Hotel Management System

Requirements Specification and Analysis

Table of Contents

REQUIREMENTS ANALYSIS DOCUMENT [1]

1. Introduction

1.1. Purpose of the System

The main purpose of the Hotel Management System is to make Hotel Staff comfortable to manage their employees and perform tasks, allow them to ensure their guests are pleased by their accommodation and experience. The system also creates a safe and fast booking and satisfying interface for the guests. By using our system to book a room, the user no longer needs to call the hotel and spend too much time on it. As well as being able to book a room in the hotel online safely and in a fast manner. Hotels can easily manage what their employees must do daily, get reports weekly about employee's work, feedbacks, and workloads, reach every customers and rooms details. Managers can access their employee's payrolls, and all financial expenses of their hotel.

1.2. Scope of the System

This system allows the guests to book rooms and contact the receptionist online, allows the hotel managers to keep track of their employee's progress and the changes in their hotel. The system also allows for easy access to information and contact between all its users, hotel staff can use the system to organize and schedule their tasks and reach them easily, the system allows for handling of related financial matters.

1.3. Objectives and Success Criteria of the Project

The objective of this project is to build a functioning online hotel management system, that provides secure and fast interactions between its users.
The project would be considered a success if:

- Project is completed in time.
- Project meets the appropriate quality targets.
- Project delivered all items within the agreed scope.
- Project meets the functional requirements.
- Project meets the nonfunctional requirements.
- Project used the approved technology.

1.4. Definitions, Acronyms, and Abbreviations

User: any person who uses the system.
Guest: any person with the intention to spend time in the hotel and use it's services.
Employee: any person working at the hotel in any capacity.
Admin: administrator of the hotel also known as General manager.
Manager: manages staff and receptionists and deals with events hosted by the hotel, deals with guests in cases where he is needed.
Receptionist: deals with guest related matters and helps guests when necessary.
Staff: Can be cleaners, bellboys, waiters, cooks, and chefs.

RAD: Requirement Analysis Document.
GUI: A GUI or graphical user interface is a form of user interface that allows users to interact with electronic devices through a graphical interface.
HTML: Hyper Text Markup Language. It is the standard markup language for documents designed to be displayed in a web browser.
CSS: Cascading Style Sheets. CSS is a style sheet language used for describing the presentation of a document written in HTML.
UX: User Experience. UX abbreviation is used to define the design process to create products that provide meaningful and proper experiences to users.
SQL: Structured Query Language. The SQL language is used in programming for managing data in a relational database.
MVC: Model, View, and Controller. MVC abbreviation is used to define a software design pattern commonly used for developing user interfaces that distribute the program logic into three elements.
JS: JavaScript. JavaScript is the main programming language of the Web.
IDE: Integrated Development Environment. And IDE is a software application that provides facilities to programmers for software development.
TCP/IP: Transmission Control Protocol and the Internet Protocol.

1.5. Overview

The rest of RAD contains a description of the purpose of the system, Functional requirement, nonfunctional requirements, Scenarios, use cases, Sequential diagrams, a description of the GUI, object model, Glossary and References.

The RAD is organized following the table of contents.

2. Current System

The current system suffers from delays and latency in responses between the guests and the hotel, requiring the guest to be in the hotel in person or calling in advance. Our system solves this problem by providing an instant line of communication between the guests and the hotel, removing the need for the guest to be there in person.

3. Proposed System

This proposed system allows the guests to book rooms and contact the receptionist online, allows the hotel managers to keep track of their employee’s progress and the changes in their hotel. The system also allows for easy access to information and contact between all its users, hotel staff can use the system to organize and schedule their tasks and reach them easily, the system allows for handling of related financial matters.

3.1. Overview

This system will be used by users such as guests, hotel managers and admins, staff and receptionists It will allow them to interact with each other and perform reservations and keep track of their tasks and information.

3.2. Functional Requirements

- The receptionist can access the management system, reserves rooms and cancel reservations, see information about rooms, guests, and events.
- The guest should be able to request room service and cleaning, list the events such as conferences.
- The admin should be able to access all the information related to guests, rooms, employees, and managers, as well as editing said information and salaries.
- The manager should be able to create/cancel events, access and edit employees’ information as well as rooms information and guest details.
- The employees and staff should be able to access and list their tasks and information, they can also give feedback on completed tasks.
- The system allows guests to create their own accounts.
- System only allows admins to change employee salaries.
- System authenticates the user if appropriate credentials were given.
- System show the user only the appropriate pages when valid authentications are given.
- System only give permission to the manager to create an event.
- System only allows managers and admins to make an announcement through the System.
- First time the user logs into the system with his/her account, System necessitate the user to change her/his password.

3.3. Nonfunctional Requirements

Usability

The system has user friendly GUI and designed with all type of users in mind. Documentation required to operate the system is also provided.

Reliability

The system provides reliable methods of payment and reservations, the functionality of the system also allows for uninterrupted usage and transactions using robust and secure measures, *Performance*

The system loads the pages within 2 seconds and fetches the data from the database in acceptable time interval. It also can handle a large amount of concurrent users and a high number of transactions per second.

Supportability

The system uses technologies that are widely supported by the open source community and we will be providing support after the release to fix any possible bugs.

Implementation

The system is implemented on a windows platform but designed to be accessible from all platforms since it is a web application.

Interface

The interface is designed and implemented in a way that allows ease of use and practicality, without sacrificing usability and performance, it also provides visual hierarchy to help the user perform their task uninterrupted. The system also uses HTTP and TCP/IP protocols.

Packaging

The system doesn’t need any installation whatsoever from the user, since it’s web based.

Legal

The software is developed in the scope of SOFT3101, no commercial use is expected, thus no license is needed.

3.4. System Models

Scenarios

Scenario Name: Listing empty rooms
Participating Actors: Bob: Receptionist, Ruby: Manager
Flow of Events: 1. Ruby accesses the system with her credentials. 2. Ruby clicks on get general report about all rooms button on the main page 3. The system shows Ruby general report about all rooms within a specified time interval. 4. Bob accesses the system with user Id and password. 5. Bob reaches the homepage of the system and clicks the get report button. 6. System redirects the receptionist to a page to fill the necessary information about the rooms he is looking for. 7. System creates the report about all the rooms and their general information.

Scenario Name: Editing time interval for rooms.

Participating Actors: Bob: Receptionist, Alice: Guest

Flow of Events:

1. Alice accesses the system with her credentials.
2. Alice requests to edit the time interval for her room in accordance with the hotel's policy.
3. The system creates a report for Alice's request and sends it to Bob.
4. Bob accesses the system with his user ID and password.
5. Bob checks Alice's request on the system.
6. Bob requests policy approval form the system for Alice.
7. System sends that policy to Alice.
8. After getting Alice's approval system allows Bob to edit the room info.
9. Bob reaches to the system and changes time interval as Alice asked.

Scenario Name: Create/cancel reservations

Participating Actors: Bob: Receptionist, Alice: Guest

Flow of Events:

1. Alice logs into the system with the required credentials.
2. Alice creates a request for creating/canceling a reservation.
3. The system sends the request to Bob.
4. Bob accesses the system with his user ID and password.
5. Bob accepts/denies the request from Alice.
6. The system informs Alice with an announcement.

Scenario Name: Reaching Guest Information

Participating Actors: Bob: Receptionist

Flow of Events:

1. Bob accesses the system with user ID and password.
2. Bob clicks on show all guest information button on the main page.
3. The system shows a page with the list of guests and their information.

Scenario Name: Create a default Guest account

Participating Actors: Alice: Guest

Flow of Events:

1. Alice clicks on Create account button.
2. The system gives Alice a form to fill in with the necessary information.
3. Alice fills the form and clicks on save button.
4. System automatically creates a guest account for Alice with given information.

Scenario Name: Request room service / cleaning

Participating Actors: Alice: Guest, Mark: Employee

Flow of Events:

1. Alice accesses the system with her credentials.
2. Alice clicks on request room cleaning/service button.
3. The system creates the request and sends it to Mark.
4. Mark's list of tasks is updated to match the new change.
5. Mark logs in to system with user ID and password and checks his tasks.

Scenario Name: List the services or events

Participating Actors: Alice: Guest

Flow of Events:

1. Alice accesses the system with her credentials.
2. Alice clicks on List the services or events button.
3. The system shows Alice the list of services and events provided by the hotel.

Scenario Name: Access all rooms information

Participating Actors: Clark: Admin

Flow of Events:

1. Clark logs in to the system with user ID and password.
2. Clark clicks on Access all rooms information button
3. The system shows the list of all the rooms of the hotel with their information displayed.

Scenario Name: Access all the customers information

Participating Actors: Clark: Admin

Flow of Events:

1. Clark logs in to the system with user ID and password.
2. Clark clicks on Access all the customers information button.
3. The system shows the list of all the customers information.

Scenario Name: Access staff lists

Participating Actors: Clark: Admin

Flow of Events:

1. Clark logs in to the system with user ID and password.
2. Clark clicks on Access staff lists button.
3. The system shows the list of all the staff information.

Scenario Name: Editing Room Details

Participating Actors: Clark: Admin

Flow of Events:

1. Clark logs in to the system with user ID and password.
2. Clark clicks on Edit room detail button.
3. The system shows the list of all the rooms.
4. Clark chooses a room to edit its details.
5. The system shows the details of that room.
6. Clark edits and clicks on save changes button.
7. The system updates the related information.

Scenario Name: Listing and Editing Salaries of Employees

Participating Actors: Clark: Admin

Flow of Events:

1. Clark logs in to the system with user ID and password.
2. Clark clicks on Edit Salaries button.
3. The system shows the list of all the staff salaries.
4. Clark chooses the employee or group of employees to change their salaries.
5. The system opens the editing page.
6. Clark enters the changes he wants to make.
7. The system updates the information.

Scenario Name: Creating Events

Participating Actors: Ruby: Manager

Flow of Events:

1. Ruby logs in to the system with user ID and password.
2. Ruby clicks on Create Event button.
3. The system requests the necessary information to create an event.
4. Ruby enters the necessary information.
5. The system validates the information and creates the event.

Scenario Name: The list of daily staff

Participating Actors: Ruby: Manager

Flow of Events:

1. Ruby accesses the system with her user ID and password.
2. The system redirects Ruby to the home page.
3. Ruby clicks on request the list of daily staff button on the page.
4. The system creates the list of the daily staff.
5. System brings that list to Ruby's screen.

Scenario Name: Editing Tasks of Employees

Participating Actors: Ruby: Manager, Rosalia: Employee

Flow of Events:

1. Ruby logs in to the system with user ID and password.
2. Ruby clicks on edit employee tasks button on the page.
3. The system shows the list of the employees.
4. Ruby chooses an employee to change the task for.
5. The system opens a page for Ruby with the employee's tasks to change.
6. Ruby edits/change tasks for Rosalia from the system.
7. The system updates the task list for Rosalina.

Scenario Name: Editing Employees Information

Participating Actors: Ruby: Manager, Rosalia: Employee

Flow of Events:

1. Ruby logs in to the system with user ID and password.
2. Ruby clicks on edit employee information button on the page.
3. The system lists all the employees and their respective information.
4. Ruby chooses an employee to change its information.
5. The system opens a page to edit the employee’s information.
6. Ruby edits/changes the employee’s information from the system.
7. The system updates the employee’s information.

Scenario Name: Canceling tasks for Employees

Participating Actors: Ruby: Manager, Rosalia: Employee

Flow of Events:

1. Rosalina logs in to the system with user ID and password.
2. Rosalina clicks on cancel task button.
3. The system creates a request and sends it to Ruby.
4. Ruby logs in to the system with user ID and password.
5. The system shows Ruby the request created by Rosalina.
6. Ruby clicks on cancel employee tasks button on the page.
7. The system removes the canceled task from Rosalina’s list of tasks.

Scenario Name: Listing Daily Tasks

Participating Actors: Rosalia: Employee

Flow of Events:

1. Rosalia accesses to the system with user ID and password.
2. Rosalina clicks on daily tasks button.
3. The system shows the list of daily tasks.

Scenario Name: Accessing Information

Participating Actors: Rosalia: Employee

Flow of Events:

1. Rosalia accesses to the system with user ID and password.
2. Rosalina clicks on show personal information button.
3. System shows Rosalia her own list of all information and tasks listed

Scenario Name: Giving Feedbacks

Participating Actors: <u>Rosalia: Employee</u>

Flow of Events:

1. Rosalia accesses to the system with user ID and password.
2. Rosalina clicks on give feedback button.
3. The system shows a list of Rosalina's daily tasks.
4. Rosalina selects a task to give feedback on.
5. The system shows the task's details.
6. Rosalina inputs the feedback into the system.
7. The system updates the task.

Use case model

Use case name: Booking Room

Participating actors: Receptionist, Guest

Flow of Events:

7. Guest logs into the system with the required credentials.
8. Guest creates a request for creating/canceling a reservation.
9. The system sends the request to Receptionist.
10. Receptionist accesses the system with his user ID and password.
11. Receptionist accepts/denies the request from Guest.
12. The system informs the guest with an announcement.
13. The guest gives their feedback.
14. The system updates according to the new changes and lets the user know.

Entry condition: Receptionist and Guest logged into system.

Exit conditions: Guest payment failed, or Guest payment succeeded, or Receptionist completed the booking procedure.

Quality requirement:

- Searches done by the receptionist should return results withing an acceptable delay.
- The guest should be able to cancel a reservation withing a fixed period.

Use case name: see Guest information

participating actors: Receptionist

flow of events:

4. Receptionist accesses the system with user ID and password.
5. Receptionist clicks on show all guest information button on the main page.
6. The system shows a page with the list of guests and their information.

entry condition: Receptionist logged into system

exit condition: The receptionist successfully reached the guest using the information found on the system, or the guest could not be reached/ wrong number.

quality requirement:

- The receptionist should be able to access the guest information in a timely manner.
- The customer's sensitive information should be protected without compromising his safety while looking for relevant information.

Use case name: Check daily task

participating actors: Employee

flow of events:

1. The employee logs into the system.
2. The employee clicks the Daily Task Button.
3. System displays the list to employee.

entry condition: Employee clicks the Daily Task Button

exit condition: System displays the list to employee.

quality requirement:
- Employee should be able to access their information and task in a timely manner.
- The employee should only be able to see his/her own task.

Use case name: Check the events

participating actors: Guest

flow of events:
1. The guest logs into the system.
2. The guest clicks on the Check events button.
3. The system shows the guest a list of the events they can attend.

entry condition: The guest clicks on the Check events button.

exit condition: The system shows the events available.

quality requirement:
- The guest listed the events in a timely manner.
- The guest did not have to compromise any personal information to perform this task.

