#### UserManagement

***Class Methods***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *1* | *doGet()* | *Handles the HTTP <code>GET</code> method.* |

#### UserDAO

***Class Methods***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *1* | *getByAccess()* | *Method used to return the list of user accounts with lower level than admin account login .Inputs are admin level .Return a List of <Code>model.User</Code> represents the user’s data* |

#### DBContext

***Class Methods***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *1* | *getConnection* | *Method used to connect to the database. Return a <Code>java.sql.Connection</Code> represents a connection to the database.* |

Select \* from [User] where [is\_super] < ?

Diagram

Description automatically generatedDiagram

Description automatically generated

#### UserManagement

***Class Methods***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *1* | *doPost()* | *Handles the HTTP <code>POST</code> method.* |

#### LoadUser

***Class Methods***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *1* | *doPost()* | *Handles the HTTP <code>POST</code> method.* |

#### UserDAO

***Class Methods***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *1* | *getByAccess ()* | *Method used to return the list of user accounts with lower level than admin account login .Inputs are admin level .Return a List of <Code>model.User</Code> represents the user’s data* |
| *2* | *editRank()* | *Method used to increase or decrease the level of an user account .*  *Input is the id of the user.* |
| *3* | *disableUser()* | *Method used to set the account level lowest . Input is the id of the user.* |
| *4* | *searchByName()* | *Method used to search accounts have the detail has the similarities with text input . Input are the admin level , text need to search . Return a List of <Code>model.User</Code> represents the user’s data* |
| *5* | *sortUser()* | *Method used to sort accounts depend on title . Input are the admin level , title type . Return a List of <Code>model.User</Code> represents the user’s data* |

#### DBContext

***Class Methods***

|  |  |  |
| --- | --- | --- |
| **No** | **Method** | **Description** |
| *1* | *getConnection* | *Method used to connect to the database. Return a <Code>java.sql.Connection</Code> represents a connection to the database.* |

disableUser() :

update [User]

set [is\_super] = 0

where [id] = ?

editRank() :

update [User]

set [is\_super] = ?

where [id] = ?

searchByName():

select \* from [User]\

where [is\_super] <? and

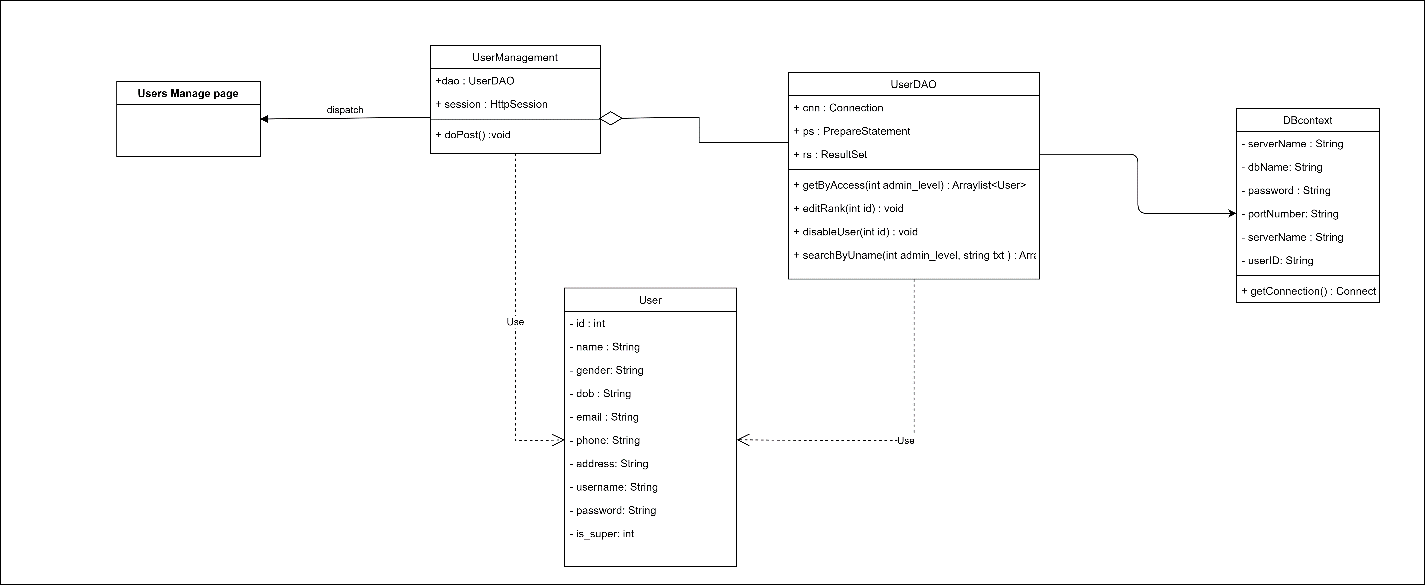
fullname] like ?

