GoBears
Iteration -1

Mahee Tayba
Nishan Karki
Pranish Bhagat
Yudeep Rajbhandari

September 3, 2022

Contents

1 Problem Statement

The objective of the 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. Booking room: Enable users to book an existing room of a building by searching and finding the most relevant option.
2. Booking resource: Enable users to book a resource from a room of a specific building.
3. Navigation: Allow the users to find directions to a room of a building from the building gate.
4. 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:

Use case: Add building

ID: UC1

Actors: Admin

Precondition:

1. Actor needs to be logged in to the system with the role as admin.

The flow of Events:

1. Actor selects add a building.
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.
3. The actor saves the building information with a building name.
    a. If the actor tries to put a name the same as an existing one.
        i. The system shows the user a message specifying that the name has to be different from an already existing name.
    b. If the actor tries to give an empty name.
        i. The system denies the creation of an empty building name and shows the name can not be an empty message.
4. The system creates a new building with the name introduced and the creation date.

PostConditions:

Alternate flow:

Use Case	*Add room.*
ID	UC11
Description	This describes the process of adding a new room to a building in the system.
Notes	1. No restrictions on the number of rooms that can be on the system. 2. Only Admin can add a new room to the system.
Actors	Admin
Pre-Conditions	1. The Admin is logged into the system.
Main Flow	Create a New Room 1. The Admin selects an option to add a new room to the system. 2. The system prompts the Admin for: a. the following required information: i. Associated building name/number. ii. Room number. iii. Allowed to book the room or not (Bookable?) b. and the following non-required information: i. Room Name if any. ii. Room Type iii. Co-ordinates of the room. iv. Special info about the room if any. v. Assigned to (For example: Professor's office, etc.) 3. The Admin enters these parameters and submits. 4. If any of the required fields are not completed, the admin is informed which required fields are missing a. Input focus is returned to the first missing field 5. If the roomnumber is already in use, the system informs the Administrator that they must choose a different room number. 6. Else, continue 7. If the roomnumber is invalid, the Administrator is informed of this, and valid naming criteria is displayed 8. The system creates the new room.
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	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 ocurred when the error dialog is discarded. 2. RoomNumber are unique in the database. 3. RoomNumber and RoomName are not case sensitive. 4. A room cannot be named "UNKNOWN ROOM"

Use case: Assign a room
ID-UC3
Actors: Admins
PreCondition: Actors have to be logged with role admin.
Flow of Events: 1. Admin searches for the desired user. a. If the user exists then the system returns the user i. Admin searches for a room. 1. If a room exists then the system returns the room. a. Admin looks up the resources available for the room. b. The system returns all available resources in the room. c. Admin selects the resource then the system gives the option to choose from date and duration. d. Admin assigns the room and the resource to the user for the given duration. e. The system returns the assignment added message. 2. Else a. No room found message is displayed. ii. Else 1. No user-found message is displayed
Post Condition: The schedule will be added or modified in the database.
Alternate flow:
Extensions:

Use case: Reserve a room

Actors: Registered User

Precondition:

1. User needs to be logged in to the system.
2. The room should be bookable
3. Constraints on dates will be applied as described by admin.

The flow of Events:

1. Registered user selects a room.
2. System displays information about the room (roomType, bookable).
3. Registered user selects the Reserve room button.
    a. Availability calendar is displayed and the user selects the dates from them.
    b. On submission of valid date,
        i. The reservation request will be added to the system.
        ii. The system will forward the request to the admin and he will process the request.
        iii. Approval/Decline status will be sent back to the user.
    c. For inavlid date, the user will be redirected to select room.

PostConditions:

Alternate flow:

1. Users can go back to the room without booking the room.

Use case: Reserve and Checkout room resources

Actors: Students

Precondition:
Actor needs to be logged-in.
DB should be set up with resources information.
Actor selects a room.
Actor selects a resource.

Flow of Events (Book and Checkout room resources):

1. Actor selects a room resource.
2. System displays information about all the room resource (Can be checked out, can be booked, availability, book schedules, checkout schedules, etc)
3. Actor selects Book resources button (Active if the resource can be booked).
    a. A new form to be filled will be displayed. The form will have availability calendar for the dates and times. An actor can only choose available dates and time. Constraints on dates will be applied as described by admin.
    b. On selection of valid dates and the click of submit button, the schedule will be added to the system.
    c. The system will forward the request to the admin.
    d. Admin will approve/decline the request.
    e. Approval/Decline status will be sent back to the user.
4. Actor selects *Checkout* resources button (Active if the resource can be checked out).
    a. Actor choose available dates and time. Constraint on dates will be applied as described by admin.
    b. On selection of valid dates and the click of submit button, the checkout schedule will be added to the system.
    c. System displays a success/failure message.

PostConditions:

Alternate flow:

Use case: Insert new schedule and Update the available schedules
ID-UC2
Actors: Admin
PreCondition: Actors have to be logged with role admin.
Flow of Events: 2. User selects Add schedules. a. User provides inputs for a date, time, and room. i. If a schedule exists for the selected options: 1. System gets the schedule and displays it as well as provides an option to update or delete the schedule. 2. If the user selects delete, then the schedule is deleted. 3. If the user modifies the schedule and selects update, then the schedule is updated. ii. Else 1. User inputs the desired schedule. 2. User selects Add.
Post Condition: The schedule will be added or modified in the database.
Alternate flow:
Extensions:

Use case: Filter and view the available schedule

ID-UC1

Actors: Registered / Guest user

Precondition:

1. Schedule should be available in DB

Flow of Events:

1. Actors select view schedules.
2. System searches for all the schedules of all rooms for the current day and if the system finds any schedules for any room:
    a. The schedule is displayed.
3. Else
    a. Shows a message stating “No schedule available” and a button “Filter Schedules”
        i. If the user clicks on the filter schedules
            1. System asks for a date, then time, and then rooms.
            2. System searches for schedule with corresponding date, time, and room.
                a. If the system finds the schedule
                    i. It displays on the screen.
                b. Else
                    i. Show the message that no schedule is available with an option to go back to the previous page.

PostConditions:

Alternate flow:

Use case: View the reservation status of rooms in a building.

Level:

Actors: Admin

Precondition:

1. Actor needs to be logged in to the system with the role as admin.
2. At least one building information is saved in the system.

The flow of Events:

1. Admin selects a building.
2. System provides the building information. the list of rooms that the building has.
3. If the admin wants to see room status,
    a. Admin selects display room status.
    b. System provides the list of booked, pending and unbooked rooms sorted by recent date.
    c. Admin may have an empty list.
        i. System shows “0 result”.
4. If the admin wants to update a room status,
    a. Admin selects a room.
    b. System provides room information and reservation requests.
    c. Admin checks all the constraints for the reservation request.
    d. If the requirements doesn’t meet
        i. Admin declines the request.
    e. Admin approves the request.

PostConditions:

Alternate flow:

Use case: View, Search and Filter room resources

Actors: Students/Visitors/Admin

Precondition:
The actor is logged-in.
DB should be set up with resource information.

The flow of Events (Search, filter and view room resources):

1. On the search field actor searches for a room by resource (Projector, lab, seating space, etc).
2. System displays all the rooms that have searched resources.
    a. If the actor selects a room.
        i. System displays all the room and resource information (Different sections) available for the room.
    b. If the actor clicks on the *Filter* button.
        i. Show filter options populated from the resource information (Add resources) form.
        ii. Apply filter on the search depending on the selected filters.
        iii. Display new list of room results.
            1. If actor selects a room.
            2. System displays all the room and resources information (Different sections) available for the room.

PostConditions:

Alternate flow:

Use case: Update, Add and Remove room resources

Actors: Admin

Precondition:

1. Actor is logged-in.
2. DB is set up with resource information.

Flow of Events (Update, Add and Remove room resources):

1. Actor selects a room.
2. System displays information about all the resources available for the room (Resource name, number available, resource category(lab, computer lab, auditorium, etc) in the Resources section.
3. Actor selects *Add* Resources button.
    a. 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.
    b. On *submit,* resource is added to the system.
    c. System displays a success/failure message.
4. Actor selects *Remove* Resources button (Available next to each resource).
    a. System checks if the resources has active book or check-out records.
        i. On click, if there are no active checkout or booking records for the resources, the resource is removed from the system.
        ii. System displays a success/failure message.
5. Actor selects *Update* Resources button.
    a. A form with all available fields with existing information for the resources will be displayed. Actor can update resource information (name, number available, can be checked out, can be booked, constrain on booking dates and time, constraint on length of checkout times, resource category, etc).
    b. On *submit* the updated resource information is added to the system.
    c. System displays a success/failure message.

PostConditions:

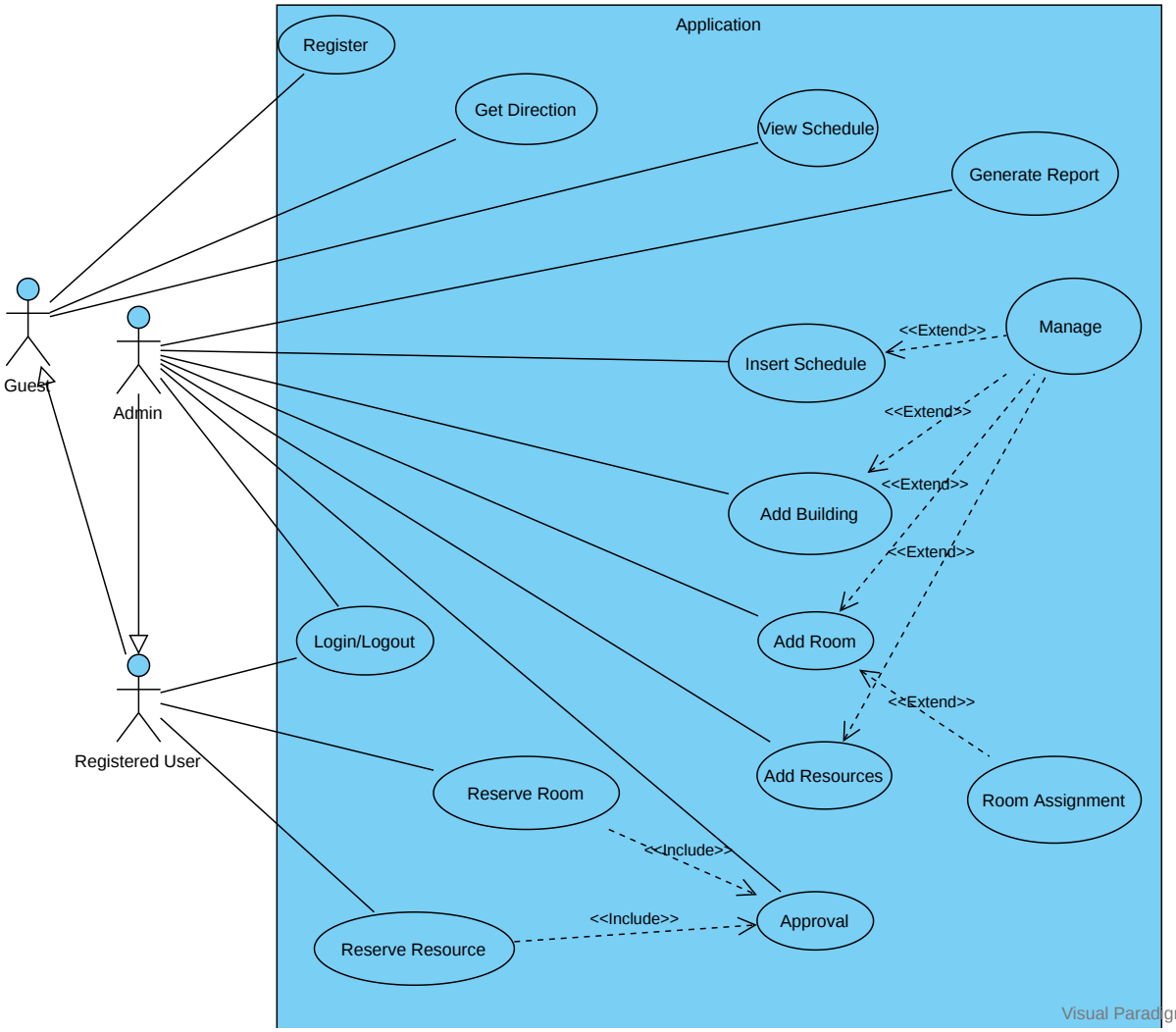
Alternate flow:

Use Case	*Generate a comprehensive report*
ID	UC10
Description	This occurs when the admin wants to generate a report of the system.
Primary Actors	Admin
Preconditions	1. The report option is configured in the system. 2. An admin exists in the system.
Main Flow	1. The admin selects an option to generate a report. 2. The system provides an option to select the range of dates. 3. The admin selects the range of dates. 4. The system provides an option to select the list of building(s). 5. The admin selects the required building(s). 6. The system provides the associated list of room(s). 7. The admin selects the required room(s). 8. The system provides associated list of resource(s). 9. The admin selects the required resource(s). 10. The admin specifies the format of the generated report. 11. The admin requests to generate the report. 12. The system generates the report and requests the user to specify whether to display or save the report. 13. The admin specifies a location to save the report. 14. The system saves the report to the specified location.
Post Conditions	
Alternative Flows	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. 2. The user does not specify how the report is delivered. The system uses the default report delivery mechanism. 3. The user does not specify the format of the generated report. The system uses the default report format. 4. The user cancels the report generation option. 5. The user requests to display the report. 6. The system displays the report.
Post Conditions	
Success end condition	1. The report is generated and saved to a specified location. 2. The report is generated and displayed on the screen.
Failed end condition	The report is not generated.

Use Case	*Navigate to Building*
ID	UC12
Description	This describes the process to navigate to a building.
Actors	Guest User.
Pre-Conditions	1. The required building should be in our system. 2. The user should allow location in their device.
Main Flow	Navigate to building. 1. Guest searches for a building. 2. If building exists, the system will display the building's location along with associated info. 3. Else, the system displays building info not available. 4. The actor selects the option to navigate to the location. 5. The system will provide an interface with the direction from the actor to the building.
Post-Conditions	

5 Use Case Diagram

The UCD of our project is attached below.

