bulletin’

Product Instructions

Team 19

Introduction

bulletin’ is a simple platform designed to help students spread public campus related information. On the web, students post and view flyers to advertise different events. For students looking for clubs and events to participate in, it is unlikely that they will look at all flyers on a physical bulletin board and be informed of all current extracurriculars. Even further, it's difficult for clubs and events to be able to spread their flyers to all physical bulletin boards across the whole campus.

To solve this problem, we will create a web application that contains all events within the university, their information, dates, and image of the poster. Users can see the events on a daily, weekly, and monthly time scale, able to click on any one to learn further information, and are able to “collect” flyers to refer to later.

Platform

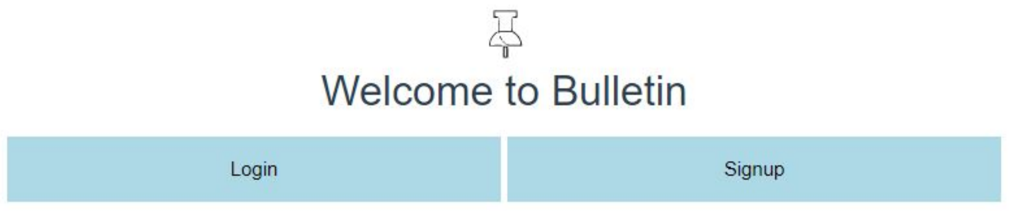
Designed as a web application, bulletin’ only requires a modern webbrowser, a nodeJS enabled backend server, and a nodeJS enable frontend server. Both servers can run on the same machine.

Program Structure

The standard process to reach all functionality of our platform is simple due to the fact that there are only a few pages that the user will be using.

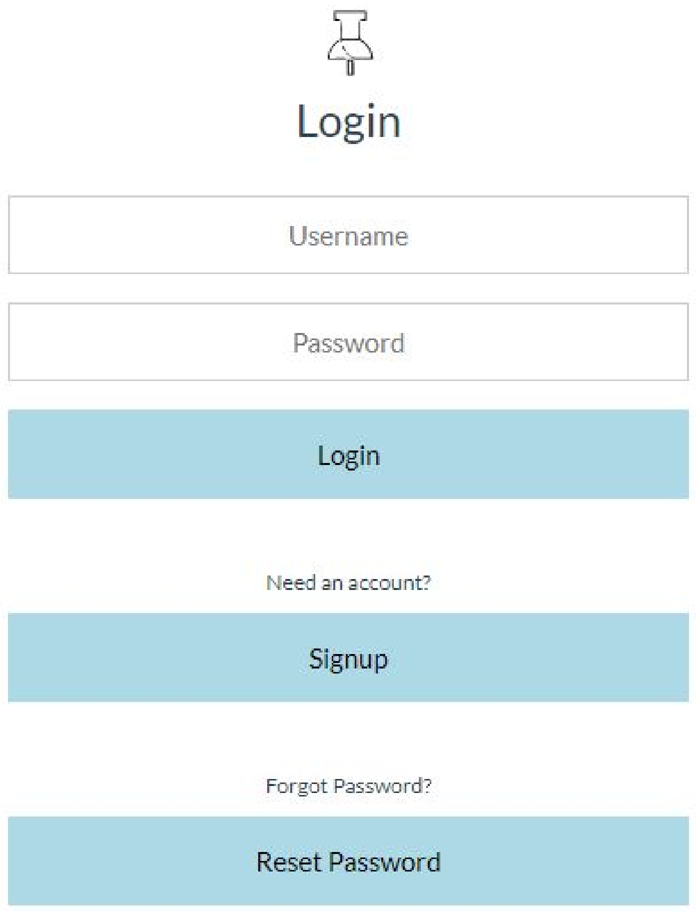
Welcome Page:

Upon first arrival, the user is given the Welcome Page. This screen simply redirects the user to either the login or signup pages.



