UNIVERSITY OF CAPE TOWN

DEPARTMENT OF COMPUTER SCIENCE

Computer Science Honours
Final Paper
2016

Title: SANCTUM: Resource- and Job Management for performing data mining on Twitter data, with Visualizations of output

Author: Pieter van der Walt

Project Abbreviation: SANCTUM

Supervisor(s): Jivashi Nagar

Category	**Min**	**Max**	**Chosen**
Requirement Analysis and Design	0	20	15
Theoretical Analysis	0	25	
Experiment Design and Execution	0	20	
System Development and Implementation	0	15	15
Results, Findings and Conclusion	10	20	15
Aim Formulation and Background Work	10	15	15
Quality of Paper Writing and Presentation	10		10
Quality of Deliverables	10		10
Overall General Project Evaluation (*this section allowed only with motivation letter from supervisor*)	0	10	
Total marks		**80**	80

SANCTUM: Resource- and Job management for performing data mining on Twitter data, with Visualizations of output

Pieter van der Walt
University of Cape Town
Cape Town, Western Cape
VWLPIE004@myuct.ac.za

ABSTRACT

The application of data mining algorithms to social media data presents interesting opportunities to researchers and businesses alike. However, extracting useful information from the results of the algorithms can often be difficult and tedious. Setting up and maintaining an environment that's capable of analyzing large amounts of data can also be complicated. This paper presents a system that makes the analysis of Twitter datasets convenient, and makes it easy to extract meaningful information by creating useful visualizations of the results. An iterative, user-centered design process was followed to create the system's interface. The interface was tested with potential users. Some problems were identified and rectified. Overall, the interface was found to be easy to use, convenient, and useful. Possible areas for improvements and extensions to the system are also identified.

CCS CONCEPTS

• **Software and its engineering** → **Process management**; • **Human-centered computing** → *Visualization techniques*;

KEYWORDS

Social media, Twitter, data mining, Information Retrieval, Association Rules, Visualizations

1 INTRODUCTION

1.1 Project Significance

Social media has become an integral part of society, and generates massive amounts of data on a daily basis [1]. By employing data mining techniques on these datasets, researchers and businesses can gain valuable insight into the dynamics of online communities and individual users. These insights include: detecting implicit groups and communities, discovering interesting associations between topics, identifying influential people in online communities, understanding the evolution of the network, and determining user sentiment towards certain topics [1].

Unfortunately, the analysis of large amounts of social media data requires significant amounts of processing power, which can often be costly to both researchers and businesses. Maintaining a cluster of your own, or renting a cluster on a permanent basis from providers such as Amazon Web Services (AWS) is an expensive exercise. Additionally, extracting meaningful information from the results of the analysis can be a time-consuming and tedious process. Therefore, having a system that displays the results in an organized manner and with helpful visualizations that can easily be analyzed, will be convenient and time-saving. When the presentation of the the data mining output is combined with automatic job-management and resource-management, this system will assist in solving the major problems faced when mining social media data.

1.2 Project Aims

The project aims to develop a system that allows for the easy perusal and analysis of large amounts of social media data (specifically Twitter data) by presenting the results in an organized and meaningful manner. This system will also automatically handle the resource- and job-management tasks that are required to mine the Twitter data. As a whole, the system will perform Information Retrieval (IR) and Association Rule Mining on the provided data, and present these results to the researcher using tables and useful visualizations. The Information Retrieval and Association Rule Mining components of this system are not part of this paper, which will focus on the user interface, work flow, and visualizations of the system. The main users of the system will be researchers who are interested in the analysis of Twitter data. The main concern for both the front-end and back-end of the system is the scalability of the functionality. More detailed requirements are given in Section 3.

1.3 Structure of Report

Section 2 discusses a few existing projects and papers that are relevant to this project. Then Section 3 describes the process followed to gather the system requirements and what these requirements are. This is followed by a description of the design and implementation process in Section 4. Section 5 describes the final user interface and system design. Section 6 discusses the evaluation techniques used, and the results obtained from the evaluations. The paper finishes with a discussion on ethics in Section 7, conclusion in Section 8, and a discussion on future work in Section 9.

2 BACKGROUND AND RELATED WORK

2.1 Data mining interfaces

2.1.1 Cluster-based data mining systems with job- and resource-management. One of the most popular open-source data mining software platforms is WEKA [1] [10]. Talia et al. [11] describe an extension to the WEKA system, called Weka4WS [2]. This extension was made to support distributed data mining on grid environments. The system utilizes the Web Services Resource Framework (WSRF

[1] https://www.cs.waikato.ac.nz/ml/weka/
[2] http://scalab.dimes.unical.it/weka4ws/about/

[3]) to achieve this. This framework includes certain standards that handle resource allocation and work flow management. This is done with the help of components and properties such as Service Groups, Resource Lifetime, Resource Properties, and Job Properties. The system is divided into user-nodes and computing-nodes. Different Weka4WS clients are installed on the user- and computing nodes. The system coordinates how resources are divided between the different client jobs, and notifies users whenever there's a change in the state of their active jobs. This is a good example of a distributed framework operating in a peer-to-peer network environment. This contrasts with the system described by Medvedev et al [12] (DAMIS), which is hosted on a central Web server, which also contains the online data repository. This is different from the Weka4WS system, where the client-nodes are responsible for storing the data once it's been processed. The DAMIS system constructs scientific work flows for the data mining process, and lets users view and edit the process through the use of a drag and drop interface. This also contrasts with he Weka4WS system, where most of the back-end work flow management is closed off to the user. The centralized server-client architecture of the DAMIS system means the system is divided into two logical units, namely the Graphical User Interface (GUI) and the Computational Service Component (CSC). Communication with these components is achieved through the use of SOAP (Simple Object Access Protocol) messages [4]. Unlike the Weka4WS system, all job scheduling and resource allocation is performed by the CSC. The CSC is also the gateway for communicating with a separate computer cluster, which is used to execute the data mining jobs. The most important components in the CSC are the Service Request Listener and Batch Job Invoker. The Listener receives the data, as well as the job parameters. Once it has determined that the parameters aren't faulty, it sends a request to the Batch Job Invoker and waits for the job to be completed. The DAMIS system's architecture was a good starting point for the design of this paper's system, because of the centralized nature, and the usage of a separate cluster environment.