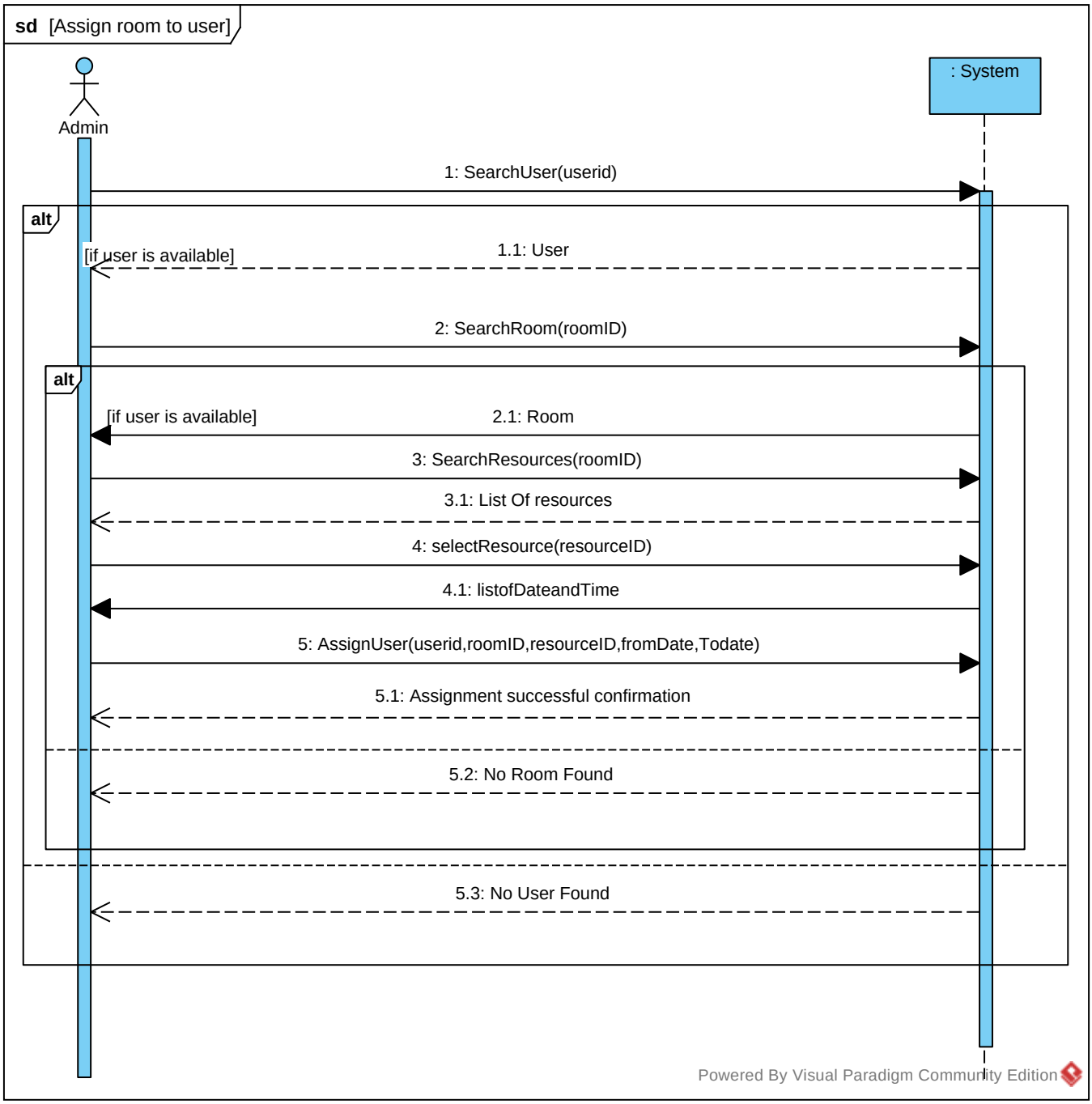


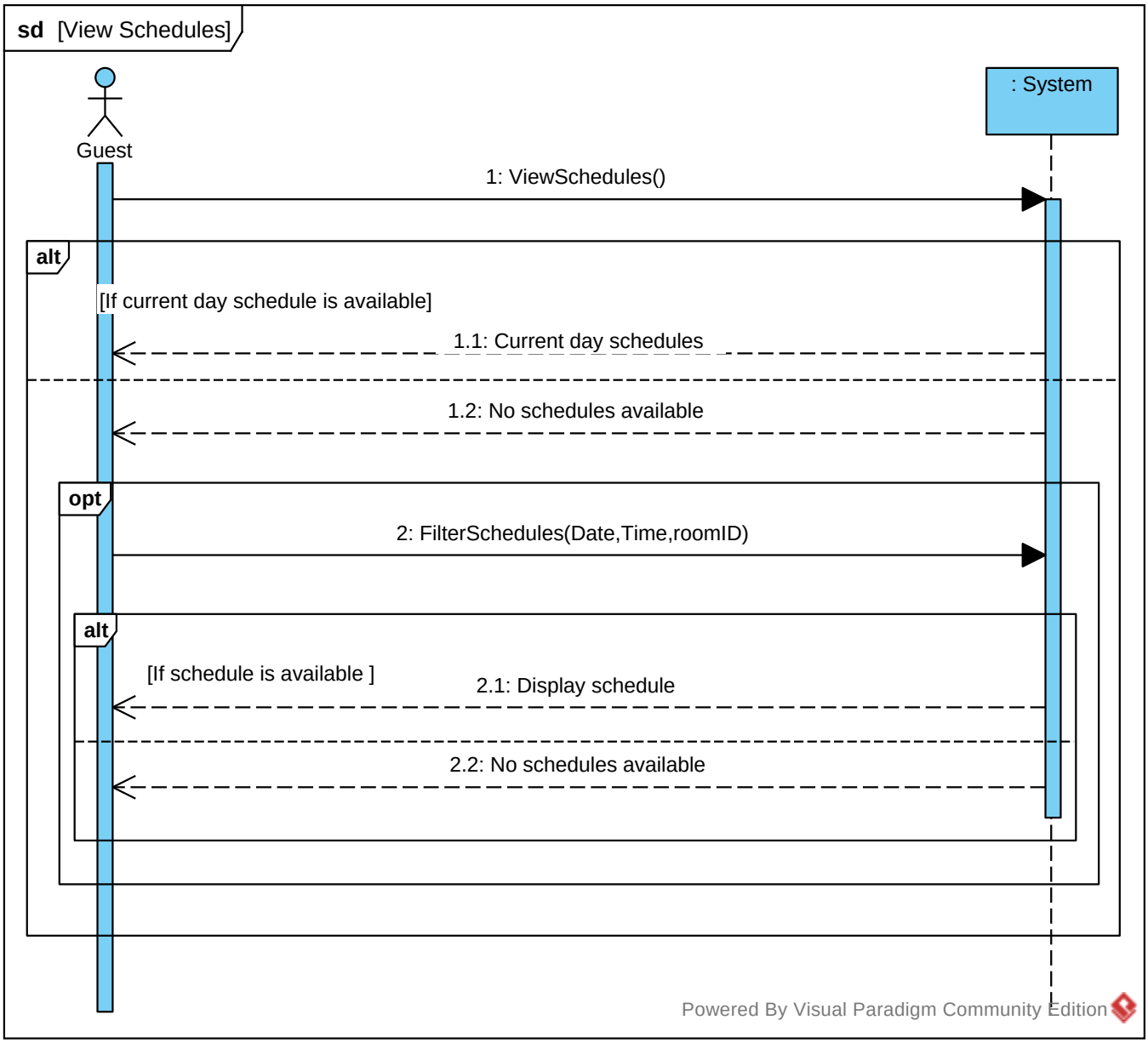
6 System Sequence Diagrams

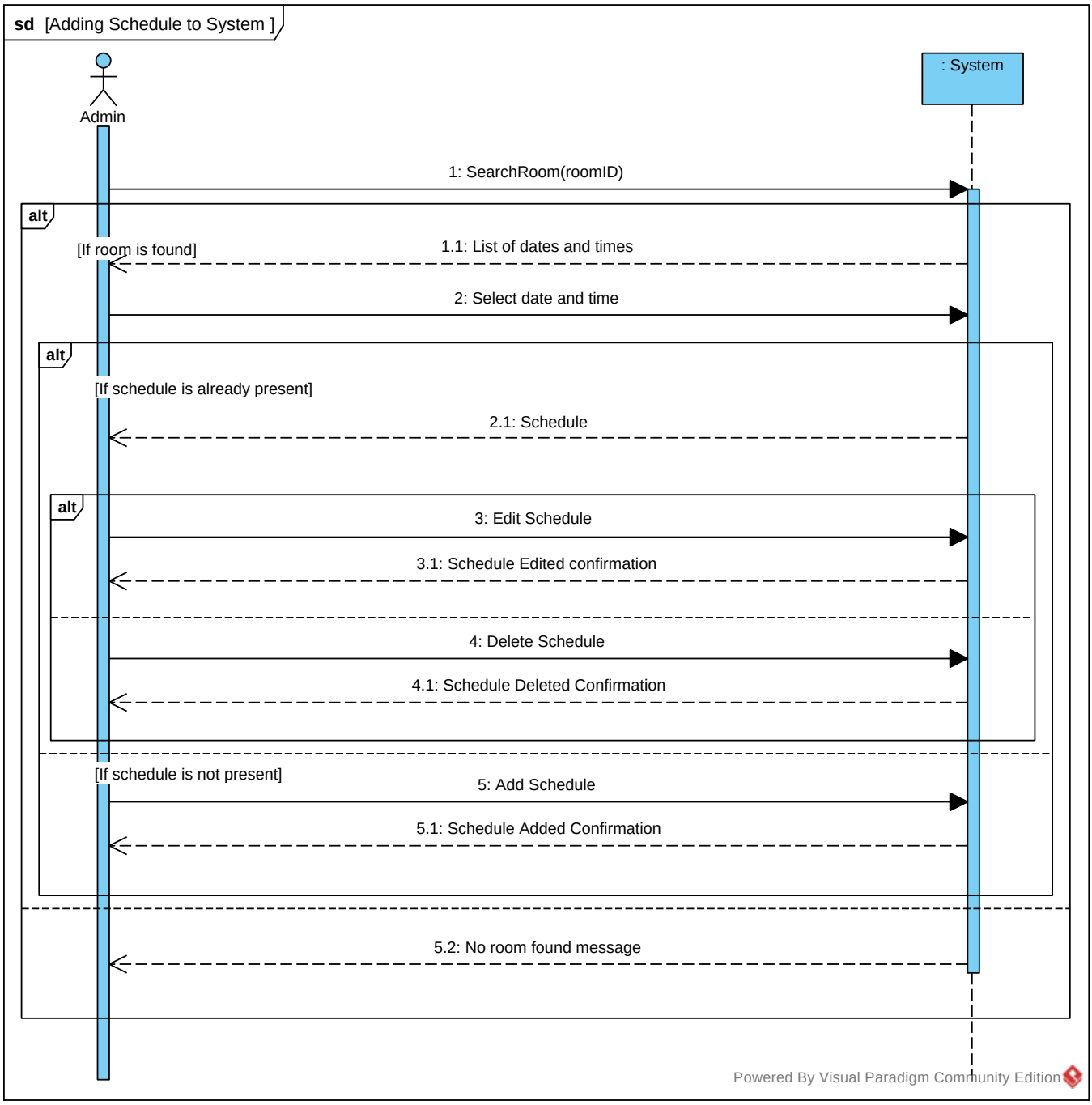
Please find the list of all the SSDs below:

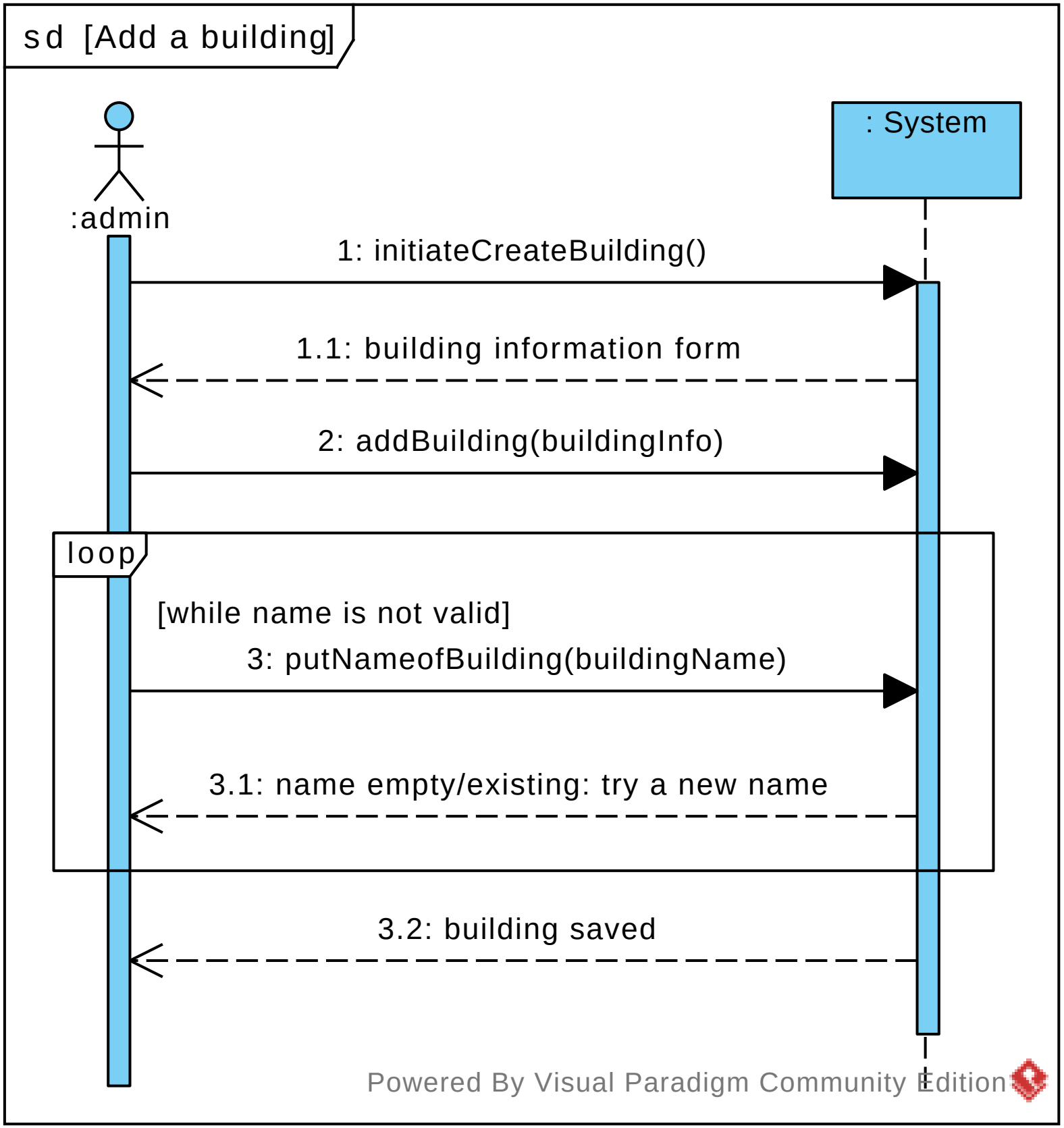
1. Assign room to registered user.
2. Add schedule.
3. View schedule.
4. Add building.
5. Reserve room.
6. Update room status.
7. Add resource.
8. Reserve resource.
9. Search resource.
10. Navigate to building.
11. Add Room.
12. Generate a comprehensive report.

