

STUDENT : Popa Ioan-Ciprian

PROJECT for SOFTWARE ENGINEERING LABORATORY

APPLICATION TITLE Employee Management System

1. APPLICATION DESCRIPTION

This project is about building a basic employee management system easy to use and accessible by multiple roles of employees.

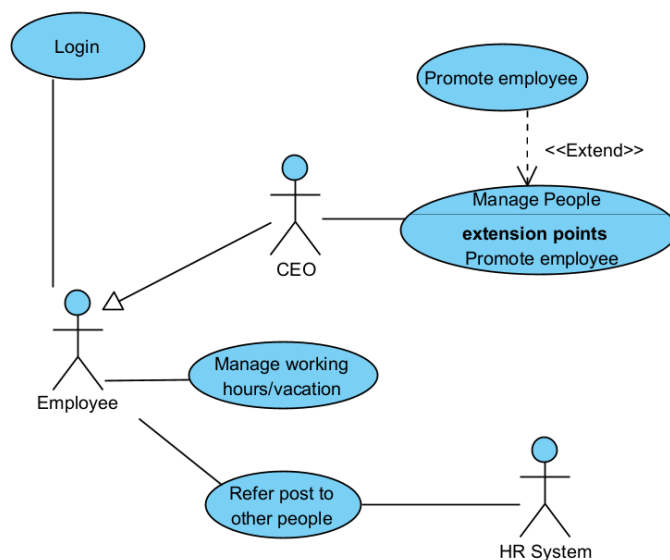
Employees can use the system for:

- Referring the post to unemployed people
- Manage their working hours/vacation

CEO's can use the system for:

- Promoting/demoting people's post to a higher/lower rank
- Hire/Fire employees

2. USE CASE DIAGRAM



3. USE CASE DETAILS : (For each UC)

UC name: Refer Employee

Actor(s) : Employee

Description : This use case describes the process by which an employee can refer another person for a job within the company.

Preconditions : - The employee must be logged into the system.

- The employee must have the necessary permissions to refer other individuals for job positions.
- The employee must know the details of the person they want to refer, including their contact information and relevant qualifications.

(Sequence diagrams at system level, specifying which flow is modelled in each of them, main flow and/or alter native flows)

1.The employee logs into the system.

2.The employee navigates to the "Refer Employee" section of the system.

3.The system presents a form for the employee to input the details of the person they want to refer, including name, contact information, and qualifications.

4.The employee fills out the form with the required information.

5.The employee submits the referral.

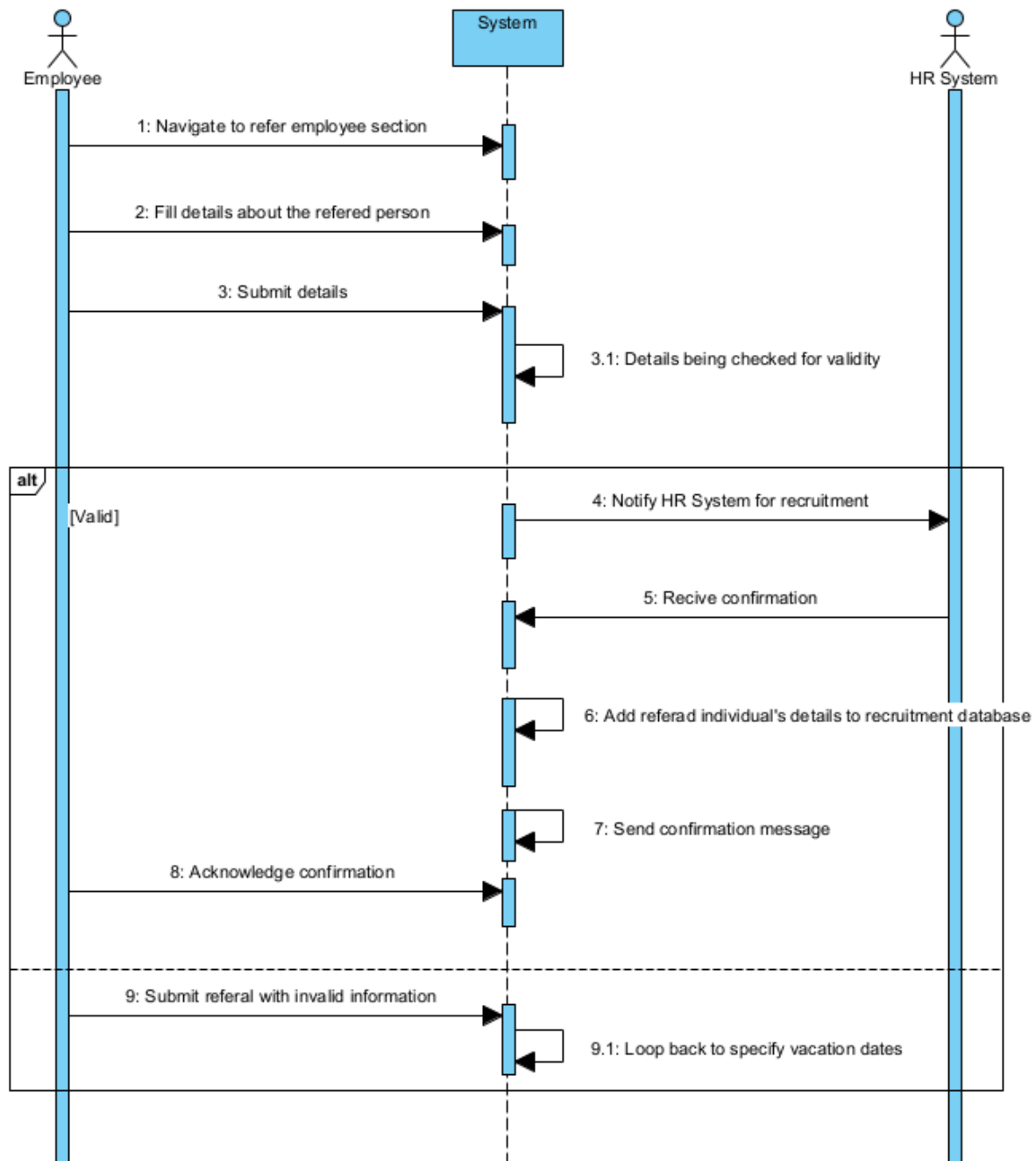
6.The system processes the referral and notifies the HR responsible for recruitment.

7.The referred individual's details are added to the recruitment database for further consideration.

Postconditions : - The referral is recorded in the system.

- The referred individual's details are available for review by the recruitment team.
- The employee receives a confirmation message indicating that the referral has been successfully submitted.

Alternative flows: - If the employee enters invalid information in the referral form (e.g., missing required fields, incorrect contact information), the system displays an error message.



Use Case Name: Manage Working Hours/Vacation

Actor(s): Employee

Description: This use case describes the process by which an employee can manage their working hours and request vacation time within the company.

Preconditions: - The employee must be logged into the system.

- The employee must have the necessary permissions to manage their working hours and request vacation time.
- The employee's current working schedule and available vacation balance should be up-to-date in the system.

Sequence Diagram (Main Flow):

- 1.The employee logs into the system.
- 2.The employee navigates to the "Manage Working Hours" or "Request Vacation" section of the system.
- 3.If managing working hours:
 - The system displays the employee's current working schedule.
 - The employee selects the option to modify their working hours.
 - The employee adjusts their working hours by specifying new start and end times or adding/removing shifts.
 - The employee submits the changes.
 - The system updates the employee's working schedule accordingly.
- 4.If requesting vacation:
 - The system displays the employee's available vacation balance.
 - The employee selects the option to request vacation time.
 - The employee specifies the dates for their vacation request.
 - The employee submits the vacation request.
 - The system processes the request and checks for conflicts with existing schedules or company policies.
 - If the request is approved, the employee's vacation balance is adjusted, and their absence is recorded in the system.
 - If the request is denied, the employee receives a notification with the reason for denial.

