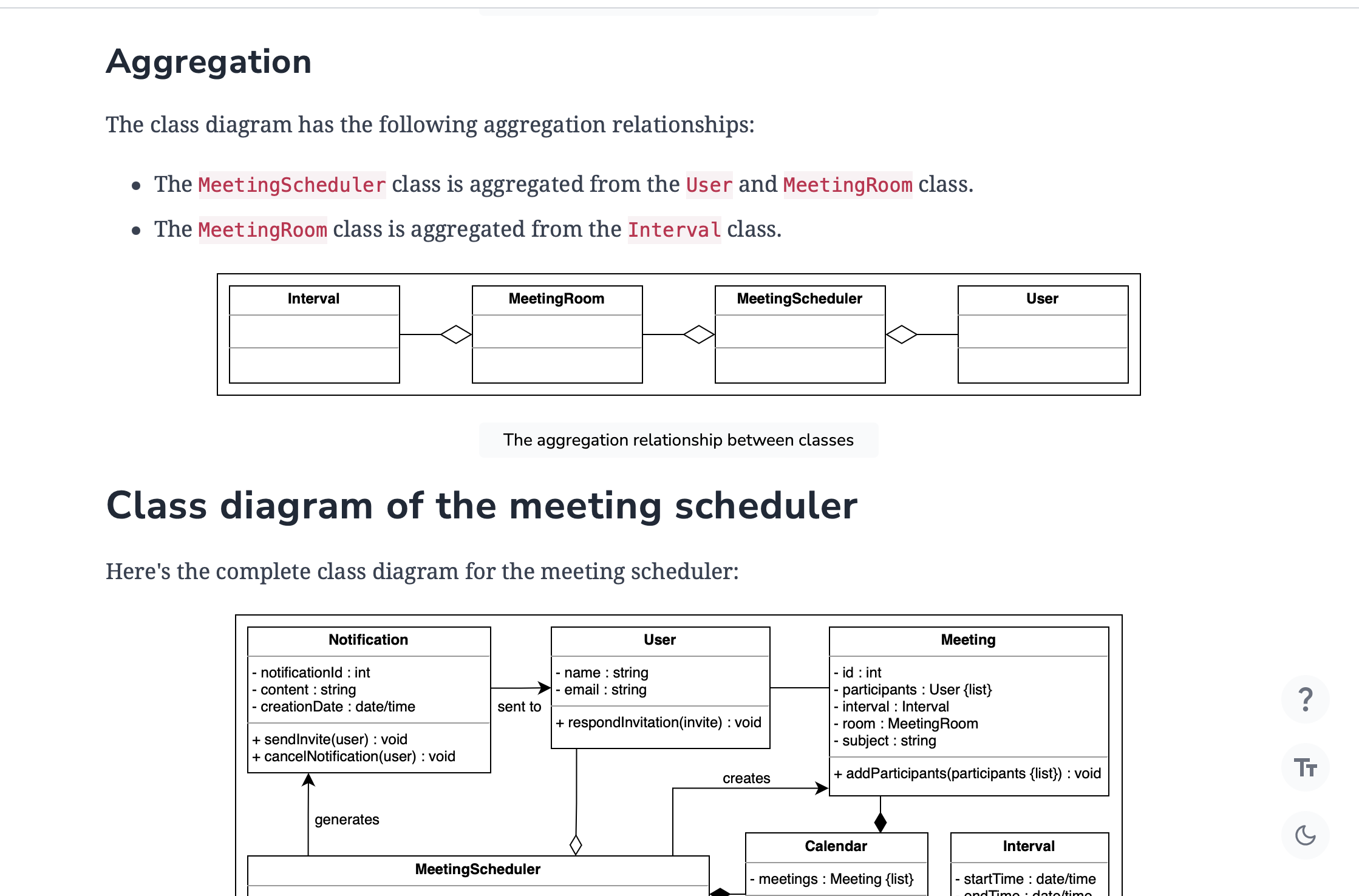


A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated



A diagram of a meeting scheduler

Description automatically generated

Therefore, we use the Singleton design pattern to ensure that only one instance of the scheduler is created and that this instance has a global point of access.

