DASS Assignment 1

Concerned TAs - Kartik, Sanchit, Abdullah, Karmanjyot, Swetha, Harshita, Mihir

Post all doubts on Moodle

Hsnayirp being an admin of confessions page is now feeling sad due to decreasing number of posts he is getting now. But as they say there are 5 stages of grief, Hsnayirp experiences a change in his state of mind and is now in extreme anger. He is now planning a bombing similar to Hiroshima and Nagasaki on UG2. But as we all know we do not have any lab which makes Atom Bombs so he asked your DASS TAs for help. Basically he wants students to be active on some social media application so we came up with this assignment of building a social media platform. And here we are with the Bomb named, Greddiit.

This assignment is based on MERN Stack

1. **M**ongoDB for Database (Non Relational, Different from MySQl)
2. Backend Framework using **E**xpress JS implementing REST API
3. **R**eact JS for Frontend
4. **N**ode JS for Backend

Requirements

Your Web Portal has users being of two kinds, one being Moderators others being regular Users. A regular user becomes the moderator of a Sub Greddiit which he/she creates. Read further to get more clarity.

Details Of Users

1. First Name
2. Last Name

3. User Name
4. Email [Unique]
5. Age
6. Contact Number
7. Password

Details of Sub Greddiit

1. Name
2. Description
3. Tags (Single Word, Lower Case, Multiple)
4. Banned Keywords (Single Word, Case Insensitive, Multiple)

Details of Report

1. Reported By
2. Reported User
3. Concern (text field)
4. Post it is associated with

Details of Posts

1. Text
2. Posted By
3. Posted In
4. Upvotes
5. Downvotes

You are free to add more parameters in each of the above mentioned details of different entities. You can also create new entities based on how you approach use cases.

Login and Registration

1. A registration page for the users having appropriate fields as mentioned earlier [3 Marks]
2. Login portal redirecting to Home Page for the User [3 Marks]
3. A user once logged in, should not be asked to login again even on website/browser/computer restart [2 Marks]
4. Logout [1 Mark]
5. Authentication & Authorization (tip: read about the difference on google) [5 Marks]
    a. The password stored in the DB must be hashed (you can use *bcrypt.js* library)
    b. All other pages must be accessible only after login/registration. Any attempt to go directly to those pages without logging in first should redirect the user to Auth page.
    c. All the backend routes should be protected using token (you can use *jsonwebtoken library)*
6. Implement Login Using Google/Facebook API [**Bonus** 2 Marks]
7. Implement CAS Login (The College One) [Link] [**Bonus** 2 Marks]

Use Cases

1. Dashboard
    a. There must be a Navbar with links to all the pages mentioned below. Each link must either be an icon or icon+text (note: icon ≠ image, check mui icons) [2 marks]
    b. Profile Page
        i. On this page show the Basic Details of the User with an Option to Edit Them [4 Marks]

ii. Here, we are introducing Followers and Following, you need to display the corresponding numbers. These numbers would be **Clickable,** which when clicked will open a list of the User Names of the people in that list.

1. Followers, clickable ⇒ list. In this list for every User Name you need to add a **remove** button beside it which when clicked will remove that particular user from your followers [3 Marks]
2. Following, clickable ⇒ list. In this list for every User Name, you need to add an **unfollow** button beside it. On clicking this button, you should be removed from the followers list of the user in context. [3 Marks]

c. My Sub Greddiits Page

i. On this page, firstly there should be a Button to create a new Sub Greddiit. When this button is clicked a form opens having appropriate fields. [5 Marks]

1. A newly created Sub Greddiit has 1 follower by default, the creator itself.
2. Add a field for uploading an Image also [**Bonus** 2 Marks]

ii. On this page itself, you need to show the list of existing Sub Greddiits and for each one in the list show the following Details [4 Marks]

1. Number of people in the Sub Greddiit
2. Number of posts posted in the Sub Greddiit until now
3. Name
4. Description
5. Comma-separated list of banned keywords

Along with the basic Details, you need to add

1. Delete Button which when clicked results in deletion of that particular Sub Greddiit [2 Marks]
    a. Deletion of a Sub Greddiit must also delete all the related posts, reports, etc.

