System Requirement Specification

**GDP2\_MOCK2\_G2**

**Version: 1.0**

|  |  |
| --- | --- |
| **Issued Status:** | Approved |
| **Issued Date:** | 12-28-2017 |
| **Owner:** | Group 2 – GDP2 |
| **Author:** | NNDuy |
| **Location:** | <http://101.99.14.196:3030/GDP2/GROUP2/tree/development> |
| **Confidential Class:** | Confidential |

|  |  |
| --- | --- |
| **Date:** | <mm-dd-yyyy> |
| **Approved by:** | <Name> |
| **Signature:** |  |

**Review Information**

|  |  |  |
| --- | --- | --- |
| **Role** | **Required / Suggested** | **Comment** |
|  |  |  |
|  |  |  |

**Approval Information**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Approver Name** | **Role** | **Date**  (mm-dd-yyyy) | **Revision** | **Comment** |
|  |  |  |  |  |
|  |  |  |  |  |

**Revision History**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Revision** | **Date**  (mm-dd-yyyy) | **Description** | **Revised by** | **Reviewer** | **Date**  (mm-dd-yyyy) |
| *Create* | *12-28-2017* | *Create new Srs version 1* | *NNDuy* |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

**Table of Content**

*[Press F9 to edit or update table of content. Please remove this text when applying.]*

[1. Introduction 1](#_Toc502216988)

[1.1. Purpose 1](#_Toc502216989)

[1.1. Scope 1](#_Toc502216990)

[1.2. Definitions, Acronyms and Abbreviations 1](#_Toc502216991)

[1.3. References 1](#_Toc502216992)

[2. Architecture design 1](#_Toc502216993)

[2.1 Usecase diagram 1](#_Toc502216994)

[2.2 State diagram 2](#_Toc502216995)

[2.3 Architectural Request of Spring MVC 3](#_Toc502216996)

[2.4 Component diagram 5](#_Toc502216997)

[2.4.1. Class diagram 5](#_Toc502216998)

[2.4.2. Dao diagram 6](#_Toc502216999)

[2.4.3. Service diagram 6](#_Toc502217000)

[3. Database Design 8](#_Toc502217001)

[3.1. Entity Relationship Diagram 8](#_Toc502217002)

[3.2. Schema 8](#_Toc502217003)

[3.3. Detail schema 9](#_Toc502217004)

[3.3.1. User 9](#_Toc502217005)

[3.3.2. Request 9](#_Toc502217006)

[3.3.3. Comment 10](#_Toc502217007)

[3.3.4. Experience 10](#_Toc502217008)

[3.3.5. Education 10](#_Toc502217009)

[3.3.6. Department 10](#_Toc502217010)

[3.3.7. StatusRequest 11](#_Toc502217011)

[3.3.8. Priority 11](#_Toc502217012)

[3.3.9. Skill 11](#_Toc502217013)

[3.3.10. Request\_Skill 11](#_Toc502217014)

[3.3.11. HistoryRequest 11](#_Toc502217015)

[3.3.12. Action 12](#_Toc502217016)

[3.3.13. Notification 12](#_Toc502217017)

[4. Functional Requirements 12](#_Toc502217018)

[4.1. HR Member 12](#_Toc502217019)

[4.1.1. Manager request 12](#_Toc502217020)

[4.1.1.1. View list published request 12](#_Toc502217021)

[4.1.1.2. View list request by filter and search 14](#_Toc502217022)

[4.1.1.2.1. Search request 14](#_Toc502217023)

[4.1.1.2.2. Filter request 15](#_Toc502217024)

[4.1.1.2.3. Sort request 15](#_Toc502217025)

[4.1.1.3. View list request by status 18](#_Toc502217026)

[4.1.1.4. View request detail 20](#_Toc502217027)

[4.1.1.5. Assign 25](#_Toc502217028)

[4.1.1.6. Update field 28](#_Toc502217029)

[4.2. HR Manager 33](#_Toc502217030)

[4.2.1. Manager request 33](#_Toc502217031)

[4.2.1.1. View list published request 33](#_Toc502217032)

[4.2.1.2. View request detail 33](#_Toc502217033)

[4.2.1.3. Assign 39](#_Toc502217034)

[4.2.1.4. Update field 42](#_Toc502217035)

[4.2.1.5. Publish 46](#_Toc502217036)

[4.2.1.6. Reject 49](#_Toc502217037)

[4.2.1.7. Close request 53](#_Toc502217038)

[5. Non-functional requirements 57](#_Toc502217039)

[Usability 57](#_Toc502217040)

[<Usability Requirement One> 57](#_Toc502217041)

[Reliability 57](#_Toc502217042)

[<Reliability Requirement One> 57](#_Toc502217043)

[Performance 58](#_Toc502217044)

[<Performance Requirement One> 58](#_Toc502217045)

[Supportability 58](#_Toc502217046)

[<Supportability Requirement One> 58](#_Toc502217047)

[Design Constraints 58](#_Toc502217048)

[<Design Constraint One> 58](#_Toc502217049)

[On-line User Documentation and Help System Requirements 58](#_Toc502217050)

[Purchased Components 58](#_Toc502217051)

[Interfaces 58](#_Toc502217052)

[User Interfaces 58](#_Toc502217053)

[Hardware Interfaces 58](#_Toc502217054)

[Software Interfaces 59](#_Toc502217055)

[Communications Interfaces 59](#_Toc502217056)

[Environment 59](#_Toc502217057)

[Target Environment 59](#_Toc502217058)

[Development Environment 59](#_Toc502217059)

[Database 59](#_Toc502217060)

[Licensing Requirements 59](#_Toc502217061)

[Legal, Copyright, and Other Notices 59](#_Toc502217062)

[Applicable Standards 59](#_Toc502217063)

[6. Appendix 59](#_Toc502217064)

1. Introduction
   1. Purpose

*[Specify the purpose of this* ***SRS****. The* ***SRS*** *should fully describe the external behavior of the application or subsystem identified. It also describes nonfunctional requirements, design constraints and other factors necessary to provide a complete and comprehensive description of the requirements for the software.]*

* 1. Scope

*[A brief description of the software application that the* ***SRS*** *applies to; the feature or other subsystem grouping; what Use-Case model(s) it is associated with; and anything else that is affected or influenced by this document.]*

* 1. Definitions, Acronyms and Abbreviations

*[Provide the definitions of all terms, acronyms, and abbreviations required to properly interpret the* ***SRS****. This information may be provided by reference to the project Glossary.]*

* 1. References

*[This subsection should:*

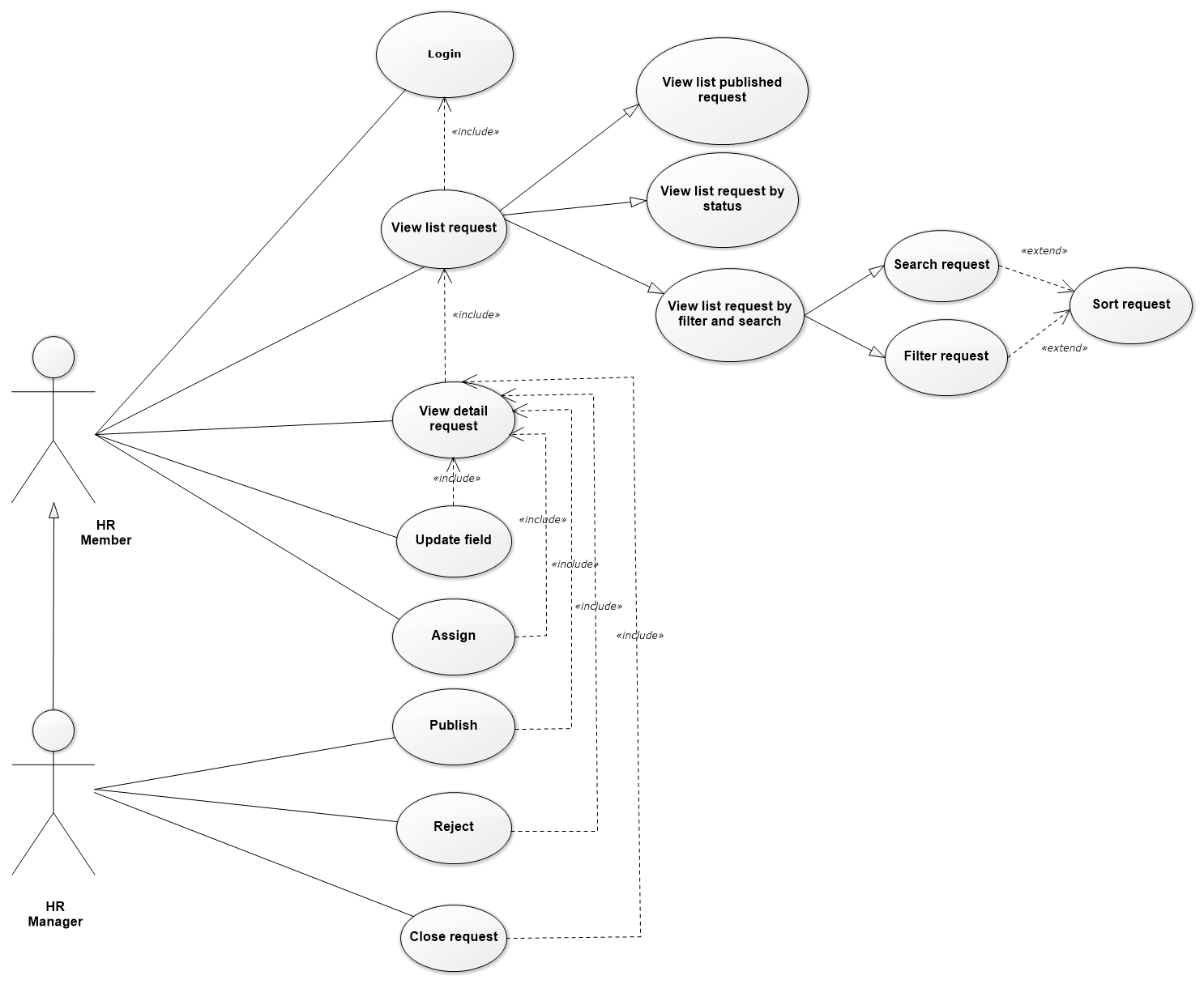
1. *Provide a complete list of all documents referenced elsewhere in the SRS.*
2. *Identify each document by title, report number (if applicable), date and publishing organization.*
3. *Specify the sources from which the references can be obtained.*

*This information may be provided by reference to an appendix or to another document.]*

1. Architecture design

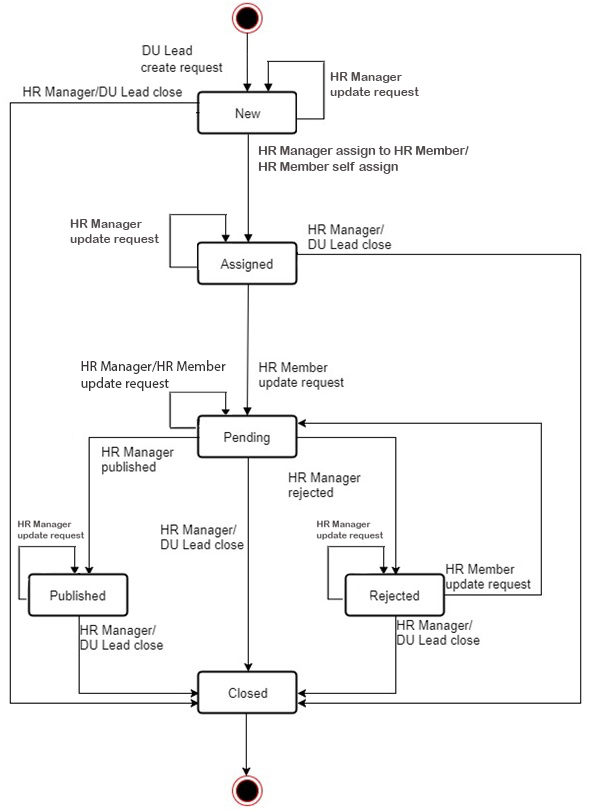
*[Describe the general factors that affect the product and its requirements. This section does not state specific requirements. Instead, it provides a background for those requirements, which are defined in detail in Section 3, and makes them easier to understand. Include such items as:*

1. *Product overview.*
2. *Product functions.*
3. *User characteristics.*
4. *Constraints.*
5. *Assumptions and dependencies.*
6. *Apportioning of requirements]*
   1. Usecase diagram
      1. ***Usecase for HR Manager and HR Member***
         1. ***Use case Manager request***



* 1. State diagram

***2.2.1 Request state diagram***



* 1. Architectural Request of Spring MVC

The following diagram shows the primary components in the proposed Spring MVC architecture.

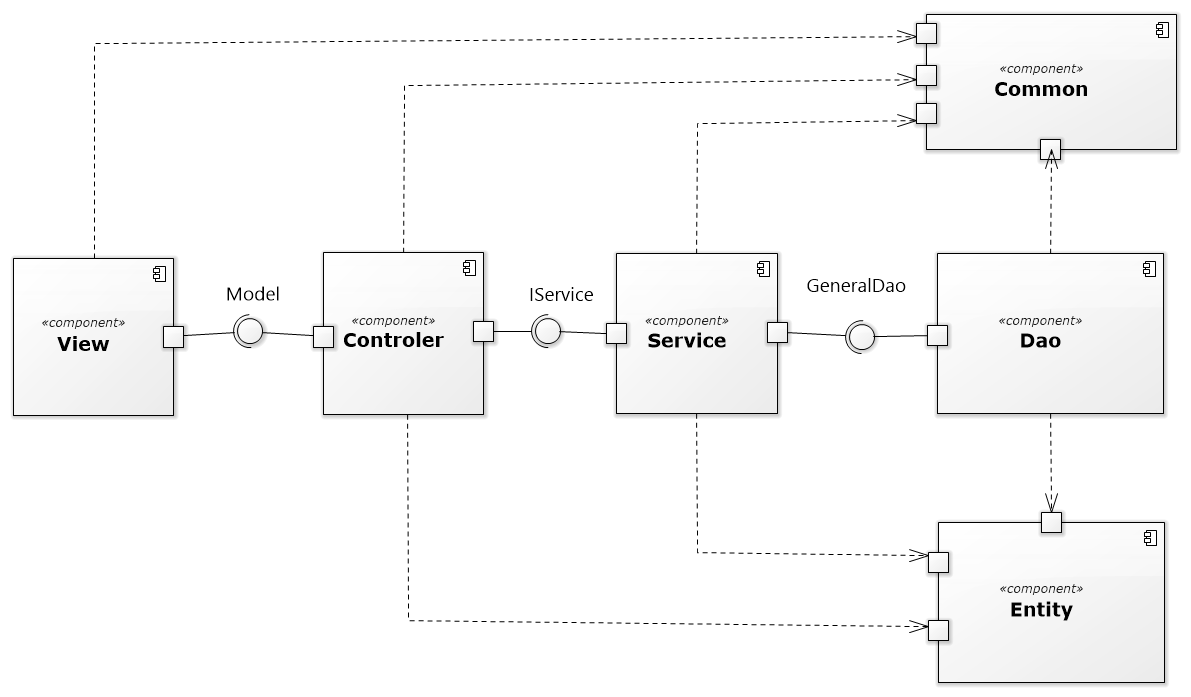
This diagram shows the main layers in this architecture and the vision of how they fit together.



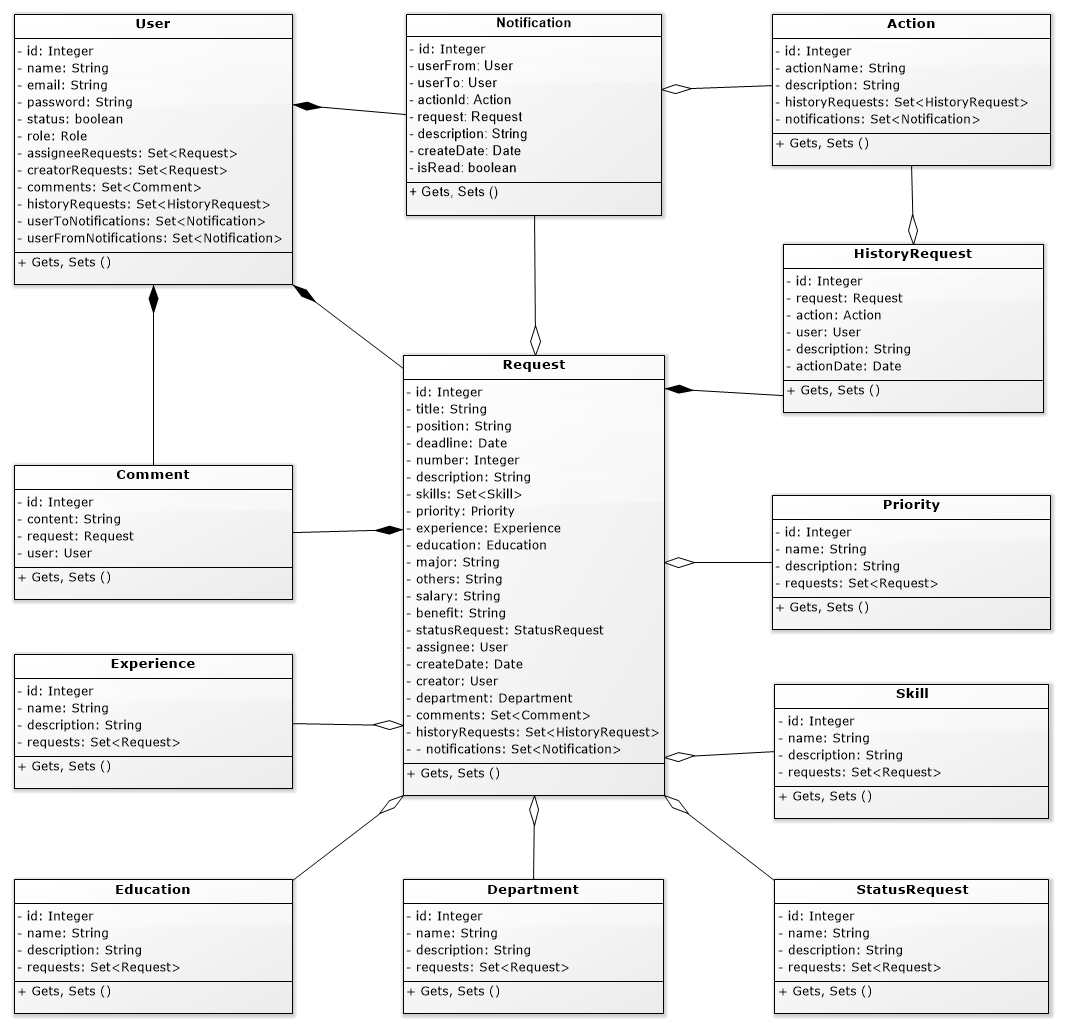
Figure 1 – Spring MVC architecture of request from customer