We have completed the class diagram of the meeting scheduler according to the requirements. Now let's design the sequence diagram of the meeting scheduler in the next lesson.

**Sequence Diagram for the Meeting Scheduler**

Visualize the sequence diagram for scheduling a meeting, and practice concepts by solving a challenge.

**We'll cover the following**

* [Schedule a meeting](https://www.educative.io/order-confirmation/stripe/subscription-buy?payment_intent=pi_3O1FFeKhXp6R50hI1xakUX3g&payment_intent_client_secret=pi_3O1FFeKhXp6R50hI1xakUX3g_secret_EDla1wT2fgsm34I6up6bRJM7s&transaction_id=1d4a8dd5-516d-4906-aa95-9e9effdf447d#Schedule-a-meeting)
* [Sequence challenge: Cancel meeting](https://www.educative.io/order-confirmation/stripe/subscription-buy?payment_intent=pi_3O1FFeKhXp6R50hI1xakUX3g&payment_intent_client_secret=pi_3O1FFeKhXp6R50hI1xakUX3g_secret_EDla1wT2fgsm34I6up6bRJM7s&transaction_id=1d4a8dd5-516d-4906-aa95-9e9effdf447d#Sequence-challenge:-Cancel-meeting)

Sequence diagrams are a great way to understand the interactions between different entities and objects in the system. There can be different sequence diagrams that we can create for our meeting scheduler. In this lesson, we will create sequence diagrams for the following interactions:

* **Schedule a meeting:** The meeting organizer schedules a meeting time for some attendees.
* **Sequence challenge:** The meeting organizer cancels a scheduled meeting.

**Schedule a meeting**

The sequence diagram to schedule a meeting should have the following actors and objects that will interact with each other:

* **Actor:** Organizer
* **Objects:** Scheduler, Calendar, MeetingRoom, and Meeting

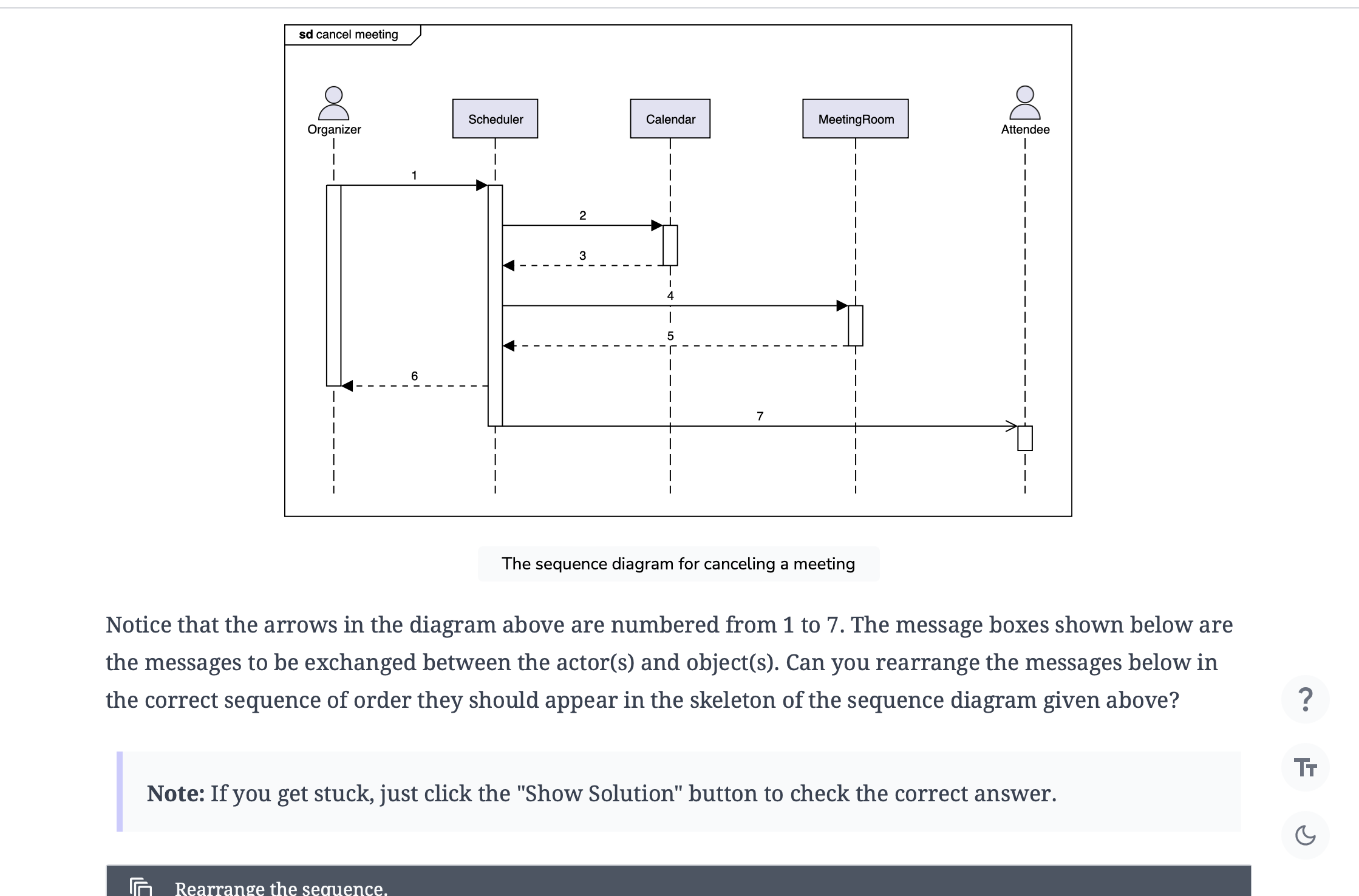
The steps in the schedule meeting interaction are listed below:

1. The meeting organizer schedules a meeting for some attendees at a given interval.
2. The scheduler checks for the availability of a meeting room.
3. If a room is available:
   1. The scheduler books the meeting room.
   2. The scheduler creates a new meeting.
   3. The scheduler updates the calendar with the new meeting.
   4. The scheduler informs the organizer that the meeting is scheduled.
   5. The scheduler sends an invite to the attendee.
4. Else if no room is available:
   1. The scheduler informs the organizer to select another meeting time.

Based on the order above, the sequence diagram for scheduling a meeting in the meeting scheduler system is given below:

A diagram of a meeting

Description automatically generated



A screenshot of a computer screen

Description automatically generated

# Activity Diagram for the Meeting Scheduler

Create some activity diagrams for the meeting scheduler problem.

**We'll cover the following**

* [Schedule meeting](https://www.educative.io/order-confirmation/stripe/subscription-buy?payment_intent=pi_3O1FFeKhXp6R50hI1xakUX3g&payment_intent_client_secret=pi_3O1FFeKhXp6R50hI1xakUX3g_secret_EDla1wT2fgsm34I6up6bRJM7s&transaction_id=1d4a8dd5-516d-4906-aa95-9e9effdf447d#Schedule-meeting)
  + [States](https://www.educative.io/order-confirmation/stripe/subscription-buy?payment_intent=pi_3O1FFeKhXp6R50hI1xakUX3g&payment_intent_client_secret=pi_3O1FFeKhXp6R50hI1xakUX3g_secret_EDla1wT2fgsm34I6up6bRJM7s&transaction_id=1d4a8dd5-516d-4906-aa95-9e9effdf447d#States)
  + [Actions](https://www.educative.io/order-confirmation/stripe/subscription-buy?payment_intent=pi_3O1FFeKhXp6R50hI1xakUX3g&payment_intent_client_secret=pi_3O1FFeKhXp6R50hI1xakUX3g_secret_EDla1wT2fgsm34I6up6bRJM7s&transaction_id=1d4a8dd5-516d-4906-aa95-9e9effdf447d#Actions)
* [Activity challenge: Respond to an invite](https://www.educative.io/order-confirmation/stripe/subscription-buy?payment_intent=pi_3O1FFeKhXp6R50hI1xakUX3g&payment_intent_client_secret=pi_3O1FFeKhXp6R50hI1xakUX3g_secret_EDla1wT2fgsm34I6up6bRJM7s&transaction_id=1d4a8dd5-516d-4906-aa95-9e9effdf447d#Activity-challenge:-Respond-to-an-invite)