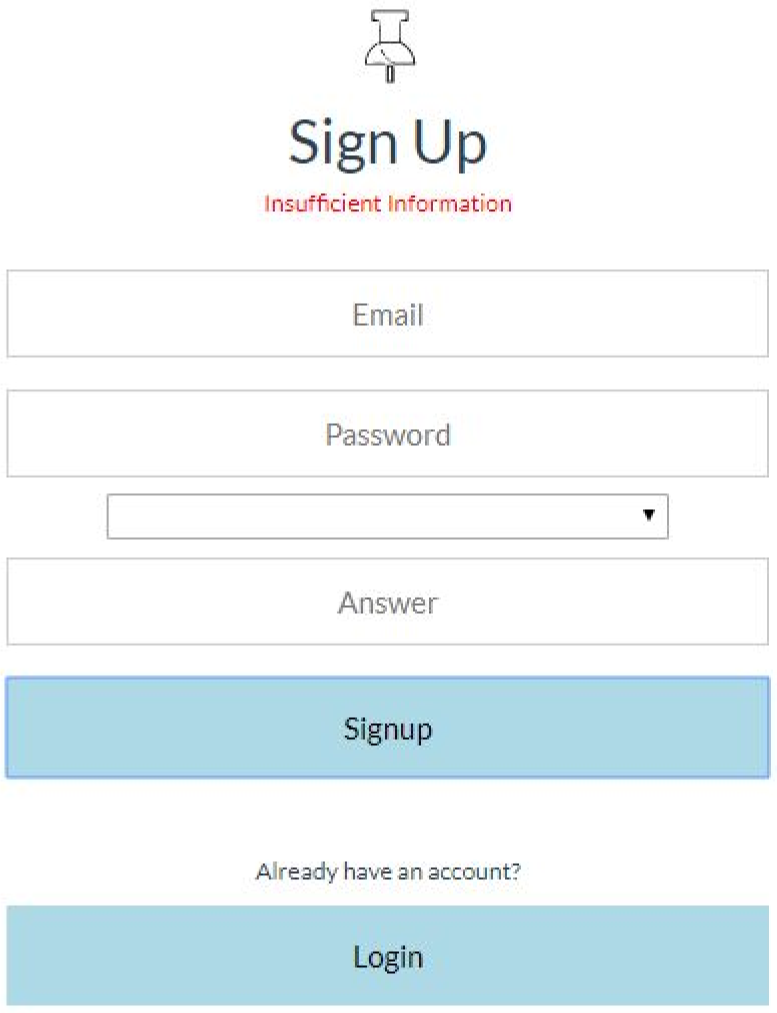
Login Page:

A very standard login screen that gives you ability to login with a valid account, recover a forgotten password, or redirect to the signup page.



Signup Page:

The signup page is used to create an account for future login. The signup page demands all inputs be filled in, requires @purdue.edu in the email, and a password of at least 8 characters in length. Upon failure, the user is notified of any shortcomings. Upon success, the user is redirected to the Homepage.

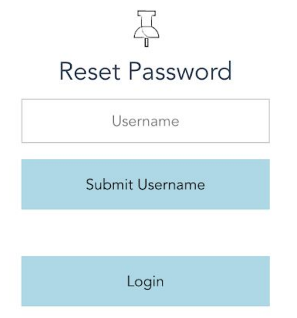


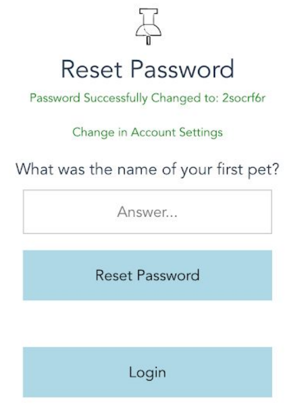
Forgot/Reset Page:

In this page, a user is able to receive a new, temporary password in case they forget their own. It requires 2 steps.

- First a user must type their username correctly.
- After, give the correct answer to the security question chosen by the user upon Sign Up.

After successful attempts, the user will then receive a new 8 character long password that they can use to login to their accounts and change.





