KING'S College LONDON

University of London

Prototype for a population visualization tool

Final Project Report

Author: Jan Aldous Torres

Supervisor: Dr Simon Miles

Student ID: 1323454

April 16, 2017

Abstract

(Will be filled after the completion of the report). The abstract is a very brief summary of the report's contents. It should be about half-a-page long. Somebody unfamiliar with your project should have a good idea of what your work is about by reading the abstract alone.

Originality Avowal

I verify that I am the sole author of this report, except where explicitly stated to the contrary. I grant the right to King's College London to make paper and electronic copies of the submitted work for purposes of marking, plagiarism detection and archival, and to upload a copy of the work to Turnitin or another trusted plagiarism detection service. I confirm this report does not exceed 25,000 words.

Jan Aldous Torres

April 16, 2017

Acknowledgements

I would like to thank my supervisor Dr. Simon Miles for his support in this project. It is much appreciated.

I thank Lambeth Council for their support in the requirements and evaluation stages of the project. Taking time off their busy schedules was much appreciated.

Contents

List of Figures

Chapter 1

Introduction

[**?**] [5] [6] [7] [**?**] [9] [1] [2] [8] [4] [3]

Local government councils are challenged with the recent cuts in local government funding and a diverse population with different needs. Therefore, there is a need to efficiently allocate resources. The usual approach is to tackle issues raised by individual social services. However, there has been a shift in local government, which is promoted by the national government, to utilize the concept of customer segmentation. This will let them understand the needs of the population better thus letting them utilize and allocate resources in the most effective way based on those needs [4]. This approach would require the council to identify and therefore focus resources on vulnerable population groups. For example, instead of focusing on issues on education, they would focus on the children in care who are most susceptible to lower educational, social and employment outcomes. Some social services deal with the same population groups so this approach would require an integration of data from the different services. This gives an overview of all data from services provided by the councils. This is unlike the former approach which requires individual social services to conduct data analysis using their own data.[TJA1]

Lambeth Council in London wants to implement the new approach. However, current methods to analyze and visualize data take a long time before the analyses and visualizations would reach the commissioners, the people in the council who formulate strategies to tackle issues.

There are off-the-shelf software packages available which supports users in customer segmentation from clustering to the visualization of their data. In government, using data analysis in policy decision making is becoming more popular(new york guy video). The Local Government

Agency has created a guideline to support councils in tailor-made clustering and visualization [4]. Moreover, there have been projects by other local councils to visualizations of data on prominent population groups.

Lambeth Council would like to have a bespoke population segmentation tool that would be customized to their own needs and available data. This project aims to implement a prototype of such a tool as a Django web application.

The tool will focus on the situation in which population groups are in through visualizations. The situations they face will be defined by the data. Visualizations of this data will also help to compare characteristics of the group to the wider population and subgroups. This comparison may lead to observations on which problems a group or subgroup is facing. This may lead to the conclusion that the council may want to increase resources spent on a group or service. The tool basically aid the commissioner to apply customer segmentation on their population and identify which problems segments of the population are facing thus help make decision on the allocation of their resources.

Clustering will provide a more complete perspective into the different situations groups face instead of viewing a group as one entity and defines variables which distinguish the group from the rest of the population using clustering algorithms. Data integration from open government datasets will create a more in depth view of groups. And possible relations between geographical locationsâĂŹ characteristics ie facilities (GP), situation (crime) in a ward and whether they are on the London living wage. Since the user may not necessarily know what they are looking for when they read the data, the visualizations will be in the form that will let the user explore more aspects of the data.

1.1 Report Structure

The remainder of this paper is organized as follows. Chapter 2 introduces related work to the tool and data exploration, and Chapter 3 discusses the requirements of the tool. Chapter 4 outlines the specification and design of the tool. Chapter 5

Chapter 2

Background

This chapter includes work done on formulating better ways of with data exploration. It will later describe in more detail the work done on the access and visualization of open government data to non-technical people who require access to this type of data. The chapter ends with related application to the prototype of this project.

incorporates all relevant issues

main issues are analysed and developed

2.1 Data analysis in government

Give overview of using data in policy decision making

There has been work to make government data more accessible to people which do not have the technical abilities to merge and manipulate data such as journalists, data journalists, etc. [10]. This is especially true in open government data platforms where datasets take non-standard forms and whose data may be encoded in a way that is not immediately understandable. There have been efforts to evaluate government platforms and to enumerate features that would effectively let users make sense of the data [8].

2.2 Customer segmentation

Market segmentation or customer segmentation is concept used in business regarding the division of a homogenous population into segments which have similar attributes, wants, needs or demands. âĂIJIts objective is to design a marketing mix that precisely matches the expectations of customers in the targeted segment. Few companies are big enough to supply the needs

of an entire market; most must breakdown the total demand into segments and choose those that the company is best equipped to handle.âĂİ (businessdictionary.com). âĂIJThe four basic market segmentation-strategies are based on: behavioral, demographic, psychographic, and geographical differencesâĂİ (businessdictionary.com). Much of the need to have this is due to the lack of resources needed for the whole market. For many of the local government units, it is the constraint of the budget that would require an efficient allocation of resources by choosing specific groups. The national government and LGA promote the application of this concept usually used in the private sector into local government.

Clustering algorithms could be used to divide the population.

According to the LGA [], the collation of demographic data and the use of clustering algorithms such as k-means could be used to divide the population into segments. Choosing which

2.3 Related applications

There have been applications developed similar to the tool this project aims to implement. These include more general purpose customer segmentation tools which allows the user to analyze and then segment data using algorithms, to the visualization and use of that data. There have also been local governments which are following the concept of customer segmentation to also spend their resources more effectively.

Customer segmentation tool like Mosaic by Experian (mosaic) and Acorn by CACI (acorn) are tools both to allow population segmentation through customer profiling based on set of demographic and other indicators.

Customer classification is a concept promoted by the government and the Local Government Association (LGA) has created a guidance document for any council that wishes to implement such a tool (LGA) (smart cities).

Kent and Medway has tool which segments its population by social class and aims to highlight the key features which make each Group distinctive, to help you visualise the segmentation data and understand the essence of each Group (Kent Medway). It lets the user compare the values between all groups. This tool is bespoke to Kent MedwayâĂŹs data and the tool cannot be reused for other councilâĂŹs data.

There have been similarities between tools and recommendations to differentiate and describe each group. (Kent) (LGA) suggested to include maps showing the concentration of that group in each ward and in addition to a textual description, they include pictures to describe

each group. (Barnet) has a report on the customer segmentation of its population, and includes the percentage and textual description of its population. (Kent) includes graphs to visualize data but also word maps of key characteristics. (LGA) has suggested that the use of spider diagrams which detail the variables compared to the town and district average. Similarly (Acorn) (Kent), they graph pieces of data with an index of the group compared to the overall population. This visualizes both the average value and the groups relation to that average displays whether they are higher or lower than that average however each graph is created for each variable.

LGA has suggested data integration to enable other uses of data. (smart cities) defined use of explicit, âĂIJrecords of who has or is using a serviceâĂİ and implicit âĂIJknowledge of staff on customers using a serviceâĂİ customer data in an effort to know their customers in great detail.

2.4 Data visualization and exploration

Numerous applications exist to allow users to interact with data. Data exploration involves different types of interaction and functionality and there has been work to create a framework for data exploration [9].

