## **Git**

* use 'git init' to initialize project
* use 'git commit -m "message"' to commit changes
* use 'git push' to push latest changes

## **Data structure**

* user:

{

username: string,

password: string,

ofSuperAdminInRole: boolean,

ofGroupAdminInRole: boolean,

ofGroupAssistInRole: boolean,

groupList: [],

channelList: [],

adminGroupList: [],

}

* channel

{

channelname: string;

description: string;

}

* group

{

groupname: string;

description: string;

}

## **REST API**

* Overall:
  + Front End should handle how to display UI in browser
  + Back End should handle requests from front end and provide requested data. Express will communicate with mongoDB database to CRUD data.
  + All data should be JSON format
  + Back End receive the request of uploading image files and save it into a static dirctory folder 'assets' and return file's path to front end.
* User
  + POST '/login': for user lgoin
    - param: username, password
    - return: user data
  + GET '/users': get all users list
    - param: none
    - return: uers list data
  + GET '/user/:username': get a single user
    - param: username
    - return: user data
  + POST '/user': create a new user
    - param: user data
    - return: new user data
  + PUT '/user/:username': update an existing user
    - param: username and user data
    - return: user data
  + DELETE '/user/:username': delete an existing user
    - param: username
    - return: deleted user data
* Channel
  + GET '/channels': get all channels list
    - param: none
    - return: channel list data
  + GET '/channel/:channelname': get a single channel
    - param: channelname
    - return: channel data
  + POST '/channel': create a new channel
    - param: channel data
    - return: new channel data
  + PUT '/channel/:channelname': update an existing channel
    - param: channelname and channel data
    - return: channel data
  + DELETE '/channel/:channelname': delete an existing channel
    - param: channelname
    - return: deleted channel data
* Group
  + GET '/groups': get all groups list
    - param: none
    - return: group list data
  + GET '/group/:groupname': get a single group
    - param: groupname
    - return: group data
  + POST '/group': create a new group
    - param: group data
    - return: new group data
  + PUT '/group/:groupname': update an existing group
    - param: groupname and group data
    - return: group data
  + DELETE '/group/:groupname': delete an existing group
    - param: groupname
    - return: deleted group data
* Image
  + POST '/image-upload': save image to server side and return the image's path
  + param: file object
  + return: file's path

## **Angular Architecture**

* Component
  + HomeComponent
  + ChatComponent
  + ChannelListCompoent
  + GroupComponet
  + LoginComponent
  + UserComponent
* Services
  + authentication service
  + socket service
  + user service
  + channel service
  + group service
  + image service
* router
  + implement in app-routing.module.ts
  + when access to root '/' - redirect to '/login'
  + only route '/login' can access without authentication, other routes need to check permission
  + any other path will be redirect to root '/'