Parkland Student Involvement Tracker

Antony Tomy & Alexander Saccone

FBLA 2023 Coding & Programming
Parkland High School

Programming Goals

01

Give and Save Points for Student Activities

02

Award Prizes to Students According to Points Earned

03

Generate a Quarterly Report with Student Information and Points

Basic Functionality: Front-end

- HTML 5
- CSS 3
- Javascript
- Bootstrap 5 Framework
- Google Fonts

Bootstrap 5 - popular HTML, CSS, and JavaScript framework for creating responsive websites

Google Fonts

Google Fonts - Icons and Fonts

Basic Functionality: Back-end

- Node.js
- Express.js Framework
- JSON

Node.js - a back-end web environment that allows for javascript code to be run locally

Express.js - a back-end web application framework for building RESTful APIs with Node.js

Backend Node Server using Express

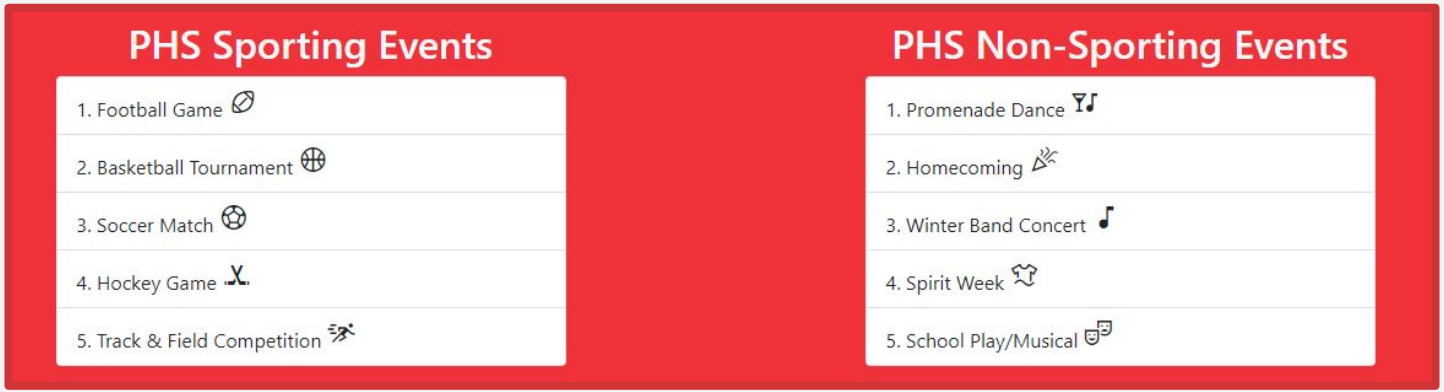
- Server renders Home.html file that is linked with accompanying CSS and Javascript File
- GET Request at */data* route that reads local data.json file with Node filesystem and responds to client with output
- POST Request at */updateData* route that deletes old data.json file and creates new file with updated data from the request

index.js > ...

```
//importing required libraries and express and intializing variables and environment for Express framework
const express = require('express')
const router = express.Router()
const fs = require('fs');
const bodyParser = require('body-parser');
const app = express()
const port = 3000
app.use(express.static("public"))
router.use(bodyParser.json());
app.use(bodyParser.json());
app.set("view engine","ejs")

//rendering Home HTML file when user goes to server
app.get('/', (req, res) => {
  res.render("home.ejs")
})

//Defining GET Request for JSON data
app.get('/data', (req, res) => {
    //reading and sending data  to user and response in JSON Format
    let JSONdata = fs.readFileSync('data.json');
    res.send(JSONdata)
})

//Defining POST Request that will update JSON file with new data from Request
app.post('/updateData', (req, res) => {
    //Deleting Old Data.json file
    fs.unlinkSync('data.json');

    //Creating new File with new data in JSON format and sending success message to client
    fs.writeFileSync('data.json', JSON.stringify(req.body));
    res.send(res.json("Added data successfully"))
  })

//Required for Express app server
app.listen(port, () => {
  console.log(`APP LISTENING ON PORT ${port}`)
})
```

