TEAM 5

Homework 5 Part: 1

Students

Tanushree Sharma - 20661460
Jaco Ye - 20655033
Markus Hamann - 20666067
Remi Shah - 20674381
Sunrise Long - 20671832
Faraaz Zaiee - 20662022
Hannah Cranham - 20677914

November 17, 2019

MSCI 342

Sprint 2

Upload Page Frontend (Remi Shah)

Description

As a user I should be able to use a graphical interface to upload both raw leader data and raw first year data to the system.

Requirements

The system shall provide the user a dedicated graphical interface to upload both raw leader data and raw first year data. There shall be clear indicators to the user as to where they will be uploading these raw data CSVs.

Download Page Frontend (Remi Shah)

Description

As a user I should be able to use a graphical interface to download a CSV containing sorted leader data and another CSV containing sorted first year data from the system.

Requirements

The system shall provide the user a dedicated graphical interface to download both sorted leader data and sorted first year data. There shall be a clear indicator to the user as to what button to press to download the sorted data CSV output by either the first year or leader algorithm.

Backend Integration for Entire Application (Remi Shah)

Description

As a user I should be able to use all functionality of the system without the system breaking.

Requirements

The frontend and backend of the system shall be integrated to ensure the user is able to upload raw data and download sorted data for both leaders and first years. The system should accept user input (CSV raw data) and provide the user their expected output (CSV sorted data).

Version 1, Algorithm (first year) (Hannah Cranham)

Description

As a user I should be able to input a raw data file containing information about all first-year students and in return have the same list of students sorted into 18 teams.

Requirements

The system shall allow the user to input their raw data file such that the algorithm can sort it. After the algorithm has completed processing the data and sorting the first years into teams, the user should be able to see the sorted output. The output file should contain all of the initial raw data as well as an extra column showing the team the first year has been placed on from 1 to 18.

Version 1, Algorithm (leader) (Hannah Cranham)

Description

As a user I should be able to input a raw data file containing information about all orientation leaders and in return have the same list of leaders sorted into 18 teams.

Requirements

The system shall allow the user to input their raw data file such that the algorithm can sort it. After the algorithm has completed processing the data and sorting the leaders into teams, the user should be able to see the sorted output. The output file should contain all of the initial raw data as well as an extra column showing the team the leader has been placed on from 1 to 18.

Sprint 3

Upload Page (Tanushree Sharma)

Description

As a user I should be able to upload raw leader data into the system and separately upload a leader summary and raw first year data into the system.

Requirements

The system shall have a home page that includes two upload boxes. One that allows the user to upload raw leader data in CSV format and a button the user can press to run the algorithm. Another that allows the user to upload a sorted leader data summary and raw first year data, both in CSV format and a button the user can press to run the algorithm.

Leader Algorithm (Tanushree Sharma)

Description

As a user, after I have uploaded the leader data, the algorithm should sort this data in accordance with even gender split, even program split and anti-request constraints and output a sorted CSV.

Requirements

The system shall sort leaders into groups ensuring an even gender split across all teams, an even program split across all teams (except Software, which should only be in 6 teams) and any anti-requests should be taken into consideration. Once split, the algorithm should return a CSV that includes the raw leader data, along with an additional column indicating team assignments.

First Year Algorithm (Sunrise Long)

Description

As a user I should have first year students sorted into even teams by program and gender with consideration for the leader teams they're assigned to.

Requirements

The system shall sort students into even groups. Each group should have a similar distribution of programs and genders. For a given list of leader groups, student groups must be assigned such that each student has a minimum of one leader in the same program.

Summary (Hannah Cranham)

Description

As a user I should be able to see a summary of the algorithm (for first year and leader) results to verify the output. For leaders this summary output shall act as an input to run the first-year algorithm.

Requirements

The system shall output a summary of the results upon completion of the algorithm. The user should be able to download the summary so that they are able to easily see the distribution of first years or leaders easily.

Deployment (Tanushree Sharma)

Description

As a user I should be able to deploy the application without the assistance of the development team.

Requirements

Given that I am a user, when I want to deploy the application then I shall be able to follow a clear set of instructions outlining the steps independently and without the assistance of the development team.

Sprint 4

Version 3, Algorithm (first year): Dynamic Parameters (Markus Hamman)

Description

As a user, I should be able to upload a CSV of leader's data and have the first years sorted by the number of teams that the leader data has been sorted into (no fixed number).

Requirements

The system shall sort first-year students into the correct number of teams as specified in the leader CSV file that is also uploaded. For the outputted list of first years, all should still be met for correct implementation.

Version 3, Algorithm (leader): Add up to 2 Anti-Requests (Sunrise Long)

Description

As a user, I should be able to upload a CSV of leaders, each with up to 2 anti-requests.

Requirements

The system shall sort leaders into groups with consideration for their anti-requests. In addition, to the previous sorting algorithm, if a leader specifies an anti-request(s), they will be assigned to a team that their anti-request(s) is not on.

Version 3, Algorithm (leader): Add Requests (Faraaz Zaiee)

Description

As a user, I should be able to upload a .csv file containing all the leaders, each with up to 1 request of another leader they would like to be on the same team as and have the leaders be sorted into teams based on their requests.

Requirements

The system shall sort leaders into groups with consideration for their requests. Leaders will be placed on a team that will satisfy their request.

Version 3, Algorithm (leader): Balance Returning Leaders in each group (Jaco Ye)

Description

As a user, I should be able to distribute a balanced number of returning leaders to each group. There should not be any group that has many more returning leaders than another.

Requirements

The system shall ensure that there is no group that has no returning leader and the returning leaders are distributed evenly. For example, if there are a total of 180 returning leaders, each group should have approximately 10 returning leaders.

Version 3, Algorithm (leader): Improve gender split in each group (Jaco Ye)

Description

As a user, I should be able to have an even distribution of leaders from one gender in each group. There should not be any group that has many more leaders from one gender than another.

Requirements

The system shall ensure that each group should have approximately the same amount of leaders from each gender. For example, if there are a total of 180 female leaders, each group should have approximately 10 female leaders.

Upload Page: Amount of Teams Input (Markus Hamman)

Description

As a user, I should be able to input the number of teams to split leaders and first-years into for orientation week.

Requirements

The system's front end shall provide the user with a prompt to enter the number of teams they would like to create. The system shall not continue until this is fulfilled. Lastly, the output should reflect the number of teams inputted.

Upload Page: Run-time notice (Markus Hamman)

Description

As a user, I should be notified of the approximate run-time of the first-year sorting algorithm to ensure it does not time out.

Requirements

The system's front end shall provide the user with a prompt to suggest the approximate run-time of the student sorting algorithm before a download of the file will be available.

Download Page (Faraaz Ziaee)

Description

As a user I should be able to download a .csv file containing either the first-year students or leaders sorted into teams.

Requirements

The system's front-end shall give the user the option to download a .csv file once the algorithm has completed running. The user will be able to download the .csv file with a click of a button. The downloaded .csv file will follow the correct naming convention to match the data it contains (i.e. first-year.csv or leader.csv).