Use case name: Change salary

participating actors: Admin

flow of events:
1. The admin logs into the system.
2. The admin clicks on list of employees.
3. The system shows the list of the employees.
4. The admin selects an employee or group of employees.
5. The admin clicks Change salary button and changes the salary/salaries.
6. The system updates the employees' information accordingly.

entry condition: The admin clicks on list of employees.
exit condition: The system updates the employees' information successfully.
quality requirement: • The admin should be able to access the staff list in a timely manner • System interface gives meaningful feedbacks.

Use case name: Edit room details
participating actors: Admin
flow of events: 1. The admin logs into the system. 2. The admin clicks on list of rooms button. 3. The system shows the list of the rooms. 4. The admin selects a room from the list and clicks edit button. 5. The admin inputs the changes and clicks save button. 6. The system updates the room information.
entry condition: The admin clicks on list of rooms.
exit condition: The system updates the room information
quality requirement: • The admin accessed the room details in a timely manner • The admin changed the information in question without compromising other sensitive data.

Use case name: Cancel an event
participating actors: Manager
flow of events: 1. The Manager logs into the system. 2. The Manager clicks on list of events.

3. The system shows the list of the events available.
4. The Manager selects an event.
5. The Manager clicks cancel event button.
6. The system updates the list of events accordingly.
7. The system sends a notification to all attendees and staff related.

entry condition: The manager logged into the system.

exit condition: The system updates the list of events successfully, or the cancelation is not permitted for legal reasons.

quality requirement:

- The manager should be able to cancel the event withing the period agreed on with the attendees and guests.
- System interface gives meaningful feedbacks

Use case name: Edit employee’s tasks.

participating actors: Manager

flow of events:

1. Manager logs into the system.
2. Manager clicks on Edit employee’s task button
3. System shows the manager a list of all the employees.
4. The manager selects an employee to edit a task for.
5. The system shows the selected employee’s current tasks.
6. The manager chooses a task to edit.
7. The system updates the employee’s list of tasks.

entry condition: The manager logged into the system.

exit condition: The manager successfully edited the task for that employee.

quality requirement:

- The manager accessed the employee’s list of tasks in a timely manner.
- The manager should be able to edit the employee’s task without interfering with other employees’ tasks.

Use case name: log into the system

Participating actors: System User

Flow of Events:

1. The user opens the website.
2. The user clicks the log in button.
3. System shows a form with username and password
4. User enters username and password and submit the form
5. System validates the inputs.
6. System authenticate the user and redirect him to the dashboard.

Entry condition: Click the log in button

Exit conditions: Submitting the wrong information to system or logging in successfully.

Quality requirement:

- System should respond in few seconds
- System interface gives meaningful feedbacks.

Use case name: See all Guests in a specific time interval

Participating actors: Manager/Admin

Flow of Events:

1. Manager/Admin logs into the system.
2. Manager/Admin clicks on List Guests.
3. The system shows Manager/Admin the list of all guests and input for time interval.
4. The Manager/Admin fill in the time interval.
5. The system updates the list of the guests that meet that requirement.

Entry condition: Clicking the list guest button

Exit conditions: updating the list of the guests successfully.

Quality requirement:

- System should respond in few seconds.
- System interface gives meaningful feedbacks.

Use case name: Get report of employee’s tasks

Participating actors: Manager/Admin

Flow of Events:

1. Manager/Admin logs into the system
2. Manager/Admin clicks on get report of employee’s task button.
3. The system shows the Manager/Admin a list of employees to choose from.
4. Manager/Admin chooses an employee from the list and clicks get report button.
5. The system generates the employee’s report and displays the report.

Entry condition: Manager/Admin clicks on get report of employee’s task button.

Exit conditions: Successfully generating the report.

Quality requirement:

- System should respond in a timely manner.
- System interface gives meaningful feedbacks.

Use case name: request off day

Participating actors: Employee, Admin/Manager.

Flow of Events:

1. Employee logs into the system.
2. Employee clicks on request off day.
3. The system asks about the date of the off day.
4. The Employee selects a date interval.
5. The system creates a request to be sent to the Admin/Manager.
6. The Admin/Manager gets the request and gives his/her feedback.
7. The system updates the Employee request.

Entry condition: Employee/Receptionist clicks on request off day

Exit conditions: The system updates the Employee/Receptionist request.

Quality requirement:

- System interface gives meaningful feedbacks.
- Employee/Receptionist cannot request more than two weeks of break.
- Employee/Receptionist cannot request an off day in the current week of the request.

Use case name: Create employee shift

Participating actors: Manager

Flow of Events:

1. Manager logs into the system.
2. Manager clicks on Create employee shift button.
3. System shows the manager a list of all the employees.
4. The manager selects an employee to create a shift for.
5. The system shows the selected employee's working hours and days.
6. The manager chooses working hours for the shift.
7. The system updates the employee's schedule.

Entry condition: Manager clicks on Create employee shift button

Exit conditions: The system updates the employee's schedule, or if the employee does not have suitable hour for the shift.

Quality requirement:

- System interface gives meaningful feedbacks.
- The system returns the list of employees in seconds.
- The manager should be able to create the employee's shift without interfering with other employees' shifts.

Use case name: check employee's shift

Participating actors: Manager

Flow of Events:

1. Manager logs into the system.
2. Manager clicks on check shift button
3. System shows the manager a list of all the employees.
4. The manager selects an employee to check his shift.
5. The system shows the selected employee's working hours and days.

Entry condition: Manager clicks on check shift button

Exit conditions: The system shows the selected employee's working hours and days.

Quality requirement:

- System interface gives meaningful feedbacks.
- The system returns the list of employees in seconds.

Use case name: Check weekly schedule.

Participating actors: Employee

Flow of Events: 1. Employee logs into the system. 2. Employee clicks on weekly schedule. 3. The system shows the employee’s schedule.
Entry condition: Employee clicks on weekly schedule
Exit conditions: The system shows the employee’s schedule
Quality requirement: • System should respond in a short time of seconds. • System interface gives meaningful feedbacks.

Use case name: Create Guest account
Participating actors: Guest
Flow of Events: 1. The Guest opens the website. 2. Click the Create Account Button. 3. System shows a form with necessary information. 4. Guest enters necessary information and submits the form. 5. System validates if the guest already exists or not. 6. System saves guest to the database and redirects the guest to the login page.
Entry condition: The guest clicks on the register button.
Exit conditions: Trying to register an already existing guest or successfully registered.
Quality requirement: • System should respond in a short time of seconds. • System interface gives meaningful feedbacks.

Use case name: Create new user
Participating actors: System User

Flow of Events:

1.The user opens the website.
2.Click the Register User Button.
3.System shows a form with necessary information.
4.User enters necessary information and submits the form.
5.System validates if the user already exists or not.
6.System saves user information and redirects him/her to the dashboard.

Entry condition: Clicking the register button.

Exit conditions: Trying to register an already existent user or successfully registered.

Quality requirement:

- System should respond in a short time of seconds.
- System interface gives meaningful feedbacks.

Use case name: Edit employee’s information

Participating actors: Manager

Flow of Events:

1. Manager logs in to the system.
2. Manager clicks on show employees’ list.
3. The system shows the list of all the employees.
4. Manager chooses the employee from the employee list.
5. System shows the chosen Employee’s detailed information.
6. Manager edits the information wanted on that page.
7. Manager clicks on the save button to save the changes on employee’s information.
8. System saves the changes on the screen and shows them to the Manager.

Entry condition: Clicking edit employee’s information button.

Exit conditions: Entering invalid format of information to different areas, deleting necessary information, or successfully editing and saving the information tab.

Quality requirement:

- System should respond in a short time of seconds.
- System interface gives meaningful feedbacks.
- System must take the compulsory information.

Use case name: Refund payment

Participating actors: Guest, Manager

Flow of Events:

1. Guest logs in to the system with his/her user ID and password.
2. Guest clicks on the refund button.
3. System opens the cancellation tab according to hotels policy.
4. System shows a form to guest about cancellation.
5. Guest fills the form and clicks on the save button.
6. System saves and sends this request form to the Manager.
7. According to hotel policy and hotel rules Manager can approve the form.
8. Once the system gets the approval from the manager, refunds the payment to the Guest, and deletes the reservation from the system.

Entry condition: Clicking cancel reservation button.

Exit conditions: Cancellation is not possible according to hotel policy and rules. Or request successfully sent to the Manager.

Quality requirement:

- System should respond in a short time of seconds.
- System interface gives meaningful feedback
- In case of refunding, system should send the money to Guest’s account within a reasonable time interval.

Use case name: Request room service

Participating actors: Guest, Staff

Flow of Events:

1. Guest logs in to the system with his/her user ID and password.
2. Guest clicks on the request room service button.
3. System sends this request to the available Staff page.
4. System adds the task to the Staff’s task list.
5. Staff goes to the Guest’s room with necessary items according to the request.
6. Staff access their page and mark the new task as complete.
7. The guest’s bill is updated in accordance with the service provided.

Entry condition: Clicking request room service button.

Exit conditions: Request successfully added to the current task list.

Quality requirement:

- System should respond in a short time of seconds.
- System interface gives meaningful feedbacks.

Use case name: Check task for specific time interval

Participating actors: Employee

Flow of Events:

1. Employee clicks the see All Tasks Button.
2. System shows two inputs for starting and end-date.
3. Employee enters the starting date and end-date.
4. System creates a task list for specified time interval.
5. System displays the list to the employee.

Entry condition: Clicking see All Tasks button.

Exit conditions: System show the list for specific time interval, or no invalid date was given.

Quality requirement:

- System should respond in few seconds
- System interface gives meaningful feedbacks.

Use case name: Request Room Cleaning

Participating actors: Guest, Staff

Flow of Events:

1. Guest logs in to the system with his/her user ID and password.
2. Guest clicks on the request room Cleaning button.
3. System sends this request to the Staff's page.
4. System adds the task to the Staff 's task list.
5. Staff goes to the Guest's room with necessary items according to the request.
6. Employee accesses to his/her page and marks the new task as complete.

Entry condition: Guest clicks on the request room Cleaning button

Exit conditions: The task is successfully added to the tasks list of the Staff.

Quality requirement:

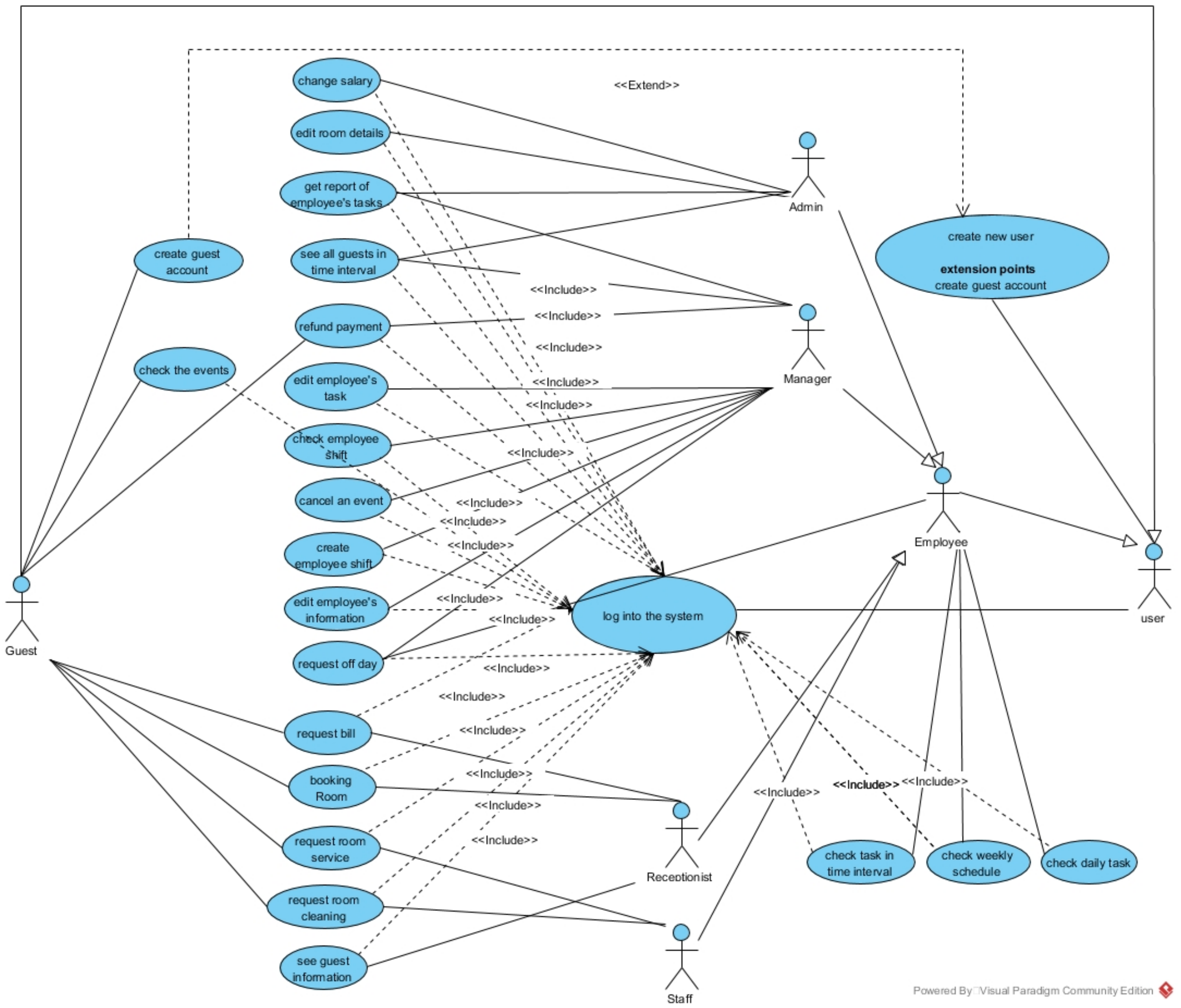
- System should respond in a short time
- System interface gives meaningful feedbacks.