Postconditions: -For managing working hours:

- The employee's working schedule is updated according to the changes made.

For requesting vacation:

- If approved, the employee's vacation balance is adjusted, and their absence is recorded.
- If denied, the employee's vacation balance remains unchanged, and they receive a notification of denial.

Sequence Diagram (Alternative Flow - Conflicting Vacation Request):

- 1.If the requested vacation dates conflict with existing schedules or company policies:
 - The system notifies the employee of the conflict.
 - The employee revises the vacation request with alternative dates.
 - The system reevaluates the request based on the new dates.
 - The process continues until a non-conflicting request is submitted.

Use Case: Manage Working Hours

Use Case Name: Manage Working Hours

Actor(s): Employee

Description: This use case describes the process by which an employee can manage their working hours within the company.

Preconditions:

- The employee must be logged into the system.

Main Flow:

1. The employee logs into the system.
2. The employee navigates to the "Manage Working Hours" section of the system.
3. The system displays the employee's current working schedule.
4. The employee selects the option to modify their working hours.
5. The employee adjusts their working hours by specifying new start and end times or adding/removing shifts.
6. The employee submits the changes.
7. The system updates the employee's working schedule accordingly.
8. The system confirms the update to the employee.

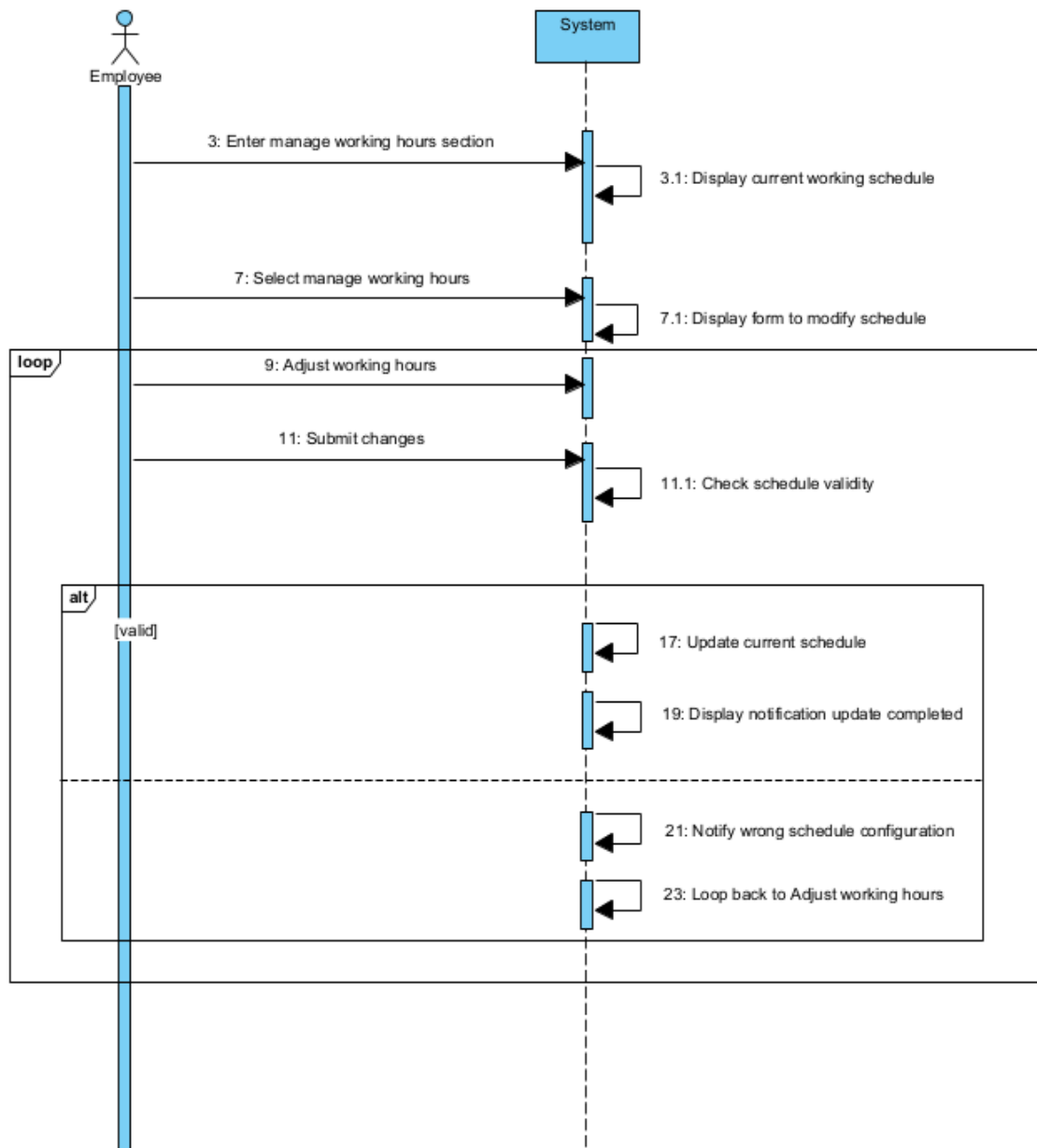
Alternative Flow:

6. Invalid schedule

Loop Back to adjust working hours page

Postconditions:

- The employee's working schedule is updated according to the changes made.



Use Case: Request Vacation

Use Case Name: Request Vacation

Actor(s): Employee

Description: This use case describes the process by which an employee can request vacation time within the company.

Preconditions:

- The employee must be logged into the system.
- The employee must have the necessary permissions to request vacation time.
- The employee's available vacation balance should be up-to-date in the system.

Main Flow:

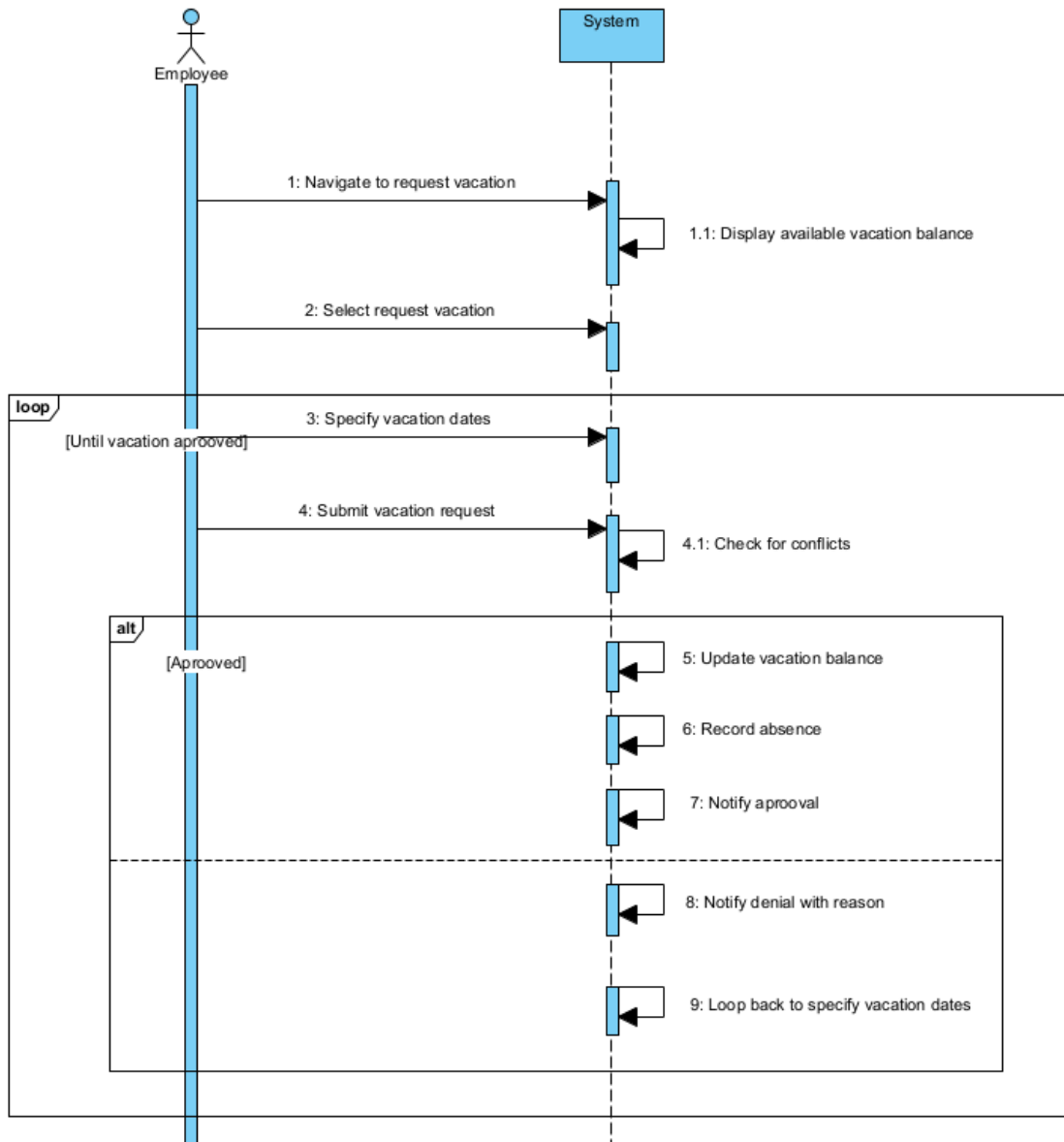
1. The employee navigates to the "Request Vacation" section of the system.
2. The system displays the employee's available vacation balance.
3. The employee selects the option to request vacation time.
4. The employee specifies the dates for their vacation request.
5. The employee submits the vacation request.
6. The system processes the request and checks for conflicts with existing schedules or company policies.
7. If the request is approved, the employee's vacation balance is adjusted, and their absence is recorded in the system.
8. If the request is denied, the employee receives a notification with the reason for denial.

Postconditions:

- If approved, the employee's vacation balance is adjusted, and their absence is recorded.
- If denied, the employee's vacation balance remains unchanged, and they receive a notification of denial.

Alternative Flow - Conflicting Vacation Request:

1. If the requested vacation dates conflict with existing schedules or company policies:
 - The system notifies the employee of the conflict.
 - The employee revises the vacation request with alternative dates.
 - The system reevaluates the request based on the new dates.
 - The process continues until a non-conflicting request is submitted or the employee decides not to proceed with the vacation request.



Use Case: Manage Employee Status

Use Case Name: Manage Employee Status

Actor(s): CEO

Description: This use case describes the process by which the CEO can hire, fire, promote,

or demote an employee within the company.

Preconditions:

- The CEO must be logged into the system.
- The CEO must have the necessary permissions to manage employee status.
- The employee data in the system should be up-to-date.

Main Flow:

1. The CEO logs into the system.
2. The CEO navigates to the "Manage Employee Status" section of the system.
3. The CEO selects one of the following options: Hire, Fire, Promote, Demote.

Alternative Flow 1: Hire

1. The system displays the hiring form.
2. The CEO fills out and submits the hiring form.
3. The system processes the hiring by adding the new employee to the database and setting initial details.
4. The system confirms the hiring to the CEO.

Postconditions for Hire:

- The new employee is added to the system with initial details set.

Alternative Flow 2: Fire

1. The system displays the employee list.
2. The CEO selects an employee to fire.
3. The system prompts the CEO for confirmation.
4. The CEO confirms the firing.
5. The system processes the firing by removing the employee from the database and updating records.
6. The system confirms the firing to the CEO.

Postconditions for Fire:

- The selected employee is removed from the system, and records are updated accordingly.

Alternative Flow 3: Promote

1. The system displays the employee list.
2. The CEO selects an employee to promote.
3. The system displays the promotion form.
4. The CEO fills out and submits the promotion form.
5. The system processes the promotion by updating the employee's position and adjusting salary/benefits.
6. The system confirms the promotion to the CEO.

Postconditions for Promote:

- The selected employee's position is updated, and their salary/benefits are adjusted accordingly.

Alternative Flow 4: Demote

1. The system displays the employee list.
2. The CEO selects an employee to demote.
3. The system displays the demotion form.
4. The CEO fills out and submits the demotion form.
5. The system processes the demotion by updating the employee's position and adjusting salary/benefits.
6. The system confirms the demotion to the CEO.

Postconditions for Demote:

- The selected employee's position is updated, and their salary/benefits are adjusted accordingly.

Sequence Diagram (Main Flow)

1. **Log In:**
 - CEO -> System: Log In
2. **Select Option:**
 - CEO -> System: Navigate to "Manage Employee Status"
 - CEO -> System: Select Option (Hire, Fire, Promote, Demote)

Sequence Diagram (Alternative Flow 1: Hire)

1. **Display Hiring Form:**
 - System -> CEO: Display Hiring Form
2. **Submit Hiring Form:**
 - CEO -> System: Fill Out and Submit Hiring Form
3. **Process Hiring:**
 - System -> System: Process Hiring (Add new employee to database, set initial details)
4. **Confirm Hiring:**
 - System -> CEO: Confirm Hiring

Sequence Diagram (Alternative Flow 2: Fire)

1. **Display Employee List:**
 - System -> CEO: Display Employee List
2. **Select Employee to Fire:**
 - CEO -> System: Select Employee to Fire
3. **Prompt for Confirmation:**
 - System -> CEO: Prompt for Confirmation
4. **Confirm Firing:**