**Explain workflow of MVC Architecture of request from customer**

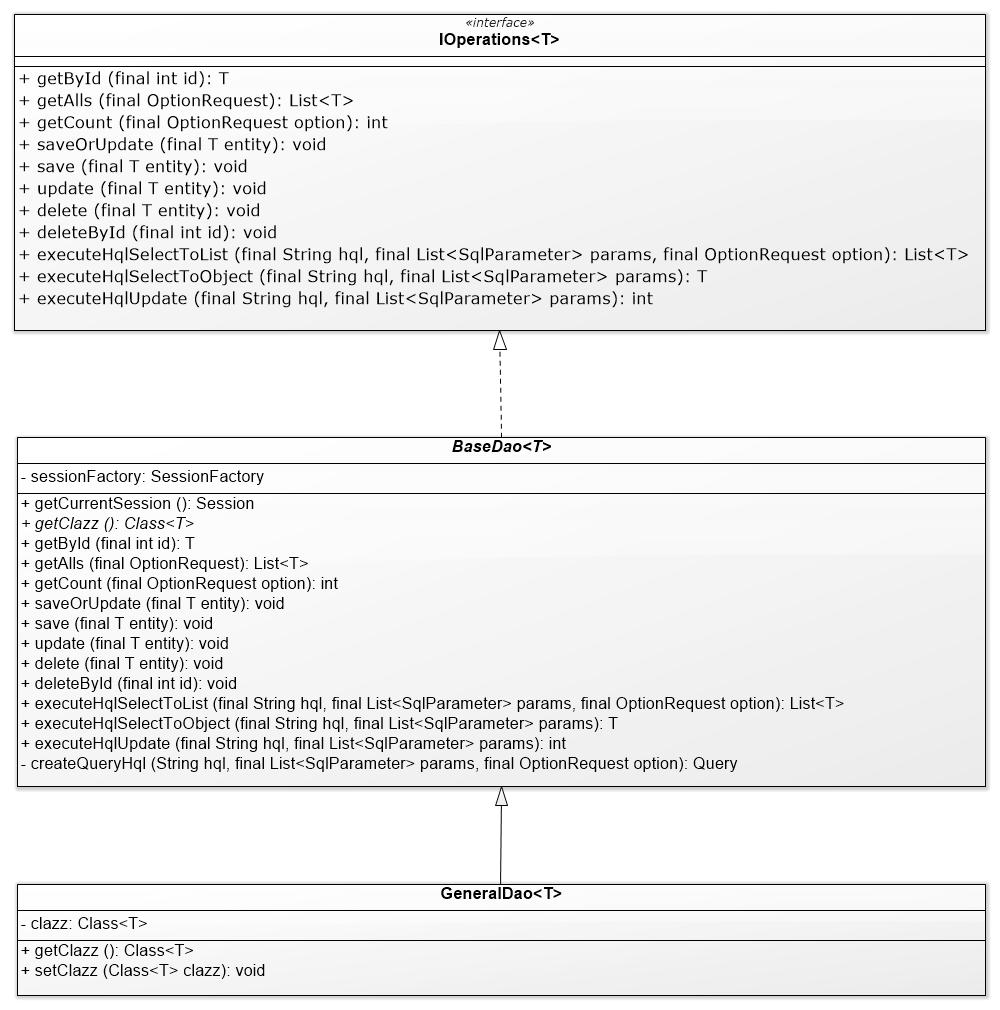
1. DispatcherServlet receives the request.
2. DispatcherServlet dispatches the task of selecting an appropriate controller to HandlerMapping. HandlerMapping selects the controller which is mapped to the incoming request URL and returns the (selected Handler) and Controller to DispatcherServlet.
3. DispatcherServlet dispatches the task of executing of business logic of Controller to HandlerAdapter.
4. HandlerAdapter calls the business logic process of Controller.
5. Controller executes the business logic, sets the processing result in Model and returns the logical name of view to HandlerAdapter.
6. DispatcherServlet dispatches the task of resolving the View corresponding to the View name to ViewResolver. ViewResolver returns the View mapped to View name.
7. DispatcherServlet dispatches the rendering process to returned View.
8. View renders Model data and returns the response.
   1. Component diagram



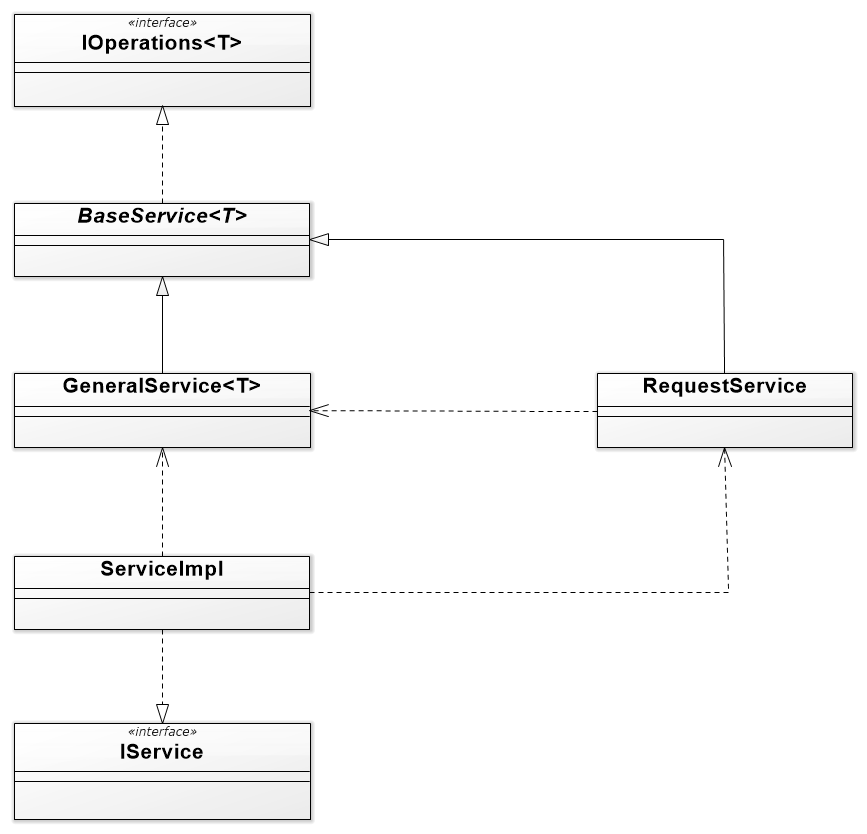
* + 1. Class diagram

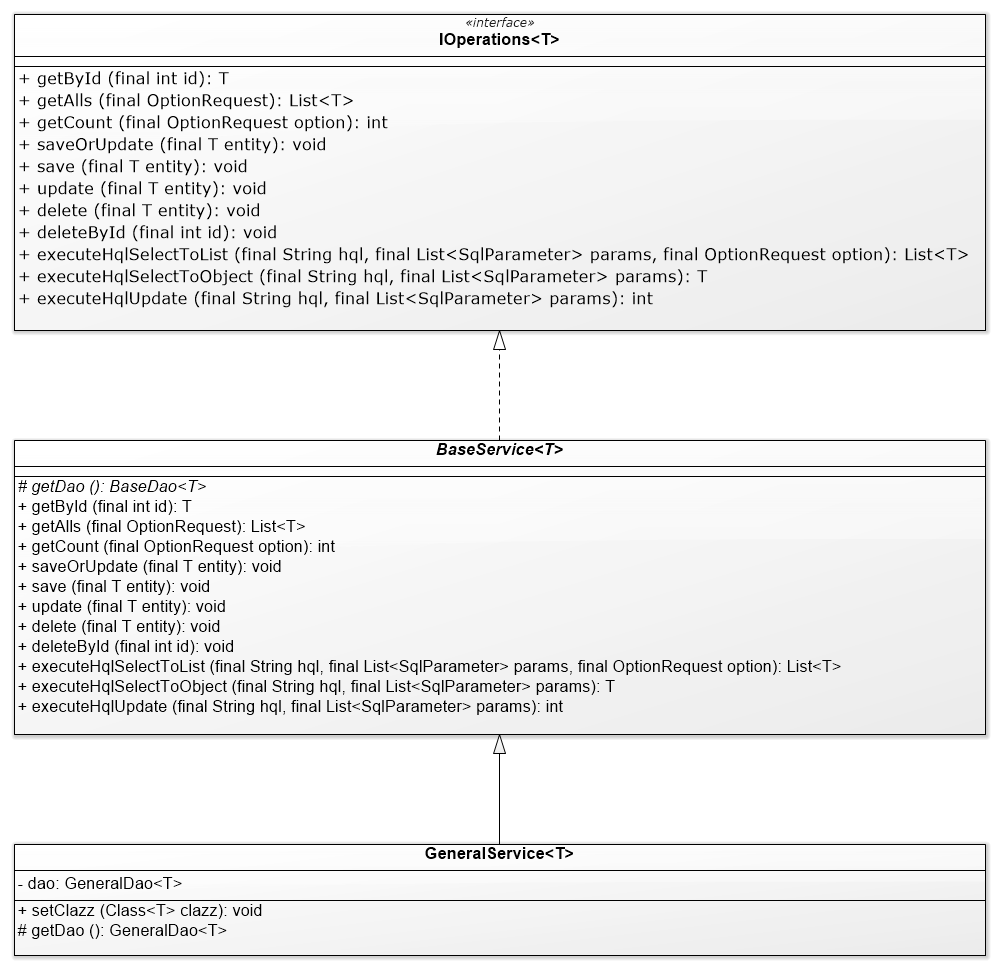


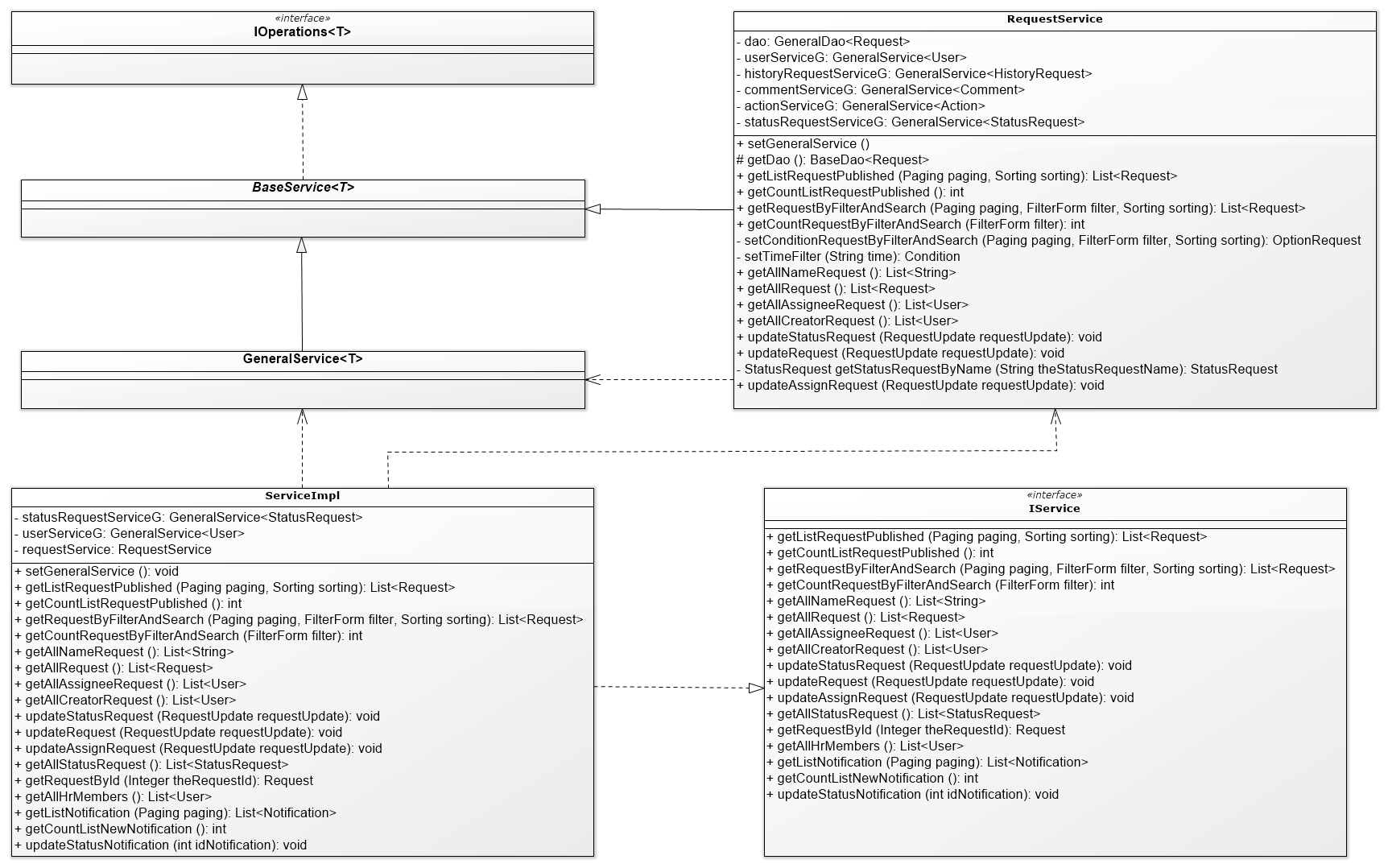
* + 1. Dao diagram



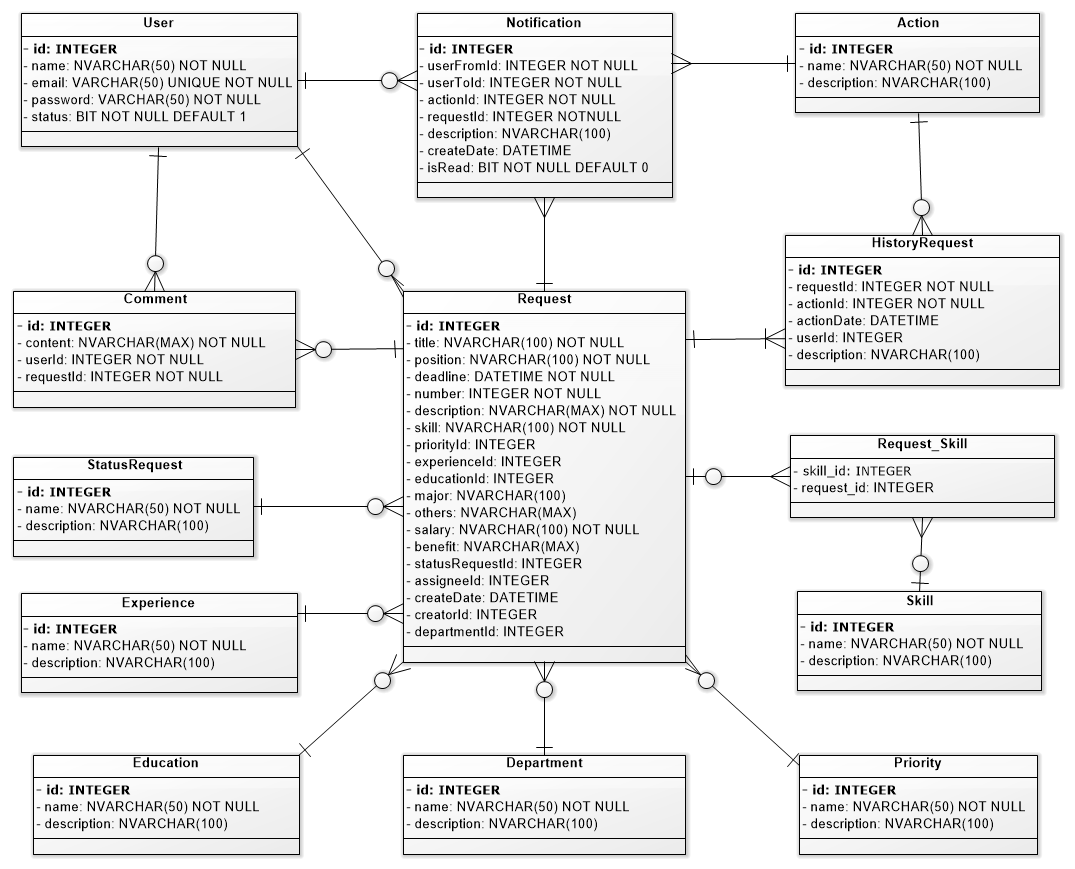
* + 1. Service diagram







1. Database Design
   1. Entity Relationship Diagram



* 1. Schema

|  |  |  |
| --- | --- | --- |
| **STT** | **Name** | **Description** |
| 1 | User | List all users in system |
| 2 | Request | List all requests in system |
| 3 | Comment | Log comments about the request |
| 4 | Experience | Number of work experience year |
| 5 | Education | Current level of education |
| 6 | Department | List all departments in system |
| 7 | StatusRequest | The statuses of the request |
| 8 | Priority | List all the priorities of request |
| 9 | Skill | List all the skills of work |
| 10 | Request\_Skill | Relationship of table request and skill |
| 11 | HistoryRequest | Log all request in system |
| 12 | Action | The user's actions on the request |
| 13 | Notification | List all Notification of request |

* 1. Detail schema
     1. User

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| User | | | | |
|  | **Field name** | **Type** | **Max Length** | **Descrition** |
| 1 | id | int |  | User ID of user |
| 2 | name | nvarchar | 50 | User name of user |
| 3 | email | varchar | 50 | Mail of user |
| 4 | password | varchar | 50 | Password of user |
| 5 | status | int |  | Status of user |
| 6 | roleId | int |  | Store Role ID |

* + 1. Request

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Request | | | | |
|  | **Field name** | **Type** | **Max Length** | **Descrition** |
| 1 | id | int |  | ID of log record |
| 2 | title | nvarchar | 100 | Name of request |
| 3 | position | nvarchar | 100 | Positions to recruit |
| 4 | deadline | datetime |  | The final time must be completed |
| 5 | number | int |  | Quantity to be recruited |
| 6 | description | nvarchar | 100 | Detailed description of request |
| 7 | skill | nvarchar | 100 | Skills Required |
| 8 | priorityId | int |  | Id of request |
| 9 | experienceId | int |  | Id of experience |
| 10 | educationId | int |  | Id of education |
| 11 | major | nvarchar | 100 | Majors have studied |
| 12 | others | nvarchar | Max | Other descriptions |
| 13 | salary | nvarchar | 100 | Month's salary |
| 14 | benefit | nvarchar | Max | Other benefits |
| 15 | statusRequestId | int |  | ID of status request |
| 16 | assignee | int |  | ID of assignee |
| 17 | createDate | datetime |  | Time to create request |
| 18 | creatorId | int |  | ID of creator |
| 19 | departmentId | int |  | ID of receiver account |

* + 1. Comment

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Comment | | | | |
|  | **Field name** | **Type** | **Max Length** | **Descrition** |
| 1 | id | int |  | Id of comment |
| 2 | content | nvarchar | Max | Comment that user enter |
| 3 | userId | int |  | Id of user |
| 4 | requestId | int |  | Id of request |

* + 1. Experience

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Experience | | | | |
|  | **Field name** | **Type** | **Max Length** | **Descrition** |
| 1 | id | int |  | Id of experience |
| 2 | name | nvarchar | 50 | Name of experience |
| 3 | description | nvarchar | 100 | Description of experience |

* + 1. Education

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Education | | | | |
|  | **Field name** | **Type** | **Max Length** | **Descrition** |
| 1 | id | int |  | Id of education |
| 2 | name | nvarchar | 50 | Name of education |
| 3 | description | nvarchar | 100 | Description of education |

* + 1. Department

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Department | | | | |
|  | **Field name** | **Type** | **Max Length** | **Descrition** |
| 1 | id | int |  | Id of department |
| 2 | name | nvarchar | 50 | Name of deapartment |
| 3 | description | nvarchar | 100 | Description of department |

* + 1. StatusRequest

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| StatusRequest | | | | |
|  | **Field name** | **Type** | **Max Length** | **Descrition** |
| 1 | id | int |  | Id of status request |
| 2 | name | nvarchar | 50 | Name of status request |
| 3 | description | nvarchar | 100 | Description of status request |

* + 1. Priority

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Priority | | | | |
|  | **Field name** | **Type** | **Max Length** | **Descrition** |
| 1 | id | int |  | Id of priority |
| 2 | name | nvarchar | 50 | Name of priority |
| 3 | description | nvarchar | 100 | Description of priority |

* + 1. Skill

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| RequestPulished | | | | |
|  | **Field name** | **Type** | **Max Length** | **Descrition** |
| 1 | id | int |  | Id of skill |
| 2 | name | nvarchar | 50 | Name of skill |
| 3 | description | nvarchar | 100 | Description of skill |

* + 1. Request\_Skill

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| RequestPulished | | | | |
|  | **Field name** | **Type** | **Max Length** | **Descrition** |
| 1 | skill\_id | int |  | Id of skill |
| 2 | request\_id | int |  | Id of request |

* + 1. HistoryRequest

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| HistoryRequest | | | | |
|  | **Field name** | **Type** | **Max Length** | **Descrition** |
| 1 | id | int |  | Id of request history |
| 2 | requestId | int |  | Id of request |
| 3 | actionId | int |  | Id of action |
| 4 | userId | int |  | Id of user |
| 5 | actionDate | Datetime |  | Date execute action |
| 6 | description | nvarchar | 100 | Description of history request |

* + 1. Action

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Action | | | | |
|  | **Field name** | **Type** | **Max Length** | **Descrition** |
| 1 | id | int |  | Id of action |
| 2 | name | nvarchar | 50 | Name of action |
| 3 | description | nvarchar | 100 | Description of action |

* + 1. Notification

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Action | | | | |
|  | **Field name** | **Type** | **Max Length** | **Descrition** |
| 1 | id | int |  | Id of notification |
| 2 | userFromId | int |  | Id of user send |
| 3 | userToId | int |  | Id of user reciever |
| 4 | actionId | int |  | Id of action |
| 5 | requestId | int |  | Id of request |
| 6 | description | nvarchar | 100 | Description of action |
| 7 | createDate | datetime |  | Create Date of notification |
| 8 | isRead | bit |  | read notification? |

1. Functional Requirements
   1. HR Member
      1. Manager request
         1. View list published request

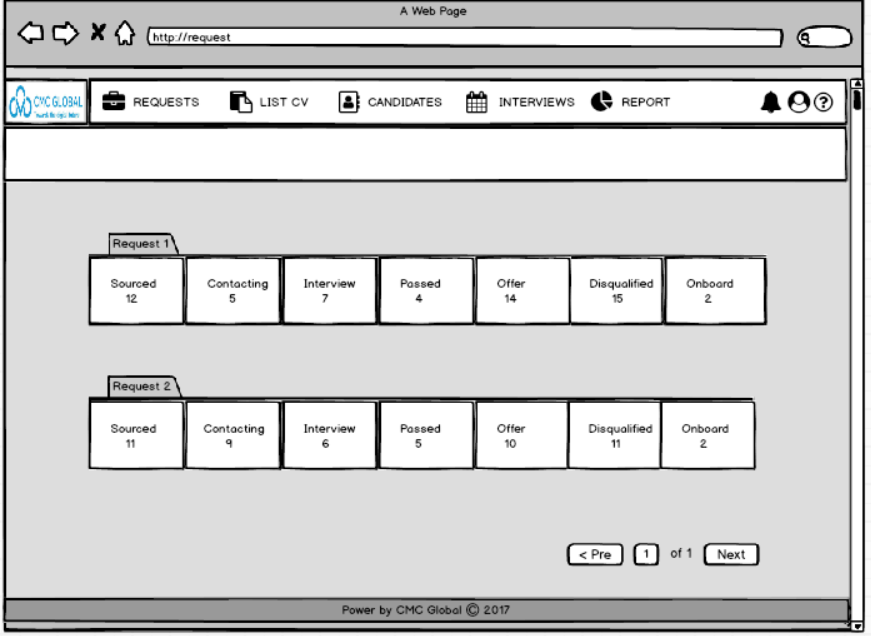
* Actor: HR Member
* Purpose: Displays a list of published requests
* Request:

 ⮚In the main screen containing the list published request  (the main screen when HR Member has logged in), the full list of published requests will be displayed. The list includes basic information: the request title, the status of the recruitment process and the number of applicants for each of them.