Use case name: Request Bill

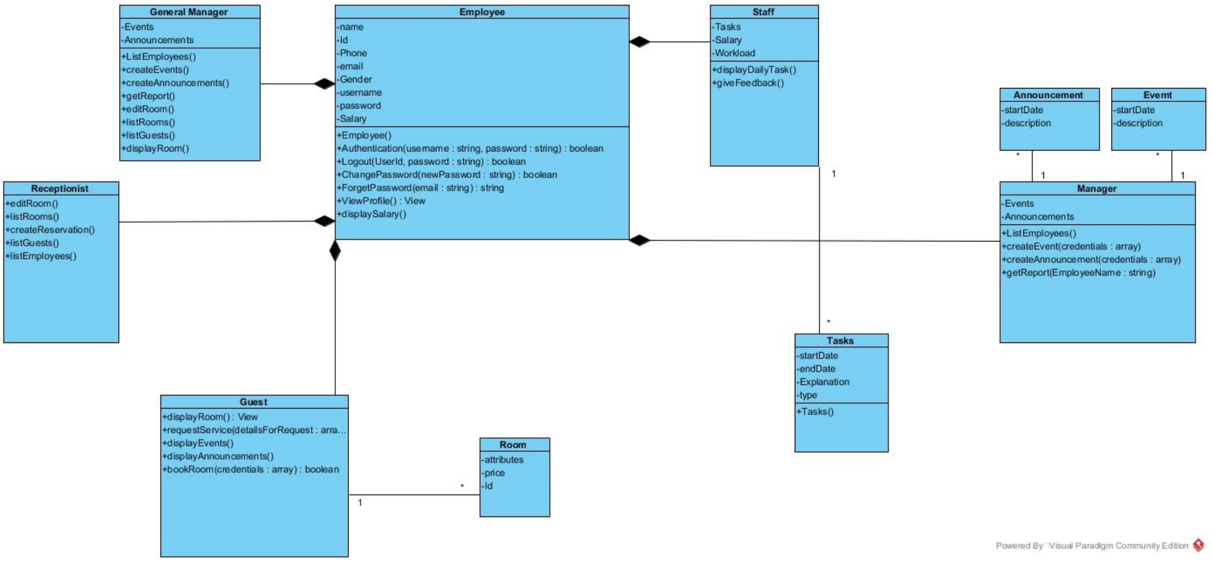
Participating actors: Guest, Receptionist

Flow of Events: 1. Guest logs into the system 2. The guest clicks on request bill button. 3. The system sends the request to the receptionist. 4. The receptionist accesses the guest's request. 5. The receptionist approves the guest's request. 6. The system sends a bill to the guest's account.
Entry condition: Guest clicks request bill button
Exit conditions: Bill successfully sent to the Guest, there is no such a bill.
Quality requirement: • System should respond in few seconds • System interface gives meaningful feedbacks.