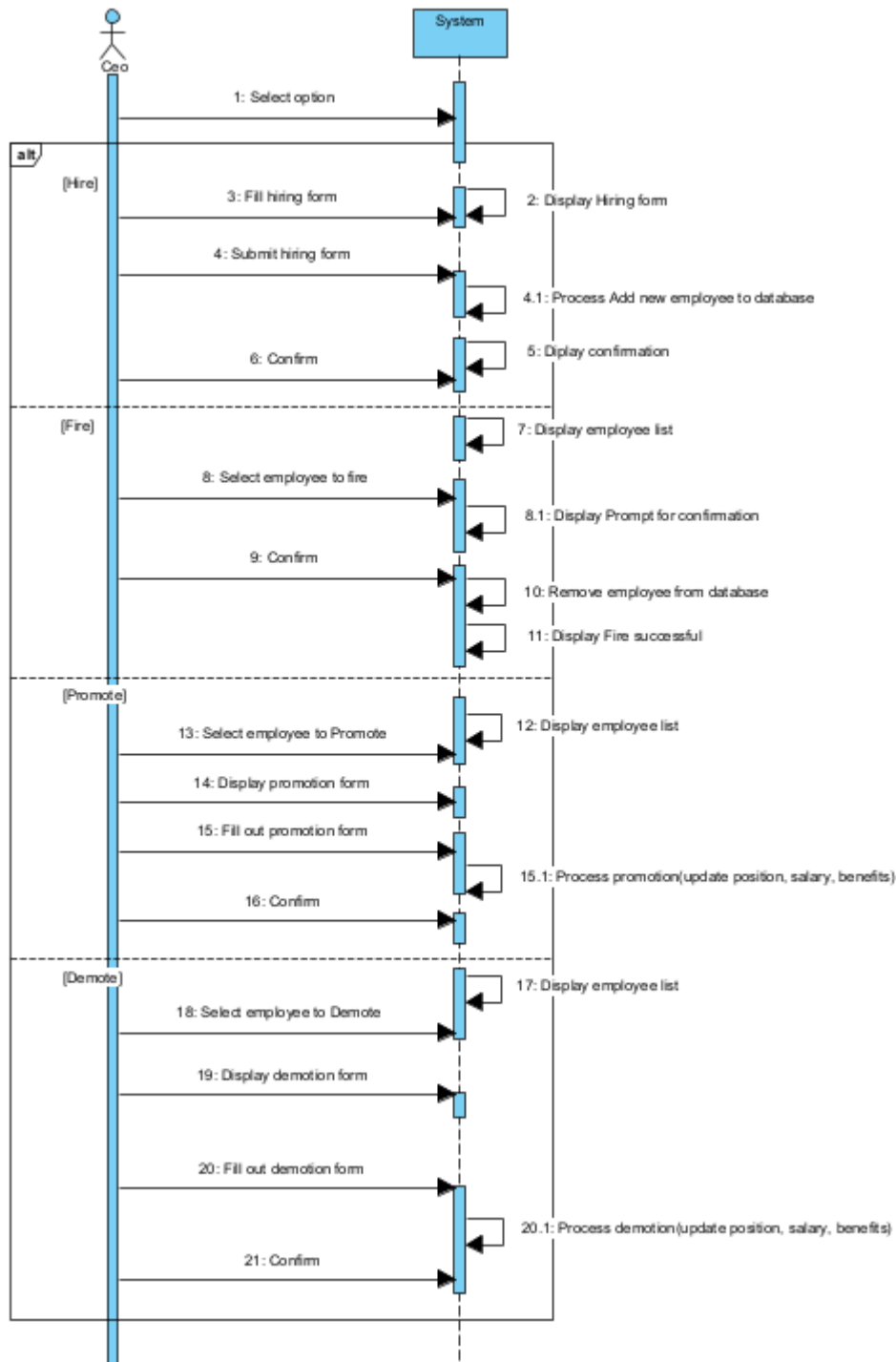
- CEO -> System: Confirm Firing
- 5. **Process Firing:**
 - System -> System: Process Firing (Remove employee from database, update records)
- 6. **Confirm Firing:**
 - System -> CEO: Confirm Firing

Sequence Diagram (Alternative Flow 3: Promote)

1. **Display Employee List:**
 - System -> CEO: Display Employee List
2. **Select Employee to Promote:**
 - CEO -> System: Select Employee to Promote
3. **Display Promotion Form:**
 - System -> CEO: Display Promotion Form
4. **Submit Promotion Form:**
 - CEO -> System: Fill Out and Submit Promotion Form
5. **Process Promotion:**
 - System -> System: Process Promotion (Update employee's position, adjust salary/benefits)
6. **Confirm Promotion:**
 - System -> CEO: Confirm Promotion

Sequence Diagram (Alternative Flow 4: Demote)

1. **Display Employee List:**
 - System -> CEO: Display Employee List
2. **Select Employee to Demote:**
 - CEO -> System: Select Employee to Demote
3. **Display Demotion Form:**
 - System -> CEO: Display Demotion Form
4. **Submit Demotion Form:**
 - CEO -> System: Fill Out and Submit Demotion Form
5. **Process Demotion:**
 - System -> System: Process Demotion (Update employee's position, adjust salary/benefits)
6. **Confirm Demotion:**
 - System -> CEO: Confirm Demotion



Use Case: Login

Actor(s): Employee Description: This use case describes the process by which an employee logs into the system. Preconditions:

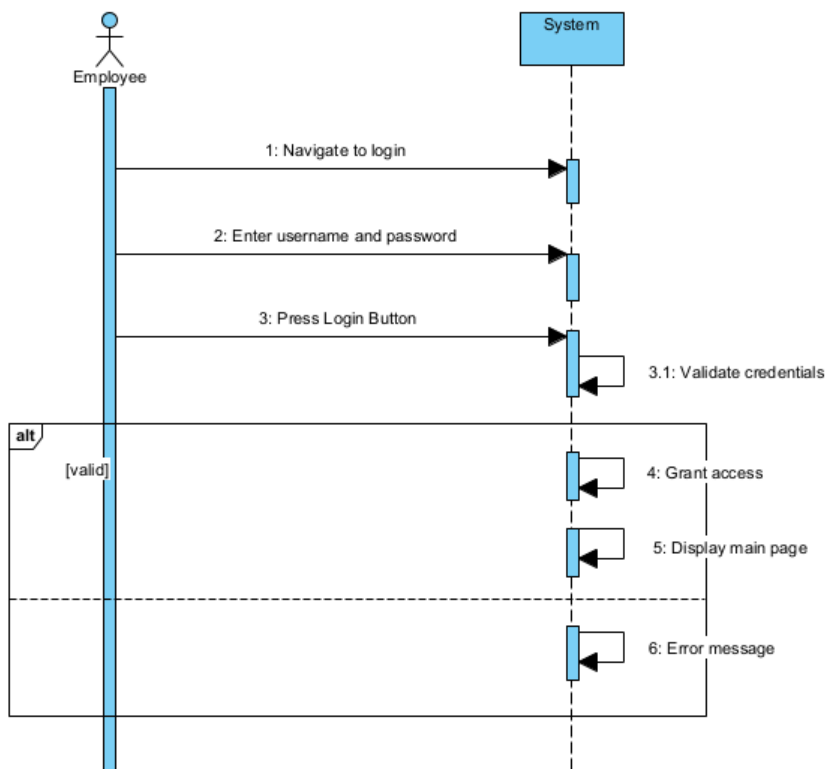
- The employee must have valid login credentials.

Main Flow:

1. The employee navigates to the login page.
2. The employee enters their username and password.
3. The employee submits the login form.
4. The system validates the login credentials.
5. If the credentials are valid, the system grants access to the employee.
6. If the credentials are invalid, the system displays an error message.

Postconditions:

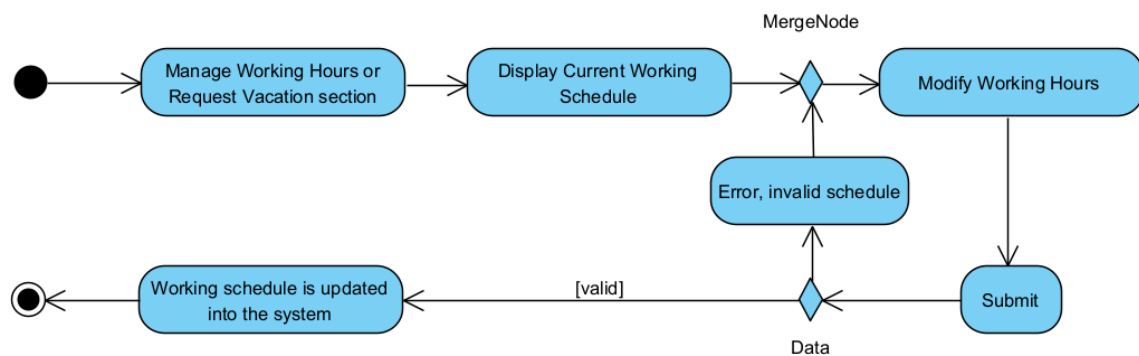
- The employee is either logged into the system or receives an error message if the credentials are invalid.