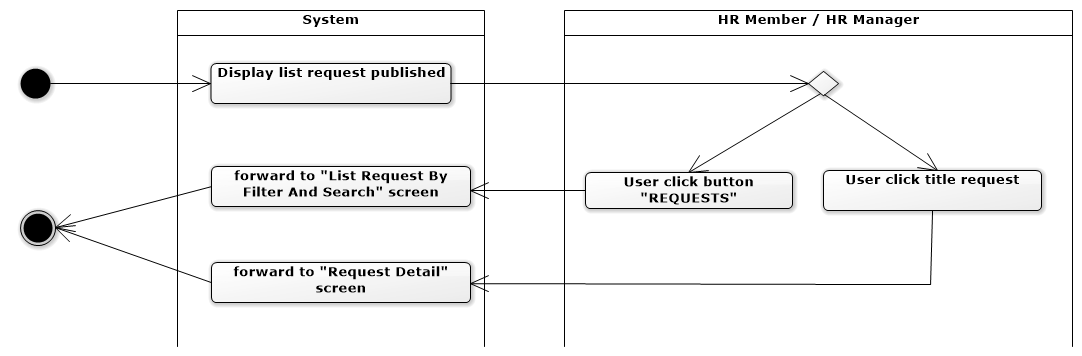
⮚       In addition, the HR Member can click on the "REQUETS" button in header of the screen to select the function that displays all requests in the system.

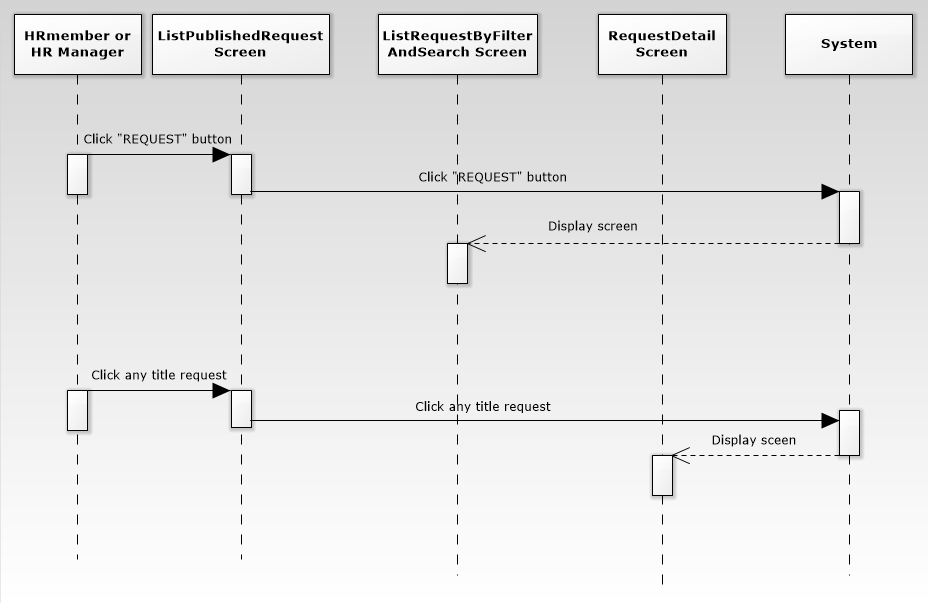
⮚       HR Member can click on the title of each request to view the details of the request.

* Related functions: See request list *[4.1.1.2], [4.1.1.4]*.
* Screen Design & Data Description:



* Activities Flow



* Sequense diagram
* Pseudo code

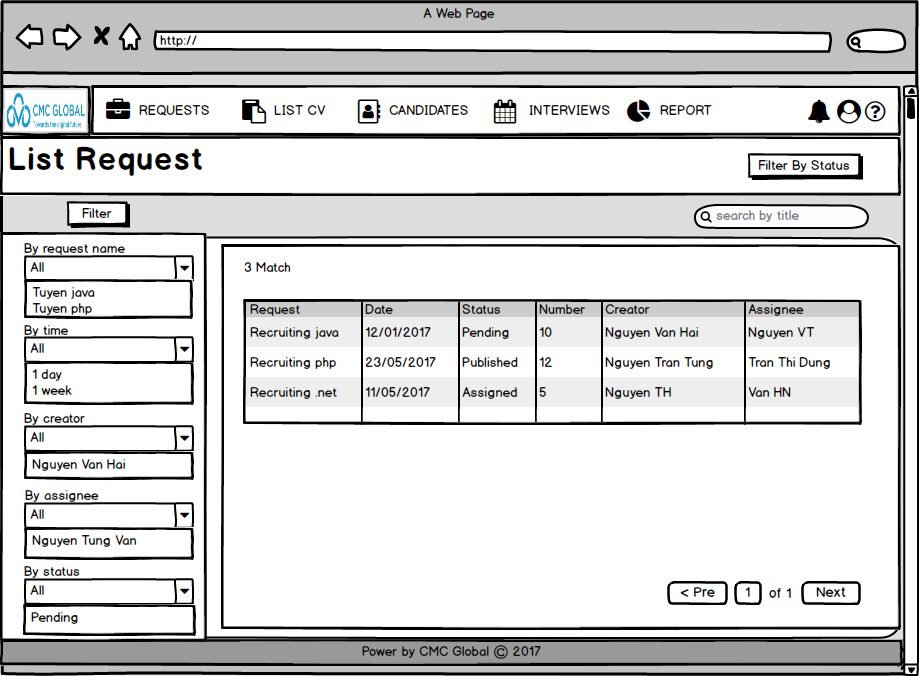
**BEGIN MAINPROGRAM getListRequestPublished()**

Get list paging published requests;

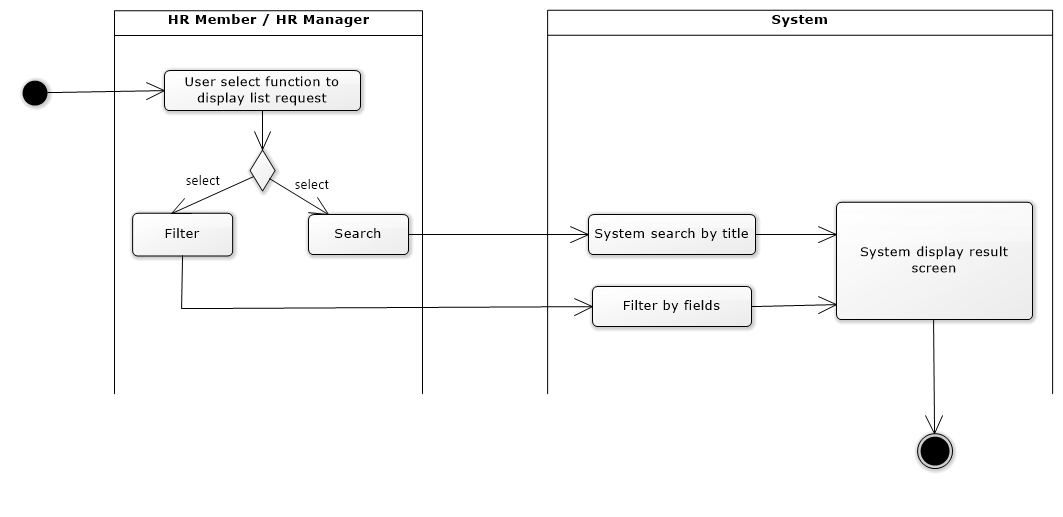
Display to list published request screen;

**END MAINPROGRAM getListRequestPublished()**

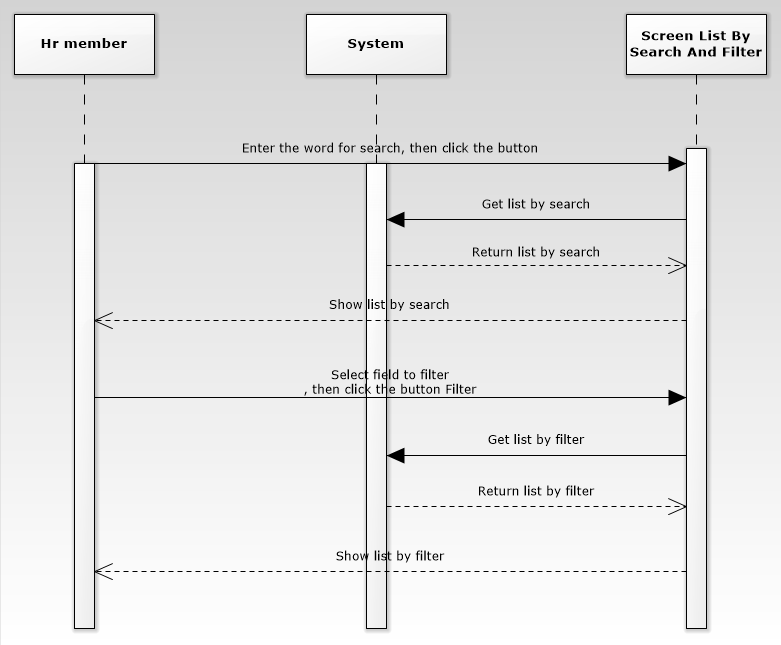
* + - 1. View list request by filter and search
         1. Search request
* Actor*:* HR Member*.*
* Purpose*:* HR Member finds a request by title to facilitate updating the specified request.
* Request:
* On the List Request By Filter And Search screen, HR Member enters keywords in the Search box, then press "Search" button. A list request appears matching the keywords (title).
* In addition, HR Member can combined search with filter and sort.
* More related functions: See request list *[4.1.1.1], [4.1.1.2.2], [4.1.1.2.3].*
  + - * 1. Filter request
* Actor: HR Member
* Purpose*:* HR Member can request the criteria according to criteria.
* Request:
* On the List Request By Filter And Search screen, HR Member can filter requests according to the following criteria: request name, the time period (day, week, month, year ...), creator, assignee, request status. HR Member click on these criteria in the corresponding combo box.
* In addition, HR Member can combined filter with search and sort.
* More related functions: See request list *[4.1.1.1], [4.1.1.2.1], [4.1.1.2.3].*
  + - * 1. Sort request
* Actor: HR Member
* Purpose*:* HR Member can arrange request according to criteria.
* Request:
* On the List Request By Filter And Search screen, HR Member can organize requests the criteria: request name, request status, creator, creation date, assignee. HR Member click on the column name corresponding criteria to proceed to sort.
* In addition, HR Member can combined sort with filter and search.
* More related functions: See request list *[4.1.1.1], [4.1.1.2.1], [4.1.1.2.2].*
* Screen Design & Data Description:



* Activities Flow



* Sequense diagram



* Pseudo code

**BEGIN SUBPROGRAM getRequestByFilterAndSearch(Paging paging, Filter filter, Sorting sorting)**

Display screen list all request with hr member select click button REQUESTS;

Get keyword from keyboard for funtion search;

Get all fields that HR member has filter, paging, sorting selected;

**IF** (Search) **THEN**

Condtion add Search

**IF** (Filter) **THEN**

Condition add few fields filter

[Display screen show list all request with condition by paging]

**END SUBPROGRAM getRequestByFilterAndSearch(Paging paging, Filter filter, Sorting sorting)**

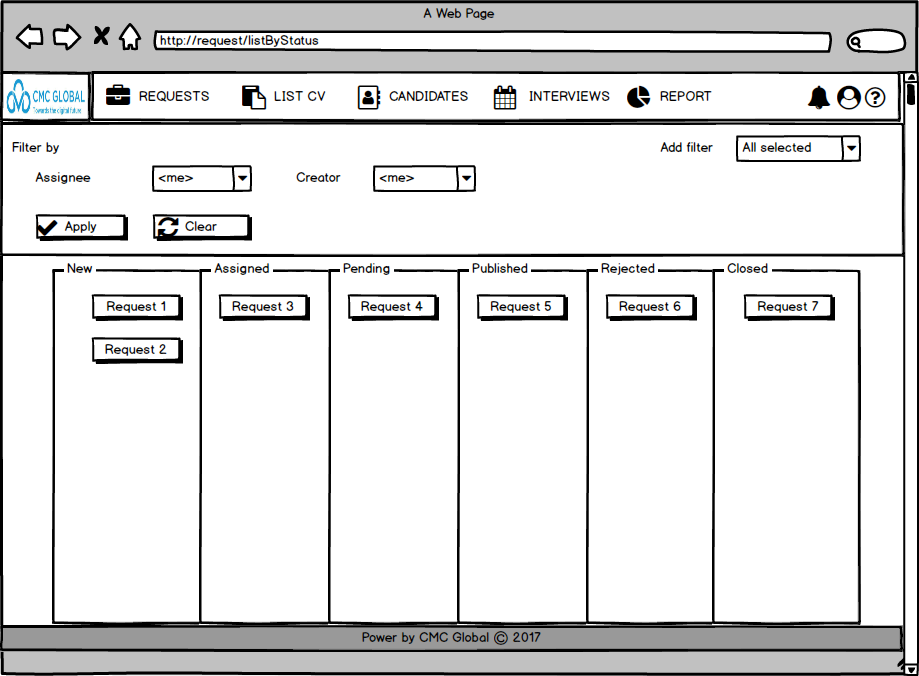
* + - 1. View list request by status
* Actor: HR Member
* Purpose: Displays list requests by status
* Request:

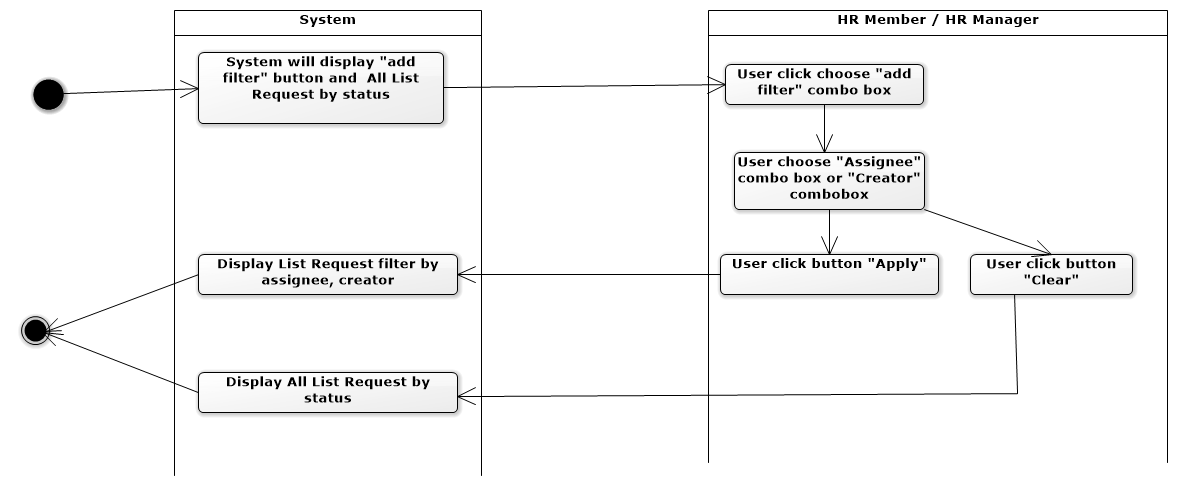
 ⮚ When click the “List request by status” button on the "List request by filter and search" screen will forward to the screen including containing the list name of request was sort by status (New, Assigned, Pending, Published, Rejected, Closed).

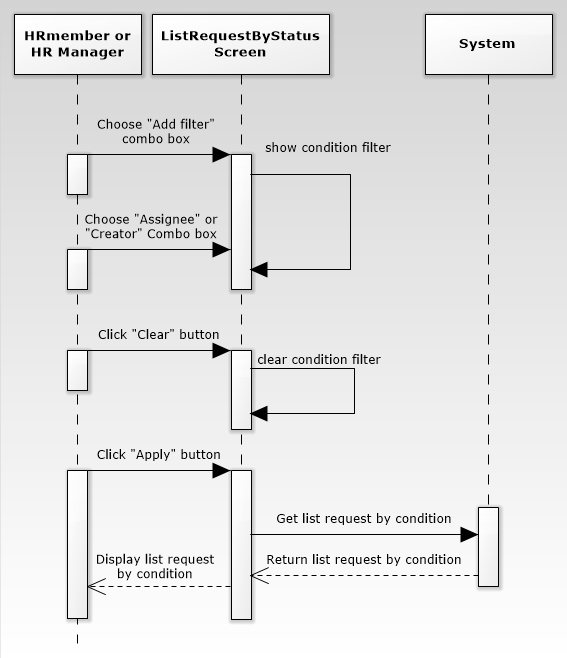
⮚      HR Member can choose condition filter list request base on the Assignee or/and Creator.

⮚       In addition, the  HR Member click on the title of each request to view the details of the request.