There has been work done on enumerating the problems with data exploration through visualizations [10].

There has also been work done on the visualization of clusters through graphs, which show how closely related the clusters are to each other [7].

Show thinking of why some features of related applications are used here Use cases/features of Kent and Medway: - See group data, more specifically see pictures and phrases which describe the group, distribution between wards, data about services used, benefits used and demographics of the group as an index value (0-200) and a map of the groupâĂŹs population density in the councilâĂŹs area Features of neighbourhood.statistics.gov.uk - The user selects the data to be displayed on the map and chart. Map and geographical unit breakdown chart interacts with each other where hovering over one will highlight the area on both map and chart. The map is divided into geographical units (i.e. ONSâĂŹ lower output areas) which are colored according to the band which the area is in. The chart is according to the userâĂŹs setting. âĂć CSV of LambethâĂŹs 2016 Residential Survey: a survey of a sample of LambethâĂŹs population consisting of 1024 people about their quality of life, what they thought of LambethâĂŹs services and about the respondents themselves. The data is in the form of

CSV text files where each row contains a personâĂŹs entries as categorical data in the form of code (see below Survey Code Translation). Each single answer question has its own column and entries are in code as an integer between 1-100. For the questions which have multiple answers, each answer is regarded as a sub question in the form of âĂIJQ5AâĂİ meaning question âĂIJ5âĂİ, choice code âĂIJAâĂİ (makes it look like a string but its supposed to be an integer) (see below Survey Code Translation). Therefore, a sub question has its own column and entries are in code as a âĂIJ1âĂİ or âĂIJ0âĂİ. There are also other fields which have been added to the original survey results such as group, subgroup, quintile, which is a result of previous data analysis. âĂć CSV of LambethâĂŹs 2016 Residential Survey: a survey of a sample of LambethâĂŹs population consisting of 1024 people about their quality of life, what they thought of LambethâĂŹs services and about the respondents themselves. The data is in the form of CSV text files where each row contains a personâĂŹs entries as categorical data in the form of code (see below Survey Code Translation). Each single answer question has its own column and entries are in code as an integer between 1-100. For the questions which have multiple answers, each answer is regarded as a sub question in the form of âĂIJQ5AâĂİ meaning question âĂIJ5âĂİ, choice code âĂIJAâĂİ (makes it look like a string but its supposed to be an integer) (see below Survey Code Translation). Therefore, a sub question has its own column and entries are in code as a âĂIJ1âĂİ or âĂIJ0âĂİ. There are also other fields which have been added to the original survey results such as group, subgroup, quintile, which is a result of previous data analysis. âĂć LambethâĂŹs 2016 Residential Survey Code Translation: a Microsoft Word document of the original survey with the original questions and code used in the data associated with the question. Under each question is the choice code and English meaning of the choice. The choice code is either a number for single answer questions (e.g. 1. Male, 2. Female) or letters for multiple answer questions (e.g. A. Access to nature, B. Activities for teenagers, etc.). The system may be able to save access (through URLs to CSV and JSON files) or actual files of the datasets (CSV or JSON files themselves). The URLs must be kept in persistent storage. If the input is in the form of a file, the file must be stored.The system may be able to save access (through URLs to CSV and JSON files) or actual files of the datasets (CSV or JSON files themselves). The URLs must be kept in persistent storage. If the input is in the form of a file, the file must be stored.'''

Chapter 3

Requirements

The functional and non-functional requirements was created based on communication, through interviews and email, with a member of the Policy and Communications Team, and requirements which I have created based on initial requirements by our contact. Our contact described the context in which the tool may be used, the overall objective of the software, the users' technical abilities and some functional requirements, namely the method to create groups (see section 3.4). Communication with our contact was not sustained after initial requirements were collected due to the council's large workload. The rest of the requirements was created in accordance with the overall goal, initial requirements and according to my own analysis of the data.

3.1 Objective

The purpose of this application is to support those in local government councils, in charge of creating policies, to explore data on residents demographics and satisfaction with council services. User-defined groups will allow the user to focus on specific portions of the population. Aided by visualizations through graphs and maps, it may lead the user to identify which segments of the population requires the which resources.

3.2 Description of data to be used

The following local data provided by Lambeth should be used:

- **CSV of Lambeth's 2016 Residential Survey**: a survey of a sample of Lambeth's population consisting of 1024 people about their quality of life, what they thought of

Lambeth's services and about the respondents themselves. The data is in the form of a CSV text file. Residents'answers are categorical data in the form of code specified by the Survey Code Translation (see point below, Survey Code Translation). Single answer questions have its own column and its choice code is a positive integer. Multiple answer questions, each answer is regarded as a sub question in the form of 'Q5A' meaning question 5, choice code 'A' (see below Survey Code Translation). Therefore, a sub question has its own column and entries are in code as a 1 or 0 meaning yes or no respectively. There are also other fields which have been added to the original survey results such as group, subgroup, quintile, which is a result of previous data analysis.

- **Microsoft Word document of Lambeth's 2016 Residential Survey Code Translation**: a Microsoft Word document of the original survey with the original questions and code used in the data associated with the question. Under each question is the choice code and English meaning of the choice. The choice code is either a number for single answer questions (e.g. 1. Male, 2. Female) or letters for multiple answer questions (e.g. A. Access to nature, B. Activities for teenagers, etc.).
- **GeoJSON of Lambeth's Ward's Boundary Specifications**: downloaded from Lambeth's website, it is a GeoJSON, simple geographical features encoded as a JSON, of the geographical boundaries of each ward.

3.3 Functional Requirements

Functional requirements with the words textbfshould or textbfmust is a required feature. Requirements with the word textbfmay, is a desirable feature which may or may not be implemented should there not be enough development time.

1. Data storage requirements
   (a) 1. The system must save the actual files of the datasets (CSV, text files or JSON) and not access the file through a URL of another website.
2. Grouping requirements
   (a) The user should be able to create groups based on the parameters to the 5 factors (see section 3.4). A group's information should be kept in a database.
   (b) The user should be able to view data about a group through graphs.

i. Single answer questions or questions which requires only one answer should be visualized as stacked bar charts or pie charts.

ii. Multi-code questions or questions which requires more than one answer should be visualized as bar charts.

iii. Clicking data on a graph should display which wards have answered the selected question and choice on the map (see functional requirement 4).

iv. The questions under section Questions to be visualized should be visualized following requirements 3a and 3b.

(c) Like a population density map, which shows higher density concentrations of people as a darker color and lower density concentrations as a lighter color on particular areas of the map, the map should show the selected data (see functional requirement 3c). The map should be divided into the council's wards; the boundaries of each ward should be obvious. The color of the ward should depend on the number of residents who have the selected question and answer (see functional requirement 3c).

(d) The user should be able to compare values of a data variable between all groups such that groups should be compared to the average value of the variable in the whole population (i.e. compare the percentage of disabled people who are male to the percentage of the whole population who are male).

(e) 1. The main data used in the visualizations should be the CSV of Lambeth's Residential Survey and its Code Translation.

3. Clustering requirements

(a) The system should segment a group using a clustering algorithm.

(b) The system should display any statistical information (e.g. the mean of the answers of a question) on the differences between the clusters.