2.1.2 Social media data mining interfaces. Zhang et al. [2] developed a pattern based topic detection and analysis system that analyzes tweet-like data from the Weibo website (a Twitter-like platform in China). The system is called PTDAS. It consists of three sub-processes, namely the data acquisition, the topics detection/analysis, and the topics presentation. Like the system described in this paper, the PTDAS system utilizes Hadoop as part of its storage system, but uses Spark instead of Hadoop for the data mining jobs. The system provides a good example of what the system architecture and work flow functionality of a social media data mining system looks like. The front-end, however, wasn't a Web interface. For an example of a system that analyzes Twitter data and displays the results in an Ajax-based HTML dashboard, we look at the system described by Wang et al. [3].It is a system that analyzes Twitter data in real time to do sentiment analysis of the 2012 United States presidential election cycle. They built their system on the IBM InfoSphere Streams platform. The dashboard displayed graphs that depict the volume of tweets related to a candidate, as well as the sentiment towards a candidate.

[3]https://www.service-architecture.com/articles/web-services/web$_{s}ervices_{r}esource_{f}ramework_{w}srf.html$
[4]https://www.tutorialspoint.com/soap/soap_message_structure.htm

2.2 Visualizations

Couturier et al. [4] developed a system that lets users run Association Rule mining algorithms, and then presents the results in a clustered 3-D (3 dimensional) visualization (see Figure 1).

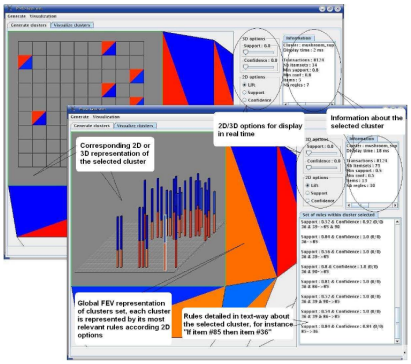

Figure 1: Example of how association rules are presented in a 3D bar-graph [4]

Hahsler [5] presented an interactive system based on R (a software environment for statistical computing and graphics). He described three main methods for visualizing association rules, which served as the main inspiration for the visualizations used in this project. These strategies are displayed below in Figure 2. The scatterplot in Figure 2 (a) provides an easy way of spotting outliers in the data mining results. The grouped matrix visualization (Figure 2 (b)) represents the support of the association rule using the size of the square where the two itemsets intersect. The network graph in Figure 2 (c) is a useful method for detecting groups of rules that are implicitly related.

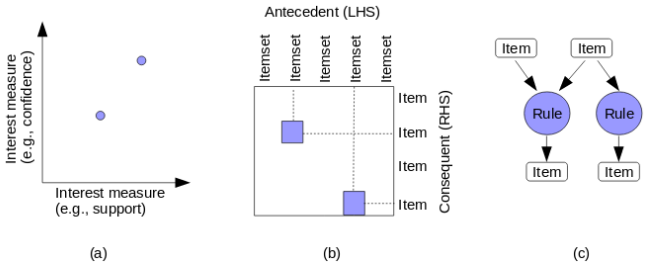

Figure 2: The main visualization components: a scatterplot, matrix visualization, and network-based visualization [5]

3 REQUIREMENTS ANALYSIS

3.1 Process

When the project was started, the initial requirements were gathered by sending surveys to three potential users of the system. They were two UCT postgraduate Computer Science students and

a postgraduate Engineering student from UCT who had worked with data mining systems before. They were chosen because of their prior experience and for the sake of convenience. The survey posed questions such as: What kind of information would they expect to see on the home screen of a data mining interface? What is the process for running queries and kicking off jobs in these systems? How are the data mining results usually displayed? From their responses, we gained useful insight into what users would expect from the system and what kind of functionality needed to be built.

3.2 Result

The following front-end requirements were agreed upon:

- Each user must have their own profile and not have access to the datasets and jobs of other users.
- The interface must provide a method for users to upload datasets to the server.
- Users must be able to see a list of ongoing and completed jobs.
- The system must display an estimate of when a job will be finished.
- The interface should provide a method for ring-fencing the data through the execution of queries. These queries include searching for hashtags, usernames, or individual words.
- Data mining results have to be displayed in a table format, with filtering and sorting options on the tweet content, the support levels, and the confidence levels.
- The interface must provide useful visualizations of the rules generated from the Association Rule mining.

The use case diagram (Figure 3) summarizes the system's functional requirements.

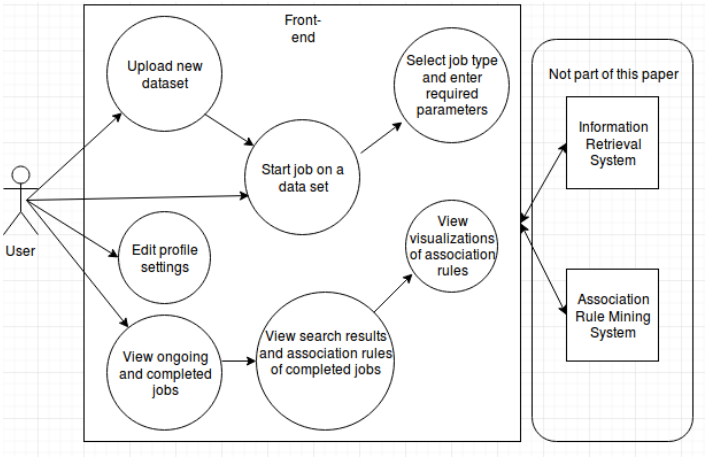

Figure 3: Use case diagram

4 DEVELOPMENT STRATEGY

4.1 Development Team

The project team had three members: Matt Young, Eric Dai, and Pieter van der Walt. All three are Computer Science Honours students at the University of Cape Town. Matt Young developed the algorithms for indexing and performing Information Retrieval on the Twitter dataset. Eric Dai developed the algorithm that performs Association Rule mining on the dataset. Pieter van der Walt developed the front-end for the system, which handles the separate user profiles, lets users create data mining jobs, as well as view the results in a table format and in terms of useful visualizations.

