

# GoBears

## Iteration -1

Mahee Tayba  
Nishan Karki  
Pranish Bhagat  
Yudeep Rajbhandari

September 24, 2022

### Contents

|   |           |
|---|-----------|
| <b>1 Problem Statement</b>                | <b>2</b>  |
| <b>2 Project Vision</b>                   | <b>2</b>  |
| <b>3 List of Requirements</b>             | <b>2</b>  |
| 3.1 Functional Requirements . . . . .     | 2         |
| 3.2 Non-Functional Requirements . . . . . | 2         |
| <b>4 Use Case</b>                         | <b>3</b>  |
| <b>5 Use Case Diagram</b>                 | <b>16</b> |
| <b>6 System Sequence Diagrams</b>         | <b>18</b> |
| <b>7 System Operations</b>                | <b>31</b> |
| <b>8 Operation Contracts</b>              | <b>33</b> |
| <b>9 Wireframe</b>                        | <b>38</b> |
| <b>10 Domain Model</b>                    | <b>41</b> |
| <b>11 Activity Diagram</b>                | <b>43</b> |
| <b>12 Gantt Chart</b>                     | <b>54</b> |
| <b>13 TimeCard</b>                        | <b>60</b> |

# **1 Problem Statement**

The objective of this project is to provide an easy and secure University resource management system that is both conducive and supportive for the University business and students by digitizing navigation and access to buildings and rooms.

# **2 Project Vision**

The new system will provide easy and secure management in a friendly environment within the University that is both conducive and supportive for the University business and students by digitizing navigation and access to buildings and rooms.

Our vision is that our software will have the best system in its field and will work as promised. This system will drive University management and allow the University to plan successful Room/Resource Management System. The system will delight the University students by allowing them easily manageable Room/Resource Management System. With the incorporation of a digital system, and database management system, a secure online E-Room system can also be implemented in future, and a Web portal for students, the new system will enable University build its business goals and encourage future growth.

# **3 List of Requirements**

## **3.1 Functional Requirements**

1. View Schedule: Allow guests to view the schedule of various events going in in the University.
2. Booking room/resource: Enable users to reserve an existing room/resource of a building by searching and finding the most relevant option.
3. Building Navigation: Allow guest to find directions to a building from their location.
4. Indoor Navigation: Allow guest to find directions of rooms inside building.
5. Presenting data: Allow user to export pdf documents that show current/previous booking history.

## **3.2 Non-Functional Requirements**

1. Search results should be available within a time window of 5 seconds.
2. The schedules should be shown only after approval from the admin.

## 4 Use Case

Please find all use cases below:

1. Add building.
2. Add a room.
3. Assign a room.
4. Reserve a room
5. Insert new schedule and Update the available schedules.
6. Filter and view the available schedule.
7. See the reservation status of rooms in a building.
8. View, Search and Filter room resources
9. Update, Add and Remove room resources
10. Book and Checkout room resources
11. Generate a comprehensive report
12. Navigate to building.

Individual use cases are fully explained from the next page.

|   |
|---|
| Use case: View schedule   |
| Actors: Guest   |
| Precondition:<br>Schedule should be available in system   |
| <p>Flow of Events:</p> <ol style="list-style-type: none"> <li>1. Actors select view schedules.</li> <li>2. System searches for all the schedules of all rooms for the current day and if the system finds any schedules for any room: <ol style="list-style-type: none"> <li>a. The schedule is displayed.</li> </ol> </li> <li>3. Else <ol style="list-style-type: none"> <li>a. Shows a message stating “No schedule available for the day” and a button “Filter Schedules” <ol style="list-style-type: none"> <li>i. If the user clicks on the filter schedules <ol style="list-style-type: none"> <li>1. System asks for a date, then time, and then rooms.</li> <li>2. System searches for the schedule with the corresponding date, time, and room. <ol style="list-style-type: none"> <li>a. If the system finds the schedule <ol style="list-style-type: none"> <li>i. It displays on the screen.</li> </ol> </li> <li>b. Else <ol style="list-style-type: none"> <li>i. Show the message that no schedule is available with an option to go back to the previous page.</li> </ol> </li> </ol> </li> </ol> </li> </ol> </li> </ol> </li></ol> |
| PostConditions:   |
| Alternate flow:   |

|  |
|--|
| Use case: Insert Schedule  |
| Actors: Admin  |
| PreCondition: Actors have to be logged with role admin.  |
| <p>Flow of Events:</p> <ol style="list-style-type: none"> <li>1. User selects Add schedules. <ol style="list-style-type: none"> <li>a. User provides inputs for a date, time, and room and schedule name. <ol style="list-style-type: none"> <li>i. If a schedule exists for the selected options: <ol style="list-style-type: none"> <li>1. System gets the schedule and displays it as well as provides an option to update or delete the schedule.</li> <li>2. If the user selects delete, then the schedule is deleted.</li> <li>3. If the user modifies the schedule and selects update, then the schedule is updated.</li> </ol> </li> <li>ii. Else <ol style="list-style-type: none"> <li>1. User inputs the desired schedule.</li> <li>2. User selects Add.</li> </ol> </li> </ol> </li> </ol> </li> </ol> |
| Post Condition: The schedule will be added or modified in the database.  |
| Alternate flow:  |
| Extensions:  |

|  |
|--|
| Use case: Assign room  |
| Actors: Admins   |
| PreCondition: Actors have to be logged with role admin.  |
| <p>Flow of Events:</p> <ol style="list-style-type: none"> <li>2. Admin searches for the desired user. <ol style="list-style-type: none"> <li>a. If the user exists then the system returns the user <ol style="list-style-type: none"> <li>i. Admin searches for a room. <ol style="list-style-type: none"> <li>1. If a room exists then the system returns the room. <ol style="list-style-type: none"> <li>a. Admin looks up the resources available for the room.</li> <li>b. The system returns all available resources in the room.</li> <li>c. Admin selects the resource then the system gives the option to choose from date and duration.</li> <li>d. Admin assigns the room and the resource to the user for the given duration.</li> <li>e. The system returns the assignment added message.</li> </ol> </li> <li>2. Else <ol style="list-style-type: none"> <li>a. No room found message is displayed.</li> </ol> </li> </ol> </li> <li>ii. Else <ol style="list-style-type: none"> <li>1. No user-found message is displayed</li> </ol> </li> </ol> </li> </ol> </li> </ol> |
| Post Condition: The schedule will be added or modified in the database.  |
| Alternate flow:  |
| Extensions:  |

|  |
|--|
| Use case: Add building   |
| Actors: Admin  |
| Precondition:<br><ol style="list-style-type: none"><li>1. Actor needs to be logged in to the system with the role as admin.</li></ol>  |
| The flow of Events:<br><ol style="list-style-type: none"><li>1. Actor selects add a building.</li><li>2. System will ask for the following information: building name, building address, number of floors, building description, number of rooms on each floor, room name, room number, room type and bookable.</li><li>3. The actor saves the building information with a building name.<ol style="list-style-type: none"><li>a. If the actor tries to put a name the same as an existing one.<ol style="list-style-type: none"><li>i. The system shows the user a message specifying that the name has to be different from an already existing name.</li></ol></li><li>b. If the actor tries to give an empty name.<ol style="list-style-type: none"><li>i. The system denies the creation of an empty building name and shows the name can not be an empty message.</li></ol></li></ol></li><li>4. The system creates a new building with the name introduced and the creation date.</li></ol> |
| PostConditions:  |
| Alternate flow:  |

|  |
|--|
| Use case: Manage reservation status  |
| Actors: Admin  |
| Precondition: <ol style="list-style-type: none"> <li>1. Actor needs to be logged in to the system with the role as admin.</li> <li>2. At least one building information is saved in the system.</li> </ol>   |
| The flow of Events: <ol style="list-style-type: none"> <li>1. Admin selects a building.</li> <li>2. System provides the building information. the list of rooms that the building has.</li> <li>3. If the admin wants to see room status, <ol style="list-style-type: none"> <li>a. Admin selects display room status.</li> <li>b. System provides the list of booked, pending and unbooked rooms sorted by recent date.</li> <li>c. Admin may have an empty list. <ol style="list-style-type: none"> <li>i. System shows "0 result".</li> </ol> </li> </ol> </li> <li>4. If the admin wants to update a room status, <ol style="list-style-type: none"> <li>a. Admin selects a room.</li> <li>b. System provides room information and reservation requests.</li> <li>c. Admin checks all the constraints for the reservation request.</li> <li>d. If the requirements doesn't meet <ol style="list-style-type: none"> <li>i. Admin declines the request.</li> </ol> </li> <li>e. Admin approves the request.</li> </ol> </li> </ol> |
| PostConditions:  |
| Alternate flow:  |



|   |
|---|
| Use case: Reserve a room  |
| Actors: Registered User   |
| Precondition: <ol style="list-style-type: none"> <li>1. User needs to be logged in to the system.</li> <li>2. The room should be bookable</li> <li>3. Constraints on dates will be applied as described by admin.</li> </ol>  |
| The flow of Events: <ol style="list-style-type: none"> <li>1. Registered user selects a room.</li> <li>2. System displays information about the room (roomType, bookable).</li> <li>3. Registered user selects the Reserve room button. <ol style="list-style-type: none"> <li>a. Availability calendar is displayed and the user selects the dates from them.</li> <li>b. On submission of valid date, <ol style="list-style-type: none"> <li>i. The reservation request will be added to the system.</li> <li>ii. The system will forward the request to the admin and he will process the request.</li> <li>iii. Approval/Decline status will be sent back to the user.</li> </ol> </li> <li>c. For invalid date, the user will be redirected to select room.</li> </ol> </li> </ol> |
| PostConditions:   |
| Alternate flow: <ol style="list-style-type: none"> <li>1. Users can go back to the room without booking the room.</li> </ol>  |

|  |
|--|
| Use case: Search resource  |
| Actors: Registered user  |
| Precondition:<br>The actor must be logged-in.<br>There should be resource available in the system.   |
| The flow of Events (Search resource): <ol style="list-style-type: none"> <li>1. On the search field actor searches for a room by resource (Projector, lab, seating space, etc).</li> <li>2. System displays all the rooms that have searched resources. <ol style="list-style-type: none"> <li>a. If the actor selects a room. <ol style="list-style-type: none"> <li>i. System displays all the room and resource information available for the room.</li> </ol> </li> <li>b. If the actor clicks on the Filter button. <ol style="list-style-type: none"> <li>i. System displays all the filter options.</li> <li>ii. Actor selects all the filter options.</li> <li>iii. System applies the selected filters on the search.</li> <li>iv. System displays new list of room results.</li> </ol> </li> </ol> </li> </ol> |
| PostConditions:  |
| Alternate flow:  |

|   |
|---|
| Use case: Add resource  |
| Actors: Admin   |
| Precondition: <ol style="list-style-type: none"> <li>1. Actor is logged-in as admin.</li> </ol>   |
| Flow of Events (Add resource): <ol style="list-style-type: none"> <li>1. Actor selects a room.</li> <li>2. System displays information about all the resources for the room (Resource name, number available, resource category(lab, computer lab, auditorium, etc)).</li> <li>3. Actor clicks on the option for adding resource. <ol style="list-style-type: none"> <li>a. System provides an information page to be filled.</li> <li>b. Actors provide the information (name, number available, can be checked out, can be booked, constraint on booking dates and time, constraint on length of checkout times, resource category, etc) about the resource.</li> <li>c. User clicks on Submit, button. <ol style="list-style-type: none"> <li>i. If the resource already exist. <ol style="list-style-type: none"> <li>1. System displays resource already exist message.</li> <li>2. If actor select Update resource. <ol style="list-style-type: none"> <li>a. System displays update form.</li> <li>b. Actor provides updated informations.</li> <li>c. User clicks on Submit, button.</li> <li>d. System adds the record and displays a success/failure message.</li> </ol> </li> </ol> </li> <li>ii. Else system adds the record and displays a success/failure message.</li> </ol> </li> </ol> </li> </ol> |
| PostConditions:   |
| Alternate flow:   |

|   |
|---|
| Use case: Reserve resource  |
| Actors: Registered user   |
| Precondition:<br>Actor needs to be logged-in as a registered user   |
| Flow of Events (Reserve resource): <ol style="list-style-type: none"><li>1. Actor selects a resource.</li><li>2. System displays information about the resource (Can be checked out, can be booked, availability, book schedules, checkout schedules, etc)</li><li>3. Actor clicks the option to reserve a resource.<ol style="list-style-type: none"><li>a. System displays all available booking dates.</li><li>b. Actor chooses a date.</li><li>c. Actor submit the request.<ol style="list-style-type: none"><li>i. The system will forward the request to the admin for approval.</li><li>ii. Approval/Decline status will be sent back to the user.</li></ol></li></ol></li></ol> |
| PostConditions:   |
| Alternate flow:   |

|  |
|--|
| Use Case: Navigate to a building   |
| Description: This describes the process to navigate to a building.   |
| Actors: Guest.   |
| Preconditions: <ol style="list-style-type: none"><li>1. The required building should be in our system.</li><li>2. The user should allow location in their device.</li></ol>  |
| Main Flow:<br>Navigate to building. <ol style="list-style-type: none"><li>1. Guest searches for a building.</li><li>2. If building exists, the system will display the building's location along with associated info.</li><li>3. Else, the system displays building info not available.</li><li>4. The actor selects the option to navigate to the location.</li><li>5. The system will provide an interface with the direction from the actor to the building.</li></ol> |
| Post Conditions:   |