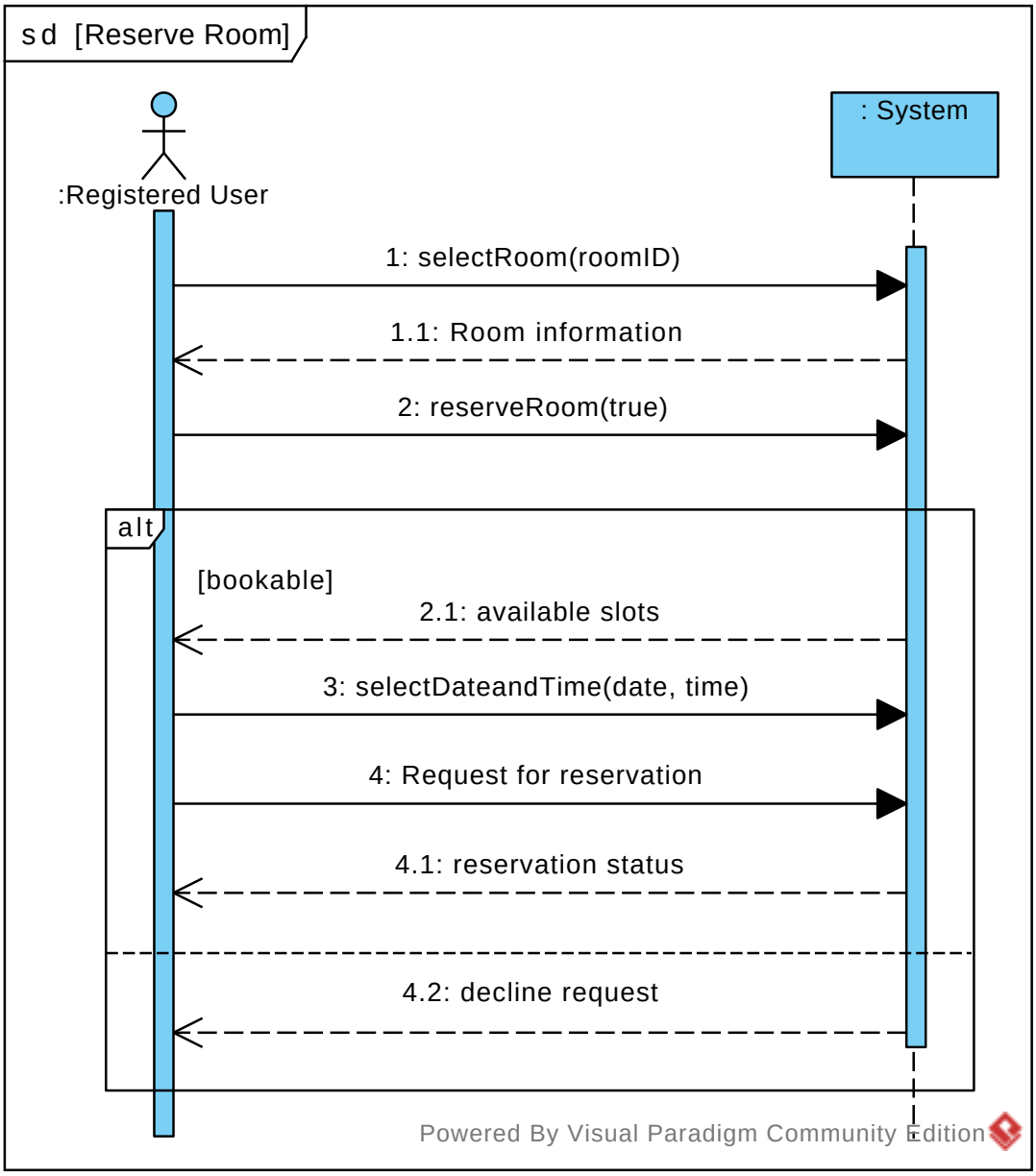
The SSDs are presented from the next page.

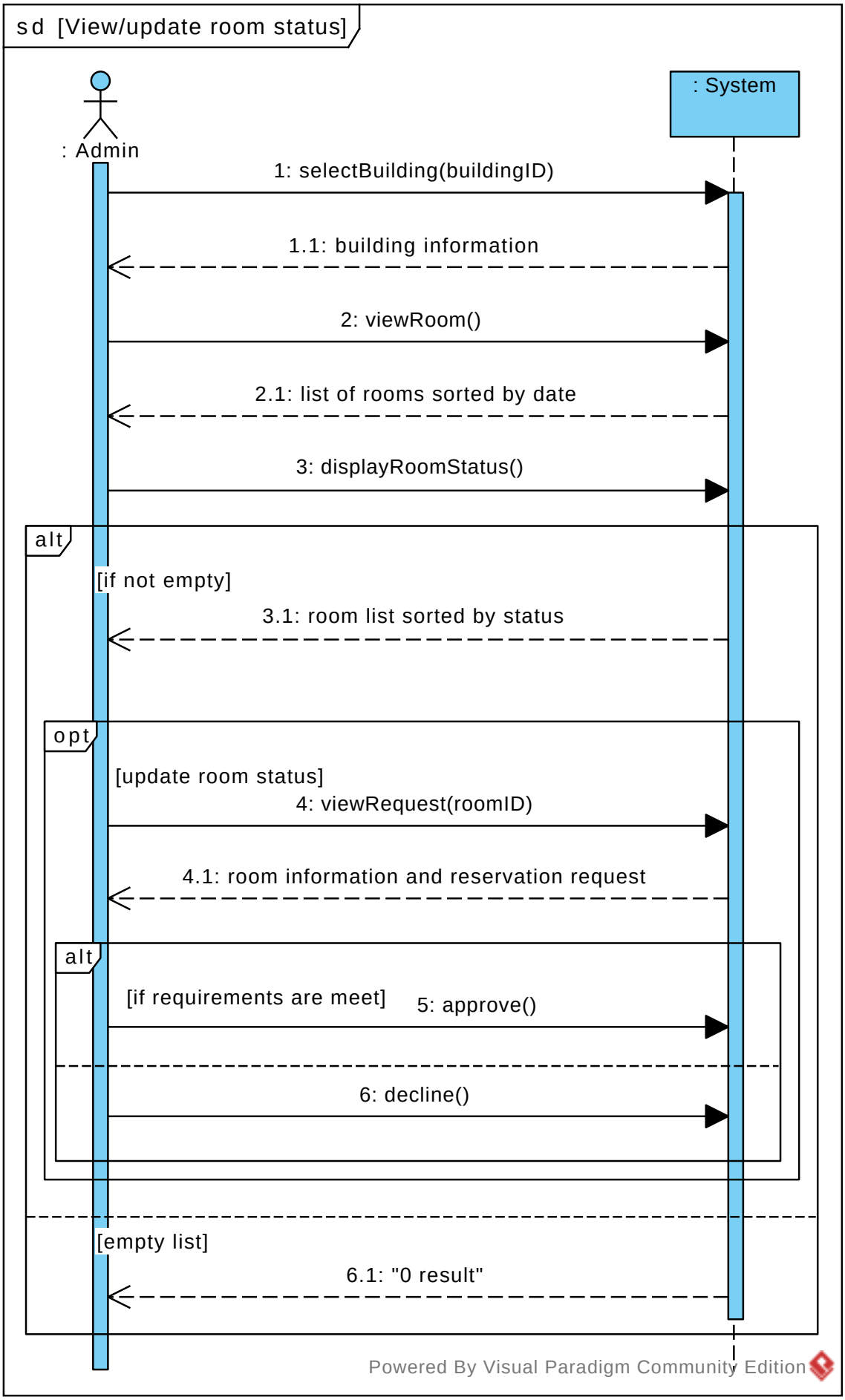

sd [Assign room to user]
Admin
: System
1: SearchUser(userid)
alt
[if user is available]
1.1: User
2: SearchRoom(roomID)
alt
[if user is available]
2.1: Room
3: SearchResources(roomID)
3.1: List Of resources
4: selectResource(resourceID)
4.1: listofDateandTime
5: AssignUser(userid,roomID,resourceID,fromDate,Todate)
5.1: Assignment successful confirmation
5.2: No Room Found
5.3: No User Found
Powered By Visual Paradigm Community Edition

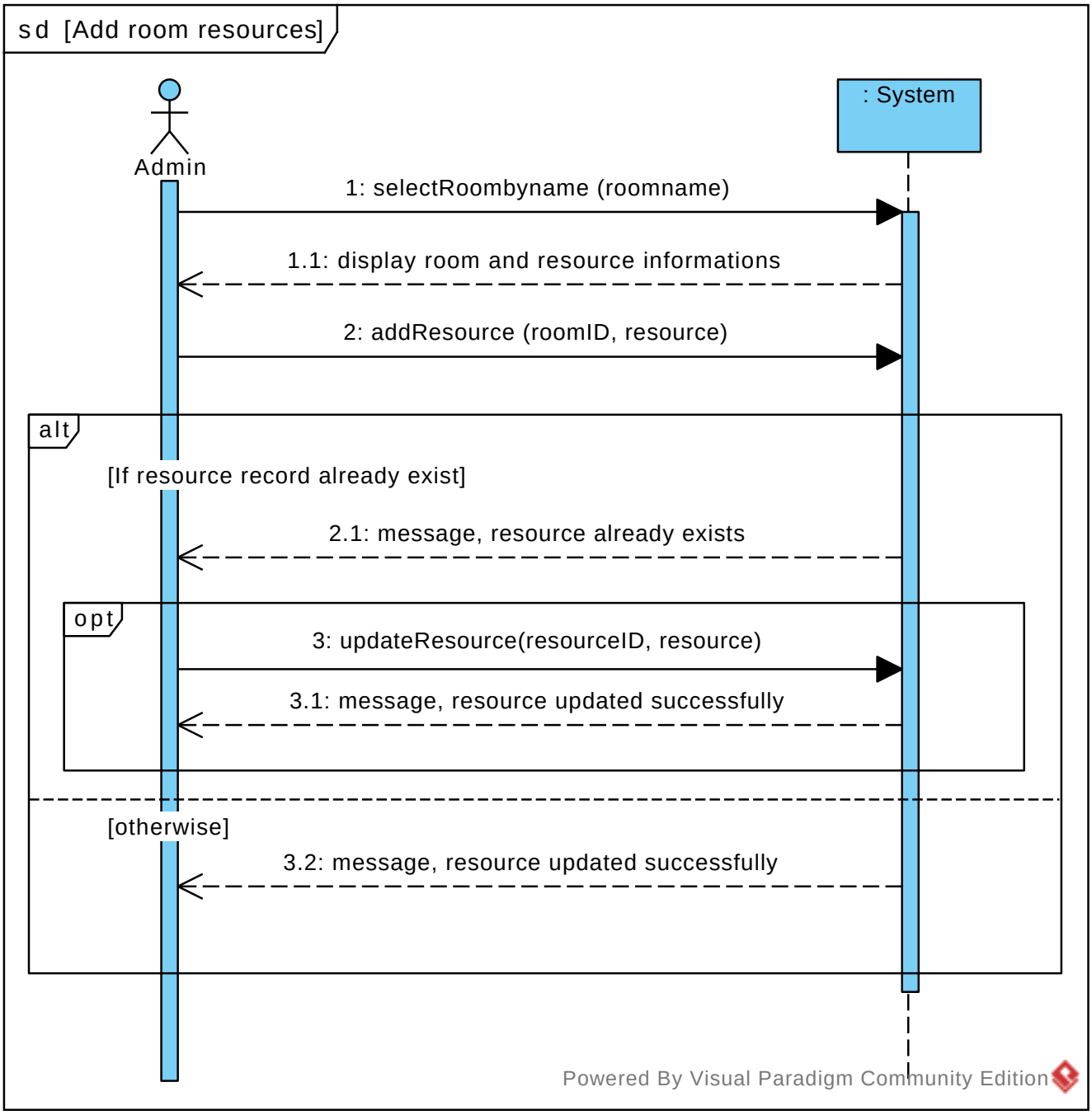

sd [View Schedules]
Guest
: System
1: ViewSchedules()
alt
[If current day schedule is available]
1.1: Current day schedules
1.2: No schedules available
opt
2: FilterSchedules(Date,Time,roomID)
alt
[If schedule is available]
2.1: Display schedule
2.2: No schedules available
Powered By Visual Paradigm Community Edition

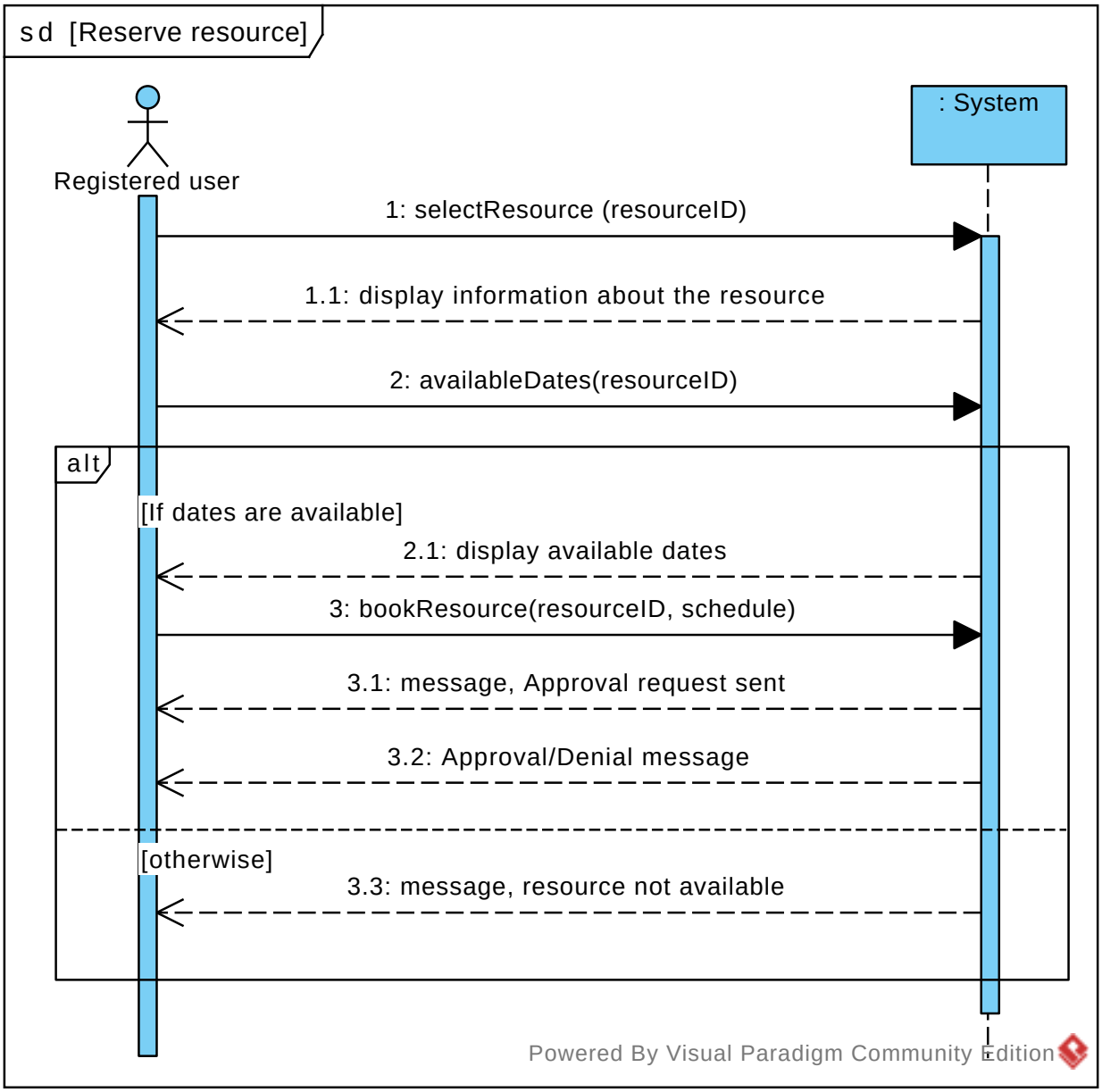

sd [Adding Schedule to System]
Admin
: System
1: SearchRoom(roomID)
alt
[If room is found]
1.1: List of dates and times
2: Select date and time
alt
[If schedule is already present]
2.1: Schedule
alt
3: Edit Schedule
3.1: Schedule Edited confirmation
4: Delete Schedule
4.1: Schedule Deleted Confirmation
[If schedule is not present]
5: Add Schedule
5.1: Schedule Added Confirmation
5.2: No room found message
Powered By Visual Paradigm Community Edition