4.2 Development Methodology

The project consisted of a small team, and had strict time constraints. It also featured a front-end that needed to be usable. Because of this, the Rapid Application Development (RAD) method was chosen and applied throughout the project [6]. The key objective of RAD is high quality systems, fast development and delivery, and low costs [6]. When designing a user interface, the RAD method promotes an iterative, user-centered design process. User-centered design is a broad term used to describe design processes in which end-users are involved early in the design process in order to influence how a design takes shape [7]. This ensures that the system is developed in accordance with user expectations and needs. Instead of trying to design an interface from scratch and presenting it to users at the end of the project, developers create prototypes and present these to users. The users then comment on the prototypes, the developer makes the changes, and then presents the revised version. This iterative process prevents the wastage of time and other resources when developers need to make changes to the actual system instead of a simple prototype.

4.3 Development Environment and Tools

The user interface is a Web-based interface and was designed to be used in popular Internet browsers such as Google Chrome, Safari, and Firefox. This is because they are used by the largest percentage of the population [8]. A Web-based interface was chosen because it is the most convenient way for users to access the system. The interface required a front-end as well as a back-end server. The server starts up the cluster (Amazon Web Services Elastic Mapreduce [5] cluster) and uses the Information Retrieval and Association Rule mining components of the project to run jobs on the cluster. It also acts as the intermediary between the front-end and the cluster. To implement the server-client setup, the NodeJS [6] server environment was utilized. NodeJS is a free, open-source Javascript runtime environment that can run on various platforms (Windows, Linux, Mac OS). It was chosen because the fact that the server is also written in Javascript made the communication between the front-end and the server easy and convenient. This reduced the amount of research and skill development that was needed before the system could be developed. Additionally, NodeJS has a built-in package manager (NPM [7]). This allows for many useful open source libraries/packages to be easily incorporated into the project. This

[5] https://aws.amazon.com/emr/
[6] https://nodejs.org/en/
[7] https://www.npmjs.com/

includes packages that allow for the easy manipulation of the file system, communication via sockets, communication with a NoSQL server, as well as encryption libraries, and useful graph libraries. The front-end was written in HTML (Hypertext Markup Language [8]), but a templating engine called Jade was used because it made the writing of the HTML code easier and was automatically included in the NodeJS setup. The styling of the front-end was done using CSS (Cascading Style Sheets [9]). The communication with the server was done using asynchronous Ajax requests. Javascript was also used in the front-end to render dynamic content and send/receive communications to/from the server. A document store database called Couchbase [10] was used to hold the user profile data, as well as handle session storage. This type of database was chosen because it can handle large amounts of unstructured and semi-structured data. NPM also had a package that made the integration of the Couchbase server easy and convenient.

4.4 Development Cycle

4.4.1 Initial Prototyping. After gathering the initial requirements, the prototyping phase was started. During the project's prototyping phase, two high-fidelity prototypes were developed and presented to potential users. Once the comments on the second prototype had been integrated, a third prototype was built, which was used as a model while building the interface. High-fidelity prototypes were developed because the interface consisted of many interactive components that would have been harder to simulate using low-fidelity prototypes. The prototypes were developed with a program called Pencil Project [11], which is a free and open-source GUI (Graphical User Interface) prototyping tool. The program was easy to learn and use, which meant that even though they were high-fidelity prototypes, developing the prototypes didn't take much longer than low-fidelity prototypes would have taken. With each iteration, the user's comments were incorporated into the next iteration, thereby refining the interface to be as usable as possible. These comments are discussed in Section 6.

4.4.2 First functional interface version. After the prototyping, the interface was developed according to the final prototype. Certain aesthetic changes were made to make the interface similar to other UCT websites in terms of colour scheme. Additionally, there was a problem with the implementation of one of the graphical representations of the Association Rules. The matrix visualization (see Figure 2: b) consisted of discrete values (as opposed to continuous values) on its axis, namely the right- and left hand itemsets of the association rules. No browser-compatible graphing library could be found that had this functionality. Google Charts, for example, only allows for one axis to consist of discrete values, while the other axis has to be continuous. For this reason, the visualization was replaced with a chord diagram.

4.4.3 Interface redesign. Though I had not realized it, the interface had a task-centric design, where the interface layout prioritized making tasks available to the user in a central location. However, most data mining systems adopt a data-centric approach for the user interface, because the data is the most important component of the system. In the data-centric approach, the interface's main priority is showing the user the available data, and letting them access the various tasks through the datasets, instead of grouping all the tasks together. The changes that were made are discussed in Section 6.3.

4.4.4 Usability testing. When the interface was complete, user testing was performed to evaluate its usability. Before the main testing, a pilot test was done to ensure that there were no problems with the test format and to check that all the instructions were clear. During the testing, the system's purpose was explained to the testers, and if they were not familiar with association rules, they were given an explanation. This is because it's important to understand the concept before trying to interpret the interface's visualizations. The testers were given a list of tasks to complete on the interface. The tasks were designed to cover every piece of functionality that the interface offers. The task list included tasks such as creating a profile, logging in, logging out, editing profile settings, uploading a dataset, creating a new job, viewing the table-based results of a job, manipulating the results using the sorting and searching functionality, and interpreting the visualizations of the association rules. The amount of time taken to perform the tasks, as well as the number of mistakes made while performing the tasks were recorded. The users were asked to think out loud and mention if anything on the interface seemed unclear or confusing. These comments were also recorded and are discussed in Section 6. After the test, users were also asked to complete a questionnaire that rated the interface based on Nielsen's Heuristics. Nielsen's Heuristics are ten measures that are commonly used in the Human Computer Interaction (HCI) field to measure an interface's usability [9]. The testers consisted of five Computer Science honours students, and five Computer Science students who were at the Masters or PhD level. They were chosen because they fall into the category of people who would utilize a system like this, and also because of convenience and the project's time constraints.