<Hotel Management System>



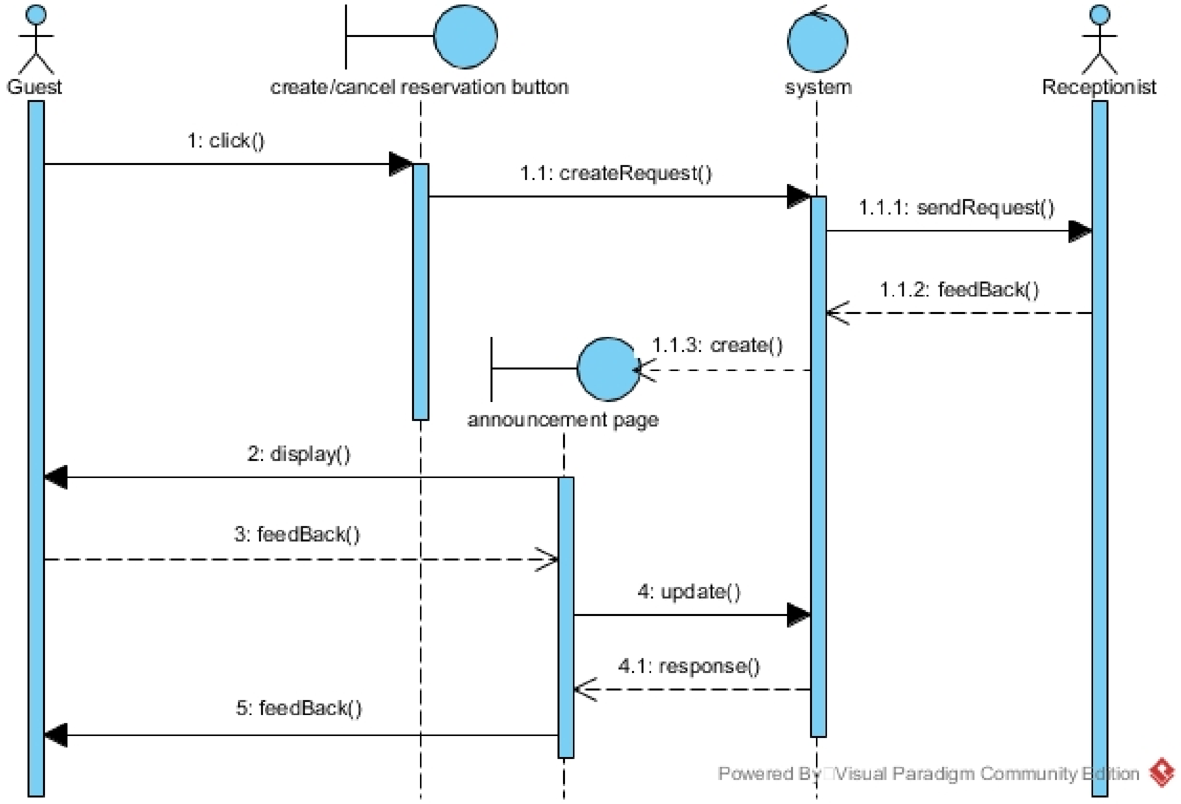
Object model



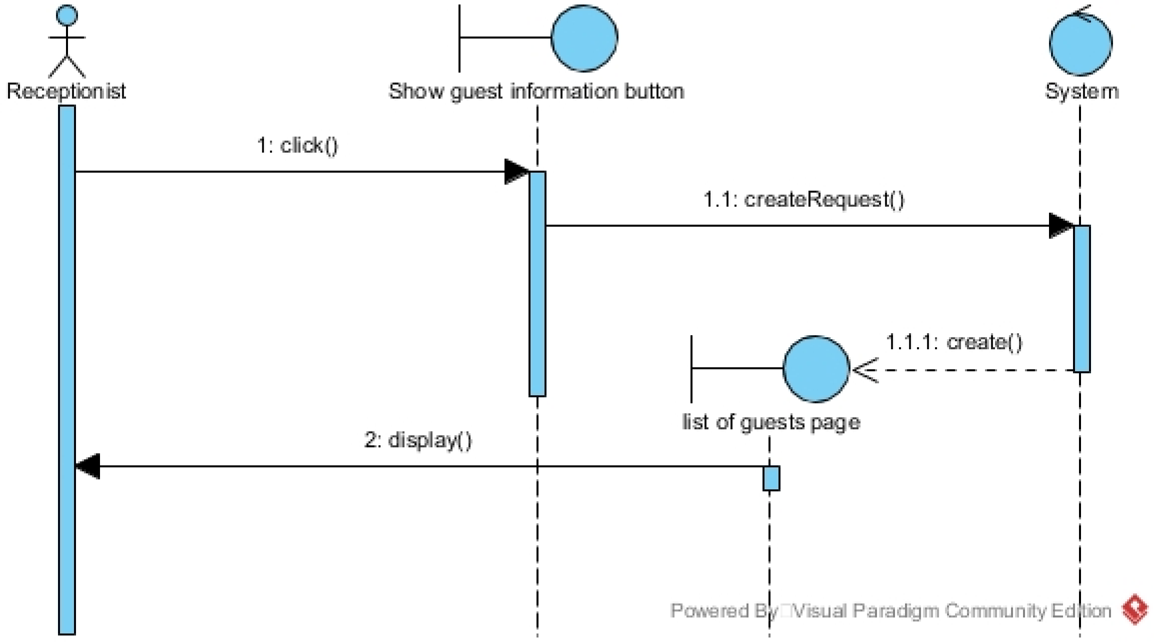
Dynamic model

1.booking room

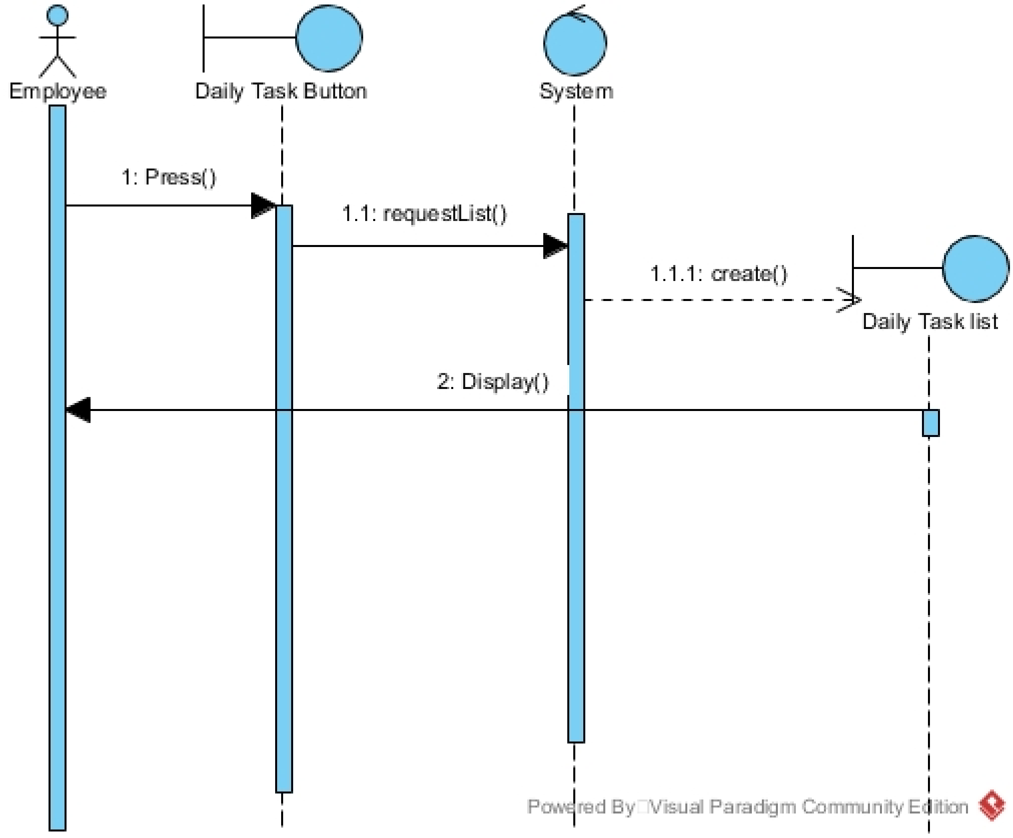
<Hotel Management System>



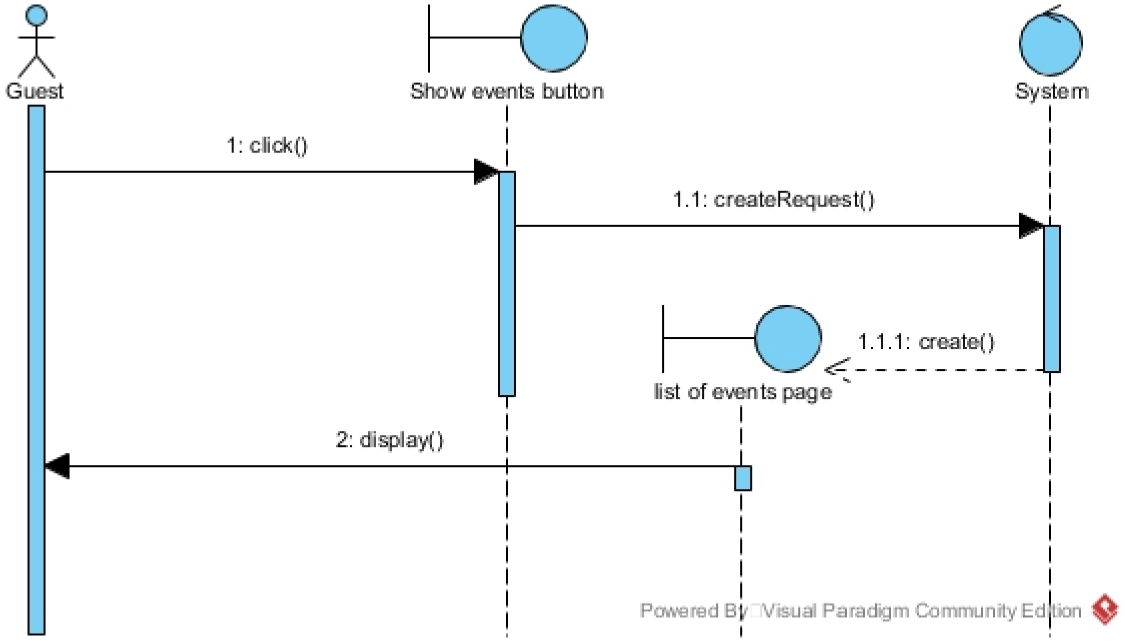
2.See guest information



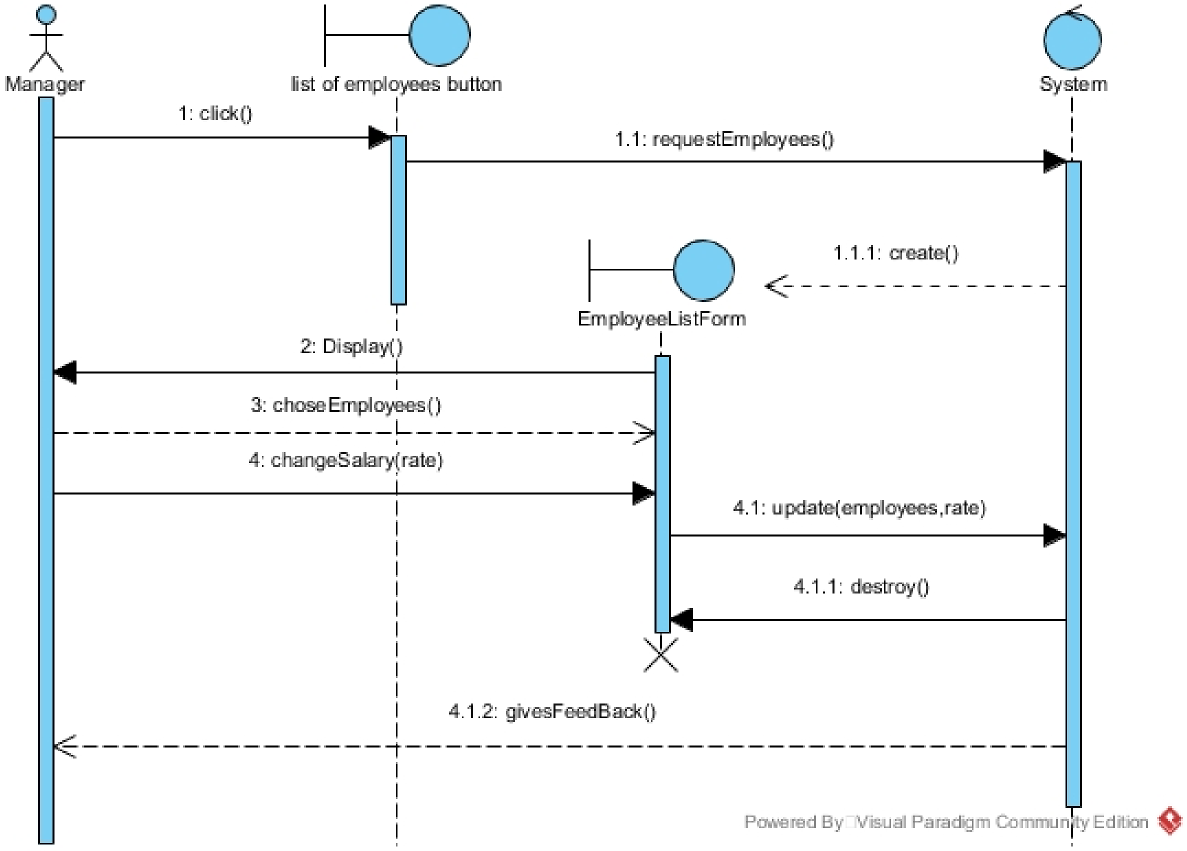
3.check daily tasks



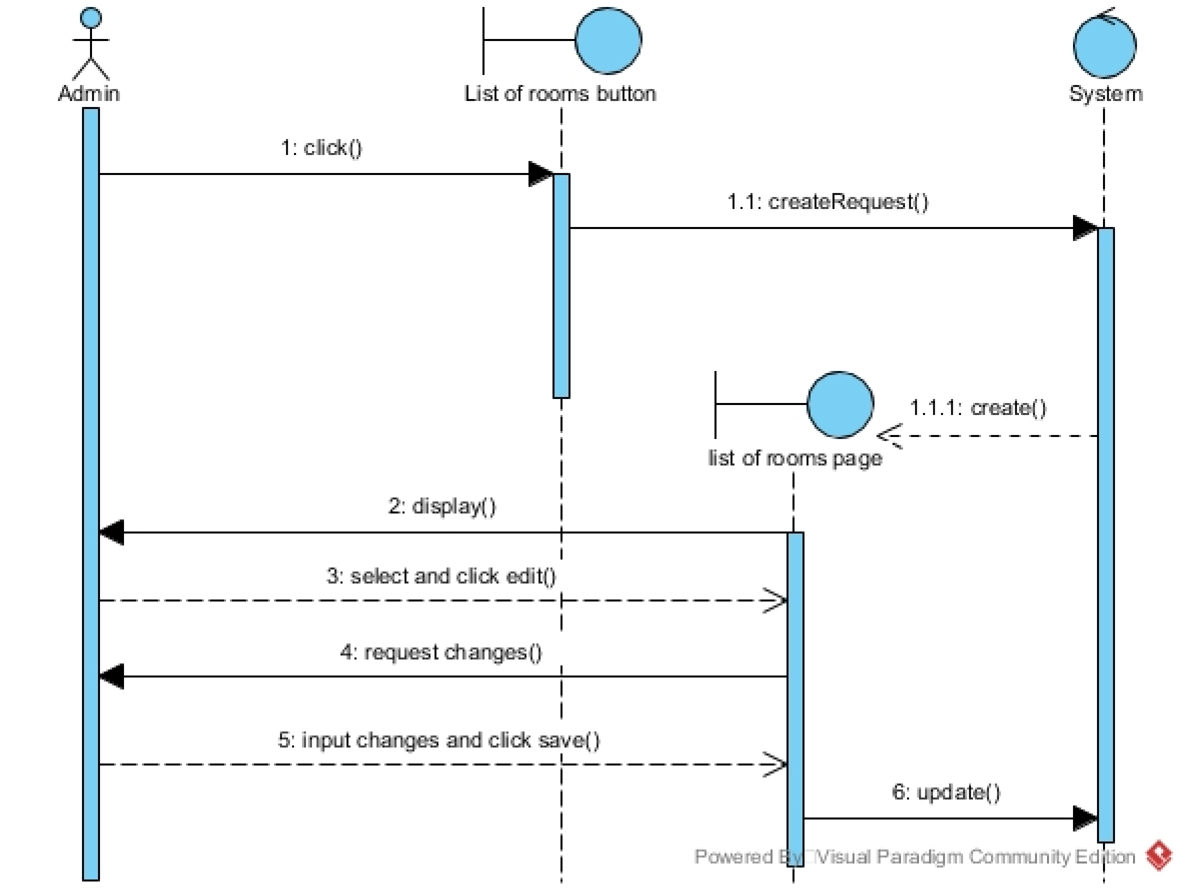
4.check the events



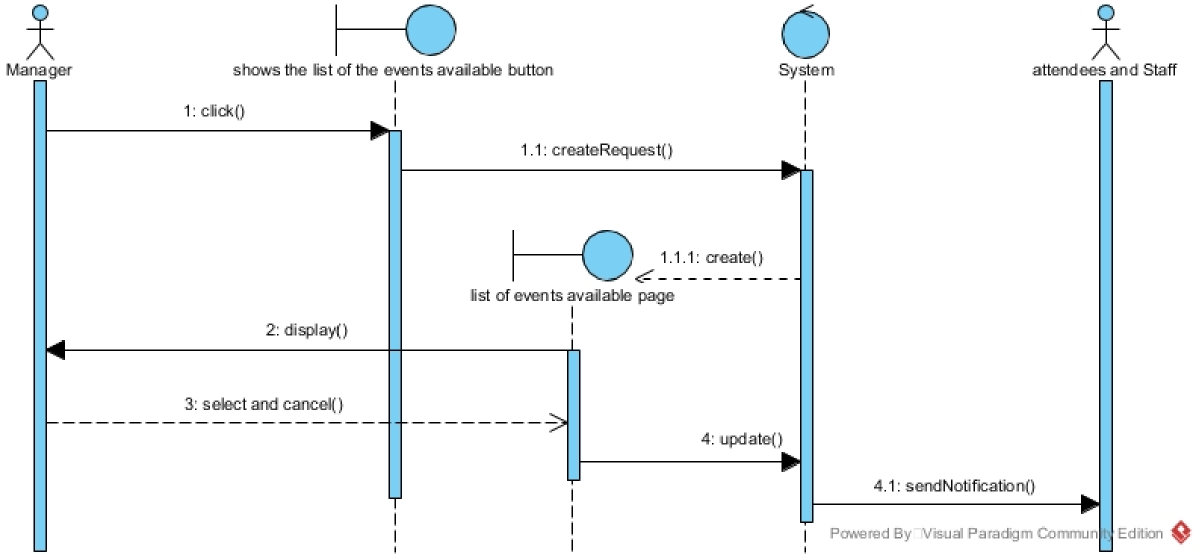