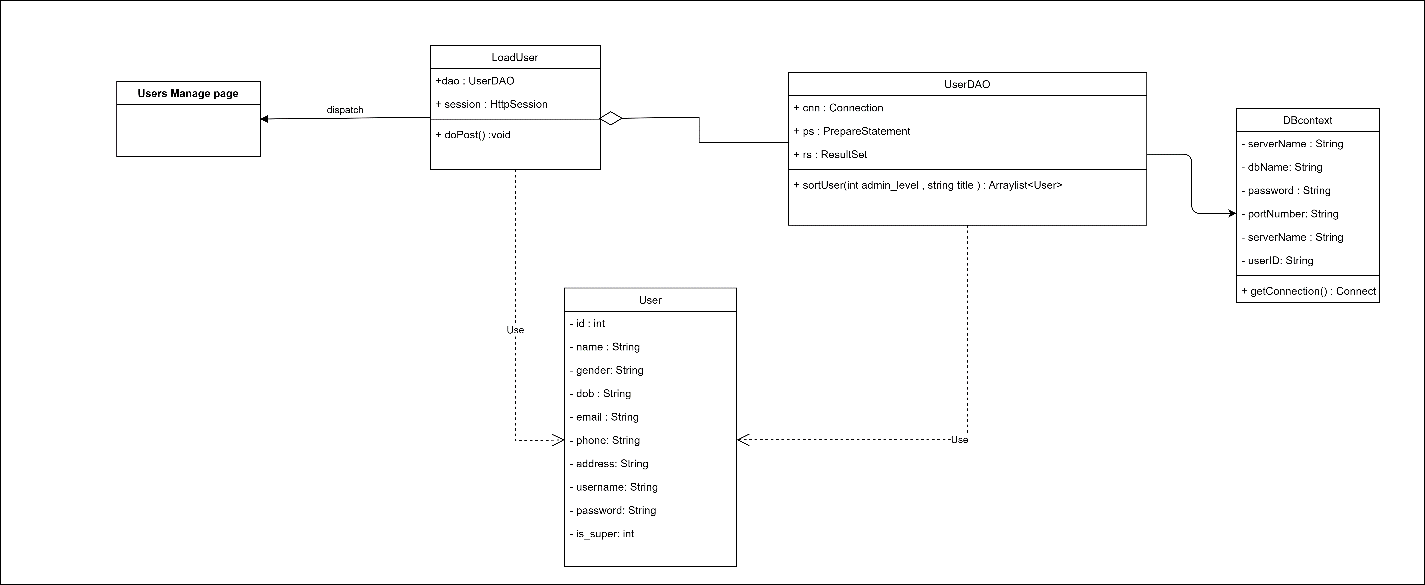
sortUser():