5 SYSTEM DESIGN

5.1 User interface

The final user interface's design can be seen in Figures 4 to 11. When a user logs in, they are shown a list of their active and completed jobs (Figure 5). On the Datasets tab (Figure 6) they can view their available datasets and upload new ones. When a user views the Information Retrieval results of a completed job, they are presented with Figure 7. Note that in these images an IBM dataset was used for testing purposes, hence the numbers instead of words in the document content. When a user wants to view the Association Rule results, they are presented with Figure 8. They can sort the results according to any field in the table, and also apply filters and adjust the rule priority slider. The Association Rule priority slider that can be seen in Figure 8 is used to determine which Association Rules to send from the server. This was needed because a job could potentially return thousands of rules, and it would be more resource-efficient and useful to the user to only send the most significant rules to the client-side. If the slider is all the way on the "Support" side, the significance of the rules will only be determined by their

[8] https://www.w3schools.com/html/html$_{i}ntro.asp$
[9] https://www.w3schools.com/css/
[10] https://www.couchbase.com/
[11] https://pencil.evolus.vn/

support levels. The same concept applies if it was positioned all the way on the "Confidence" side. Figures 9, 10, and 11 demonstrate the visualizations of the association rules that are available to the user.

The scatterplot in Figure 9 has the support levels on the horizontal axis, and the confidence levels on the vertical axis. When a user hovers over a dot on the scatterplot, it will display the rule's content, as well as its exact support and confidence level. This visualization makes it easy to spot outliers that have a high confidence level, as well as a high support level. The chord diagram in Figure 10 splits the rules into their respective left-hand side and right-hand side itemsets. These itemsets are then placed around the circle's parameter. When a rule exists that binds two itemsets together, it is represented in the form of a chord between the two itemsets. The higher the support level for the rule, the thicker the chord will be. This visualization makes it easy to spot itemsets that form part of many different sets of rules. This is more difficult to spot when viewing the rules in the form of a table. When a user hovers the mouse over a specific itemset, the graph will adjust to display only the itemsets that are connected to the itemset that's being hovered on. The network graph in Figure 11 represents itemsets as nodes on the network, and rules between itemsets as edges between the nodes. The higher the support level for the rule, the thicker the edge will be. The confidence levels for the rules are displayed according to the colour scale, which is shown above the visualization. While the chord diagram is good for spotting the direct connections between itemsets, the network graph lets the user spot itemsets that might be indirectly connected. When a user clicks on a node and drags it, all the nodes connected to the selected node will move and adjust as the selected node is dragged around. The combination of these three visualizations may provide the user with interesting insights that may not have been possible without the help of the graphs.

Note that every page contains instructions at the top that instruct the user on how to utilize the page's functionality. It is assumed that these instructions will assist first-time users of the interface, and help them when they feel stuck. Additionally, the pages that contain the results of the data mining jobs (Figures 7 to 11) display the job name, as well as all the job's parameters. This is in line with the Nielsen's Heuristic [9] "Recognition rather than recall", which states that the interface should show the user all the information they might need, and not rely on the user having to remember anything.

Navigation through the interface is done mainly by clicking on the appropriate links in the navigation bars. The button that's used for logging out is placed on its own on the right-hand side of the main panel because that's where most interfaces place the button, and therefore users would expect it to be there. Overall, the design aims to make navigation through the interface easy via the navigation bars, while also presenting the available functionality in an intuitive manner.