Password Successfully Changed to: 2socrf6r

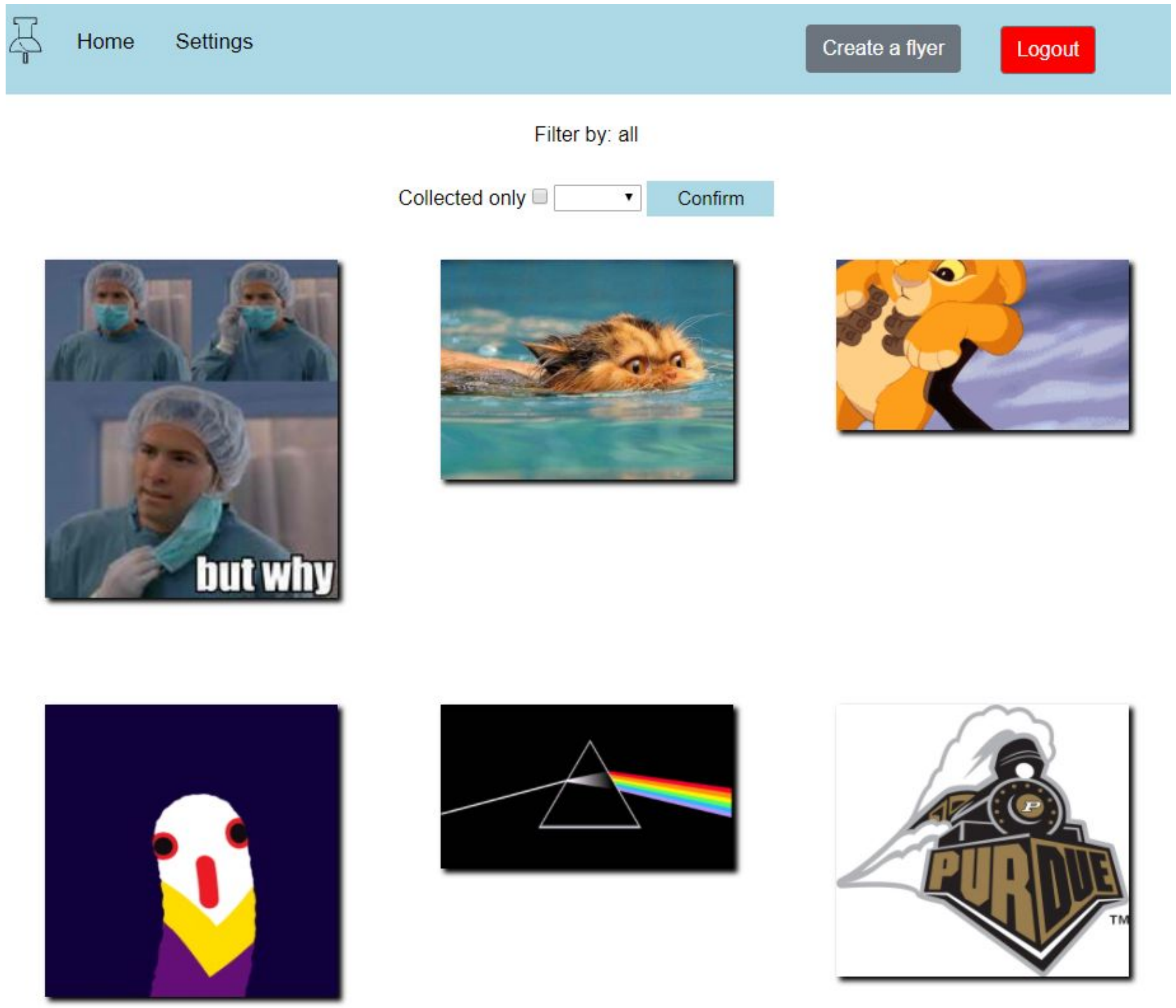
Change in Account Settings

What was the name of your first pet?