* Related functions: See request list *[4.1.1.2], [4.1.1.4]*.
* Screen Design & Data Description:



* Activities Flow
* Sequense diagram



* Pseudo code

**BEGIN MAINPROGRAM getListRequestByStatus(Assignee assignee, Creator creator)**

**IF** only filter by assignee **THEN**

Get list request with condition only assignee

**ELSE IF** only filter by creator **THEN**

Get list request with condition only creator

**ELSE IF** both filter by assignee, creator **THEN**

Get list request with both condition assignee, creator

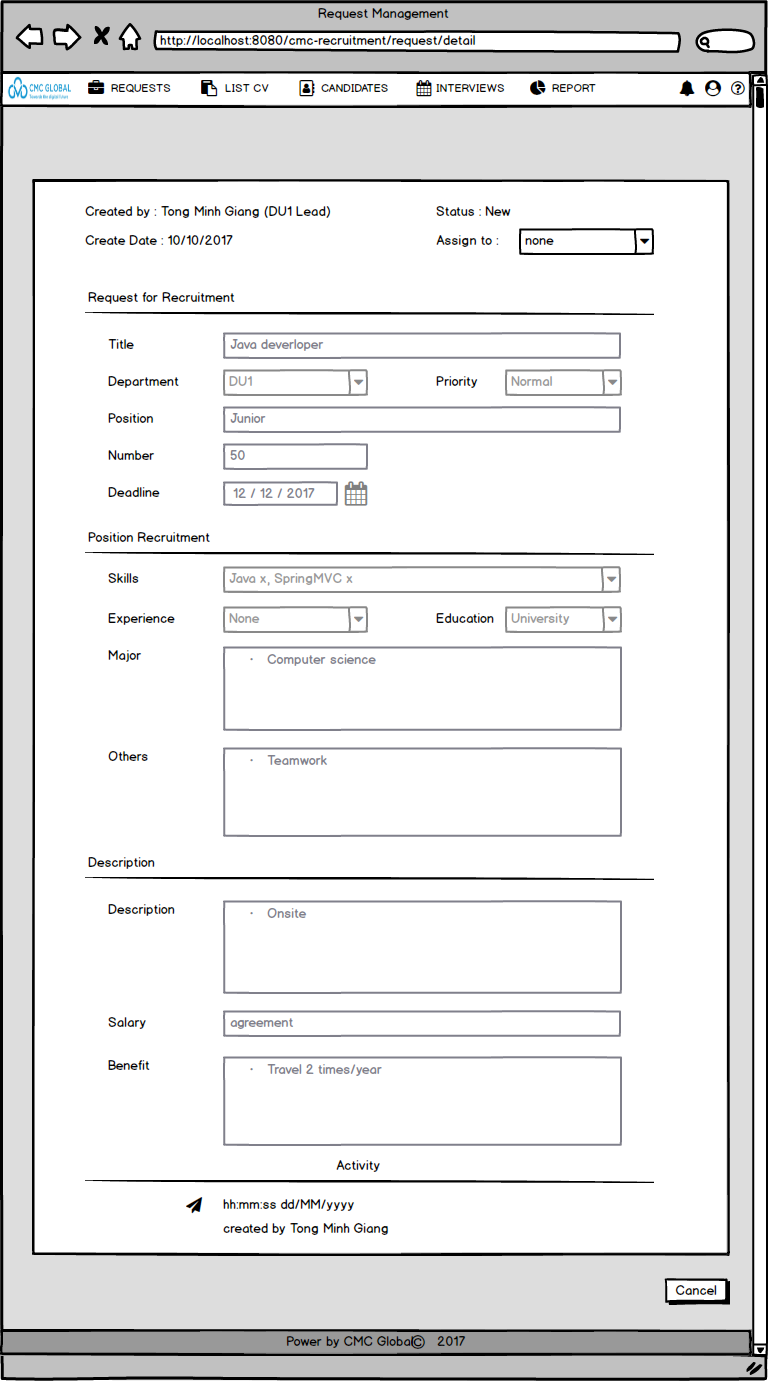
**ELSE THEN**

Get list all request

**ENDIF**

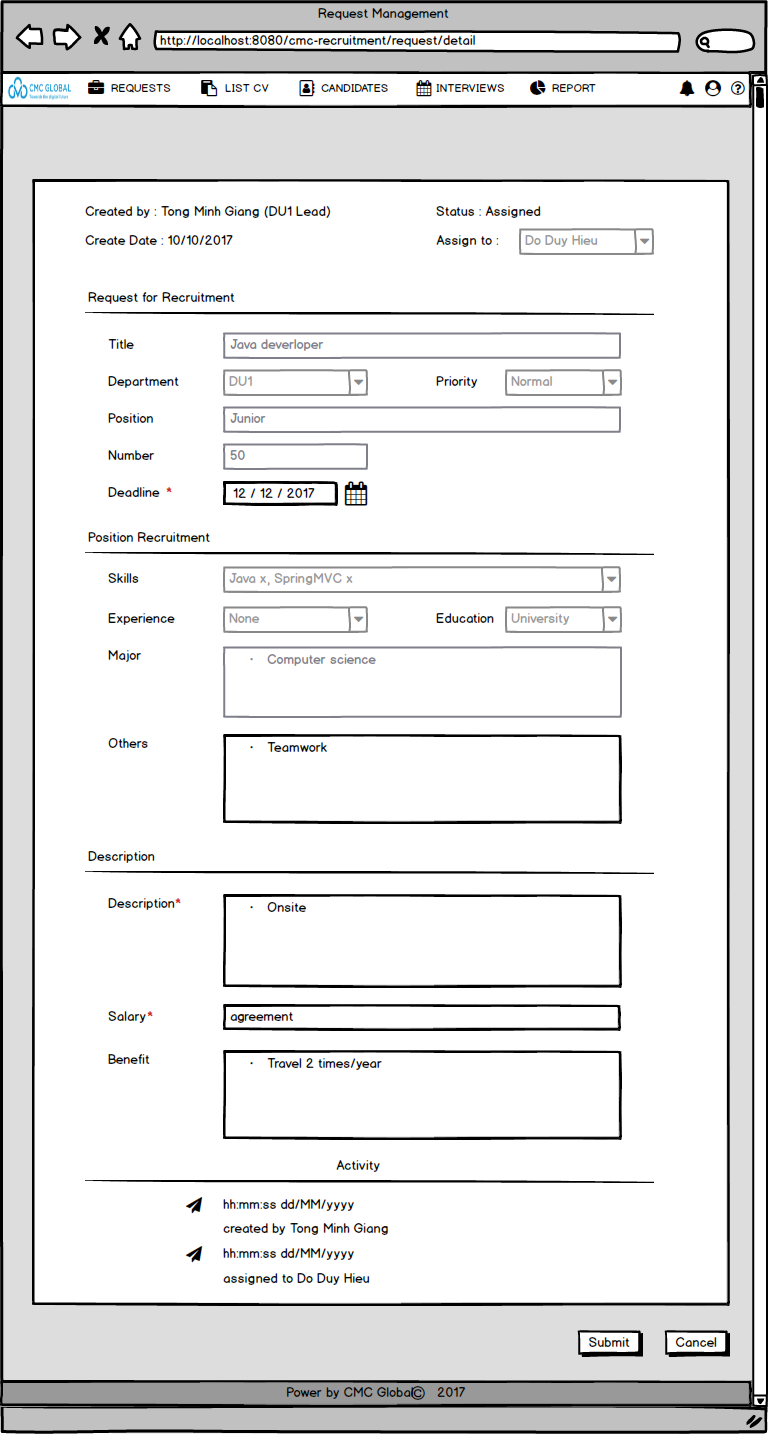
Display to screen list request result by status;

**END MAINPROGRAM getListRequestByStatus(Assignee assignee, Creator creator)**

* + - 1. View request detail
* Actor: HR Member
* Purpose: Display request detail.
* Required:
* From list requests screen (list published request screen, list request by status screen, list request by filter and search screen), user click on any request to view detail of that request.
* Other relevant functions: View List request *[from 4.1.1.1 to 4.1.1.3].*
* Screen Design & Data Description:
* The detail screen of new request 

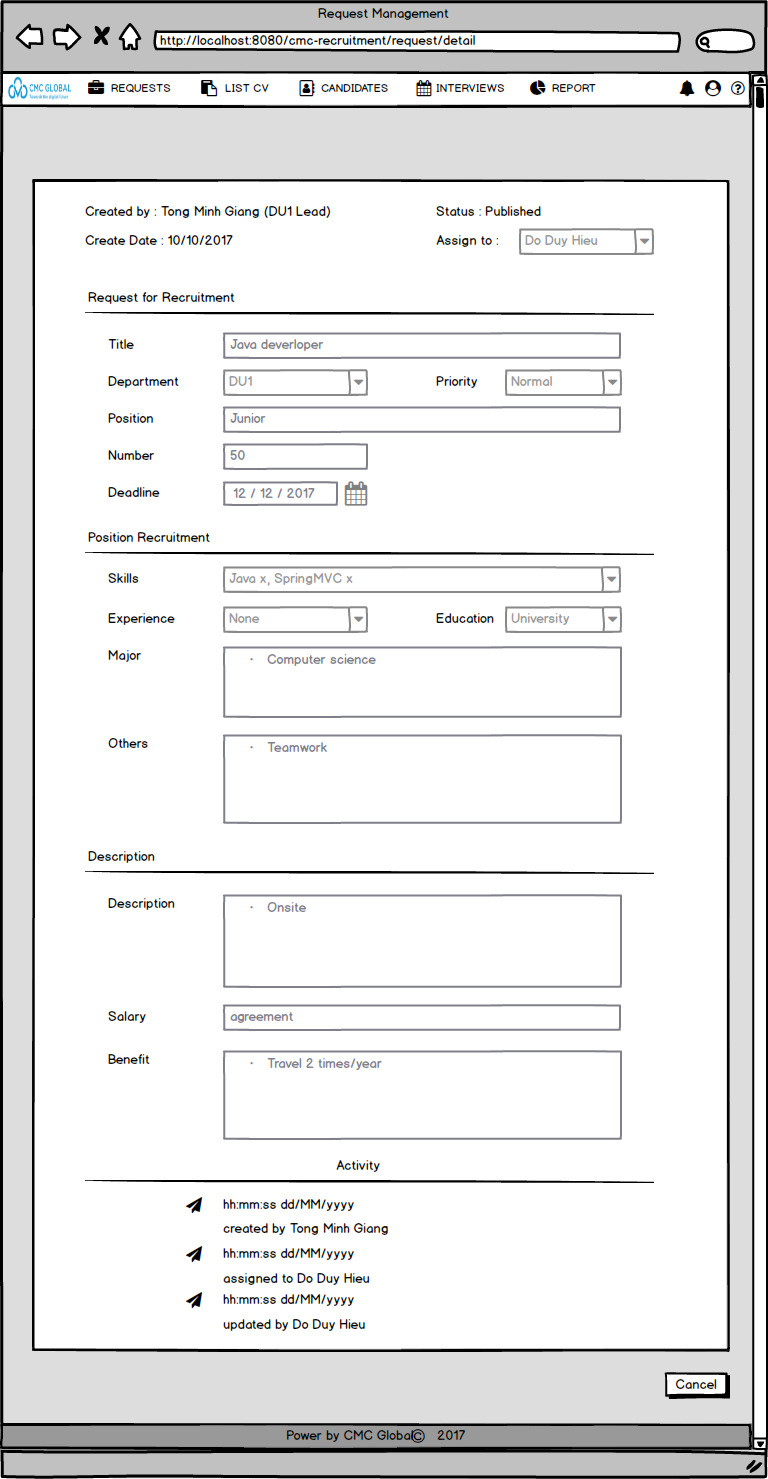
At this screen, HR Member can assign request to herself/himself.

* The detail screen of assigned/pending/rejected request



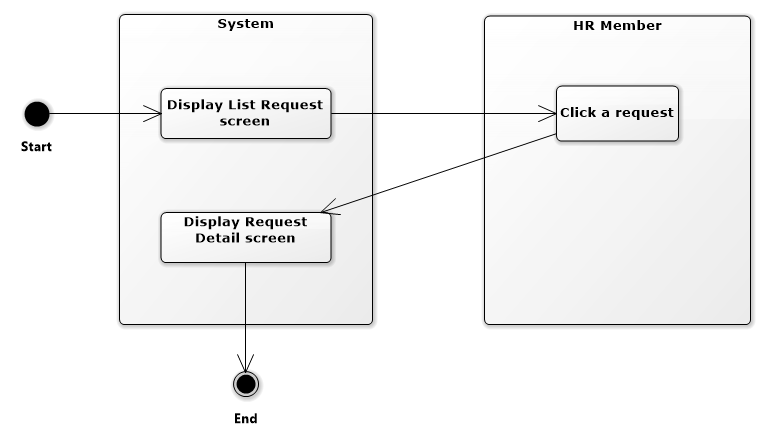
At this screen, HR Member can update some fields (deadline, others, description, salary, benefit).

* The detail screen of published/closed request

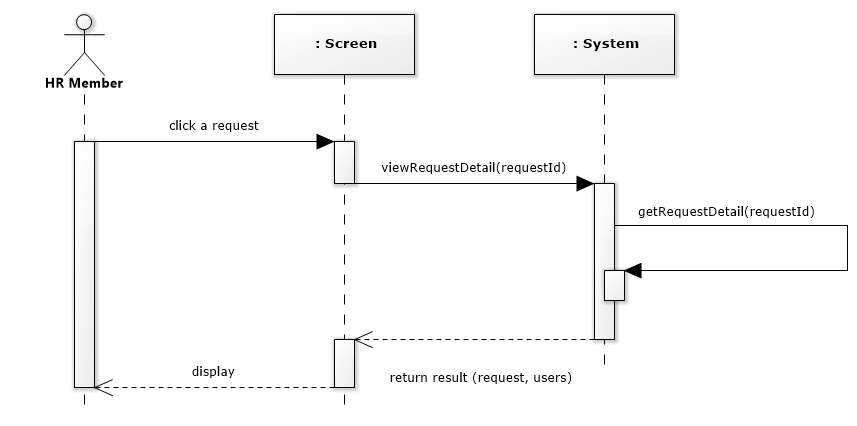


At this screen, HR Member only views request detail and do nothing.

* Activities Flow



* Sequense diagram



* Pseudo code

Require:

HR Member has logined.

HR Member click a request on list requests screen.

------------------------------------------------------------------------

**BEGIN MAINPROGRAM** **getRequestDetail(requestId)**

getRequestByRequestId(requestId);

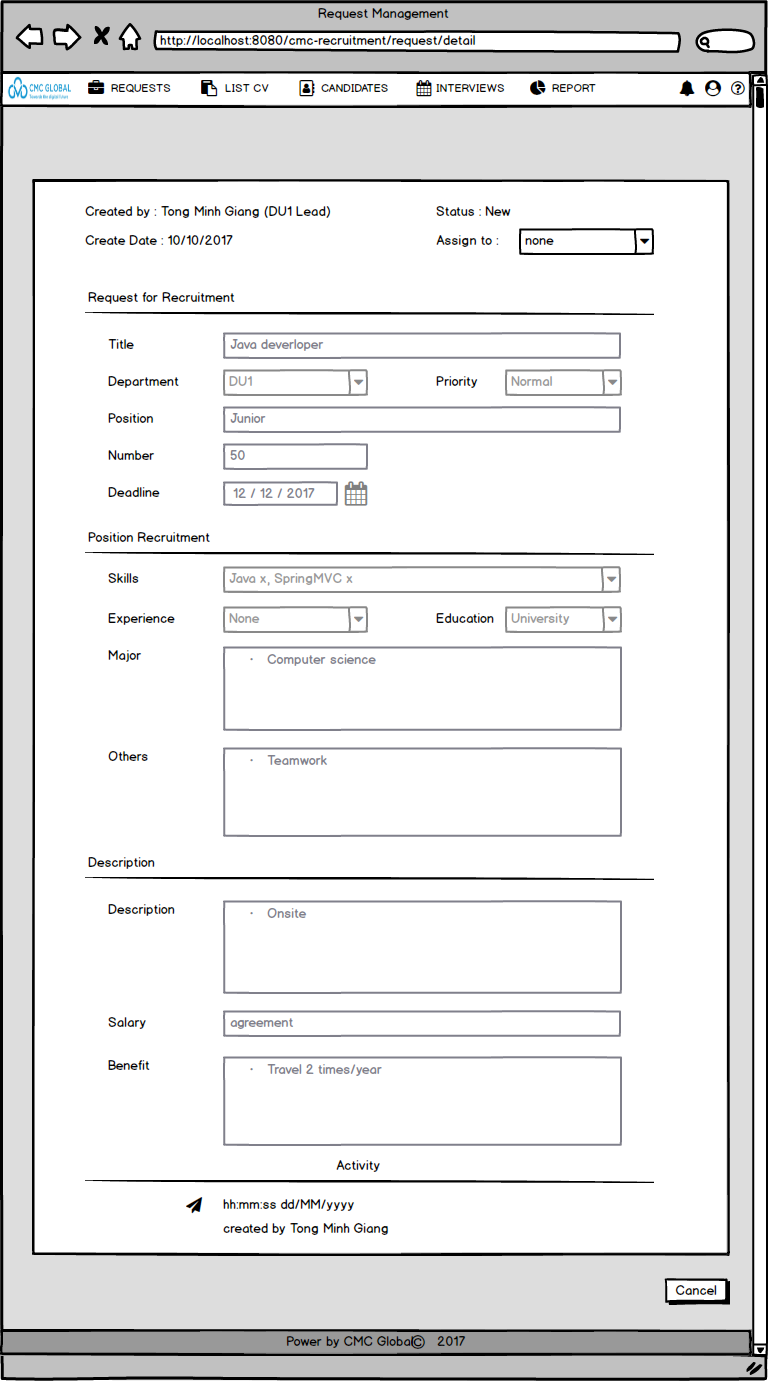
getAllHrMembers();

Set <<ShowedScreen>> = [Request Detail screen];

**END MAINPROGRAM getRequestDetail(requestId)**

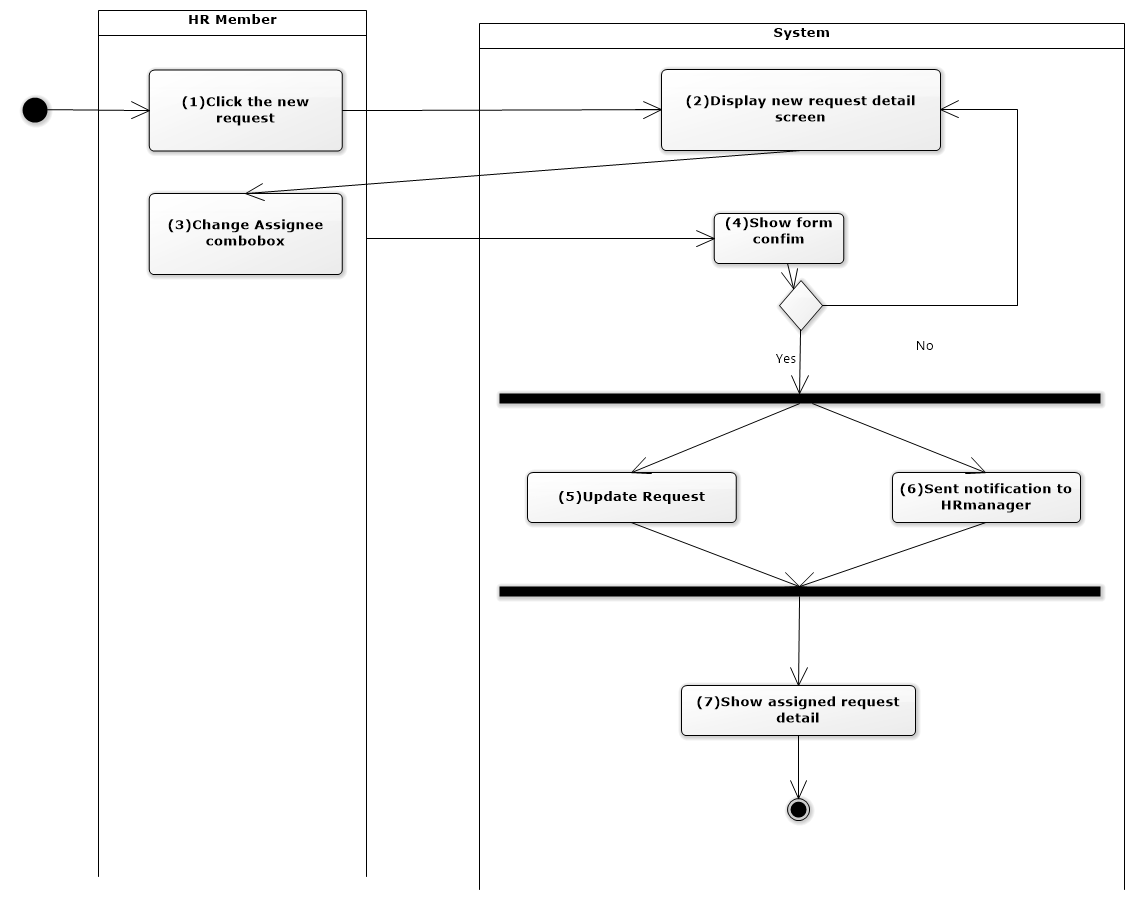
---------------------------------------------------------------------

* + - 1. Assign
* Actor : HR Member
* Purpose: HR Member can assign request themselves
* Request :
* For requests not assigned to HR Member, the request status will display "new", request detail screen has : creator, create date, assignee, and information necessary of request. HR Member can view and assign themselves by selecting “Me” in “Assignee” combobox and confirm “Yes” when system displays Confirm dialog.
* Other relevant functions: View List request *[4.1.1.4].*
* Screen Design & Data Description:

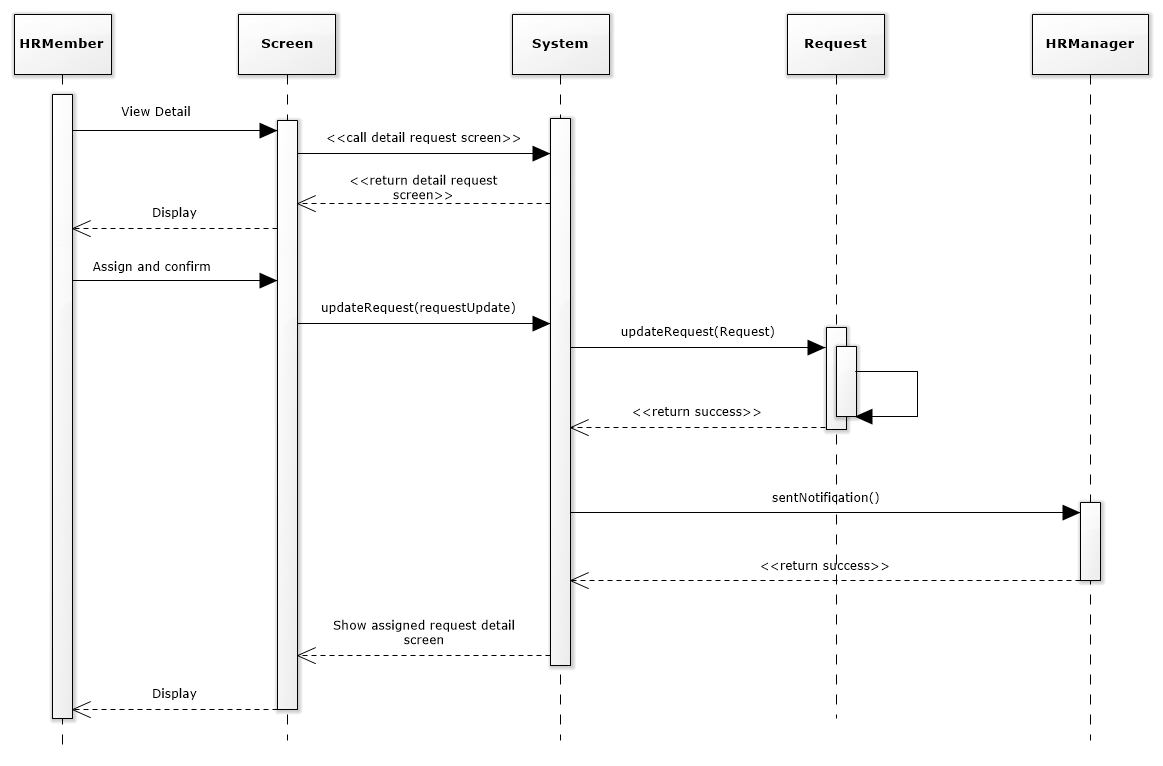




* Activities Flow



* Sequense diagram



* Pseudo code

Require :

HR Member has logined.