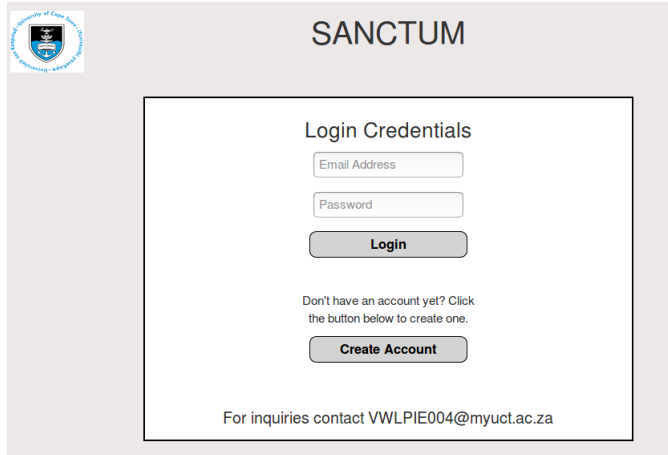

Figure 4: Final design: Login Page

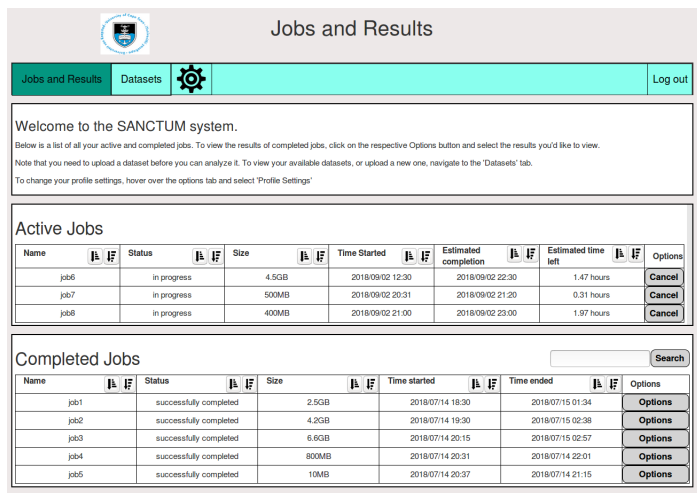

Figure 5: Final design: Home page

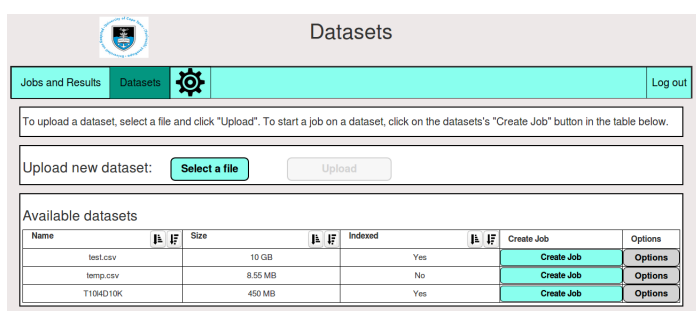

Figure 6: Final design: Datasets page

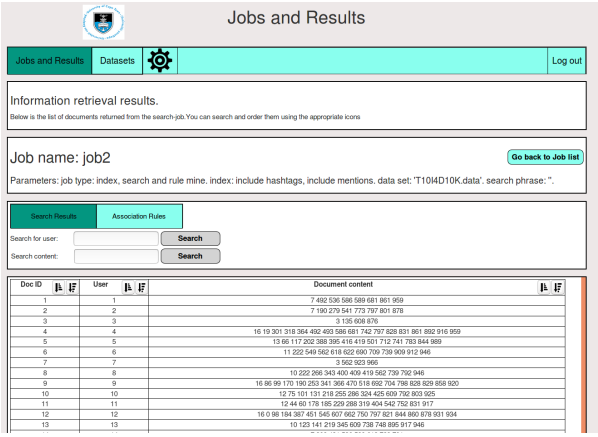

Figure 7: Final design: Information retrieval results

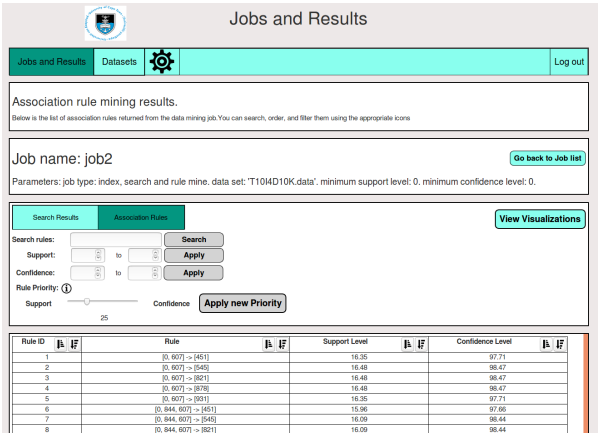

Figure 8: Final design: Association Rule Mining results

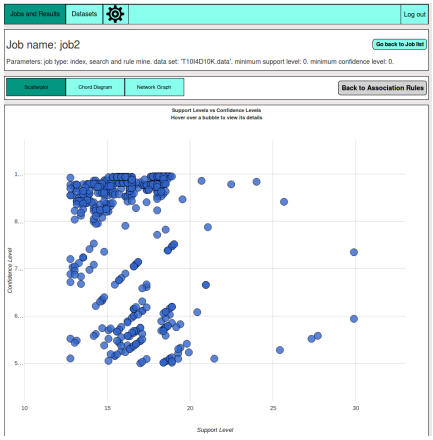

Figure 9: Final design: Scatterplot visualization

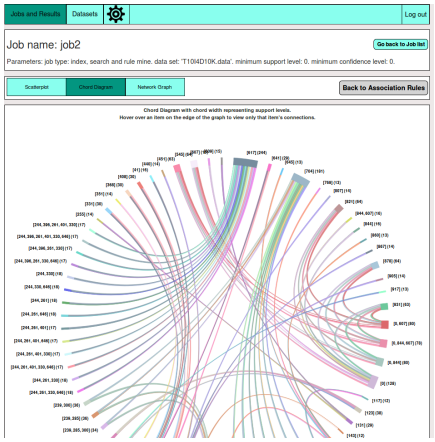

Figure 10: Final design: Chord Diagram visualization

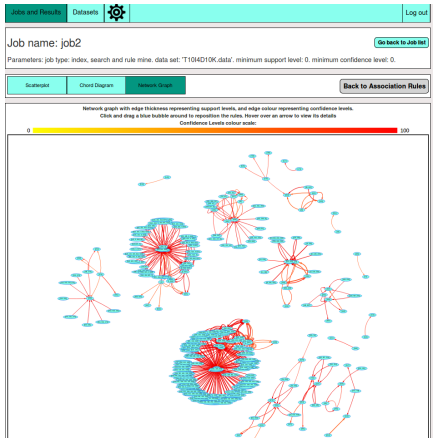

Figure 11: Final design: Network visualization

5.2 System overview

5.2.1 Package Diagram. The diagram in Figure 14 depicts how the different components of the system are wired together to form the final product.

The system consists of a back-end server and a front-end Web interface. The server communicates with the database and the local file system. It also manages the cluster, sends jobs to the cluster and receives the results once the jobs are completed. Communication with the database and the file system is facilitated by NPM packages called "connect-couchbase" and "fs". Management and communication with the cluster is done through NPM packages that contain the Amazon Web Service CLI (Command Line Interface) and APIs (Application Programming Interface). A package called "node-ssh" allows for ssh (Secure Shell) commands to be sent to the cluster, which is used to send jobs and receive results.

The front-end of the system is divided into the interface components and the package that handles communication with the server and renders dynamic content on the interface. The "Interface components and styling" package contains the HTML and CSS files that specifies the user interface's layout.

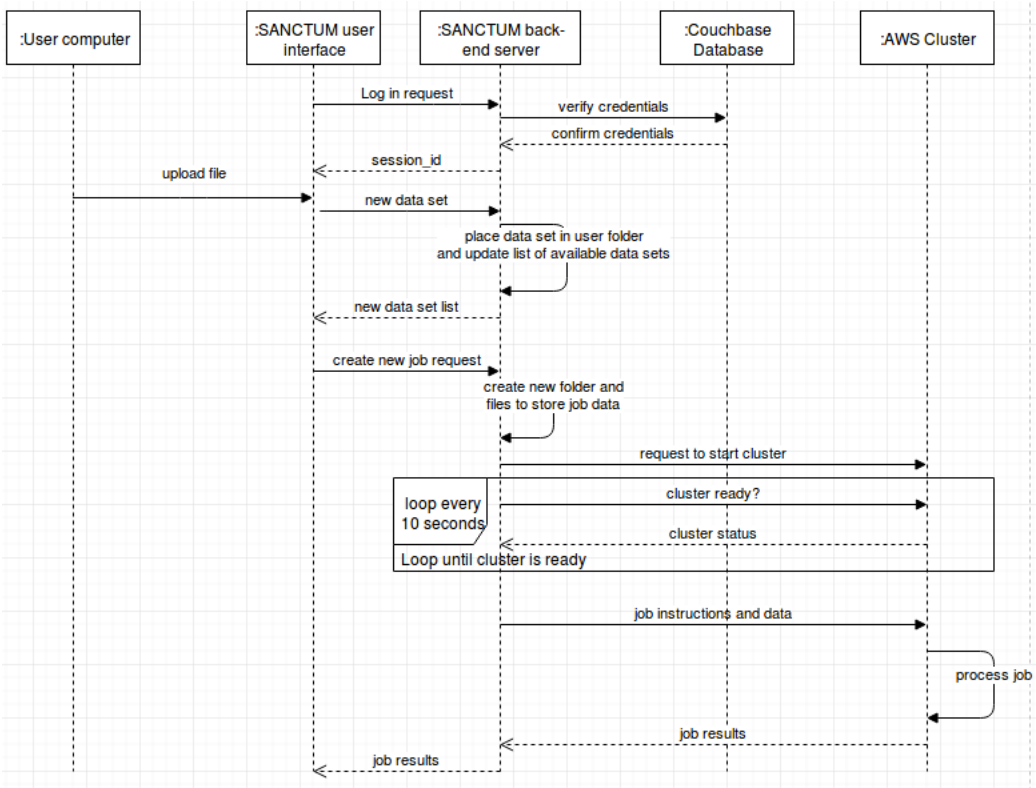

Figure 12: dataset upload and job start sequence diagram

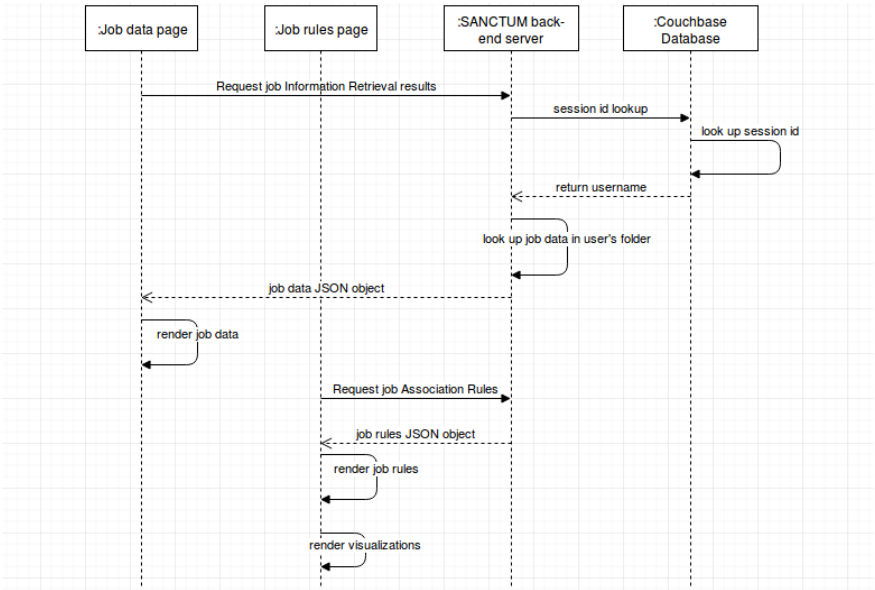

Figure 13: View results sequence diagram

The "Server Communication and Dynamic Content rendering" section contains a Javascript file. The server and the front-end communicate via a combination of HTTP (Hypertext Transfer Protocol) requests and AJAX (Asynchronous JavaScript and XML) requests. The server listens on port 3000 and responds when a request arrives. The HTTP requests are used to request specific pages on the website, and the AJAX requests are used to request, receive and render dynamic content on the front-end.

5.2.2 Front-end logic. The diagram in Figure 15 provides more detail about the functionality that's built into the front-end Javascript package. It is divided up per Web-page.

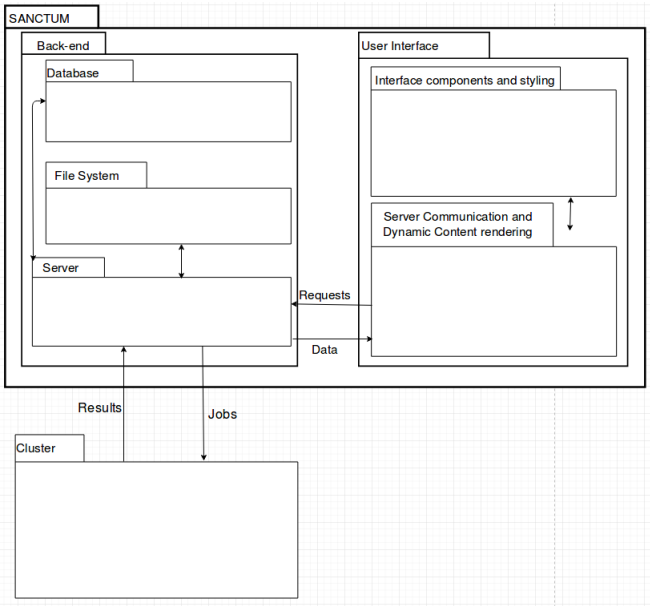

Figure 14: Package diagram

5.2.3 Sequence Diagrams. The sequence diagram in Figure 12 describes the process that takes place when a new dataset is uploaded, and a job is then created on the dataset. Figure 13 describes the sequence that occurs when the results of an already-completed job is requested.

