3a.

The programming language that I used to create my website as HTML, CSS, & Javascript. The purpose of my website was to be part of a bigger website that was going to be used in a previous presentation of innovation for the University of Nebraska at Omaha's Industrial Technology Innovation Competition. It was going to be used for a user to get assigned a scooter for their personal use in an attempt to solve the last mile problem of today's transportation. The video demonstrates a mock user clicking on their location from a dropdown, and the program outputting a scooter near them along with a map of their location. The input of the program is the user selecting their location from a dropdown menu, and the output is a scooter assigned to them along with a map of their location compared to the scooter.

3b.

```
class scooter{  //a group of all the elements needed for functions below
    constructor(){
        this.scooters = ["ETZ5-IG26", "9AHJ-Q81F", "5RM5-KU5I", "8IT5-R3OR", "ZQXR-3LYV"]; //list of scooters that the user will be interacting with
        this.locations = ["by the Bank", "by Bob and Judy's Hotel", "by North High School", "by Redcircle Stadium", "by the Mayor's Office"];
                                                                    //list of possible locations where the scooters and user might be
        this.scooterLocations = []; //list of locations of scooters corresponding with their same indexed scooter that is in the scootersInUse list
        this.locationChosen = 0; //variable used for a random index for the scooter(s) location
        this.num = 0; //variable used for the random number needed for the random location index
        this.numberOfScootersInUse = 10; //variable used to represent the amount of scooters in use for the site
        this.userAndScooterLocation = 0; //variable used to hold the same location of the scooter and user
        this.scooterChosen = ""; //variable used to hold the scooter chosen for the user by the same location
        this.time = 0; //variable used to hold the amount of times that it has to rerun the random location functions
                                                            //to get the same location for the user and scooter
        this.locationOfScooterAndUser = ""; //variable used to hold the location of the same location and user
    }
}
```

```
function getRandomNumber(){ //gets a random number for the location index and stores the random value in the num variable
    this.projectHoneycomb.num = Math.floor(Math.random() * 5); //uses math function built into Javascript to receive a random number in the constraint of 5
}

function getRandomLocation(){ //gets a random location using the getRandomNumber() function and stores it in the variable used for the random index of locations
    getRandomNumber(); //calls the getRandomNumber() function to get a random number for the random location
    this.projectHoneycomb.locationChosen = this.projectHoneycomb.num; //assigns the random number to the location of the scooter in question
}

function assignRandomLocationToScooter(){ //puts the random location into the scooterLocations array so that the location randomly
                                          //chosen will correspond with a scooter in the scooters array with the same index number
    this.projectHoneycomb.scooterLocations.push(this.projectHoneycomb.locations[this.projectHoneycomb.locationChosen]);
                                //adds the random location to the list of scooter locations to line up with the scooter assigned
}

function getAllLocationOfScooters(){ //gets random locations for all the scooters in the scooters array list
    for(i = 0; i < this.projectHoneycomb.scooters.length; i++){ //used to loop the calling of the following functions
                                        //until the variable 'i' is equal to the scooters list amount in the list
        getRandomLocation(); // calls the function to get a random location
        assignRandomLocationToScooter(); //calls the function to assign that random location to a scooter
    }
}
```

The list shown above is titled scooterLocations. The data that is represented in the list is all the locations that the scooters are located as generated through student wrote algorithms. These algorithms output random numbers that are later used to generate random locations for the scooters. These random locations are then stored in the list as shown in the second code segment. If I did not use the list as shown above in the two code segments, then I would have had to use a lot of it, and, else loops in place of the list. For the list to even exist in the first place, I would have to have several variables that would represent one item in each list. Each variable would have to be either a string or a number which you cannot mix without errors. Once I had these variables initialized then I would have to use if, and, else loops for each variable to check them against the user's input. Furthermore, I would have to create new variables each time I wanted to add a new thing, in addition to the fact that I would have to have new algorithms to view those variables too.

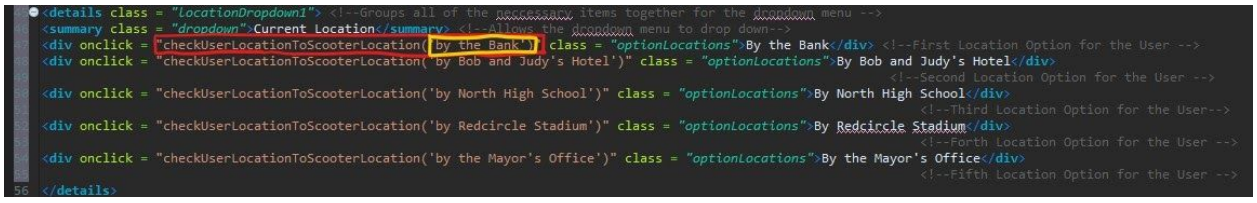
3c.

