AI-VVO sdmay22-36 Spring 2022 Update #2

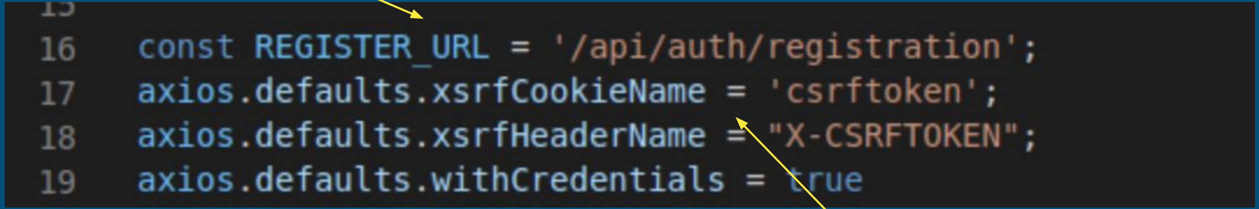
2/4/2022 - 2/9/2022

Front-end (This Week)

- Implemented Register.js file to verify end-to-end communication
- Researched axios.post requests
- Worked with backend team to get end-to-end communication to work

Axios Setup

Must have path leading to database as set up by Backend



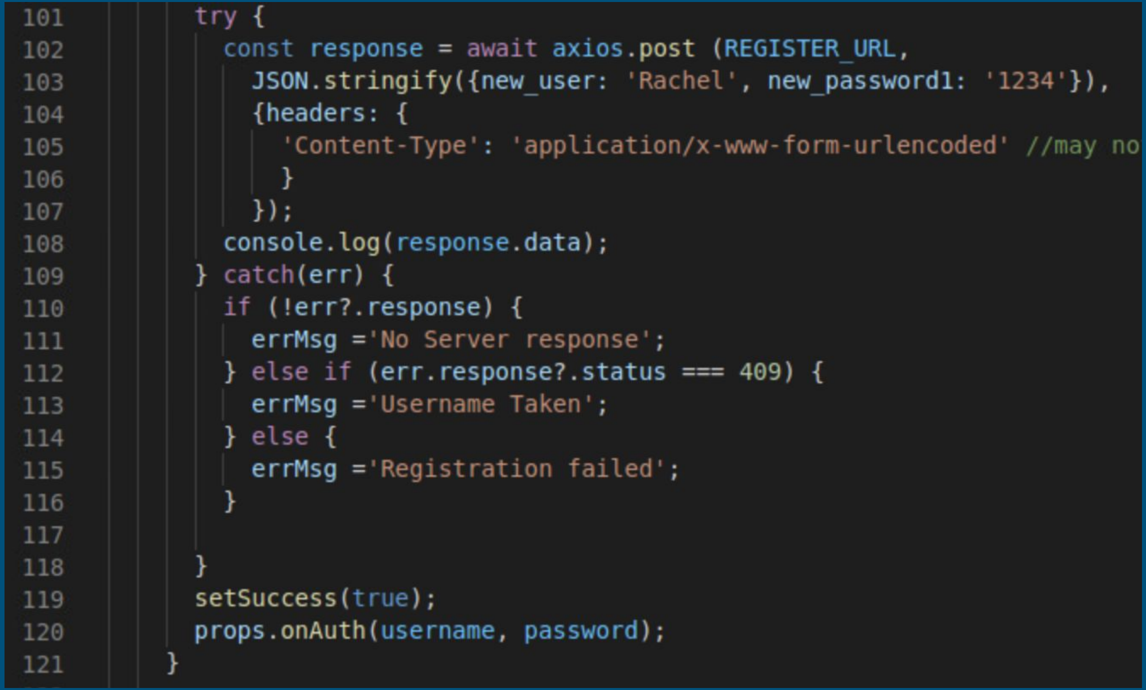
```
const REGISTER_URL = '/api/auth/registration';
axios.defaults.xsrfCookieName = 'csrftoken';
axios.defaults.xsrfHeaderName = "X-CSRFTOKEN";
axios.defaults.withCredentials = true
```

Must assign a CSRF Token for the secure method to be passed to the backend

Axios Post Method

- Send POST method using Axios library using JSON stringify method to format the data to be sent.

