**Technical Manual for Web Based Population Clustering**

By Nathan Barthen, Connor Barthen, Cole Cassano

March 10, 2022

1. **Overview**

**This manual gives a detailed description on how to obtain the data used in this project (how to add more states), DBScan explanation, SMTP explanation, the UML diagrams, and how the system needs to be setup to run the program.**

1. **Outline**

**Pages: 2-6 --- Dataset Download Information**

**Pages: 6-8 --- Excel/Text File Format**

**Pages: 9-10 --- Location of these files in Eclipse**

**Pages: 11-12 --- DBScan Clustering Explanation/Format (& Charter School Info)**

**Pages: 13 --- SMTP Explanation**

**Pages: 14-18 --- Hook Points**

**Pages: 19-31 --- UML Diagrams**

**Required Programs**:

-IDE (Eclipse IDE EE – 2021-12)

- Microsoft Excel

- GitHub Account

**Spring Maven Information**:

- <https://spring.io/>

- Dependencies needed from Spring Maven:

- Thymeleaf, Spring Web, H2 Database, Spring Data JPA.

- To acquire the dependencies, go to <https://start.spring.io/> click ADD DEPENDENCIES and search for each one. Lastly, click generate.

**To download and setup the workspace:** *See Config Manual*

* ***Hook points are located in the Config Manual.***

**Dataset Download Information (*process for adding more states*):**

**Shapefile - Data files**:

* This link allows you to access TIGER/shapefiles for both school districts maps and county roadmaps. (<https://www.census.gov/cgi-bin/geo/shapefiles/index.php>)
* It contains the polygons of each school district (split up by state) with labels.
* **After downloading the file**, inside of the zipped folder, you will need to extract the .dbf and .shp files. These files should be named *state*.dbf **and** *state*.shp (Ex. pennsylvania.dbf). Both of these files will be placed inside of the workspace. More specifically, they are placed in states > state(ex. pennsylvania) folder.

Graphical user interface, text, application, chat or text message

Description automatically generated

Figure 1. Example image of the folder that contains the states’ files

**How to Download the Shapefile(s)**:

* Select your desired year, select School Districts, then click “Submit”.

Graphical user interface, text, application

Description automatically generated

Figure 2. Year and layer type Dropbox selection

* Under “Unified School District (current)” select your desired state, then click download.

Graphical user interface, text, application

Description automatically generated

Figure 3. State selection Dropbox and Dowload button

**School Information – Excel (csv) files**:

* Website that finds all schools within an area (state), displays all of the school’s information, and allows a csv download. (<https://nces.ed.gov/ccd/schoolsearch/index.asp>)
* **After downloading the file**, ensure the file is of type .csv (if it is not, convert it to csv format). This file should be named *State*.csv (Ex. Pennsylvania.csv). This file will be placed inside of the workspace. More specifically, they are placed in states > state(ex. pennsylvania) folder (*See figure one*).

**How to Download the Shapefile(s)**:

* Navigate to NCES website, select state, and search.

Graphical user interface, text, application

Description automatically generated

Figure 4. State selection Dropbox and Search button

* Next you can download the Excel file for the selected state.

Graphical user interface, text, application, table

Description automatically generated

Figure 5. Download Excel File Location

**Address File(s) – Excel / Text files**:

* This site is used to obtain the addresses of homes. (<https://batch.openaddresses.io/data>).

**How to Download the Address File(s)**:

* You can download the file for your desired region. Inside of the downloaded folder, the csv files are split up by state. Inside of each state the addresses are split up by county. We will need to combine these files into one .txt file (by state).
* **After downloading the file**, they county csv files (which contains the addresses) will need to be merged into one large .txt file (.csv has a max row limit of ~1 million, so it must be a .txt file). The final .txt file that contains all of the files for the given state will be placed in the in states > state(ex. pennsylvania) folder inside of a compressed folder (*See figure one*).

Graphical user interface, table

Description automatically generated with medium confidence

Figure 6. Dowload buttons for Batch Files

**County / School District Data – Excel (csv) files**:

* This link is used to get the county and school district names. (<https://www.greatschools.org/schools/districts/Pennsylvania/PA/>)
* The school information dataset does not specify what county the school / school district is in. This link allows you to select a state and get all of the school district names with their respective county. You will need to highlight all of the data, copy and paste it into the CountrySchoolsSounty.xlsx file (*See figure one*). Additionally, in the fourth column, you will need to add the *State* name since this file will contains data for all of the states.

