At the beginning homepage/ homepagefail is displayed. If account was not there on the site, then user may Sign-up. There are two pages signup and signupfail regarding this purpose. [fail may occur if duplicate username is found so user need to change the username]. Signup will basically take to app.post(“registertemp”), where all the registration logic is done and user is finally saved to the database. Now user may proceed to login again.

Then if login success then app.get(“/profile”) is requested and all the blogs are extracted from the server and then profilenew.ejs is rendered. **In the profile section “profilenew.ejs” also has all the images with it, so by comparing the all the images username and post[blog] creator username, I figured out the right image link to be displayed**.

profilenew.ejs[main page] is the home page where basically we can post the blogs…and see other blogs. If we post a blog at this page, it would Redirects to app.post("newposttemp"), where data extraction would take place and finally an User would be created and saved to the database and finally redirect to “/profile” [GET], basically the same page only, but with updated content.

profilenew.ejs[main page] also has a search box to search for people. This form will generate a GET request to /search route.

Now the logic in the arpp.get(“/search”) [GET] in the app.js is to search the users, based on the query received.

Here, two cases are there...if the query is "all", then all of the users are displayed as results also the image of the logged user is also passed as parameter for the ejs page to be rendered in order to display the profile picture of the logged user too.

Also the results, all the required userswould be passed as argument to the rendered page- "searchresultsnew.ejs".The other case is is when query is not "all", then sliding window algorithm would be used in order to seach for eg, "Computer" -> search is "omp" -> first "com", "omp","mpu","ute"...and filtered user  would be retrieved. Done with the search logic.

**[IMP]**

Each of the post displayed has a like and comment option. Now for the like option – app.get(“/like/:blogid”), the logic is that it would find the blog and retrieve the old likes and push the current user logged user’s username (req.user.username) and update the newlikes to the DB. Searching is done using the blogid param received. Similarly “unlike/:blogid” is also there to unlike a liked image. In the profile section “profilenew.ejs” also has all the images with it, so by comparing the all the images username and post[blog] creator username, I figured out the right image link to be displayed. Now, there is a form “add comment” which redirects to app.post(“/addComment/:blogid”), it would create a new Comment [blogid, comment, commentedBy] and Comment.save() to save the comment. Like the images, all comments also passed and finally the comments which has blogid same as of the displayed blog would be collected and displayed along with the user who has commented [with their images (as we have all the images with us) ], also regarding the deletion of comment or blog, if blog[createdBy] or comment[commentedBy], if same as logged user, then I have provided the delete option too.

Now exploring the Navbar-

Home button just takes to the front page [blogs + post blog] page

Person icon – suggest some recent users whom who have not sent follow requests yet.

Sending follow requests – app.get(“/send/:userid) [GET] (update rec\_req for the user whom we are sending and for us also, so that we may not be able to send once again)

Unsending follow request- app.get(“cancel/:userid) [GET] (Same procedure as above)

Heart icon- To retrieve the follow requests loggedin user are having. [May accept/delete] (app.get(“/requests”))

Accepting follow requests- app.get(“accept/:userid”)

Deleting follow requests- app.get(“delete/:userid”)

Now the person profile picture- app.get(“viewprofile/:userid”) – It basically see if the profile you are viewing is you or your followers then display all the blogs/posts, otherwise an image of private account would appear and follow request must be sent in order to view posts.

Also options of followers/ following also avl, and only the user himself/herself can see the followers or followings.

To unfollow-