Activity diagrams are a great way to visualize the flow of messages from one activity to the other in the system. There can be different activity diagrams that we can create for our meeting scheduler. In this lesson, we will create activity diagrams for the following two activities:

* Schedule a meeting
* **Activity challenge:**Respond to an invite

## Schedule meeting

The following states and actions will be involved in this activity diagram.

### States

**Initial state:**The user opens the calendar.

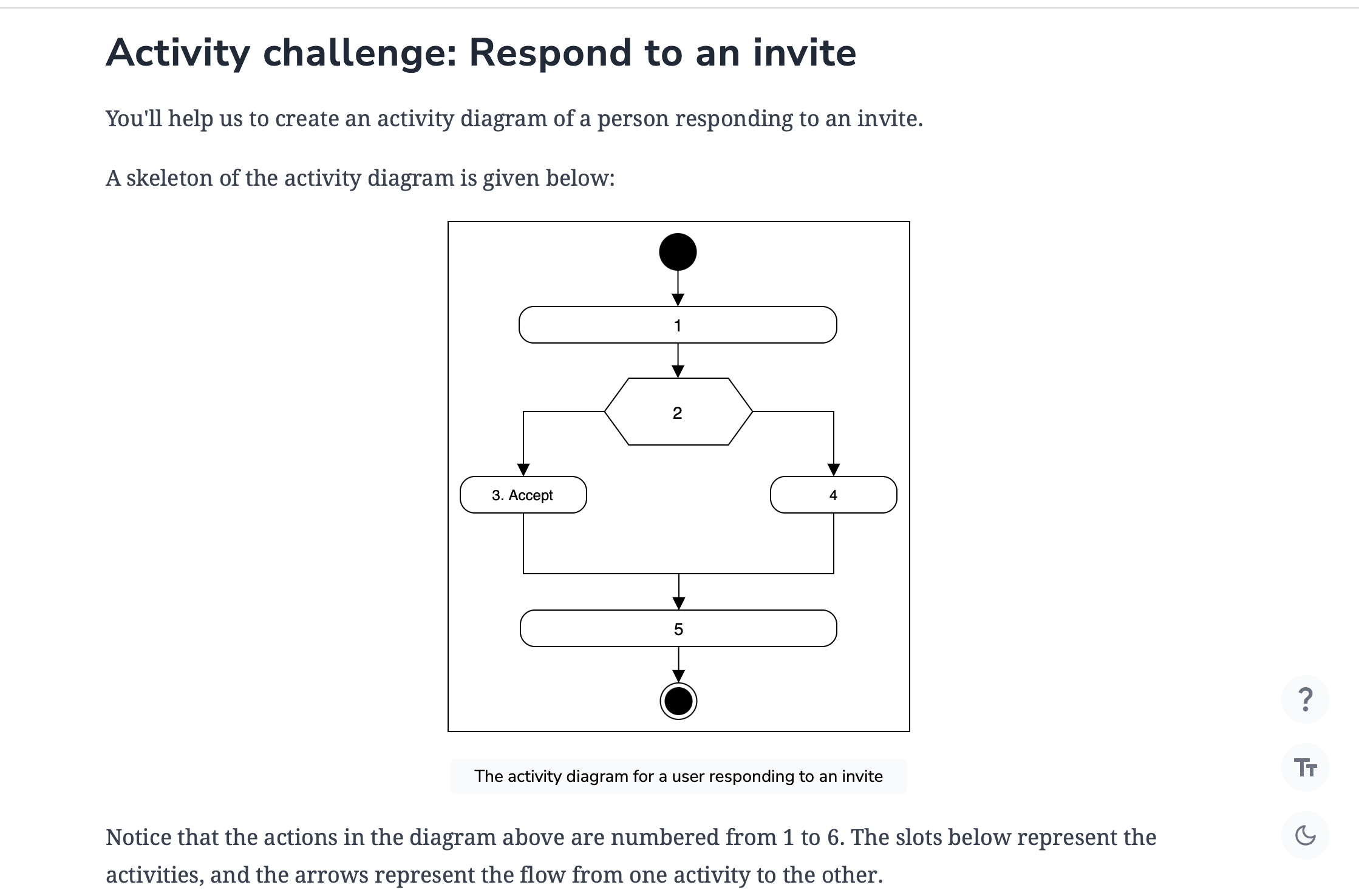
**Final state:**The notification and meeting details are sent to all the invited participants.