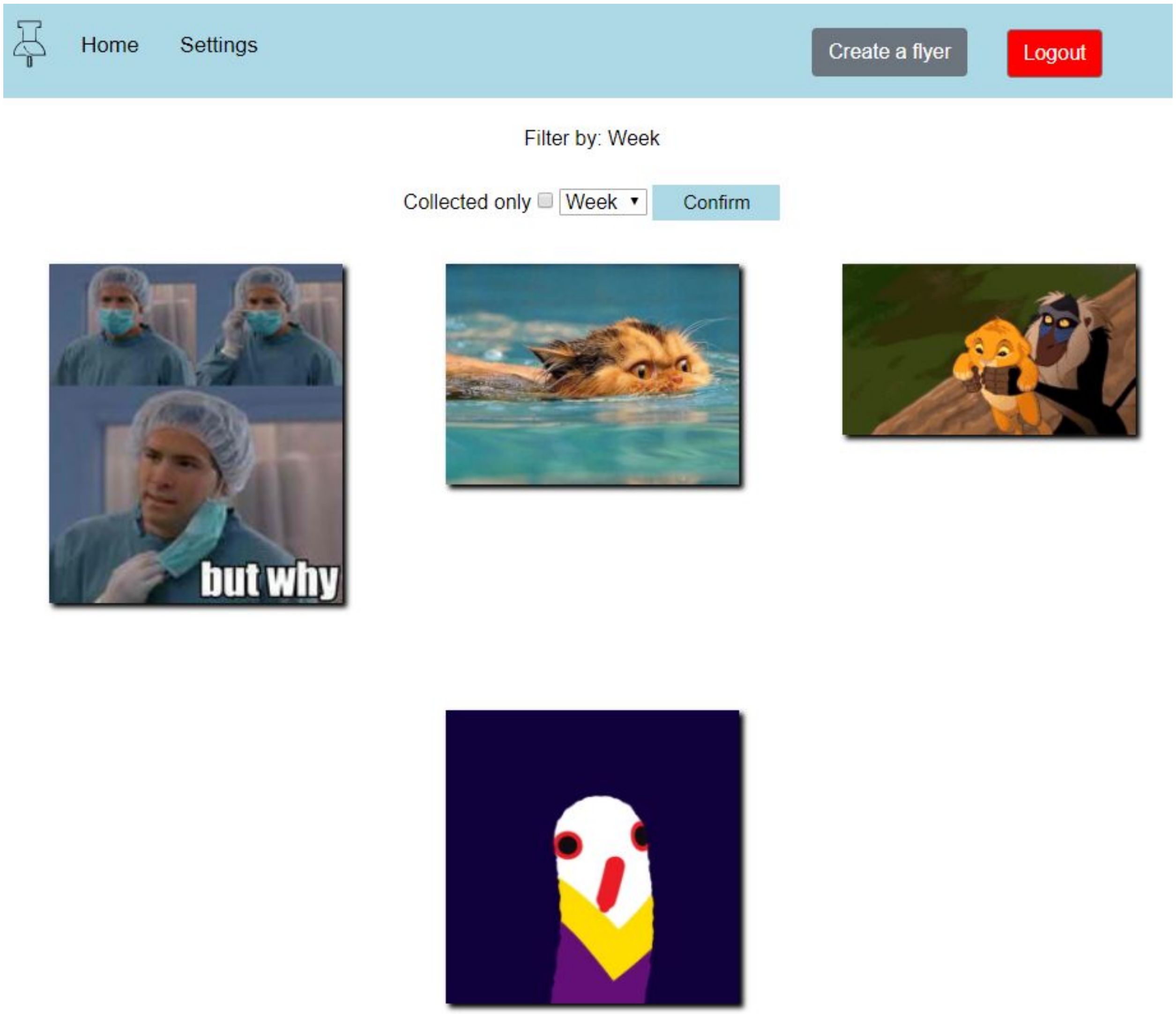
Homepage:

This page is the bulk of the application. From here, you can:

- View flyers (default view)
    - This will be the default view when first logging in or when selecting the “All” filter