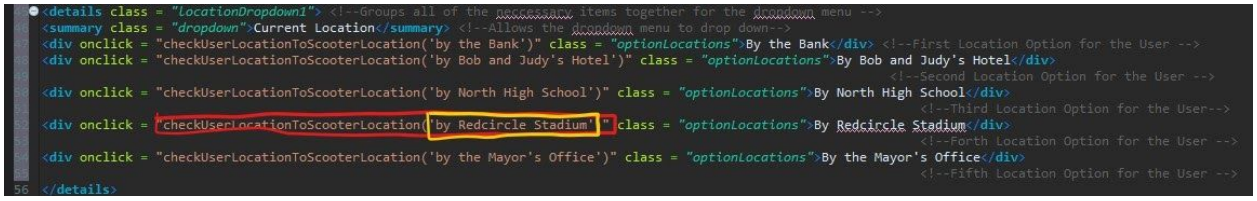
```
function checkUserLocationToScooterLocation(userLocation){ //compares the different locations in the scooterLocations array list to the userLocation
    for(i = 0; i < this.projectHoneycomb.scooterLocations.length; i++){ //used to loop until the userLocation lines up with a location for a scooter
        this.projectHoneycomb.userAndScooterLocation = this.projectHoneycomb.scooterLocations.indexOf(userLocation);
                                                    //figures out if there is a scooter sharing the same location as the user.
        this.projectHoneycomb.locationOfScooterAndUser = userLocation; //assigns the location of the user to a variable
    }
    if(this.projectHoneycomb.userAndScooterLocation > -1){ //if the scooter has a location that is the same as the user then it returns a number greater then one
        this.projectHoneycomb.scooterChosen = this.projectHoneycomb.scooters[this.projectHoneycomb.userAndScooterLocation];
                                                    //assigns the variable scooterChosen to the scooter in the array that has the same location as the user
        assignAScooterToTheUser(); //assigns the scooter that shares the same location as the user to the user
        showMapOfScooterLocation(); //calls the function to show the map of the location shared by the user and scooter
    }
    else if(this.projectHoneycomb.userAndScooterLocation == -1 || this.projectHoneycomb.time > 0){
                                    //if there isn't a scooter that shares the same location as the user or the time increment is greater the zero this runs
        alert("Please wait while we locate the nearest scooter for you."); //shows alert to notify the user if there isn't a scooter available near them
        this.projectHoneycomb.time++; //increments the time variable up to show how many times it has run
        getAllLocationOfScooters(); //runs function to get all new locations for all scooters
        checkUserLocationToScooterLocation(this.projectHoneycomb.locationOfScooterAndUser);
                                                    //calls this function again and has the locationOfScooterAndUser as a parameter
    }
}
```

```
<details class = "locationDropdown1"> <!--Groups all of the neccessary items together for the dropdown menu -->
<summary class = "dropdown">Current Location</summary> <!--Allows the dropdown menu to drop down-->
<div onclick = "checkUserLocationToScooterLocation('by the Bank')" class = "optionLocations">By the Bank</div> <!--First Location Option for the User -->
<div onclick = "checkUserLocationToScooterLocation('by Bob and Judy's Hotel')" class = "optionLocations">By Bob and Judy's Hotel</div>
                                                                                                <!--Second Location Option for the User -->
<div onclick = "checkUserLocationToScooterLocation('by North High School')" class = "optionLocations">By North High School</div>
                                                                                                <!--Third Location Option for the User-->
<div onclick = "checkUserLocationToScooterLocation('by Redcircle Stadium')" class = "optionLocations">By Redcircle Stadium</div>
                                                                                                <!--Forth Location Option for the User -->
<div onclick = "checkUserLocationToScooterLocation('by the Mayor's Office')" class = "optionLocations">By the Mayor's Office</div>
                                                                                                <!--Fifth Location Option for the User -->
</details>
```

The program above receives the user's location as an input that will be used to check against the locations of the scooters. This algorithm starts by taking the user's location as a parameter for the function. It has a loop that repeats until the user's location lines up with one of the locations of the scooters. Inside the loop, it checks for the user's location in the list of locations for the scooters. It then saves the user location in a variable for other functions. After the loop completes, there is an if, else loop that checks if the variable returned from comparing the locations is greater than -1. If it is -1 then it outputs a waiting message and reruns the other functions to generate new locations for the scooters. Then it repeats this function all over again to compare the new locations to the user location. If the variable returns greater than -1, the function displays the scooter with the same location, how many scooters are now in use, and a map of the scooter's location. The whole purpose of this function is to compare the user's location to the scooter's and then output the scooter and map.

3d.





The first call checks the parameter 'by the Bank' in its conditions for a scooter shares the same location as the user. If it does, the output of the condition is the scooter with the matching location, map, and increments the number of scooters in use up. If there isn't a scooter that has the same location as the user, conditions output a message and rerun the functions that are called on the load of the page. The second call checks the parameter 'by Redcircle Stadium' in

the conditionals for a scooter that has the same location as the user. The rest of the calls are interchangeable compared to the first call except for the map and scooter which change as there are different locations. The result of the first call includes the number of scooters being incremented up, the map of ‘by the Bank’ location displayed, and the scooter that has the same location displaying for the user. This occurs if the locations are identical, however, if that is not the case a message displays and the other functions are called. The second call's results are identical except the location changes from 'by the Bank' to 'by Redcircle Stadium'.