### Actions

The user opens the calendar and selects an available slot. The user books a meeting room and the meeting details are sent to all invited users.

A diagram of a meeting

Description automatically generated



Can you rearrange the slots below in the correct order they should appear in the activity diagram given above?

A screenshot of a computer

Description automatically generated

A screenshot of a computer screen

Description automatically generated

# Code for the Meeting Scheduler

Write the object-oriented code to implement the design of the meeting scheduler problem.

**We'll cover the following**

* [Meeting scheduler classes](https://www.educative.io/order-confirmation/stripe/subscription-buy?payment_intent=pi_3O1FFeKhXp6R50hI1xakUX3g&payment_intent_client_secret=pi_3O1FFeKhXp6R50hI1xakUX3g_secret_EDla1wT2fgsm34I6up6bRJM7s&transaction_id=1d4a8dd5-516d-4906-aa95-9e9effdf447d#Meeting-scheduler-classes)
  + [User](https://www.educative.io/order-confirmation/stripe/subscription-buy?payment_intent=pi_3O1FFeKhXp6R50hI1xakUX3g&payment_intent_client_secret=pi_3O1FFeKhXp6R50hI1xakUX3g_secret_EDla1wT2fgsm34I6up6bRJM7s&transaction_id=1d4a8dd5-516d-4906-aa95-9e9effdf447d#User)
  + [Interval](https://www.educative.io/order-confirmation/stripe/subscription-buy?payment_intent=pi_3O1FFeKhXp6R50hI1xakUX3g&payment_intent_client_secret=pi_3O1FFeKhXp6R50hI1xakUX3g_secret_EDla1wT2fgsm34I6up6bRJM7s&transaction_id=1d4a8dd5-516d-4906-aa95-9e9effdf447d#Interval)
  + [Meeting room](https://www.educative.io/order-confirmation/stripe/subscription-buy?payment_intent=pi_3O1FFeKhXp6R50hI1xakUX3g&payment_intent_client_secret=pi_3O1FFeKhXp6R50hI1xakUX3g_secret_EDla1wT2fgsm34I6up6bRJM7s&transaction_id=1d4a8dd5-516d-4906-aa95-9e9effdf447d#Meeting-room)
  + [Calendar](https://www.educative.io/order-confirmation/stripe/subscription-buy?payment_intent=pi_3O1FFeKhXp6R50hI1xakUX3g&payment_intent_client_secret=pi_3O1FFeKhXp6R50hI1xakUX3g_secret_EDla1wT2fgsm34I6up6bRJM7s&transaction_id=1d4a8dd5-516d-4906-aa95-9e9effdf447d#Calendar)
  + [Meeting](https://www.educative.io/order-confirmation/stripe/subscription-buy?payment_intent=pi_3O1FFeKhXp6R50hI1xakUX3g&payment_intent_client_secret=pi_3O1FFeKhXp6R50hI1xakUX3g_secret_EDla1wT2fgsm34I6up6bRJM7s&transaction_id=1d4a8dd5-516d-4906-aa95-9e9effdf447d#Meeting)
  + [Meeting scheduler](https://www.educative.io/order-confirmation/stripe/subscription-buy?payment_intent=pi_3O1FFeKhXp6R50hI1xakUX3g&payment_intent_client_secret=pi_3O1FFeKhXp6R50hI1xakUX3g_secret_EDla1wT2fgsm34I6up6bRJM7s&transaction_id=1d4a8dd5-516d-4906-aa95-9e9effdf447d#Meeting-scheduler)
  + [Notification](https://www.educative.io/order-confirmation/stripe/subscription-buy?payment_intent=pi_3O1FFeKhXp6R50hI1xakUX3g&payment_intent_client_secret=pi_3O1FFeKhXp6R50hI1xakUX3g_secret_EDla1wT2fgsm34I6up6bRJM7s&transaction_id=1d4a8dd5-516d-4906-aa95-9e9effdf447d#Notification)
* [Wrapping up](https://www.educative.io/order-confirmation/stripe/subscription-buy?payment_intent=pi_3O1FFeKhXp6R50hI1xakUX3g&payment_intent_client_secret=pi_3O1FFeKhXp6R50hI1xakUX3g_secret_EDla1wT2fgsm34I6up6bRJM7s&transaction_id=1d4a8dd5-516d-4906-aa95-9e9effdf447d#Wrapping-up)