Parkland High School Events

- Includes 5 Sporting Events and Non-Sporting Events
- The template for the list was created using the Bootstrap database
- The HTML code for the icons was obtained from Google Fonts

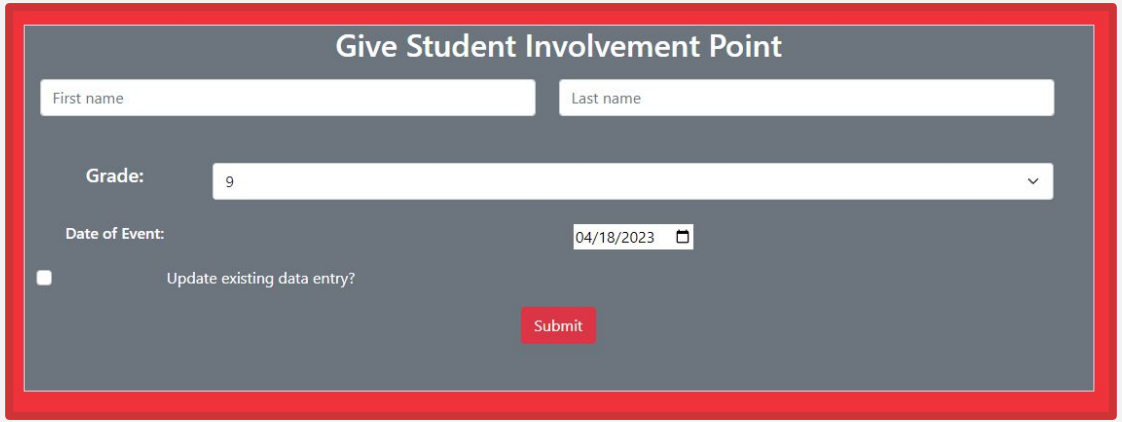


Parkland High School Events Implementation

```
<!-- PHS Events list -->
<div class="container">


    <div class="row">
      <div class="col-1"></div>

     <!-- Sporting List -->
      <div class="col-4 my-4">
        <h2 class="text-center text-light">PHS Sporting Events</h1>
        <ol class="list-group list-group-numbered">
            <li class="list-group-item">Football Game <span class="material-symbols-outlined">sports_football</span></li>
            <li class="list-group-item">Basketball Tournament <span class="material-symbols-outlined">sports_basketball</span></li>
            <li class="list-group-item">Soccer Match <span class="material-symbols-outlined">sports_soccer</span></li>
            <li class="list-group-item">Hockey Game <span class="material-symbols-outlined">sports_hockey</span></li>
            <li class="list-group-item">Track & Field Competition <span class="material-symbols-outlined">sprint</span></li>
        </ol>
      </div>

      <div class="col-2"></div>

        <!-- Non-Sporting List -->
      <div class="col-4 my-4">
        <h2 class="text-center text-light">PHS Non-Sporting Events</h1>
        <ol class="list-group list-group-numbered">
            <li class="list-group-item">Promenade Dance  <span class="material-symbols-outlined">nightlife</span></li>
            <li class="list-group-item">Homecoming  <span class="material-symbols-outlined">celebration</span></li>
            <li class="list-group-item">Winter Band Concert  <span class="material-symbols-outlined">music_note</span></li>
            <li class="list-group-item">Spirit Week  <span class="material-symbols-outlined">laundry</span></li>
            <li class="list-group-item">School Play/Musical  <span class="material-symbols-outlined">theater_comedy</span></li>
        </ol>
      </div>

      <div class="col-1"></div>
    </div>
</div>
```

Adding Involvement Point Form

- Includes Submit button with the following inputs
  - Name Text Boxes
  - Grade Drop Down
  - Date Of Event Selector
  - New Student Checkbox
  - Submit Button
- Utilizes Bootstrap 5 components and classes
- Form utilizes validation and cannot submit unless all inputs are filled properly
- Button turns green to notify if data has been successfully saved