5.change salary



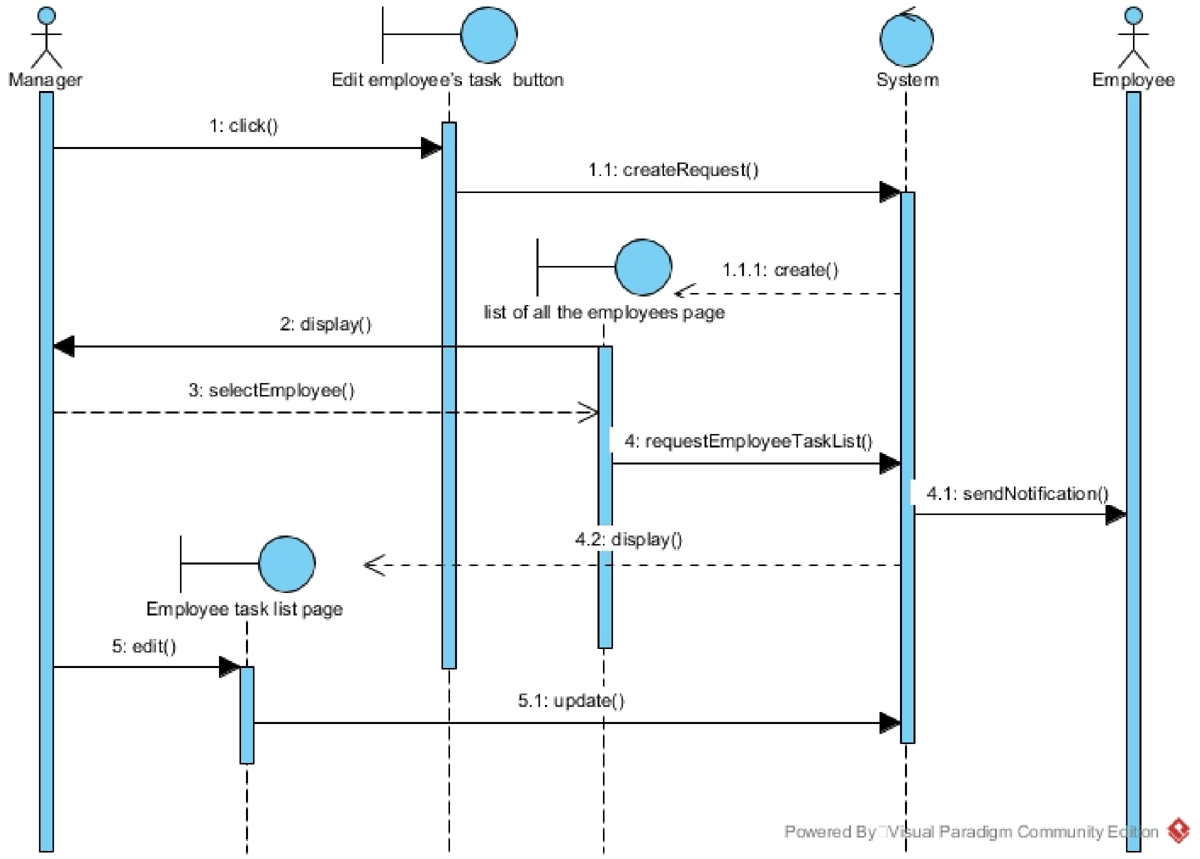
6.edit room details



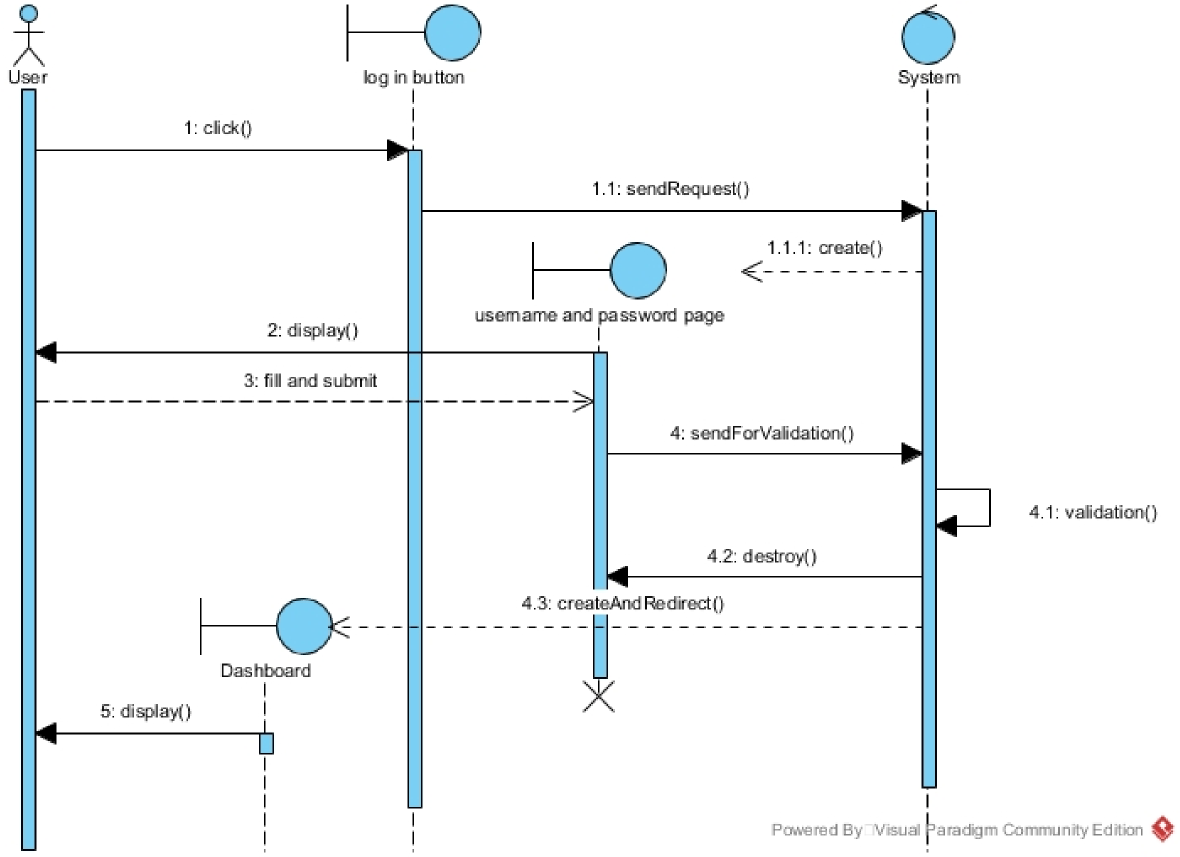
7.cancel an event



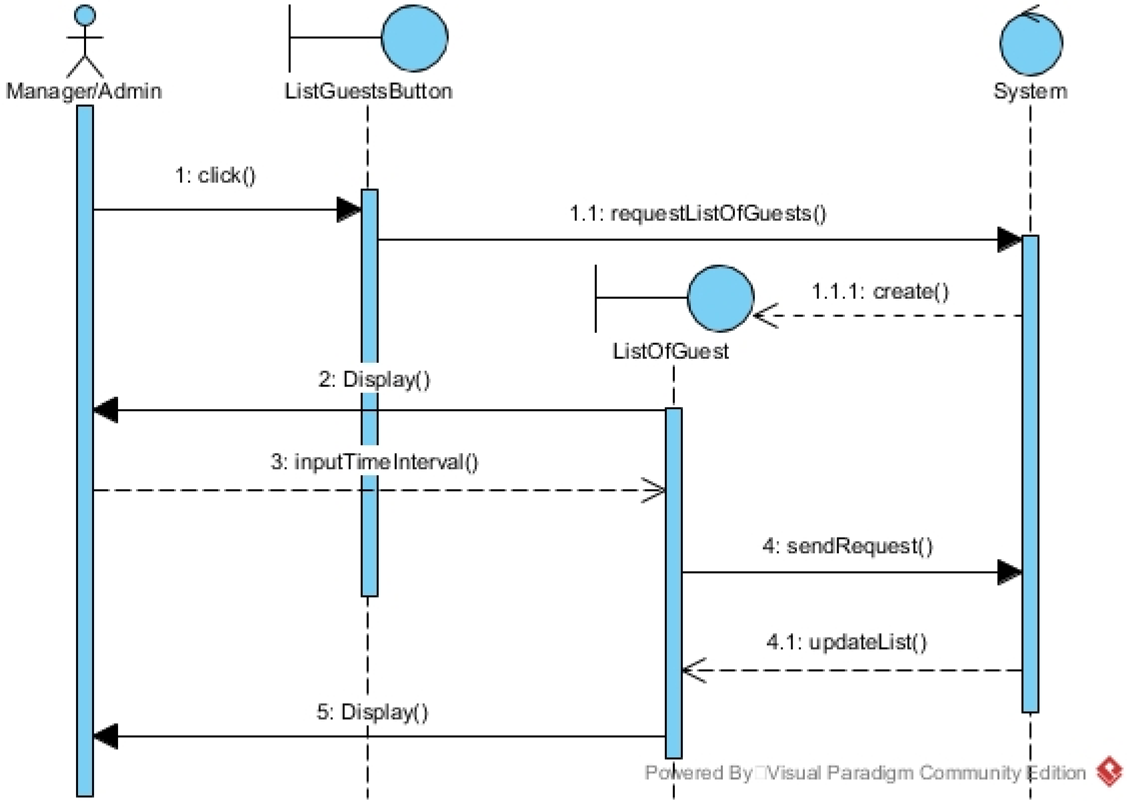
8.edit employee's task



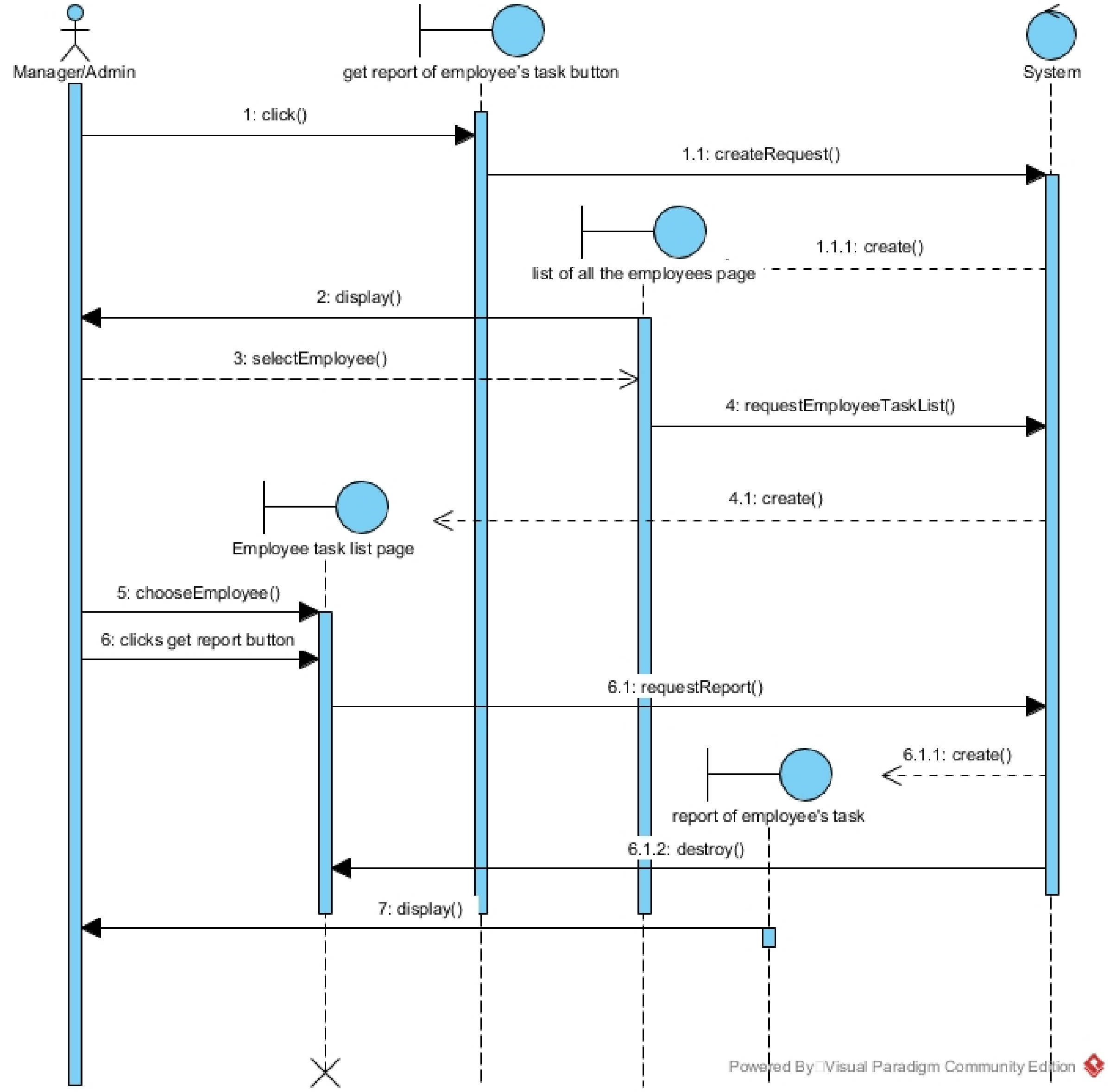
9.log into the system



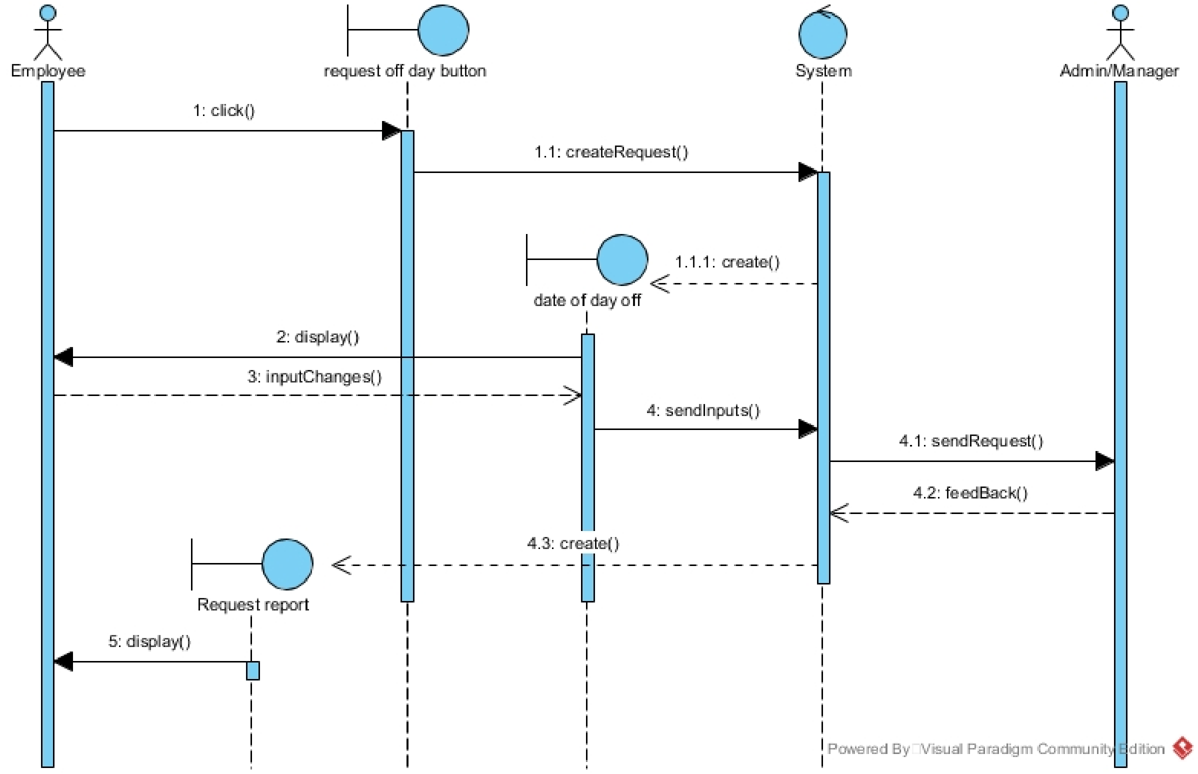
10.See all Guests in a Specific Time Interval



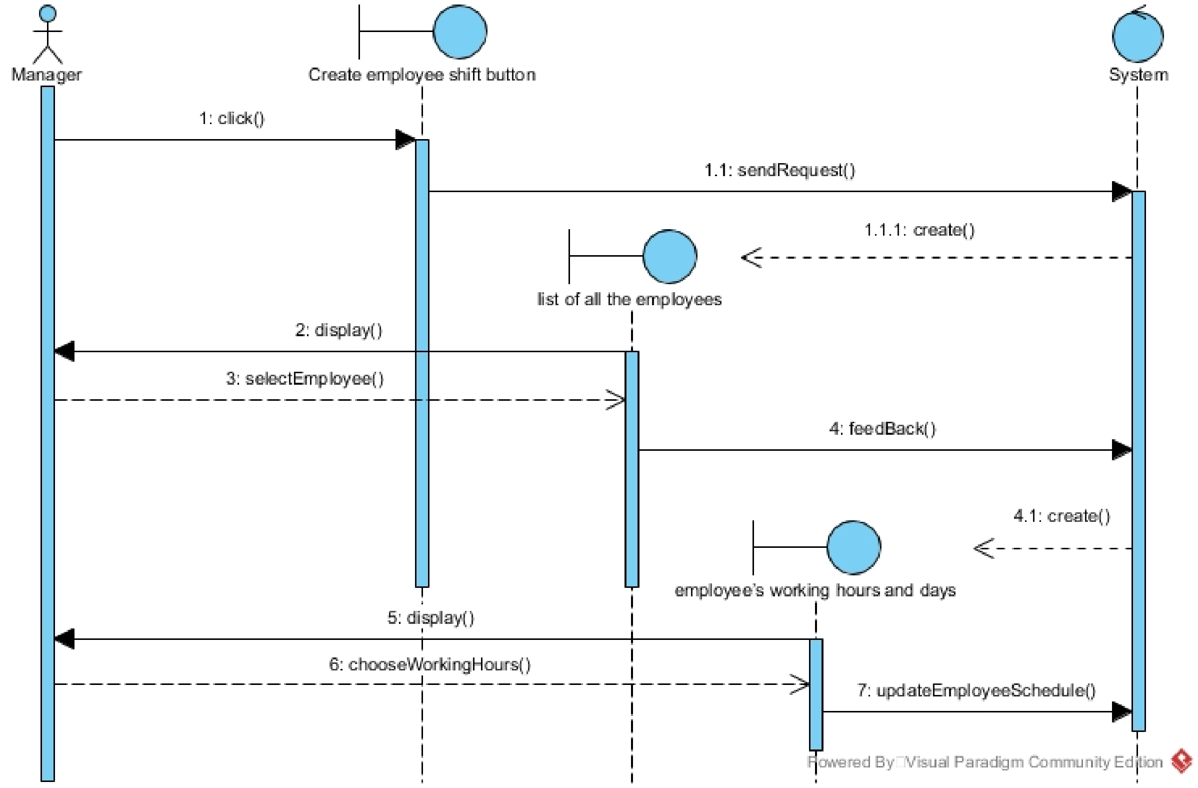
11.get report of employee's tasks