We've reviewed the different aspects of the meeting scheduler and observed the attributes attached to the problem using various UML diagrams. Let's explore the more practical side of things, where we will work on implementing the meeting scheduler using multiple languages. This is usually the last step in an object-oriented design interview process.

We have chosen the following languages to write the skeleton code of the different classes present in the meeting scheduler:

* Java
* C#
* Python
* C++
* JavaScript

## Meeting scheduler classes

In this section, we will provide the skeleton code of the classes designed in the class diagram lesson.

**Note:** For simplicity, we are not defining getter and setter functions. The reader can assume that all class attributes are private and accessed through their respective public getter methods and modified only through their public method functions.

### User

The User class refers to a participant taking part in a meeting. A user can either accept or reject an invitation. The definition of this class is given below:

public class User {

private String name;

private String email;

public void respondInvitation(Notification invite);

}

A screenshot of a computer

Description automatically generated

### Calendar

The Calendar class contains a list of meetings to keep track of all the scheduled meetings. The definition of this class is provided below:

public class Calendar {

private List<Meeting> meetings;

}

### Meeting

The Meeting class outlines the meeting details such as the number and list of participants, meeting time interval, and meeting room. It also has the option to add more participants. The definition of this class is shown below:

public class Meeting {

private int id;

private int participantsCount;

private List<User> participants;

private Interval interval;

private MeetingRoom room;

private String subject

public void addParticipants(List<User> participants);

}

### Meeting scheduler

The MeetingScheduler class is the main class of the meeting scheduler and contains the organizer, which is responsible for scheduling and canceling a meeting as well as booking or releasing a room. It also checks if any meeting rooms are available for a meeting. In addition, there will be only one instance of the scheduler in the meeting scheduler. Therefore, the MeetingScheduler class will be a Singleton class to ensure that only one instance for the scheduler is created in the entire system.

The definition of this class is shown below:

public class MeetingScheduler {

private User organizer;

private Calendar calendar;

private List<MeetingRoom> rooms;

// Scheduler is a singleton class that ensures it will have only one active instance at a time

private static MeetingScheduler scheduler = null;

// Created a static method to access the singleton instance of Scheduler class

public static MeetingScheduler getInstance() {

if (scheduler == null) {

scheduler = new MeetingScheduler();

}

return scheduler;

}

public boolean scheduleMeeting(List<User> users, Interval interval);

public boolean cancelMeeting(List<User> users, Interval interval);

public boolean bookRoom(MeetingRoom room, int numberOfPersons, Interval interval);

public boolean releaseRoom(MeetingRoom room, Interval interval);

public MeetingRoom checkRoomsAvailability(int numberOfPersons, Interval interval);

}

### Notification

The Notification class is responsible for sending notifications to users about any new meetings or cancelations. The definition of this class is provided below:

Java

public class Notification {

private int notificationId;

private string content;

private Date creationDate;

public boolean sendNotification(User user);

public boolean cancelNotification(User user);

}

## Wrapping up

We've explored the complete design of the meeting scheduler in this chapter. We've looked at how the meeting scheduler can be visualized using various UML diagrams and designed using object-oriented principles and design patterns.