(c) The user should be able to compare information between the group's data and each of the clusters'data.

3.4 Creating groups

A group should be based on residents'answers to 5 questions in the residential survey. At least 1 question is required to form a group, therefore the rest of the 4 questions could have any

answer. The group will be composed of residents who match the required answer(s) to the required questions.

There are 5 questions in which a user can create a group:

1. Whether they have a disability or long term illness (Q43)
2. What type of benefits do they acquire (Q45A-Q45K)
3. Educational/employment activity (Q46)
4. Whether they are on the London Living Wage (Q47)
5. Housing tenure (i.e. council tenant, private owner, etc.) (Q35)

*Question number in the CSV residential survey is in brackets.

3.5 Questions to visualize

Once a group has been created, the user must be able to see the answers to the following survey questions:

1. What matters most to them most (Q5) (Multiple, top 3)
2. How was their last contact with the Council was made (Q26A- Q26G) (Multiple)
3. How they use the website (Q29) (Multiple)
4. What services they have used (Q39) (Multiple)
5. How they access the internet (Q50) (Multiple)
6. How well the changes have benefited them (Q11) (Single)
7. Whether they feel they belong to their neighborhood (Q13_R1) (Single)
8. Whether they value the friendships in their neighborhood (Q13_R2) (Single)
9. Whether they could approach a neighbor for advice (Q13_R3) (Single)
10. Whether neighbors help out each other (Q13_R4) (Single)
11. Whether they would be willing to work with others to improve their neighborhood (Q13_R5) (Single)
12. Whether they would join community events in their area (Q13_R6) (Single)

13. Whether they regularly stop and talk to people in their neighborhood (Q13_R7) (Single)
14. Whether they would speak highly of their neighborhood when asked (Q13_R8) (Single)
15. Gender (QGEN) (Single)
16. Age (QAGE) (Single)
17. Ethnicity (QETH) (Single)

*Question number in the residential survey is in brackets followed by whether they are a single answer question (Single) or multiple answer question (Multiple).

The answers to the questions above should be visualized using graphs for each group and their clusters based on functional requirement 3.

3.6 Non-functional requirements

Usability: The users for the program are the policy makers of the council, those responsible for formulating strategies to allocate resources for a council based on the presentation of data analysis given to them. They do not necessarily have technical skills, in terms of being able to operate applications, or advanced statistical analysis skills. Since the users are not technical savvy, the ease of use is imperative to the design of the user interface. The visualizations should also be easily interpreted.

In terms of maintainability, this version of the system is only a prototype to show the potentials of such a system, therefore there will not be a need for maintainability of the code. In terms of security, the data being used is anonymous therefore there will be no need for security measures for the data.

Chapter 4

Specification & Design

This section describes the design of the system as a website.

4.1 Platform

A web application as a platform to create visualizations takes advantage of the numerous open source visualization libraries and HTML, CSS and JavaScript are well equipped in creating interactive and visually pleasing interfaces. For these reasons, the application will take the form of a website.

More specifically the design will be implemented as a Django (citation) web application. Django is a popular Python web application. It is well documented and has numerous open source libraries which supports the creation of visualizations. Using this framework, third-party libraries could be utilized to produce JavaScript needed to create visualizations. Python's machine learning libraries will be used to manipulate and analyze data.

4.2 Preprocessing of data

Prior to the creation of the application, some preprocessing of the data must be done. The Microsoft Word document of the survey code translation in its current form is not able to be queried since it is just a Word document. It should be created as a formatted text file that is still readable for the user so that the system can query it and and in the future, the user can create the text file themselves. Inputting the translations in one file will be more user-friendly than inputting translations of individual questions through a normal form.

For this system design, since the user has only given the survey code translation as a Word document. I will create the formatted text file myself.

<Source information>
$
<Survey question code 1>|<Shortened question in English 1>|<Original question in English 1>
<SINGLE or MULTIPLE>
<Choice code>;<Choice in English>
...
<Choice code>;<Choice in English>
$
<Survey question code 2>|<Shortened question in English 2>|<Original question in English 2>
...

Figure 4.1: Formatted text file of survey code translation (See Appendix for actual text)

As seen above the text file is still readable as each question's data is separated by a '$' sign and each code is separated by a ';' or '—'. The question text is distinguishable from the choices text.

4.3 System architecture

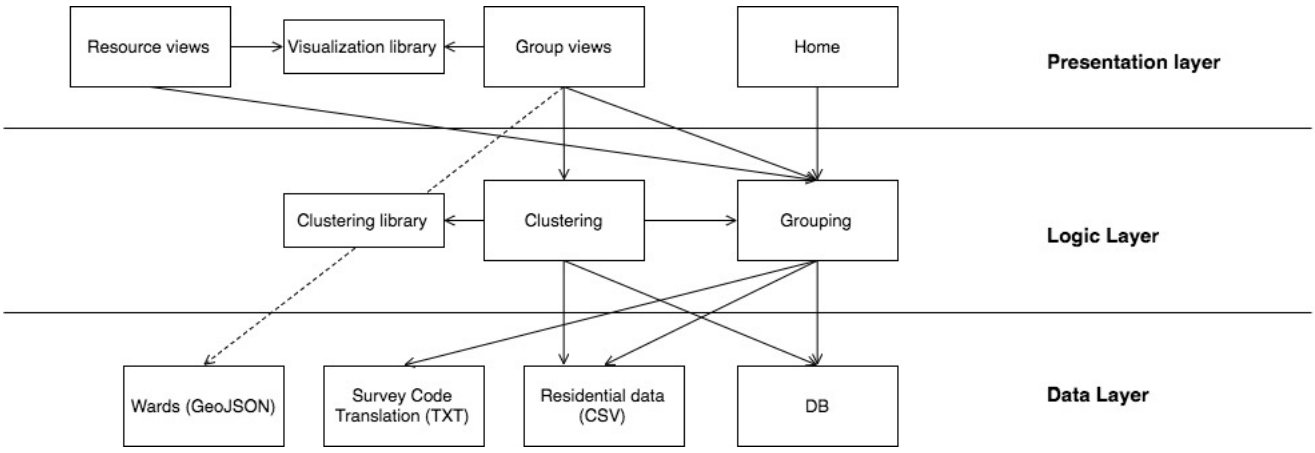

Figure 4.2: System architecture diagram

The system will follow the three-layer architecture.

The data layer will consist of the following components:

- **Ward**: a GeoJSON file containing the geographical boundaries of each ward
- **Survey Code Translation**: a text file formatted according to section 4.2 containing the translations of the survey code to English text

- **Residential data**: a CSV file of the 2016 Lambeth Residential Survey
- **DB**: the database which includes Groups created by the user and groups' Clusters produced by the clustering algorithms (see Appendix for more details)

The logic layer will consist of the following components:

- **Clustering library**: a third-party library used by the Clustering component to cluster group data
- **Clustering**: performs clustering on a group and stores results in the database
- **Grouping**: creates groups and extracts groups from survey data

The presentation layer will consist of the following components (see section 4.5 for more details):

- **Resource views**: shows visualizations data on each group for each question in the residential data set
- **Visualization library**: a third-party library used to create graph and map visualizations
- **Group views**: shows visualizations of groups and compares groups
- **Home**: main menu to choose which group to view

Though the original source of the Wards GeoJSON is from Lambeth's open data website, it will be downloaded as a JSON file and stored as part of the source code of the application. Since the ward boundaries is unlikely to changing, storing the file in the source code is preferable as it will not depend on another server's performance.