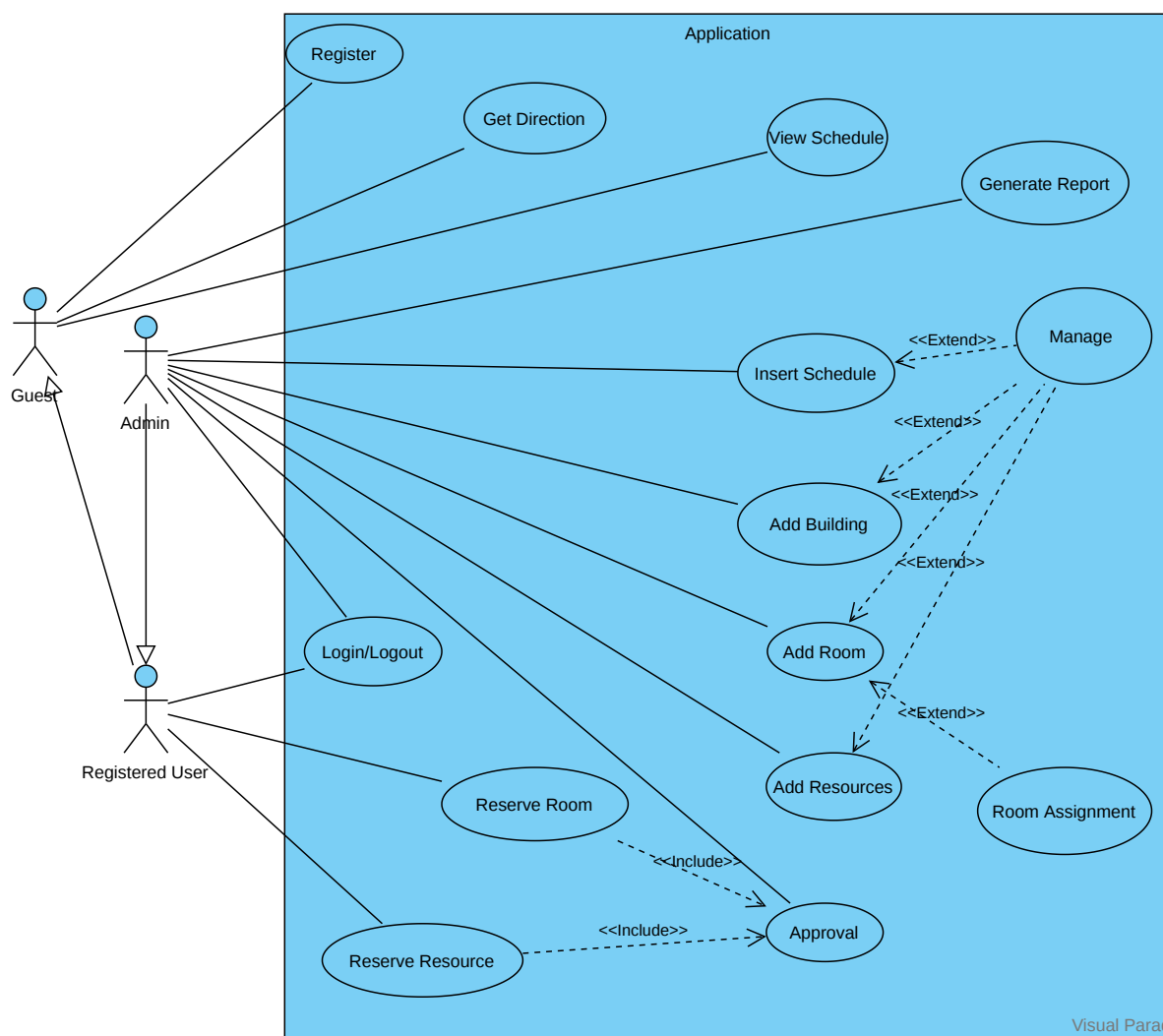
|  |
|--|
| Use Case: Add Room.  |
| Description: This describes the process of adding a new room to a building in the system.  |
| Notes: <ol style="list-style-type: none"> <li>1. No restrictions on the number of rooms that can be on the system.</li> <li>2. Only Admin can add a new room to the system.</li> </ol>   |
| Actor: Admin   |
| Preconditions: The Admin is logged into the system.  |
| Main Flow:<br>Create a New Room <ol style="list-style-type: none"> <li>1. The Admin selects an option to add a new room to the system.</li> <li>2. The system prompts the Admin for: <ol style="list-style-type: none"> <li>a. the following required information: <ol style="list-style-type: none"> <li>i. Associated building name/number.</li> <li>ii. Room number.</li> <li>iii. Allowed to book the room or not (Bookable?)</li> </ol> </li> <li>b. and the following non-required information: <ol style="list-style-type: none"> <li>i. Room Name if any.</li> <li>ii. Room Type</li> <li>iii. Co-ordinates of the room.</li> <li>iv. Special info about the room if any.</li> <li>v. Assigned to (For example: Professor's office, etc.)</li> </ol> </li> </ol> </li> <li>3. The Admin enters these parameters and submits.</li> <li>4. If any of the required fields are not completed, the admin is informed which required fields are missing <ol style="list-style-type: none"> <li>a. Input focus is returned to the first missing field</li> </ol> </li> <li>5. If the roomnumber is already in use, the system informs the Administrator that they must choose a different room number.</li> <li>6. Else, continue</li> <li>7. If the roomnumber is invalid, the Administrator is informed of this, and valid naming criteria is displayed</li> <li>8. The system creates the new room.</li> </ol> |
| Post Conditions: A new room is created on the system with at least three required attributes: associated building, room number and bookable status.  |
| Other Requirements <ol style="list-style-type: none"> <li>1. When the admin makes a data entry mistake and an error dialog is shown, always put input focus into the data entry field where the error occurred when the error dialog is discarded.</li> <li>2. RoomNumber are unique in the database.</li> <li>3. RoomNumber and RoomName are not case sensitive.</li> <li>4. A room cannot be named "UNKNOWN ROOM"</li> </ol>   |

|  |
|--|
| Use Case: Generate a comprehensive report  |
| Description: This occurs when the admin wants to generate a report of the system.  |
| Actor: Admin   |
| Preconditions: <ol style="list-style-type: none"> <li>1. The report option is configured in the system.</li> <li>2. An admin exists in the system.</li> </ol>  |
| Main Flow: <ol style="list-style-type: none"> <li>1. The admin selects an option to generate a report.</li> <li>2. The system provides an option to select the range of dates.</li> <li>3. The admin selects the range of dates.</li> <li>4. The system provides an option to select the list of building(s).</li> <li>5. The admin selects the required building(s).</li> <li>6. The system provides the associated list of room(s).</li> <li>7. The admin selects the required room(s).</li> <li>8. The system provides associated list of resource(s).</li> <li>9. The admin selects the required resource(s).</li> <li>10. The admin specifies the format of the generated report.</li> <li>11. The admin requests to generate the report.</li> <li>12. The system generates the report and requests the user to specify whether to display or save the report.</li> <li>13. The admin specifies a location to save the report.</li> <li>14. The system saves the report to the specified location.</li> </ol> |
| Post Conditions:   |
| Alternate Flow: <ol style="list-style-type: none"> <li>1. The user does not specify the list of buildings to include in the report. All the buildings with all the rooms and resources will be included in the report. The same for rooms and resources.</li> <li>2. The user does not specify how the report is delivered. The system uses the default report delivery mechanism.</li> <li>3. The user does not specify the format of the generated report. The system uses the default report format.</li> <li>4. The user cancels the report generation option.</li> <li>5. The user requests to display the report.</li> <li>6. The system displays the report.</li> </ol>   |
| Post Conditions:   |

## **5 Use Case Diagram**

The Use Case Diagram of our project is attached in the next page below.





## 6 System Sequence Diagrams

Please find the list of all the SSDs below:

1. Assign room to registered user.
2. Add schedule.
3. View schedule.
4. Add building.
5. Reserve room.
6. Update room status.
7. Add resource.
8. Reserve resource.
9. Search resource.
10. Navigate to building.
11. Add Room.
12. Generate a comprehensive report.

The SSDs are presented from the next page below.

sd [Assign room to user]

Admin

: System

1: SearchUser(userid)

alt

[if user is available]

1.1: User

2: SearchRoom(roomID)

alt

[if user is available]

2.1: Room

3: SearchResources(roomID)

3.1: List Of resources

4: selectResource(resourceID)

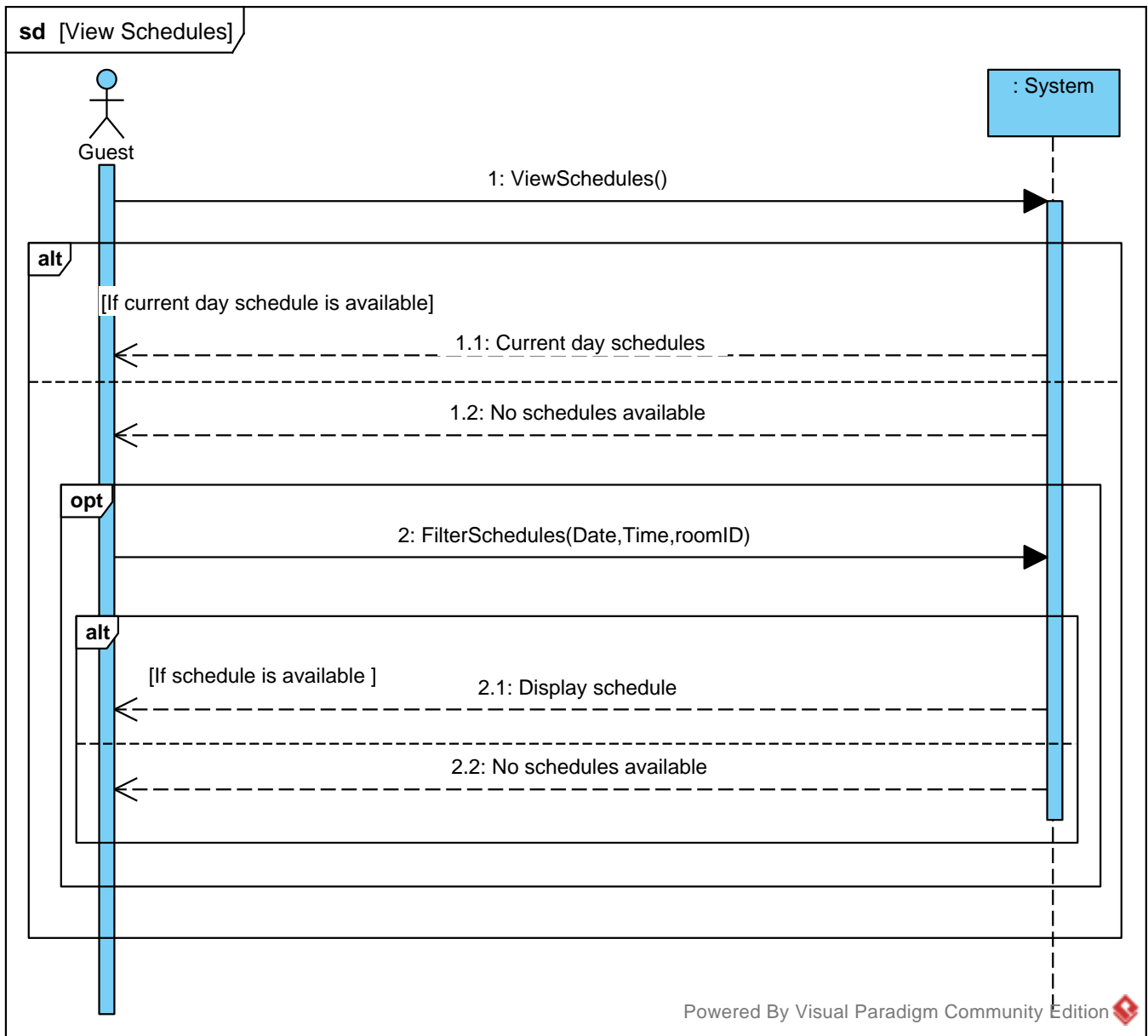
4.1: listofDateandTime

5: AssignUser(userid,roomID,resourceID,fromDate,Todate)