4. ACTIVITY DIAGRAMS : (For 2 use cases)

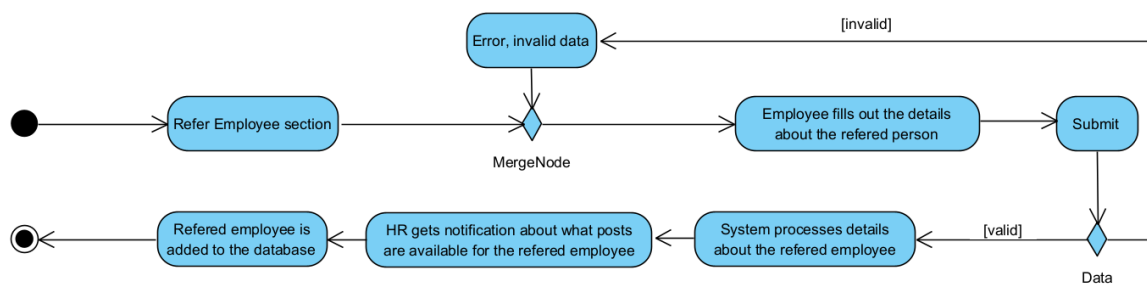
UC name: Manage Working Hours/ Request Vacation Section

Activity diagram:



UC name: Refer Employee section

Activity diagram:



5. GUI PROTOTYPE

Starting with the initial screen of the application, represent an initial screen for each use case that is connected to an actor, screens content and navigation flow (using State Machine Diagram).

If use cases with more screens exist, for each of such use case (after you write the name of

the use case) represent the screens, their content and the navigation flow (using State Machine Diagram).

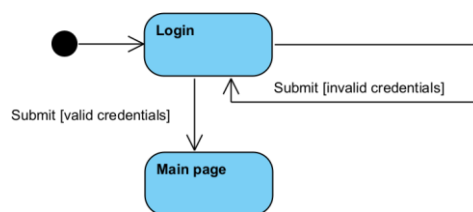
Screens, their content and the navigation flow (represented using State Machine Diagram).

UC Name: Login

Preconditions: credentials should be correct

The image shows two versions of a login form. The first version is a clean form with labels 'Username:' and 'Password:' above input fields, and a blue 'Login' button at the bottom. The second version shows the form after an invalid login attempt. A red error message 'Invalid username or password' is displayed above the Username field. The Username field contains the text 'ioan.popa03@e-uvt.ro' and the Password field contains masked characters '*****'. The 'Login' button remains at the bottom.

Screen flow diagram



UC Name: REFER EMPLOYEE

Preconditions: employee is already logged in into the system

Main Page:

The screenshot shows the main page of an employee portal. At the top is a horizontal navigation bar with buttons for 'Home', 'Refer Employee', 'Manage Working Hours', 'Request Vacation', 'Profile', and 'Logout'. Below the navigation bar, the main heading is 'Welcome to the Employee Portal'. Underneath the heading, there is a text prompt: 'Select an option from the menu to get started.'

The screenshot shows the 'Refer Employee' page. It features the same navigation bar as the main page. The main heading is 'Refer Employee'. Below the heading, there is a text prompt: 'Refer someone for a job position within the company.' At the bottom of the content area, there is a button labeled 'Refer Employee'.

[Home](#) [Refer Employee](#) [Manage Working Hours](#) [Request Vacation](#) [Profile](#) [Logout](#)

Refer Employee

Name of the Person Being Referred: Contact Information (Email): Contact Information (Phone Number): Qualifications/Relevant Experience: Additional Notes (optional):

[Home](#) [Refer Employee](#) [Manage Working Hours](#) [Request Vacation](#) [Profile](#) [Logout](#)

Referral Submitted

Thank you! Your referral has been successfully submitted.

[Refer Another Employee](#) [Back to Main Page](#)

[Home](#) [Refer Employee](#) [Manage Working Hours](#) [Request Vacation](#) [Profile](#) [Logout](#)

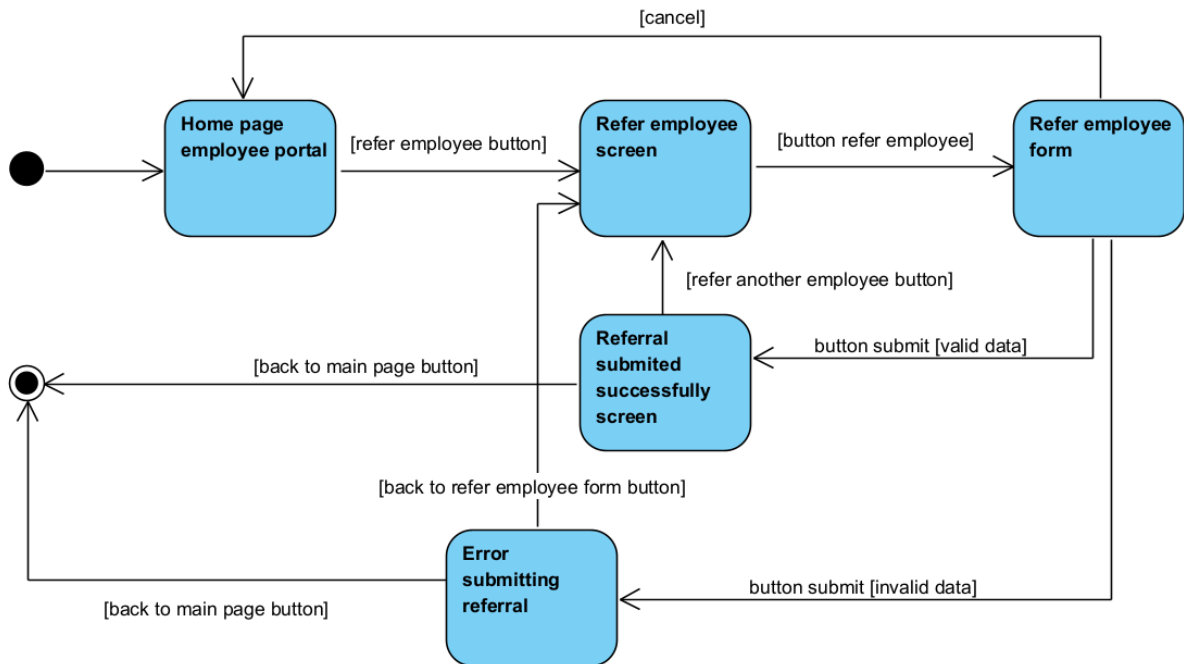
Error Submitting Referral

Error: Please correct the following errors before submitting again:

- Missing required fields
- Invalid email format

[Back to Refer Employee Form](#) [Back to Main Page](#)

Screen flow Diagram



UC Name: MANAGE WORKING HOURS

[Home](#) [Refer Employee](#) [Manage Working Hours](#) [Request Vacation](#) [Profile](#) [Logout](#)

Welcome to the Employee Portal

Select an option from the menu to get started.

[Home](#) [Refer Employee](#) [Manage Working Hours](#) [Request Vacation](#) [Profile](#) [Logout](#)

Manage Working Hours

Current Working Schedule

Monday: 9:00 AM - 5:00 PM

Tuesday: 9:00 AM - 5:00 PM

Wednesday: 9:00 AM - 5:00 PM











Thursday: 9:00 AM - 5:00 PM

Friday: 9:00 AM - 5:00 PM

[Modify Working Hours](#)

[Home](#) [Refer Employee](#) [Manage Working Hours](#) [Request Vacation](#) [Profile](#) [Logout](#)

Modify Working Hours

Day	Start Time	End Time
Monday	--:-- -- 	--:-- -- 
Tuesday	--:-- -- 	--:-- -- 
Wednesday	--:-- -- 	--:-- -- 
Thursday	--:-- -- 	--:-- -- 
Friday	--:-- -- 	--:-- -- 
Submit Changes Cancel		

[Home](#) [Refer Employee](#) [Manage Working Hours](#) [Request Vacation](#) [Profile](#) [Logout](#)

Invalid Schedule

Your schedule contains invalid entries. Please ensure that the start time is before the end time for each day.