Adding Involvement Point Form Implementation

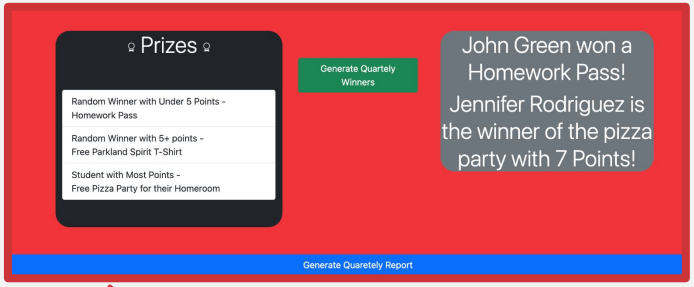
```
<!-- Form for adding points -->
<div class="col-6 text-center text-white bg-secondary border">
  <h2>Give Student Involvement Point</h1>
    <form  class="row g-3" id="pointForm" action="/givePoint">
        <div class="row my-4">
            <!-- First Name -->
            <div class="col">
              <input name="firstName" id="firstName" type="text" class="form-control" placeholder="First name" aria-label="First name" required>
            </div>
            <!-- Last Name -->
            <div class="col">
              <input name="lastName" id="lastName" type="text" class="form-control" placeholder="Last name" aria-label="Last name" required>
            </div>
          </div>

          <div class="row my-4">
           <!-- Grade Selector -->
           <div class="col-2">
              <h5><label for="select">Grade:</label></h5>
           </div>
           <div class="col-10">
              <select name="grade" id="grade" id="select" class="form-select" aria-label="Default select example" required>
                  <option value="9">9</option>
                  <option value="10">10</option>
                  <option value="11">11</option>
                  <option value="12">12</option>
                </select>
           </div>
          </div>

        <div class="row">
            <!-- Date Selector -->
            <div class="col-2"><h6><label for="start">Date of Event:</label></h5></div>
            <div class="col-10">
                <input name="date" id="date" type="date" id="start" name="date" value="2023-04-18" min="2022-08-28" max="2023-06-19" required>
            </div>
        </div>

        <!-- Checkbox -->
        <div class="col-12">
          <div class="form-check">
            <label class="form-check-label" for="gridCheck">
                Update existing data entry?
             </label>
            <input name="checkbox" id="checkbox" class="form-check-input" type="checkbox" id="gridCheck">
          </div>
        </div>

        <!-- Submission Button -->
        <div class="col-12">
          <button id="button" type="submit" class="btn btn-danger mb-5">Submit</button>
        </div>
      </form>
  </div>
```

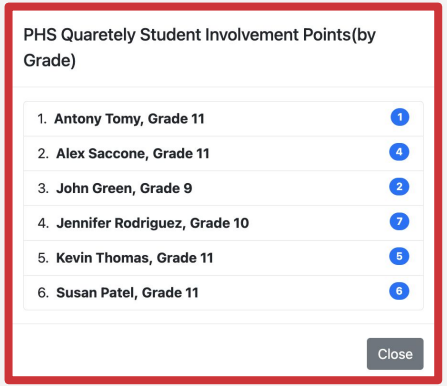
```
// if checkbox is not checked then add entry to data
if(!checkbox.checked){
    data.push([firstName.value,lastName.value,grade.value,date.value,1])
```

```
else{
    for(let i=0; i<data.length;i++){
        if (data[i][0] == firstName.value && data[i][1] == lastName.value){
            data[i][4]++;
        }
    }
```

Prize Winners and Quarterly Report

- Prizes System:
  - Random Winner with under 5 Points receives a Homework Pass
  - Random Winner with more than 5 points receives a Free PHS T-Shirt
  - Student with Most Points receives a Free Pizza Party for their Homeroom
- Quarterly Winners are generated from Button and displayed in the adjacent textbox
- Quarterly Involvement Report Button opens Bootstrap Popup Modal with Appropriate Information