- Axios is a “promise-based” resulting in more readable responses from HTTP methods



```
  try {
    const response = await axios.post (REGISTER_URL,
      JSON.stringify({new_user: 'Rachel', new_password1: '1234'}),
      {headers: {
        'Content-Type': 'application/x-www-form-urlencoded' //may no
        }
      });
    console.log(response.data);
  } catch(err) {
    if (!err?.response) {
      errMsg ='No Server response';
    } else if (err.response?.status === 409) {
      errMsg ='Username Taken';
    } else {
      errMsg ='Registration failed';
    }

  }
  setSuccess(true);
  props.onAuth(username, password);
}
```

Front-end (Next Week)

- Start looking into how to get grid data onto map
- Determine who will be doing what
- Begin updating the user interface
- Look into making SVD file icons for use as custom icons on map

Back-end (This Week)

- Create .env file for influx which automatically sets up the influxdb with proper user and password through docker.
- Create user registration API which allows users on the frontend to create their own user to login to the website with
- Updated csvParser.py to automatically create the buckets before populating them with data for ease of use
- Found a way to allow users on one VM to connect to the Influxdb running on another VM

Back-end (Next Week)

- Prepare to make sure that the VM running openDSS will be able to get the data from influx as well
- Look for any ways to improve existing scripts. Possible ways to make a more efficient query to Influx

Documentation (This Week)

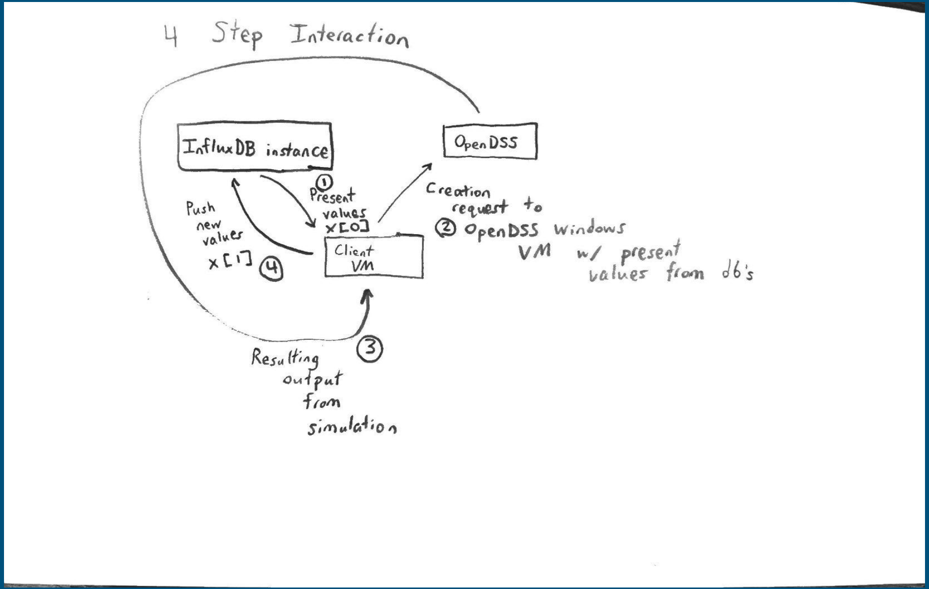
- Looked over code to see how to organize documentation
- Plan to have 4 documentation files:
    - README
    - Frontend
    - Backend
    - Machine Learning
- Started rough draft of docs (in google doc)

Docs (Next Week)

- Finish to write out documentation (.rst file)

Machine Learning (This Week)

- Coordinated with the backend team to ensure functional communication between the VMs
- Researched time-series simulations in OpenDSS
- Sketched data flow between the Windows OpenDSS machine and the client VMs


4 Step Interaction
InfluxDB instance
OpenDSS
① Present values x[0]
Push new values x[1] ④
Client VM
Creation request to ② OpenDSS Windows VM w/ present values from db's
Resulting output from simulation ③

Machine Learning (Next Week)

- Dive deeper into the Python-OpenDSS API since inter-machine communication between VMs and applications works now
- Get the Windows VM up and running
- Implement the 4-Step Interaction process proposed in the previous slide