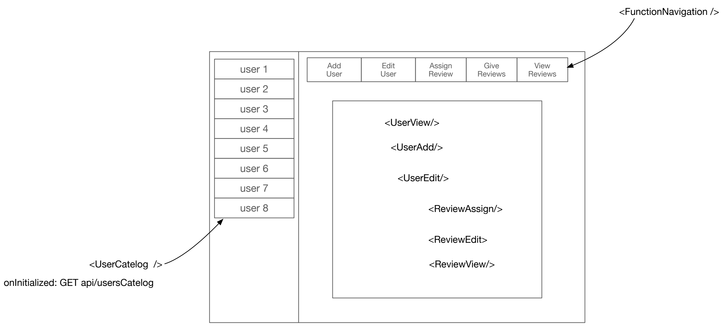
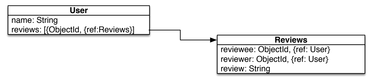
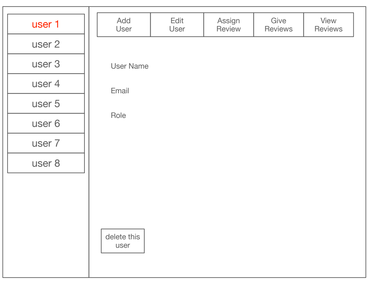


onInitialized: GET api/usersCatelog



users/id_user1



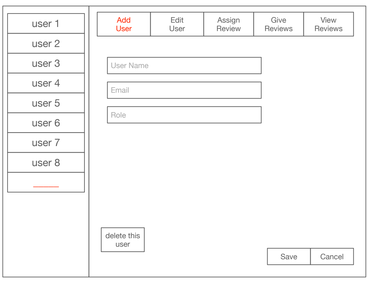
<UserView/>

onInitialized: GET api/id_user1

onDeleteBegin: DELETE api/users/id_user1

onDeleteSuccess: GET api/usersCatelog

users/new

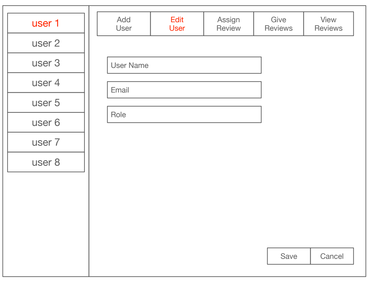


<UserAdd/>

onSaveBegin: POST api/users

onSaveSaveSuccess:
- navigate to view new user
- GET api/usersCatelog

users/id_user1/edit



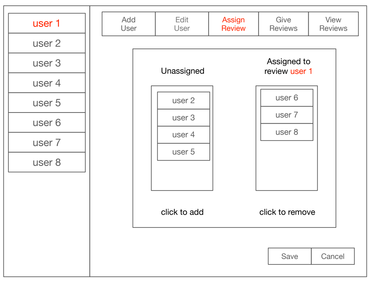
<UserEdit/>

onInitialized: GET api/users/id_user1

onSaveBegin: PUT api/users/id_user1

onSaveSuccess:
- update {user_1} on users resources
- navigate to view user_1

users/id_user1/assign



ids_all = users(GET, api/userCatelog).map(item => item._id)
ids_reviewer = Reviews.find({ reviewee:id_user1 }, 'reviewer')
ids_unassigned = _.different(ids_all ids_reviewr)
name = userCatelog(_different(ids_all ids_reviewr)

<ReviewAssign/>

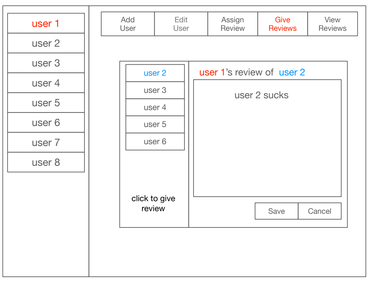
onInitialized:
- get all userId : **Ids_all**
- get all userId assigned to review user1 :**ids_reviewer**
- get unassigned users id : **ids_unassigned**
- look up name : **name**
- initialized **draft** {reviewer: [**ids_review**] , id:id_user1}

onUserAssigned(**id**):
- move **id** from **ids_unassigned** to **ids_reviewer**
- **draft**: add **id** to draft.reviewers

onUserUnassign(**id**) :
- move **id** from **ids_reviewer** to **ids_unassigned**
- **draft**: remove **id** from draft.reviewers
- get Review where id_reviewer === id

onSave:
- submit { draft.ids_reviewer, id_reviewee: draft.id_user1 }
- ***on API create new Review***:
  Iterate each ids_reviewer in draft.reviewers, find Reviews(reviewee: id_user1, reviewee: id_reviewer)
  - if does not exist create a new Review instance where:
    Review.reviewer = id_reviewer
    Review.reviewee = id_user1
    Review.review='';
- ***on API Delete Review:***
  Find all Reviews whose reviewee is id_user1, then iterate through this collection
  if Review.reviewer is not in ids_reviewer, delete this Review

users/id_user1/give-review/id_target2



<ReviewEdit>

onInitialized:
- get all **ids_Review** of Reviews where Reivew.reviewer is id_user1

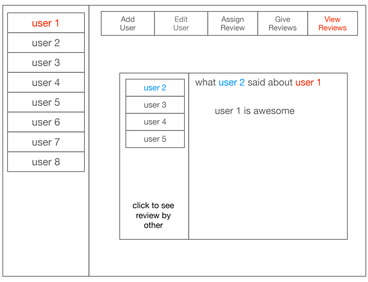
onUserSelection:
- look up Review(id_target${id}).review and assign to draft

onFieldChange:
- update draft.review's content

onSave
- submit {draft.review, draft.id_review }
- on API: Reviews.save(draft.review, draft.id_review)

onSaveSuccess:
- no task need to exec

users/id_user1/view-reviews/id_target2



<ReviewView/>

onInitialized:
- get all **ids_Review** of Reviews where Reivew.reviewee is id_target2

onUserSelection:
- look up Review(id_usert${id}).review and show in view