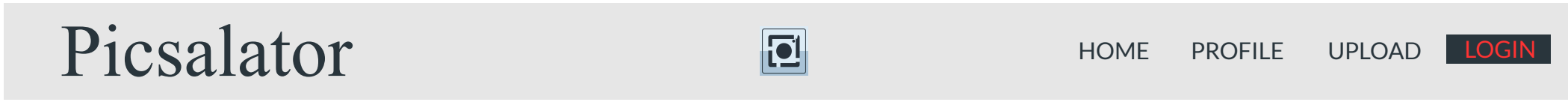
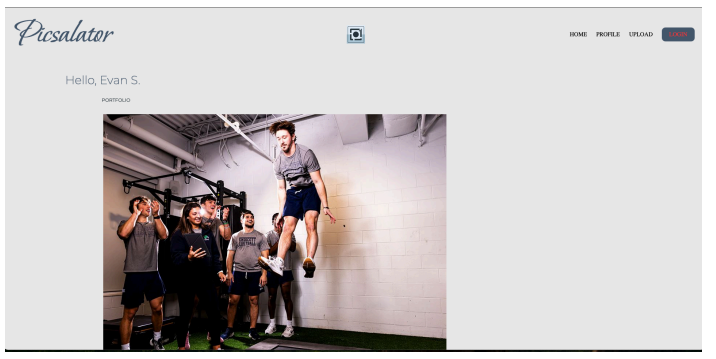


Frontend:

UI additions:
- Settings page or panel that allows users to logout, check sign in, compression settings, and color theme.
- Sharing capabilities for any photo the user has access to, they can email it or share it with other users in local storage.
- Drop down on the main home page that allows users to switch views between their photos, shared photos, or a combination of both.
- Clickable images, allow it be full screen full quality.
- Any photo is downloadable.
- Delete images, comment under images.

Backend:

- Data Storage: the current sqlite3 storage for larger image files is limited, so I will figure out a better way to do that. I may have to use blobs database.
- User Authentication: manage login credentials and user sessions.
- API endpoints: for clients to interact with the server via:
    - User registration / login
    - Photo uploads / downloads / sharing / deletion / comments
    - Retrieve a user's photo feed
    - Following other users?
    - image compression
- Security
    - Password hashing / salting
    - Secure session management



Profile Page

Frontend: user will see their name, their uploaded content, content shared with them, and compressed files (specific for IG posts).

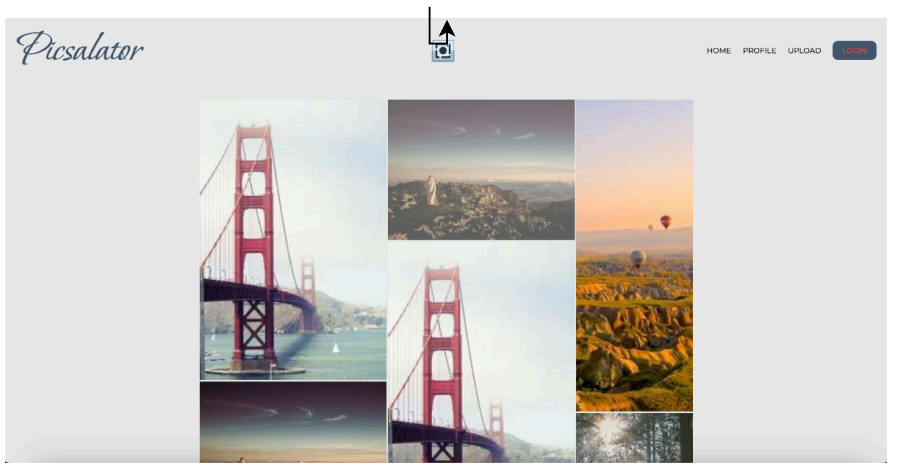
Backend: profile.php that communicates with picsalator.db > images table

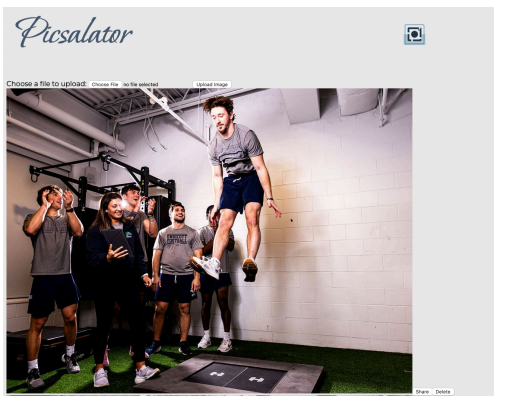
User Sign up / Login

Frontend: users decides to login or sign up, and then follows the prompts to do which ever they go with. They are able to login once they sign up.

Backend: saves username, password in sqlite with the images, and will use authentication and hashing.

Home Page:





Upload Page

Frontend: User will be able to press upload button, select an image from their hard drive, and it will show up on the page.

Backend: upload.php, and communicates with the computer's hard drive and the picsalator.db sqlite3 database and uploads.db