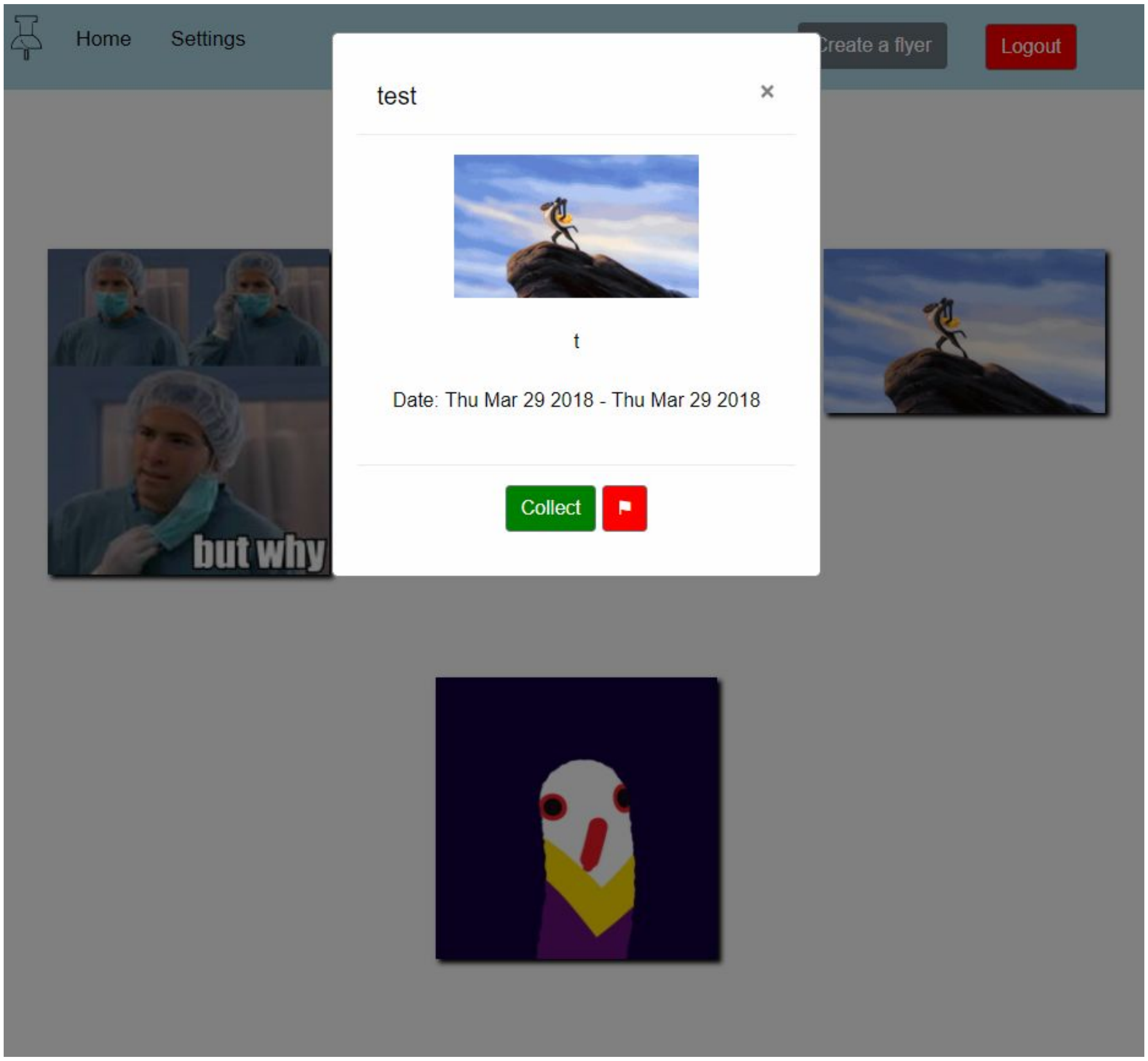
- Limit flyers displayed based on how far into the future they occur.
    - The drop down box contains 4 options: All, Day, Week, Month. This specifies how far in the future the user wants to see flyers. For example: selecting the “Week” filter, only shows events happening in the next 7 days.



Filter by: Week

Collected only ☐ Week ▾

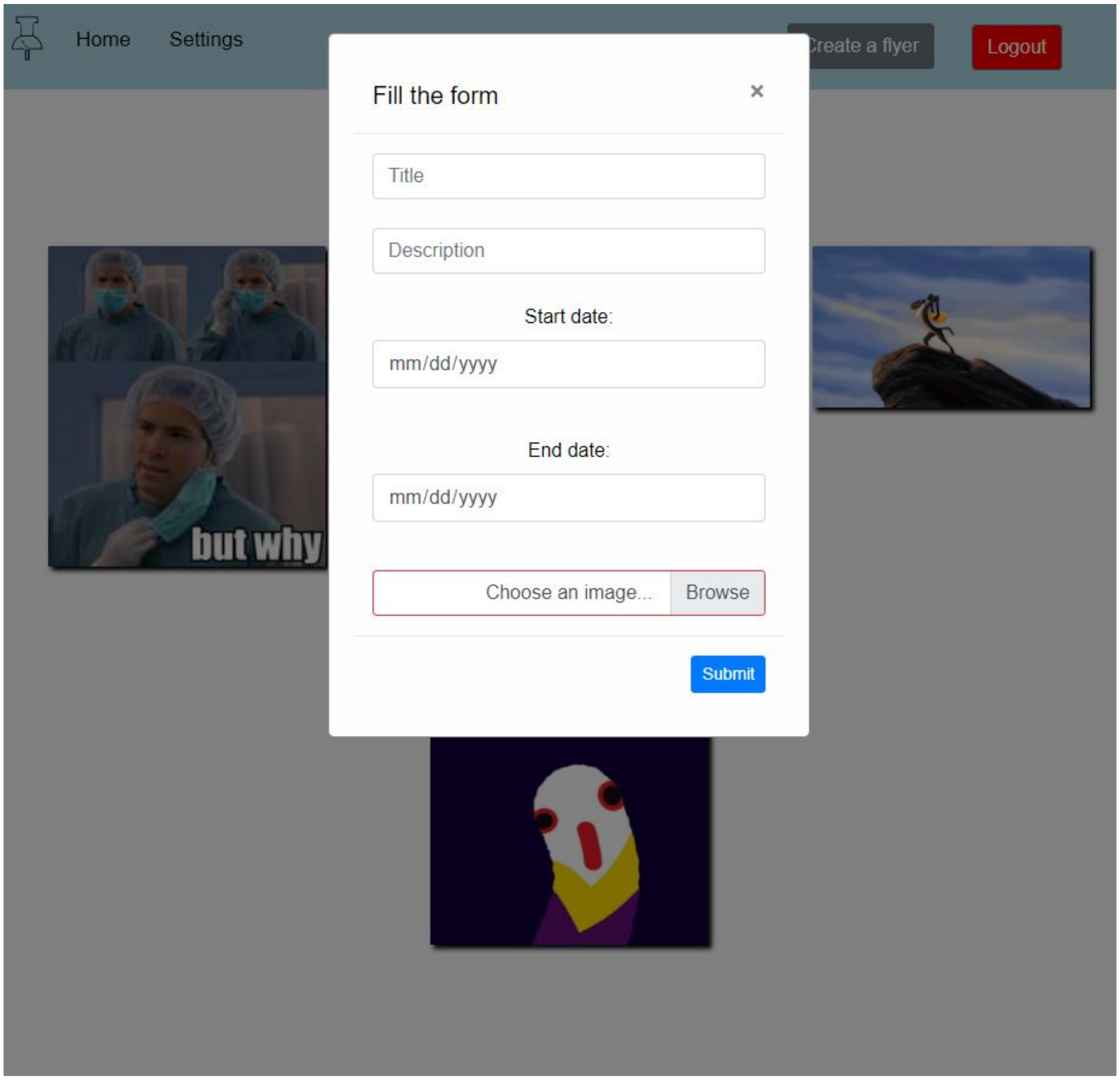
- Save a flyer to a user’s account, to be retrieved later with ease.
    - The green button marks the flyer for the user. This allows the user to save a flyer and be able to find it easily later.
- Flag a flyer by clicking the ‘flag’ icon. Currently, flagging a flyer 5 times will trigger automatic removal. The only exception is when the author of a flyer flags their own, which results in immediate removal.
    - In deployment, the counter would increase to a more reasonable number. For testing’s sake, 5 flags removes a flyer from the database. This allows the community to police itself and remove garbage.



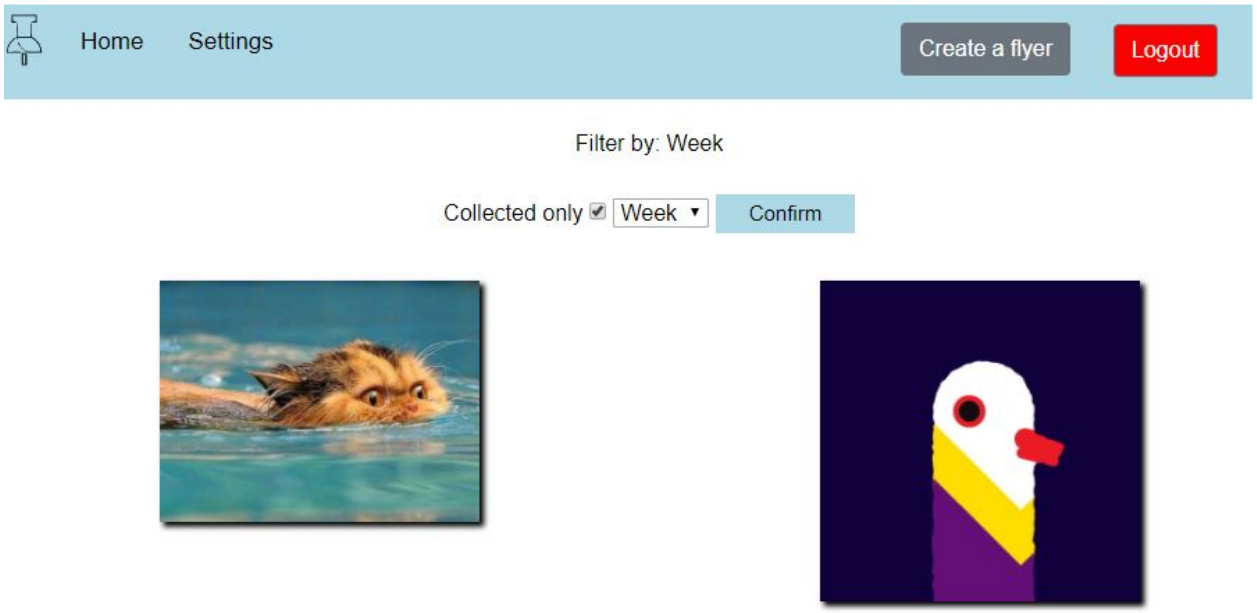
t

Date: Thu Mar 29 2018 - Thu Mar 29 2018

- Create a flyer with the ‘create’ button
    - Clicking on the create button opens a modal that asks the user for the necessary inputs to create a valid flyer. It enforces all inputs are filled and makes sure the dates make logical sense.



- Viewing a user’s collected flyers
  - The user can choose to only view flyers it collected in the past, while still following the selected time frame.
  - The example below shows the same “Week” time frame but only the collected flyers.

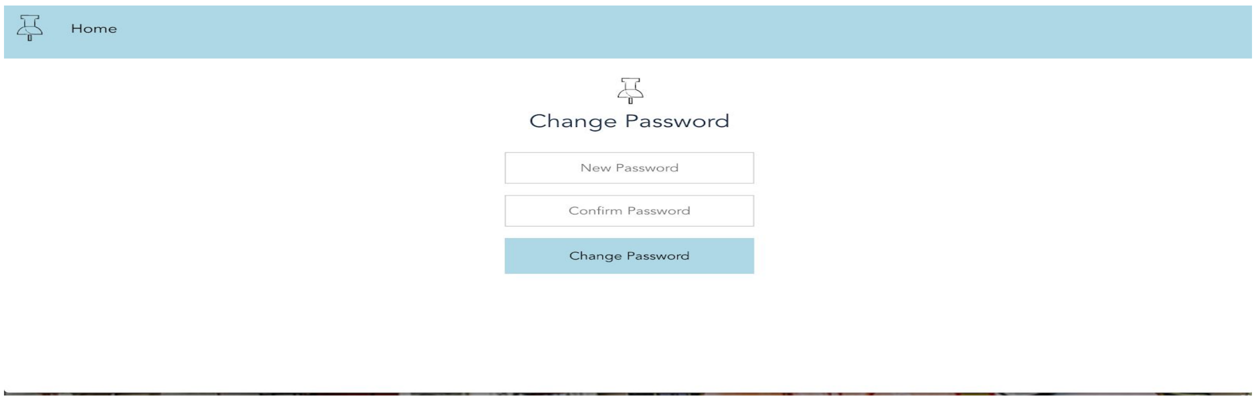


- The user can log out of their account
  - When the user clicks on the red button, they are logged out and returned to the Login Screen.

Settings:

In this page, a user is able to change their password.

- User simply types in new password they would like to have.
- Re-type the same password in the next text box, and an alert will appear to tell you if your password change was successful or not.



Test Setup

Due to the nature of the project, manually creating all necessary data (users, flyers) can be easily done, but our team has pre-filled our database with enough information to be able to start running. This includes Users to log in as, flyers to view, and variable environment variables.

What you need…

- Node package manager (npm) + nodejs
- https://www.npmjs.com/get-npm

Once you have npm installed you will need to start up both the frontend + the backend server using npm.

To initialize the backend server:

1. Go into the backend directory.
2. Run npm install
3. Run node app.js to start the server.

To initialize the frontend server:

1. Go to the frontend directory.
2. Run npm install
3. Run npm start

Once both are running, the website should be accessible via http://localhost:8080/

If you would like to test the website using your own mongo database then you can replace the MONGO_URL value located in /backend/.env with a custom mongo url.

For using any of the logins that our team have provided, the following login information should work:

- User: test@purdue.edu, password:
- User: kevin@purdue.edu, password: password