6 RESULTS AND FINDINGS

This section will discuss the findings from every phase of the development cycle, namely the first and second prototypes, the interface redesign, and the final user testing.

6.1 First prototype

The first prototype was presented to three potential users, and their comments are presented in this section. The prototype can be seen in Appendix A, Figures 16 to 21. Note that the "Results" page (Figure 21) displays the content that would be displayed on the page once a user selected a completed job from a list very similar to the one on the "Current Jobs" page (Figure 20).

The reasoning behind the design was that when a user logged into the system, the most commonly used functionality would be

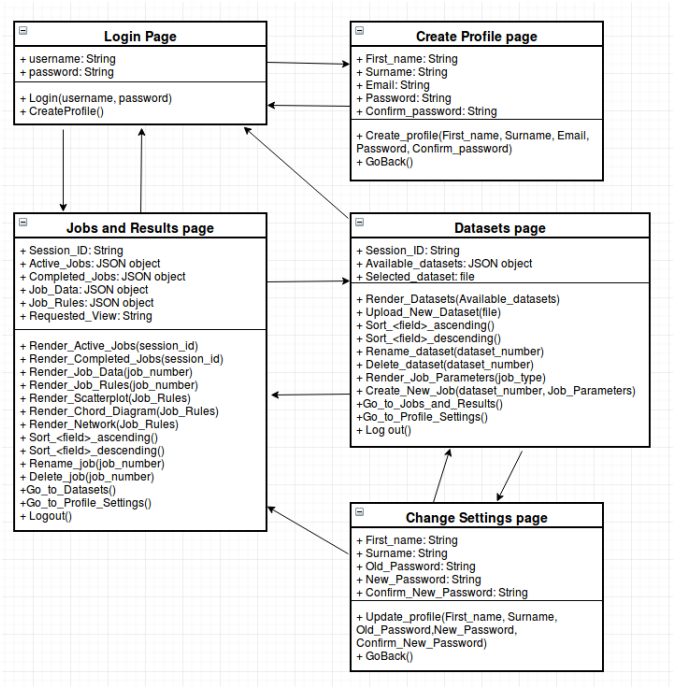

Figure 15: Front-end functionality

to either view the results of a recently completed job, or to kick off a new job. Therefore, the Home page contained a section that showed a list of the most recently completed jobs, with buttons that would take the user to that job's results. It also contained a section where users could kick off a new job by entering the appropriate parameters. It was theorized that the next thing a user would want to do, is check which jobs are still being completed, hence the "Current Jobs" tab was added. The "Results" tab would contain the results of every completed job, and not just the recently completed jobs that are displayed on the Home page. At this point in time, the best method for displaying the results of the jobs was still being debated, and therefore the results page didn't contain much detail regarding the tables and visualizations that were used later. When comparing the first prototypes to the final version of the interface, it becomes abundantly clear how important and valuable the iterative, user-centered design process is. The comments on the first prototype are shown below:

- The users pointed out that the list of recently completed jobs on the Home page was taking up a lot of space, and could be shown in a dropdown box instead.
- They suggested that instead of displaying the completed jobs and current jobs on separate pages, it would be best to combine them into one "Jobs" page
- They pointed out that the interface didn't present a way for users to indicate a job name, which is an important piece of functionality that was overlooked when the prototype was built.

- The interface didn't provide users with a way to upload datasets to the server, another important piece of functionality.
- It was suggested that the Home page should contain an introductory paragraph that explains the layout and functionality of the interface.
- The users inquired about the possibility of being able to cancel current jobs, as well as rename or delete completed jobs.
- It was pointed out that there was no button available for logging out.
- It was suggested that the "Profile Settings" tab be changed into a dropdown that contained the link to the Profile Settings page, as well as the logout button.

Overall, users felt that the design needed a lot of work.

6.2 Second prototype

The 2nd prototype can be seen in Appendix A, Figures 22 to 29. Note that reducing the size of the list of recently completed jobs on the Home page allowed for the addition of a data upload section, as well as an introductory paragraph. The "Jobs" and "Results" tabs were merged into one, so the current and completed jobs are displayed together. Additionally, searching and ordering functionality was added to the list of completed jobs.

The prototype was presented to the same three potential users who looked at the first prototype. The main comment on the second prototype was that the introductory paragraph on the Home page should be split up, so that each functional box on the Home page has its own paragraph instead. During this demo session, one user pointed out that a user might want to queue more than one job on the same dataset (for example, Searching and then running Association Rule mining on the results). Therefore the "Job Type" input, as well as the job parameters section on the Home page needed to change to accommodate for that.

6.3 Interface redesign

The interface that was developed before the switch from a task-centric to data-centric approach can be seen in Appendix B, Figures 30 and 31. Note that the data displayed in the "Jobs" tab was just sample data, since some back-end functionality hadn't been built yet at this point. Some suggestions were made for improving the interface, and the most significant one was the conversion from a task-centric to a data-centric approach. The interface was reworked and the results of those improvements can be seen in Appendix B, Figures 32 to 36. The following changes were made by changing the front-end HTML and Javascript-based rendering of the system. The Home page as it was displayed in previous iterations was removed. Instead, when users log in, they are shown the Jobs and Results page, which displays the list of active and completed jobs. There is now a separate page that displays the list of available datasets, and also contains the functionality for uploading a dataset. To create a new job, users now select "Create New Job" from the dropdown on the "Options" button displayed next to the dataset in the table. They are then shown a form that is very similar to the "Create New Job" form that used to be on the Home page. The ability to sort any column in the tables was added, so the need for a "View Recent Jobs" section disappeared.