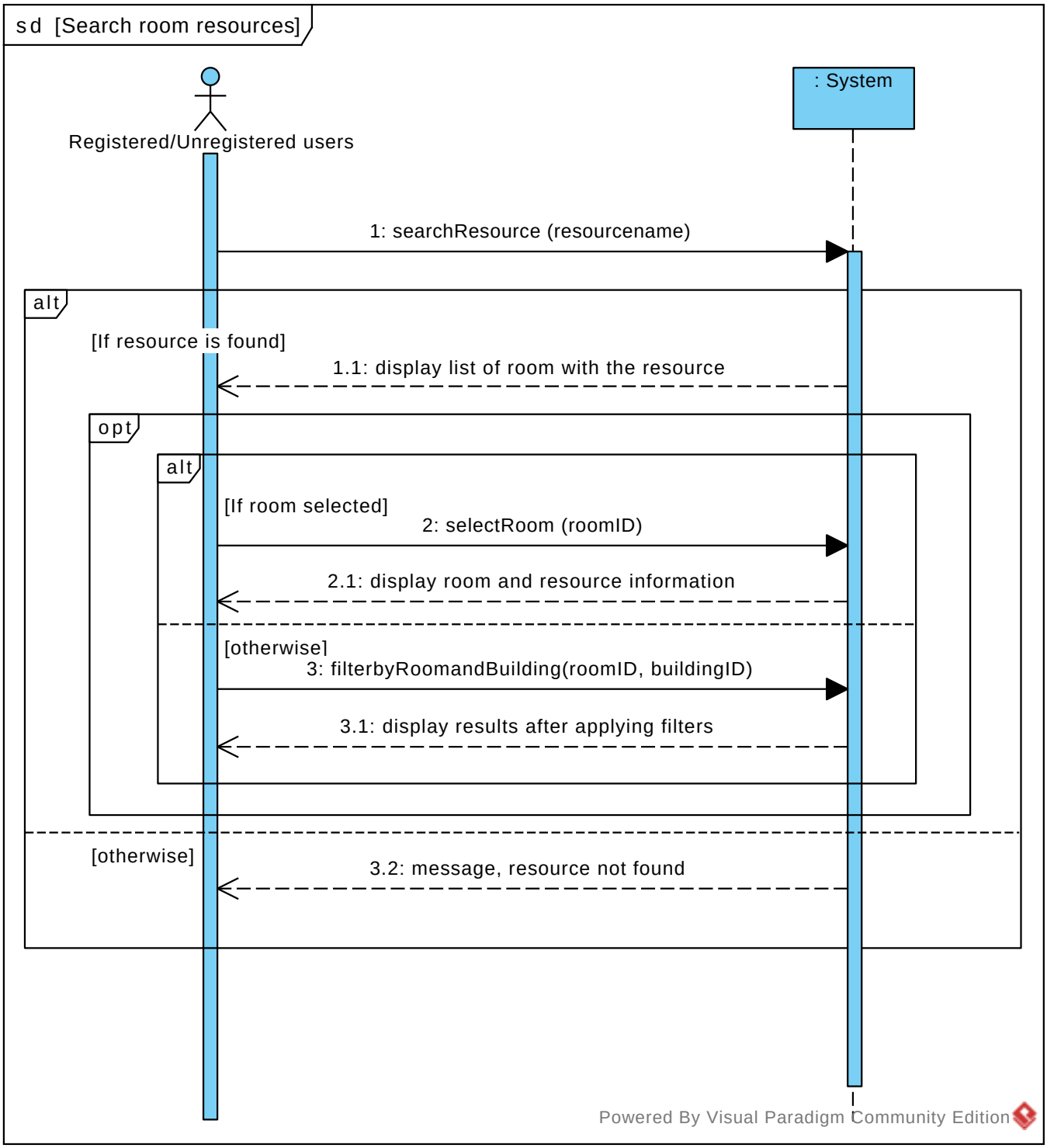

sd [Add a building]
:admin
: System
1: initiateCreateBuilding()
1.1: building information form
2: addBuilding(buildingInfo)
loop
[while name is not valid]
3: putNameofBuilding(buildingName)
3.1: name empty/existing: try a new name
3.2: building saved
Powered By Visual Paradigm Community Edition

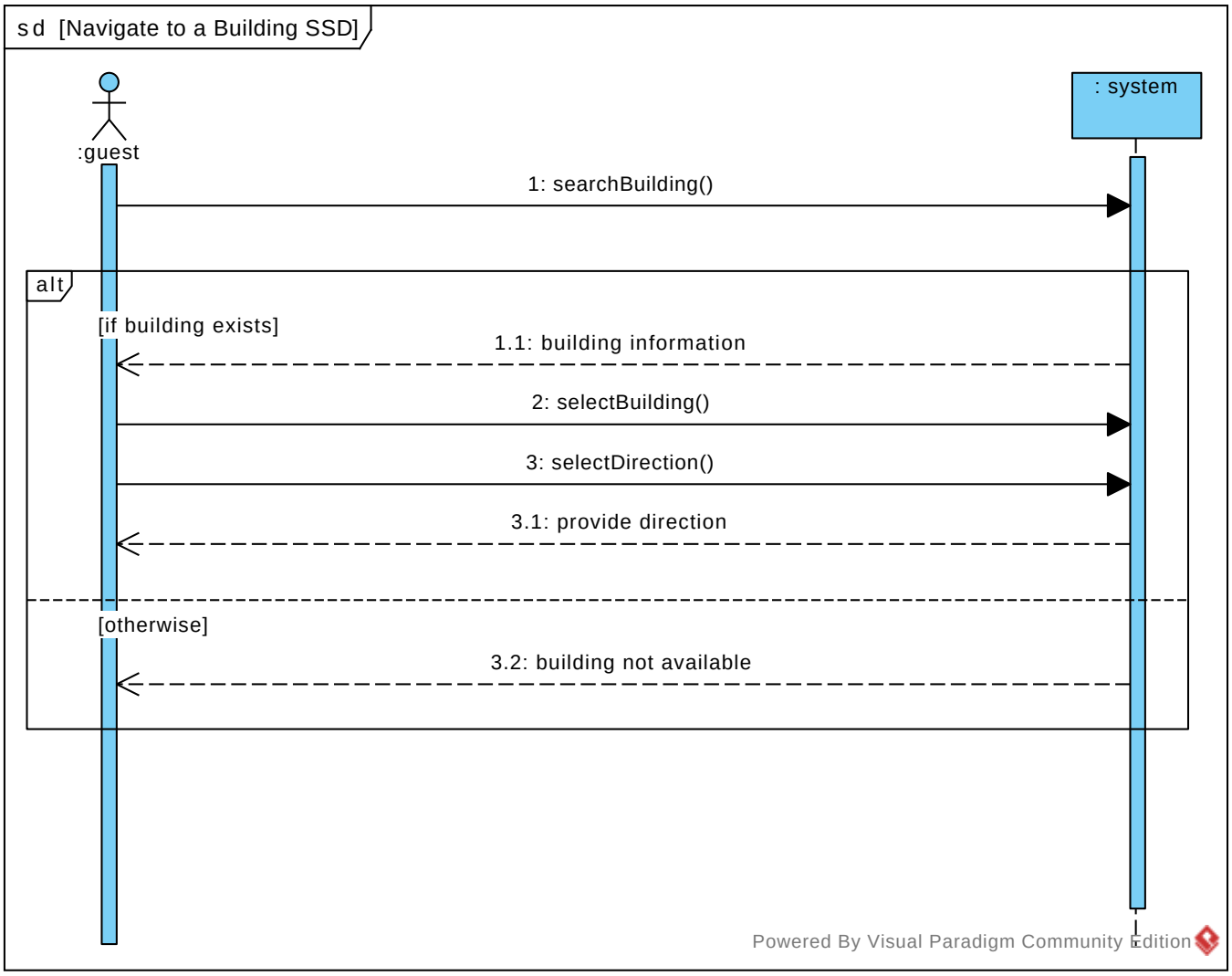

sd [Reserve Room]
:Registered User
: System
1: selectRoom(roomID)
1.1: Room information
2: reserveRoom(true)
alt
[bookable]
2.1: available slots
3: selectDateandTime(date, time)
4: Request for reservation
4.1: reservation status
4.2: decline request
Powered By Visual Paradigm Community Edition

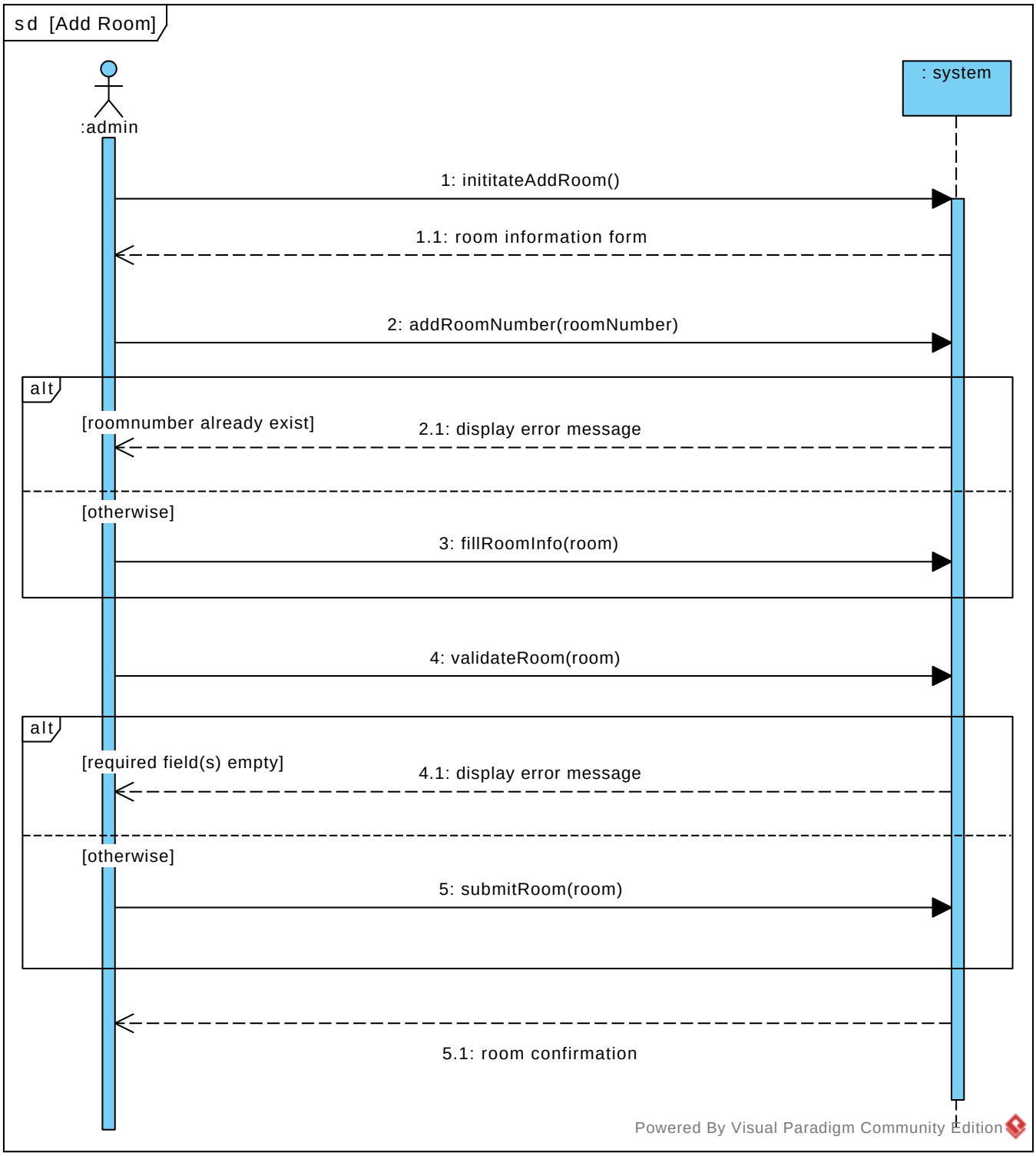

sd [View/update room status]
: Admin
: System
1: selectBuilding(buildingID)
1.1: building information
2: viewRoom()
2.1: list of rooms sorted by date
3: displayRoomStatus()
alt
[if not empty]
3.1: room list sorted by status
opt
[update room status]
4: viewRequest(roomID)
4.1: room information and reservation request
alt
[if requirements are meet]
5: approve()
6: decline()
[empty list]
6.1: "0 result"
Powered By Visual Paradigm Community Edition