2. Open Button which when clicked Opens the Web Page for that Sub Greddiit [2 marks]

iii. Sub Greddiit Page

1. When it Opens, New links appear in the existing Navigation bar which are listed below [2 Marks] [Refer to **Important Note** mentioned near the end]
2. **Users**
    a. On clicking this link, a Page Opens which shows a list of the Users who have Joined the Sub Greddiit in Context. But firstly you need to show the users who haven't been blocked by the Moderator and after these show the ones who have been blocked. These 2 should be clearly distinguishable from each other [3 Marks]
3. **Joining Requests Page**
    a. This page shows the list of Joining Requests of all the Users who have requested to Join the Sub Greddiit in Context. For each request, show the user details and there should be two buttons where first one being the one for accepting the request and other one for rejecting the request. [3 Marks, (1 For list, 1 for each button)]
4. **Stats [4 Marks + 1 Mark bonus]**

    A moderator/logged in user should be able to view stats for every sub greddiit page that they created.

    i. Growth of the sub greddiit in terms of members over time
    ii. Number of daily posts vs date
    iii. Number of daily visitors vs date (visitors are counted by the number of people clicking the sub greddiit link)
    iv. Number of reported posts vs actually deleted posts based on reports

    You have two options to display these stats:

    1. Just display them in Tabular form [4 marks]

2. Display them in any kind of graphs [4 marks of the requirement + 1 mark bonus]

5. **Reported Page**
    a. On this page you are supposed to show the reports that have been made so far on the Sub Greddiit. Basically on the Sub Greddiit the users can Post Text which can be reported by some other user. You are expected to show the following information for each Report [3 Marks]
        i. Reported By
        ii. Whom we have reported
        iii. Concern
        iv. Text of the Post which is reported

    For each report there are 3 Actions

    1. Block User / Delete Post / Ignore
        a. Ignore will fade out other buttons [2 Marks]
        b. If blocked then post remains in the Sub Greddiit but the name of the person who posted is now replaced with the name of “Blocked User” [3 Marks]
            i. The moderator of Sub Grediit will still see the original name when viewing the list of Reports
            ii. Make sure that other users are not able to see the original name even on direct API call (handle things in server)
        c. When block button is pressed it changes to another button with a countdown like “Cancel in 3 secs” (where 3 will change to 2 after 1 second and so on). If the timer reaches 0, the user is blocked, otherwise the moderator can press cancel to abort. [2 marks]
        d. If post deleted then request is removed. [2 Marks]
        e. There is a timer here, lets say a report is not handled for 10 days then this request is removed without doing any action.

(Keep this 10 Days as a variable, we might change this in evaluations) [2 Marks]

i. The report might be still there in the DB even after 10 days, but the moment someone tries to fetch that, it should get deleted.

f. [**Bonus**]

i. For each of the actions, send an email notification to the Reporter about the action taken [1 Mark]

ii. When the reported user is blocked or the post is deleted then send an email notification about the same to the reported user too. [1 Mark]

d. Sub Greddiits Page, This is the page where all the Sub Greddiits created by any user (including yourself) on the platform are shown. The previous one, My Sub Greddiits was focused on the ones the logged in user has created for which he assumes the role of a moderator.

i. This page should have a

1. Search Bar, where one can search for a Sub Greddiit based upon its Name, all items should be listed which contains that term (case insensitive) [2 Marks]

a. Implement this as Fuzzy Search [**Bonus**][2 Marks]

2. Implement a filter based on the tags of the Sub Greddiit similar to what we have on e-commerce web sites [Multiple Tags can be selected, items having any of the the selected tags should come] [2 Marks]

3. Implement Sort based on ⇒

a. Name (Ascending/Descending), [1 Mark]

b. Followers (Descending), [1 Mark]

c. Creation Date (Newly created Sub Greddiits should appear on top) [1 Mark]

d. Multiple Sorting Criteria can be selected (Nested Sort). For example if we select both Name and Followers (Order of selection

matters) then the list is first sorted alphabetically then based on the number of followers. [**Bonus**][1 Mark]