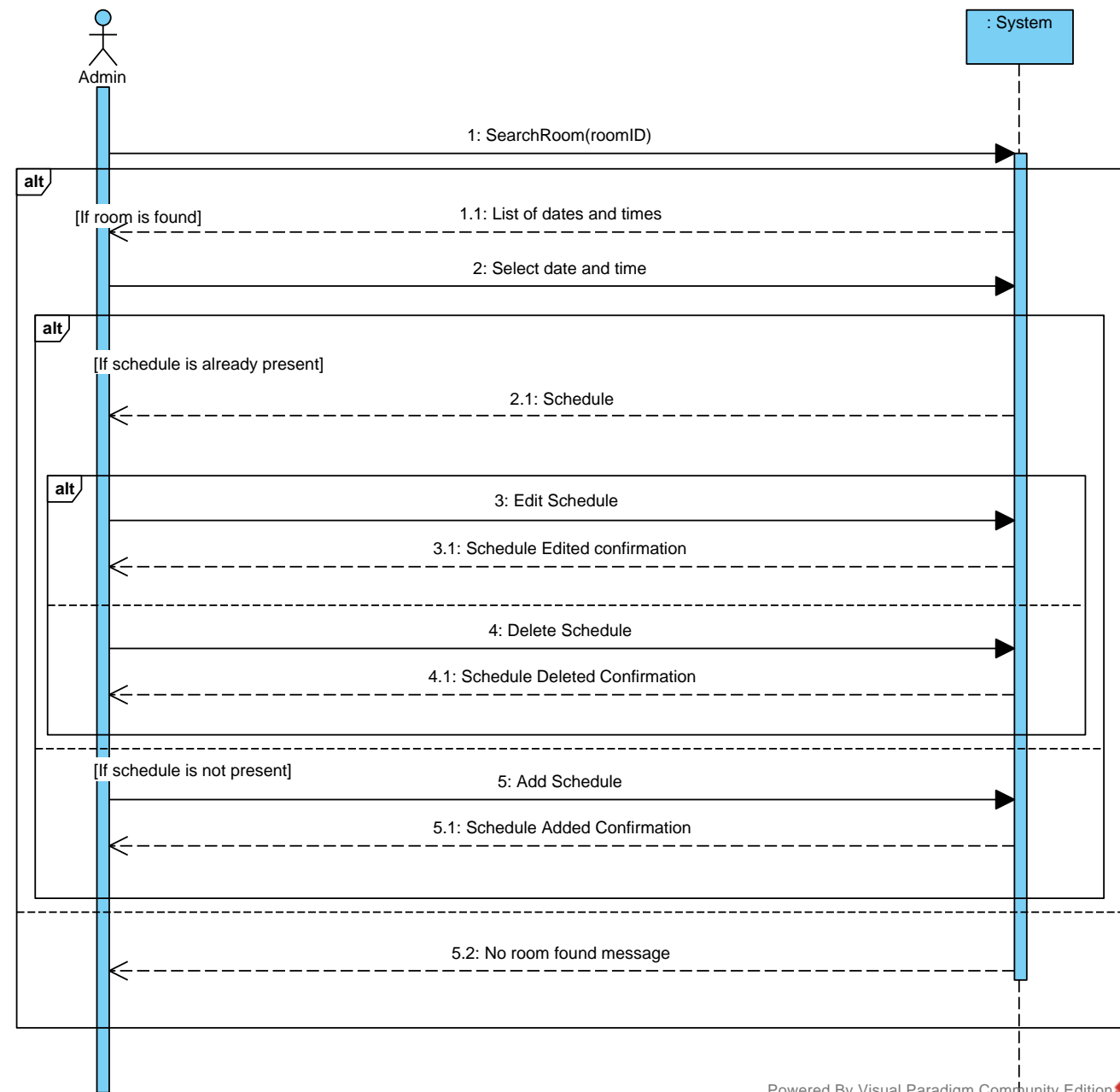
5.1: Assignment successful confirmation

5.2: No Room Found

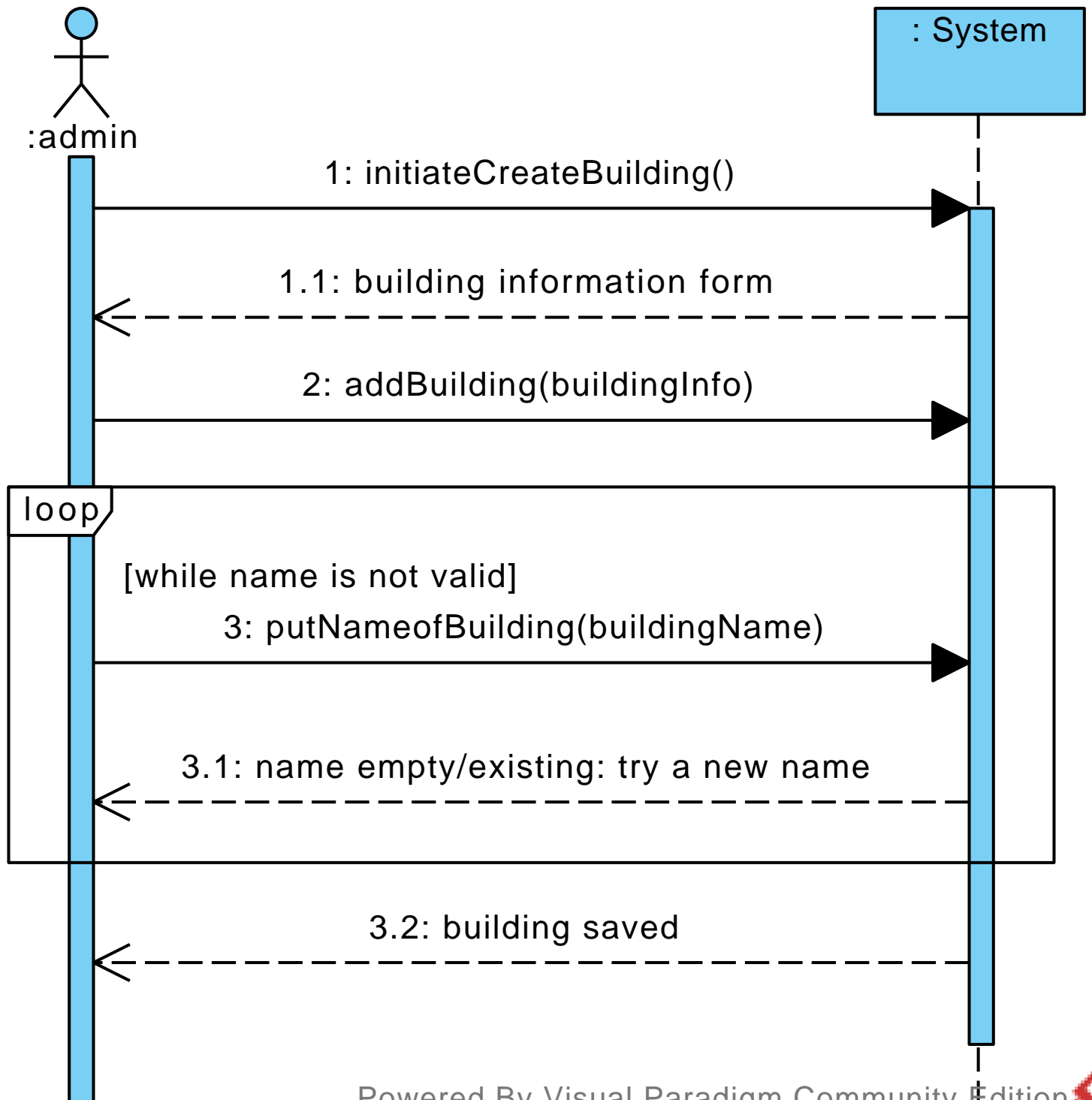
5.3: No User Found



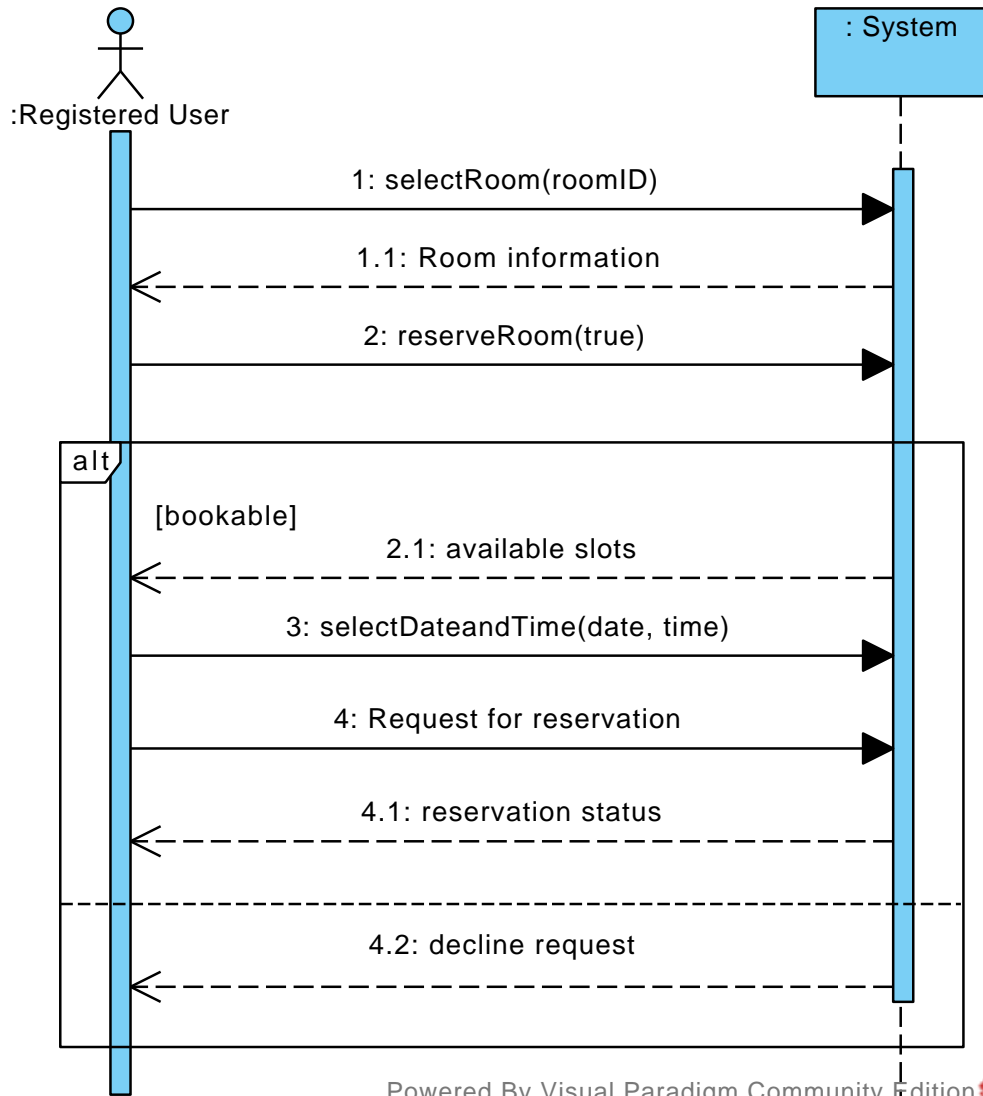
sd [Adding Schedule to System ]



sd [Add a building]



sd [Reserve Room]



# sd [View/update room status]

: Admin

: System

1: selectBuilding(buildingID)

1.1: building information

2: viewRoom()

2.1: list of rooms sorted by date

3: displayRoomStatus()

alt

[if not empty]

3.1: room list sorted by status

opt

[update room status]

4: viewRequest(roomID)

4.1: room information and reservation request

alt

[if requirements are meet] 5: approve()

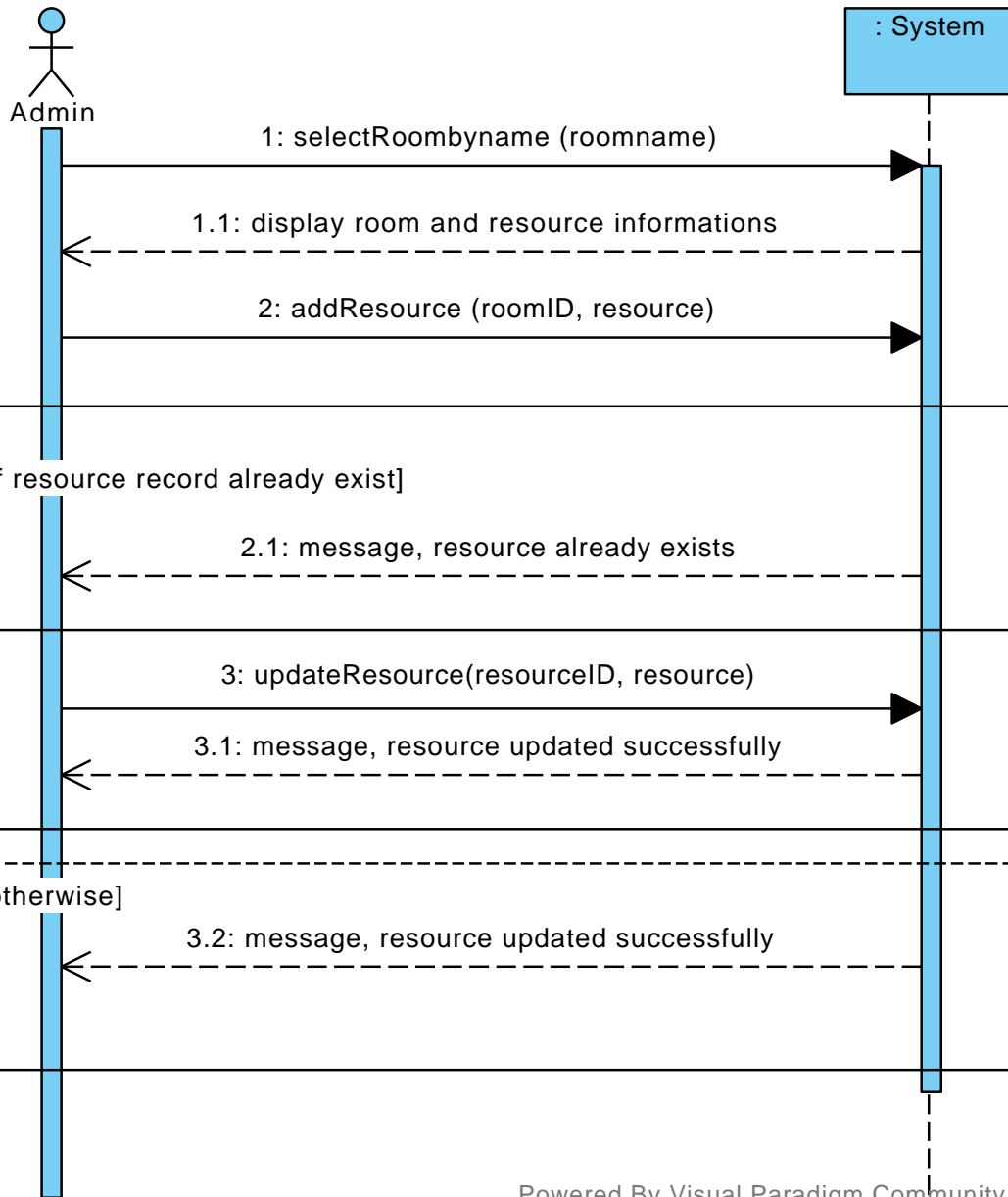
6: decline()

[empty list]

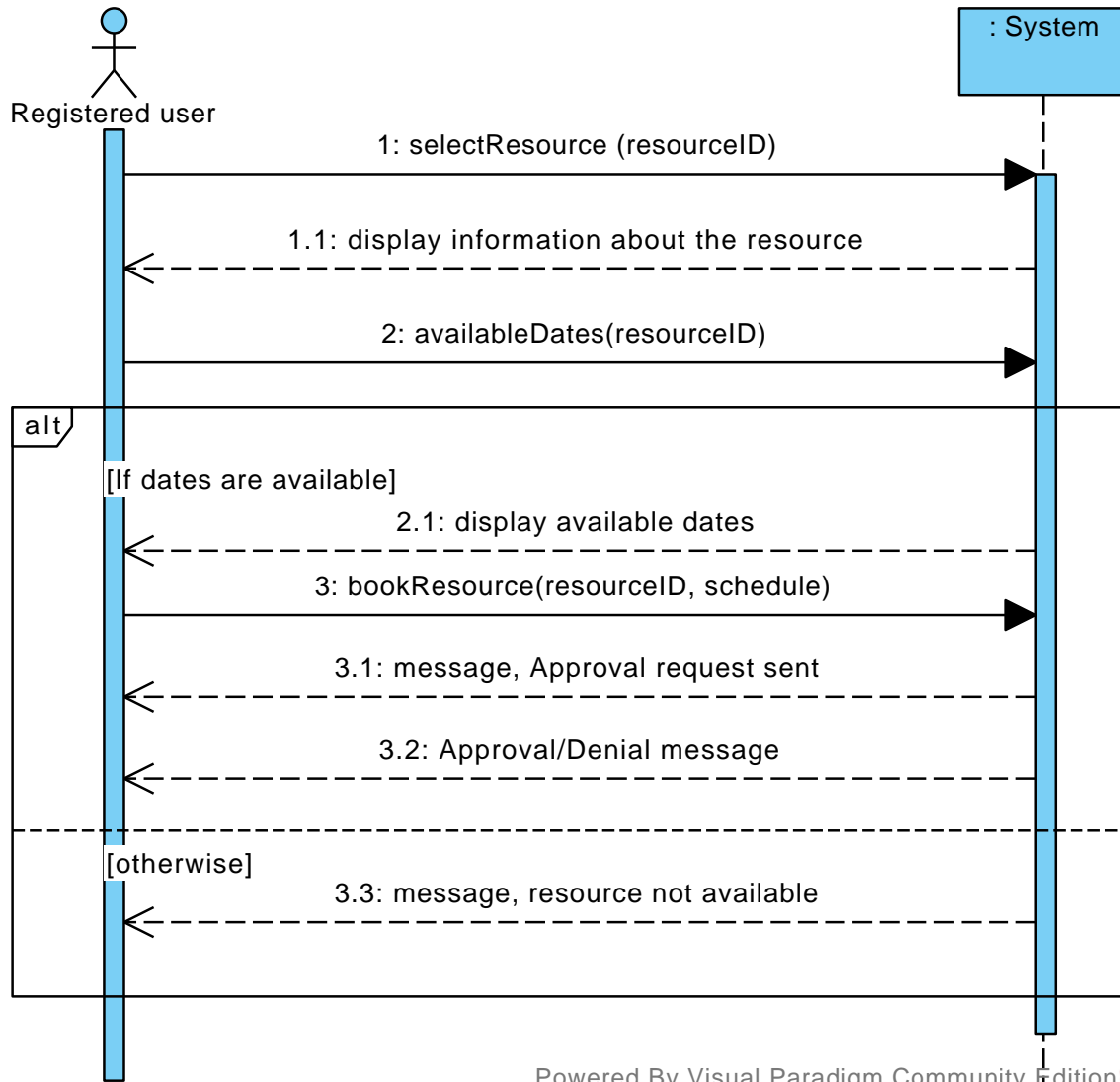
6.1: "0 result"

