**CS 201 Final Project Design**
**André J Guardia**
**A20453013**

---

## Describe the user interface. What are the menu options and how will the user use the application?

The user interface will be comprised of the following options:

- Edit Stations
    - Create Station
    - Modify Station
    - Remove Station
- Search Station
- Find Nearest Station
- Directions
- Exit

The user will be consistently prompted to select one of the options above. The following sentences will describe how the user will interact with the options above.

1. **When the user selects the option "Edit Stations"**, he will be prompted to select:
    a. Create Station: Upon selection the user will be prompted to enter the data to create a new station
    b. Modify Station: The user will be prompted to enter the name of the station he desires to modify. After entering this value, the app will print the characteristics of the station, and prompt the user to select the characteristic to change. The user will be prompted to confirm the changes to the station (Y/N). Once this information is entered, the user is prompted to exit to the main menu or make another change.
    c. Remove Station: The user will be prompted to enter the name of the station he desires to remove. After selecting said station, the user will be prompted to confirm removal of the station (Y/N).
2. **When the user selects the option "Search Station"**, he will be prompted to enter the name of the station he wants to find, the color of the train line said station is and if the station requires wheelchair accessibility. The program will then print the options and prompt the user to select his choice. Once the user enters the value of the desired station, the program will print more information on the selected station. The user is then prompted to exit to the main menu or search for another station.
3. **When the user selects the option "Find Nearest Station"**, the user will be prompted to enter his current address: Street, City, State, Country. The program will then return the

closest station to the user and print information on it. The user is prompted to exit to the main menu or to enter another address.

4. **When the user selects the option "Directions"**, he will be prompted to select the starting station and the ending station. The program will then print the directions and prompt the user to save the directions on a txt file (Y/N). After this the user is prompted to exit to the main menu or re enter the previous information.
5. **When the user selects the option "Exit"**, the user will be prompted to confirm the decision (Y/N). If the user says yes, the program is finished, if no, he returns to the main menu.

## Describe the programmers' tasks:

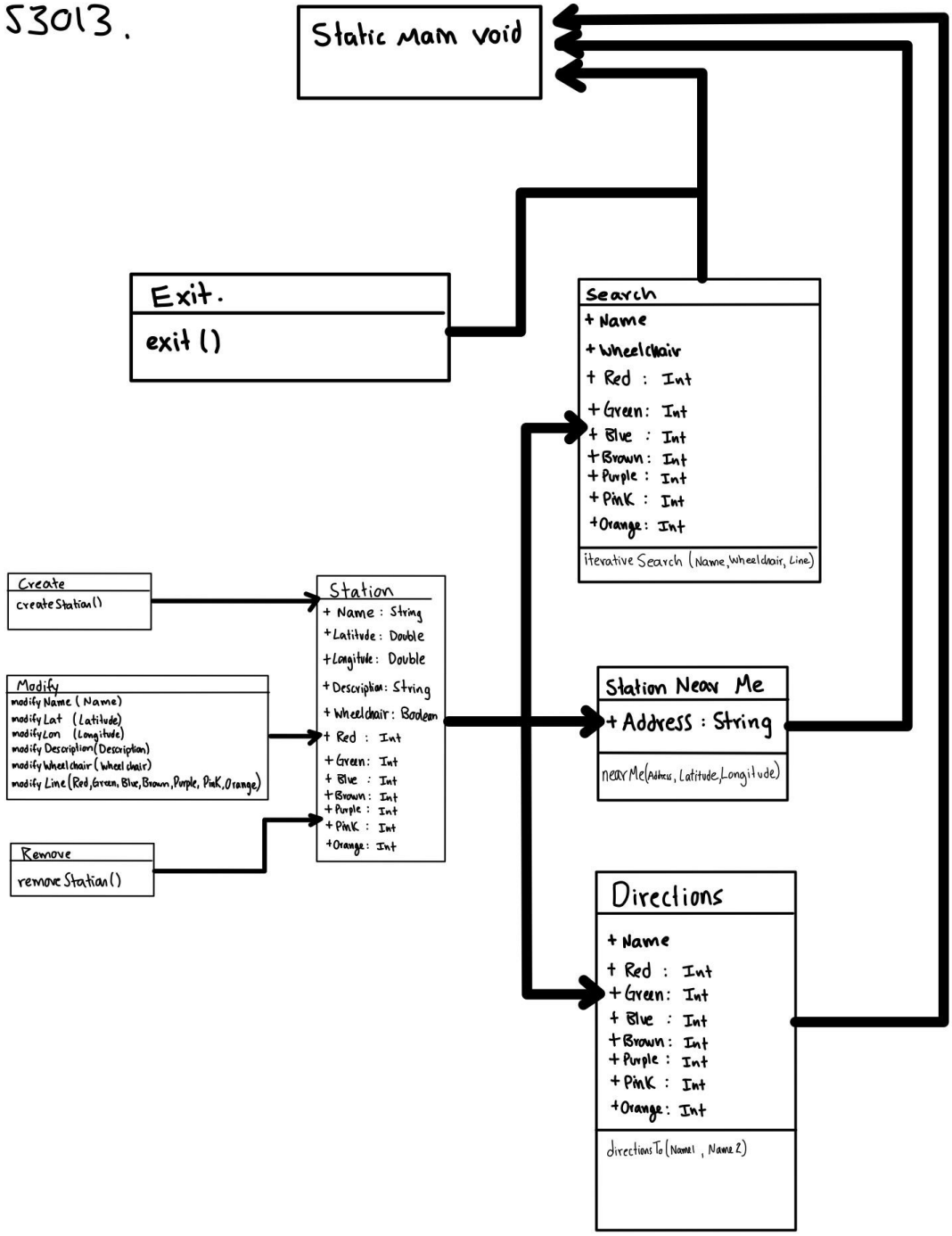
- Describe how you will read the input file.
    - The Data Processing class will be created to read the input file
    - This class will call the java Scanner class to load the file from the src/project directory
    - A while loop will be used to parse through the lines of the CSV File
- Describe how you will process the data from the input file.
    - The String.split() method will be used in a while loop to separate the values from the CSV File
- Describe how you will store the data (what objects will you store?)
    - The class "Station" will be initialized with the following objects:
        - Name (String)
        - Latitude (Double)
        - Longitude (Double)
        - Description (String)
        - Wheelchair (Boolean)
        - Red (Int)
        - Green (Int)
        - Blue (Int)
        - Brown (Int)
        - Purple (Int)
        - Pink (Int)
        - Orange (Int)
    - The split method from the String class will be used in a while loop inside the Data Processing class to separate the values from the CSV File and the "Station" class will be fed the respective data for each train station.