HTML: Prize Winners and Quarterly Report

```
<div class="row my-5">
  <div class="col-2 my-4"></div>
  <div id="prizes" class="col-3 bg-dark">
      <h1 class="text-center text-light display-6 "><svg xmlns="http://www.w3.org/2000/svg" width="16" height="16" fill="currentColor" cl
          <path d="M9.669.864 8 0 6.331.864l-1.858.282-.842 1.68-1.337 1.32L2.6 6l-.306 1.854 1.337 1.32.842 1.68 1.858.282L8 12l1.669-.8
          <path d="M4 11.794V16l4-1 4 1v-4.206l-2.018.306L8 13.126 6.018 12.1 4 11.794z"/>
        </svg> Prizes <svg xmlns="http://www.w3.org/2000/svg" width="16" height="16" fill="currentColor" class="bi bi-award" viewBox="0 0
          <path d="M9.669.864 8 0 6.331.864l-1.858.282-.842 1.68-1.337 1.32L2.6 6l-.306 1.854 1.337 1.32.842 1.68 1.858.282L8 12l1.669-.8
          <path d="M4 11.794V16l4-1 4 1v-4.206l-2.018.306L8 13.126 6.018 12.1 4 11.794z"/>
        </svg></h1>
      <ul class="list-group py-5">
          <li class="list-group-item">Random Winner with Under 5 Points -<br> Homework Pass</li>
          <li class="list-group-item">Random Winner with 5+ points -<br> Free Parkland Spirit T-Shirt</li>
          <li class="list-group-item">Student with Most Points -<br> Free Pizza Party for their Homeroom</li>
      </ul>
  </div>
  <div class="col-2"><button id="winnerButton" type="button" class="btn btn-success mt-5 mx-3">Generate Quartely Winners</button></div>
  <div class="col-3">
  <div id="winners">
      <h1 id="randWinnerText"class="display-6 text-white"></h1>
      <h1 id="topPointsWinner" class="display-6 text-white"></h1>
  </div>
  </div>
  <div class="col-2"></div>
</div>
```

```
    <!-- Button trigger modal -->
<div class="row text-center">
    <button id="report" type="button" class="btn btn-primary " data-bs-toggle="modal" data-bs-target="#exampleModal">
      Generate Quaretely Report
    </button>
</div>


<!-- Modal -->
<div class="modal fade " id="exampleModal" tabindex="-1" aria-labelledby="exampleModalLabel" aria-hidden="true">
  <div class="modal-dialog">
    <div class="modal-content">
      <div class="modal-header">
        <h1 class="modal-title fs-5" id="exampleModalLabel">PHS Quaretely Student Involvement Points(by Grade)</h1>

      </div>
      <div class="modal-body" >
          <ol class="list-group list-group-numbered" id="modal">

          </ol>
      </div>

          <div class="modal-footer">
            <button id="close" type="button" class="btn btn-secondary " data-bs-dismiss="modal">Close</button>
          </div>
      </div>

    </div>
  </div>
```

Javascript : Prize Winners and Quarterly Report

```
// Adding each entry to list in modal when user generates quaretely report when pressing on button

reportButton.addEventListener("click",function()
{
    for (let i=0;i<data.length;i++)
    {
        // Creating HTML elements,adding the data to it, and appending it to the list
        let reportItem=document.createElement("li")
        reportItem.classList.add("list-group-item","d-flex","justify-content-between","align-items-start")
        let reportItemDiv1=document.createElement("div")
        reportItemDiv1.classList.toggle("ms-2")
        reportItemDiv1.classList.toggle("me-auto")
        let reportItemDiv2=document.createElement("div")
        reportItemDiv2.classList.add("fw-bold")
        reportItemDiv2.innerHTML= data[i][0]+" " + data[i][1]+ ", Grade "+ data[i][2].toString()
        reportItemDiv1.appendChild(reportItemDiv2)
        reportItem.appendChild(reportItemDiv1)
        let reportItemSpan=document.createElement("span")
        reportItemSpan.innerHTML=data[i][4]
        reportItemSpan.classList.add("badge","bg-primary","rounded-pill")
        reportItem.appendChild(reportItemSpan)
        modal.appendChild(reportItem)

    }
}
)

// deleting modal information when modal is closed
closeButton.addEventListener("click",function()
{
    modal.innerHTML=""
})
```

```
winnerButton.addEventListener("click",function()
{

    // creates an array of all point values
    pointsArray=[]
    for(let i = 0; i<data.length;i++){
        pointsArray.push(data[i][4])
    }

    // finds the maximum points value and then finds the index of the student with that many points
    let maxPoints=Math.max(...pointsArray);
    let maxWinnerIndex = pointsArray.indexOf(maxPoints);

    // displays the winner's name on the website
    let topPointswinnerName = data[maxWinnerIndex][0]+" "+data[maxWinnerIndex][1];
    topPointsWinnerText.innerHTML=topPointswinnerName+"  is the winner of the pizza party with "+maxPoints.toString() + " Points!"

    let randWinnerNum = Math.floor(Math.random() * (data.length))


    let randWinnerName=data[randWinnerNum][0] + " "+ data[randWinnerNum][1]

    // checks which prize the student gets depending on how many points they have
    if (data[randWinnerNum][4]<5){
        randWinnerText.innerHTML = randWinnerName + " won a Homework Pass!"
    }
    else{
        randWinnerText.innerHTML = randWinnerName + " won a free PHS Spirit T-Shirt!"
    }

});
```

Data Storage

- Data is stored locally with a JSON file named data.json
- Format for each student will be a Javascript object:
  - First Name
  - Last Name
  - Grade
  - Date of most recent Event
  - Points
- When the form on the site is filled out and a request is sent to either add a student or update a student's point value
- The JSON file is either added to or updated respectively


```
{"students":[
    {"firstName":"Antony","lastName":"Tomy","grade":11,"date":"04/23/23","points":1},
    {"firstName":"Alex","lastName":"Saccone","grade":11,"date":"09/13/22","points":4},
    {"firstName":"John","lastName":"Green","grade":9,"date":"03/01/23","points":2},
    {"firstName":"Jennifer","lastName":"Rodriguez","grade":10,"date":"10/10/22","points":7},
    {"firstName":"Kevin","lastName":"Thomas","grade":11,"date":"12/20/22","points":5},
    {"firstName":"Susan","lastName":"Patel","grade":11,"date":"01/17/23","points":6}]}
```

Data Storage Implementation

```
//function that sends get request to express server and unpacks JSON data into 2D format in data variable
async function readStudentData() {
    //intializing preprocessed and processed data variables
    let preprocessedData;

    //waiting for JSON response from express server and extracting data from response object
    const response = await fetch("http://localhost:3000/data");
    preprocessedData = await response.json();
    //putting correct form of preproccessed data into 2D array data variable
    // Format of data for each student is [First Name, Last Name, Grade, Date,Points] along with some example students
    preprocessedData["students"].forEach(element => data.push([element["firstName"],element["lastName"],element["grade"],element["date"],element["points"]]));
  }

readStudentData()
```

Data Storage Implementation Cont.

```
//preparing data to be into JSON format for POST Request
let sendData={students:[]}
data.forEach(entry =>sendData["students"].push({

    firstName:entry[0],
    lastName:entry[1],
    grade:entry[2],
    date:entry[3],
    points:entry[4]

}));
```

```
// POST method implementation:
//defining function will send fetch post request to express server to save new student entry to local JSON File
async function postData() {

    //initializing default response object that will be converted to JSON
    const response = await fetch("http://localhost:3000/updateData", {
    method: "POST",
    mode: "cors",
    cache: "no-cache",
    credentials: "same-origin",
    headers: {
        "Content-Type": "application/json",
    },
    redirect: "follow",
    referrerPolicy: "no-referrer",
    body: JSON.stringify(sendData),
    });

    return response.json(); //
}

//running function
postData()
```

Program Demonstration