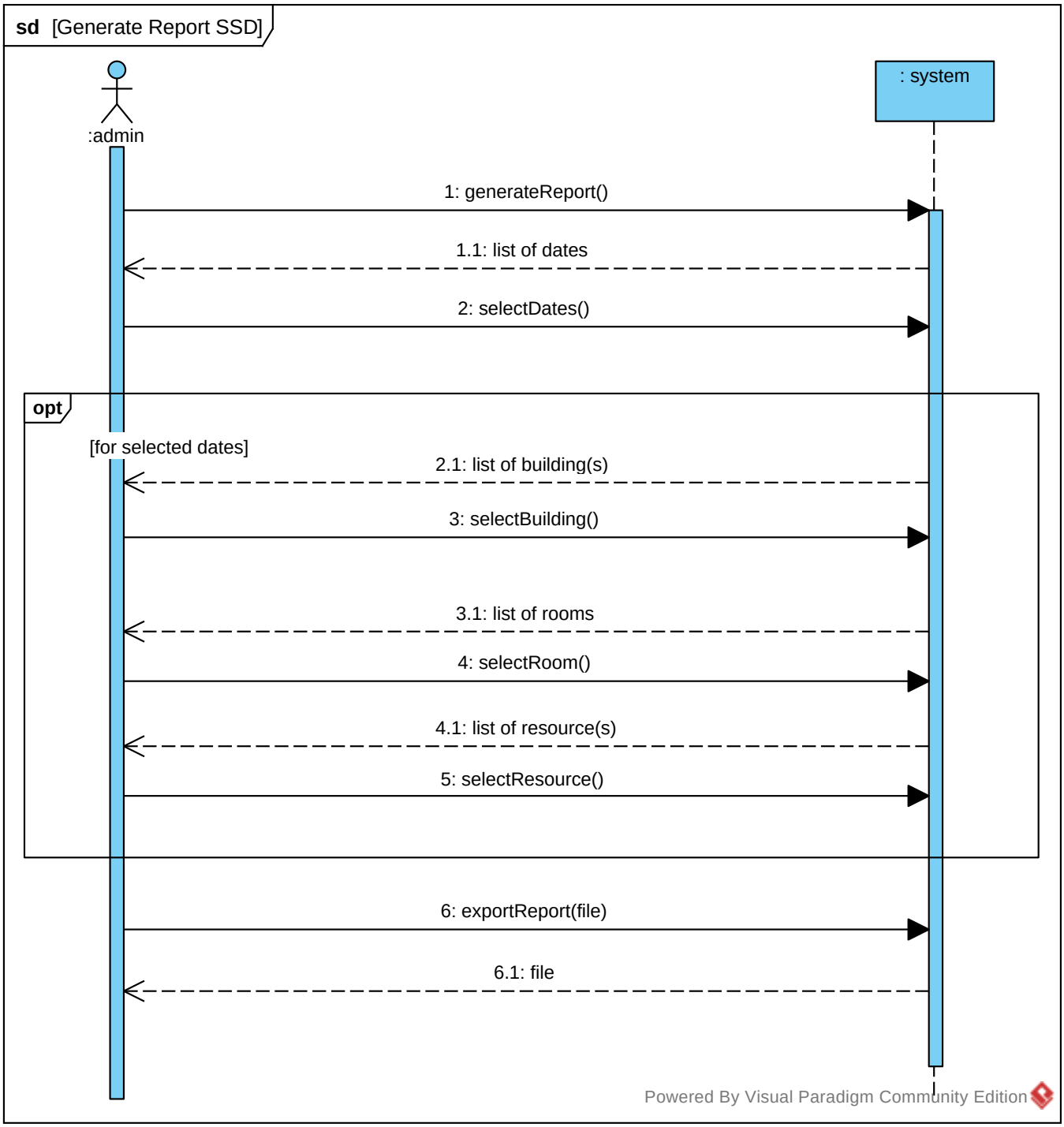

sd [Add room resources]
Admin
: System
1: selectRoombyname (roomname)
1.1: display room and resource informations
2: addResource (roomID, resource)
alt
[If resource record already exist]
2.1: message, resource already exists
opt
3: updateResource(resourceID, resource)
3.1: message, resource updated successfully
[otherwise]
3.2: message, resource updated successfully
Powered By Visual Paradigm Community Edition


sd [Reserve resource]
Registered user
: System
1: selectResource (resourceID)
1.1: display information about the resource
2: availableDates(resourceID)
alt
[If dates are available]
2.1: display available dates
3: bookResource(resourceID, schedule)
3.1: message, Approval request sent
3.2: Approval/Denial message
[otherwise]
3.3: message, resource not available
Powered By Visual Paradigm Community Edition


sd [Search room resources]
Registered/Unregistered users
: System
1: searchResource (resourcename)
alt
[If resource is found]
1.1: display list of room with the resource
opt
alt
[If room selected]
2: selectRoom (roomID)
2.1: display room and resource information
[otherwise]
3: filterbyRoomandBuilding(roomID, buildingID)
3.1: display results after applying filters
[otherwise]
3.2: message, resource not found
Powered By Visual Paradigm Community Edition

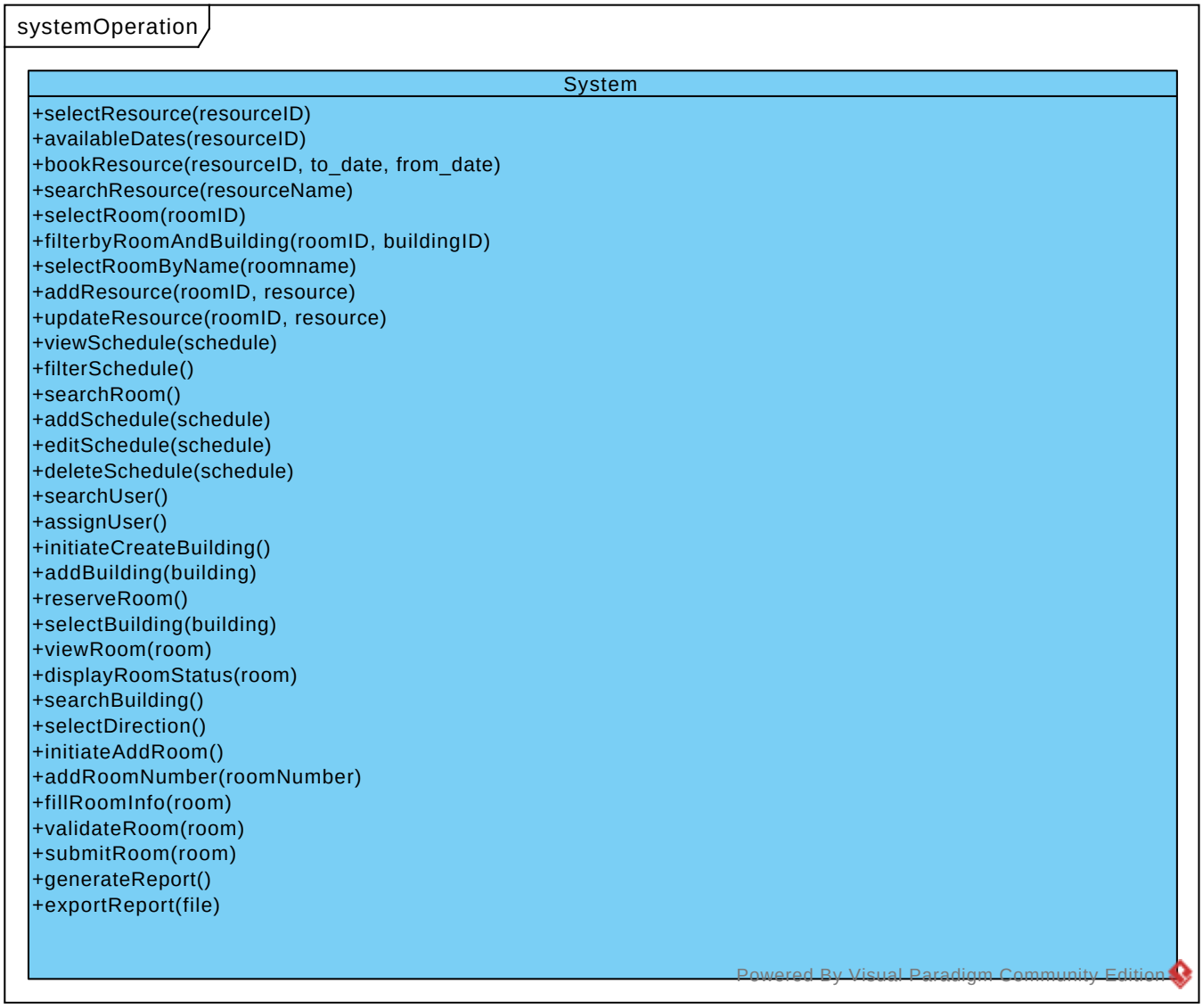

sd [Navigate to a Building SSD]
:guest
: system
1: searchBuilding()
alt
[if building exists]
1.1: building information
2: selectBuilding()
3: selectDirection()
3.1: provide direction
[otherwise]
3.2: building not available
Powered By Visual Paradigm Community Edition


sd [Add Room]
:admin
: system
1: inititateAddRoom()
1.1: room information form
2: addRoomNumber(roomNumber)
alt
[roomnumber already exist]
2.1: display error message
[otherwise]
3: fillRoomInfo(room)
4: validateRoom(room)
alt
[required field(s) empty]
4.1: display error message
[otherwise]
5: submitRoom(room)
5.1: room confirmation
Powered By Visual Paradigm Community Edition


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
Powered By Visual Paradigm Community Edition

7 System Operations

The system operations is presented in the next page.


systemOperation
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)
Powered By Visual Paradigm Community Edition

8 Operation Contracts

The operation coontracts 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, buillding: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:

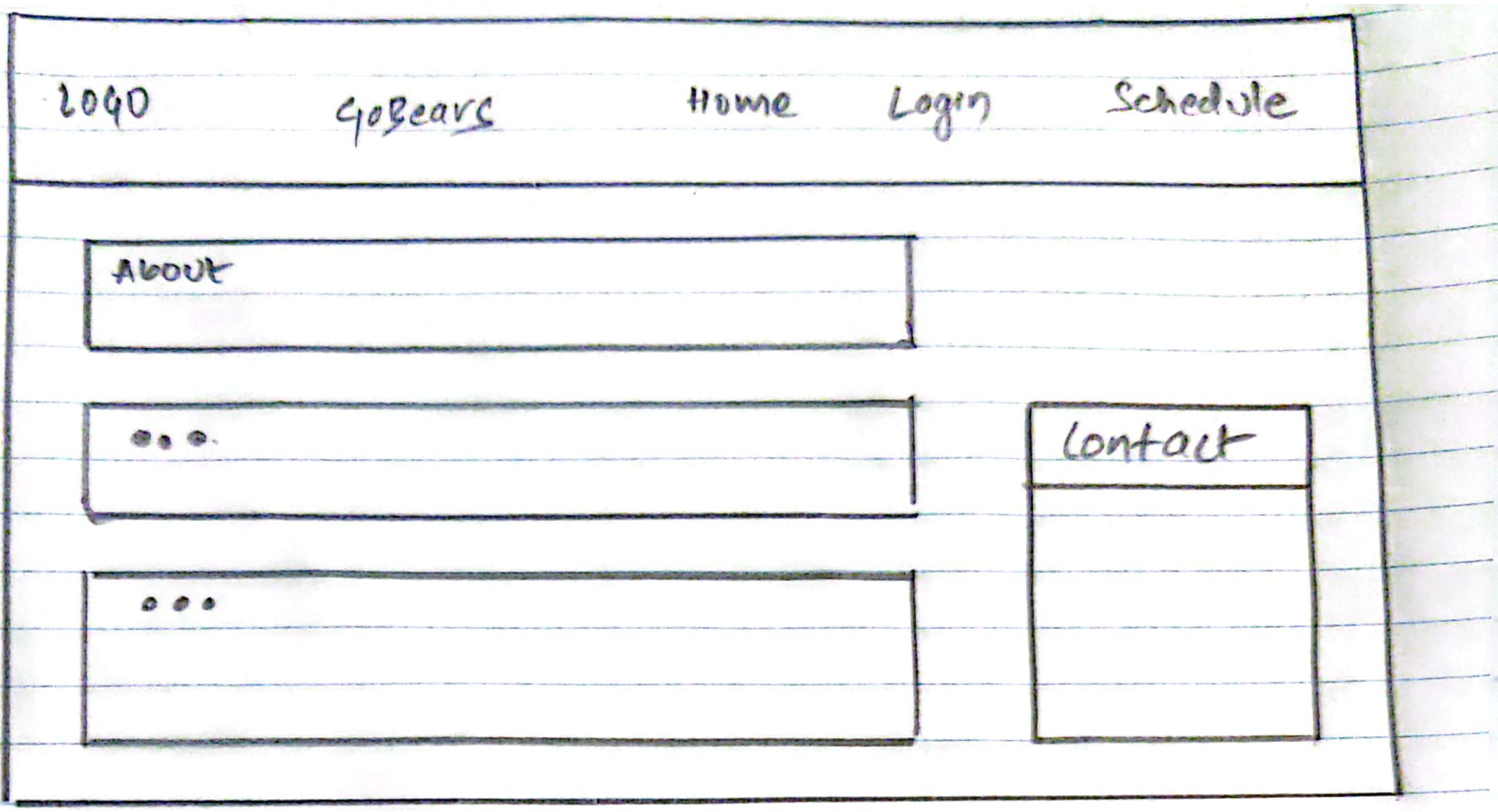
9 Wireframe

The wireframes are listed below:

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

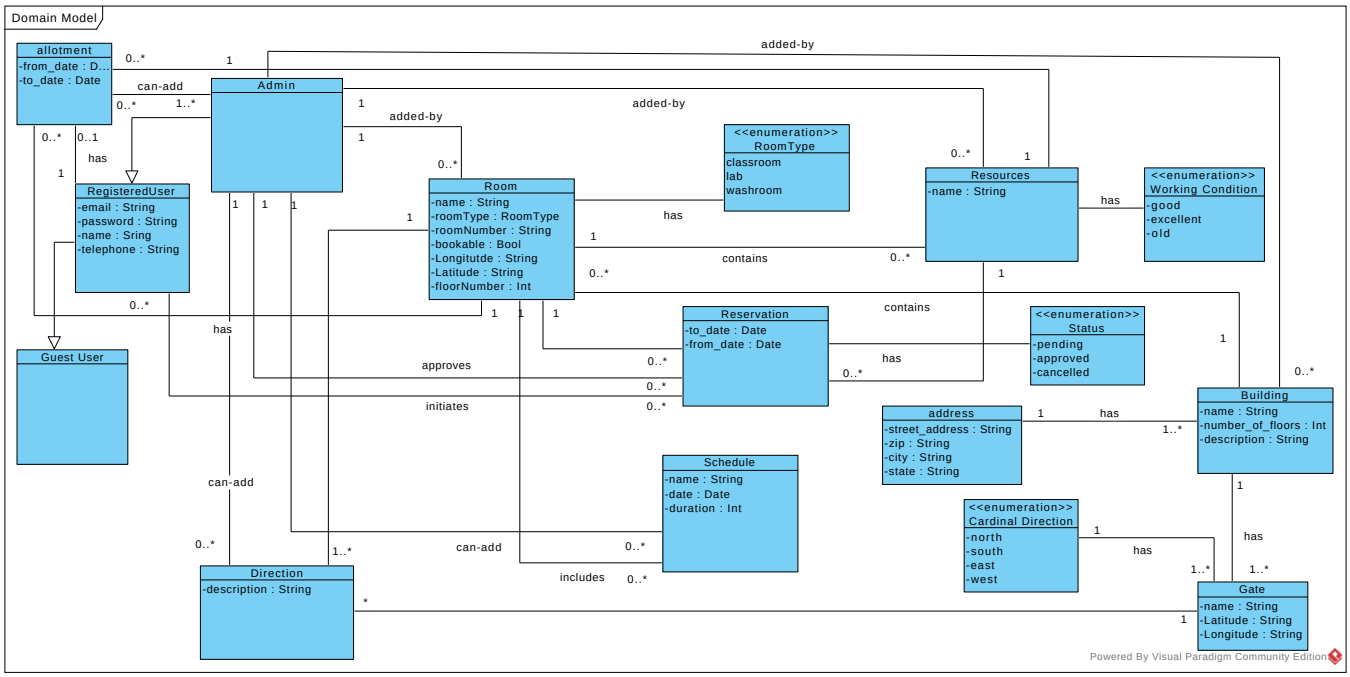
SLHEDULE

ROOM	TIME	EVENT
L315	11:00	SE 5324


LOGO
GoBears
Home
Login
Schedule
About
...
...
Contact

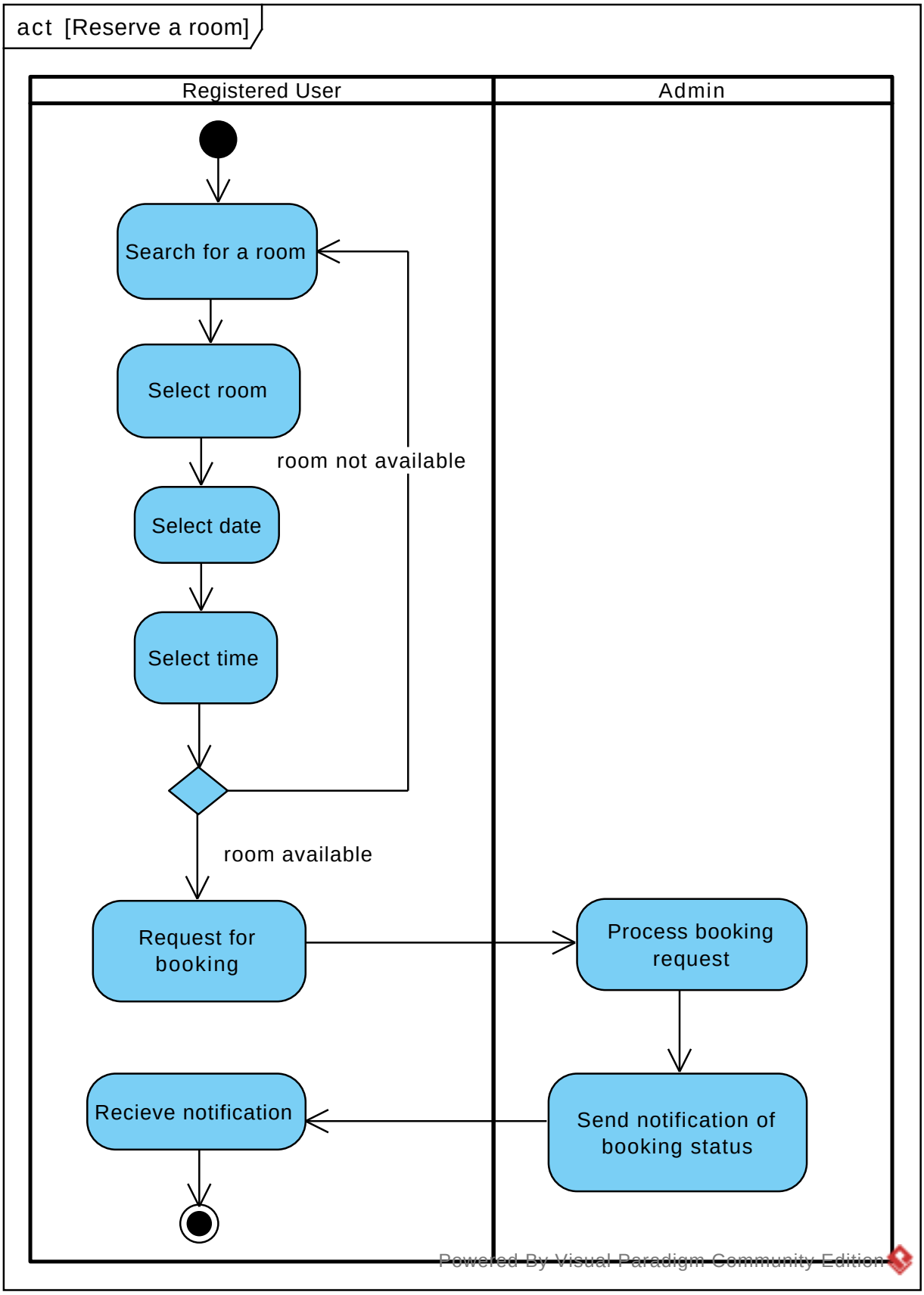
10 Domain Model

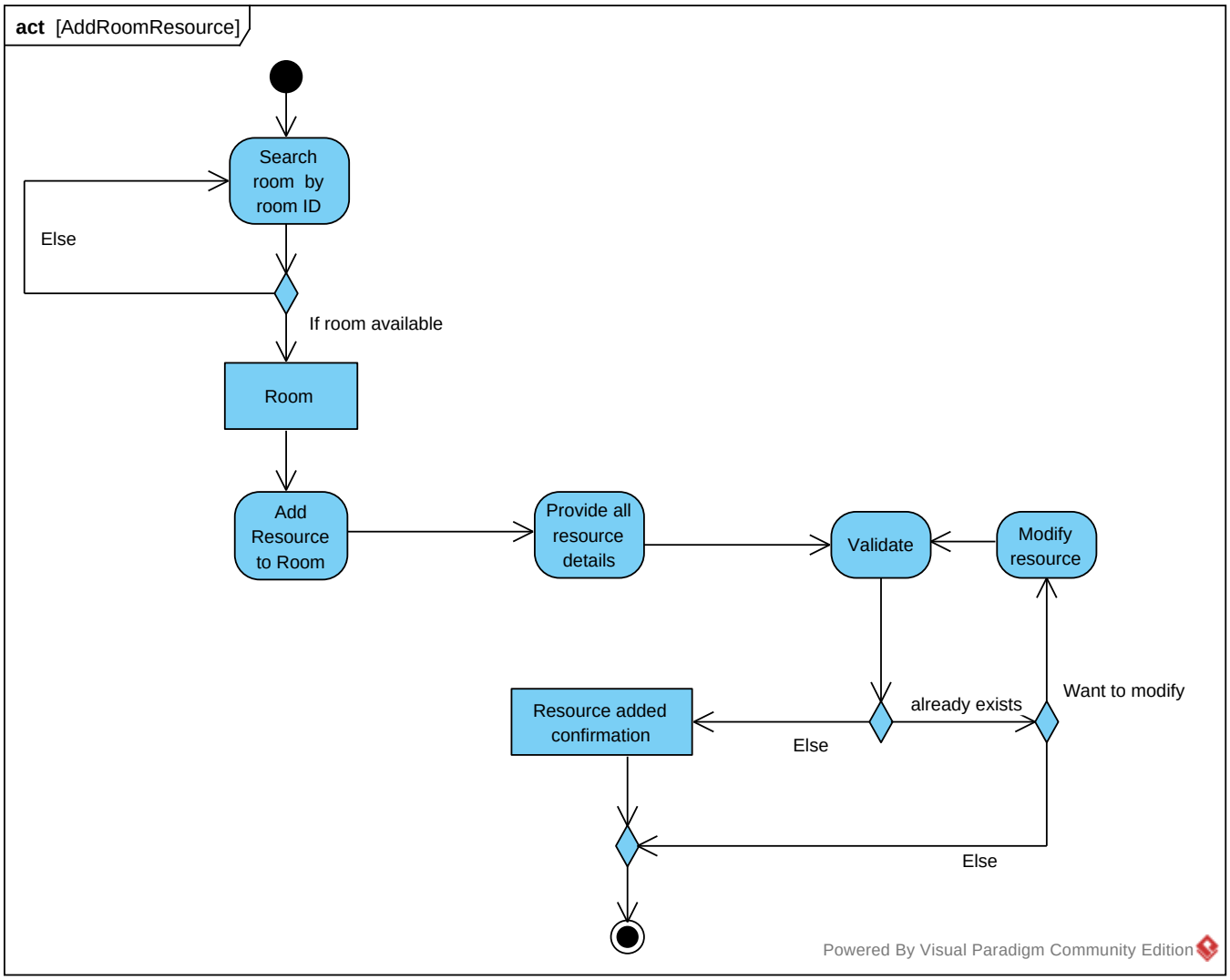
The domain model is presented in the next page.

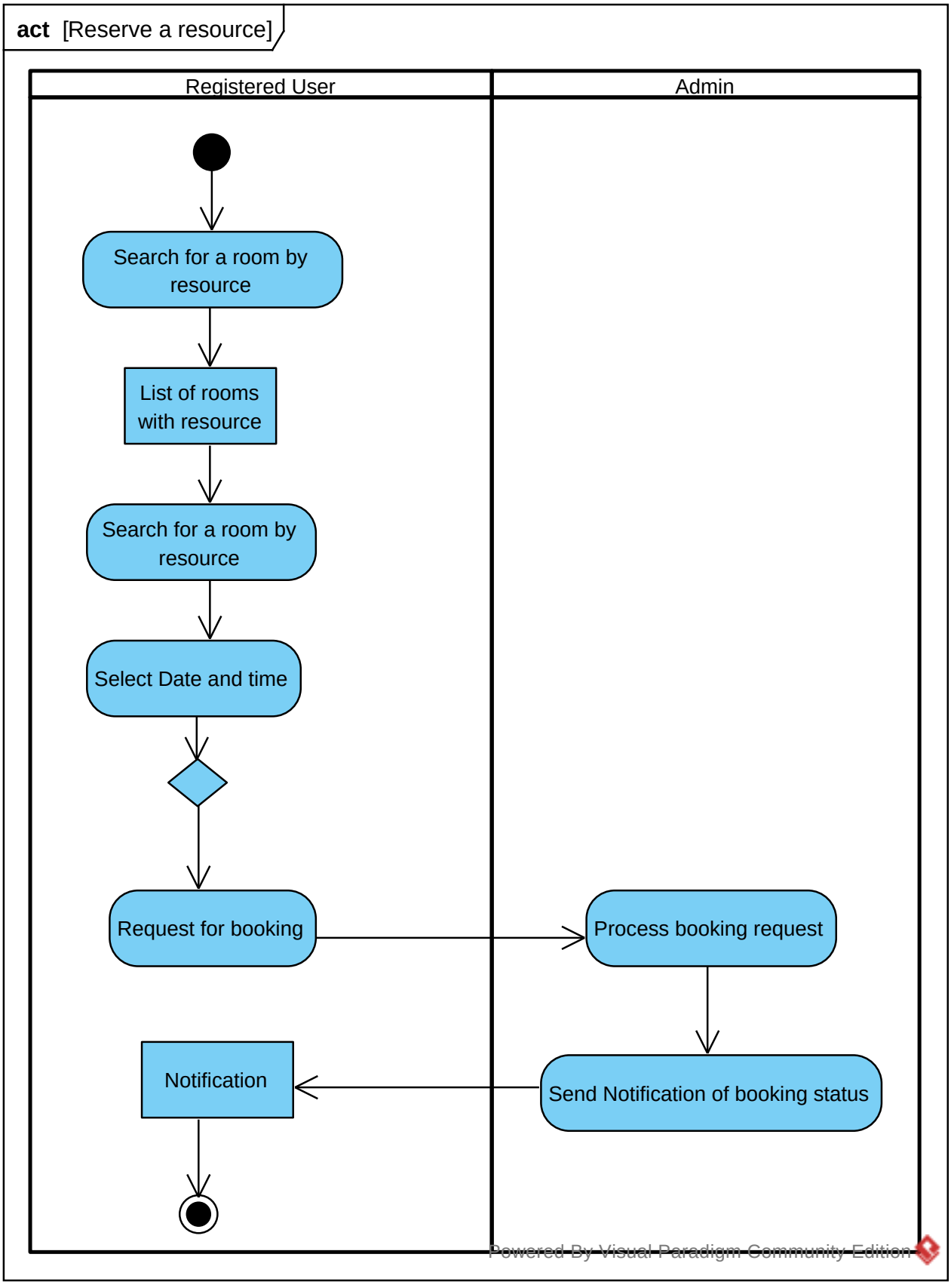

Domain Model
allotment
-from_date : D...
-to_date : Date
Admin
RegisteredUser
-email : String
-password : String
-name : Sring
-telephone : String
Guest User
Room
-name : String
-roomType : RoomType
-roomNumber : String
-bookable : Bool
-Longitutde : String
-Latitude : String
-floorNumber : Int
<<enumeration>>
RoomType
classroom
lab
washroom
Resources
-name : String
<<enumeration>>
Working Condition
-good
-excellent
-old
Reservation
-to_date : Date
-from_date : Date
<<enumeration>>
Status
-pending
-approved
-cancelled
Building
-name : String
-number_of_floors : Int
-description : String
address
-street_address : String
-zip : String
-city : String
-state : String
Schedule
-name : String
-date : Date
-duration : Int
<<enumeration>>
Cardinal Direction
-north
-south
-east
-west
Direction
-description : String
Gate
-name : String
-Latitude : String
-Longitude : String
added-by
can-add
has
contains
approves
initiates
includes
Powered By Visual Paradigm Community Edition