Graphical user interface, application

Description automatically generated

Figure 7. Schools districts by state Dropdown box

**Excel/Text File Format (*process for adding more states*):**

**School Information Excel File Format:**

* The format of the excel files is critical for the program to run correctly.
* Below are examples and in-depth details on how to set up each file.
* The first file “Pennsylvania.csv” holds the bulk of the information and should be formatted as shown below (Figure 7 and 8 are the same file but could not fit in one screenshot). The information should be ordered from left the right and in the same order. Some of this information is not used in this project but I included it all to minimize confusion and to be comprehensive.

Graphical user interface, application, Word

Description automatically generated

Figure 8. Naming of columns in the Excel File

Graphical user interface, application, table

Description automatically generated

Figure 9. Naming of columns in the Excel File

**County/District Information Excel File Format:**

* The “CountrySchoolsCounties.xlsx” file holds the District Name, City, County name, and State. Figure 9 shows the required layout for each attribute.

Graphical user interface, application, table

Description automatically generated

Figure 10. Column name setup for School Counties

* The last files only hold all of the county information along with the longitude and latitude values. From left to right the information should have the longitude, latitude, number, street, city, district, region, postcode, ID, and HASH number.

**Address Information Text File Format:**

* This is the format of the individual *countyName*.csv file. However, you must combine all these files into one .txt file. We also removed every column except for LON and LAT (others are missing a lot of data and are not needed).

Graphical user interface, application, table, Excel

Description automatically generated

Figure 11. Column name setup for the county information

-This is an example what the .txt file should look like.

Graphical user interface, text

Description automatically generated

Figure 12 Condensed example of the final .txt file of the addresses

**Location of these files in Eclipse (*process for adding more states*):**

* This section will detail the location of these excel files as well as the shapefile being used.
* The “Pennsylvania.xlsx” file is located on the left-hand side in ‘states’ > ‘pennsylvania’. It is inside the highlighted ‘pennsylvania’ folder.
  + This contains specific information for each school.
* The .shp and .dbf files are also located in ‘states’ > ‘pennsylvania’.
  + Used to get district polygon and select addresses
* The files containing the addresses are also located in ‘states’ > ‘pennsylvania’.
  + Graphical user interface, text, application

    Description automatically generatedThese addresses are currently split up by counties.

Graphical user interface, text, application, chat or text message

Description automatically generated

Figure 13. Location of file containing the addresses

* The “CountrySchoolsCounties.xlsx” file is also located on the left-hand side and is not in any folder.
  + This contains all of the county names and with the district names.

Graphical user interface, text, application

Description automatically generated

Figure 14. Location of school counties Excel File

**DBScan Clustering Explanation/Format:**

**Resources**:

- Incorrect example (but useful): <https://www.dataonfocus.com/dbscan-java-code/>

- \*Pseudocode example: [pseudocode example](https://www.researchgate.net/publication/325059373/figure/fig2/AS:624653831790593@1525940487951/Pseudocode-of-the-DBSCAN-algorithm.png)

**DBScan Clustering Algorithm** (the main classes)**:**

* **DataPoint.java** – POJO of a single address (DBScan.java will accept a List of DataPoint(s) for clustering)
* **Cluster.java** – POJO of a single cluster (DBScan.java will create a List of Cluster(s) for when executing). Each cluster contains a list of addresses (List of DataPoints) and importantly a calculateCentroid() method. After the clusters are created, it will be used to calculate each cluster’s centroid (useful when assigning clusters to schools in the district).
* **DBScan.java – This is what generates the list of clustering**.

**Parameters**: diameter, minimum cluster size.

Diameter: This is the diameter for the clusters generated. The value of

diameter is in terms of degrees (of longitude). For example, if you passed 1 (one degree), the diameter would be ~54 miles. We found 2 miles (~.029) to be a good baseline diameter (the user can choose diameter on the webpage).

Minimum Cluster Size: This parameter specifies how many addresses

must be found (inside of the given diameter) for a cluster to be made. If it does not find enough addresses, the cluster will not be made. This is useful for removing noise from the clusters. For our minimum cluster size, we made it so that it was %2 of all of the address in the given district (datapoints.size() \* .02). In summary, if a cluster does not contain at least 2% of the addresses in the district, it will be excluded as noise.