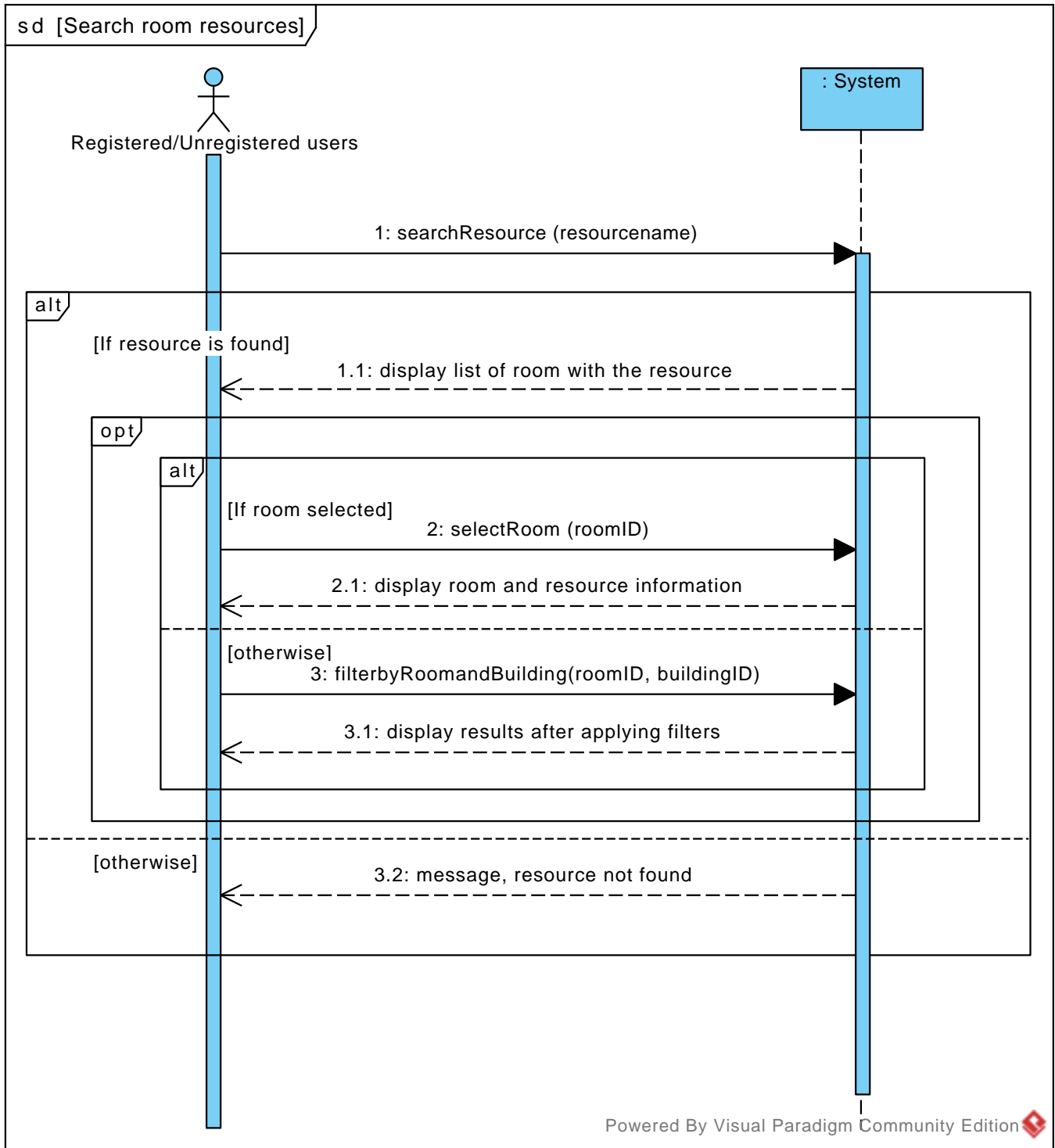


sd [Add room resources]

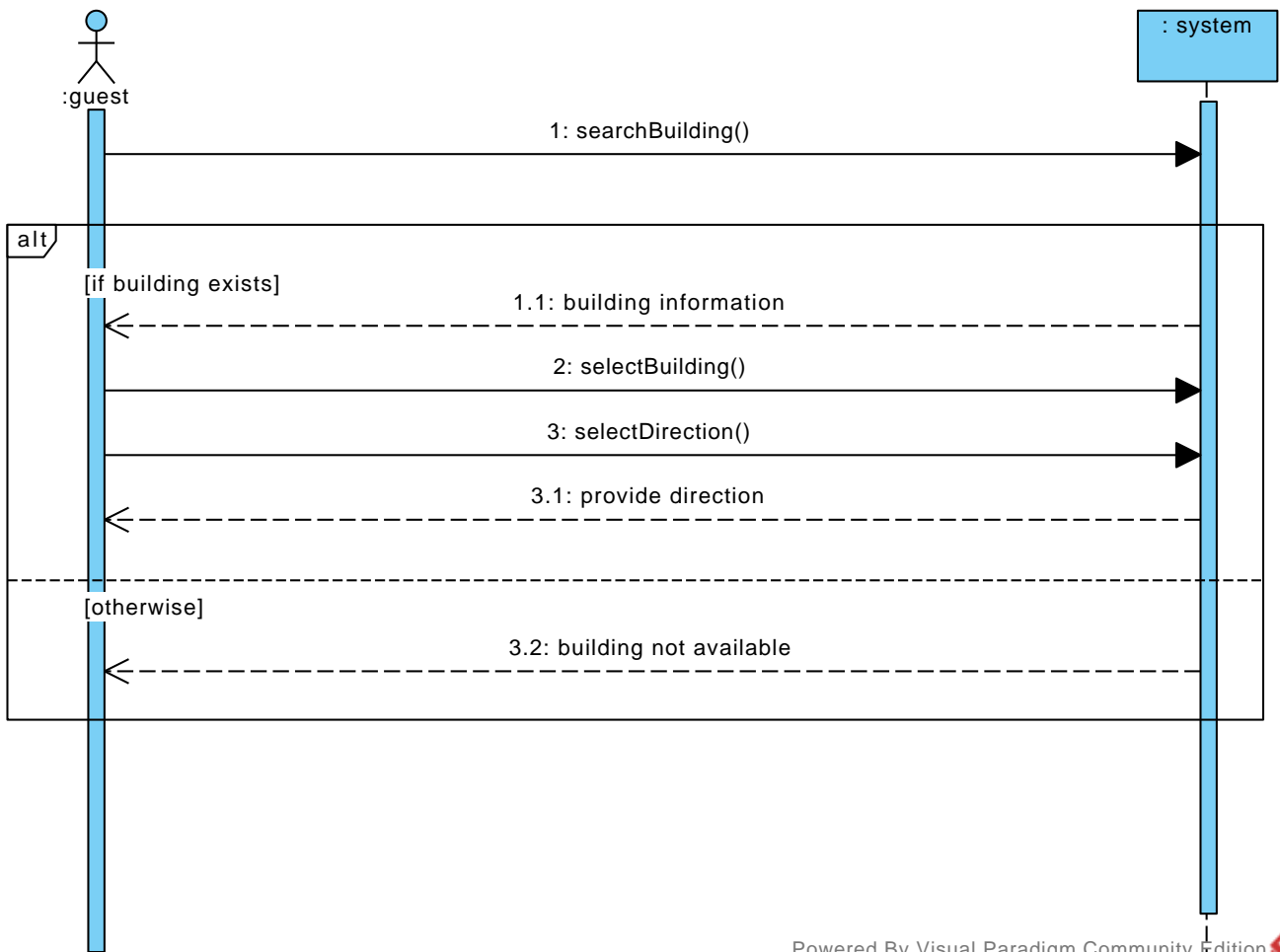


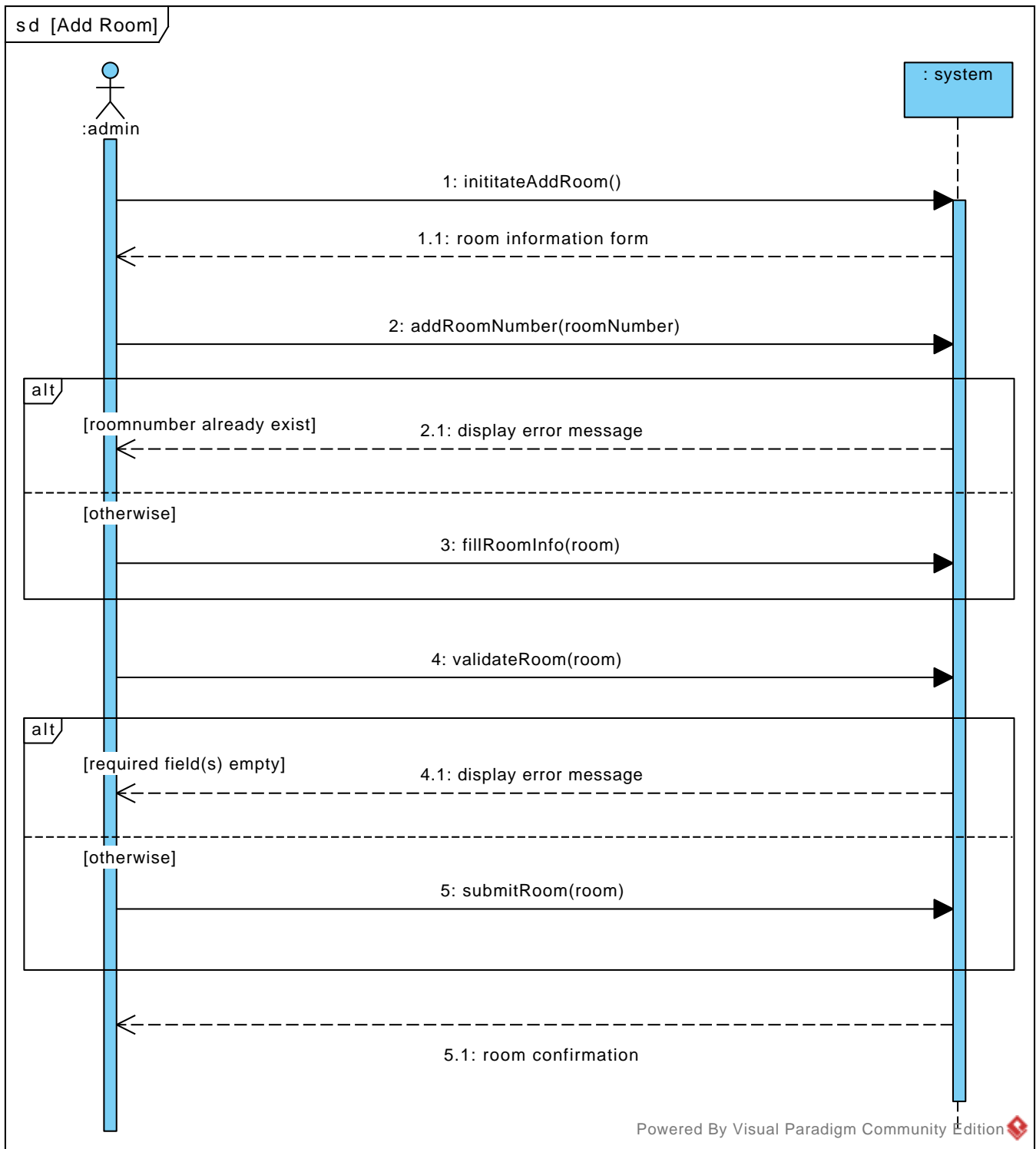
sd [Reserve resource]





s d [Navigate to a Building SSD]





**sd** [Generate Report SSD]

:admin

: system

1: generateReport()

1.1: list of dates

2: selectDates()

**opt**

[for selected dates]

2.1: list of building(s)

3: selectBuilding()

3.1: list of rooms

4: selectRoom()

4.1: list of resource(s)

5: selectResource()

6: exportReport(file)

6.1: file

## **7 System Operations**

The complete list of system operations is presented in the next page.

## System

```
+selectResource(resourceID)
+availableDates(resourceID)
+bookResource(resourceID, to_date, from_date)
+searchResource(resourceName)
+selectRoom(roomID)
+filterbyRoomAndBuilding(roomID, buildingID)
+selectRoomByName(roomname)
+addResource(roomID, resource)
+updateResource(roomID, resource)
+viewSchedule(schedule)
+filterSchedule()
+searchRoom()
+addSchedule(schedule)
+editSchedule(schedule)
+deleteSchedule(schedule)
+searchUser()
+assignUser()
+initiateCreateBuilding()
+addBuilding(building)
+reserveRoom()
+selectBuilding(building)
+viewRoom(room)
+displayRoomStatus(room)
+searchBuilding()
+selectDirection()
+initiateAddRoom()
+addRoomNumber(roomNumber)
+fillRoomInfo(room)
+validateRoom(room)
+submitRoom(room)
+generateReport()
+exportReport(file)
```



## 8 Operation Contracts

The operation contracts are presented from the next page.

#### Contract CO1: addSchedule

Operation: addSchedule(roomNumber: integer, calendar: Calendar, schedule: String)

Cross References: Use Cases: Insert/Update the schedule

Pre-Conditions:

Post-Conditions:

- Schedule instance sc was created.
- sc was associated with the Room.
- Room.schedule was set to sc.
- sc was associated with Admin.