HR Member click new request in list requests screen.

--------------------------------------------------------------------

**BEGIN MAINPROGRAM updateRequest(RequestUpdate)**

HRMember change “assignee” combobox;

show dialog confim;

**IF** (chose "Yes" button ) **THEN**

System updateRequest(RequestUpdate);

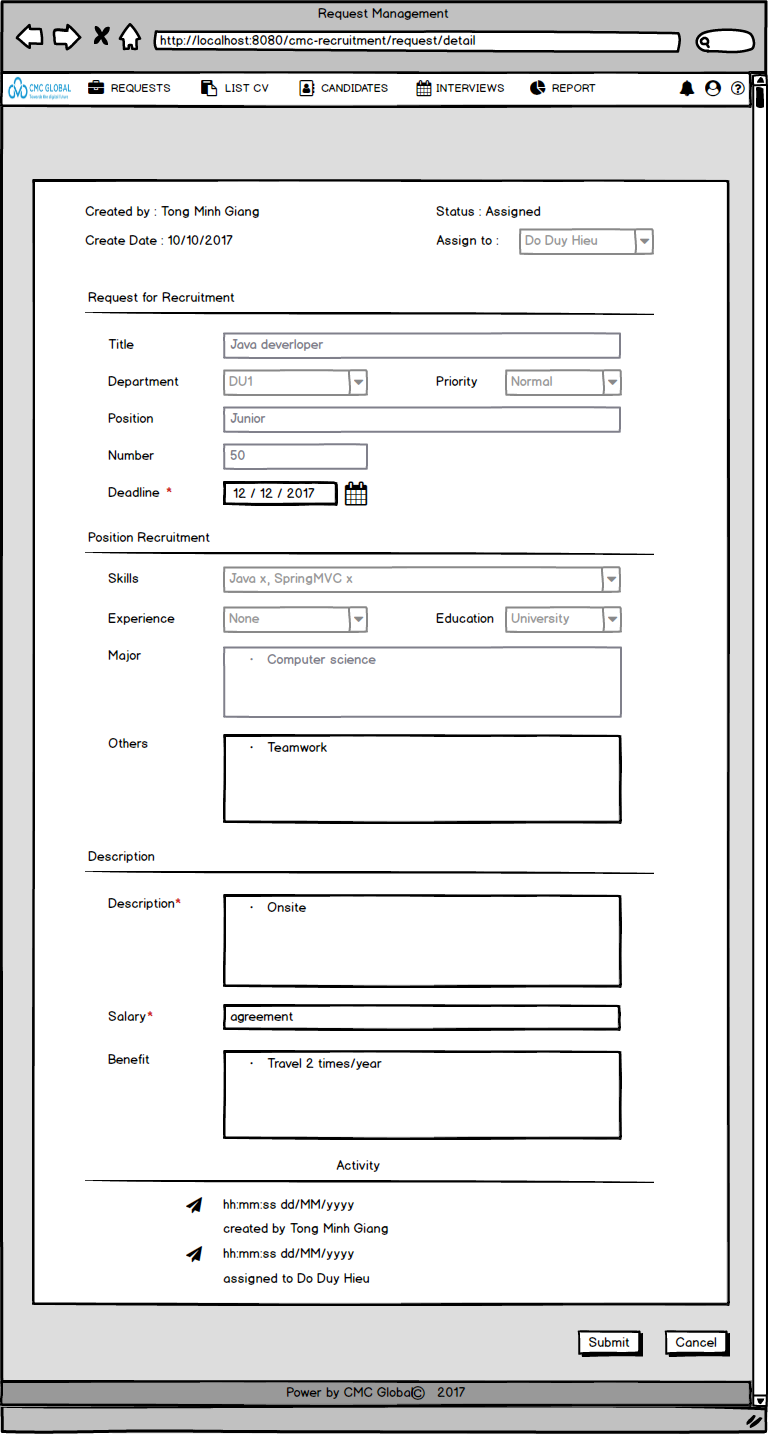
display assigned request detail screen;

**END IF**

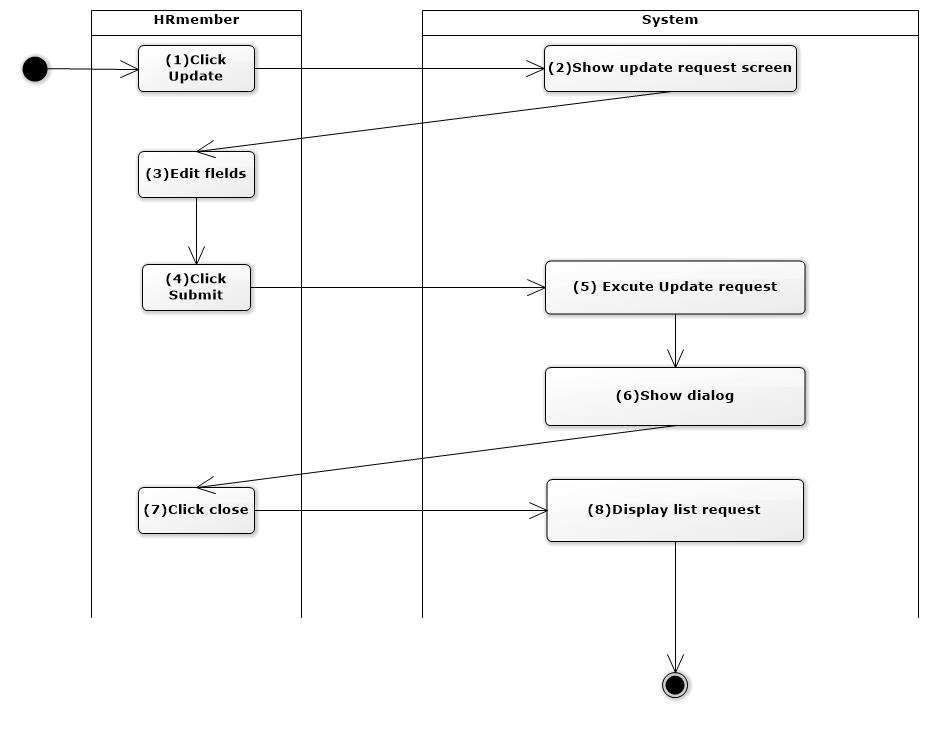
**END MAINPROGRAM updateRequest(RequestUpdate)**

-----------------------------------------------------------------

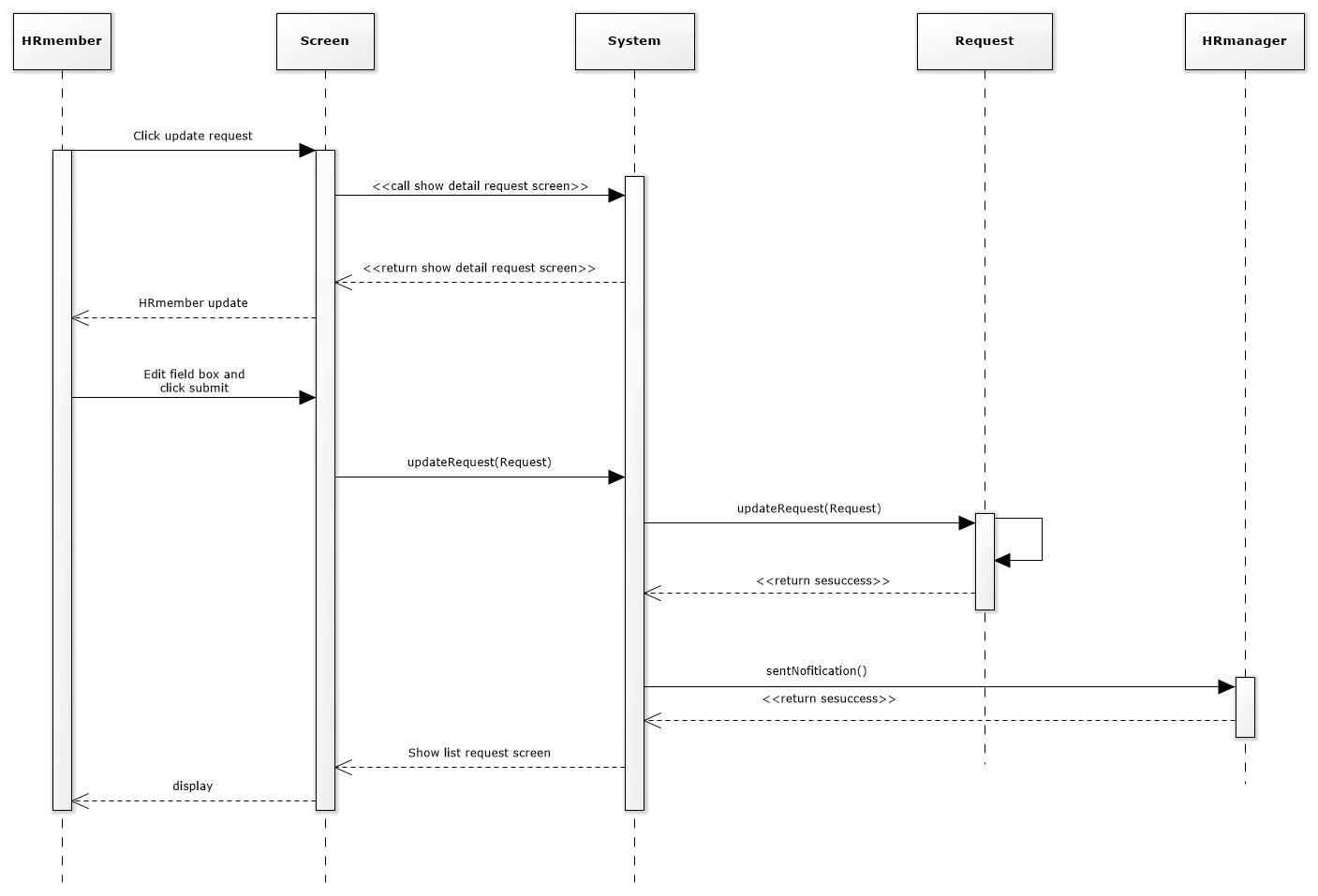
* + - 1. Update field
* Actor : HR Member
* Purpose: HR Member can update request
* Request :
* For assigned requests, request status is displayed “assigned”, request detail screen has : creator, create date, assignee, and information necessary of request.
  + HR Member will not edit or edit informtions that HR Member has the role to edit (Other, Description, Salary, Benefit, Deadline).
  + HR Member click “Submit” to update request. If request is updated successful, the request status is "Pending", and awaits HR Manager approval. The system sends notifications to HR Manager, display notifical dialog. HR Member confirm Close to return list published requests screen.
* Other relevant functions: View List request *[4.1.1.4].*
* Screen Design & Data Description:



* Activities Flow



* Sequense diagram



* Pseudo code

**Require** :

HR Member has logged-in.

HR Member views Request detail screen.

--------------------------------------------------------------------

**BEGIN MAINPROGRAM updateRequest(****RequestUpdate)**

HRMember edits fields;

HR Member click "Submit" button;

call method update(RequestUpdate);

**IF** (Request is updated successfull ) **THEN**

System shows notifical;

HR Member click "Close" button;

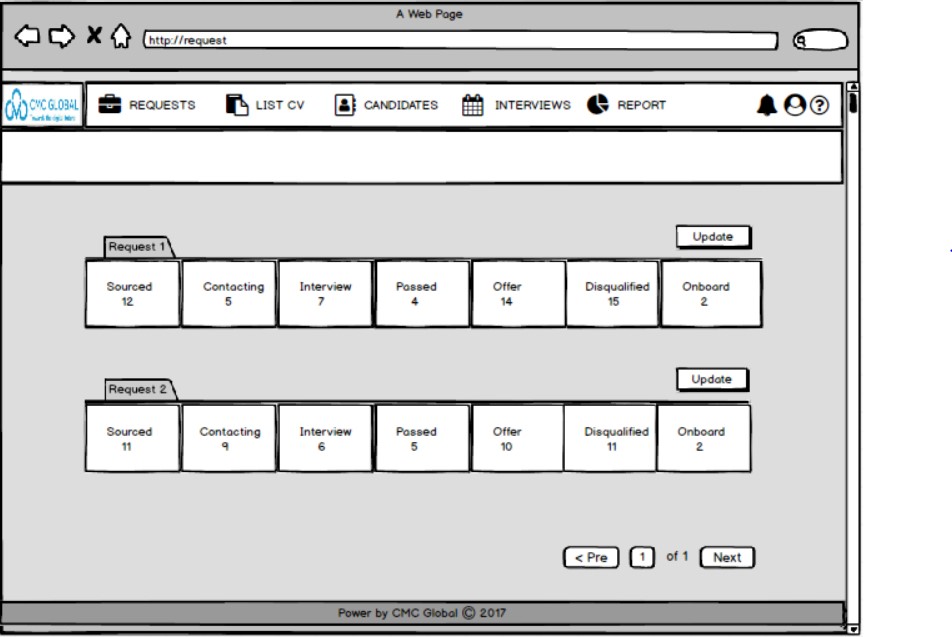
display list published requests screen;

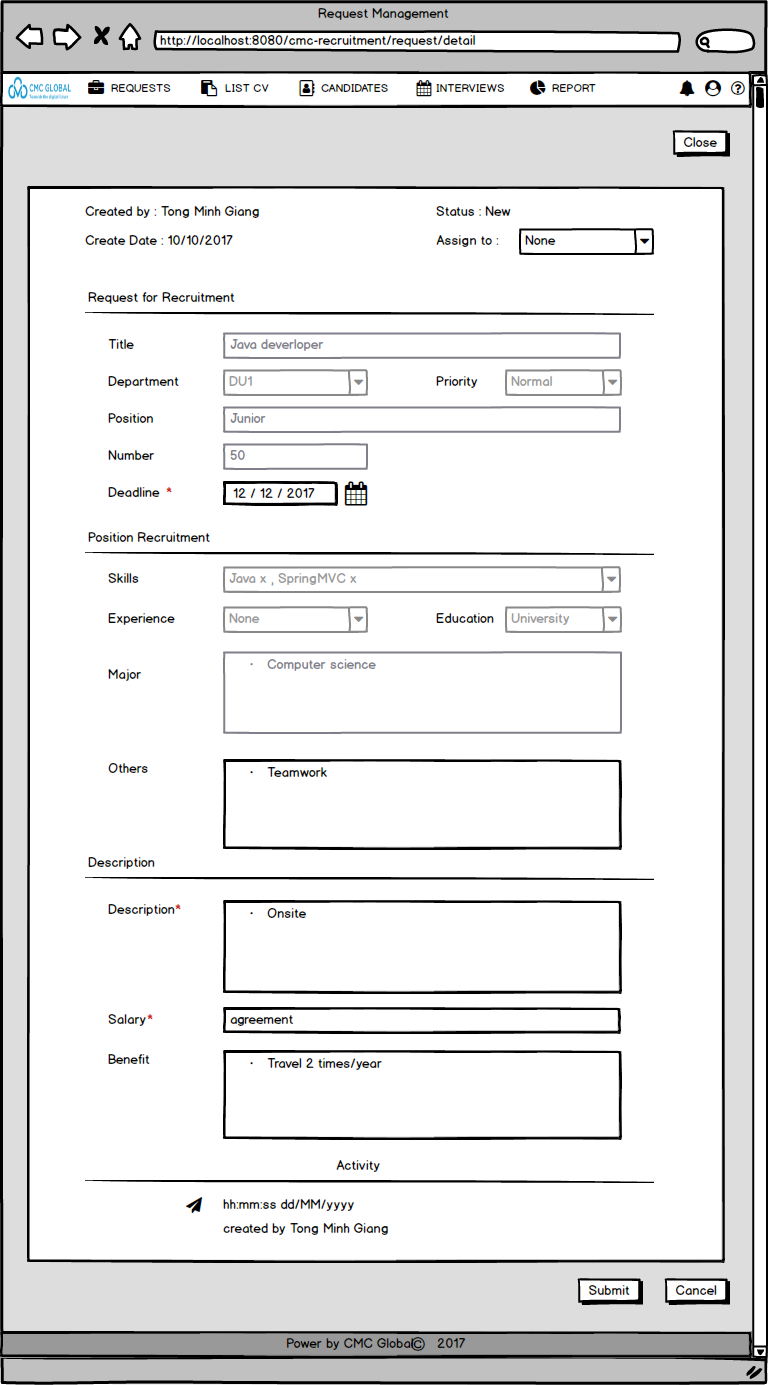
**END IF**

**END MAINPROGRAM updateRequest(RequestUpdate)**

-------------------------------------------------------------------

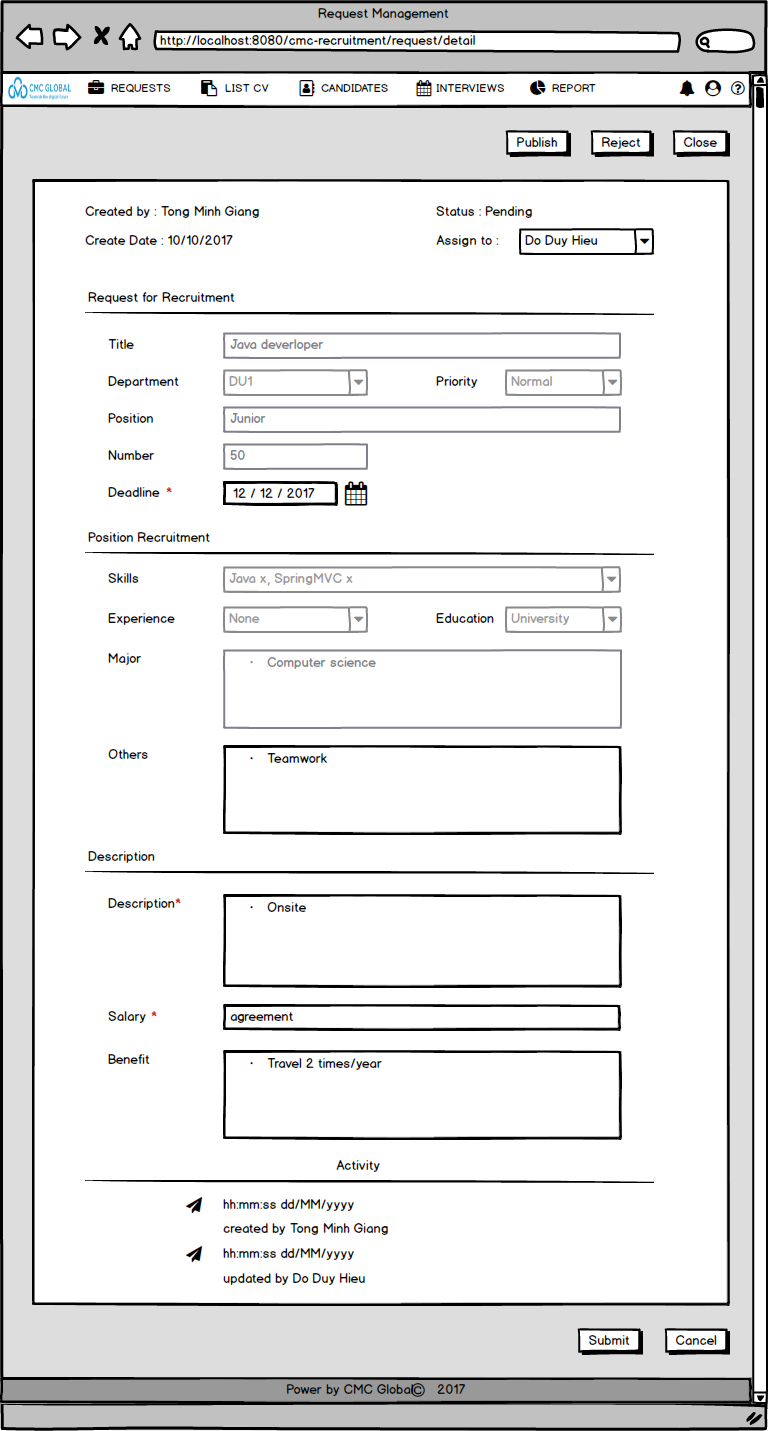
* 1. HR Manager
     1. Manager request
        1. View list published request