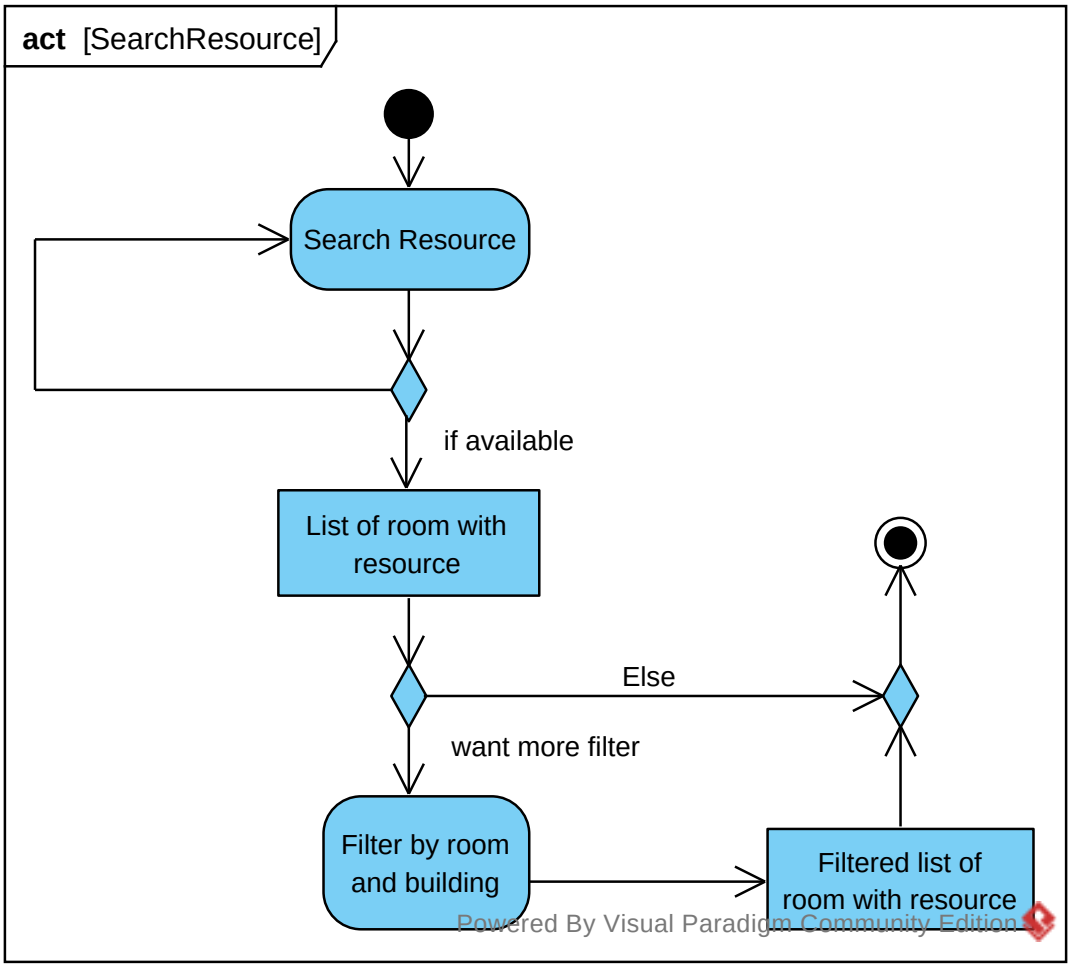
11 Activity Diagram

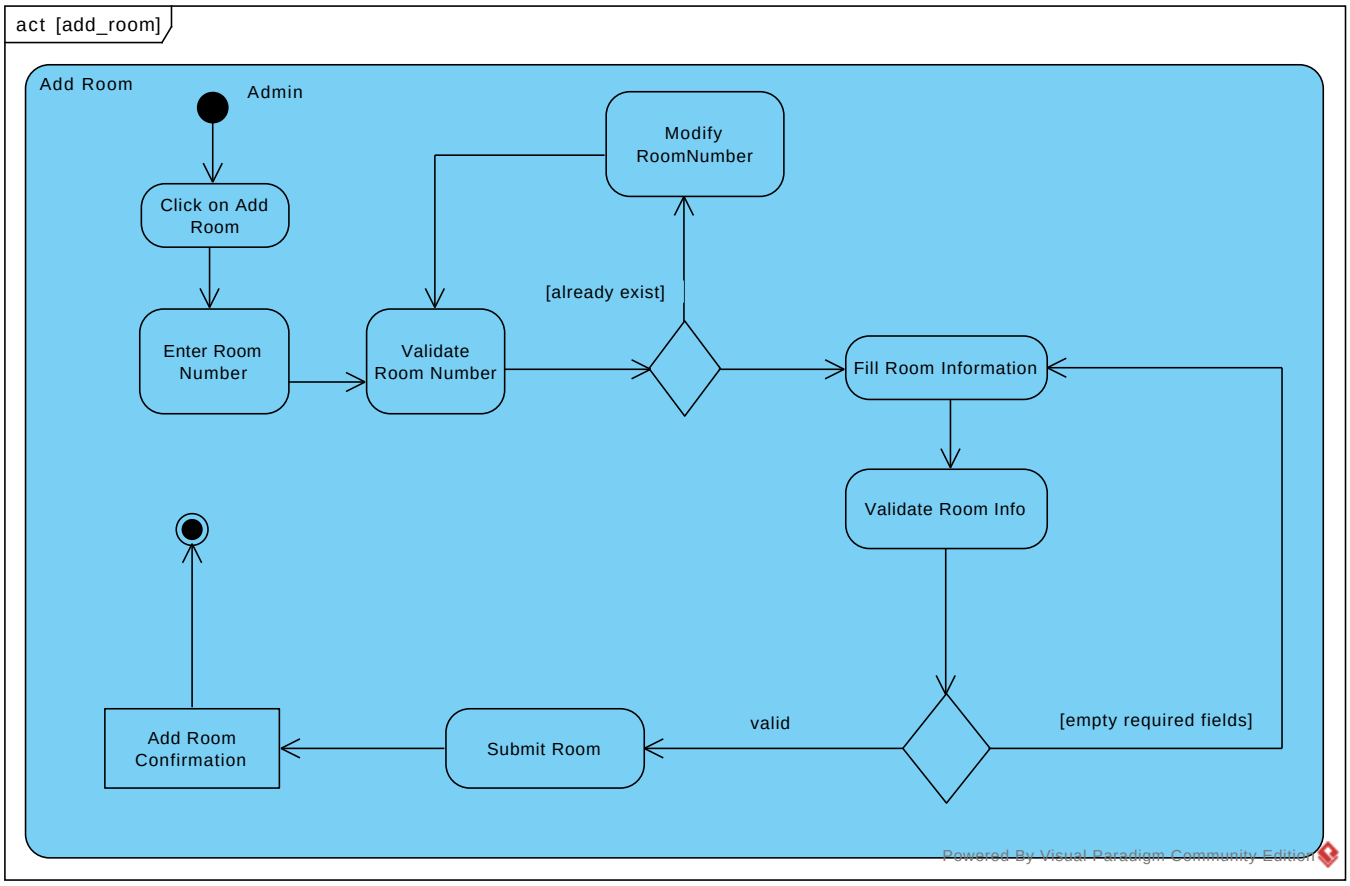
The activity diagrams are listed from the next page:

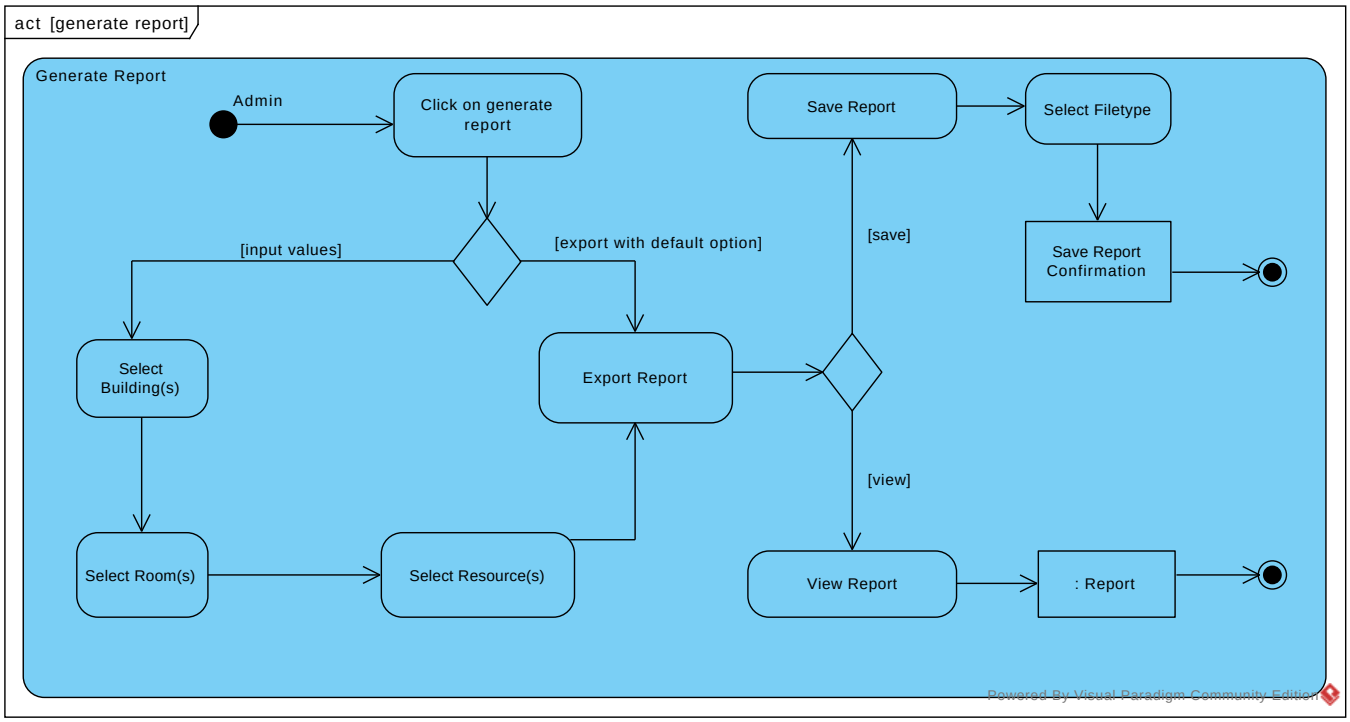

act [Reserve a room]
Registered User
Admin
Search for a room
Select room
room not available
Select date
Select time
room available
Request for booking
Process booking request
Recieve notification
Send notification of booking status
Powered By Visual Paradigm Community Edition

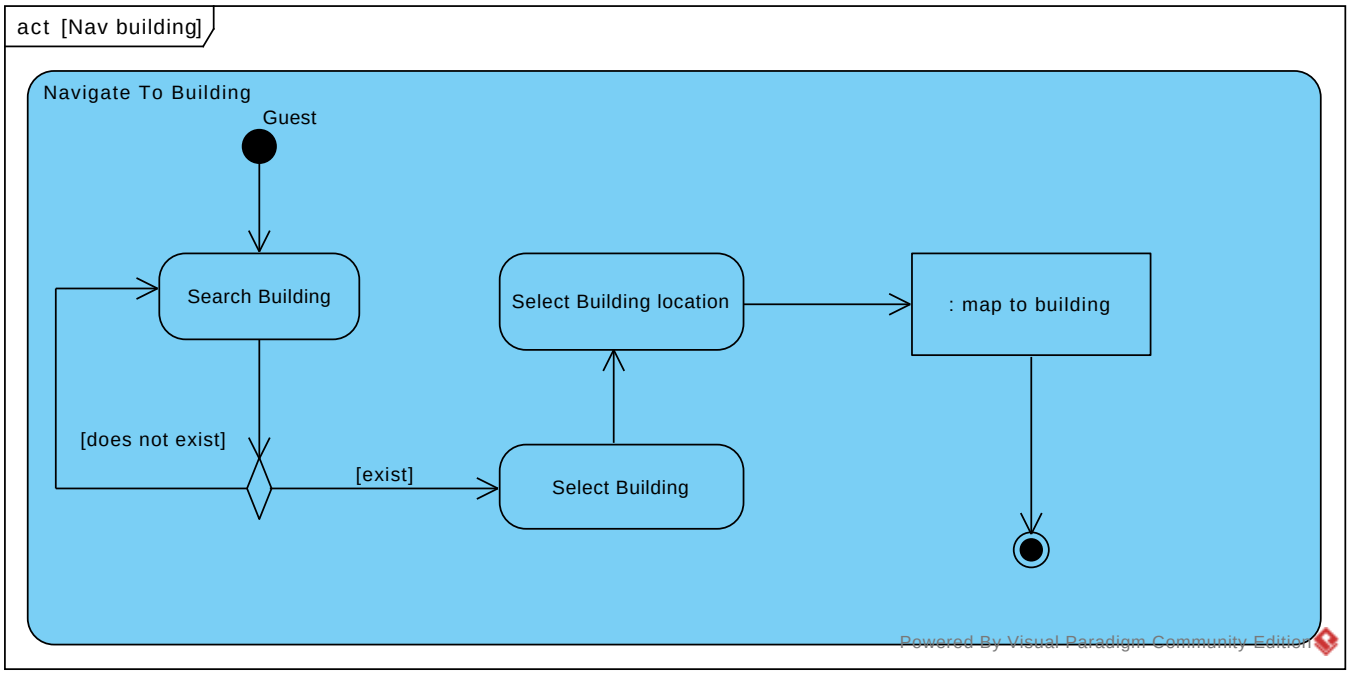

act [AddRoomResource]
Search room by room ID
Else
If room available
Room
Add Resource to Room
Provide all resource details
Validate
Modify resource
Resource added confirmation
already exists
Want to modify
Else
Else
Powered By Visual Paradigm Community Edition

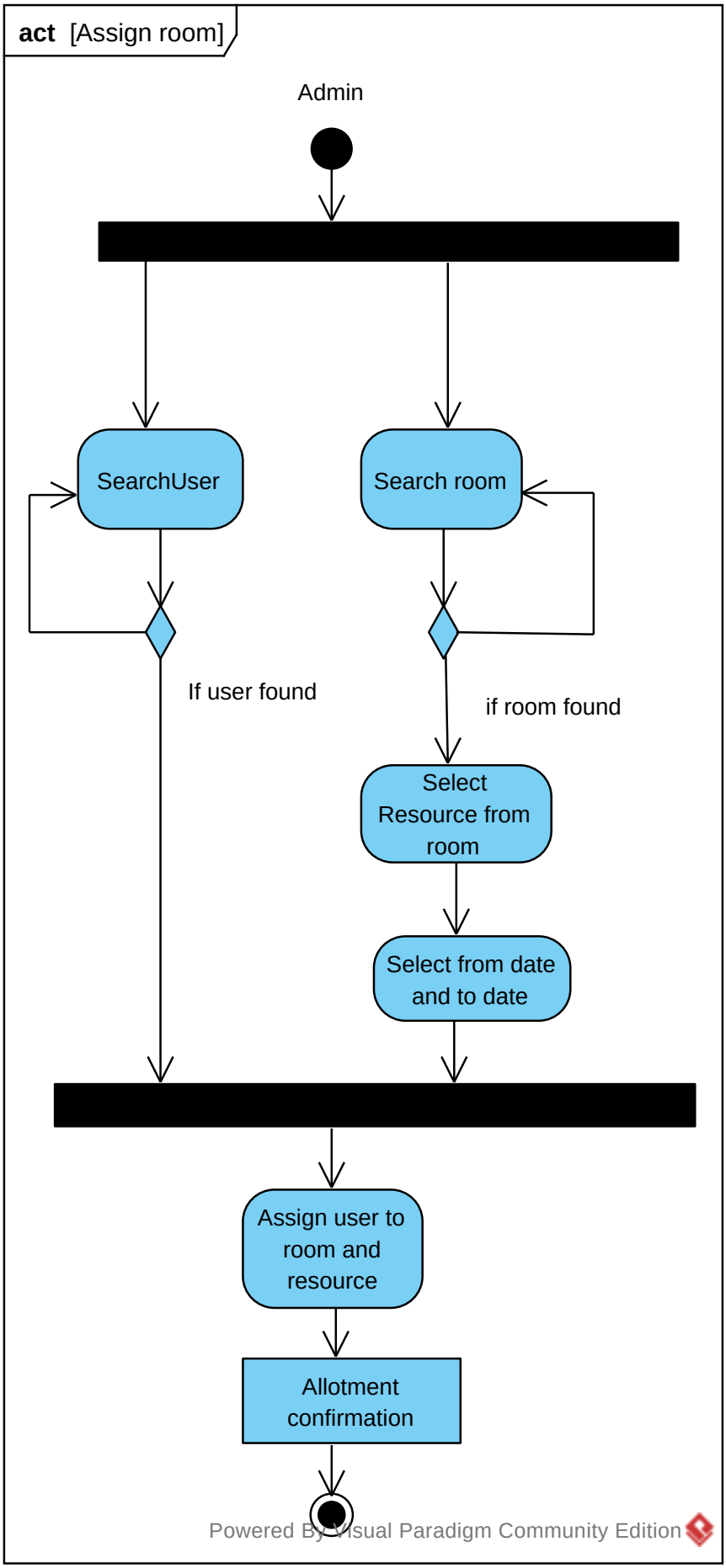

act [Reserve a resource]
Registered User
Admin
Search for a room by resource
List of rooms with resource
Search for a room by resource
Select Date and time
Request for booking
Process booking request
Notification
Send Notification of booking status
Powered By Visual Paradigm Community Edition