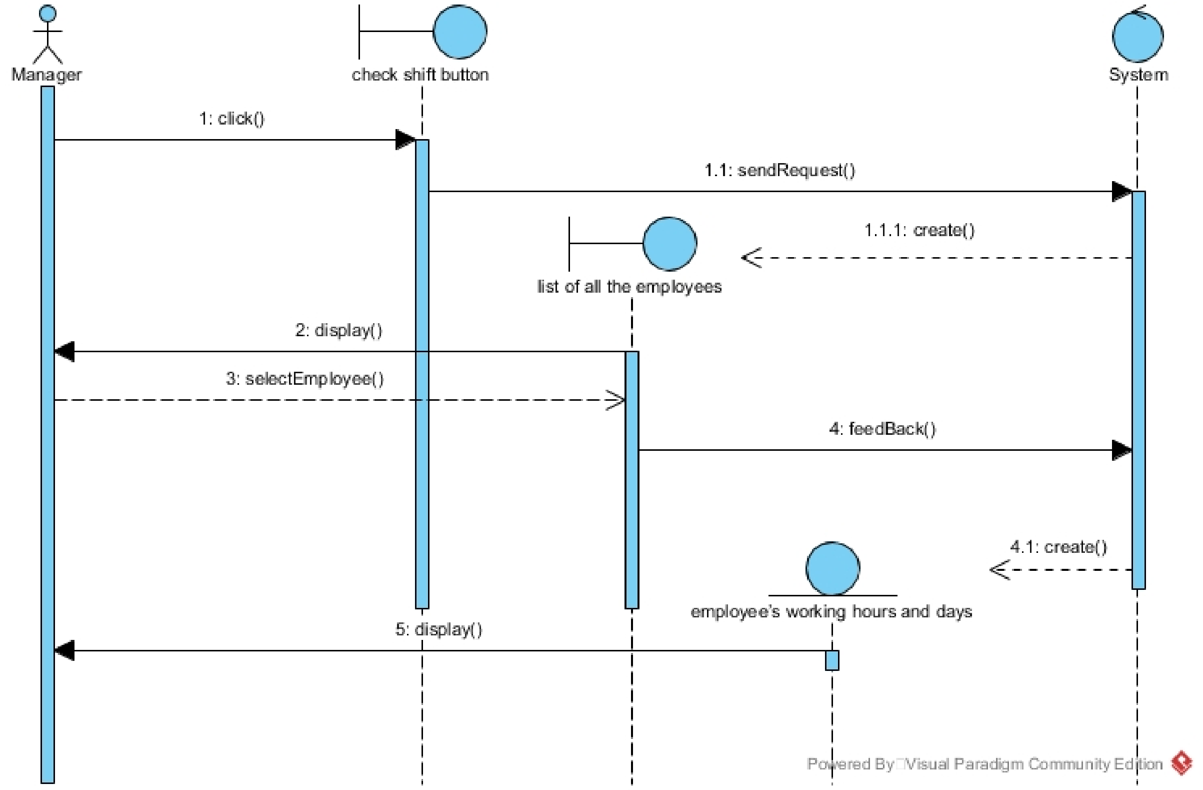
12.request day off



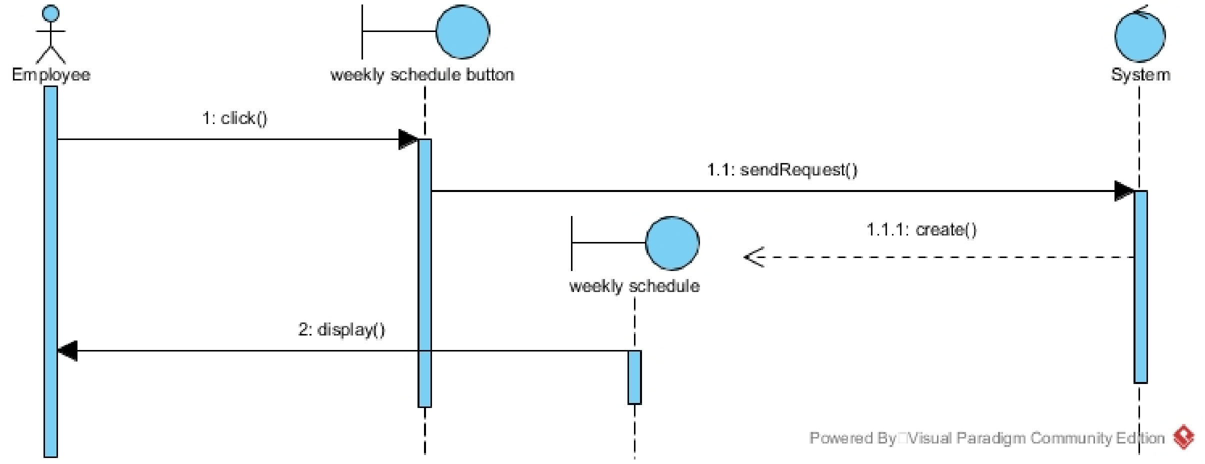
13.Create employee shift


Manager
Create employee shift button
System
1: click()
1.1: sendRequest()
1.1.1: create()
list of all the employees
2: display()
3: selectEmployee()
4: feedBack()
4.1: create()
employee's working hours and days
5: display()
6: chooseWorkingHours()
7: updateEmployeeSchedule()
Powered By Visual Paradigm Community Edition

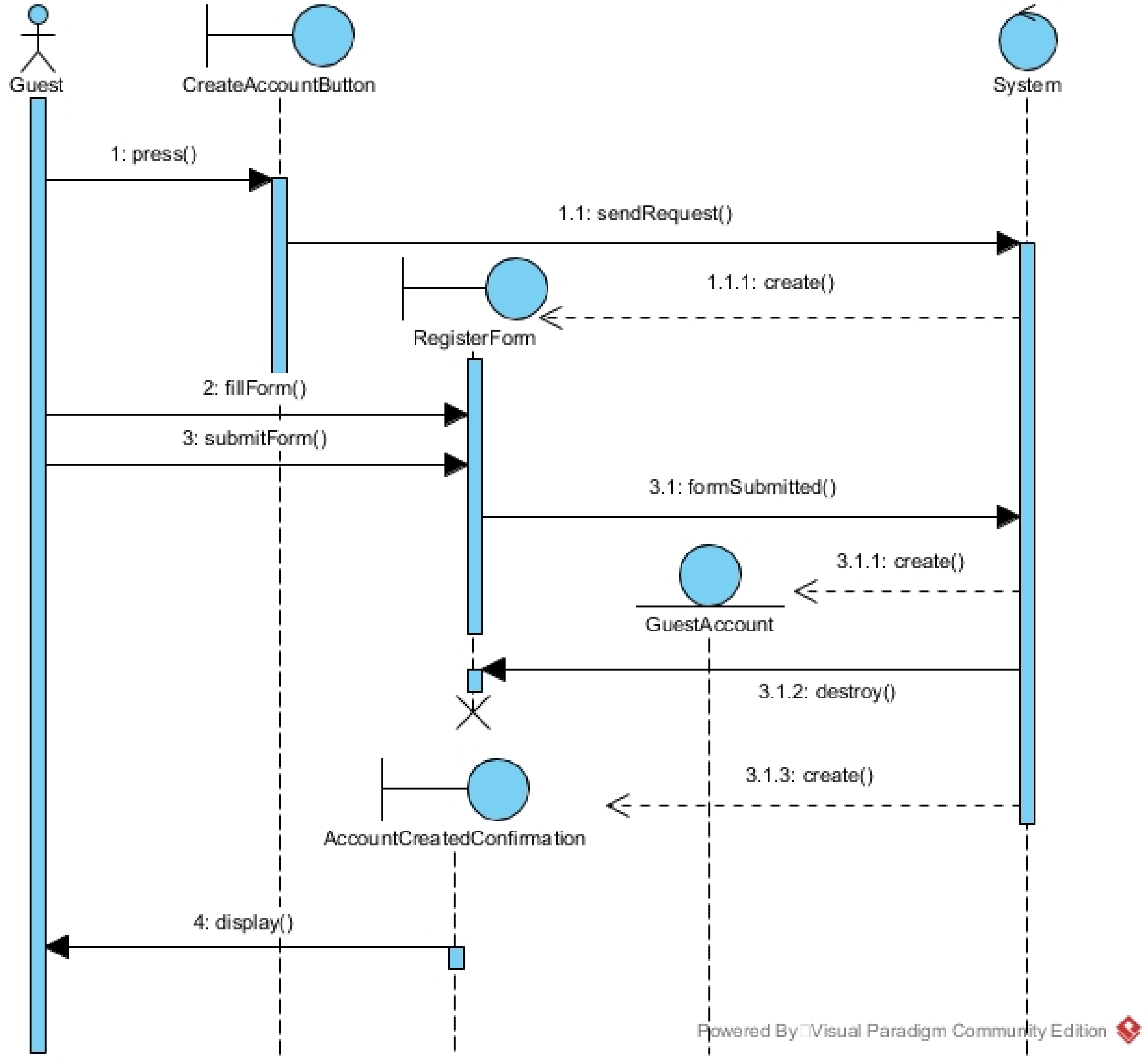
14.check employee's shift



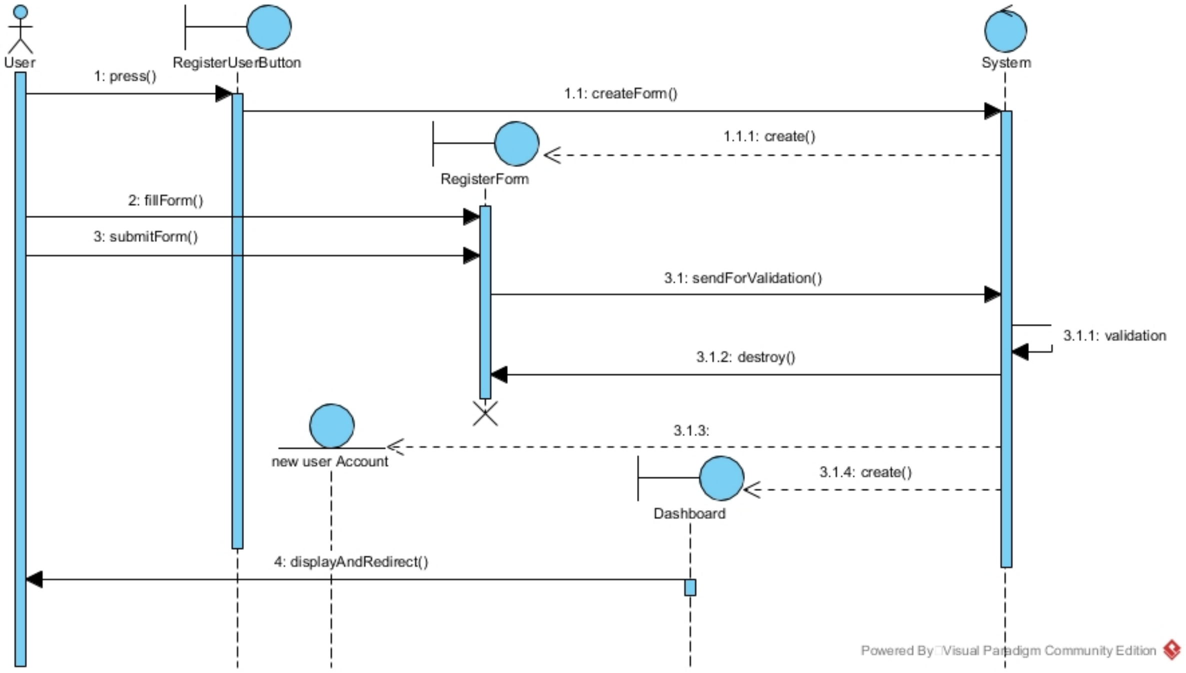
15.check weekly schedule



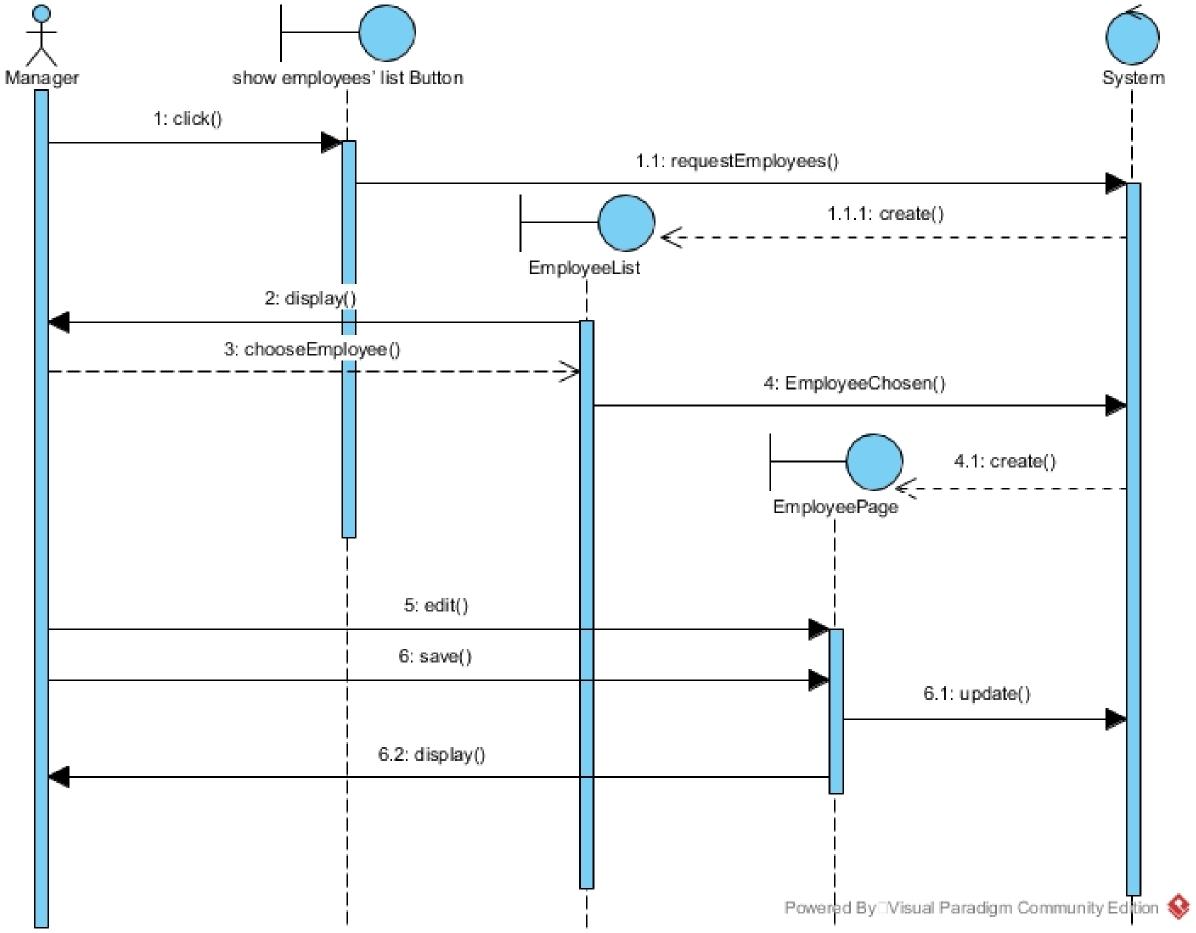
16.create guest account



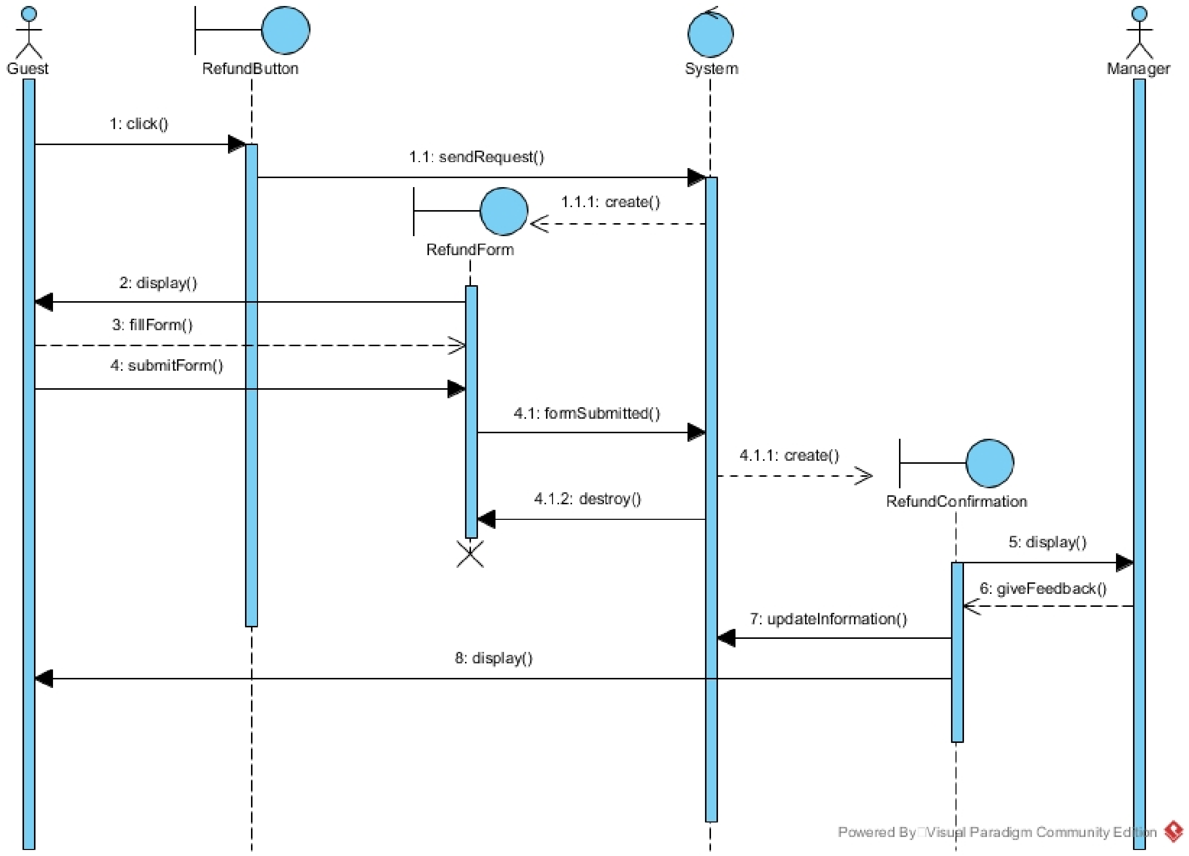
17.create new user



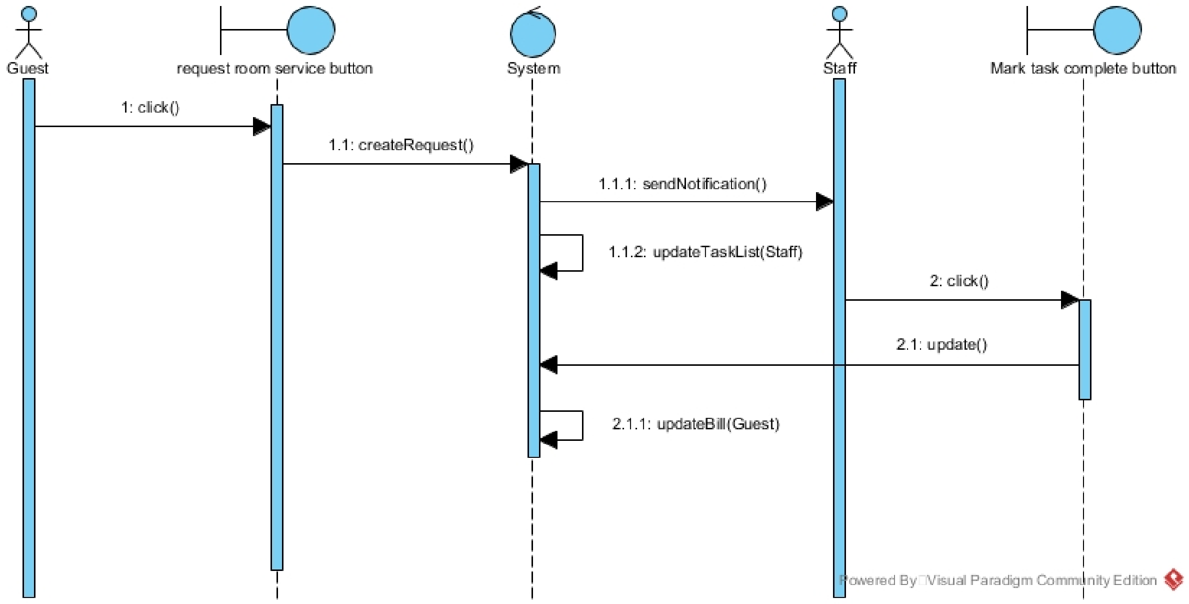
18.edit employee's information



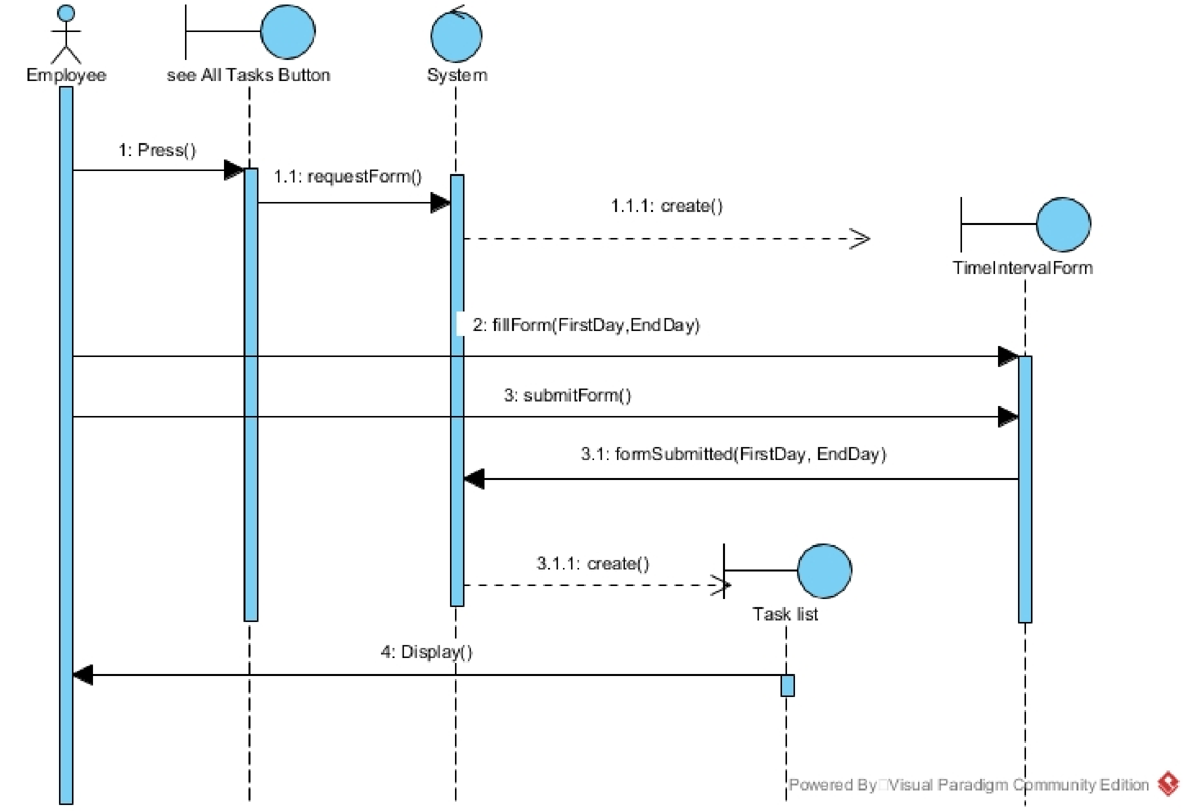
19.refund payment



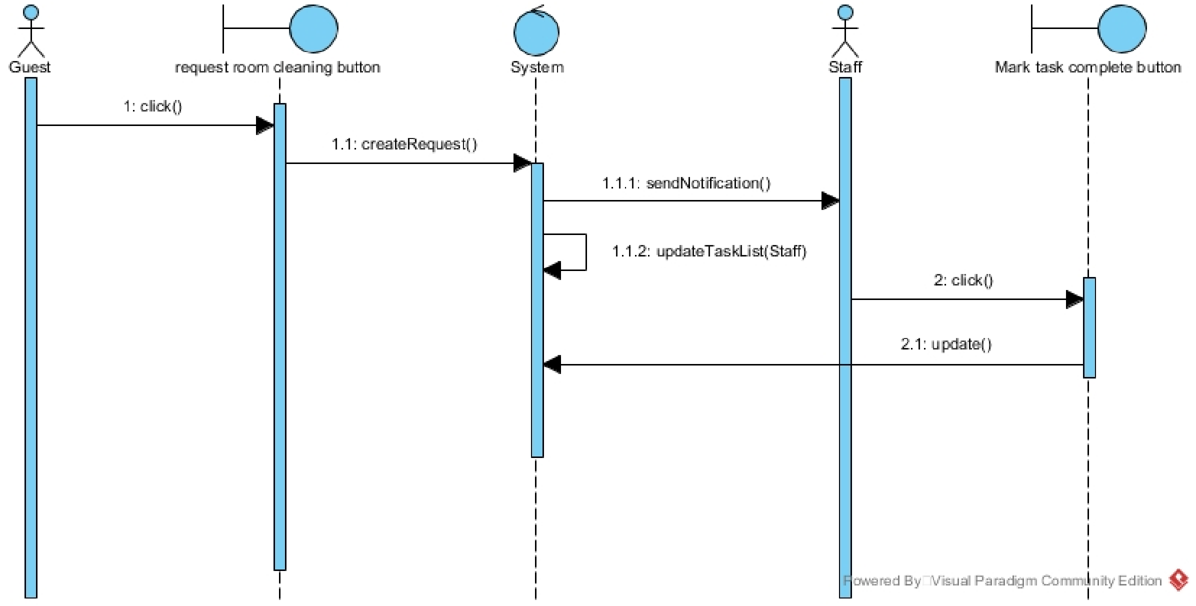
20.Request room Service


Guest
request room service button
System
Staff
Mark task complete button
1: click()
1.1: createRequest()
1.1.1: sendNotification()
1.1.2: updateTaskList(Staff)
2: click()
2.1: update()
2.1.1: updateBill(Guest)
Powered By Visual Paradigm Community Edition

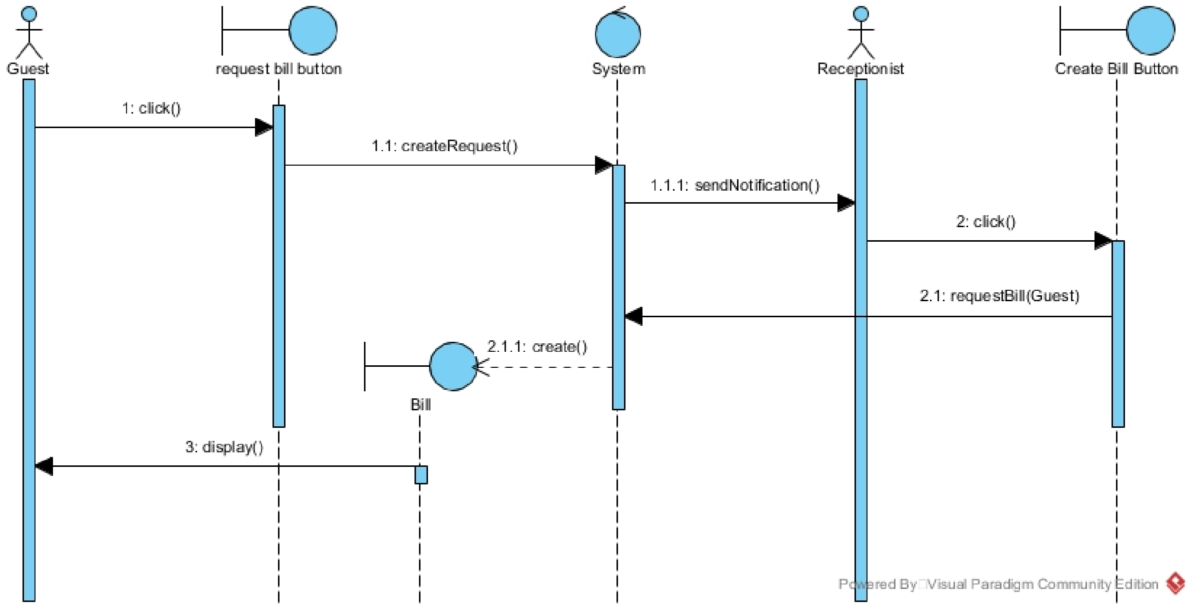
21.check tasks for specific time interval



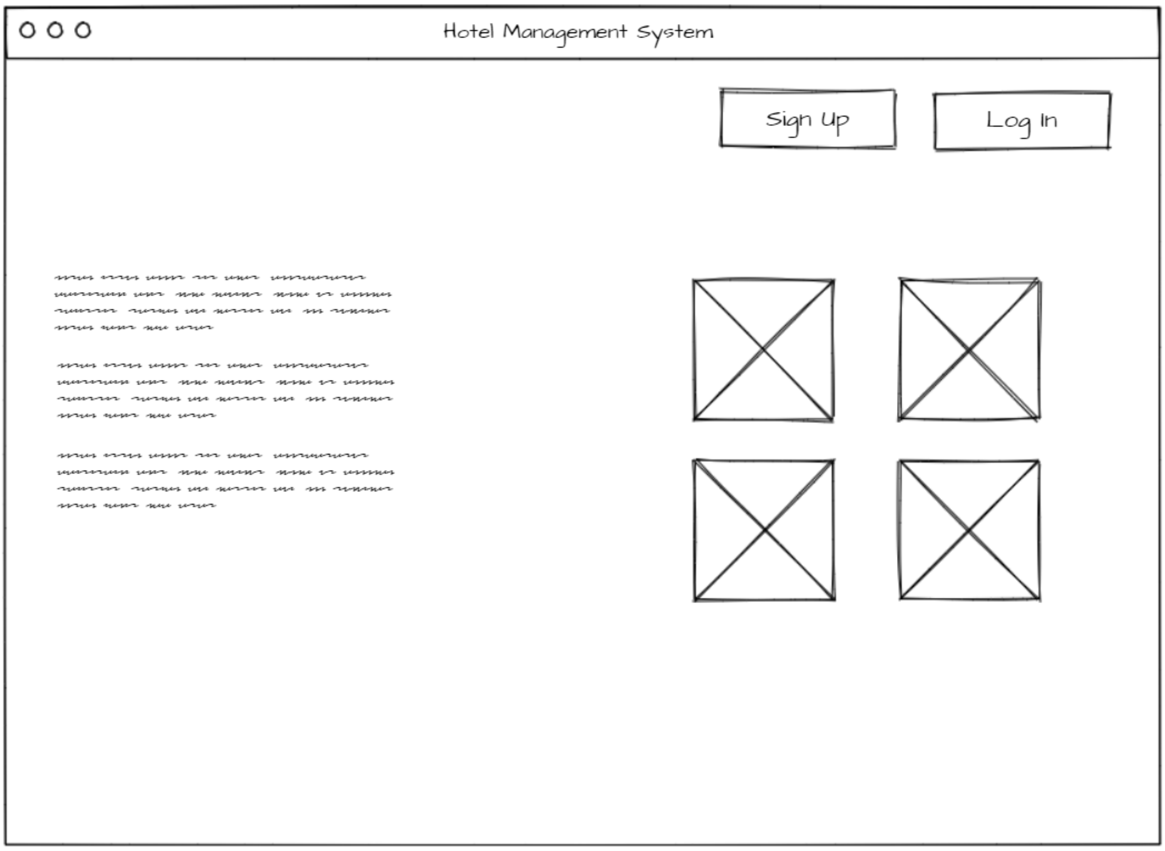
22.Request room cleaning

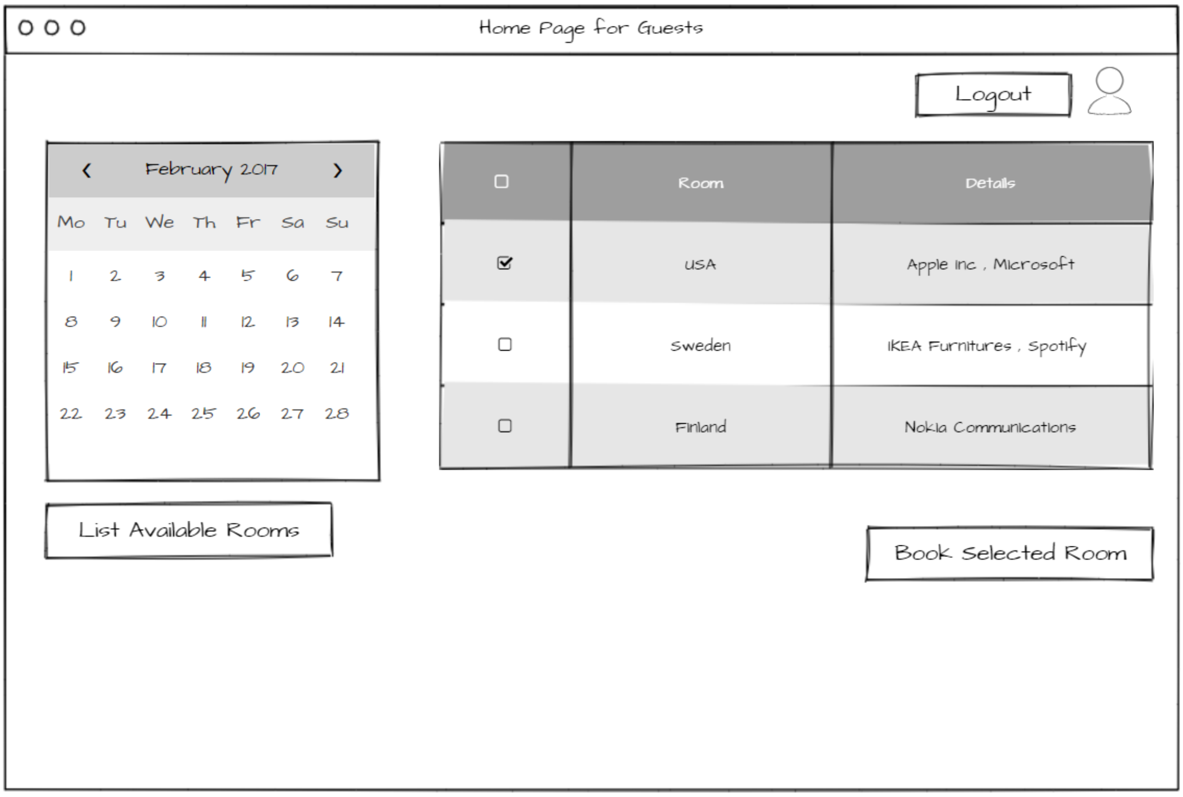

Guest
request room cleaning button
System
Staff
Mark task complete button
1: click()
1.1: createRequest()
1.1.1: sendNotification()
1.1.2: updateTaskList(Staff)
2: click()
2.1: update()
Powered By Visual Paradigm Community Edition

23.request bill



User interface—navigational paths and screen mock-ups

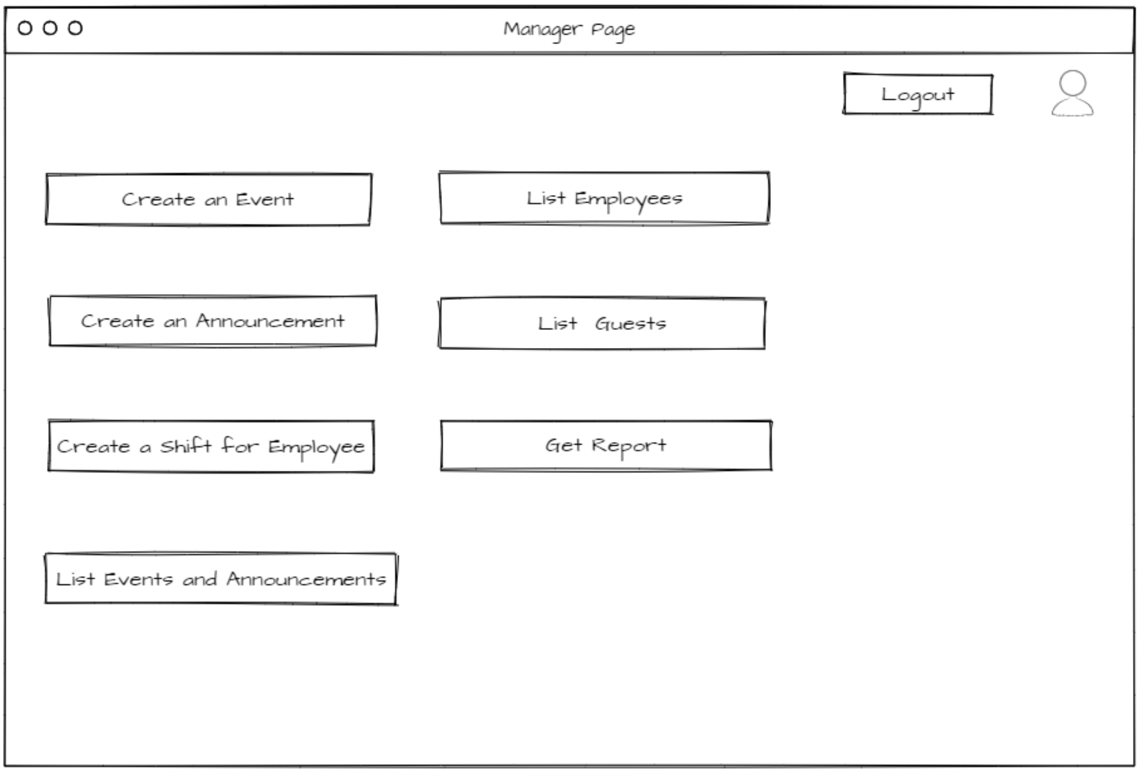


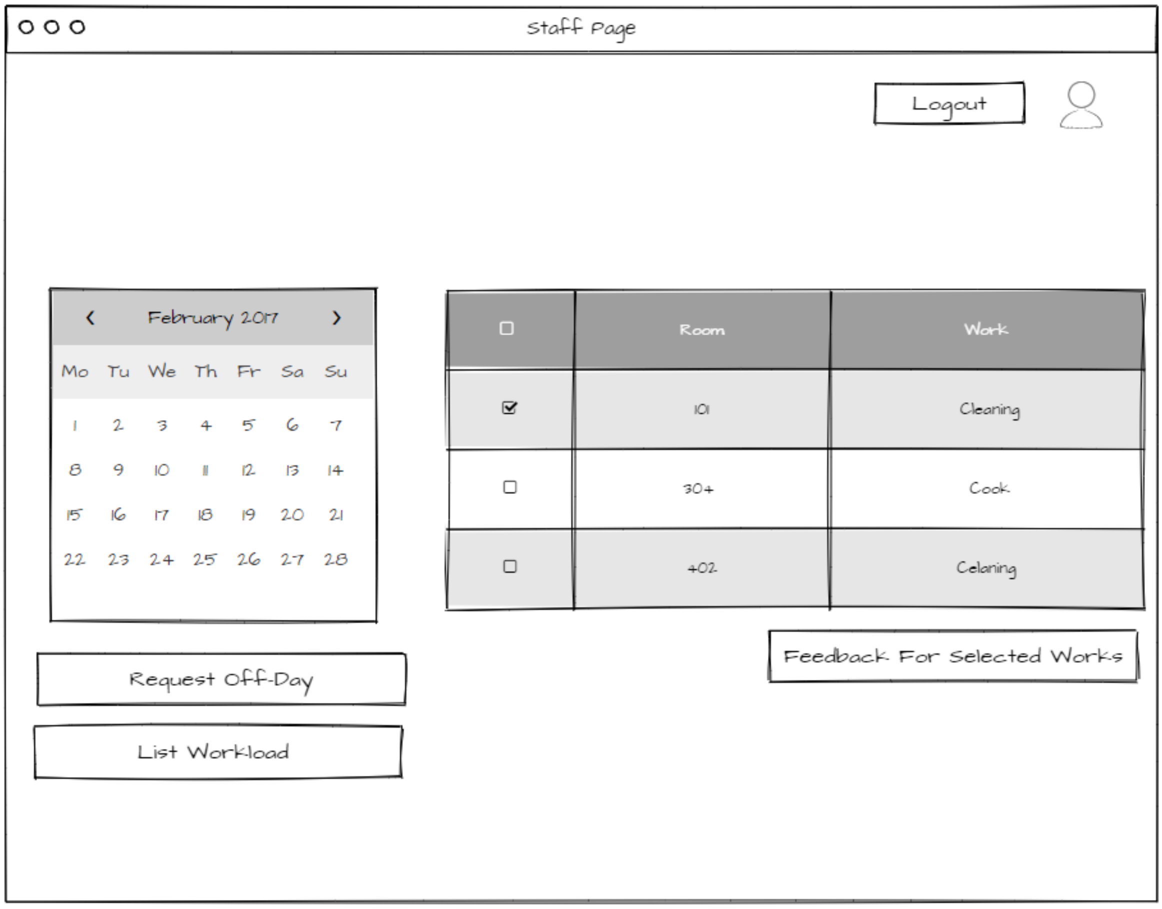


February 2017

Mo	Tu	We	Th	Fr	Sa	Su
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28

□	Room	Details
☑	USA	Apple Inc , Microsoft
□	Sweden	IKEA Furnitures , Spotify
□	Finland	Nokia Communications

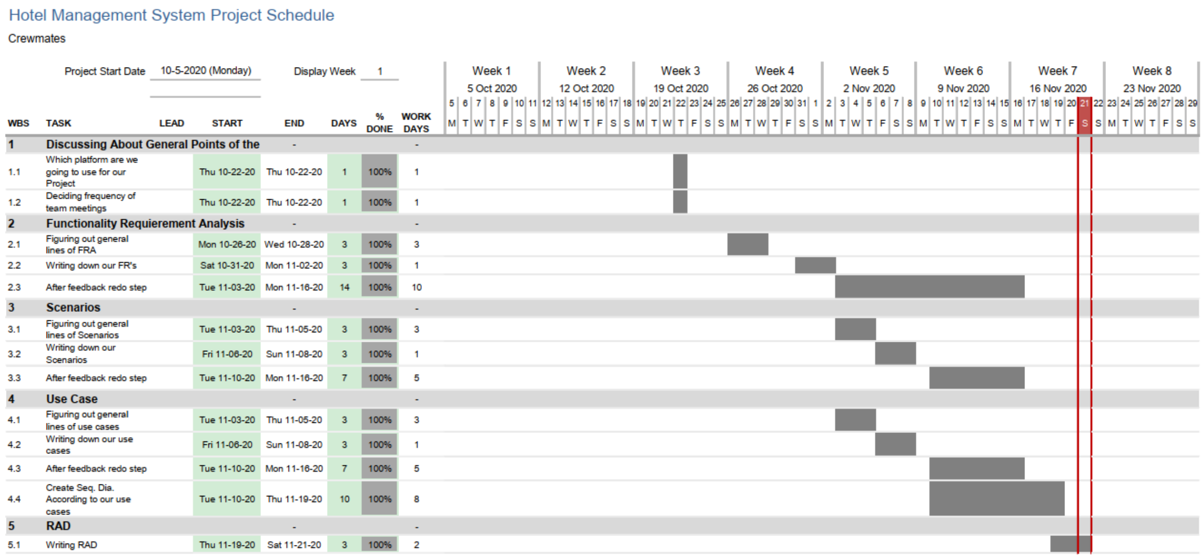

Manager Page
Logout
Create an Event
List Employees
Create an Announcement
List Guests
Create a Shift for Employee
Get Report
List Events and Announcements



‹			February 2017			›
Mo	Tu	We	Th	Fr	Sa	Su
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28

☐	Room	Work
☑	101	Cleaning
☐	304	Cook
☐	402	Celaning

3.5. Project Schedule



Hotel Management System Project Schedule

Crewmates

Project Start Date 10-5-2020 (Monday) Display Week 1

WBS	TASK	LEAD	START	END	DAYS	% DONE	WORK DAYS
1	**Discussing About General Points of the**			-			-
1.1	Which platform are we going to use for our Project		Thu 10-22-20	Thu 10-22-20	1	100%	1
1.2	Deciding frequency of team meetings		Thu 10-22-20	Thu 10-22-20	1	100%	1
2	**Functionality Requierement Analysis**			-			-
2.1	Figuring out general lines of FRA		Mon 10-26-20	Wed 10-28-20	3	100%	3
2.2	Writing down our FR's		Sat 10-31-20	Mon 11-02-20	3	100%	1
2.3	After feedback redo step		Tue 11-03-20	Mon 11-16-20	14	100%	10
3	**Scenarios**			-			-
3.1	Figuring out general lines of Scenarios		Tue 11-03-20	Thu 11-05-20	3	100%	3
3.2	Writing down our Scenarios		Fri 11-06-20	Sun 11-08-20	3	100%	1
3.3	After feedback redo step		Tue 11-10-20	Mon 11-16-20	7	100%	5
4	**Use Case**			-			-
4.1	Figuring out general lines of use cases		Tue 11-03-20	Thu 11-05-20	3	100%	3
4.2	Writing down our use cases		Fri 11-06-20	Sun 11-08-20	3	100%	1
4.3	After feedback redo step		Tue 11-10-20	Mon 11-16-20	7	100%	5
4.4	Create Seq. Dia. According to our use cases		Tue 11-10-20	Thu 11-19-20	10	100%	8
5	**RAD**			-			-
5.1	Writing RAD		Thu 11-19-20	Sat 11-21-20	3	100%	2

4. Glossary

User: any person who uses the system.
Guest: any person with the intention to spend time in the hotel and use it’s services.

<Hotel Management System>

Employee: any person working at the hotel in any capacity.
Admin: administrator of the hotel also known as General manager.
Manager: manages staff and receptionists and deals with events hosted by the hotel, deals with guests in cases where he is needed.
Receptionist: deals with guest related matters and helps guests when necessary.
Staff: Can be cleaners, bellboys, waiters, cooks, and chefs.

5. References

1. Bruegge B. & Dutoit A.H.. (2010). *Object-Oriented Software Engineering Using UML, Patterns, and Java*, Prentice Hall, 3rd ed.
2. http://www.cs.sjsu.edu/faculty/pearce/modules/lectures/ooa/requirements/IdentifyingURPS.htm
3. Ian Sommerville (2015). Software Engineering, Pearson tenth edition.