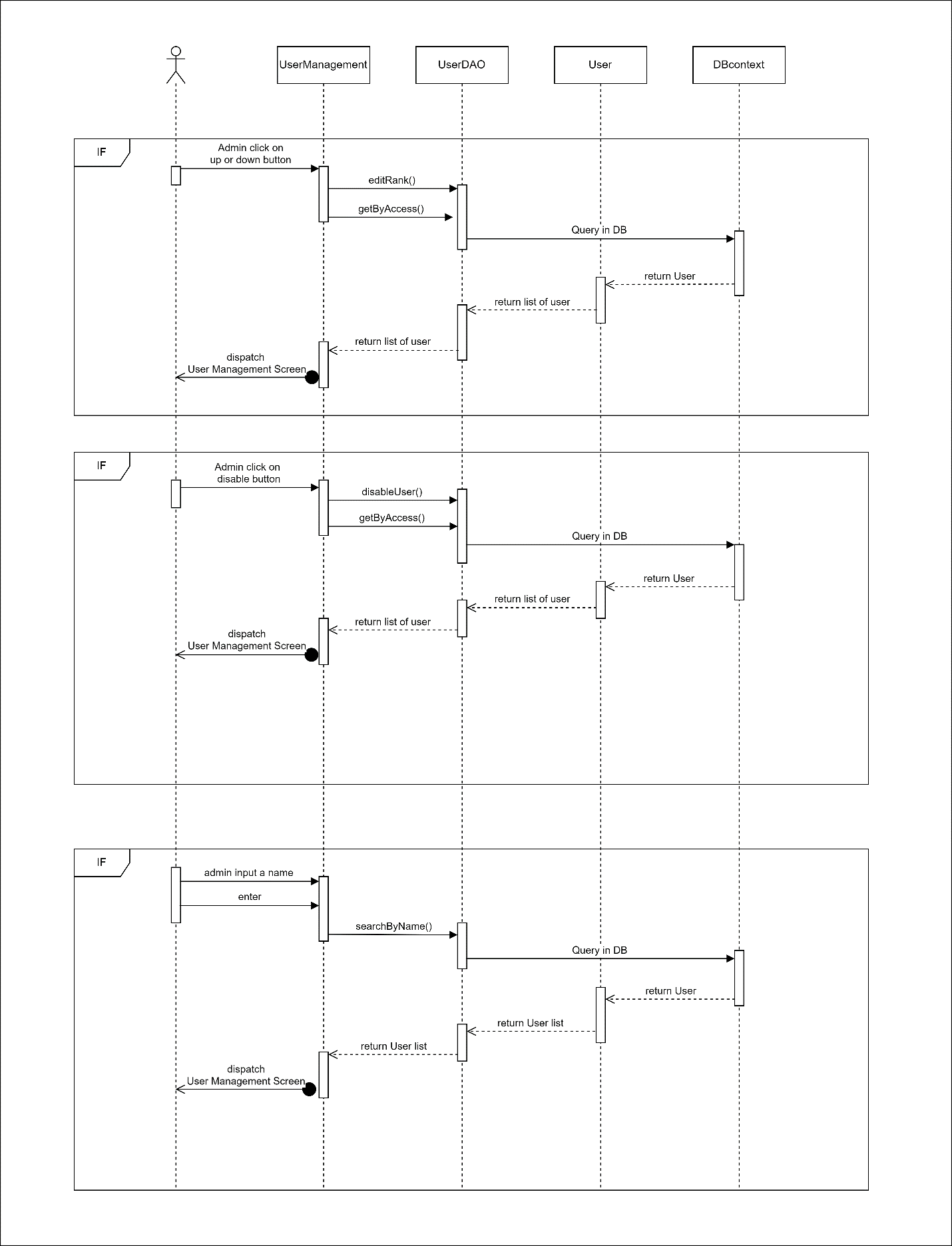
select \* from [User]

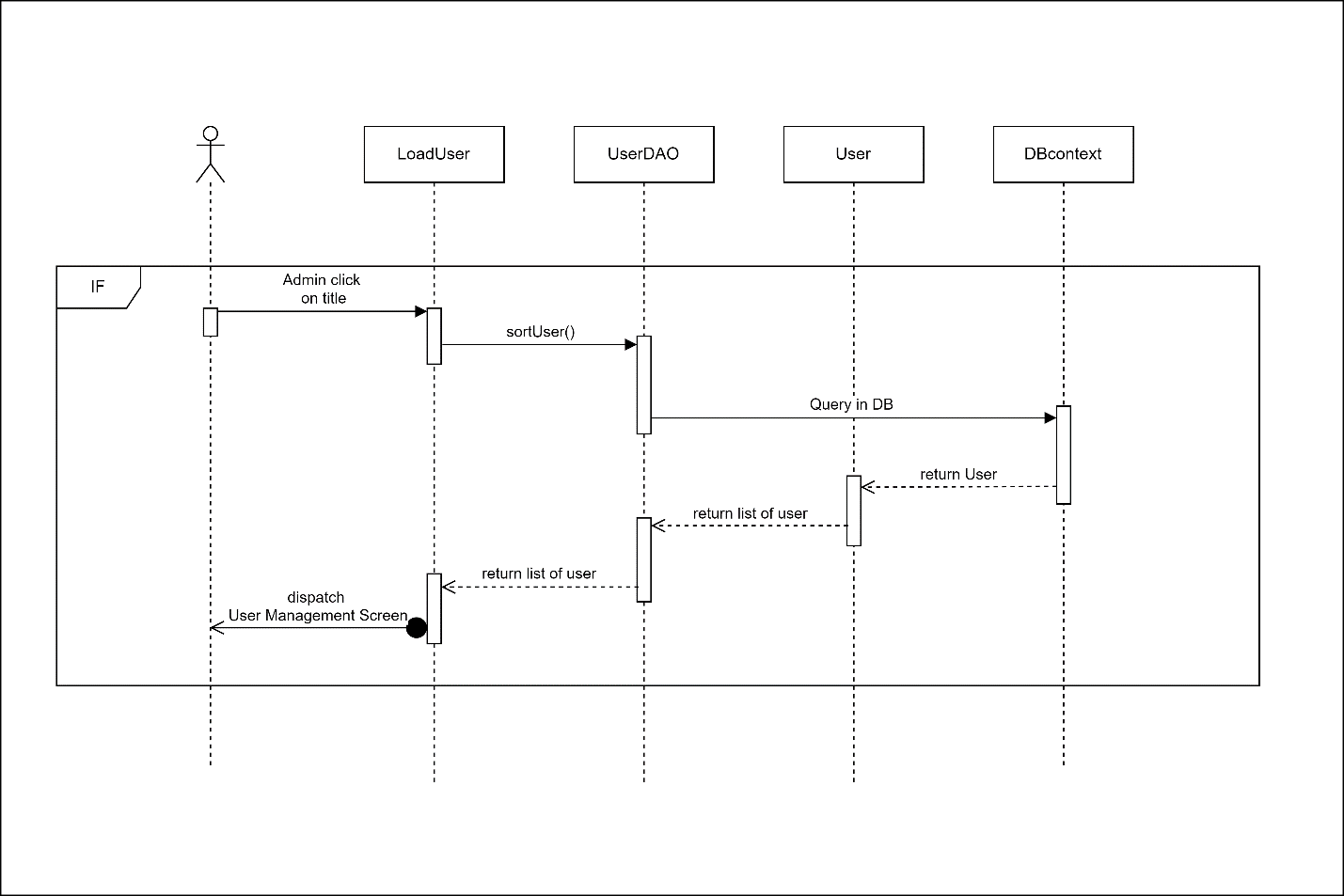
where is\_super < ?