The Clustering component could be implemented separately and another clustering algorithm can be implemented without changing the Grouping component.

4.4 Algorithms

Data analysis will be carried out by the system through data clustering. The clusters will be generated by the Clustering component and will be saved to the database. The default number of clusters is 1 and the user will be able to manipulate the number of clusters.

There are numerous clustering algorithms however, due to the categorical nature of the data, k-modes clustering will be used instead of the the usual k-means clustering. K-modes has been proven to be more effective in clustering categorical data (citation).

4.5 User interface

The system's main goal is to present data and let the user interact with the user in an effective way. The following figures describe the interaction and design of the user interfaces.

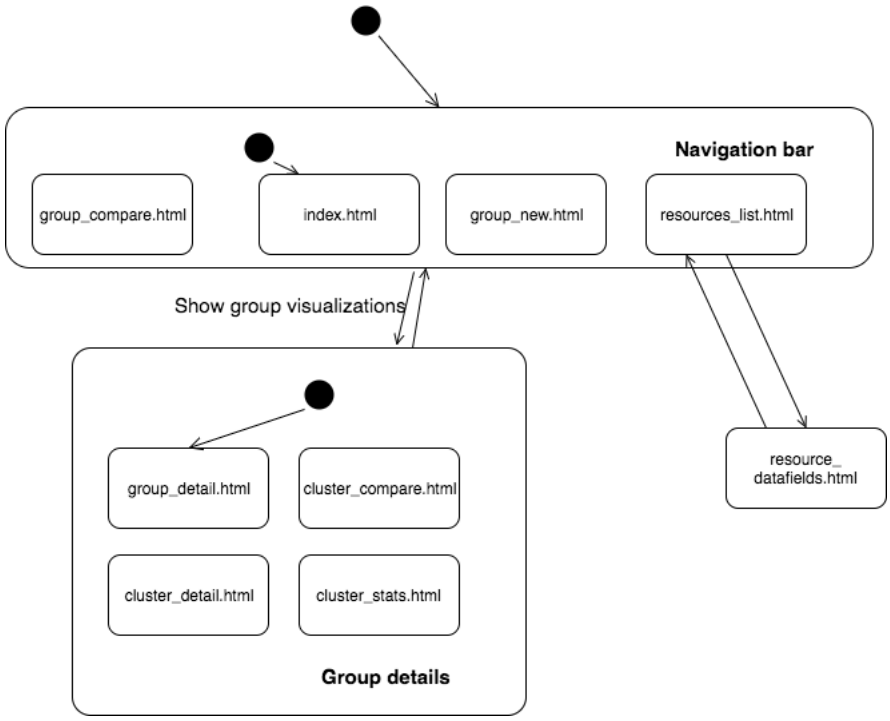

Figure 4.3: Interaction diagram of the web application

The interaction diagram above describes the flow between the pages in the website. It is designed so that switching pages is done in the least number of clicks possible. This will be done through a navigation bar where most pages will be accessible. The group detail pages will only be accessible from the group_detail.html page to avoid confusion with other groups' pages.

The following describes the functions of each page:

- **index.html**: contains a list of links to all groups
- **group_detail.html**: has visualizations of a group's data through maps and charts (see section for more detail)
- **group_compare.html**: compares all the groups to the average of the whole population of a data variable. This will visualize the comparison of the values of the cluster to the average value for that variable in the whole population (see section for more detail)
- **cluster_compare.html**: compares the data on the clusters of a group, to the group and the whole population for each question (see section for more detail)

- **cluster_stats.html**: lets the user change the number of clusters and shows differences, namely the means of an answer to a question, between each cluster (see section for more detail)
- **group_new.html**: contains a form that adds a new group to the database
- **resources_list.html**: lists links to residential survey questions
- **resources_datafields.html**: shows visualizations of data on all groups for a survey question

*Note that variable is to denote the selected question and choice (i.e. Q11, Full-time employee)

This design gives the user different perspectives on a population. Namely, a group's data (group_detail.html), data of clusters within a group (cluster_detail.html), differences between a group and its clusters (cluster_compare.html, cluster_stats.html) and differences between groups (resources_datafields.html). These features are intended to enable the user to identify interesting trends within a specific group, groups or cluster which may not be possible with fewer interfaces.

4.6 Interface design

The pages in the interaction diagram are described in more detail in the following page designs. Since a large part of the system is about the presentation of data, the layout of the visualizations and interactions within pages, shown as annotations, were designed.

The page on figure 4.4 is intended for the user to view the residential data which is visualized in graphs on the right-hand side of the page. Selecting a variable (i.e. a survey question and choice) on a graph will show its data on the map, on the left-hand side of the page, as a population density map. Each ward will be colored depending on the number of residents for which the resident chose the variable. This shows the user where residents who have chosen the variable live. Supporting the map is a bar chart of the number of residents for each ward who have chosen the variable which is ordered from greatest to least. This lets the user easily identify the distribution of the selected variable throughout the wards. The charts will follow functional requirements 3, which specifies which type of graph (e.g. bar chart, pie chart, etc.) to use for which type of question.

In addition, the meta-data about the data set used in the page is shown in the top left panel, which shows the number of residents in the group as a raw number and percentage and

Figure 4.4: Page design for group_detail.html and cluster_compare.html

the source of the data.

The same page design is used for cluster_compare.html except that the Question graphs will be in the form of stacked bar charts which show the question data on the whole population, the group and the clusters (see figure 4.5).

The figure 4.5 is the visualization for the resources_datafields.html page. However, the y-axis will instead be the groups in the database.

The page on figure 4.6 will let the user compare the percent difference from the whole population using the following equation:

(Percent of the group's population who answered selected question with selected choice-Percent of the overall population who answered selected question with selected choice)*100

Therefore groups more than 0% has a bigger proportion of their population which have answered the selected variable and vice versa for groups less than 0%. This may be an interesting

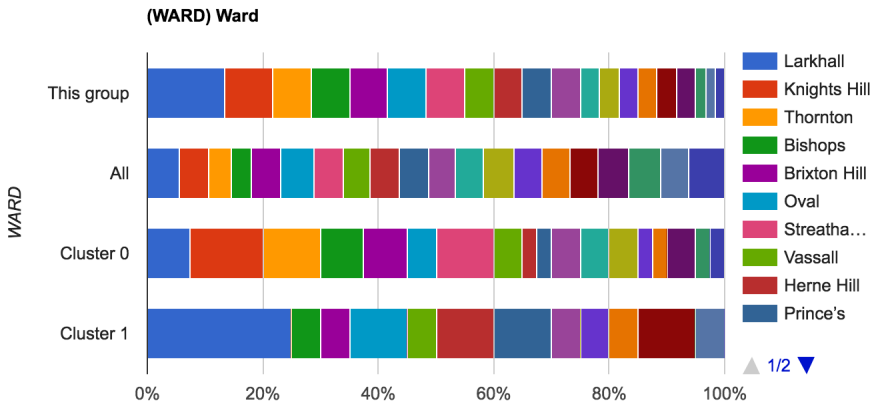

Figure 4.5: Chart for comparison of the data of the whole population, a group and its clusters

visualization for some questions more than others.

The page on figure 4.7 gives another perspective to the comparisons show in cluster_compare.html. Instead of showing the all the data variables, the set of answers are consolidated into a mean. The user can compare means of a question for each cluster which will highlight the differences between each cluster which is not always obvious in cluster_compare.html. The user may go back to cluster_compare.html to the data in more detail.

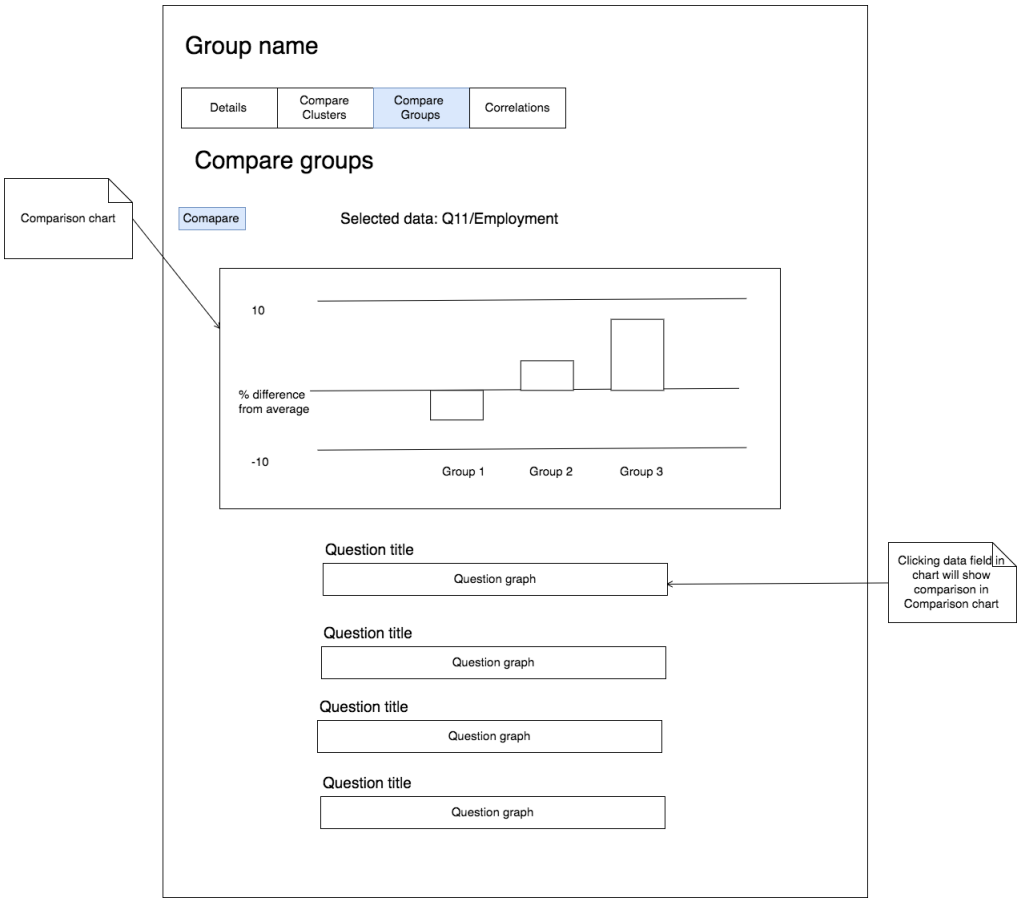

Figure 4.6: Page design for group_compare.html

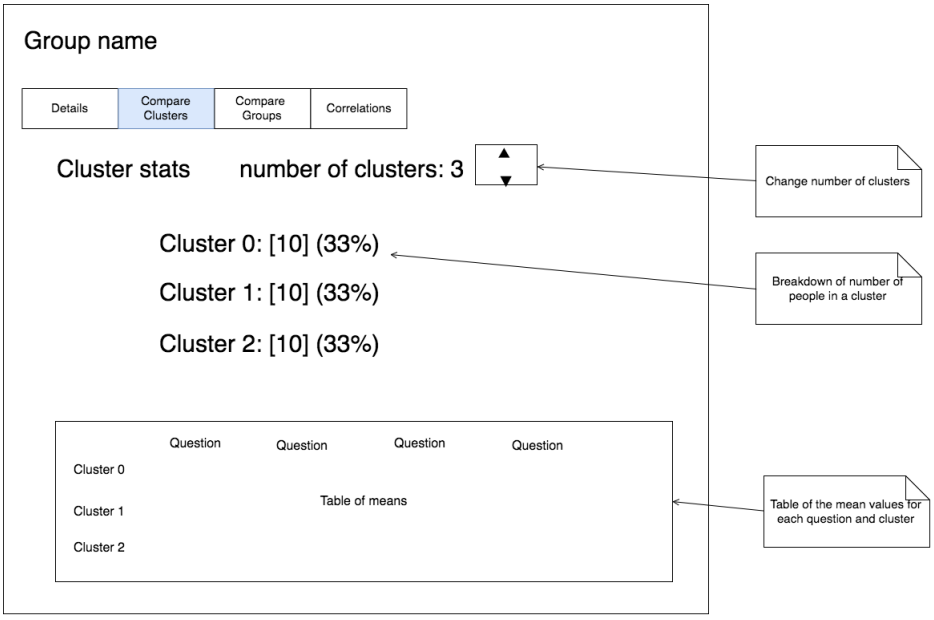

Figure 4.7: Page design for cluster_stats.html

Chapter 5

Implementation and Testing

This chapter explains the development approach used followed by how each element of the design was implemented.

5.1 Development approach

The software was developed through evolutionary prototyping. Since the initial requirements were not complete, this approach gave flexibility on what is to be implemented. Unlike throw-away prototyping where code is thrown away, this method begins with the requirements that are understood and seeks to incrementally implement additional features proposed by the user (citation). Though the usual definition is to modify the software based on feedback from the user, I have taken the place of the user and have incrementally designed and then implemented what is most likely suitable for the users' analysis of the data. Therefore, the interaction and interface designs of the application were created incrementally and the design as a whole was added in the Design chapter.

The following list outlines the iterations of the prototype including the requirements that were implemented:

- Implementation of groups in the population and graph visualizations of certain questions, group_detail.html and group_new.html using the residential survey data only (Functional Requirements 1, 2a, 2bi, 2bii, 2biv, 2e)
- Addition of survey code translations to the visualizations implemented in the iteration 1.
- Implementation of resources_list and resource_datafields.html

- Implementation of the map in group_detail.html (Functional Requirements 2iii, 2c)
- Implementation of the group_compare.html (Functional Requirement 2d)
- Implementation of clustering and cluster_detail.html (Functional Requirement 3a)
- Implementation of the interface cluster_stats (Functional Requirement 3b, 3c)
- Implementation of the interface cluster_compare.html (Funcitonal Requirement 3c)

The implementation stated with the initial requirements and each succeeding increment involved design, implementation and testing. The functional requirements were relatively done in order it was listed since each requirement usually depended on the previous requirement. The function and interface of each page was done together so that each prototype is functional. The prototype could then be tested and features for the next prototype could be suggested, which follows the spirit of evolutionary prototyping.

Development involved developing sections of the prototype that was independent of the system for some iterations. This was to familiarize myself to the different languages and libraries used in this application, which lead to the creation of an independently functioning section of the prototype. This was then integrated into the system to avoid confusion with other parts of the code. Independently developing it ensured that it was functional before it was integrated. This also reduced the risk of it not working. An example is the integration of a Google Map and the Google Charts used in group_detail.html. The clicking of a chart and display on the map was implemented in a separate HTML page and ensured that it worked. This was then integrated into the actual page, group_detail.html. The user friendliness was also improved through this method, as testing the interfaces in reality makes missing user friendly elements more obvious than could be seen in the design of the interfaces.

5.2 Third party libraries

The third party libraries are listed below:

- Django: a Python web framework used to create web applications
- Bootstrap: a CSS library used to make webpages more aesthetically pleasing
- Pandas (citation): a Python library used for data analysis and data structures
- Google Charts: a JavaScript API which creates different types of charts

- Google Maps: a JavaScript API which creates a map
- Graphos (citation): a Django app which turns data in the form of Python's list data structure into Google Charts JavaScript code âĂć K-modes: a Python library which applies a clustering algorithm most suited for categorical data on a set of data

Since it is a web application, a combination of programming languages were used, Python for the back-end with Django, HTML and CSS for the layout aesthetics of the web page and JavaScript to make the pages interactive.

5.3 Page implementations

The subsections below describe how each part of the software was implemented and some screenshots of the actual implementation.

5.3.1 Visualization of data into charts

Pandas Dataframe was used to store the Residential Survey data. The survey data as a CSV was loaded into a Dataframe and the latter was used to query the data around the system. A single instance of the Dataframe was kept to refrain from repetitively opening and reading the same file through the singleton design pattern.

Graphos was used to create the Google Charts. The required data of a chart was queried and formatted to Graphos' specficiations and a Graphos object was created. Since Django is a web framework, it has a template feature where Python objects, such as a Graphos graph object, could be turned into content for an HTML file.

Depending on the type of question, the charts on figure 5.1 was created.

As a default feature of Google Charts, hovering over a piece of data will display the value of data (see top left of figure 5.1).

5.3.2 Integration of survey code

Following the design for the preprocessing of data (see figure 5.4), the English Code translation text file was created. Readtext.py is the Python module which reads and creates each question into Question objects. Readcsv.py consolidates English and code data into a Python list. This is inputted into a Graphos class which produces the Google Charts JavaScript code.

Figure 5.4 shows the data flow from raw data to the creation of Graphos charts. This is a simplified view which omits other Django modules.

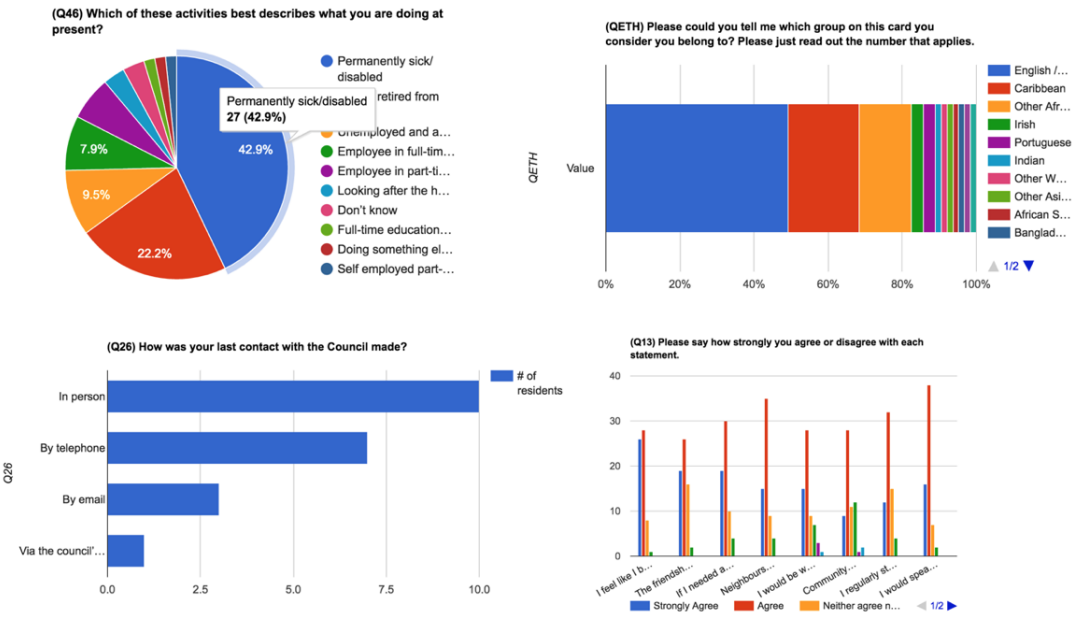

Figure 5.1: Implementation of the charts

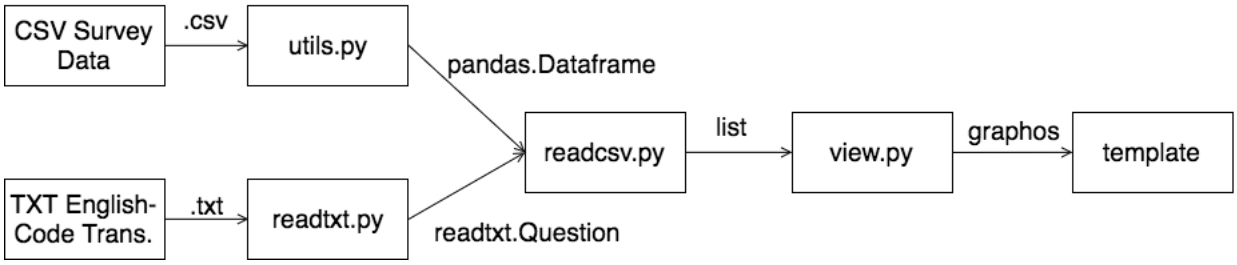

Figure 5.2: Implementation of the charts

5.3.3 Visualization of wards on a map and ward breakdown chart

The map was adapted from a tutorial in the Google Maps website which implemented the display of colored areas depending on the value of that area. In this case, the areas were changed to Lambeth's wards.

The interactivity between the Google Map and Google Chart in the group_detail page was developed as described in Development approach (see section 5.1).

Additional features to the Graphos code was added. A listener to see if the chart was clicked was added to Graphos' code. When a chart is clicked, the function which redraws the chart, map and changes the text is invoked.

This interactivity required the map's contents to be changed (i.e. change the colors of the wards depending on the question selected). A JSON containing a list, which is required in the

creation of Google Maps, is populated with the selected data from a chart (see figure **??**). This JSON is requested by clicking a chart on the right hand side of the page, which invokes the redrawing of the map. The figure below explains the data flow.

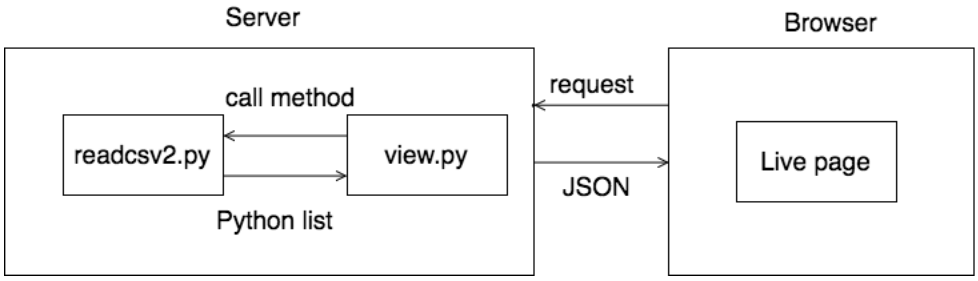

Figure 5.3: Implementation of the charts

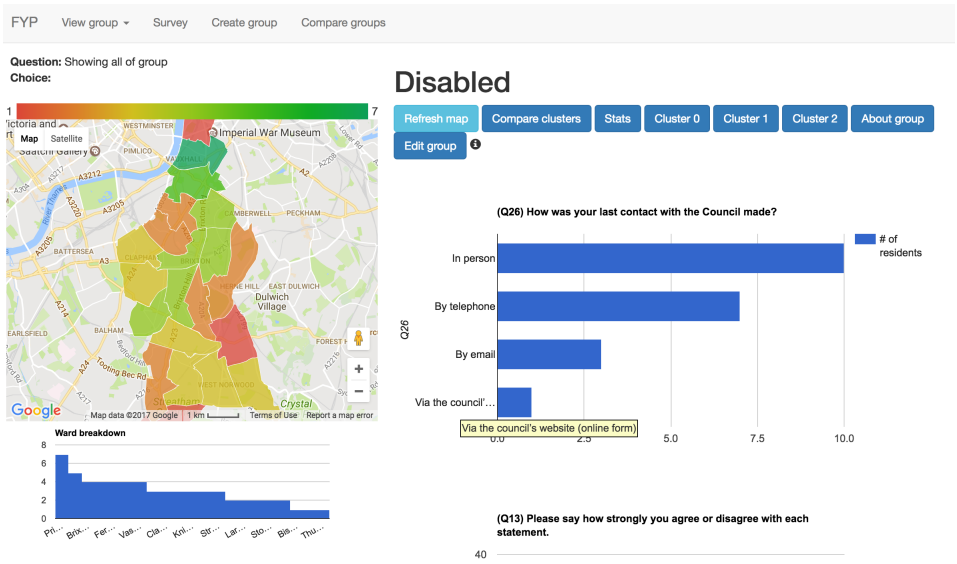

Figure 5.4: Screenshot of the group_detail page

Hovering over the map's wards will display a box containing the ward's name and value on the bottom left hand side of the map (see figure 5.5).

5.3.4 Visualization the Comparison of Clusters

A group's data, the whole survey data and clusters' data were visualized together for each question in a similar method to the visualization of a single group. To differentiate between each of the clusters' data for each question, the cluster id was included into the chart array. This was kept hidden when the array is displayed on the chart, but it is used to know which cluster is clicked when the user wants to display a particular variable on the map.

All charts are stacked bar charts. The click to display on map feature is applicable to these

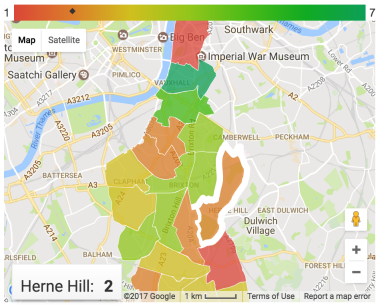

Figure 5.5: Screenshot of hovering over a ward on the Google Map

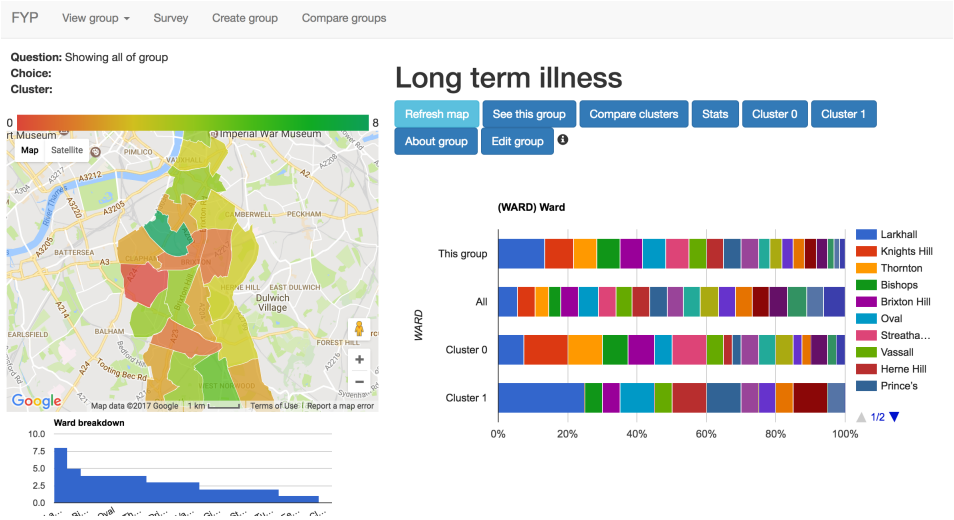

Figure 5.6: Screenshot of the cluster_compare page

charts as well.

5.3.5 Comparison of groups

On the right-hand side of the page (see figure 5.7), the charts generated from the All group with the same method as in a group_detail page. However, the left-hand side of the page is a bar chart of the group's percent difference from the whole survey population. As with a group's Detail page, clicking a piece of data on a chart will instigate a change in the left-hand side bar chart.

The percent difference is calculated using the equation found in the Design chapter (see section s4.6)

5.3.6 Clustering algorithm

Each group was clustered according to the number of clusters set by the user in the Stats page. The default number of clusters is 1. K-modes clustering (citation) was used since it is more

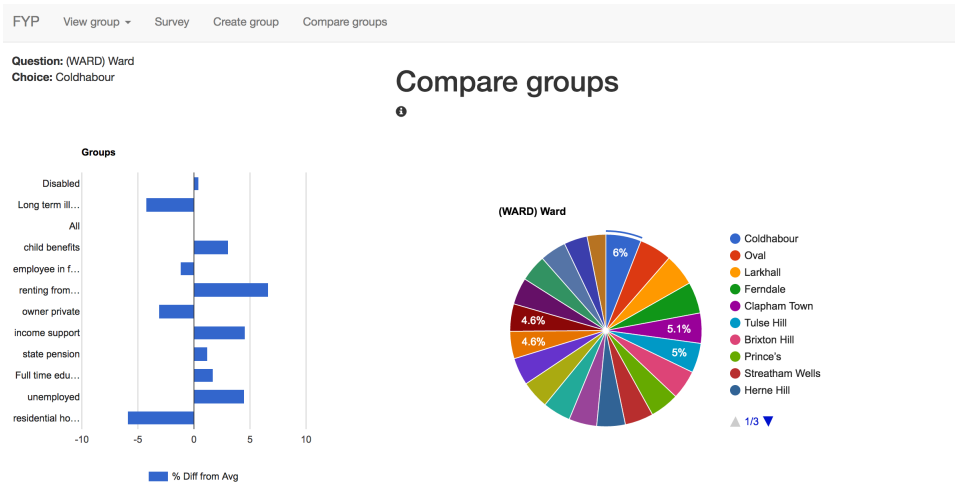

Figure 5.7: Screenshot of the group_compare page

effective than k-means clustering to cluster categorical data, which the survey data is mainly composed of.