#### Contract CO2: ViewSchedule

Operation: ViewSchedule()

Cross References: Use Cases: ViewSchedule

Pre-Conditions: The schedule should be in the system.

Post-Conditions:

#### Contract CO3: AssignUser

Operation: AssignUser(userid:String,roomID:String,resourceID:String)

Cross References: Use Cases: Assign users to room

Pre-Conditions:

Post-Conditions:

- Assignment instance am was created.
- Am is associated with room
- Room.am was set to am
- Am is associated with resource.
- Resource.am was set to am
- Am was associated with admin

#### Contract CO4: addRoom

Operation: addRoom(roomNumber: string, roomName:String, building:Building, type: RoomType, bookable: Boolean, location: coordinates)

Cross reference: Use Cases: Add Room

Pre-conditions:

Post-conditions:

- An addRoom instance *ar* was created
- *ar* is associated with an Admin and a building.

Contract CO5: editAssignment

Operation: editAssignment(assignment:Assignment, room:Room, resource:Resource)

Cross References: Use Cases: Assign users to room

Pre-Conditions:

Post-Conditions:

- Assignment.Room becomes room
- Assignment.Resource becomes Resource

Contract CO6: addBuilding

Operation: addBuilding(buildingName: String,description:String,numberOfFloor:Int,room:Room)

Cross reference: Use Cases: Create a building

Pre-conditions:

Post-conditions:

- An addBuilding instance ab was created
- ab is associated with an Admin.

Contract CO7: deleteAssignment

Operation: deleteAssignment(assignment:Assignment)

Cross References: Use Cases: Assign users to room

Pre-Conditions:

Post-Conditions:

- Association of assignment with Room is deleted
- Association of assignment with Resourceis deleted
- The instance assignment is deleted.

Contract CO8: editSchedule

Operation: editSchedule(schedule:Schedule, fromDate: Date,toDate:date,room:Room)

Cross References: Use Cases: Assign users to room

Pre-Conditions:

Post-Conditions:

- schedule.fromDate becomes fromDate
- schedule.toDate becomes toDate
- Schedule.room becomes room

Contract CO9: FilterSchedules

Operation: FilterSchedules(fromDate: Date,toDate:date,roomID:String)

Cross References: Use Cases: ViewSchedule

Pre-Conditions: Schedule must be available in the system.

Post-Conditions:

Contract CO10: DeleteSchedule

Operation: DeleteSchedule(schedule:Schedule)

Cross References: Use Cases: Assign users to room

Pre-Conditions:

Post-Conditions:

- Association of schedule with Room is deleted
- The instance schedule is deleted.

Contract CO11: displayRoomStatus

Operation: displayRoomStatus()

Cross reference: Use Cases: Show/Update room status

Pre-conditions:

- At least one building information is saved in the system.

Contract CO12: reserveRoom

Operation: reserveRoom(toDate:Date, fromDate:Date)

Cross reference: Use Cases: Reserve a room

Pre-conditions: There is a booking request underway.

Post-conditions:

- A reserveRoom instance rr is created and associated with a Registered User.

Contract CO13: approve

Operation: approve(Reservation reservation, Status status)

Cross reference: Use Cases: Show/update room status

Pre-conditions: There is a reservation request underway

Post-conditions:

- A reservation.status will be associated with status

Contract CO14: addResource

Operation: addResource(roomID: integer, resource: Resource)

Cross References: Use Cases: Add room resources

Pre-Conditions: There is an add resource request underway

Post-Conditions:

- Resource instance re was created.
- re was associated with Room.
- sc was associated with Admin.
- re was associated with Reservation

Contract CO15: bookResource

Operation: bookResource(resourceID: integer, to\_date: Date, from\_date Date)

Cross References: Use Cases: Book room resources

Pre-Conditions: There is a book resource request underway

Post-Conditions:

- Reservation instance re was created
- re was associated with Registered user.
- re was associated with Resource.
- Re was associated with Room.
- Reservation.to\_date was set to to\_date.
- Reservation.from\_date was set to from\_date.

Contract CO16: updateResource

Operation: updateResource(resourceID: integer, resource: Resource)

Cross References: Use Cases: Add room resources

Pre-Conditions: There is a add resource request underway

Post-Conditions:

- Resource.id is set to resourceID

Contract CO17: filterbyRoomandBuilding

Operation: filterbyRoomandBuilding (roomID :String, buildingID: String);

Cross References: Use Cases: Search room resources

Pre-Conditions: There is a search resource request underway

Post-Conditions:

## 9 Wireframe

The wireframes are presented from the next page below:

|   |   |
|---|---|
| <p>Building infos:<br/>This is hankamar building<br/>built in 1996, this<br/>building has its<br/>importance....</p> <p>Important offices:<br/>International affairs<br/>office(Floor 1)<br/>Nav: Right from gate 1</p> <p>Rest rooms available in:<br/>Lobby<br/>Floor 2 Nav: Left from<br/>Lobby stair<br/>Floor 3 Nav: Right from<br/>elevator 3</p> | <p><b>Schedules</b></p> <p><b>8 - 8 :30:</b><br/><b>C314 - CSI 5V94 lectures</b><br/><b>C315 - CSI 3345</b></p> <p><b>8:45 - 9:15</b><br/><b>C123 - Conference on data mining</b><br/><b>C222 - GSA Meeting</b></p> |
|---|---|

Figure 1: Design 1 for the intended viewing of the schedule

| SCHEDULE |       |         |
|----------|-------|---------|
| ROOM     | TIME  | EVENT   |
| L315     | 11:00 | SE 5324 |
|          |       |         |
|          |       |         |
|          |       |         |
|          |       |         |
|          |       |         |
|          |       |         |

Figure 2: Design 2 for the intended viewing of the schedule

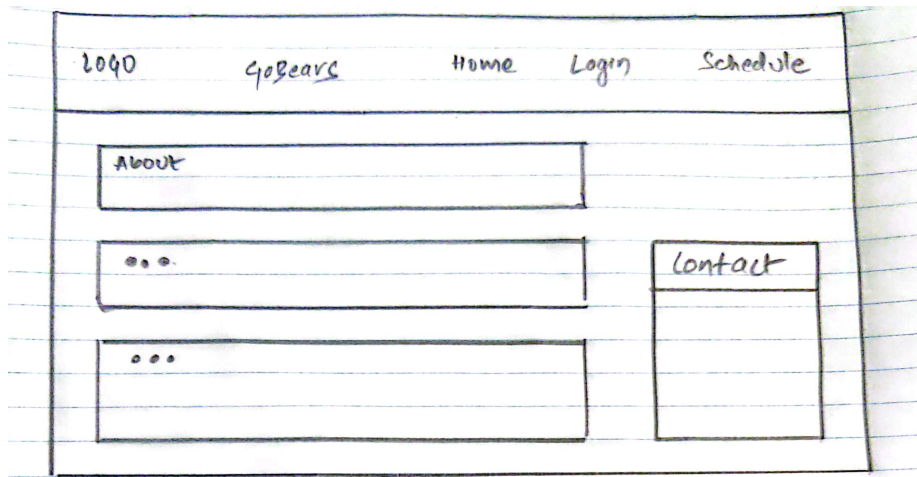
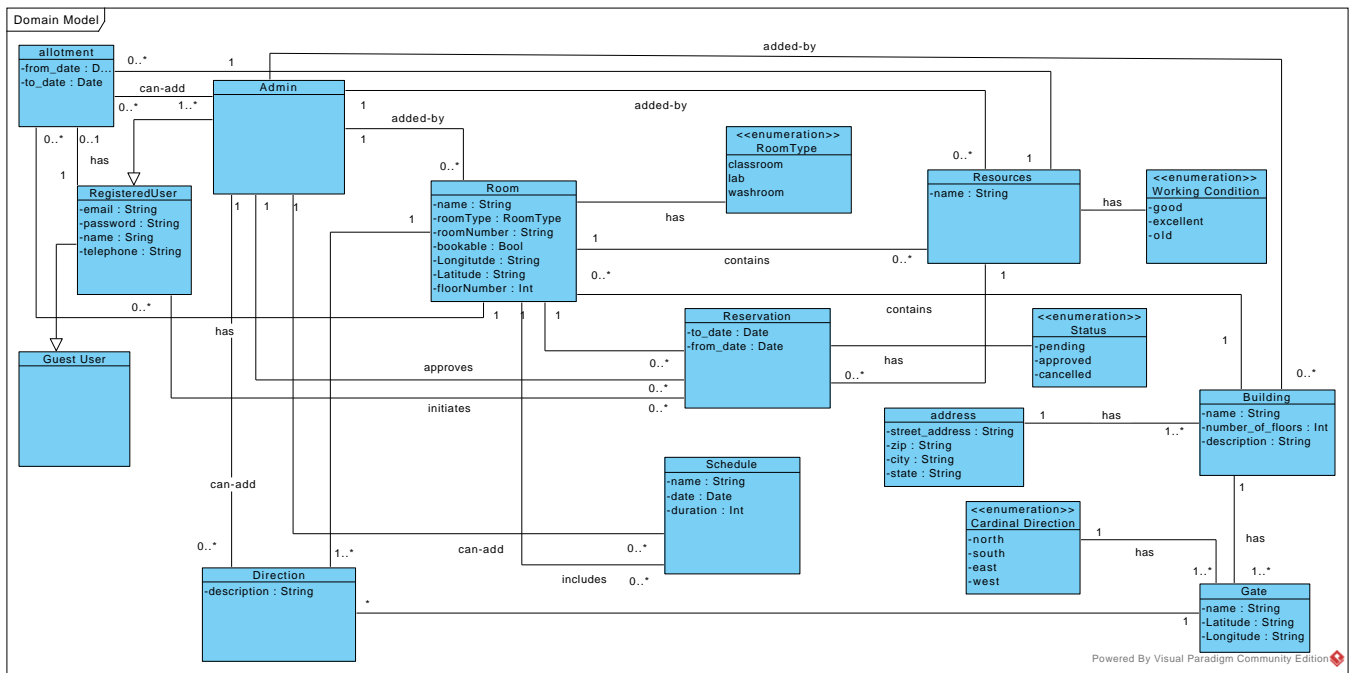


Figure 3: Landing Page Concept



## 10 Domain Model

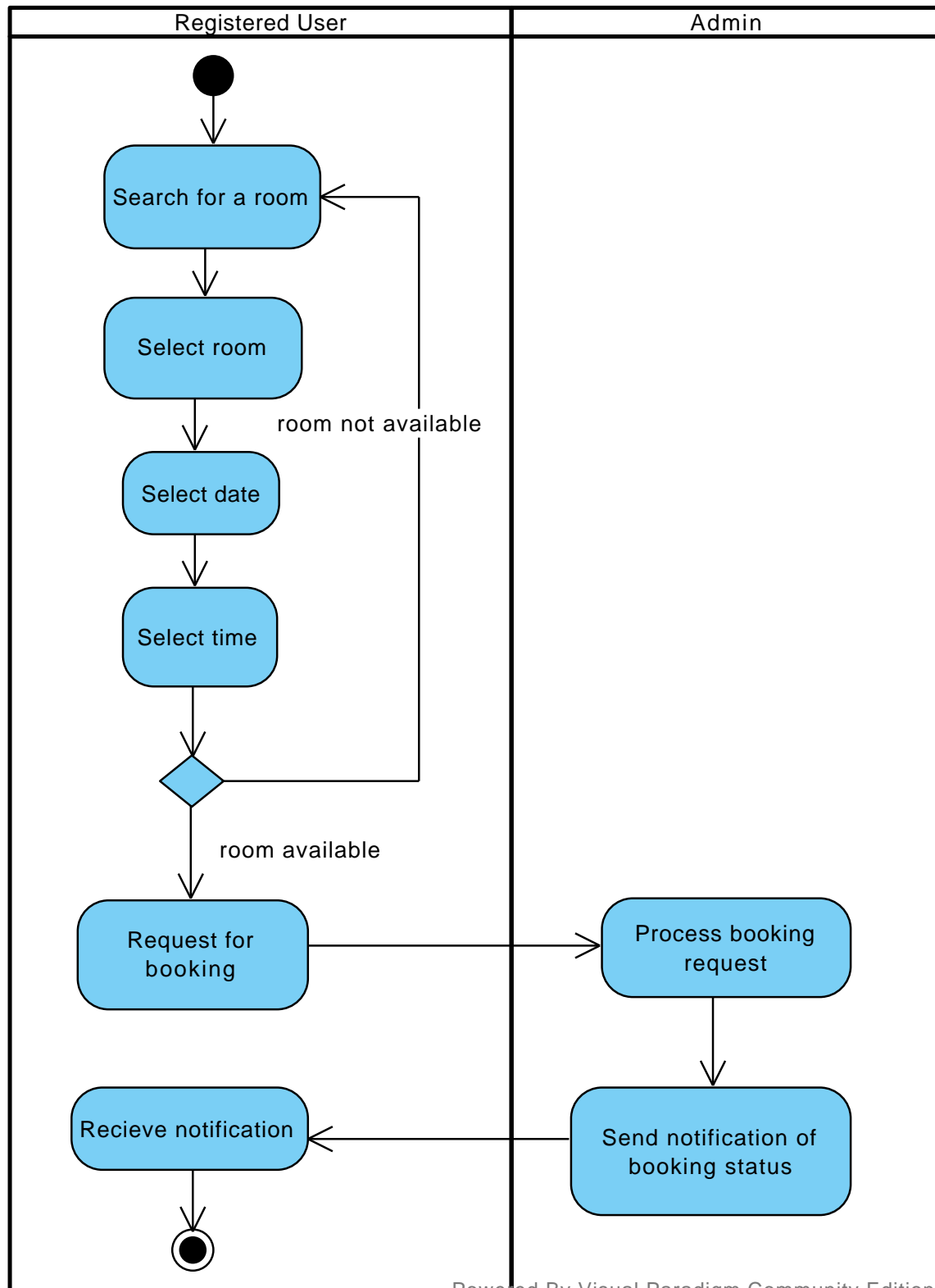
The domain model of our project is presented in the next page.

