**Features (Reels app)**

In this section we will see the features available on the reels app and how they work. The **technical** aspects (tech stack specific details) are discussed in the **next section**.

1. **Authentication:**

Authentication provided is email and password based. Once a user is logged (through signup or login) in, then the authentication details are saved in **cookies**. These cookies are then used for **persisting** the logged in state of the user in case the page is manually refreshed or the web app is opened in a new tab. Once the user logouts of the web app, these cookies are reset.

NOTE: There is a **Guest login** functionalityusing which anyone can login and explore the web app **without providing authentication details**. However, the guest users cannot like or comment on any post. The guest users are also **prohibited** from uploading content on the web app.

1. **Post:**

The posts created by all the users show up on the feeds page. There are two types of posts: **Image posts** and **video posts**. Each post can optionally contain a **caption**. The post owner as well as **time of the post** is displayed on the top part of each post. On the bottom, we have three sections:

* 1. **Likes:** The heart button can be used to like the post. Total number of likes are also displayed. The “like” (as well as “unlike”) actions **persist in the database** (for non-guest users).
  2. **Comments:** A list of user comments with their respective profile pictures and usernames.
  3. **Posting comment:** A logged in user can post a comment if he is not a guest user. The comment is immediately added to the list of comments on the post, as well as in the database.

1. **Auto play:**

Whenever the page loads, if a video post is visible (70%) on the screen, then it is played automatically. All the videos are muted by default. When a video has completed its playback, then page automatically scrolls to the next post. If the next post also contains video, then the cycle continues. Image posts however stay on the screen until the user manually scrolls the page.

1. **Creating a post**

A non-guest logged in user can create a new post by going to /upload page. Depending on the media uploaded, image or video post will be created. The **caption** field is optional. The media (image or video) is uploaded to the database and subsequently a post is created for the same. The user is then forwarded to home page where he/she will be able to see his/her post along with other posts.

1. **Private route**

Not all routes are available at a particular instance. Depending on the state of authentication, certain pages won’t be available. If users are not authenticated, then only Login and sign-up pages are available. For logged in users, all the pages available except login and sign-up. The routes like /upload or /profile or home page are only available for logged in users. These routes are called private routes.

1. **Custom Alerts**

To show certain information to users, custom alters are used that are friendly and non-blocking. Such alerts are show on top of the header and **automatically disappear** after some time (usually 5 seconds). The alters also have a close button to aid the user experience further (as opposed to the default alerts that block the screen).

**Technical details**

This section describes which components of the tech stack were used to implement the above features.

1. **Authentication:**

Firebase auth is used for creating as well as logging in users. The persistent login feature is achieved using “onAuthStateChanged” property of firebase auth object. It automatically recognizes (using cookies) whether a user was logged in before and did not log out.

1. **Post:**

The posts are loaded when a page loads or a new post is uploaded by any user. The useEffect hook is used to attach an “onSnapshot” listener. The dependency array of useEffect is kept as an empty array as we only need to attach the listener **once**. The callback of this listener updates the local list of posts whenever there is any update in the database.

The user interactions like commenting and liking/disliking are saved in the database as a part of post’s data.

1. **Auto play:**

IntersectionObserver is used to achieve auto play feature. All the videos on feeds page are given to the intersection observer object for observing them. The conditional object defines threshold as 0.6. This means, whenever 60% of the video is on the screen, the callback will be called. This callback pauses all the videos except those, who are at least 60% visible on the screen.

1. **Creating a post:**

A separate collection is maintained in the database for post’s data. Whenever a user creates a post, the data related to post (like media link, caption, creator’s unique id, type of post: video/ image) is saved to the posts collection. The unique post id is then saved to the user’s document in database.

1. **Private route:**

For implementing private routes, we use a separate component which receives a component as a prop and then conditionally renders it. The condition checked is whether user is logged in or not. The authentication context has the user authentication related information. We use useContext hook to consume this data and check whether any user exists in the context. If so, the private route component loads the component received as a prop. Otherwise, it redirects to login page.

1. **Custom alerts:**

Custom alters are implemented using material-ui **snackbars**. It is rendered in the main app component and can be triggered using a **notification object state in the context**. Any component that wishes to show the snackbar just sets this notification object (using the consumed setNotificationObj function from context) with new data. As a result, snackbar is triggered and displayed.