The results of clustering are stored in the database. The Subcluster Django model created the database table with fields: serial (i.e. serial of row in Survey data), group (i.e. Cluster model) and cluster (i.e. id of cluster). Should the number of clusters be changed by the user, the Subcluster records of the group will be deleted and replaced. If the clusters are queried, then the system will query the Subcluster database table so that the clusters remain the same.

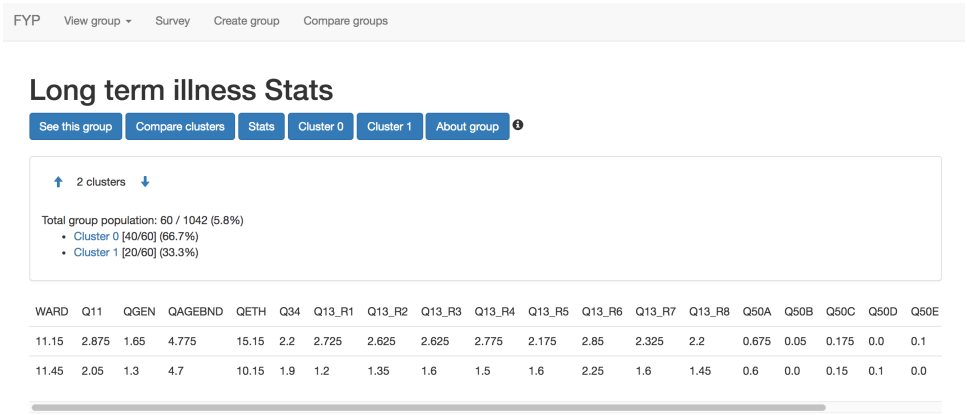

Figure 5.8: Screenshot of the cluster_stats page

5.3.7 Resource question page

Similar to the clusters_compare page and implemented with a similar Python operation, the survey question page displays more than one group in the same chart.

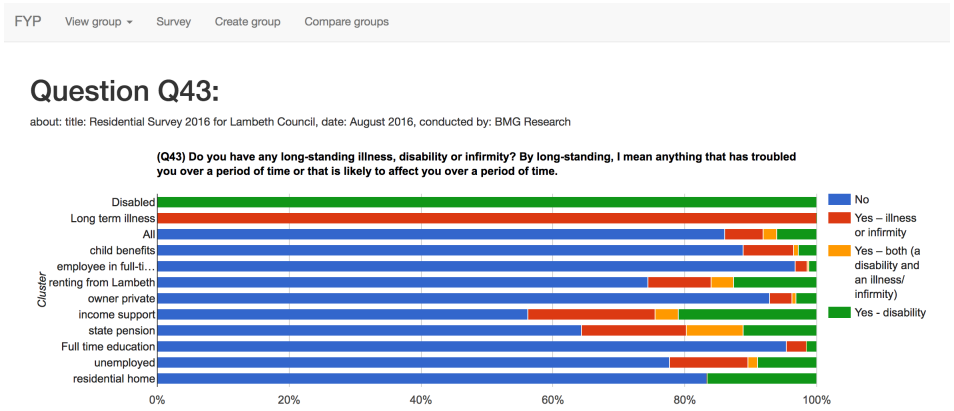

Figure 5.9: Screenshot of the resource_datafield page

Chapter 6

Professional and Ethical Issues

Either in a seperate section or throughout the report demonstrate that you are aware of the **Code of Conduct & Code of Good Practice** issued by the British Computer Society and have applied their principles, where appropriate, as you carried out your project.

6.1 Section Heading

Chapter 7

Results/Evaluation

7.1 Software Testing

7.2 Section Heading

Chapter 8

Conclusion and Future Work

The project's conclusions should list the key things that have been learnt as a consequence of engaging in your project work. For example, "The use of overloading in C++ provides a very elegant mechanism for transparent parallelisation of sequential programs", or "The overheads of linear-time n-body algorithms makes them computationally less efficient than $O(n \log n)$ algorithms for systems with less than 100000 particles". Avoid tedious personal reflections like "I learned a lot about C++ programming...", or "Simulating colliding galaxies can be real fun...". It is common to finish the report by listing ways in which the project can be taken further. This might, for example, be a plan for turning a piece of software or hardware into a marketable product, or a set of ideas for possibly turning your project into an MPhil or PhD.

References

[1] 3. customer segmentation and profiling.

[2] Customer segmentation.

[3] Customer segmentation.

[4] Developing a customer classfication tool: Guidance document for local authorities.

[5] Kent & medway interactive guide.

[6] Brigitte Boden, Roman Haag, and Thomas Seidl. Detecting and exploring clusters in attributed graphs: a plugin for the gephi platform. In *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management*, CIKM '13, pages 2505–2508. ACM.

[7] Drew Conway. Data science through the lens of social science. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '14, pages 1520–1520. ACM.

[8] Anna Gogolou, Marialena Kyriakidi, and Yannis Ioannidis. Data exploration: A roll call of all user-data interaction functionality. In *Proceedings of the Third International Workshop on Exploratory Search in Databases and the Web*, ExploreDB '16, pages 31–33. ACM.

[9] Alvaro Graves and James Hendler. Visualization tools for open government data. In *Proceedings of the 14th Annual International Conference on Digital Government Research*, dg.o '13, pages 136–145. ACM.

Appendix A

Extra Information

A.1 Tables, proofs, graphs, test cases, ...

The appendices contain information that is peripheral to the main body of the report. Information typically included in the Appendix are things like tables, proofs, graphs, test cases or any other material that would break up the theme of the text if it appeared in the body of the report. It is necessary to include your source code listings in an appendix that is separate from the body of your written report (see the information on Program Listings below).

Appendix B

User Guide

B.1 Instructions

You must provide an adequate user guide for your software. The guide should provide easily understood instructions on how to use your software. A particularly useful approach is to treat the user guide as a walk-through of a typical session, or set of sessions, which collectively display all of the features of your package. Technical details of how the package works are rarely required. Keep the guide concise and simple. The extensive use of diagrams, illustrating the package in action, can often be particularly helpful. The user guide is sometimes included as a chapter in the main body of the report, but is often better included in an appendix to the main report.

Appendix C

Source Code

C.1 Instructions

Complete source code listings must be submitted as an appendix to the report. The project source codes are usually spread out over several files/units. You should try to help the reader to navigate through your source code by providing a "table of contents" (titles of these files/units and one line descriptions). The first page of the program listings folder must contain the following statement certifying the work as your own: "I verify that I am the sole author of the programs contained in this folder, except where explicitly stated to the contrary". Your (typed) signature and the date should follow this statement.

All work on programs must stop once the code is submitted to KEATS. You are required to keep safely several copies of this version of the program and you must use one of these copies in the project examination. Your examiners may ask to see the last-modified dates of your program files, and may ask you to demonstrate that the program files you use in the project examination are identical to the program files you have uploaded to KEATS. Any attempt to demonstrate code that is not included in your submitted source listings is an attempt to cheat; any such attempt will be reported to the KCL Misconduct Committee.

You may find it easier to firstly generate a PDF of your source code using a text editor and then merge it to the end of your report. There are many free tools available that allow you to merge PDF files.