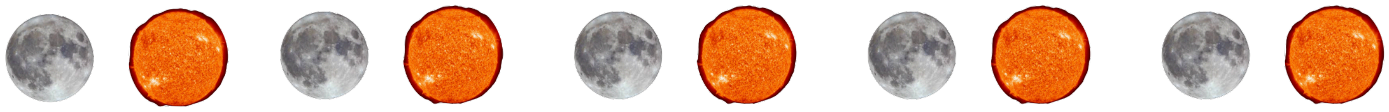
SUN & MOON

Overview:

It was a paired project using Rect router and API to build an app. Marissa and I are decided on taking astro-theme. A Web App that accurately shows the state and phase of the Sun and Moon in any place with daily-updated horoscope.

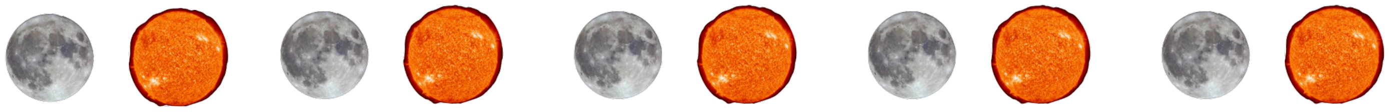
We used few APIs that transfer data from one to another and few pages for a good UX.

Brief:



- Render a React App in the browser;
- API uses;
- Include separate HTML / scss / Components files;
- Use React for DOM manipulation;
- Deploy App online, using Github Pages, where the rest of the world can access it;
- Use semantic markup for HTML and CSS (adhere to best practices).

Technologies Used:



- HTML;
- CSS3 with animation;
- JavaScript (ES6);
- JSX;
- Git;

- GitHub;
- React.
- Bulma.

Approach Taken:

We had to think of UX, user-interaction and data flow. Crete home page and implement react router.

```
import React from 'react'
import ReactDOM from 'react-dom'
import { BrowserRouter, Switch, Route } from 'react-router-dom'

import Home from './components/Home'
import Weather from './components/Weather'
import SunAndMoon from './components/SunAndMoon'
import Horoscope from './components/horoscope'
import Landing from './components/Landing'



const App = () => (
  <BrowserRouter>
    <Switch>
      <Route exact path="/" component={Landing} />
      <Route exact path="/home" component={Home} />
      <Route path="/weather/:city/:country" component={Weather} />
      <Route path="/sunandmoon/:lat/:lon" component={SunAndMoon} />
      <Route path="/horoscope/" component={Horoscope} />
    </Switch>
  </BrowserRouter>
)
```

Components:

├───Home.js ├───horoscope.js ├───Landing.js ├───LocationForm.js

├───SunAndMoon.js ├───Weather.js └───WeatherForm.js

Weather

API

```
    componentDidMount() {
      const { city, country } = this.props.match.params
      console.log(city)
      console.log(country)

      axios.get(`http://api.openweathermap.org/data/2.5/weather?q=${city},${country
        .then(resp => this.setState({ data: resp.data }))
        .catch(err => this.setState({ errors: err.response.data.errors }))

    }
```

Sun and Moon

API

```
  componentDidMount() {
      const { lat, lon } = this.props.match.params
      console.log(lat)
      console.log(lon)

      axios.get(`https://api.ipgeolocation.io/astronomy?apiKey=c75530c7c6e2481ea5f8
        .then(resp => this.setState({ data: resp.data }))
        .catch(err => this.setState({ errors: err.response.data.errors }))

      axios.get(`http://api.farmsense.net/v1/moonphases/?d=${newDate}`)
        .then(res => this.setState({ moonData: res.data }))
    }
```

Horoscope

API

```
    componentDidMount() {
      const { lat, lon } = this.props.match.params
      console.log(lat)
      console.log(lon)
      axios.get('https://www.horoscopes-and-astrology.com/json')
        .then(resp => this.setState({ data: resp.data }))
        .catch(err => this.setState({ errors: err.response.data.errors }))
    }
```

Render issue

I had to transform string in order to receive needed information from API.

```
   <div className="content">
   {this.state.data.dailyhoroscope.Aries.substr(0, this.state.data.dailyhoroscope.A
   {/* {<img src="https://horoscopes-and-astrology.com/images/aries.svg"></img>}  <

  <div>{<a href="https://horoscopes-and-astrology.com/aries?LANGUAGE=EN" target=" b
```

Moving data

We had to pass on data from one page to another. `<Link className="button" id="sunMoonButton" to={` /weather/${this.state.data.city}/${this.state.data.country} `}> Enter </Link>`

` <Route path="/weather/:city/:country" component={Weather} />`

```
      const { city, country } = this.props.match.params
      console.log(city)
      console.log(country)
```

` axios.get(`http://api.openweathermap.org/data/2.5/weather?q=${city},${country}&appid=429736441cf3572838aa10530929f7cd`)`

```
        .then(resp => this.setState({ data: resp.data }))
        .catch(err => this.setState({ errors: err.response.data.errors }))
    }
```

Final Product:

Future Enhancement

There are several potential future features that can be implemented, such as:

- Add Astro - Maps;
- Add Compass;
- Mobile version (Responsive).