* In addition to the functions of Hr member, Hr manager also has an update feature. Hr member can only update information for himself, Hr manager can update for all Hr member. When click on the “Update” button to view detail of that request then update information of request.
  + - 1. View request detail
* Actor: HR Manager
* Purpose: Display request detail.
* Required:
* From list requests screen (list published request screen, list request by status screen or list request by filter and search screen), user click on any request title or “Update” button to view request detail.
* Other relevant functions: View list request *[from 4.1.1.1 to 4.1.1.3].*
* Screen Design & Data Description:
* The detail screen of new/assigned/published/rejected request 

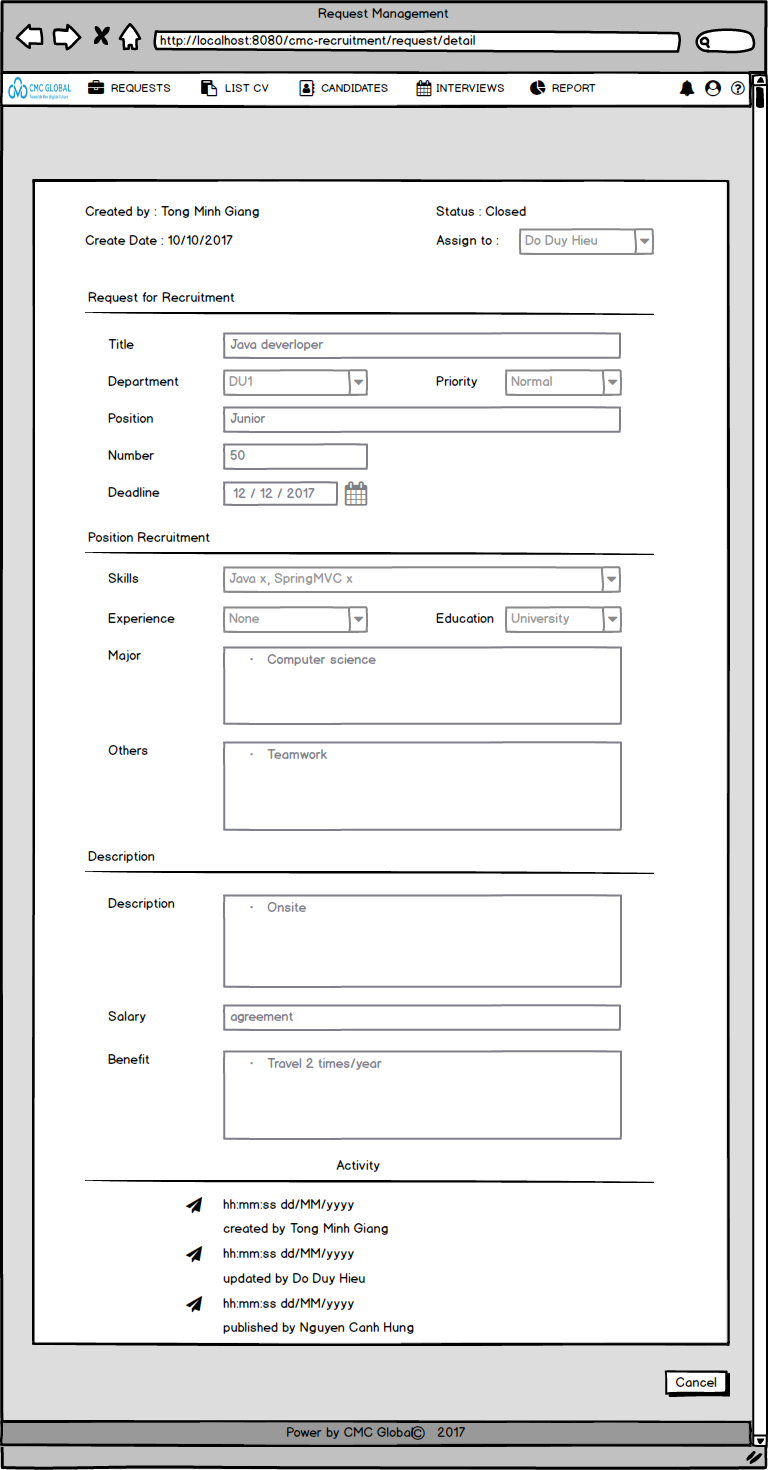
At this screen, HR Manager can assign request to HR Member, update some fields of the request or close it.

* The detail screen of pending request



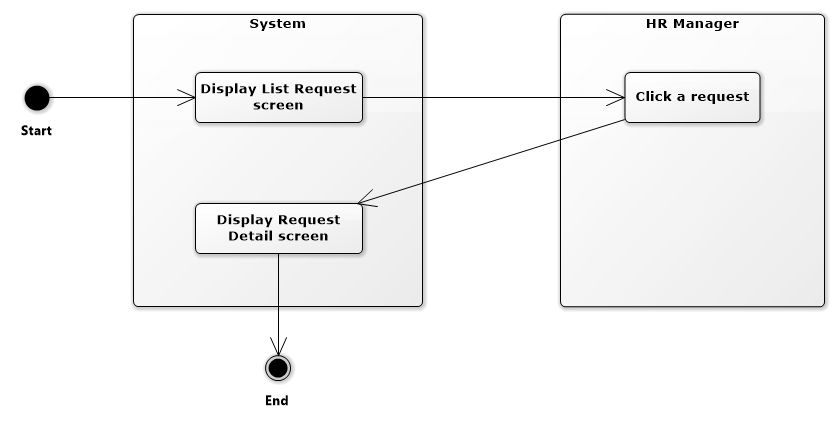
At this screen, HR Manager can assign request to other HR Member, update some fields of the request. In addition, she/he can publish, reject or close it.

* The detail screen of closed request

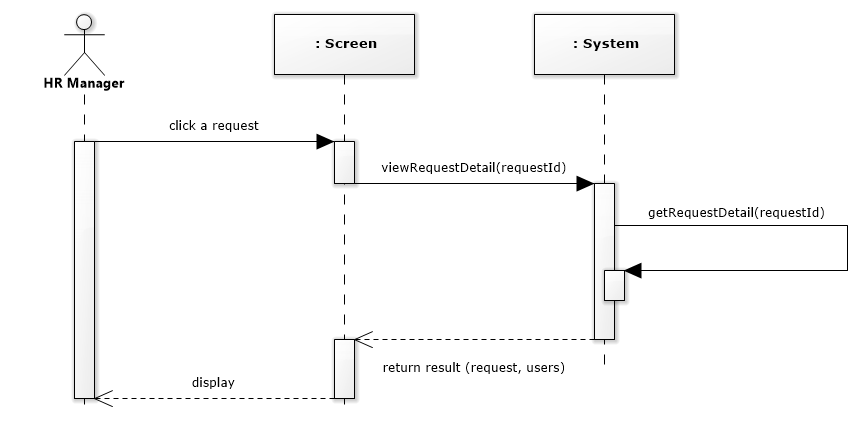


At this screen, HR Manager only views request detail and do nothing.

* Activities Flow



* Sequense diagram



* Pseudo code

Require:

HR Manager has logged-in.

HR Manager click a request on list requests screen

------------------------------------------------------------------------

**BEGIN MAINPROGRAM getRequestDetail(requestId)**

getRequestByRequestId(requestId);

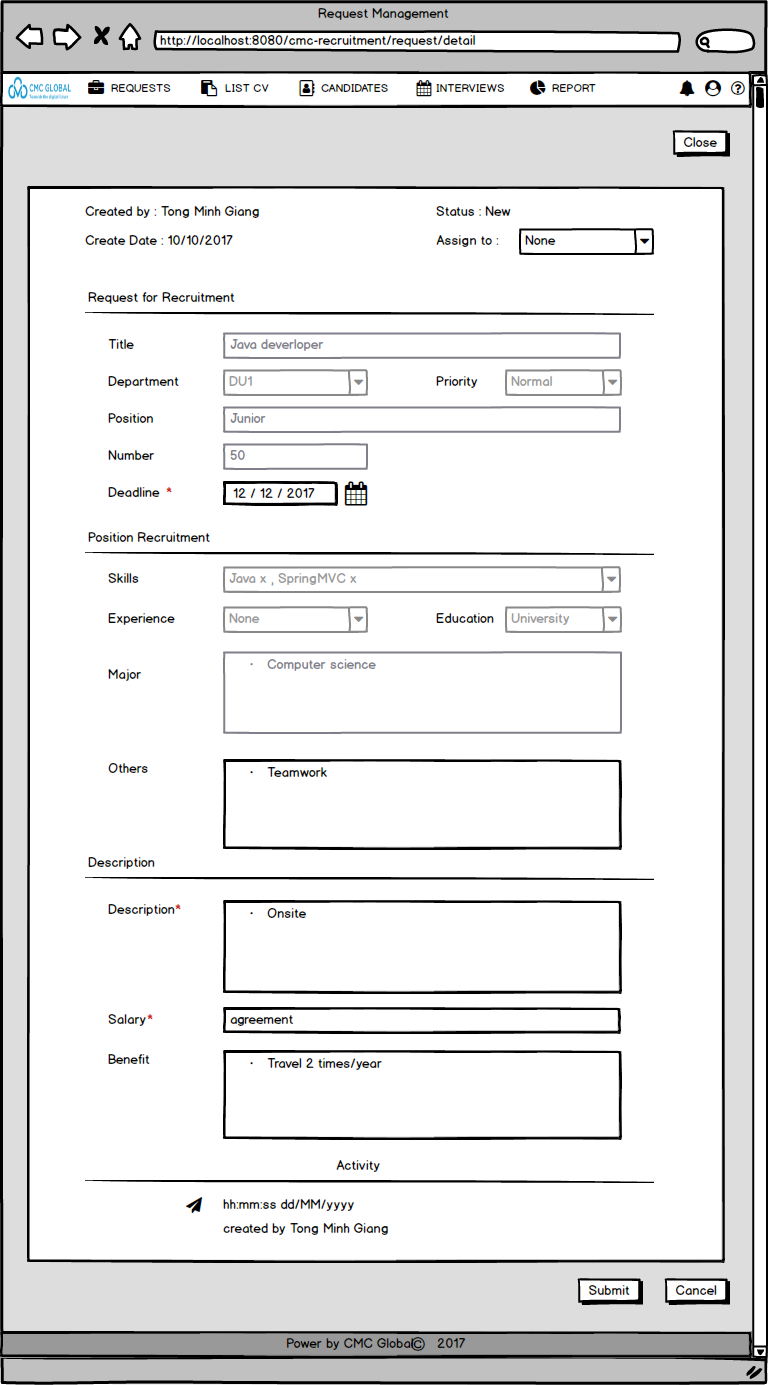
getAllHrMembers();

Set <<ShowedScreen>> = [Request Detail screen];

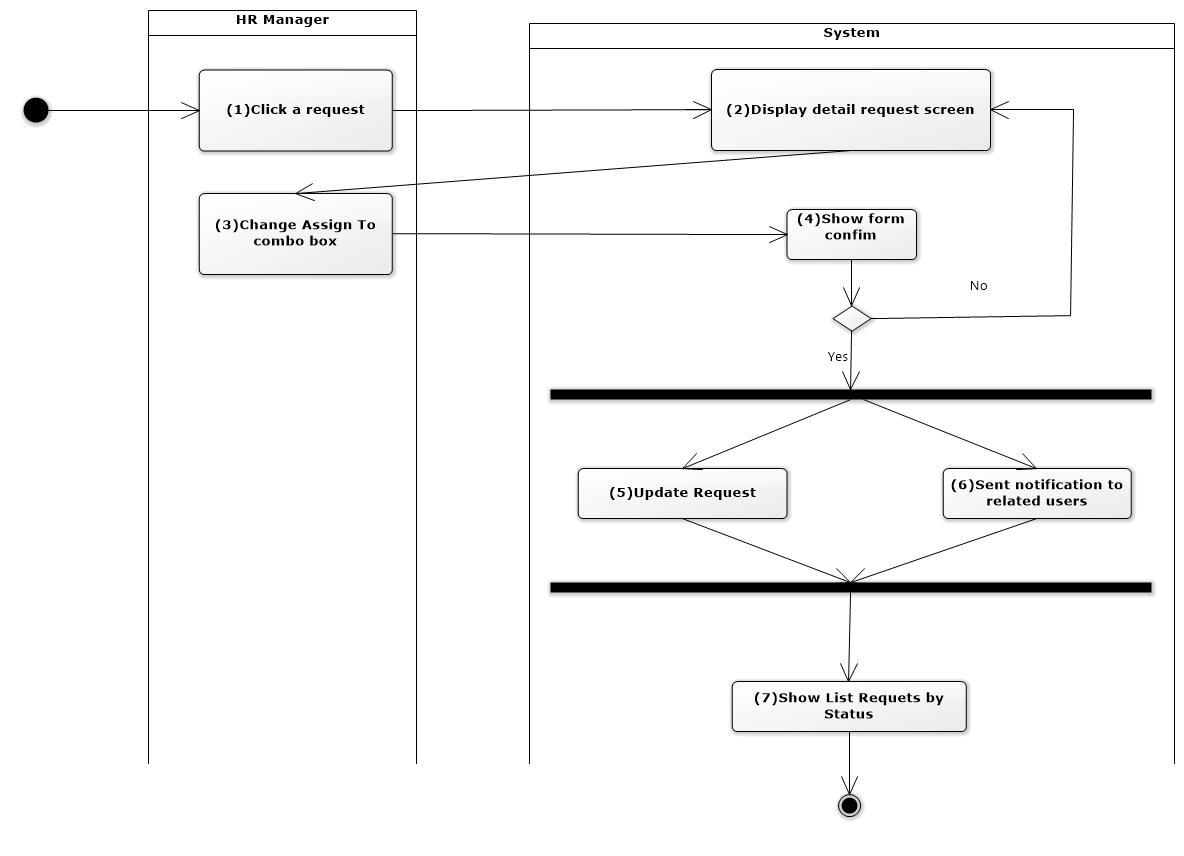
**END MAINPROGRAM getRequestDetail(requestId)**

---------------------------------------------------------------------

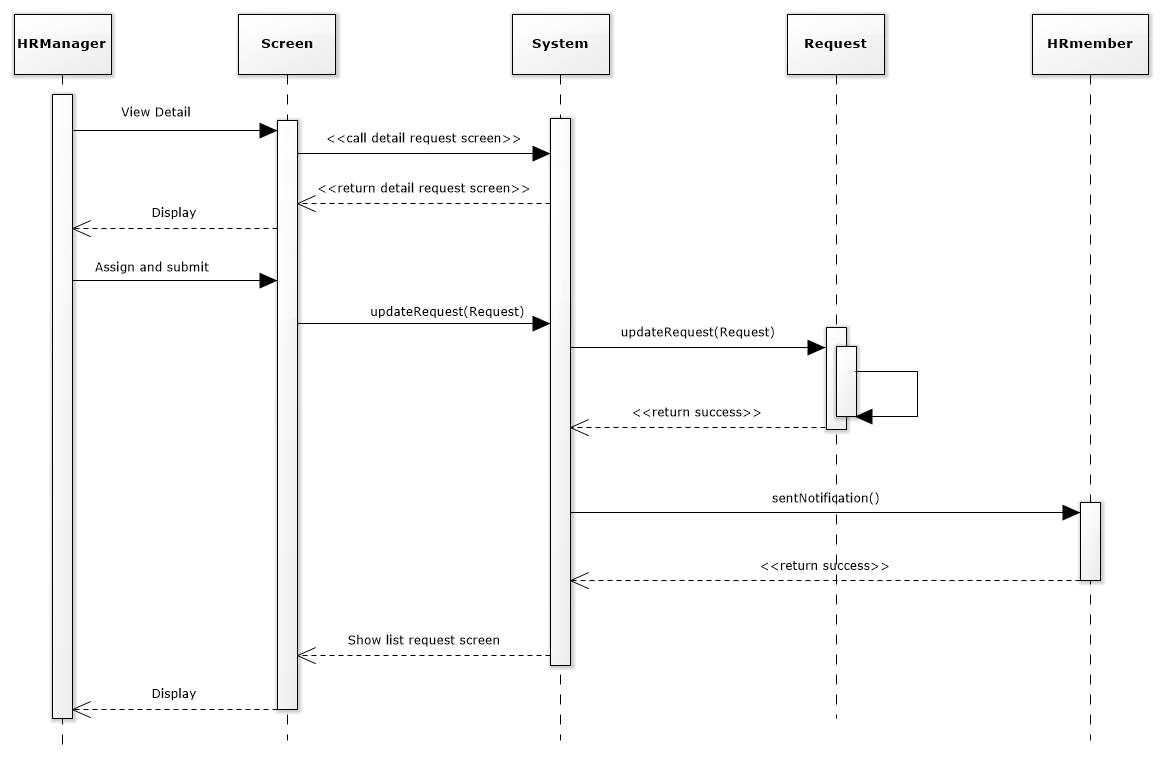
* + - 1. Assign
* Actor : HR Manager
* Purpose: HR Manager can assign request
* Request :
* At Request detail screen:
  + HR Manager will assign the request to any HR Member by selecting HR member name in “Assignee” combobox.
  + The system display a dialog confirm. If select “Yes”, system will execute assign request.
* Other relevant functions: View List request *[4.2.1.2].*
* Screen Design & Data Description:



* Activities Flow



* Sequense diagram



* Pseudo code

Require :

HR Manager has logged-in.

HR Manager click a request in list requests screen.