* + To run DBScan you must first create an instance passing the diameter, and minClusterSize in the constructor. Then you will give DBScan your list of addresses (that are inside of the given school district) (.setPoints(ListOfDataPoints);). Next, you will call .cluster() which generates the large clusters for the district. After you get these large clusters (list of Cluster.java) (.getClusters();).
  + After you have the List of large clusters, you must loop through every cluster and call calculateCentriod(). This is necessary since it will be needed for assigning clusters to school in the district and randomly selecting address from each cluster.

**Assigning Clusters to School(s) Clustering Algorithm:**

* After we have the list of clusters, we can use this for randomly selecting address to use as the student’s houses for school bus routing. However, there are many different combinations of schools that can be passed. For example, they may only pass one elementary school (they want each elementary school routed separately) or they may pass all of the elementary schools or just 2 of 3 elementary school. I will explain how these combinations are handled (**ClustersForSchool.java** and **SelectAddressesByCluster.java**).

Selecting Addresses based on which schools were passed:

**High School** – it will randomly select address from all the clusters proportional to the size of each cluster.

**Middle School** – Similarly, it will randomly select address from all the clusters proportional to the size of each cluster.

**High School & Middle School** – Again, it will combine all the addresses the middle and high schools, and it will randomly select address from all the clusters proportional to the size of each cluster.

*\*These next examples will be explained as if the district contains 3 elementary school*

**One-All Elementary Schools Passed** – it will take all 3 of the elementary school, use the schools’ lon/lat (*which must be* *acquired*) and the clusters centroid to assigns clusters to the school they are closest to*. For example, if there are 15 total clusters in the district, Elementary School #1 may only be assigned 3 clusters. School #2 may get 7 and School #3 may get 5 clusters*. After the large clusters are split up, it will then randomly select addresses from the assigned clusters proportional to the size of the cluster.

**Middle/High & Elementary Schools** – If they want all the schools to be bussed together, it will just use the large cluster list, add of the students together, and randomly select address from each cluster proportional to their size.

**Charter School Additions:**

* Our dataset lists charter schools as their own district, but we want them to be placed in the actual school district they are located in.

**Solution**: We removed charter schools from being displayed in the district page. Instead, when they select on an ‘actual’ school district, it will locate all the schools in that district THEN look at all the charter schools in that county. If a charter school’s address (lon/lat) is located inside of the district’s polygon, it will also be added to that district’s school list.

***Note****: Our dataset did not include private schools, so we only addressed charter schools (though the process to add private schools would be a similar process). Alternatively, any missing school could also be added using the ‘Add School’ button on the html page*.

**SMTP Example/Explanation:**

**Resources Ex**: <https://www.journaldev.com/2532/javamail-example-send-mail-in-java-smtp>

**Sending An Email Through SMTP:**

**- EmailFile.java - creates the email that will be sent to the specified email**

**Parameters:** session, toEmail, subject, body, rountingName

session: this parameter is of data type Session. It holds the information about the properties and the authentication information about the email.

toEmail: This parameter is of type String that holds the email address of the email that will be the recipient of the message created.

subject: This parameter holds the String that will be the subject line of the email sent.

body: This parameter holds the String for the main body of the email

rountingName: This parameter holds the String for the name of the routing file that will be attached to the email.

* + In the EmailFile class an instance of MimeMessage is created with the session parameter.
  + Then the method (.setFrom(new InternetAddress("routingfileemail@gmail.com", "NoReply-JD"));) is called.
    - We created an email that will send the email with the routing file.
      * Username: [routingfileemail@gmail.com](mailto:routingfileemail@gmail.com)
      * Password: RoutingPass123
    - This should be changed to whatever email you want to use when you use this code.
  + Similarly you use the .setRecipients(Message.RecipientType.TO, InternetAddress.parse(toEmail, false)); ) with the toEmail parameter.
  + The class then adds a new MimeBodyPart() and set it equal to the body parameter
  + Then the routingName is added to the MimeBodyPart()
  + Finally the method Transport.send(msg);
    - Where msg is the instance of MimeMessage that has had the MimeBodyPart() added to it.

- In the controller the variables fromEmail and password are made.

* + These variables are used for authentication and logging into the email set up previously.
    - For the authentication to work, two step authentication was needed to be set up in the security of the email created.
      * And the “app password” was used to sign from the code
        + App password: lazzoslzlprwcgxc

**Hook Points:**

**Addresses Folder/File - Hook Point**:

* The folder containing the file of all addresses is located under the “states” folder 🡪 “pennsylvania” folder 🡪 “Addr-pennsylvania.zip” folder. This contains a text file that holds all of the longitude and latitude locations.

Text

Description automatically generated with low confidence

Figure 15. House address location.

**Dr. Thangiah Routing Algorithm - Hook Point**:

* The hook point for the bus routing code is in the “edu.sru.group1.proj.controller” package and in the “StateCountyController.java” file.

Text

Description automatically generated with low confidence

Figure 16. Controller Location

Text

Description automatically generated with low confidence

Figure 17. StateCountyController Location

* Once in the “StateCountyController.java” file the hook point is located near the bottom (Line 638).

Graphical user interface, text, application

Description automatically generated

Figure 18. Code location for Routing Hook Point

**Adding new clustering method - Hook Point**:

Text

Description automatically generated with low confidence

Figure 19. Controller Location

Text

Description automatically generated with low confidence

Figure 20. StateCountyController Location.

* Once in the “StateCountyController.java” file you will see the hook point

(Line ~503).

Text

Description automatically generated

Figure 21. Code hook point for adding new clustering method

* The centroid is calculated using the calculateCentroid method. This method is located in the “Cluster.java” file under the “edu.sru.group1.proj.dbscan” package.

A picture containing text

Description automatically generated

Figure 22. Cluster.java file location.

* The method itself is located on line 74.

Text

Description automatically generated

Figure 23. Code location for calculating the centroid

**Changing the DBScan Diameter – Hook Point**:

* The DBScan diameter is calculated using the option on the website and the html file “district-info.html”. This file is located under the “src/main/resources” folder 🡪 “templates” folder.

Text

Description automatically generated with medium confidence

Figure 24. district-info.html file location

* The code itself is located on lines 33-46.

Text, timeline

Description automatically generated

Figure 25. Code location for editing the diameter in miles

* The diameter is calculated by dividing the miles by 69 (Example: .25 / 69).

**Batch Processing – Hook Point:**

* The Batch processing is found under the “src/main/java” folder 🡪 “edu.sru.group1.proj.batch” package 🡪 “BatchProcessing.java” file.
* This hook point can be used to edit what schools are clustered together, what counties you want to be processed and what districts you want to be processed.

Text, application

Description automatically generated with medium confidence

Figure 26. BatchProcessing.java file location

* The code can be found in the file on lines 31 – 46.

Text

Description automatically generated

Figure 27. Code location for the Batch Processing

**UML Diagrams:**

* UML diagrams to better show the functionalities of the program and the connections between the functions and the actors

**Use Case Diagram**

* Here is a use case diagram for the project
  + The actors will be the user, the Tomcat server, and the database of the users

Diagram

Description automatically generatedFigure 28. Case Diagram for the Project

**Class Diagrams**

* Figure 29 is a class diagram for the dbscan package
  + The dbscan package is used to generate clusters of addresses that are used to generate random addresses that are used to route the buses

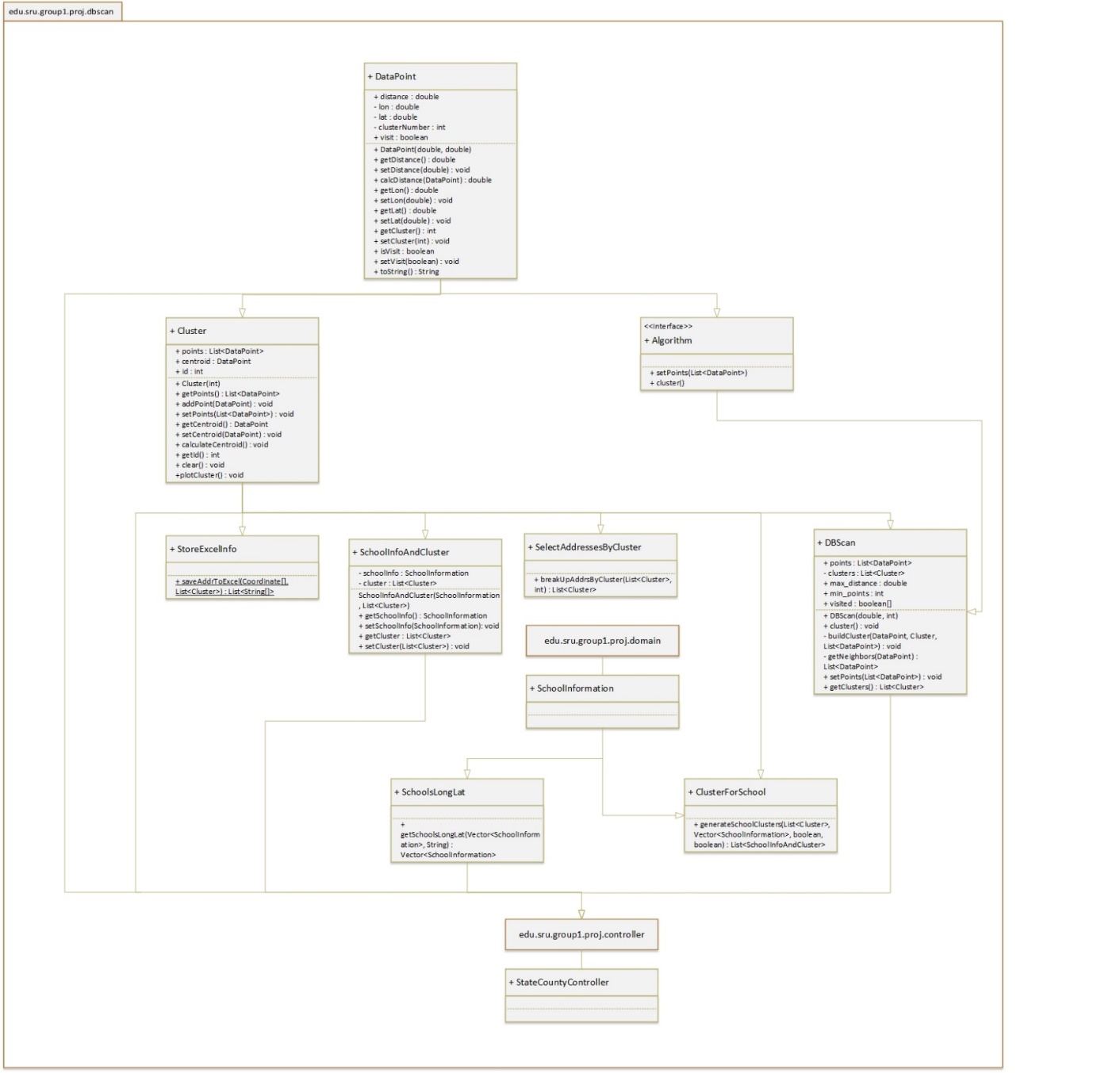


Figure 29. Class diagrams for the Project

* Figure 30 is the class diagram for the shapefile package
  + The shapefile package is used for converting the shapefile into a display on the webpage

Table

Description automatically generated with medium confidenceFigure 30. Class diagram for the Project

* Figure 31 is of the domain package
  + The domain package is used to store, change, and save information from datasets and the user

Diagram

Description automatically generated

Figure 31. Class Diagram for the Project

* Figure 32 is of the controller package
  + The controller is used to connect all other packages into one and run the program

A picture containing table

Description automatically generated

Figure 32 Class Diagram for the Project

Diagram

Description automatically generated

Figure 33. Sequence Diagram for DisplayDistrictInfo

Diagram

Description automatically generated with medium confidence

Figure 34. Sequence Diagram for the GeoJson File

Graphical user interface

Description automatically generated with low confidence

Figure 35. Sequence Diagram for DisplayClustering

Diagram

Description automatically generated

Figure 36. Activity Diagram for clusterSchools

Diagram

Description automatically generated

Figure 27. Activity Diagram for addSchool

A picture containing text, sign

Description automatically generated

Figure 38. Activity Diagram for selectDistrict

Diagram

Description automatically generated

Figure 29. State Diagram for selectDistricts

Diagram

Description automatically generated

Figure 40. State Diagram for displayClustering

Diagram

Description automatically generated

Figure 41. State Diagram for addSchool