[Modify Working Hours](#) [Back to Manage Working Hours](#)

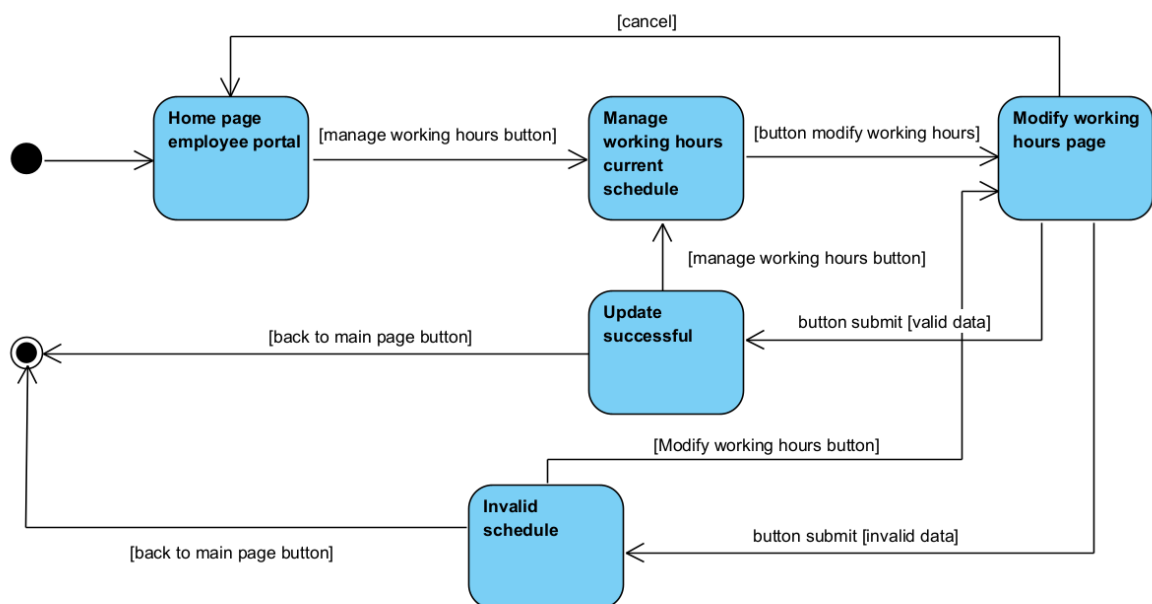
[Home](#)
[Refer Employee](#)
[Manage Working Hours](#)
[Request Vacation](#)
[Profile](#)
[Logout](#)

Update Successful

Your working hours have been updated successfully.

[Manage Working Hours](#)
[Back to Main Page](#)

Screen flow diagram



UC Name: Manage REQUEST VACATION

[Home](#)
[Refer Employee](#)
[Manage Working Hours](#)
[Request Vacation](#)
[Profile](#)
[Logout](#)

Request Vacation

Available Vacation Balance

You have 10 days of vacation remaining.

[Home](#) [Refer Employee](#) [Manage Working Hours](#) [Request Vacation](#) [Profile](#) [Logout](#)

Request Vacation

Available Vacation Balance

You have 10 days of vacation remaining.

[Request Vacation Time](#)

Request Vacation Time

Start Date: End Date: [Submit Request](#) [Cancel](#)

[Home](#) [Refer Employee](#) [Manage Working Hours](#) [Request Vacation](#) [Profile](#) [Logout](#)

Request Submitted

Your vacation request has been submitted successfully.

[Return to Home](#)

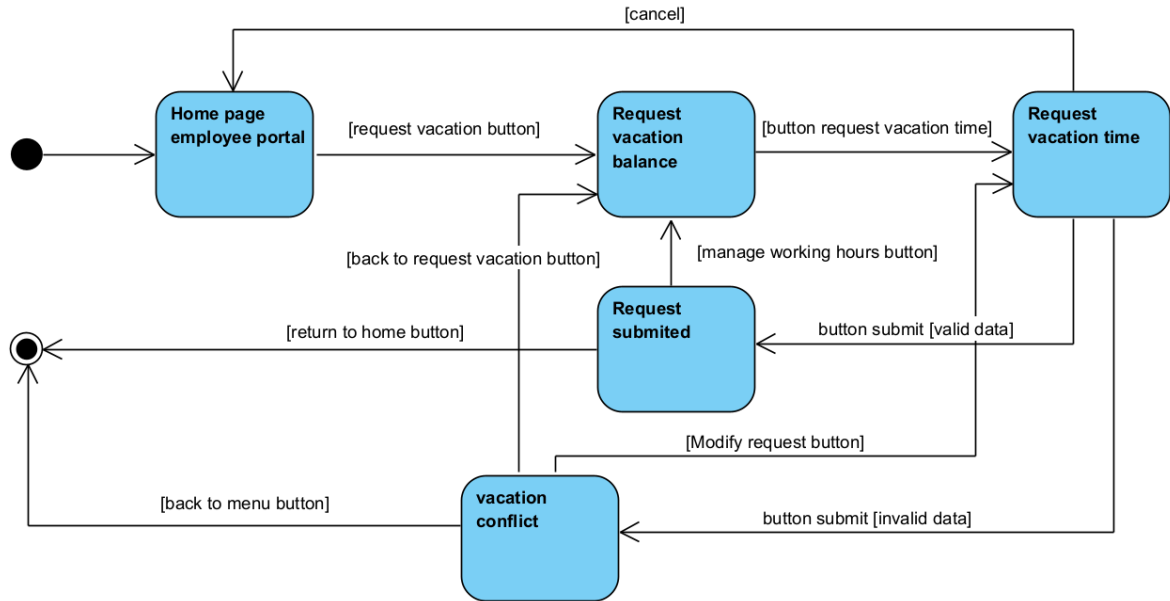
[Home](#) [Refer Employee](#) [Manage Working Hours](#) [Request Vacation](#) [Profile](#) [Logout](#)

Vacation Conflict

Your requested vacation dates conflict with existing schedules or company policies.

[Modify Request](#) [Back to Request Vacation](#) [Back to Menu](#)

Screen flow diagram



UC Name: Manage MANAGE EMPLOYEE STATUS

[Home](#)
[Manage Employee Status](#)
[Profile](#)
[Logout](#)

Manage Employee Status

[Hire](#)
[Fire](#)
[Promote](#)
[Demote](#)

[Back](#)

Hire Employee

Name:

Position:

Department:

Salary:

[Submit](#)

Employee Hired

Popa Ciprian has been successfully hired as CEO in the IT department with a salary of \$10,000.

[Back to Menu](#)

Back

Fire Employee

Select the employee you want to fire:

- Popa Ciprian
- Andrei Frincu

This page says

Are you sure you want to fire Andrei Frincu?

OK

Cancel

Back

Fire Employee

Select the employee you want to fire:

- Popa Ciprian
- Andrei Frincu

This page says

Andrei Frincu has been fired successfully.

OK

Employee Fired

Andrei Frincu has been successfully fired.

Back to Menu

Back

Promote Employee

Select the employee you want to promote:

- Popa Ciprian
- Andrei Frincu

[Back](#)

Promotion Form

Promote employee:

New Position:

New Salary:

[Submit Promotion](#)

[Back to Menu](#)

Successful Promotion

Employee Popa Ciprian has been promoted to CEO with a new salary of \$15,000.