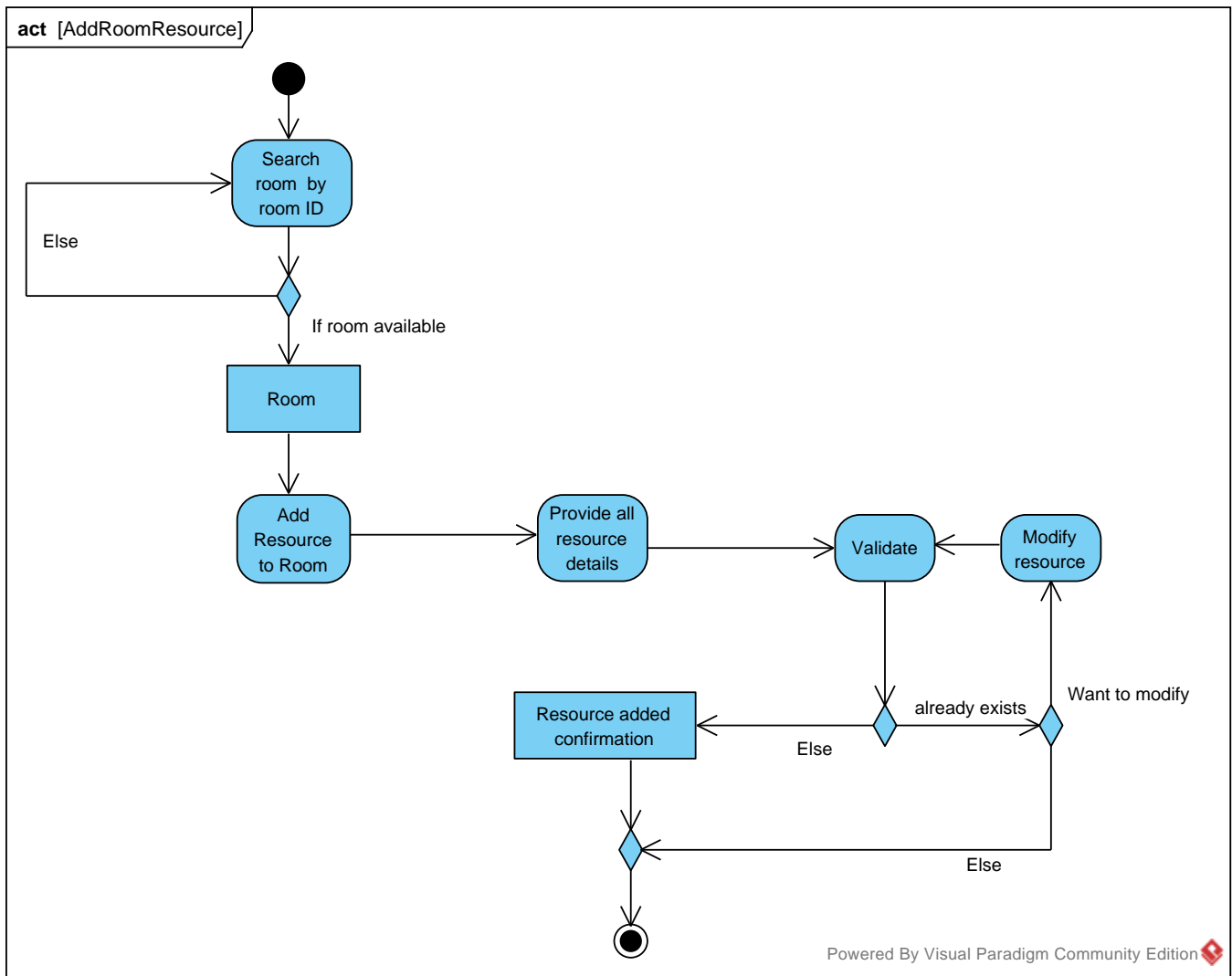


## 11 Activity Diagram

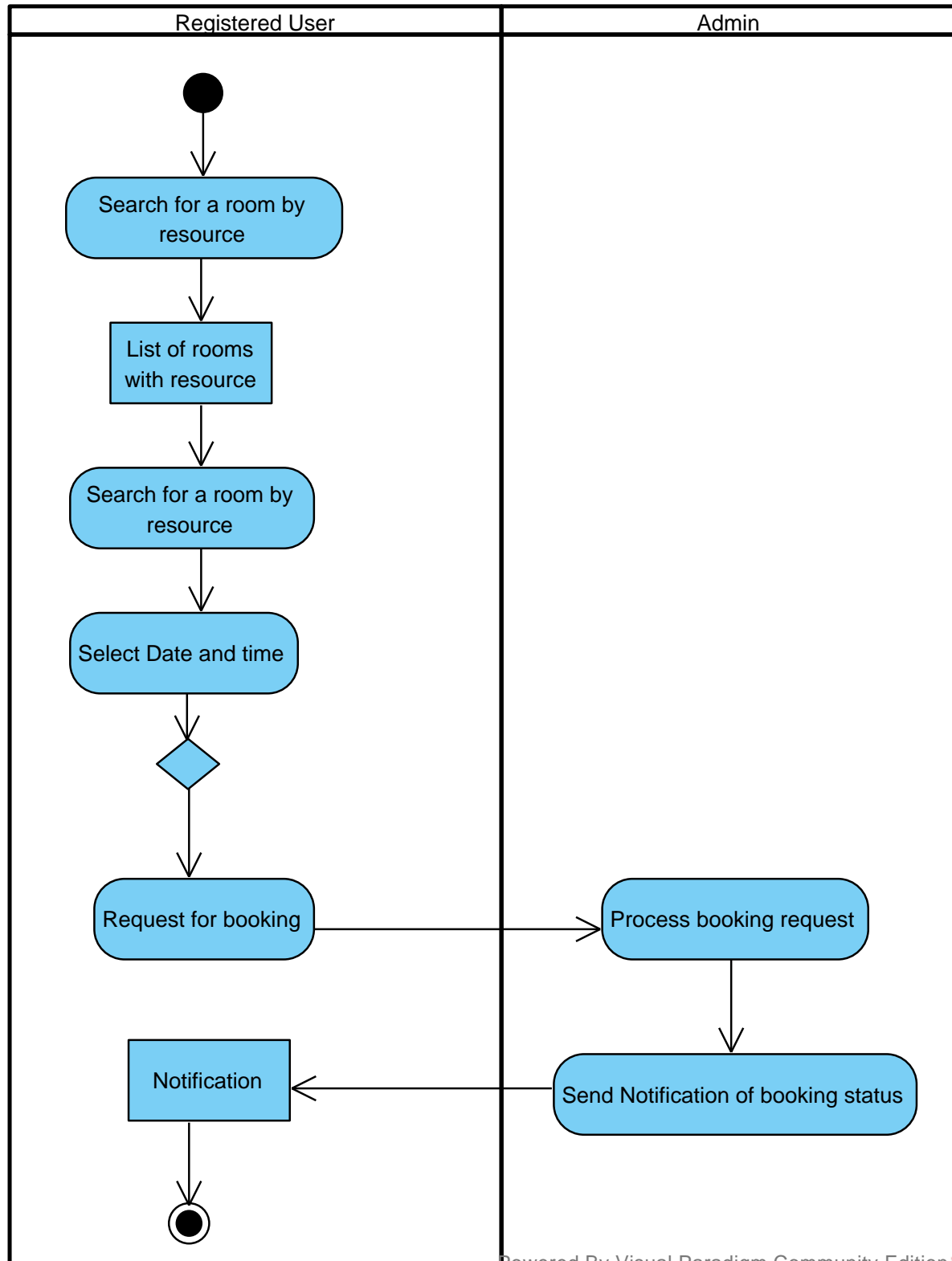
The activity diagrams are presented from the next page:

act [Reserve a room]

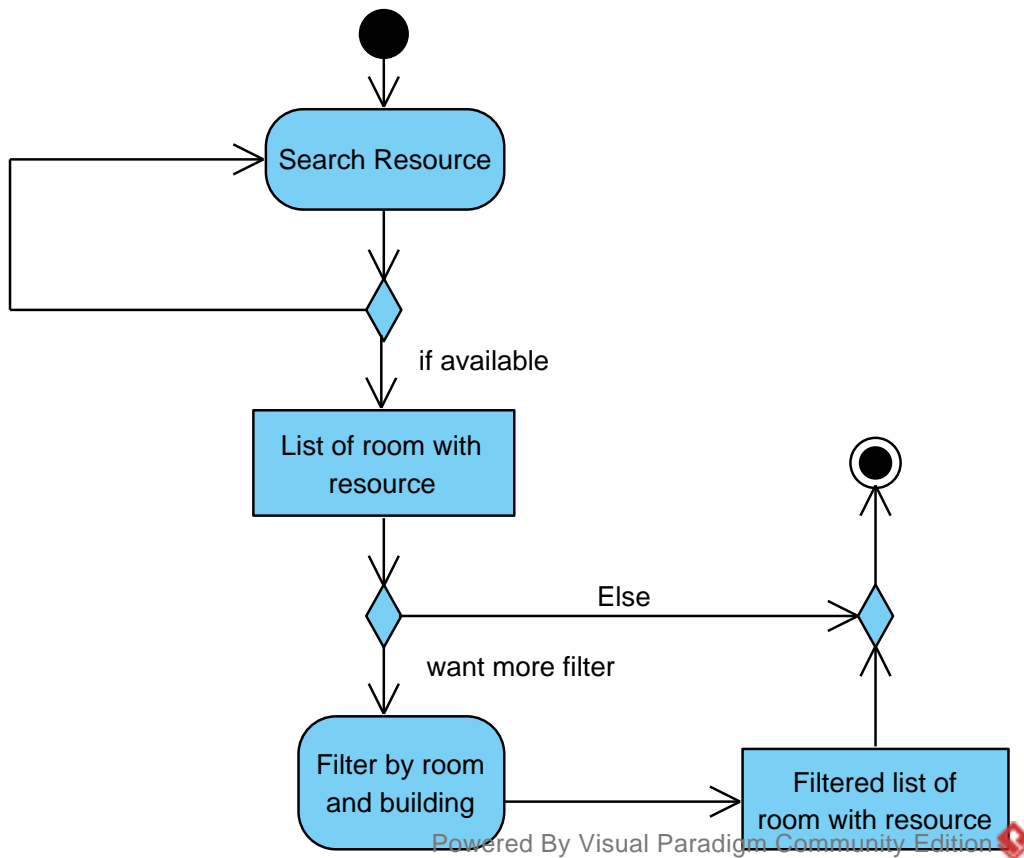




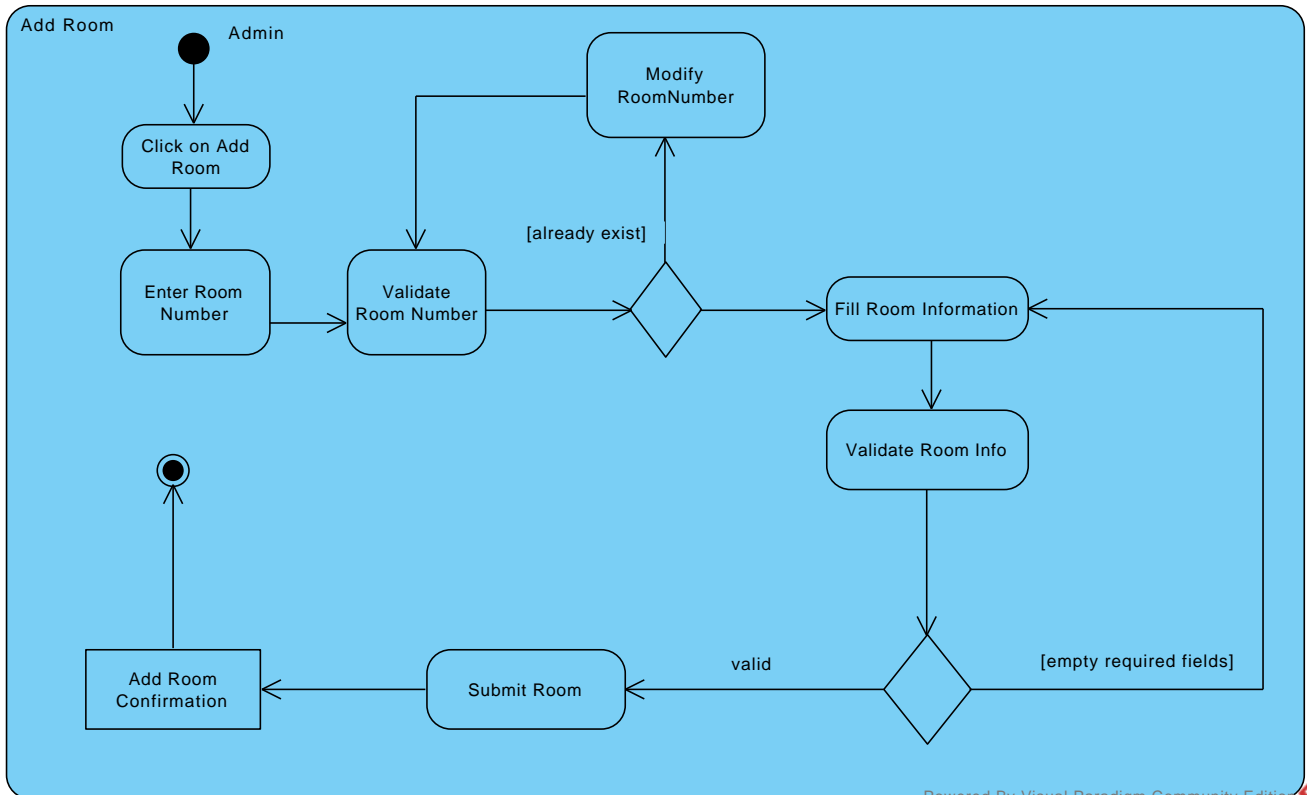
**act** [Reserve a resource]