- How will you add/delete/modify data?
    - The class “Station” will have the following subclasses:
        - Remove (subclass)
            - **Methods:**
                - removeStation() *//This method removes the selected station*
        - Create(subclass)
            - **Methods:**
                - createStation(Name, Latitude, Longitude, Description, Wheelchair, Red, Green, Blue, Brown, Purple, Pink, Orange) *//This method will create another station instance in the class and rearrange the order of all other stations?*
        - Modify (subclass)
            - **Methods:**
                - modifyName(Name)
                - modifyLat(Latitude)
                - modifyLon(Longitude)
                - modifyDescription(Description)
                - modifyWheelchair(Wheelchair)
                - modifyLine(Red, Green, Blue, Brown, Purple, Pink, Orange)
- How will you search the data?
    - The Search class will be initialized with the following objects:
        - Name (Name of the station)
        - Wheelchair (True or False)
        - Line (Red, Green, Blue, etc…)
    - **Methods:**
    - iterativeSearch(Name, Wheelchair,Line)
        - The iterative search algorithm (as presented in the lectures using a while loop) will be used to search for stations that match the following criterion of search:
            - Name (Name of the station)
            - Wheelchair (True or False)
            - Line (Red, Green, Blue, etc…)
        - The program will then initialize a list called “Matches” and populate it with the train stations that match the criteria stated by the user
- List the classes you will need to implement your application.
    - Data Processing
        - This class will read the file, process the data from said file and save it to the Station class
    - Station

        - This class has all the objects that describe each train station
    - Remove Station
        - This class will remove a class from the list of stations
    - Create Station
        - This class will create a class from the list of stations
    - Modify Station
        - This class will modify each attribute of the selected station
    - Search Station
        - This class will search for a station based on the station's Name, Wheelchair and Line
    - Station Near Me
        - This class takes in the user's address, calls the mapbox geolocation api to convert the user's address to Latitude and Longitude, and then calculates the linear distance between the user's coordinates and all other Stations using a while loop to find  the one with the lowest distance parameter.
    - Directions
        - This class takes in the Attributes from two stations and lists the stations the user must take to get from station A to station B including connections.
    - Exit
        - This class terminates the program.

# UML Diagram

A20453013.



## Develop a test plan based on the above description

*How will you test that the expected outputs have been achieved for every menu option? Be sure this test plan is complete. Your test plan should minimally test each option in the menu-driven user interface for expected behavior as well as error-handling. Your test plan should be in a spreadsheet format (preferably Excel, CSV, or TSV).*

| Class | Section | Action to Test | Result |
|---|---|---|---|
| Data Processing | Read Input File | Read csv file and print to console | |
| Data Processing | String Split Method | String Split CSV File and print to console | |
| Data Processing | Storing Data | Store data from CSV File and print to console | |
| Remove Subclass | Remove Method | Call method and print Stations list to console | |
| Create Subclass | Create Method | Call method and add a Station, print to console | |
| Modify Subclass | Modify Name | Test Modification with three different Values | |
| Modify Subclass | Modify Latitude | Test Modification with three different Values (one Int, one double and a string) | |
| Modify Subclass | Modify Longitude | Test Modification with three different Values (one Int, one double and a string) | |
| Modify Subclass | Modify Description | Test Modification with three different Values (one Int, one double and a string) | |
| Modify Subclass | Modify Wheelchair | Test Modification with three different Values (one Int, one double and a string) | |
| Modify Subclass | Modify Line | Test Modification with three different Values (Changing different lines, float values and string values) | |
| Search Class | Search Algorithm | Test search with a uniquely named station | |
| Search Class | Search Algorithm | Test search with a repeated named station | |
| Search Class | Search Algorithm | Test search with an int | |
| Search Class | Search Algorithm | Test search with a name not in the list | |
| Station Near Me | Geolocation | Test Geolocation API with three different addresses in the city | |
| Station Near Me | Distance from Coordinates | Compare distance calculated with online distance calculator | |
| Station Near Me | Distance Minimizer | Try out three different addresses in the city and compare results with Google | |

| | | | |
|---|---|---|---|
| | | Maps | |
| Directions | Within a Line | Input Start and Finish Location for stations within the same line and compare plan with Google Maps | |
| Directions | One Connection | Input Start and Finish Location for stations with two different lines and compare plan with Google Maps | |
| Directions | Two Connections | Input Start and Finish Location for stations with three different lines and compare plan with Google Maps | |
| Exit | Exit Command | Select Exit from Main Menu | |