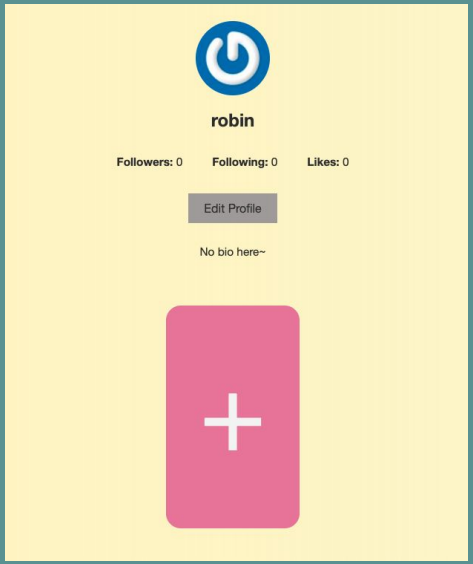
EdTok

Team 5: Jenna Blank, Jingnu An, Wei Liu, Guobin Chen

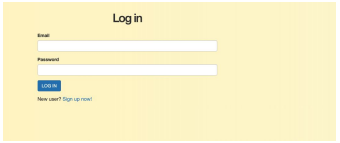
Overview



The EdTok aims to spread more knowledge in the form of short videos, helping users to utilize their leisure time to gain some interesting things through enjoyable ways.

Users can follow any other users on the platform, find who has followed them and who they are following. Users share their work or expertise by uploading videos to the app and others can find it, respond (similar to a like button) to it if they like it, or make a comment on it.

Functionality



Basic Functionality :

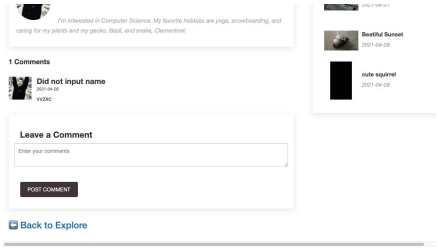
Login / Sign Up（User provide email address, name, password）

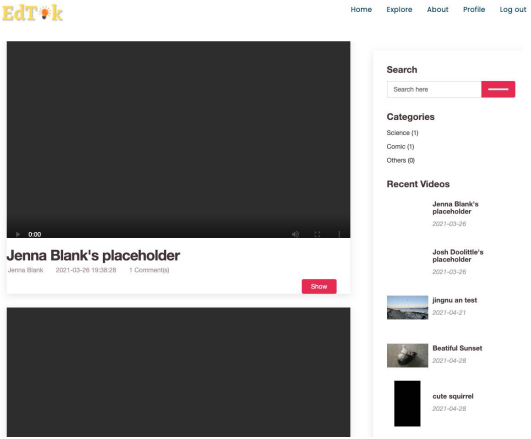
User profile (upload video, see followers && following)

Explore page (display all the videos)

-- sub-type(science, comic, others)

Comment page





Technical Architecture (Jenna)

Name: User
Columns: integer id, string name, string username, string email, string bio

Name: Video
Columns: integer id, string name

Name: Subject
Columns: integer id, string name, integer grade_level

Name: Comment
Columns: integer id, integer author, string text

Name: Like
Columns: integer id, integer liked_by

Name: Follower
Columns: integer id, integer user_id, integer follower_id

Name: Following
Columns: integer id, integer user_id, integer following_id

* Association table
Name: Follow
Columns: integer id, integer user_id, integer following_id
Note: user_id refers to the id of the user that has made the request to follow someone following_id refers to the user who the request has been made to

*Association table between User and Video -- one to many
Name: Upload
Columns: integer id, integer user_id, integer video_id

*Association table between Video and Subject -- many to many
Name: Tag
Columns: integer id, integer video_id, integer subject_id

*Association table between Video and Like -- one to many
Name: Reaction
Columns: integer id, integer video_id, integer like_id

*Association table between Video and Comment -- one to many
Name: Response
Columns: integer id, integer video_id, integer comment_id

Local storage (Jingnu)

First attempt: ffmpeg, ffmpegthumbnails, and carrierwave gem

Achieved: successfully extracted thumbnails from a video upload and stored the video file along with the thumbnails under local test.

Problem: Heroku don’t allow to store big file

Solution: link our project to an AWS S3 bucket and store videos on cloud.

Cloud storage (Wei)

First attempt: figaro: store configuration information

fog-aws: Library to support Amazon cloud service

Achieved: successfully store video to Amazon basket

Problem: not working when pushed to heroku

Solution: Need to use heroku “Add one” to add carrierwave and ffmpeg

Technical Architecture (continued)

- What parts of it were especially challenging

  Getting videos uploaded to work on local, switching carrierwave to AWS S3 for storage, config heroku to work with ffmpeg and ffmpegthumbnails

Future work (functionality) (Guobin)

Adding a process bar / loading circle on the video upload page,indicating the app is processing the video upload

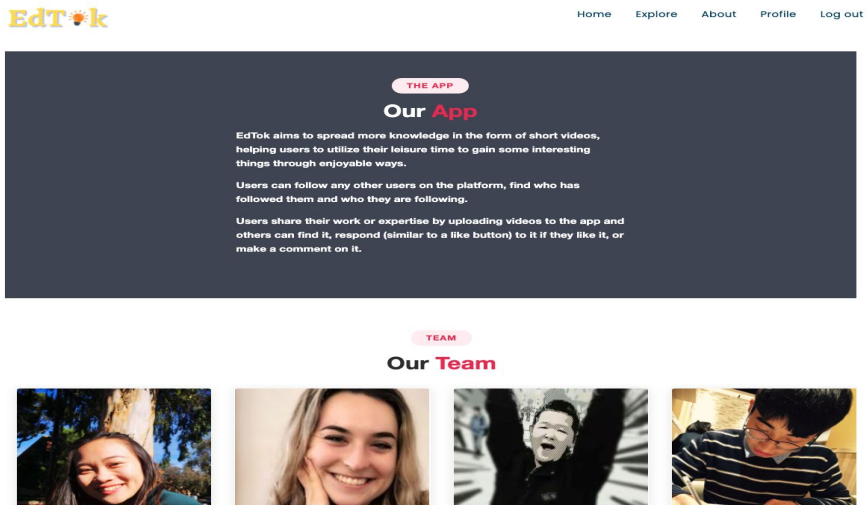
Make it mobile responsive

Invoke camera for taking video and upload the video immediately after.

Future work (UI)

Perfect the about us page(add some icons and

Add more categories in the explore page...



Question?