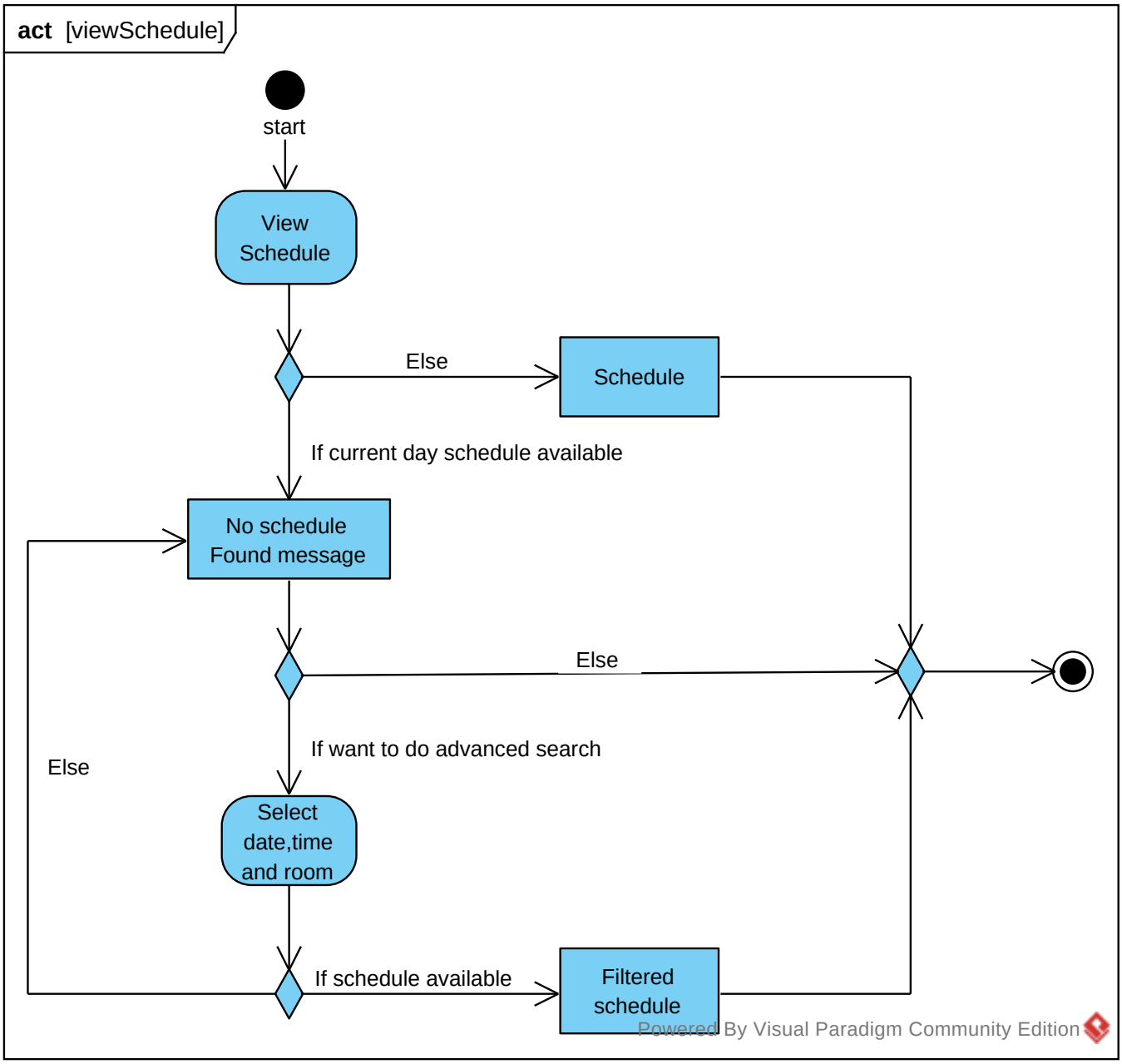

act [SearchResource]
Search Resource
if available
List of room with resource
Else
want more filter
Filter by room and building
Filtered list of room with resource
Powered By Visual Paradigm Community Edition

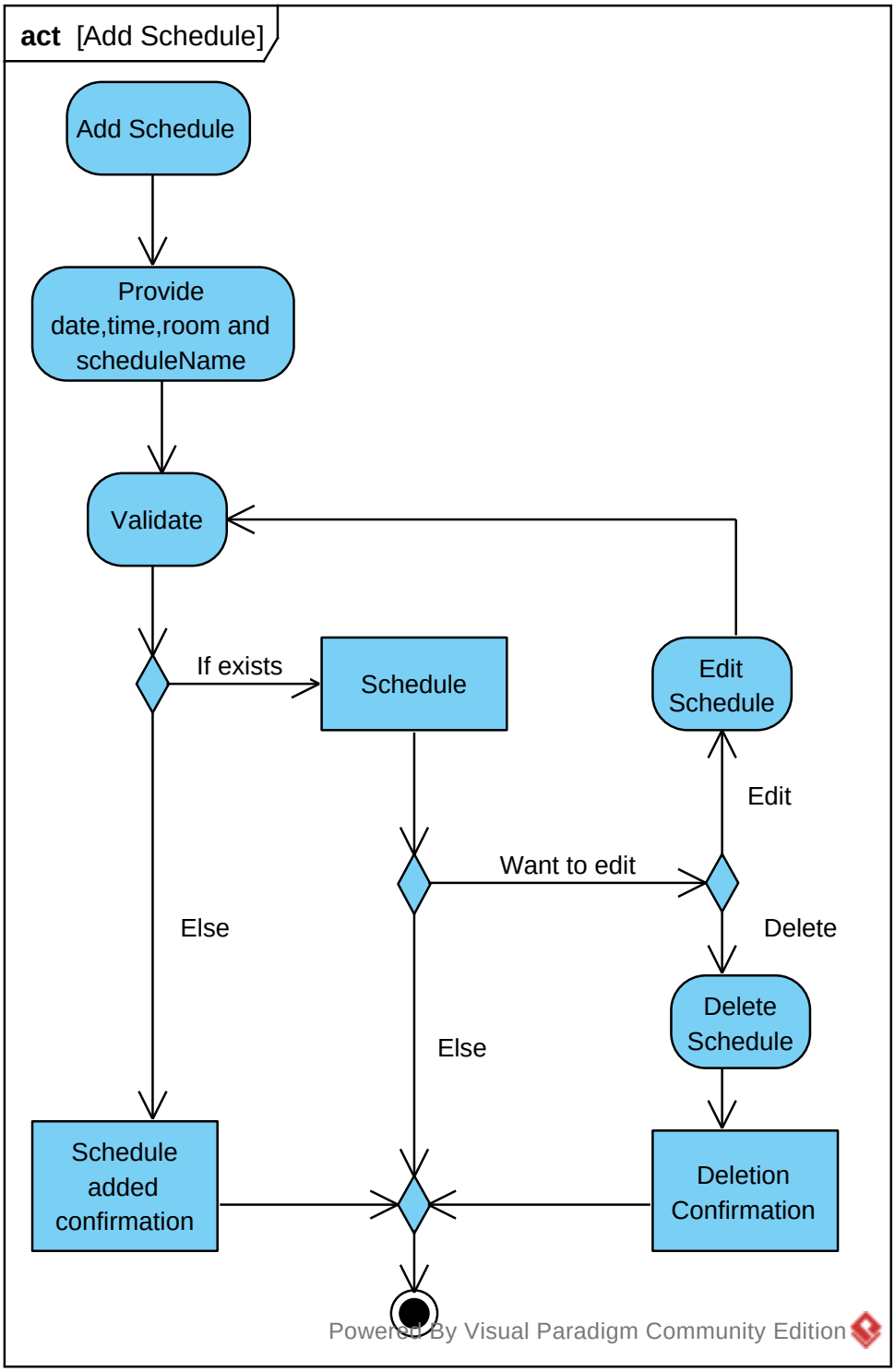

act [add_room]
Add Room
Admin
Click on Add Room
Enter Room Number
Validate Room Number
[already exist]
Modify RoomNumber
Fill Room Information
Validate Room Info
[empty required fields]
valid
Submit Room
Add Room Confirmation
Powered By Visual Paradigm Community Edition


act [generate report]
Generate Report
Admin
Click on generate report
[input values]
[export with default option]
Select Building(s)
Select Room(s)
Select Resource(s)
Export Report
[save]
[view]
Save Report
Select Filetype
Save Report Confirmation
View Report
: Report
Powered By Visual Paradigm Community Edition


act [Nav building]
Navigate To Building
Guest
Search Building
[does not exist]
[exist]
Select Building
Select Building location
: map to building
Powered By Visual Paradigm Community Edition


act [Assign room]
Admin
SearchUser
Search room
If user found
if room found
Select Resource from room
Select from date and to date
Assign user to room and resource
Allotment confirmation
Powered By Visual Paradigm Community Edition


act [viewSchedule]
start
View Schedule
Else
Schedule
If current day schedule available
No schedule Found message
Else
If want to do advanced search
Else
Select date,time and room
If schedule available
Filtered schedule
Powered By Visual Paradigm Community Edition


act [Add Schedule]
Add Schedule
Provide date,time,room and scheduleName
Validate
If exists
Schedule
Edit Schedule
Edit
Want to edit
Else
Delete
Delete Schedule
Else
Schedule added confirmation
Deletion Confirmation
Powered By Visual Paradigm Community Edition

12 Gantt Chart

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

Project manager	
Project dates	Aug 22, 2022 - Dec 9, 2022
Completion	31%
Tasks	32
Resources	4

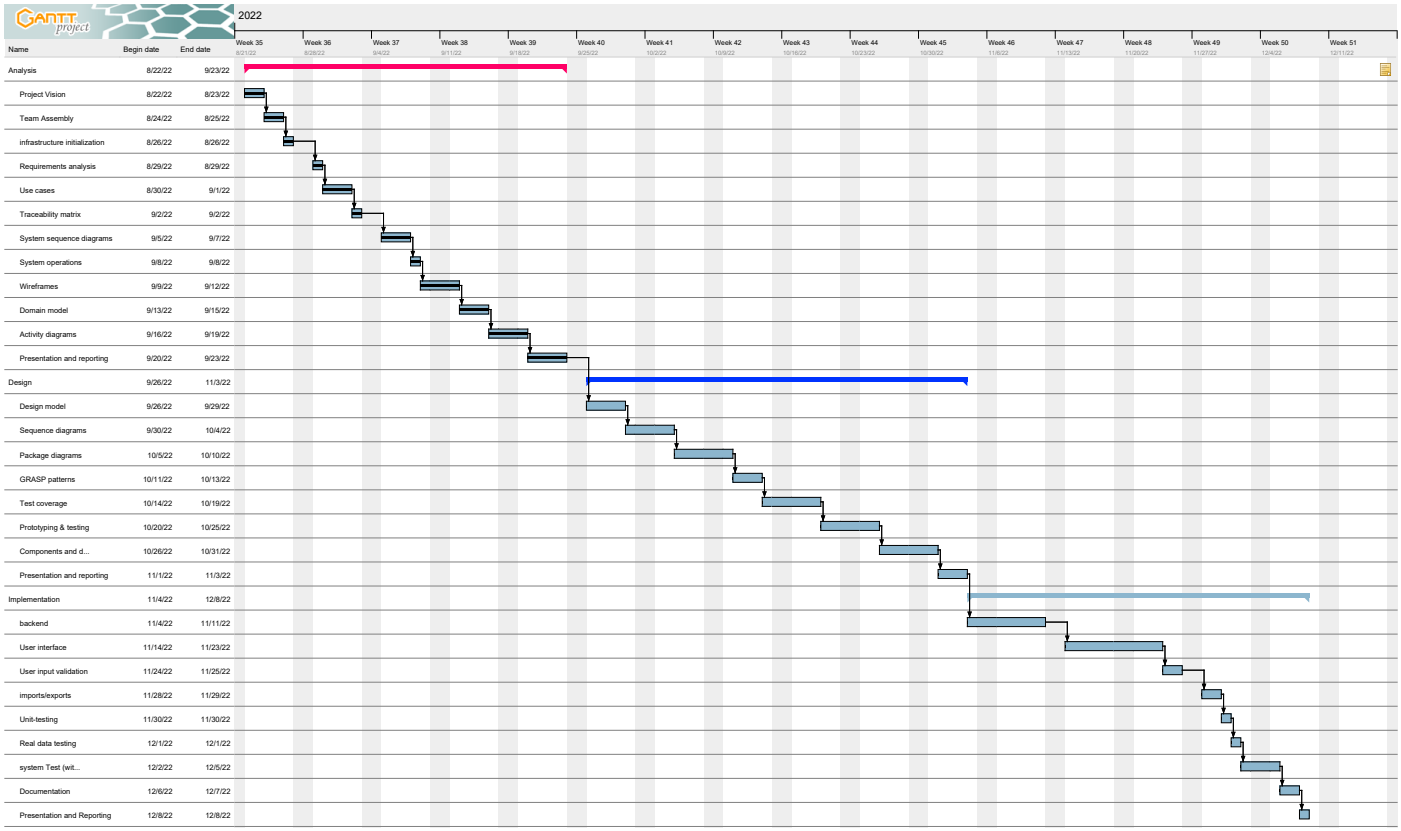
Tasks

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

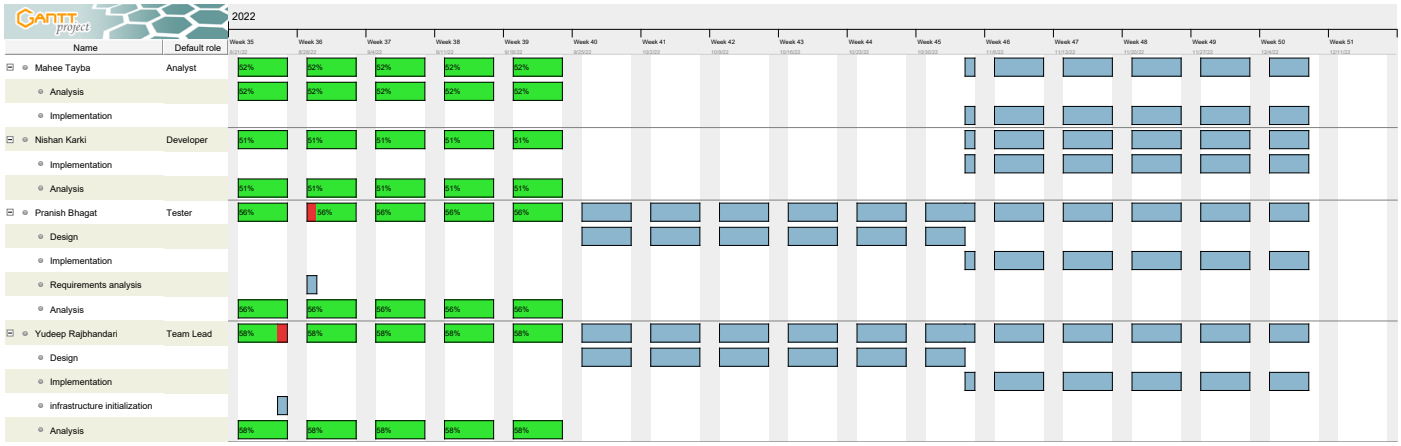
Name	Default role
Mahee Tayba	Analyst
Nishan Karki	Developer
Pranish Bhagat	Tester
Yudeep Rajbhandari	Team Lead

Gantt Chart



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 d...	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 (wit...	12/2/22	12/5/22
Documentation	12/6/22	12/7/22
Presentation and Reporting	12/8/22	12/8/22

Resources Chart



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