**act** [SearchResource]

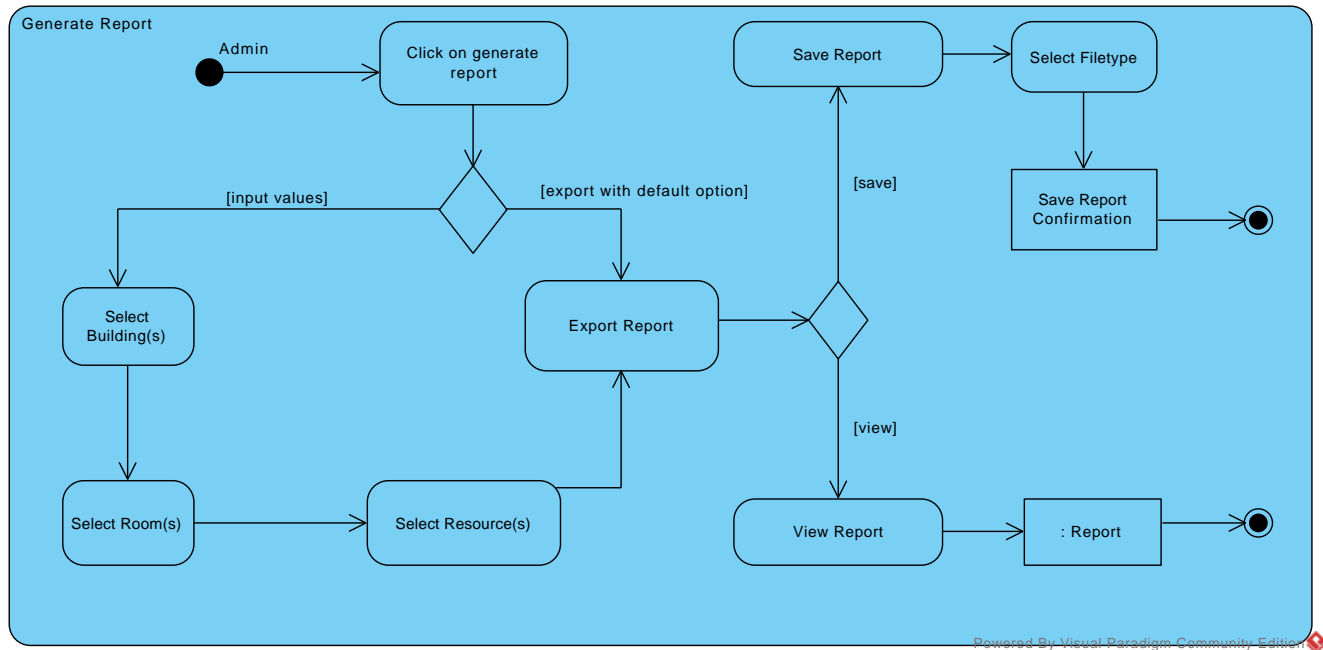


act [add\_room]

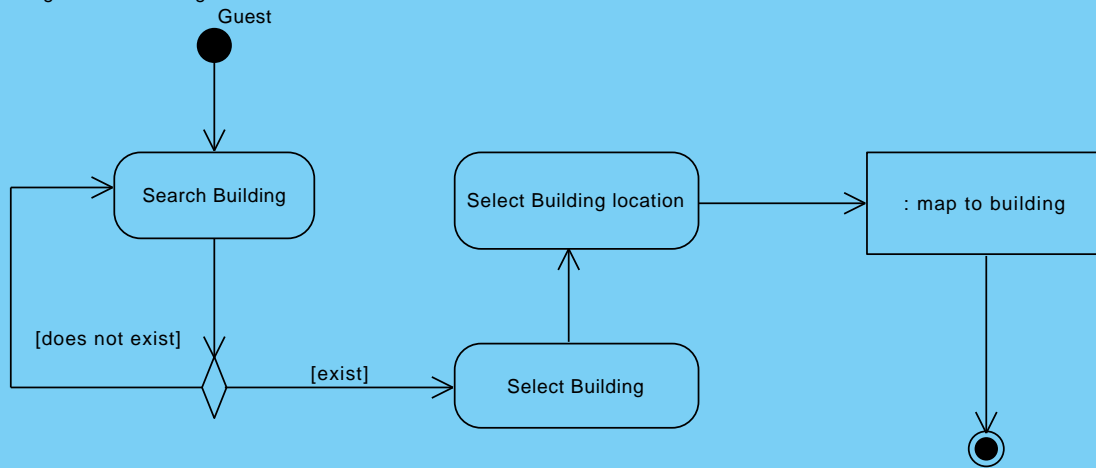




act [generate report]

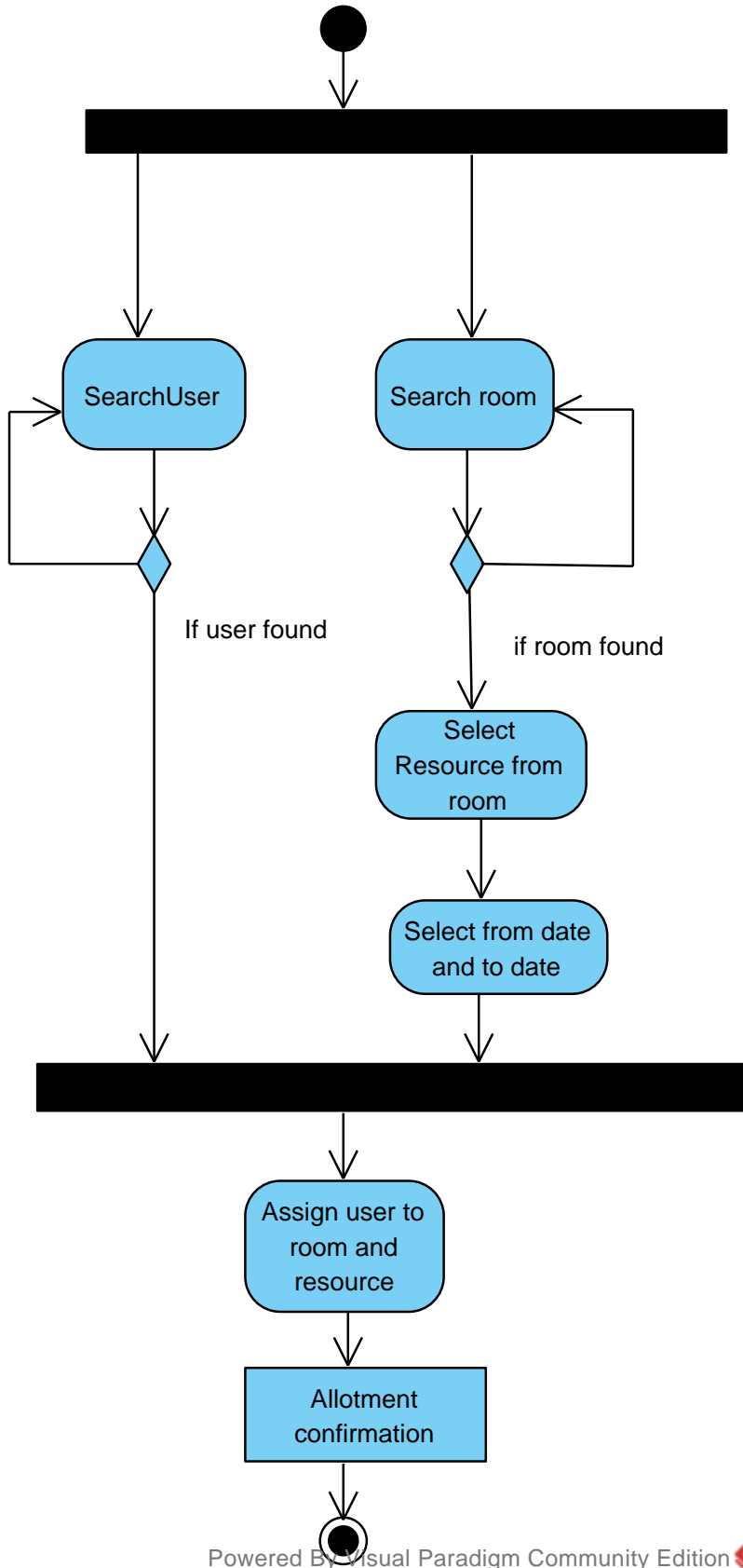


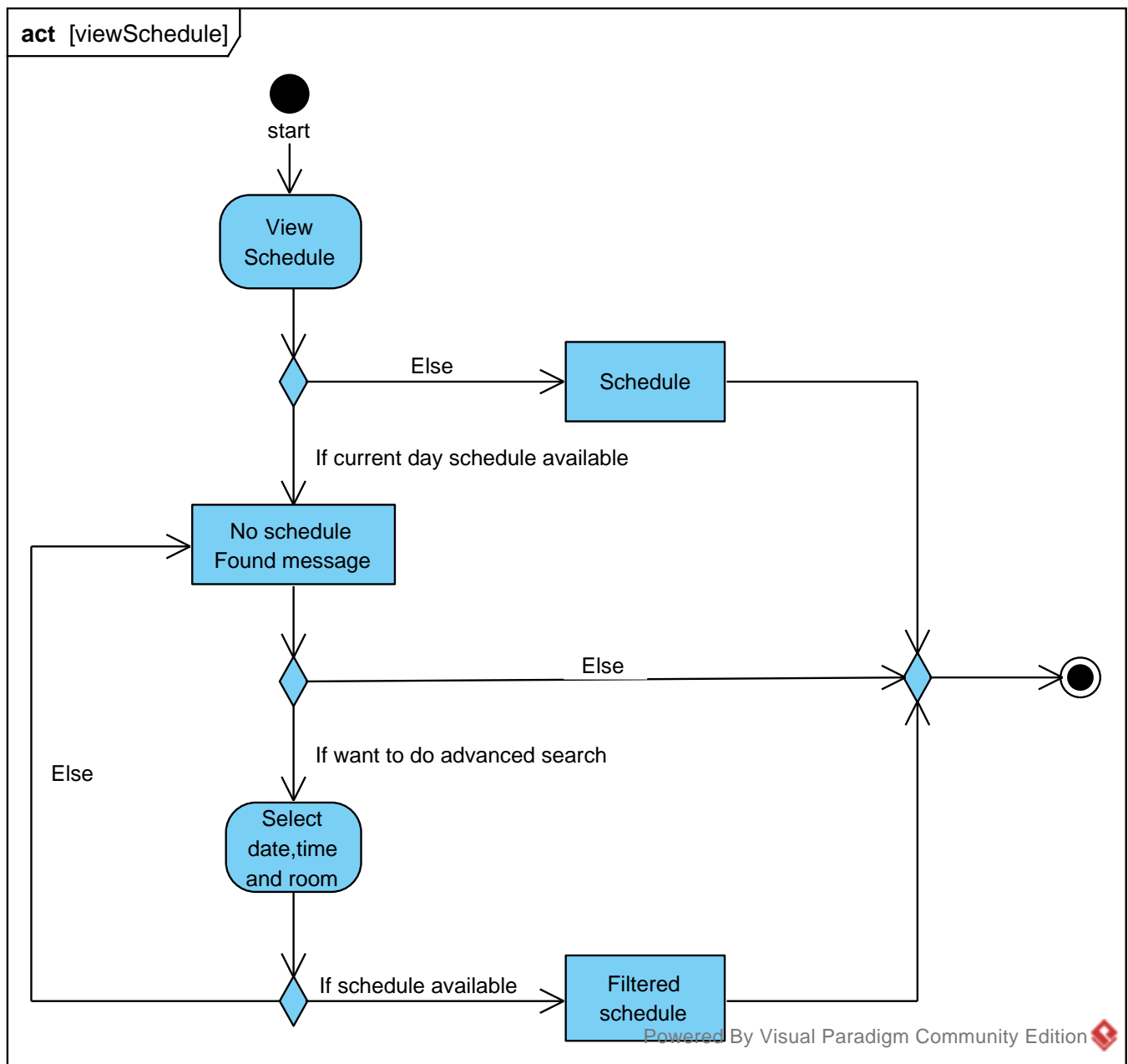
Navigate To Building



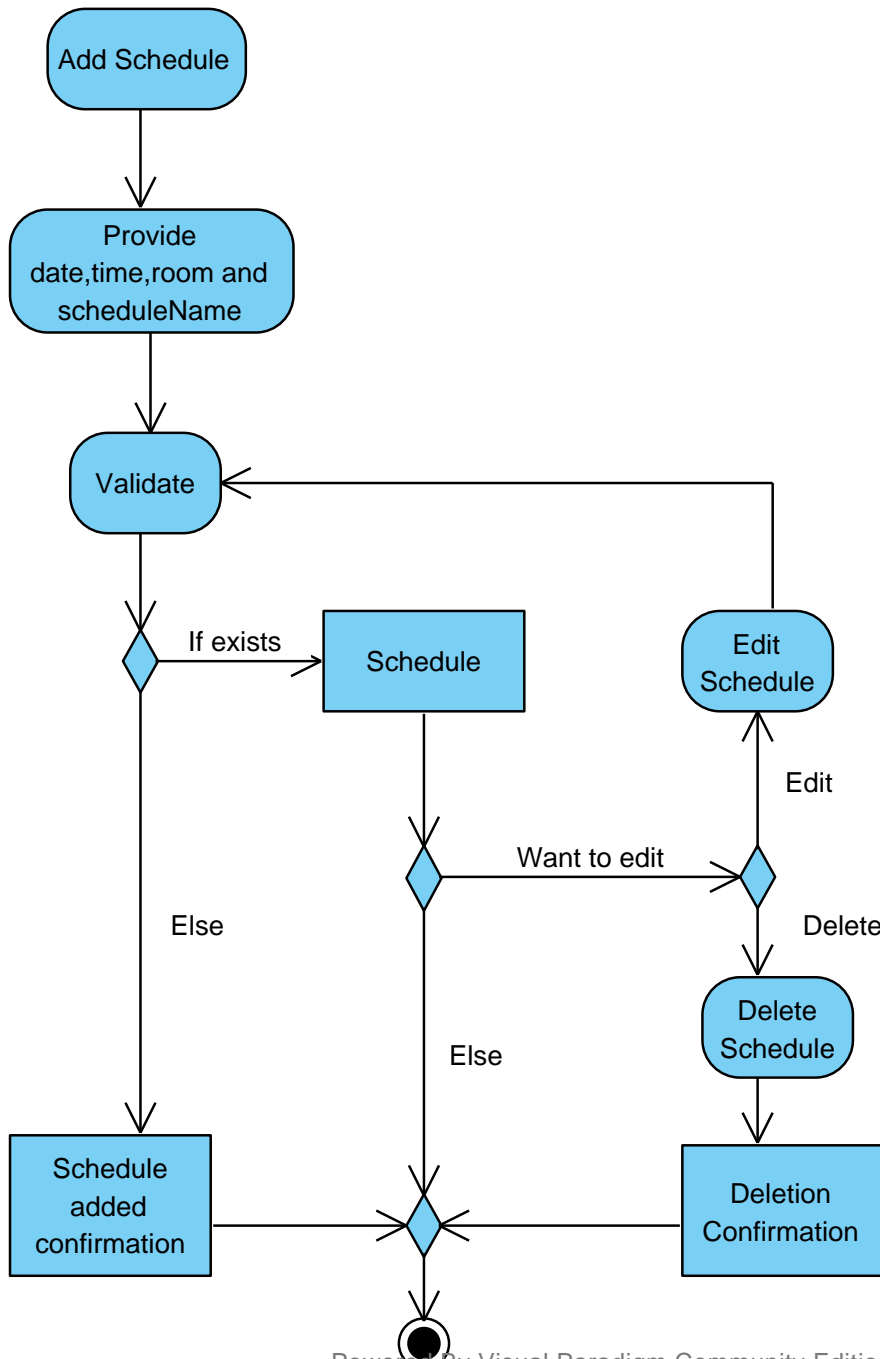
**act** [Assign room]

Admin





**act** [Add Schedule]



## 12 Gantt Chart

The gantt chart and other related artifacts are presented from the next page.