ii. On this page you are expected to

1. First show Joined Sub Greddiits, where basic details of each Sub Greddiit are also shown (Details same as what were shown for Moderator [c.ii]), with each being a clickable item. [2 Marks]
    a. For these Joined Sub Greddiits their should be a Leave Button, clicking on which will immediately kick you out of it. Once you leave a Sub Greddiit you cannot send a join request to that Sub Greddiit Again. [2 Marks]
        i. If the current user is the moderator of that Sub Greddiits, the button should be disabled.
    b. On clicking a sub Greddiit, you are redirected to a page where on the left side you display the Image(Show a random image if not implementing the image Bonus), Name, Description associated with the clicked Sub Greddiit. [2 Marks]
        i. Even if the current user is the moderator of this Sub Greddiit, this page only will be shown.
    c. Then on this page there should be Create Post Button, clicking on which a dialog box (modal) appears with appropriate fields [2 Marks]
    d. On the same page, you show the posts that have been posted until now. For each Post
        i. Add Buttons for Upvoting and Downvoting [2 Marks]
        ii. Add Commenting feature for posts (Single Level text Only Comments) [2 Marks]
    e. IMPORTANT: All the Posts are text based only
    f. For every post, there should be a save button to save the post for later reference [2 Marks]

g. With every post, there should be a follow button, wherein a user can follow the user who posted that specific post. [2 Marks]

2. After Joined Sub Greddiits you need to show all the remaining sub Greddiits that are created with basic information (same as before), there should be a Join Button clicking which will send the associated moderator a Joining Request. [3 Marks]

    a. In case you have left a Sub Greddiit Before and you are trying to join it again then an alert should pop up displaying the relevant message [2 Marks]

e. Saved Posts Page

    i. Here you show all the Posts that have been saved by the logged in User across all the joined Sub Greddiits. Now for each post you need to display it the same way it was done in 1.d.iv.1.d along with the Sub Greddiit it belongs to [2 Marks]

    ii. Add a button here to remove the post from saved ones [2 Marks]

2. Banned Keywords [3 Marks]

    If the post contains some specific words (The Banned Words which were added when the Sub Greddiit was Being Created), then an alert should be popped to the user telling him that your post has certain banned keywords (These keywords are case insensitive, both “sky” and “SKy” should be flagged). Furthermore, these words are supposed to be replaced with asterisks when the post is shown anywhere on the portal (again, this should happen even on direct API call).

3. Miscellaneous details

    a. While editing/creating anything, input validation should happen, both in frontend and backend. On frontend, if input is invalid the button should also be disabled. [5 Marks]

    b. Wherever we say that the user should not be able to do this or that, this must be restricted from backend also. It’ll ensure that someone doing direct API calls without using your frontend is also not able to mess up things. [2 Marks]

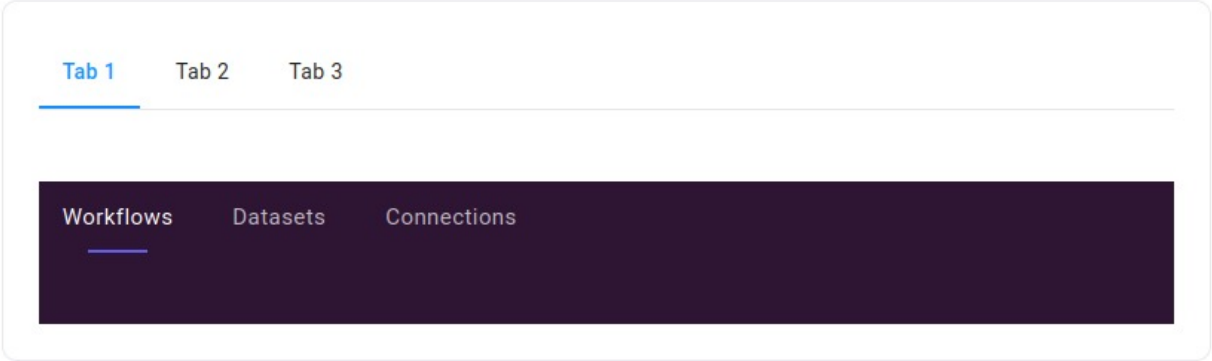
    c. At each place where the user is waiting for the API call to finish (like after clicking on create post button), the associated button should be disabled (or

better, turned into a loader). [1 Mark]

d. IMPORTANT: Wherever we say a different page should be created for this or that, there must be a corresponding URL change.
    i. The new page must not be dependent on the previous page, so even if the user copy paste the URL and open in a new tab, the same result should come.
e. There are no specific marks for UI, but try to make it good :)

Important Note

The new links appearing in the existing navbar is just one way to approach it. One another way could be to NOT have separate pages, instead just data separated in different tabs. As shown in the image below



Here Tab1 Tab2 Tab3 are the links in the main navigation bar (each linking to a new page, ie, URL). And Workflows, Datasets and Connections are simply the headings of Tabs that will come on Sub Greddit page.

