Twitter Clone

The main purpose of the app is to give users a convenient way to connect and share with others.

Technologies

**Front end** – Flutter

**Backend** – Nodejs and Express

**DB** – Mongo DB



The user can see other people posts and interact with them by liking and commenting.

Also, the user can upload a post by himself.

***Main Pages***

**\*All the mockups shown are not the finale design**

Home Page

Presents the latest Posts uploaded by the users friends that he follows.   
At the top presented a header with the twitter logo. Then all the posts will be shown in a scrollable component, so that the more the user scroll, the older the posts the user see.

**Data to fetch** - List of posts according to the users that the logged user is following, sorted by date and time.



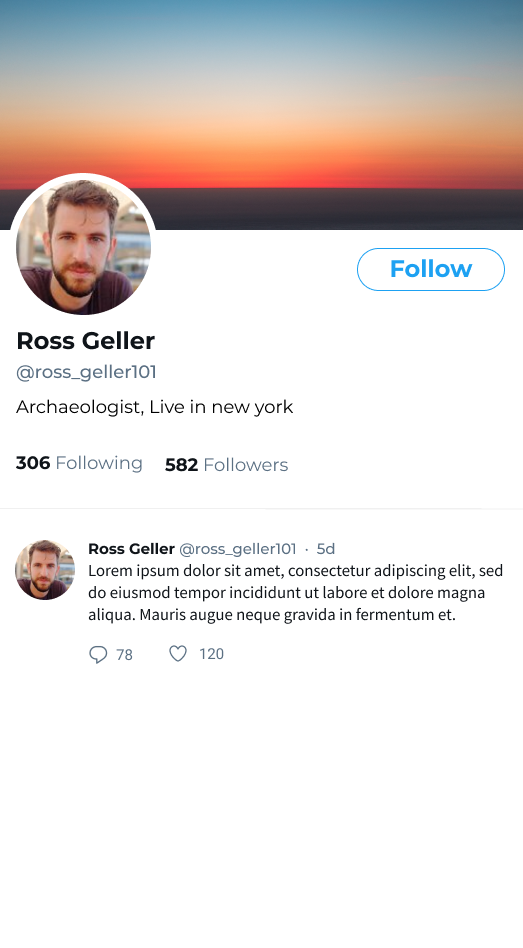
User Page

Presents the user details including he's profile-picture, header, user-name, name and biography (optional).

A button to follow/unfollow the user (wont show if the user is in he's own profile page).

At the rest of the page there is all of the user's posts in a chronological order.

**Data to fetch** – the user details along with all his post sorted by date and time.



*Main Components*

*Post component*

Displays a single post content of a user. Contains an image, video, or just text. The user user-name will appear at the top of the post along with he's user-image. At the bottom of post, there is an action bar that lets the users choose to like and comment the post.

***Main Objects - FrontEnd***

User:

* UserID (Integer)
* Name (String)
* Handle (String)
* Email (String)
* Bio (String)
* Photo (String)
* HeaderPhoto (String)
* Followers (List<Integer> user id's)
* Posts (List<Integer> post id's)
* Following (List<Integer> user id's)
* JoinDate ( String )
* FollowersCount (Integer)
* FollowingCounbt (Integer)

Post:

* PostID (Integer)
* UserID (Integer)
* Content
* Likes (List<Integer> user id's)
* Comments (List<Comments>)
* UploadTime (String)

Comment:

* CommentID (Integer)
* UserID (Integer)
* Body (String)
* UploadTime

**GraphQL - Queries**

The way the frontend is going to fetch data is by GraphQL queries.

Relation between data object example:

