

Elaboration Iteration 2

Report

Updated: 12 August 2020

Prepared By:

Nehaben Kapadiya: kapadiyn@uwindsor.ca

Chandni Desai: desai32@uwindsor.ca

Deepti Bhatia: bhati122@uwindsor.ca

Contents

1. Objective
 - 1.1. Key Experiments & Results
2. Keywords
3. Acknowledgements
 - 3.1. Nehaben Kapadiya
 - 3.2. Chandni Desai
 - 3.3. Deepti Bhatia
4. Introduction
5. Related Work
6. Approach
 - 6.1. Architecture
 - 6.2. Algorithm for OpenDataCSVReader class
 - 6.3. Algorithm for Graphic viewer class
 - 6.4. Algorithm for driver class
 - 6.5. Class Diagram prototype
7. Experimental setup or Demonstration
8. Discussion
9. Conclusion
10. Future Work
11. References

1. **Objectives:**

- Open data represents the idea that certain set of data should be freely provided for public. There are various open data access portals in existence today. The aim of our project is to make the dynamic raw open data available on the city's website more user friendly by making a real time visual analysis of the necessary facts, instead of expecting the user to dig into the data & make analysis.
- Users can explore the updated information from open portal sitting at home. Exploring opportunities, Improving efficiency.
- With the help of open data, user can benefit both economically as well as socially. Public can make their decisions wisely for business purpose which helps in growth and development of society.

1.1 Key Experiments & Results:

In the Elaboration iteration 1 phase, we worked on a prototype which will read the static csv files downloaded from the city's open data website & provide a user interface with various options to get specific information from the huge raw data. For example, a set of open data about road maintenance is read by the software. User selects the option to know which mode was widely used by city residents to report road maintenance requests on city's website, e-mail, web or phone. The data is analyzed & user is provided information about the mode most used.

In the Elaboration iteration 2, we worked on improving the initial prototype to present the requirement of the user in a visual format using charts. This will enable the user to have a quick glimpse of the chart & analyze the data.

2. **Keywords**

Open Data, CSV, Data Visualization

3. **Acknowledgements:**

3.1. *Nehaben*

- Analyzed open data on different data portal such as Toronto, Edmonton, as well as CKAN to see format of data availability as well as usage of data in routine life.
- Prepared 3 artifacts in inception phase report after data referral and decide to work on crime data. In Inception report, worked on vision and business case of project , User case model and Supplementary specification.

- Elaboration - Iteration I provide contribution with some of part in coding such as print csv data on console as well as provide data according to street.
- Prepare report for all Elaboration - Iteration phase I
- Created repository on GitHub
- For final phase created code for graphical representation of chart
- Provided contribution with report

3.2. Chandni

- For Inception report worked on Glossary, Risk list and Risk management plan, Prototype and proof of concepts. Created Prototype.
- Worked on Unit testing and UML diagram creation. Patch test of github and contributed in editing project report on latex.
- Contributed in updating Tester code and on architecture and report editing and presentation.

3.3. Deepti

- Worked on the prototype code for both iterations
- Worked on Report Creation for inception & both iterations
- Updates on Project Task Board
- Worked on Final Project Presentation

4. Introduction

- Problem statement: To provide visual data analysis for users of dynamic open data.
- Motivation: Raw open data is hard to interpret for a lay man. If presented to a city resident in a visual interpretation, it becomes easy to understand.
- Background: To present easy to understand representation of open data for users
- Objectives:
 - i. To provide a user-friendly data analysis of raw open data.
 - ii. To make use of dynamic open data sets to visualizing data
- Report breakdown description: Refer to Contents

5. Related Work

City of Toronto open data portal has been used as a reference to develop the prototype. In this portal, for every data set there are various options for data users like data preview, data features, download as a csv file, data visualization, & API for developers. This approach is very user friendly as all options are available for users of data sets.

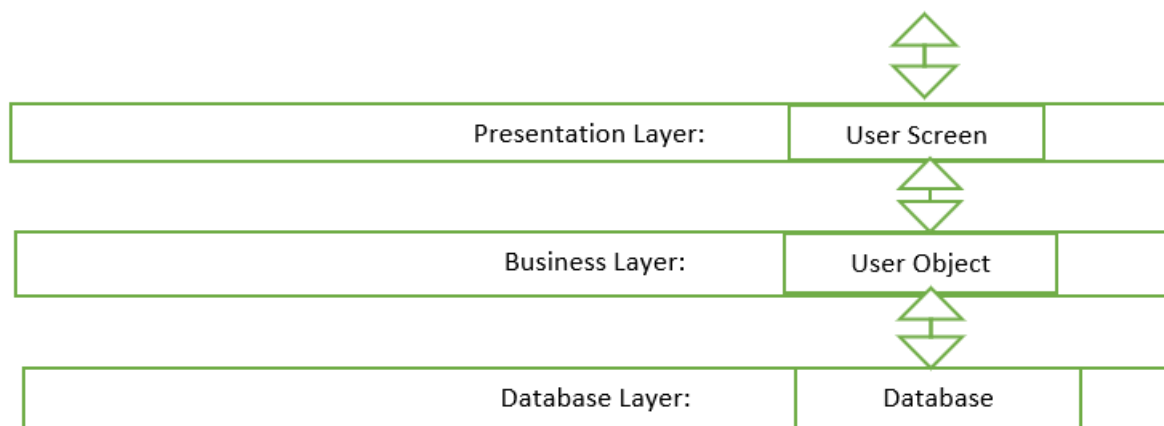
City of Toronto open data portals also has an Add-on which can be used with Google spreadsheets. This enables users to just enable the add-on to their google sheets & data can be accessed directly without a need to access the open data portal.[4]

6. Approach

This project was undertaken to enable a better user experience for the users of city's open data portal. Currently, the open data portal is only providing raw data to its users which they need to download & then do analysis on their own means. This might be easy for some but mostly people are not very conversant with handling data & thus are not able to use it in the best manner.

So the initial design of the prototype was built with the intention of preparing an analysis for the raw data. The first step was to prepare a prototype that would provide two major functionalities, first reading the CSV files, second, providing a user interface with various options to analyze the data.

6.1. Architecture



- This architecture is separated into 3 layers Presentation (UI), domain logic (business logic), and data access.
- Once the User screen receives a request to fetch information, it then forwards that request onto the Business layer.
- The User object in the business layer is responsible for aggregating all the information available and needed by the User request from database.
- This is an effective form of modularization as it allows to reduce the scope of attention and think about the three topics relatively independently.
- While working on Business logic layer UI can be completely ignored. It is easier to focus on 1 layer at a time (like refactoring's two hats).

6.2. Following algorithm was developed for the OpenDataCSVReader class:

- Create String ArrayList of ArrayList
- Create constructor of OpenDataCSVReader class with the parameter of a filename with type String. Create a new file object & the a scanner object to read the file. Then create a string array & add the contents of each row of the csv file to the string array, using a while loop. Close the scanner.
- Add public method to check number of rows.
- Add public method to check number of fields in a row.
- Add public method to get a field.

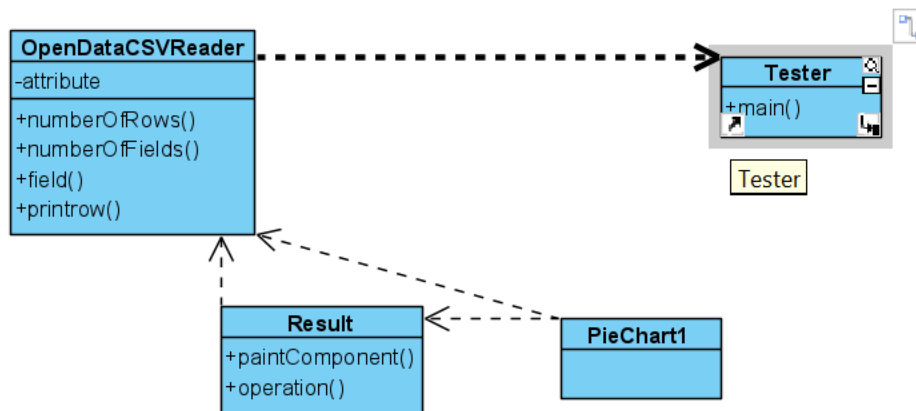
6.3 Following is graphical view class[1]

- Create method paintComponent() method to draw graphics on panel to take graphic object as parameter.
- Find center of panel for x and y axis using function and get radius.
- Create 3 labels to create sections.
- Create 3 different arcs using fill arc will all parameters x axis, y axis, width , height and start and end angle.
- Call setColor() is method to add font color and arc.

6.1. Following algorithm was developed for the driver class, Tester:

- In the main method, create an instance of the OpenDataCSVReader class & take the parameter as the csv file downloaded from the city's open data file.
- A menu is provided to the city resident using switch, where he can select if he wants to know how the road maintenance request was received-phone, web or e-mail.
- User can get to know the total number of requests from any of the above mentioned modes or the total number of requests received till date.
- User can get graphical view of data using part chart

6.2. Following is the Class Diagram for the prototype:



7. Experimental setup or Demonstration

- User selects from a menu to analyze the open data regarding road maintenance requests received by the city, year till date.

The menu options are as follows:

- Number of requests received by Phone
 - Number of requests received by Web Intake
 - Number of requests received by Email
 - Select a record by Street name
 - Create a visual representation of how requests were received
 - Print all records
- If User selects option 1, the total number of phone requests are displayed. Similarly, option 2 & 3 display web & email requests.

```

7  public static void main(String[] args) {
8
9      OpenDataCSVReader reader1 = new OpenDataCSVReader("C:\\Users\\dips9\\eclipse-workspace\\COMP 3220\\src\\finalPrototype\\R
10
11      System.out.println("Total number of road Maintenance service request YTD: " + reader1.numberOfRows());
12
13      System.out.println("Check how the request was received: Select 1 for Phone, 2 for Web Intake, 3 for E-Mail, 4 for selecti
14      Scanner in = new Scanner(System.in);
15      int selection = in.nextInt();
16
17      switch(selection) {
18          case 1:
19              int count = 0;
20              for(int i=1; i<reader1.numberOfRows(); i++) {
21                  String o = "Phone";
22                  if(reader1.field(i, 3).equals(o)) {
23                      count++;
24                  }
25              }
26              System.out.println("Total Phone requests: " + count);
27              break;
28          case 2:
29              int count1 = 0;
30              for(int i=1; i<reader1.numberOfRows(); i++) {
  
```

Message Console

```

<terminated> Tester (4) [Java Application] C:\Program Files\Java\jdk-13.0.1\bin\javaw.exe (Aug 11, 2020, 12:34:36 PM - 12:35:11 PM)
Total number of road Maintenance service request YTD: 85
Check how the request was received: Select 1 for Phone, 2 for Web Intake, 3 for E-Mail, 4 for selecting record by street name, 5 for cr
1
Total Phone requests: 64
  
```

```

19      int count = 0;
20      for(int i=1; i<reader1.numberOfRows(); i++) {
21          String o = "Phone";
22          if(reader1.field(i, 3).equals(o)) {
23              count++;
24          }
25      }
26      System.out.println("Total Phone requests: " + count);
27      break;
28      case 2:
29          int count1 = 0;
30          for(int i=1; i<reader1.numberOfRows(); i++) {
31              String o = "Web Intake";
32              if(reader1.field(i, 3).equals(o)) {
33                  count1++;
34              }
35          }
36          System.out.println("Total Web requests: " + count1);
37          break;
38      case 3:
39          int count2 = 0;
40          for(int i=1; i<reader1.numberOfRows(); i++) {
41              String o = "E-Mail";
42              if(reader1.field(i, 3).equals(o)) {
43                  count2++;
44              }
45          }
46          System.out.println("Total e-mail requests: " + count2);
47          break;
48      case 4:
49          Scanner scan = new Scanner (System.in);
50          System.out.println("Please enter street name for search record: ");
51          String streetname = scan.next();
52          int count3 = 0;
53          for (i = 0; i< reader1.numberOfRows(); i++) {
54              if (reader1.field(i ,6).equals(streetname)) {
55                  count3++;
56              }
57          }
58          System.out.println("Total number of request for entered street is : " + count3);
59          break;
60      case 5:
61          PieChart1 frame = new PieChart1();
62          frame.setSize(300, 300);
63          frame.setLocationRelativeTo(null); // Center the frame
64      }
65      }
66      }
67      }

```

Message Console

terminated> Tester (4) [Java Application] C:\Program Files\Java\jdk-13.0.1\bin\javaw.exe (Aug 11, 2020, 12:36:56 PM - 12:37:00 PM)

Total number of road Maintenance service request YTD: 85

Check how the request was received: Select 1 for Phone, 2 for Web Intake, 3 for E-Mail, 4 for selecting record by street name, 5 for cr

Total Web requests: 11

```

29      int count1 = 0;
30      for(int i=1; i<reader1.numberOfRows(); i++) {
31          String o = "Web Intake";
32          if(reader1.field(i, 3).equals(o)) {
33              count1++;
34          }
35      }
36      System.out.println("Total Web requests: " + count1);
37      break;
38      case 3:
39          int count2 = 0;
40          for(int i=1; i<reader1.numberOfRows(); i++) {
41              String o = "E-Mail";
42              if(reader1.field(i, 3).equals(o)) {
43                  count2++;
44              }
45          }
46          System.out.println("Total e-mail requests: " + count2);
47          break;
48      case 4:
49          Scanner scan = new Scanner (System.in);
50          System.out.println("Please enter street name for search record: ");
51          String streetname = scan.next();
52          int count3 = 0;
53          for (i = 0; i< reader1.numberOfRows(); i++) {
54              if (reader1.field(i ,6).equals(streetname)) {
55                  count3++;
56              }
57          }
58          System.out.println("Total number of request for entered street is : " + count3);
59          break;
60      case 5:
61          PieChart1 frame = new PieChart1();
62          frame.setSize(300, 300);
63          frame.setLocationRelativeTo(null); // Center the frame
64      }
65      }
66      }
67      }

```

Message Console

terminated> Tester (4) [Java Application] C:\Program Files\Java\jdk-13.0.1\bin\javaw.exe (Aug 11, 2020, 12:38:06 PM - 12:38:09 PM)

Total number of road Maintenance service request YTD: 85

Check how the request was received: Select 1 for Phone, 2 for Web Intake, 3 for E-Mail, 4 for selecting record by street name, 5 for cr

Total e-mail requests: 9

- User selects option 4 to view records for a particular street name

```

44      }
45      }
46      System.out.println("Total e-mail requests: " + count2);
47      break;
48      case 4:
49          Scanner scan = new Scanner (System.in);
50          System.out.println("Please enter street name for search record: ");
51          String streetname = scan.next();
52          int count3 = 0;
53          for (i = 0; i< reader1.numberOfRows(); i++) {
54              if (reader1.field(i ,6).equals(streetname)) {
55                  count3++;
56              }
57          }
58          System.out.println("Total number of request for entered street is : " + count3);
59          break;
60      case 5:
61          PieChart1 frame = new PieChart1();
62          frame.setSize(300, 300);
63          frame.setLocationRelativeTo(null); // Center the frame
64      }
65      }
66      }
67      }

```

Message Console

terminated> Tester (4) [Java Application] C:\Program Files\Java\jdk-13.0.1\bin\javaw.exe (Aug 11, 2020, 12:41:09 PM - 12:41:50 PM)

Total number of road Maintenance service request YTD: 85

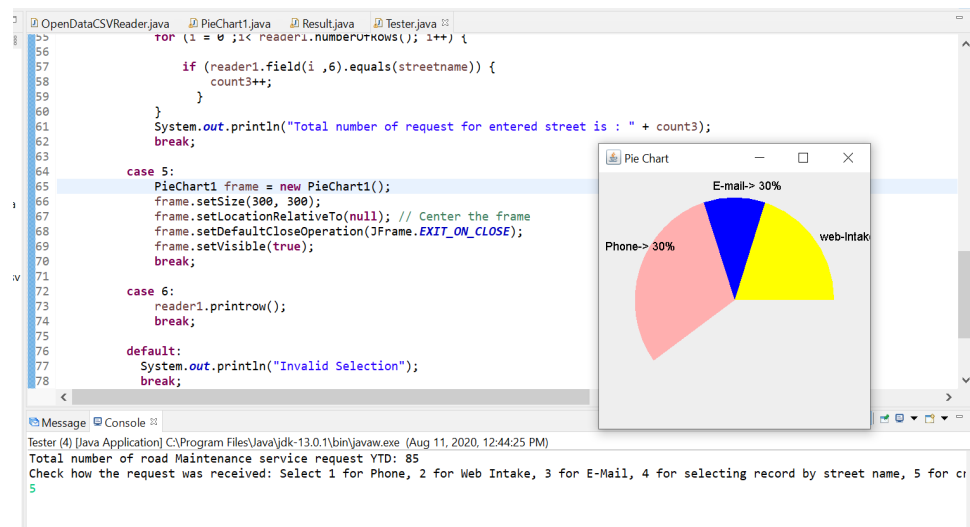
Check how the request was received: Select 1 for Phone, 2 for Web Intake, 3 for E-Mail, 4 for selecting record by street name, 5 for cr

Please enter street name for search record:

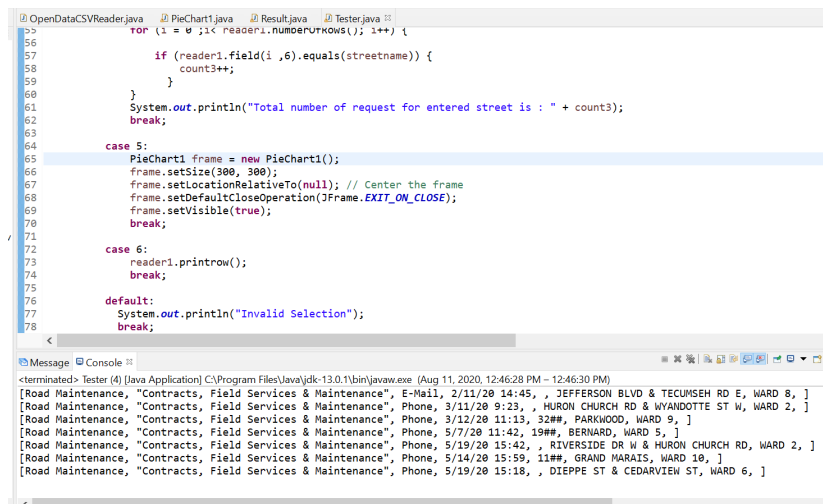
TECUMSEH

Total number of request for entered street is : 3

- User selects visual representation of mode of requests



- User selects option 6 to print all records



8. Discussion

The initial prototype was designed to fetch data from the open data portal as a CSV file & then analyzed to present output to the user in a non-visual format. Subsequently, in the present elaboration iteration, an option to present visual output was added to the prototype using demo data from the csv file. The user can look at a pie chart & in one glance get to know the results. In the subsequent iteration, the improvisation plan for the prototype includes fetching the dynamic open data from the portal directly & then using the same to analyze the data. This has the benefit of using the most up to date data. Also, the demo data for the charts will be replaced with fetching the dynamic data from the portal.

Junit testing has been completed for the current prototype.

9. Conclusion

In the current iteration of the project, we were able to achieve a visual functionality to present data to the user. However, the data used for the chart is currently a demo data & not directly fetched from the CSV file available on the open data portal. Design of this code has low coupling as it does not require change for final phase. It just requires additional work update for graphical presentation.

10. Future work

In the next iteration, we plan to incorporate two things that we were unable to cover in the current prototype-first using dynamic CSV files as a feed for the entire software, second, incorporating CSV file reader code for chart creation.

11. References:

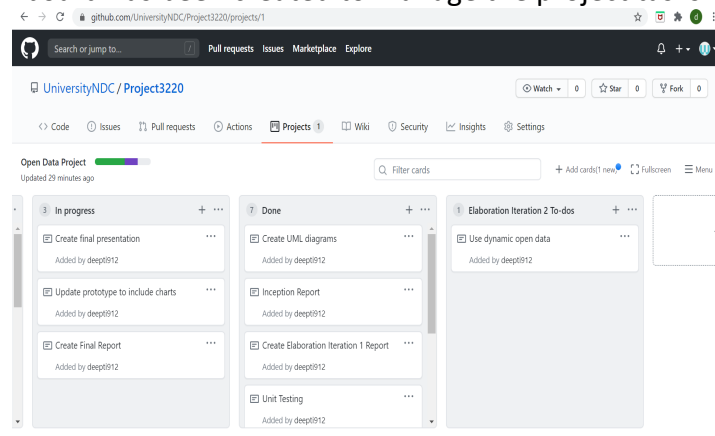
- [1] <https://www.tutorialspoint.com/how-can-we-implement-the-paintcomponent-method-of-a-jpanel-in-java>
- [2] <https://data.edmonton.ca>
- [3] <https://ckan.org>
- [4] <https://open.toronto.ca/>

12. Appendix - A Group Work :

<https://github.com/UniversityNDC/Project3220/projects/1>

Github Projects has been used for collaboration & project management.

12.1 A Kanban task board has been created to manage the project tasks.



12.2. GitHub main page with Project name & created repository

The screenshot shows the GitHub main page for the repository **UniversityNDC / Project3220**. The page includes navigation tabs for Code, Issues, Pull requests, Actions, Projects, Wiki, Security, Insights, and Settings. Below the navigation bar, there are buttons for 'Go to file', 'Add file', and 'Code'. The main content area displays a list of files and their commit history:

File	Commit Message	Commit Time
nsk2106 Update Result.java	213398d 2 hours ago	24 commits
docs	Create index.md	26 days ago
3220projectUML.ucls	UML Class diagram	26 days ago
Community_Centres.csv	Adding csv files	26 days ago
Elaboration_iteration_1.pdf	Add files via upload	26 days ago
GraphicView.java	Add files via upload	12 hours ago
Inception_Report.pdf	Inception report added	26 days ago
OpenDataCSVReader.java	Update OpenDataCSVReader.java	26 days ago
OpenDataCSVReaderTest.java	Update OpenDataCSVReaderTest.java	26 days ago
README.md	Initial commit	27 days ago
Result.java	Update Result.java	2 hours ago

On the right side, there are sections for 'About', 'Releases', 'Packages', and 'Contributors'. The 'About' section states 'No description, website, or topics provided.' The 'Releases' section states 'No releases published' and 'Create a new release'. The 'Packages' section states 'No packages published' and 'Publish your first package'. The 'Contributors' section shows 3 contributors: deepti912, deepti912, and deepti912.

12.3. Commit created by each member throughout the project

The screenshot shows the GitHub commit history for the repository **UniversityNDC / Project3220**. The page displays a list of commits, grouped by date. The commits are as follows:

Commit Message	Commit Time	Commit Hash
Update Result.java	nsk2106 committed 2 hours ago	213398d
Add files via upload	Chandni380 committed 10 hours ago	79b72be
Add files via upload	nsk2106 committed 12 hours ago	54c49f7
Add files via upload	nsk2106 committed 26 days ago	80ed739
Create index.md	nsk2106 committed 26 days ago	f8ec145
Inception report added	nsk2106 committed 26 days ago	45143d5
Update OpenDataCSVReaderTest.java	deepti912 committed 26 days ago	21c7ab6
Update Tester.java	deepti912 committed 26 days ago	c186c54
Adding csv files	deepti912 committed 26 days ago	73388a1
Update Tester.java	nsk2106 committed 26 days ago	d851864
Merge branch 'master' of https://github.com/UniversityNDC/Project3220	deepti912 committed 26 days ago	a43ba53
Add files via upload	Chandni380 committed 26 days ago	86a7a18
tester modified	nsk committed 26 days ago	1bd64e2
added by nehaben	nsk committed 26 days ago	75854ab
Add files via upload	deepti912 committed 26 days ago	451bc44
Corrected class name	deepti912 committed 27 days ago	5b9659b
Corrected class name	deepti912 committed 27 days ago	52e39de
Rename Tester.txt to Tester.java	deepti912 committed 27 days ago	d63b492
Rename OpenDataCSVReader.txt to OpenDataCSVReader.java	deepti912 committed 27 days ago	8bea59e

12.4. Branches

UniversityNDC / Project3220

Watch 0 Star 0 Fork 0

Code Issues Pull requests Actions Projects 1 Wiki Security Insights Settings

Overview Yours Active Stale All branches Search branches...

Default branch

master Updated 2 hours ago by nsk2106 Default Change default branch

Active branches

Chandni380-patch-1 Updated 26 days ago by Chandni380 14 | 0 #1 Merged

12.5 Software version control has been managed by creating a master on GitHub & subsequent updates by commits & pull requests.
Bug & Issue tracking is being done.