--------------------------------------------------------------------

**BEGIN MAINPROGRAM updateRequest(Request)**

HRMember change “assignee” combobox;

show confim dialog;

**IF** (chose "Yes" button ) **THEN**

System updateRequest(Request);

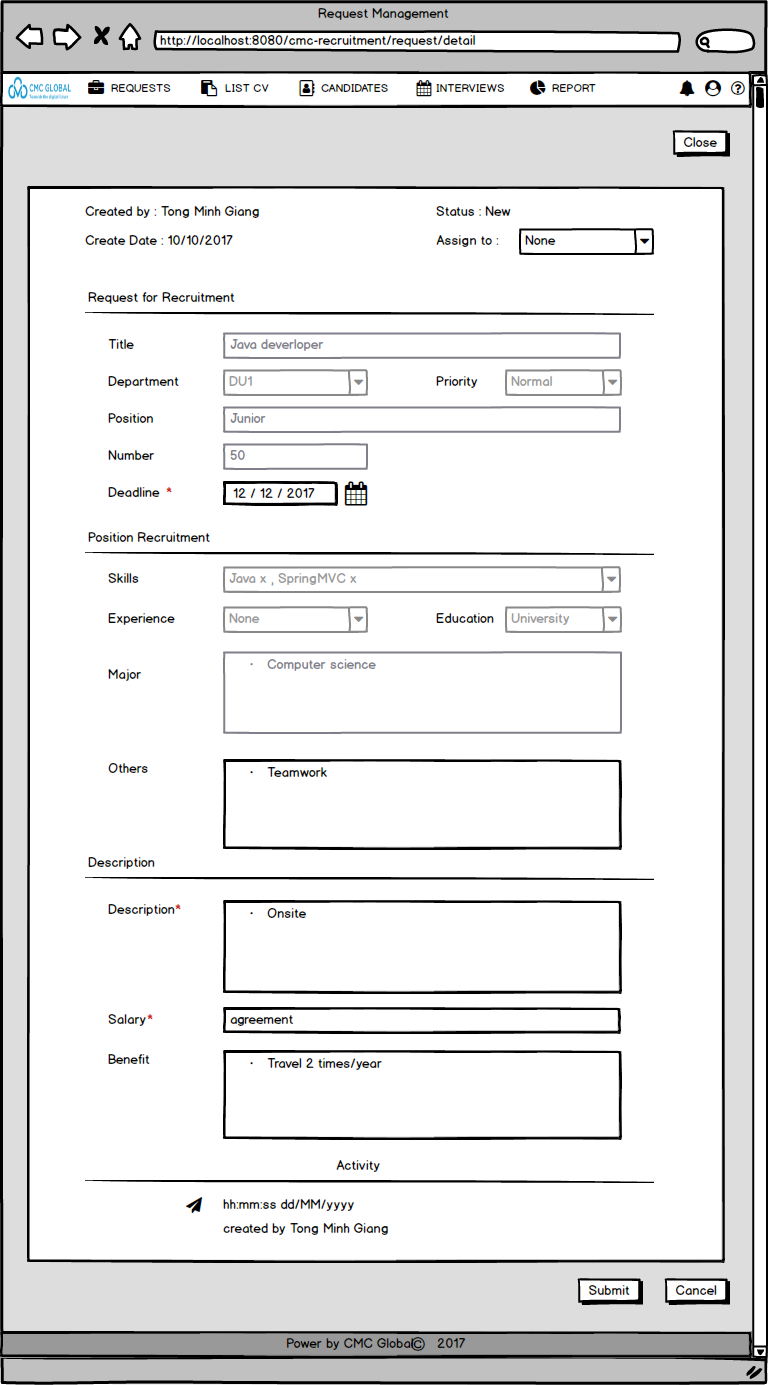
display list requests screen;

**END IF**

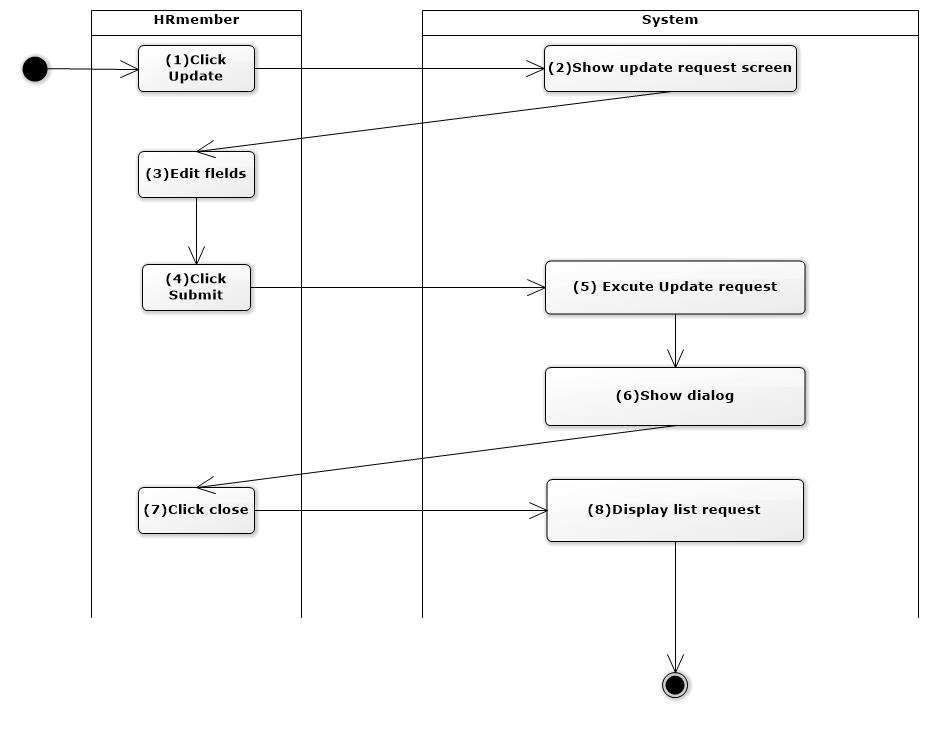
**END MAINPROGRAM updateRequest(Request)**

-----------------------------------------------------------------

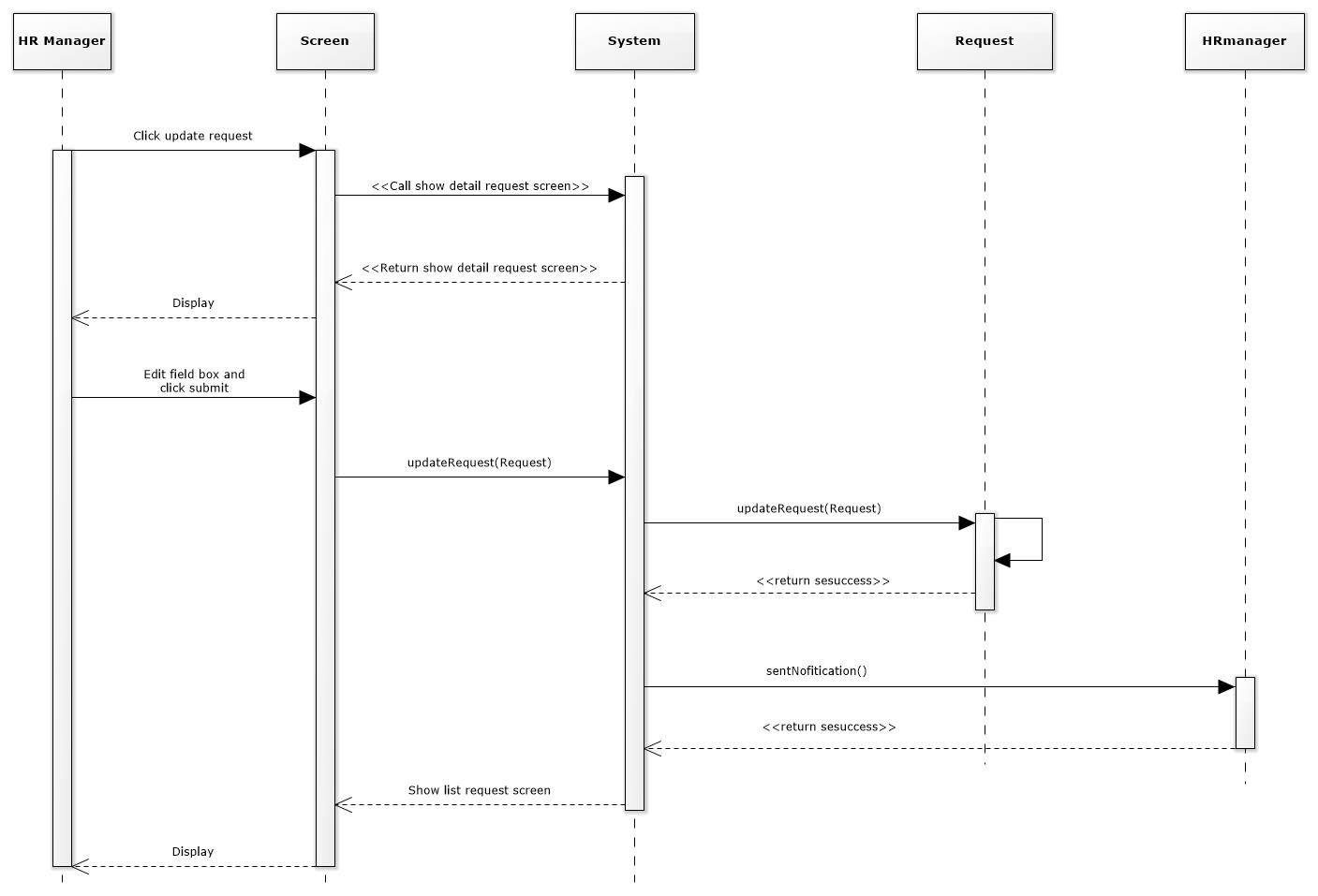
* + - 1. Update field
* Actor : HR Manager
* Purpose: HR Manager can update request
* Request :
* At Request detail screen:
  + HR Manager will not edit or edit informtions that HR Manager has the role to edit (Other, Description, Salary, Benefit, Deadline).
  + HR Manager click “Submit” to update request. If request is updated successful, system display notifical dialog. HR Manager confirm “Close” to return list previous requests screen.
* Other relevant functions: View List request *[4.2.1.2].*
* Screen Design & Data Description:



* Activities Flow



* Sequense diagram



* Pseudo code

Require :

HR Manager has logined.

HR Manager views Request detail screen.

--------------------------------------------------------------------

**BEGIN MAINPROGRAM updateRequest(RequestUpdate)**

HR Manager edits fields;

HR Manager click "Submit" button;

call method update(RequestUpdate);

**IF** (Request is updated successfull ) **THEN**

System shows notifical;

HR Manager click "Close" button;

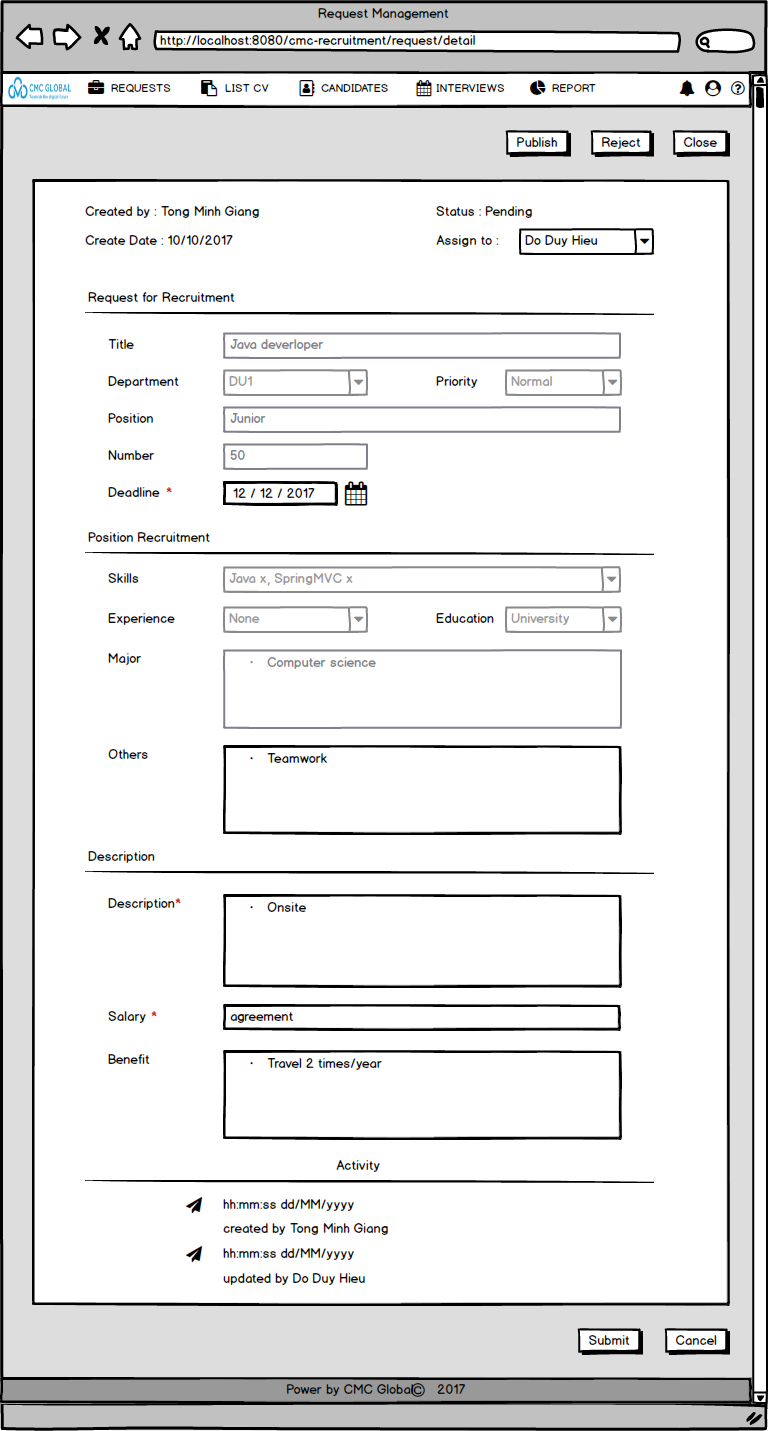
display list published requests screen;

**END IF**

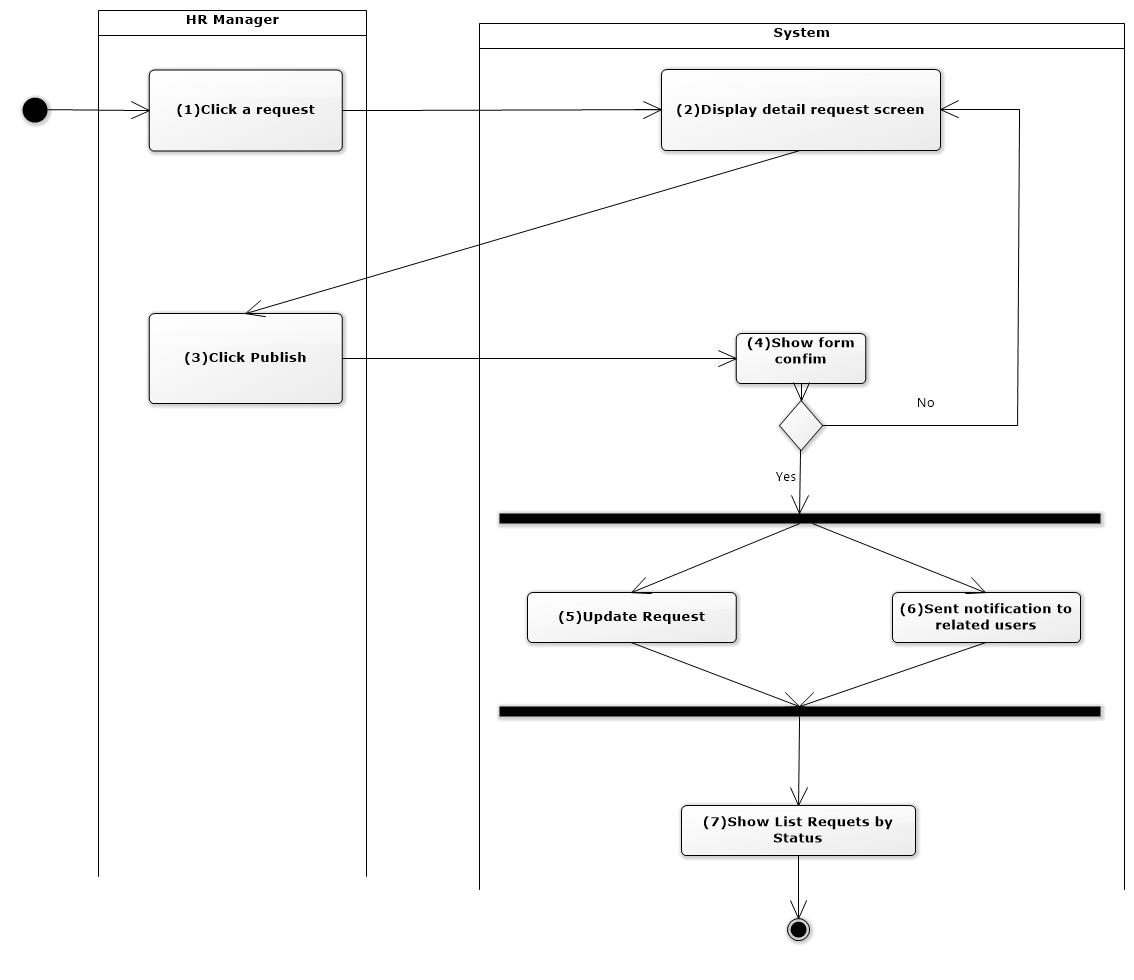
**END MAINPROGRAM updateRequest(RequestUpdate)**

-------------------------------------------------------------------

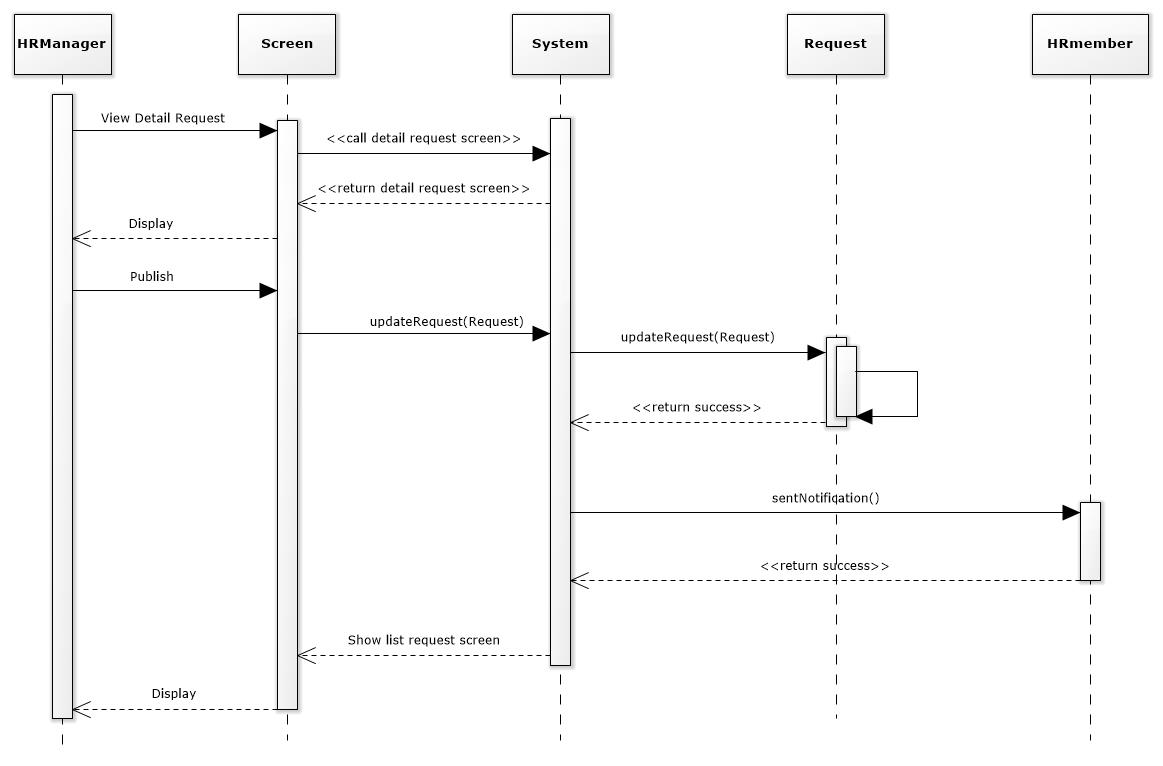
* + - 1. Publish
* Actor : HR Manager
* Purpose: HR Manager can publish request
* Request :
* At Request detail screen:
  + For pending requests on request detail screen, HR Manager click “Publish” to update request status to “Published”, the system display a confirm dialog. If select “Yes”, system execute publish request and the request will be updated.
* Other relevant functions: View List request *[4.2.1.2].*
* Screen Design & Data Description:



* Activities Flow



* Sequense diagram



* Pseudo code

Require :

HR Manager has logined.

HR Manager click a pending request in list requests screen.