[Back](#)

Demote Employee

Select the employee you want to demote:

- [Popa Ciprian](#)
- [Andrei Frincu](#)

[Back](#)

Demotion Form

Demote employee:

New Position:

New Salary:

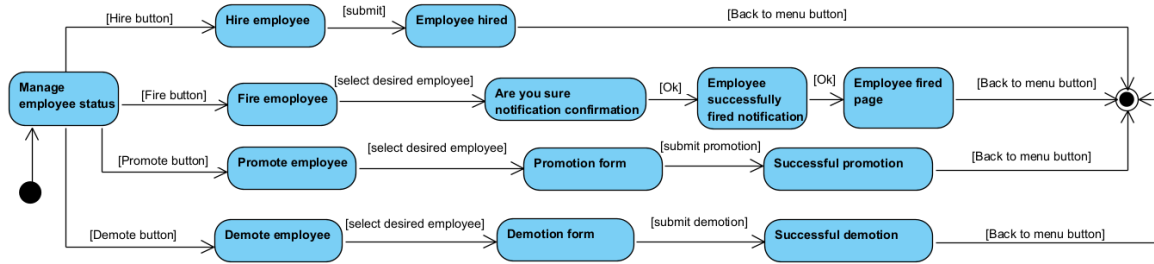
[Submit Demotion](#)

[Back to Menu](#)

Successful Demotion

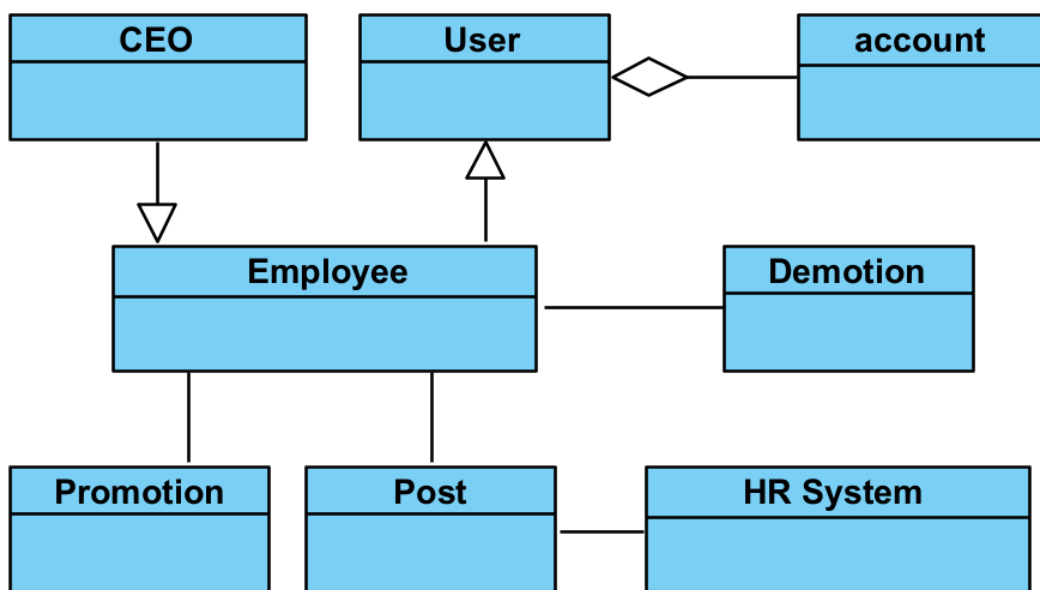
Employee Popa Ciprian has been demoted with a new position of Junior Developer and a salary of \$8,000.

Screen flow diagram



6. DOMAIN MODEL

Class diagram containing domain model : classes and relations (possibly some attributes).

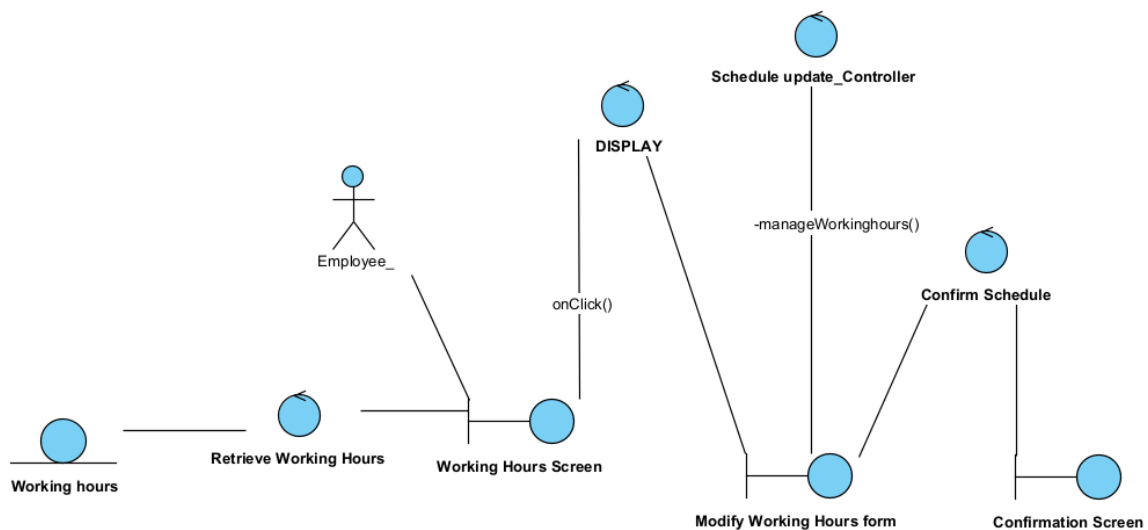


7. ROBUSTNESS DIAGRAMS

The robustness diagrams for 2 use cases. It must be chosen at least one complex use case, that implies more than simple data transfers between the system (application) and the user.

UC name: Manage Working Hours

Robustness diagram



Steps in the Robustness Diagram:

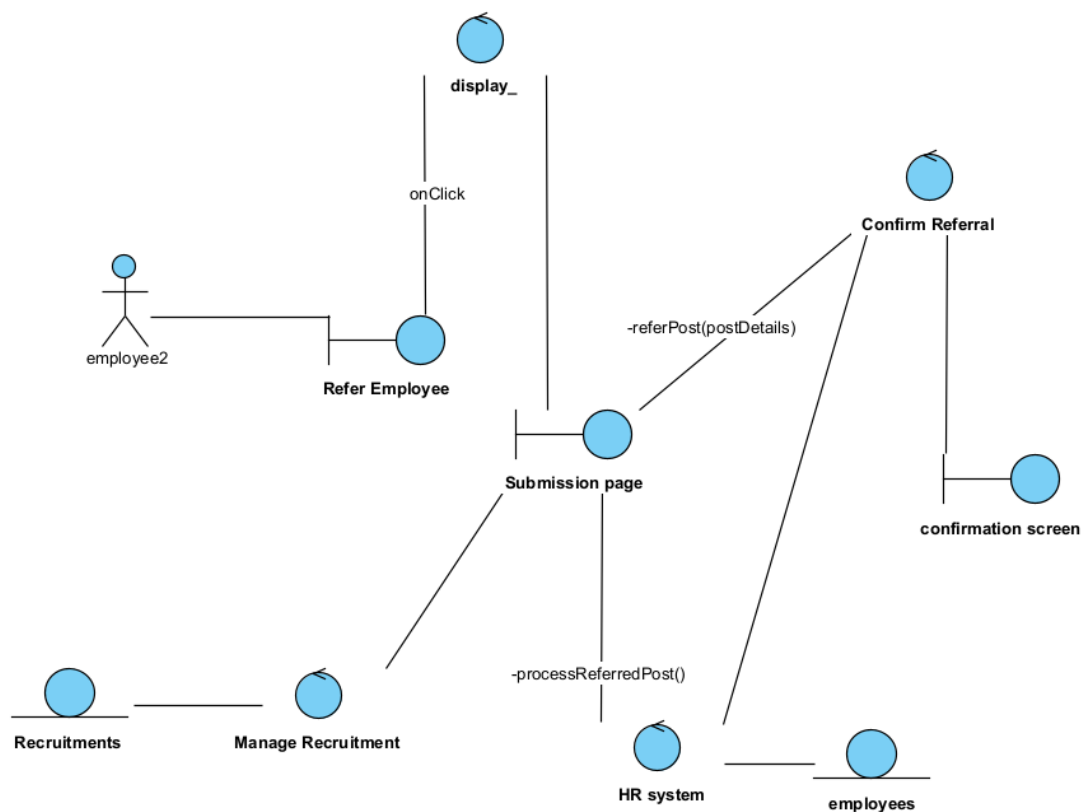
1. **Login Page** interacts with **Login Controller** to authenticate the **Employee**.
2. **Main Menu** navigates to **Working Hours Screen** via the **display**.
3. **Working Hours Screen** interacts with **Retrieve Hours Controller** to fetch and display the **Working Schedule**.
4. **Modify Working Hours Form** is used by the employee to input new working hours.
5. **Modify Working Hours Form** interacts with **Schedule Update Controller** to process the changes.
6. **Schedule Update Controller** updates the **Working Schedule** in the system.
7. **Schedule Update Controller** triggers the **Confirmation Message** to notify the employee about the successful update.

Main Flow:

1. The employee logs into the system.
2. The employee navigates to the "Manage Working Hours" section of the system.
3. The system displays the employee's current working schedule.
4. The employee selects the option to modify their working hours.
5. The employee adjusts their working hours by specifying new start and end times or adding/removing shifts.
6. The employee submits the changes.
7. The system updates the employee's working schedule accordingly.
8. The system confirms the update to the employee.

UC name:Refer Employee

Robustness diagram



Main Flow:

1. The employee logs into the system.
2. The employee navigates to the "Refer Employee" section of the system.
3. The system presents a form for the employee to input the details of the person they want to refer, including name, contact information, and qualifications.
4. The employee fills out the form with the required information.
5. The employee submits the referral.
6. The system processes the referral and notifies the HR responsible for recruitment.
7. The referred individual's details are added to the recruitment database for further consideration.

Alternative Flow:

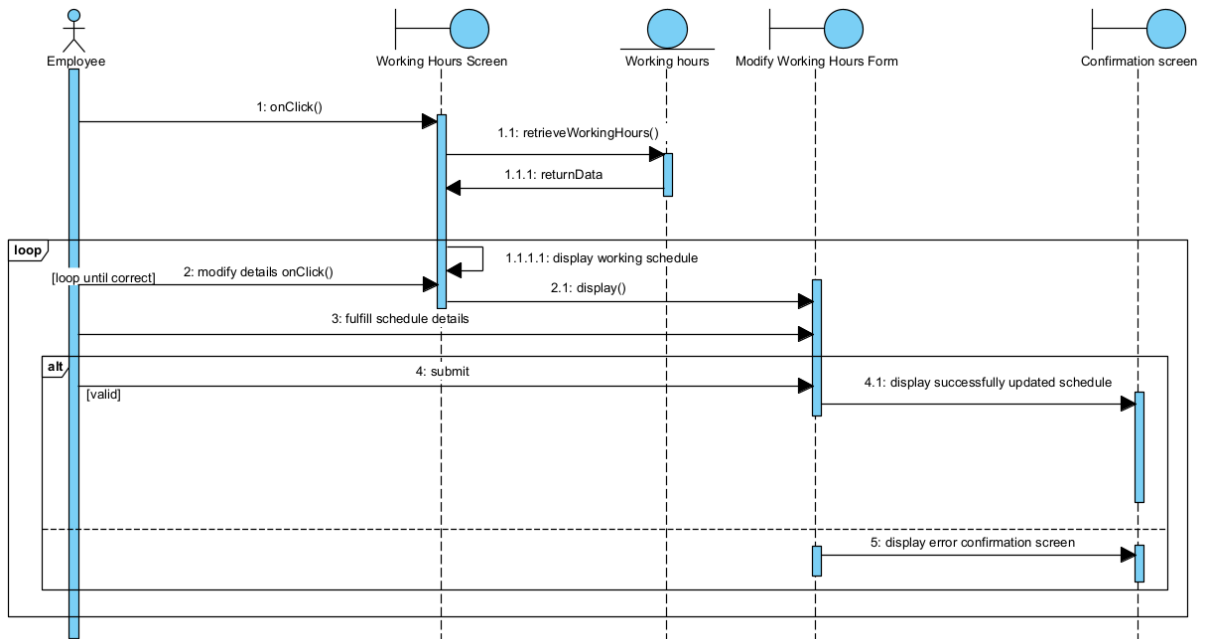
- If the employee enters invalid information in the referral form (e.g., missing required fields, incorrect contact information), the system displays an error message.

8. SEQUENCE DIAGRAMS

The sequence diagrams for the same 2 use cases.

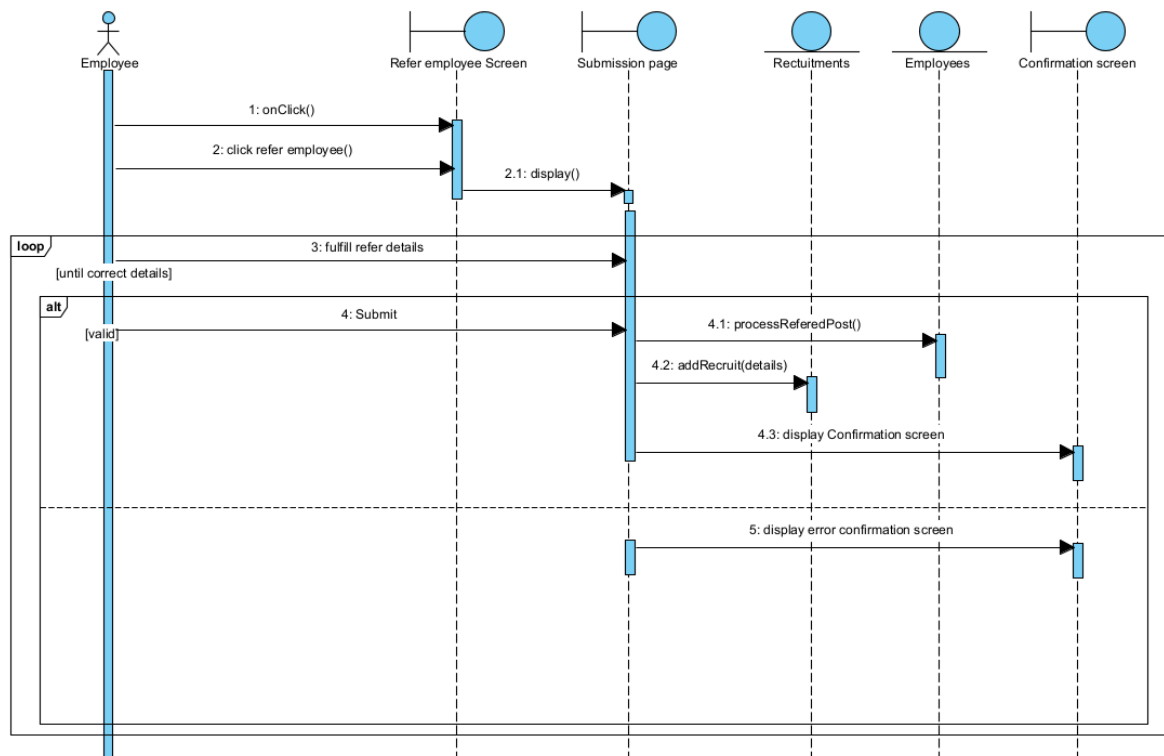
UC name:Manage working hours

Sequence diagram



UC name:Refer Employee

Sequence diagram



9. THE EXTENDED CLASS DIAGRAM

The class diagram resulted from the robustness analysis of the 2 use cases.