An another way to implement this would be some sort of toggle button. You might have realised by now that there are roughly two roles one is moderator and the other one being a regular user. So you can design you frontend in such way that you can toggle between these roles (Similar to what some sites offer as dark/light mode). And based on what role you are on currently you should display the relevant things.

A general note on everything is, we want you people to stick to the original descriptions given but you are free to make some design choices here and there as long as you

have the intended functionality you are good to go!!

Bonuses

Bonuses have been mentioned inside the text only. There is one more bonus

- Chat Functionality [5 Marks]

  You would have noticed we have this followers/following thing. So in this bonus, you are being asked to build a Chat Box where users can chat with the person they follow. Both the users need to follow each other for chat to happen. [Hint: if you choose to explore this, there are 2 ways to do it- Sockets & ******* (won't spoon-feed you 😛), with the latter one being simpler so you may go ahead with that]

- Confirming to go back [1 Mark]

  While editing profile, if the user has changed anything and then presses back button (of the browser) then a confirmation box (UI is upto you, feel free to use normal `window.prompt()` as well) should come. Only if the user confirms, go back, otherwise cancel the back operation. You may assume that you have to do this for user following the normal flow (ie, if you have separate URL for edit, then user will never directly open that URL instead will always go by pressing the edit button on profile page).

- Infinite Loading [5 Marks]

  Instead of fetching all posts at once, server calls should be paginated (first 10, then next 10 and so on). From the UI side, user shouldn't manually change the page, instead as the user scrolls, you should detect it and fetch the required posts.

  Note: This is 5 Marks bonus because if you're doing pagination from server side, all the filters/sorting must also be done in the backend only.

- Multi-level Comments/Replies [3 Marks]

  Instead of having only one level of comments/replies for any post, you can have multiple (as much as the user wants, not hardcoded) level of replies (ie, replies to a reply, recursion).

  On the list of posts, when the user expands the replies, only the first level reply text is shown along with the number of level-2 replies each reply has. On expanding any level-1 reply, level-2 replies will come with text and level-3 replies will come and so

on. Make sure to fetch only the required data from server, nothing more than required.

- Keyboard Shortcuts [1 Mark]

  As explained above, My Sub Greddiits Page > Sub Greddiit Page is divided into 4 pages (or sections). There should be keyboard shortcut to go to each page. So, you can have on press “U” go to Users page, on “J” go to “Joining Requests Page” and so on (shortcut keys are upto you).

Deliverables

Submission must be in the following format

```
<roll_no>/
|--backend/
  |--package.json
  ... other files
|--frontend
  |--package.json
  ... other files
|--README.md
```

Backend and Frontend are directories with the Express.js and React app respectively.

Do NOT submit node_modules folder.

Zip this and submit it as a single <roll_no>.zip on moodle

> 10% Penalty for wrong format submissions and a straight 0 for submitting corrupt zips

Additional Instructions

- You are allowed to use any npm package unles it’s not off the shelf (Confirm from TAs, on moodle)
- You can use any styling or component library (bootstrap, tailwind, styled-components, Material UI, etc.)

- Final Submissions should have correct package.json files.
- DO NOT COPY, we would be checking plagiarism with everyone in your batch and also with your senior batches
- Do thorough testing, any errors during evaluations will be penalized
- Take care of edge cases, think about it
- An advice, Start Early!!

Deadlines (Courtesy of Hsnayirp)

The assignment is broken into 3 parts

- Part 1, You need to submit it 25th January 11:59 PM. For part 1 the only requirements for you are to build the login/registration page along with the profile page of the user.
- Part 2, Submission Deadline for Part 2 is 8th February 11:59 PM. For this part we expect you to extend the web app and build all the remaining features as mentioned above.
- Part 3, This part majorly has dockerization. Here you will dockerize your webapp and this would be the final submission. Deadline is 13th February 11:59 PM.

Hsnayirp requested us to be a little harsh with the deadlines but we then decided to be strictly harsh with them.

- In case you want to attempt the Bonuses and you are already past the deadlines, you can submit the Bonuses by 15th February 11:59 PM.