--------------------------------------------------------------------

**BEGIN MAINPROGRAM updateRequest(RequestUpdate)**

HR Member click "Publish" button;

show dialog confim update;

**IF** (chose "Yes" button ) **THEN**

System updateStatusRequest(RequestUpdate);

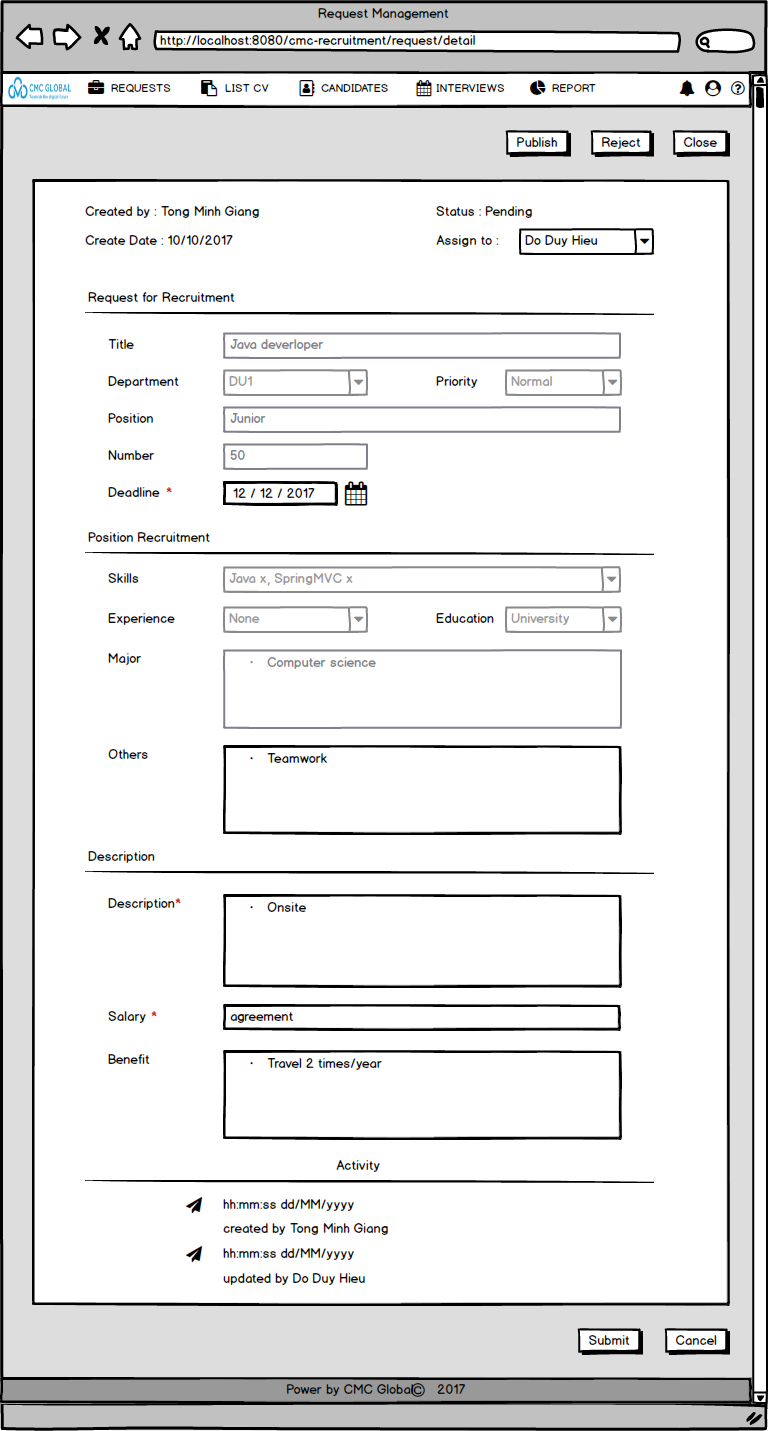
display list requests screen;

**END IF**

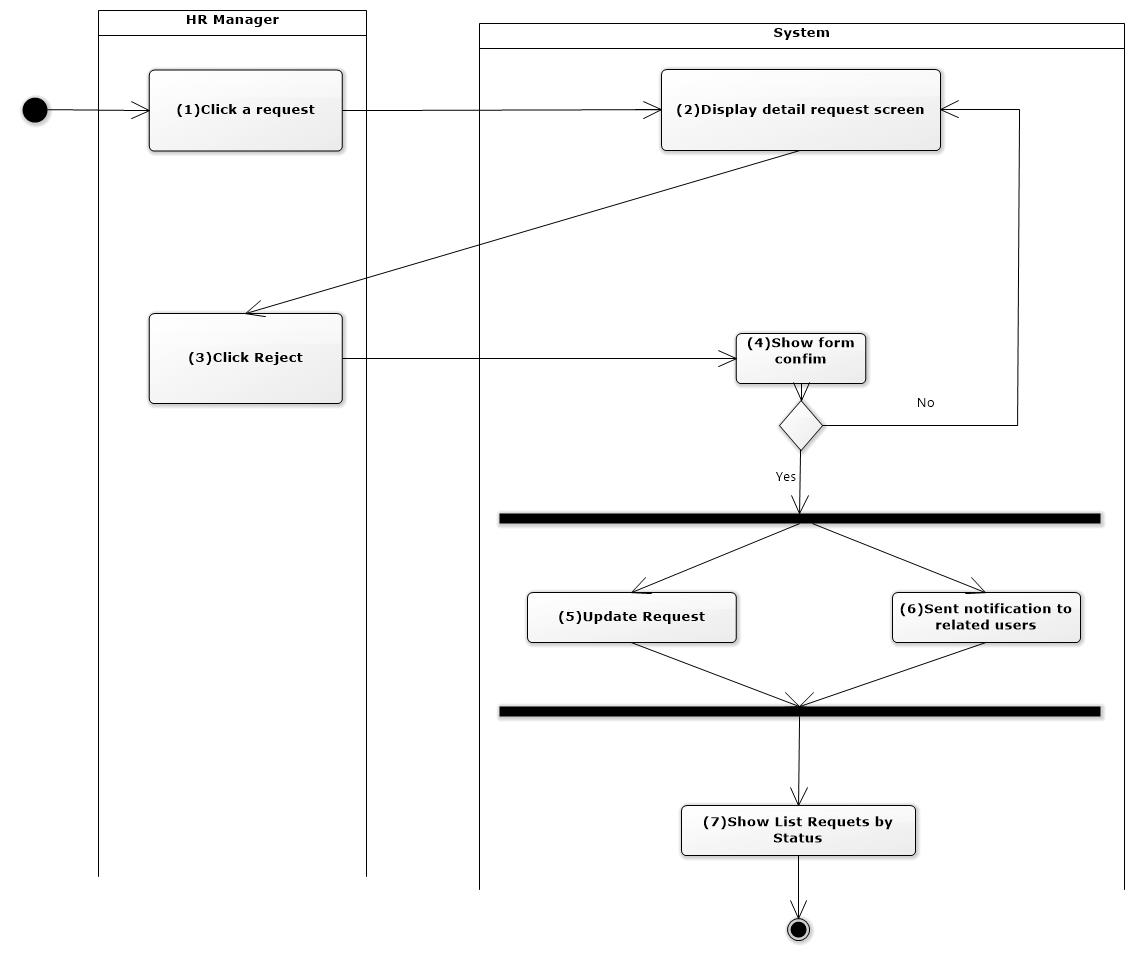
**END MAINPROGRAM updateRequest(RequestUpdate)**

-----------------------------------------------------------------

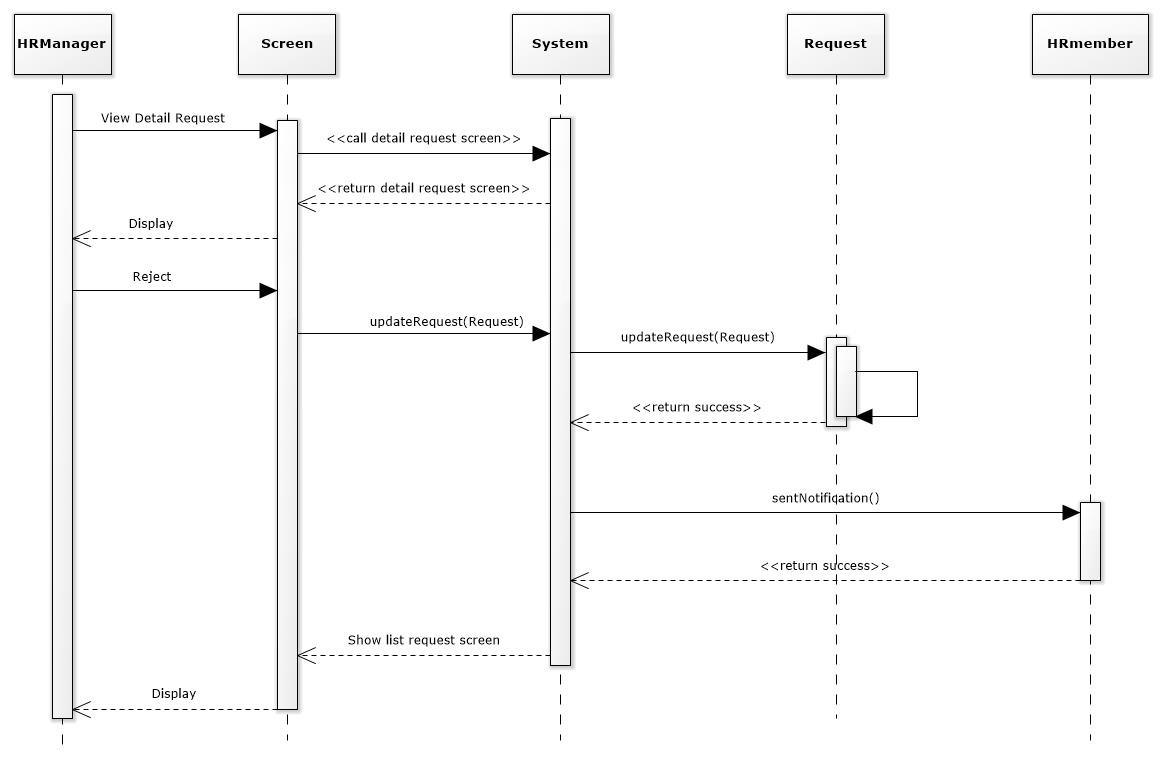
* + - 1. Reject
* Actor : HR Manager
* Purpose: HR Manager can reject request
* Request :
* At Request detail screen:
  + For pending requests on request detail screen, HR Manager click “Reject” to reject request, the system display a dialog confirm. If select “Yes” execute reject request and the request will be updated.
* Other relevant functions: View List request *[4.2.1.2].*
* Screen Design & Data Description:



* Activities Flow



* Sequense diagram



* Pseudo code

Require :

HR Manager has logged-in.

HR Manager click a pending request in list requests screen.

--------------------------------------------------------------------

**BEGIN MAINPROGRAM updateRequest(RequestUpdate)**

HR Manager click "Reject" button;

show confim dialog;

**IF** (chose "Yes" button ) **THEN**

System updateRequest(RequestUpdate);

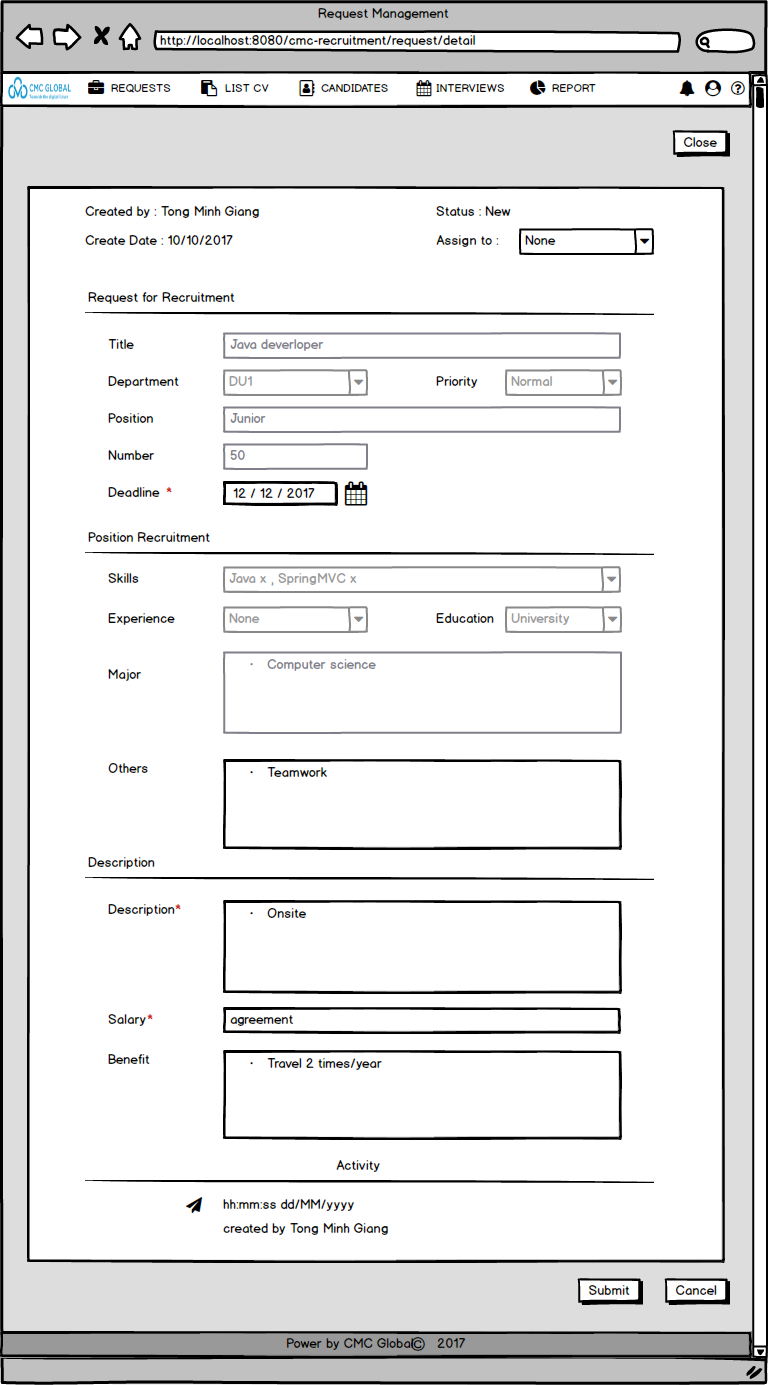
display list requests screen;

**END IF**

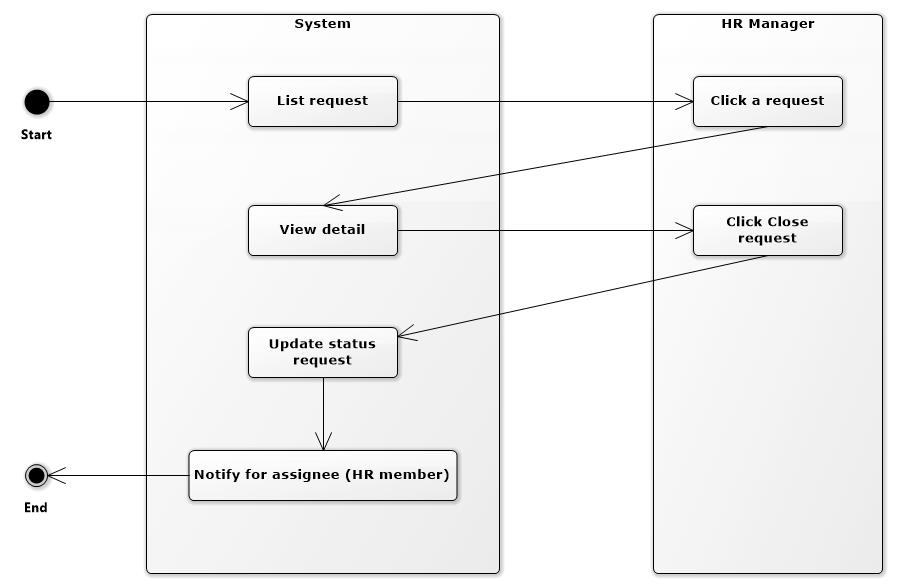
**END MAINPROGRAM updateRequest(RequestUpdate)**

-----------------------------------------------------------------

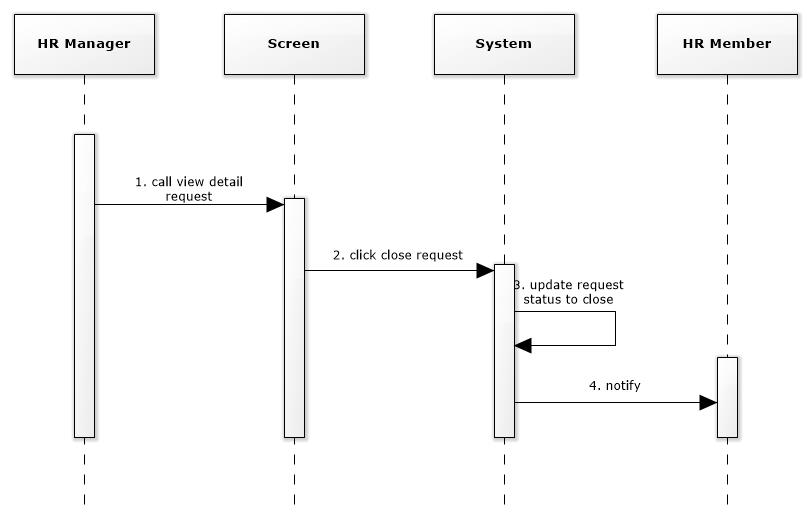
* + - 1. Close request
* Actor: HR Manager
* Purpose: Close request
* Request:
* On the request view detail screen, when the HR Manager clicks the "Close" button, the System displays a confirm dialog. If the HR Manager selects "Yes", the request will be closed and display the notifical to the creator (DU Lead), the assigned person (HR Member).
* Other relevant functions: View List request *[4.2.1.2].*
* Screen Design & Data Description:



* Activities Flow



* Sequense diagram



* Pseudo code

Require :

HR Manager has logined.

HR Manager views request detail screen.

------------------------------------------------------

**BEGIN MAINPROGRAM closeRequest()**

HR Manager click "Close" button;

show confim dialog;

**IF** (chose "Yes" button ) **THEN**

System updateRequest(RequestUpdate);

display list requests screen;

**END IF**

**END MAINPROGRAM closeRequest()**

----------------------------------------------------

1. Non-functional requirements

Usability

*[Include all of requirements that affect usability. For example:*

1. *Specify the required training time for a normal users and a power user to become productive at particular operations.*
2. *Specify measurable task times for typical tasks or base the new system’s usability requirements on other systems that the users know and like*
3. *Specify requirement to conform to common usability standards, such as IBM’s CUA standards Microsoft’s GUI standards]*

<Usability Requirement One>

*[The requirement description goes here.]*

Reliability

*[Requirements for reliability of the system should be specified here. Some suggestions follow:*

1. *Availability—specify the percentage of time available (xx.xx %), hours of use, maintenance access, degraded mode operations, etc.*
2. *Mean Time between Failures (MTBF) — this is usually specified in hours, but it could also be specified in terms of days, months or years.*
3. *Mean Time To Repair (MTTR)—how long is the system allowed to be out of operation after it has failed?*
4. *Accuracy — specify precision (resolution) and accuracy (by some known standard) that is required in the system’s output.*
5. *Maximum Bugs or Defect Rate—usually expressed in terms of bugs per thousand of lines of code (bugs/KLOC) or bugs per function-point (bugs/function-point).*
6. *Bugs or Defect Rate — categorized in terms of minor, significant, and critical bugs: the requirement(s) must define what is meant by a “critical” bug; for example, complete loss of data or a complete inability to use certain parts of the system’s functionality.]*

<Reliability Requirement One>

*[The requirement description goes here]*

Performance

*[The system’s performance characteristics should be outlined in this section. Include specific response times. Where applicable, reference related Use Cases by name.*

1. *Response time for a transaction (average, maximum).*
2. *Throughput, for example, transactions per second.*
3. *Capacity, for example, the number of customers or transactions the system can accommodate.*
4. *Degradation modes (what is the acceptable mode of operation when the system has been degraded in some manner).*
5. *Resource utilization, such as memory, disk, communications, etc.]*

<Performance Requirement One>

*[The requirement description goes here.]*

Supportability

*[Indicate any requirements that will enhance the supportability or maintainability of the system being built, including coding standards, naming conventions, class libraries, maintenance access, maintenance utilities.]*

<Supportability Requirement One>

*[The requirement description goes here.]*

Design Constraints

*[This subsection should:*

1. *Indicate any design constraints on the system being built.*
2. *Design constraints represent design decisions that have been mandated and must be adhered to.]*

<Design Constraint One>

*[The requirement description goes here.]*

On-line User Documentation and Help System Requirements

*[Describe the requirements, if any, for on-line user documentation, help systems, help about notices, etc.]*

Purchased Components

*[Describe any purchased components to be used with the system, any applicable licensing or usage restrictions, and any associated compatibility and interoperability or interface standards.]*

Interfaces

*[Define the interfaces that must be supported by the application. It should contain adequate specificity, protocols, ports and logical addresses, etc. so that the software can be developed and verified against the interface requirements.]*

User Interfaces

*[Describe the user interfaces that are to be implemented by the software.]*

Hardware Interfaces

*[Define any hardware interfaces that are to be supported by the software, including logical structure, physical addresses, expected behavior, etc.]*

Software Interfaces

*[Describe software interfaces to other components of the software system. These may be purchased components, components reused from another application or components being developed for subsystems outside of the scope of this* ***SRS*** *but with which this software application must interact.]*

Communications Interfaces

*[Describe any communications interfaces to other systems or devices such as local area networks, remote serial devices, etc.]*

Environment

Target Environment

*[Describe target environment or running environment where the application will be run after developing]*

Development Environment

*[Describe development environment where the application will be developed]*

Database

*[Describe database]*

Licensing Requirements

*[Define any licensing enforcement requirements or other usage restriction requirements that are to be exhibited by the software.]*

Legal, Copyright, and Other Notices

*[Describe any necessary legal disclaimers, warranties, copyright notices, patent notices, wordmark, trademark, or logo compliance issues for the software.]*

Applicable Standards

*[Describe by reference any applicable standard and the specific sections of any such standards, which apply to the system being described. For example, this could include legal, quality and regulatory standards, industry standards for usability, interoperability, internationalization, operating system compliance, etc.]*

1. Appendix

*[Make the* ***SRS*** *easier to use. These may include use-case storyboards or user-interface prototypes. When appendices are included, the* ***SRS*** *should explicitly state whether or not the appendices are to be considered part of the requirements.]*