## SE Project

<http://>

Project manager

Project dates

Aug 22, 2022 - Dec 9, 2022

Completion

31%

Tasks

32

Resources

4

## Tasks

2

| Name                                     | Begin date | End date |
|--|------------|----------|
| Analysis                                 | 8/22/22    | 9/23/22  |
| Project Vision                           | 8/22/22    | 8/23/22  |
| Team Assembly                            | 8/24/22    | 8/25/22  |
| infrastructure initialization            | 8/26/22    | 8/26/22  |
| Requirements analysis                    | 8/29/22    | 8/29/22  |
| Use cases                                | 8/30/22    | 9/1/22   |
| Traceability matrix                      | 9/2/22     | 9/2/22   |
| System sequence diagrams                 | 9/5/22     | 9/7/22   |
| System operations                        | 9/8/22     | 9/8/22   |
| Wireframes                               | 9/9/22     | 9/12/22  |
| Domain model                             | 9/13/22    | 9/15/22  |
| Activity diagrams                        | 9/16/22    | 9/19/22  |
| Presentation and reporting               | 9/20/22    | 9/23/22  |
| Design                                   | 9/26/22    | 11/3/22  |
| Design model                             | 9/26/22    | 9/29/22  |
| Sequence diagrams                        | 9/30/22    | 10/4/22  |
| Package diagrams                         | 10/5/22    | 10/10/22 |
| GRASP patterns                           | 10/11/22   | 10/13/22 |
| Test coverage                            | 10/14/22   | 10/19/22 |
| Prototyping & testing                    | 10/20/22   | 10/25/22 |
| Components and deployment                | 10/26/22   | 10/31/22 |
| Presentation and reporting               | 11/1/22    | 11/3/22  |
| Implementation                           | 11/4/22    | 12/8/22  |
| backend                                  | 11/4/22    | 11/11/22 |
| User interface                           | 11/14/22   | 11/23/22 |
| User input validation                    | 11/24/22   | 11/25/22 |
| imports/exports                          | 11/28/22   | 11/29/22 |
| Unit-testing                             | 11/30/22   | 11/30/22 |
| Real data testing                        | 12/1/22    | 12/1/22  |
| system Test (with production deployment) | 12/2/22    | 12/5/22  |
| Documentation                            | 12/6/22    | 12/7/22  |
| Presentation and Reporting               | 12/8/22    | 12/8/22  |



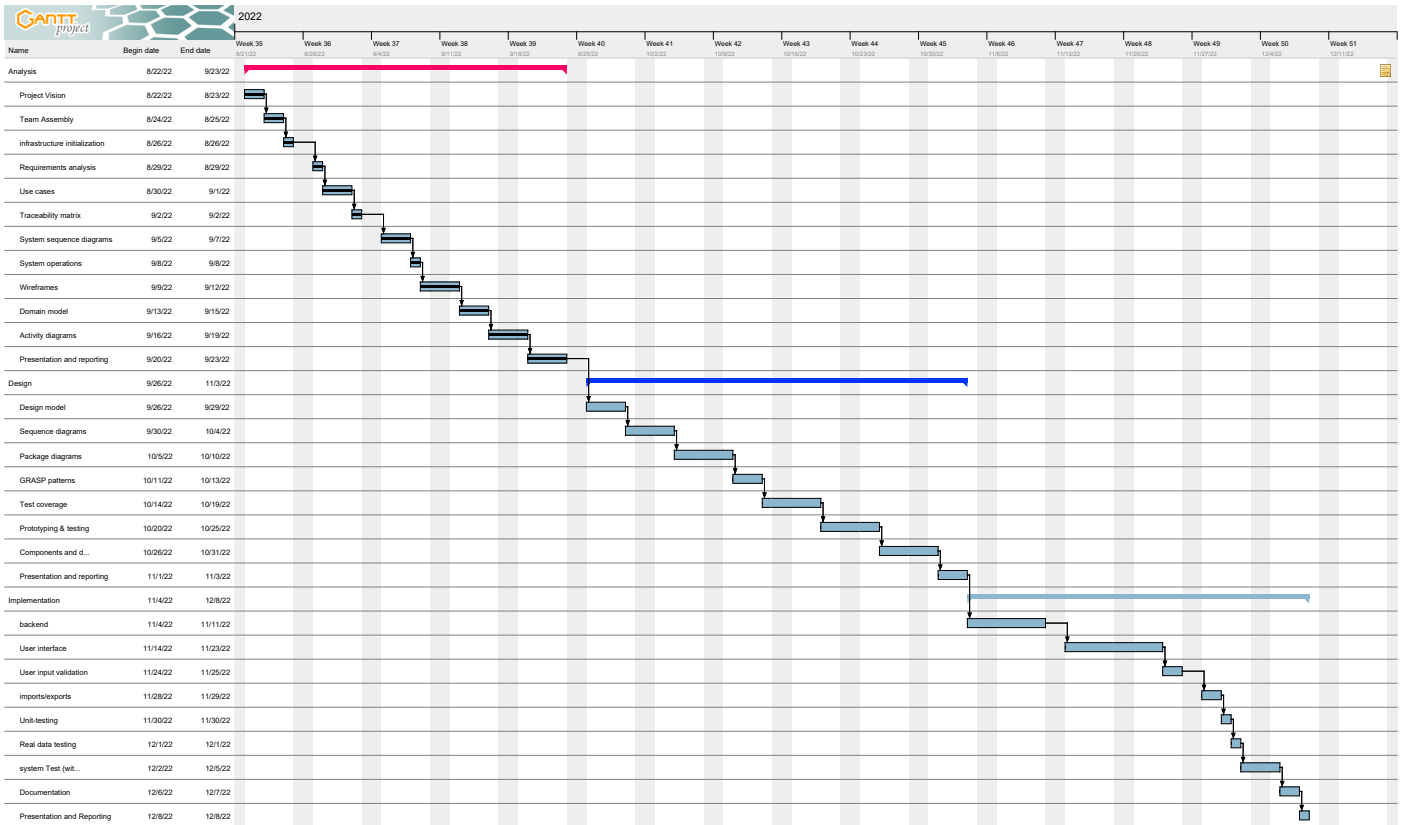
## Resources

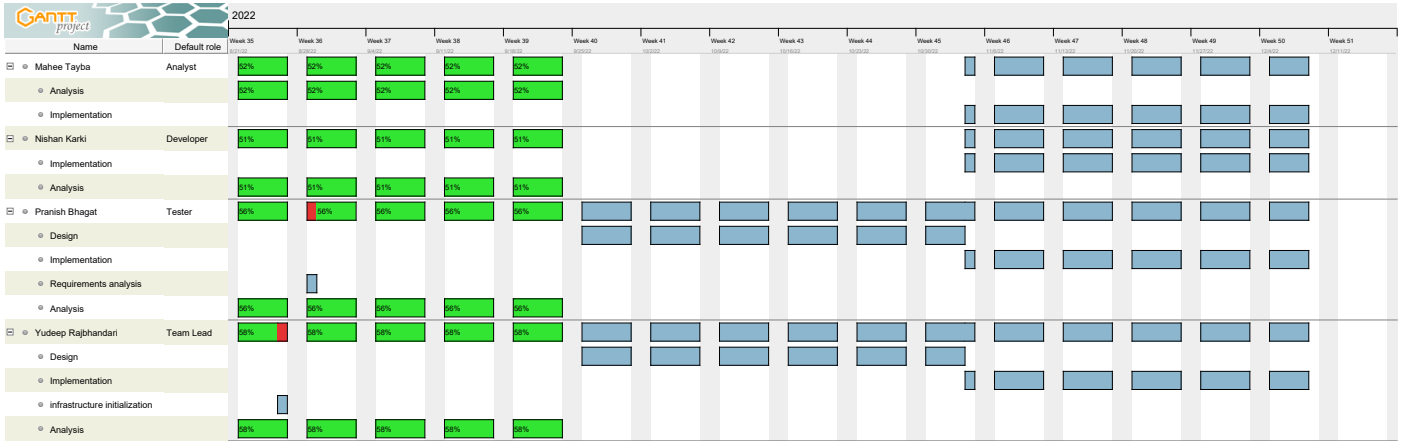
3

| Name               | Default role |
|--------------------|--------------|
| Mahee Tayba        | Analyst      |
| Nishan Karki       | Developer    |
| Pranish Bhagat     | Tester       |
| Yudeep Rajbhandari | Team Lead    |

## Gantt Chart

4





## **13 TimeCard**

The timecard for our project is presented from the next page.

| log_date  | log_name            | log_hours | card_name  |
|-----------|---------------------|-----------|--|
| 9/20/2022 | mahee_tayba1        | 1.7       | Activity diagrams                                |
| 9/20/2022 | yudeep_rajbhandari1 | 7         | Add JPA models from domain model                 |
| 9/20/2022 | nishan_karki3       | 3         | Create a entity class for resources              |
| 9/20/2022 | mahee_tayba1        | 2         | Create DB entity                                 |
| 8/30/2022 | mahee_tayba1        | 1         | Create Trello for the project                    |
| 9/20/2022 | pranish_bhagat1     | 1         | Create Trello for the project                    |
| 9/20/2022 | pranish_bhagat1     | 2         | Create wireframes for the main page              |
| 9/1/2022  | yudeep_rajbhandari1 | 2         | Create wireframes for the main page              |
| 9/20/2022 | pranish_bhagat1     | 5         | Daily Stand-Up                                   |
| 9/20/2022 | yudeep_rajbhandari1 | 3         | Database diagram auto-creation.                  |
| 9/20/2022 | yudeep_rajbhandari1 | 6         | Database initialization-postgres installation ar |
| 9/11/2022 | mahee_tayba1        | 5         | Domain model                                     |
| 9/20/2022 | nishan_karki3       | 3         | Domain model                                     |
| 9/20/2022 | pranish_bhagat1     | 7         | Domain model                                     |
| 8/24/2022 | mahee_tayba1        | 2.5       | Gantt Chart                                      |
| 9/20/2022 | pranish_bhagat1     | 2         | Gantt Chart                                      |
| 8/22/2022 | yudeep_rajbhandari1 | 4         | Gantt Chart                                      |
| 9/20/2022 | nishan_karki3       | 2         | Git and Trello link                              |
| 9/20/2022 | pranish_bhagat1     | 1         | Git and Trello link                              |
| 8/27/2022 | yudeep_rajbhandari1 | 1         | Git and Trello link                              |
| 9/20/2022 | nishan_karki3       | 3         | Git repo setup                                   |
| 8/31/2022 | yudeep_rajbhandari1 | 2         | Git repo setup-backend                           |
| 8/31/2022 | yudeep_rajbhandari1 | 5         | Infrastructure Initialization                    |
| 9/6/2022  | mahee_tayba1        | 5         | List out all the important use cases             |
| 9/20/2022 | nishan_karki3       | 7         | List out all the important use cases             |
| 9/20/2022 | pranish_bhagat1     | 5         | List out all the important use cases             |
| 9/8/2022  | yudeep_rajbhandari1 | 5         | List out all the important use cases             |
| 9/20/2022 | pranish_bhagat1     | 2         | Make a landing page for the web                  |
| 9/20/2022 | pranish_bhagat1     | 2         | Make a PageNotFound in the frontend              |
| 9/8/2022  | mahee_tayba1        | 3         | Operation Contracts                              |
| 9/14/2022 | mahee_tayba1        | 5         | Overleaf documentation                           |
| 9/20/2022 | mahee_tayba1        | 5         | Presentation                                     |
| 9/20/2022 | pranish_bhagat1     | 5         | Presentation                                     |
| 9/20/2022 | yudeep_rajbhandari1 | 6         | Presentation                                     |
| 9/20/2022 | nishan_karki3       | 5         | Project Architecture                             |
| 9/20/2022 | pranish_bhagat1     | 2         | Project Architecture                             |
| 9/20/2022 | pranish_bhagat1     | 1         | Project Documentation - First Commit             |
| 8/24/2022 | mahee_tayba1        | 2         | Project main idea.                               |
| 9/20/2022 | nishan_karki3       | 4         | Project main idea.                               |
| 9/20/2022 | pranish_bhagat1     | 3         | Project main idea.                               |
| 9/20/2022 | nishan_karki3       | 5         | Project Vision                                   |
| 9/20/2022 | pranish_bhagat1     | 2         | Project Vision                                   |

|           |                     |   |                                    |
|-----------|---------------------|---|------------------------------------|
| 8/27/2022 | mahee_tayba1        | 4 | Requirements Analysis              |
| 8/27/2022 | nishan_karki3       | 5 | Requirements Analysis              |
| 9/20/2022 | pranish_bhagat1     | 5 | Requirements Analysis              |
| 9/10/2022 | yudeep_rajbhandari1 | 6 | s/o - 4 - yudeep                   |
| 9/20/2022 | pranish_bhagat1     | 1 | s/o -1 -pranish                    |
| 9/20/2022 | nishan_karki3       | 2 | s/o -2 - nishan                    |
| 9/12/2022 | mahee_tayba1        | 4 | s/o -3 - mahee                     |
| 9/20/2022 | pranish_bhagat1     | 3 | SSD -1 -Generate Report            |
| 9/11/2022 | mahee_tayba1        | 3 | SSD -2 -Add a Building SSD         |
| 9/3/2022  | yudeep_rajbhandari1 | 5 | SSD -3 -Add schedule to the system |
| 9/20/2022 | nishan_karki3       | 3 | SSD -4 -Book a resource            |
| 9/6/2022  | mahee_tayba1        | 3 | System Operations                  |
| 9/20/2022 | nishan_karki3       | 5 | System Operations                  |
| 9/20/2022 | pranish_bhagat1     | 1 | System Operations                  |
| 9/3/2022  | yudeep_rajbhandari1 | 1 | System Operations                  |
| 9/20/2022 | pranish_bhagat1     | 1 | Team Assembly                      |
| 9/20/2022 | pranish_bhagat1     | 5 | Use Case - 1 - Pranish             |
| 8/27/2022 | yudeep_rajbhandari1 | 5 | Use Case -2 - Yudeep               |
| 9/20/2022 | nishan_karki3       | 3 | Use Case -3 -Nishan                |
| 9/9/2022  | mahee_tayba1        | 4 | Use Cse -4 -Mahee                  |