order by ?







Access Level

● Function trigger: After login successfully

● Function description:

o Actor: Admin , User

o User and Admin depend on their level can view or redirect to another page.

● Function detail: After login success

o Normal user will be send back to the formal page

o With admin will send to the admin profile

o Admin also depend on the level which can see and cannot:

View List User

● Function trigger: Click Users Management

● Function description:

o Actor: Admin

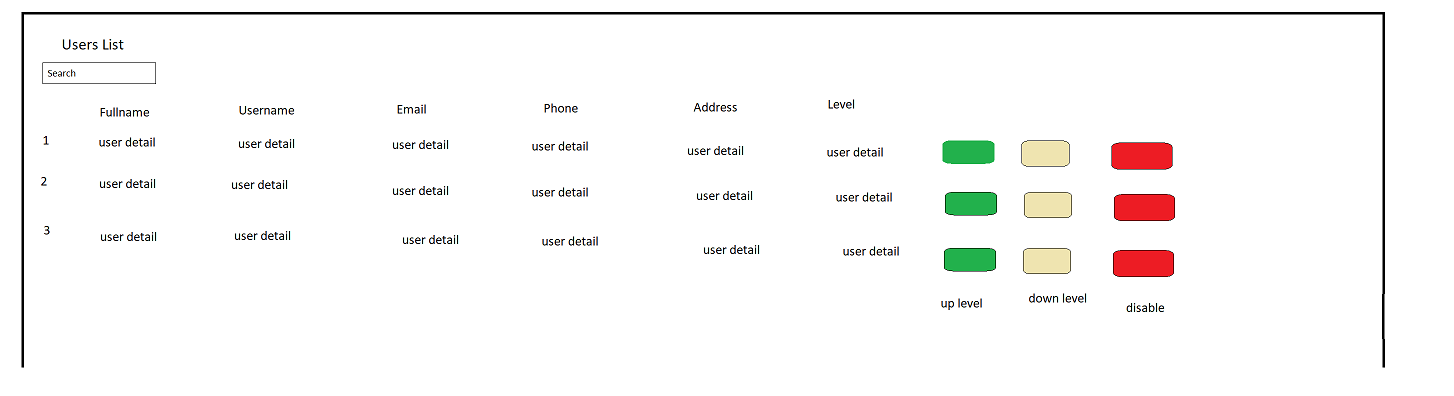
o Admin can view list of user

● Function detail:

o User list will be display on screen

o Depend on the admin ‘s level , it can see the lower user account

● Screen layout:



Users management

● Function trigger: From User management page

● Function description:

o Actor: Admin

o Admin can view the list of user’s accounts with details

o Admin can also adjust the account

● Function detail:

o Sort the list of users depend on title

o Adjust the level account

o Disable account

o Searching

● Screen layout:

