README.md

Austin Conservation Project Website

This project contains the website created for the Austin Conservation Project, hosted on AWS at austinconservationproject.org.

This project was built by the 501c3 nonprofit Vidare

What is needed

This project is built using React, a javascript library. It also requires npm to run. Python3 will also be required to make changes to the more dynamic parts of the website. You can learn more about React here You can learn about how to install and use npm here You can learn about Python3 here (it's probably already installed on your system)

When developing this project, we used git and github to keep track of different project versions. Here is a brief introduction to git and github. A version control system isn't necessary to change and deploy the website. However, it's useful when more than one person is working on the website, or when you want to save your changes somewhere safe.

Starting From Zero on MacOS

Follow this process to install all the necessary software to download and run the website on your local machine. These commands are for a specific user, but can be adapted for others.

1. Create a github account at github.com, and request access to the project
2. Open the terminal program - this comes standard on MacOS
3. Brew is a package manager for MacOS that lets you easily install new software. Install brew by pasting the following command (excluding the $) in the terminal then pressing enter. `$/bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh)"`
4. Type the following command in the terminal to install git (always exclude the $) `$brew install git`
5. Type the following command in the terminal to install node (which contains npm) and python3 `$brew install node python3`
6. Type the following command to change into the 'Desktop/ACP Stuff' `$cd "Desktop/ACP Stuff"`
7. Download this project using git `$git clone https://github.com/cdfritz7/ACPWebsite`
8. Change into the newly downloaded folder 'ACPWebsite' `$cd ACPWebsite`
9. Change into the 'acp-app' folder, which contains the code for the project `$cd acp-app`
10. Use npm to run the code, this should open up a browser window with the website pulled up `$npm start`

Common Commands for Unix (MacOS) and Linux Systems

Use the following commands to navigate your computer's file system while in the terminal

Think of the file system like an upside tree - you start at the 'root' and each new folder is a 'branch' that contains other folders (other 'branches') or files ('leaves')

Folders are also called 'directories' The <> symbols indicate parts of the command you should replace

`$cd <folder-name>` : **change directory** - change into the directory called folder-name - Again, think of directories like an upside down tree, - `./` is the directory you are currently 'in' - `../` is the directory 'above' you, so you'd use `$cd ..` to move up one directory - `~/` is the 'root' directory, so you'd use `$cd ~` to change to the root directory, which probably contains directories like 'Desktop' and 'Downloads'

`$ls` : **list** - list all the files and directories contained by the directory you are currently in

`$pwd` : **print working directory** - print the name of the directory you are currently in

How Everything Works and Project Structure

How React Works

React is a javascript library used to build user interfaces. It uses 'components' as building blocks. These components can be things like images, sections in a website, or buttons. You can apply individual styling to each component, or reuse components. You can change the presentation of reused components by passing in information through 'props' (short for properties) when the component is created. Components can also keep track of their own 'state'.

Example - on the faces of austin page, the 'carousel' containing the Faces of Austin videos is a component. When the component is created, a javascript object containing the links to eac of the videos is passed to the carousel as a prop. The currently displayed video link is stored in the components state. When a user clicks one of the buttons, it updates the state of the component, resulting in a new video being displayed.

The base of this website is the index.html document. This is what AWS sends to users when they visit this website. React injects code into index.html. It performs this injection in index.js by adding the App component to the 'root' of the index.html document. The App component comes from App.js file, and contains the navigation bar and website footer. The App component also renders another component between the navigation bar and website footer. This other component makes up the bulk of the page, and comes from another file such as AboutComponent.js.

The Project Structure

Directories shown in bold

- README.md - this file
- update_content.py - Python3 code to that transforms information stored in csv files into a form useable by the front end
- **content_csv** - contains .csv files containing information stored in the more dynamic parts of the website that have to be updated often
  - events.csv - .csv file containing information to be stored in the website calendar
  - faces_of_austin.csv - .csv file containing information to be stored in the faces of austin carousel
- **acp-app** - contains the code for the frontend of the project
  - **build** - contains an optimized version of the website code. Upload the files here to the AWS s3 bucket to deploy the website publically
  - **node_modules** - contains extra code from outside sources that the website uses - ignore this
  - package.json - contains information about the website and is used by npm. ignore this
  - package-lock.json - ignore this
  - **public** - contains the index.html file among other things. The index.html file is the 'base' of the website. React injects code into this to render the website. If you want to change things like picture in the browser tab, index.html is the place to do it. ignore everything else in this directory.
  - **src** - contains the actual code of the website
    - AboutComponent.js - contains the component corresponding to the about page
    - App.js - contains the navbar, footer, and routing information. Components from other .js files are rendered here between the header and footer
    - ApplyComponent.js - contains the component corresponding to the page with applications
    - ContactComponent.js - contains the component corresponding to the contact page
    - **content_json** - this directory holds information that components use. ex - contains a file with the links to each of the Faces of Austin videos. update_content.py transforms the .csv files in **content_csv** into the files in this directory.
    - **css** - contains .css files for each component. These hold styling information.
    - EventsComponent.js - contains the component corresponding to the page with the calendar
    - FacesComponent.js - contains the component corresponding to the page with the Faces of Austin Carousel
    - **fonts** - contains the fonts (like cocogoose bold)
    - HomeComponent.js - contains the component corresponding to the page with the ACP logo and skyline
    - **imgs** - contains the images rendered in the website

- **imgs_original** - contains the images in **imgs** but in their original, full sizes
- index.js - contains the code that injects into public/index.html - ignore this
- PeopleComponent.js - contains the component corresponding the page with leadership and membership monday
- reportWebVitals.js - ignore this

How to Make Changes

Uploading New Faces of Austin Videos

1. Open up `./content_csv/faces_of_austin.csv` in excel or another spreadsheet editor.
2. Edit each of the columns to update or add new vides. Do not change the columns themselves, and be sure to fill out each column when adding a new video
3. Run update_content.py with a command such as `$python3 update_content.py` or a python IDE
4. Your changes should be reflected in the next build. If using the development website on localhost:3000, you may need to rerun `$npm start`

Uploading New Events

1. Open up `./content_csv/events.csv` in excel or another spreadsheet editor.
2. Edit each of the columns to update or add new events. Do not change the columns themselves, and be sure to fill out each column when adding a new event
3. Run `update_content.py` with a command such as `$python3 update_content.py` or a python IDE
4. Your changes should be reflected in the next build. If using the development website on localhost:3000, you may need to rerun `$npm start`

Uploading New Member Monday Pictures

1. Simply put the pictures in `./acp-app/src/imgs/member_monday/` folder
2. The website will pull the images from this folder on deploy to populate the Member Monday carousel

Uploading New Pictures to the Front Page

1. Simply put the new pictures in `./acp-app/src/imgs/front_page_imgs/` folder
2. The website will pull the images from this folder on deploy to populate the front page

Anything Else

Changing anything else will require editing the javascript and css files in `./acp-app/src/` . For instance, if you wanted to change the a paragraph on the about page, you would do the following.

1. Open `./acp-app/src/AboutComponent.js` in a text editor or IDE
2. Change the content
3. Change `./acp-app/src/css/App.css` or `./acp-app/src/css/About.css` to change the styling of the paragraph.

Often, it will be beneficial to find an example of what you want to do somewhere in the codebase. For example, if you want to add a new image, find somewhere in the codebase with an image, understand how the image is added, then use that knowledge to add an image in a new place. Google is also your friend!

Each React component will look something like this : `<p className="p-txt"> example text </p>` . `className` is a prop that all React components have. It determines what css styles are applied to that component. If you wanted to change the style of this paragraph, you would need to find the `p-txt` css class, or add a new css class (then list this new class in the `className` prop. These classes are found in css files, and look something like this :

```
.p-txt{
```

```
        font-size: 12pt;
        color: black;
}
```

Running on a Local Machine (Your Own Computer)

This project is built using React, and requires npm to run. Type the following commands in the command line to run the app on a linux or MacOS system. To view the development website, type "localhost:3000" into the browser search bar. Saved changes in the code base should be immediately reflected on the development website. This makes development a lot easier since you should only have to run the following commands once!

```
$cd acp-app
$npm start
```

Building the Website to Deploy

Use the following commands to build a static version of the application to be deployed. The results will be stored in ./acp-app/build/

```
$cd acp-app
$npm run build
```

Deploying the Website

0. Perform the steps under **Building the Website to Deploy**
1. Navigate to AWS here and log in with your username and password.
2. Click on `austinconservationproject.org` under `name`
3. Click the `upload` button on the right
4. Upload ALL of the files and directories from the `./acp-app/build` directory
5. Wait for a few minutes, any new changes should be reflected on the website

Uploading Your Changes to Github

1. Change to the `ACPWebsite` directory, which contains the `acp-app` website
2. Type the following command in the terminal to 'stage' your changes to be uploaded `$git add .`
3. Type the following command in the terminal to 'commit' your changes with a message `$git commit -m "your-message-here"`
4. Upload your changes to github.com `$git push`