6.4 Final user testing

6.4.1 Comments. The comments received from users during the final testing, as well as problem-areas that were identified by the test facilitators, are shown in Table 1. The comments were identified by looking at the tasks where users made the most mistakes and/or took the longest time to complete the task.

Table 1: User comments received during testing

Comment/Problem	Number of users
Users had trouble finding the button to create a new job, despite the instructions on the screen	7
Users thought adding a tooltip when hovering over the buttons that sort fields in the tables would be useful	6
Users thought they had to click on "Add Phrase" after entering a phrase in the search parameters	5
Some users felt the instructions for the graphs could be clearer	5
When asked to create a profile, some users mistook the login page as the page to create a profile on	4
Users tended to click on the "Upload" button before selecting a file	4
Users expected the "Options" and "Log out" button to be on the right-hand side of the navigation bar	4

The user comments were considered, and some changes were made to the final interface to rectify the identified problems.

The section of the interface where users had to click to create a new job on a dataset can be seen in Appendix B, Figures 34 and 35. It was decided to move the "Create new Job" button to its own column in the table instead. This can be seen in the final version of the interface, on Figure 6. In retrospect, it makes sense to place the functionality that will be used most often on its own. The "rename" and "delete" functionality for datasets are likely to not be used nearly as often as the "Create new Job" functionality. This is in line with an important HCI concept, called the 80/20-rule (or Pareto's Principle). This rule states that 20% of an interface's functionality will be used 80% of the time, while 80% of its functionality will only be used 20% of the time [13]. Therefore, commonly used functionality needs to be clearly visible and require the fewest number of clicks, while functionality that's used less often can be placed behind dropdown menus. Tooltips that contain the words "Sort Ascending" and "Sort Descending" were added to the buttons in the interface's tables' column headings (see Figures 5 to 8). The "Add Phrase" button that caused some confusion during the testing can be seen in Figure 36 (Appendix B). After consulting with the users who made the comments, it was decided to change the text of the button to simply show a plus sign (+). The users felt this removed the

confusion of thinking one has to click on the button to add the current phrase to the search, and also made it obvious that to add additional phrases, one has to click on the plus sign. The instructions for the graphs were rewritten to include more details and be more explicit. The login page, which had remained virtually unchanged since the second prototype (see Figure 22 in Appendix A), caused some confusion during the user testing. Users thought that by entering their email address and password into the input fields that are used for the login, and then clicking "Create Profile" they would create a new profile. The page's layout was therefore changed to what can be seen in Figure 4, in the final design section. A heading with the text "Login Credentials" was added to make it clear that those input boxes are meant for logging in. Additionally, the "Login" and "Create Account" button positions were changed to be below each other, instead of next to each other. Additional instructions were also added above the "Create Account" button to make it even clearer. The section in the "Datasets" tab where users can upload their datasets was also changed. The "Upload" button was changed to be inactive, and have a faded background colour whenever a file had not been selected. This would prevent users from clicking on the "Upload" button before clicking on a file. Additionally, the "Options" tab that contained the links to the user settings page as well as the Log out button, was removed. Instead, the Log out button was placed on its own on the right-hand side of the navigation bar.

6.4.2 Nielsen's Questionnaire. The interface that resulted after the switch to the data-centric approach was scored based on Nielsen's heuristics [9]. The average scores (with a maximum score of 5) for each heuristic are displayed in Table 2.

Table 2: Heuristic Scores obtained from user testing

Heuristic	Average Score
Visibility of system status	4.1
Match between system and real world	3.6
User control and freedom	3.9
Consistency	4.4
Prevent errors	3.8
Recognition rather than recall	4.0
Flexibility and efficiency of use	3.9
Aesthetic and minimalist design	4.5
Help users recognize, diagnose and recover from errors	3.4
Help and documentation	3.3

Overall the users were happy with the flow and design of the interface. Some expressed interest in using the system once it was fully functional.

6.4.3 Visualizations. During the testing, the users were able to successfully interpret the visualizations by identifying outliers, and implicitly connected itemsets. The users felt that the visualizations that were provided were useful and helped with their interpretation of the association rule results.

7 ETHICAL, PROFESSIONAL, AND LEGAL ISSUES

Ethical clearance was obtained from the University of Cape Town's Science Ethics Committee. During user testing, users were fully informed about the testing process, and all ethical procedures were followed. A voluntary consent form was signed by all participants, and they were assured that their personal data would be kept safe.

8 CONCLUSION

This paper demonstrated that it is possible to design a system that makes the analysis of Twitter data convenient and easy. It showed that the analysis of association rules can be improved and made more intuitive by providing useful visualizations of the results. It also showed that it is possible to build a system that can automatically handle the entire process of starting a cluster, sending the data and job request to the cluster, and then retrieving the results. By showing how the interface changed and evolved during the iterative, user-centered design process, the paper has clearly demonstrated the value that can be gained from following this process. User feedback was successfully incorporated in the design in order to develop a more usable and useful system. The interface was evaluated based on Nielsen's heuristics [9], and although some heuristics received an average score, the overall feedback was very positive. The system can still be improved, and certain additional features could be built to make the system more useful. These are discussed in the Future Work section.

9 FUTURE WORK

A possible addition to the system would the ability to do sentiment analysis of a specific topic/person/organization on a set of Twitter data. This was initially pitched as something that would be nice to add if there was time, but unfortunately that was not the case. If the system had a way to perform sentiment analysis it could also provide a useful graph that displays how the sentiment towards the topic changed over time. Another improvement to the system would be to rework the storage of the datasets and job results. At the moment, the datasets and the results of completed jobs are stored on the local file system of the website's server. It would be more effective if the system used the Couchbase database for this instead. This is because the access speed from the Couchbase database will be faster than the access speed from csv files in the regular file system. While the Couchbase database was initially only implemented to facilitate the session handling and user profiles, it would also function well as a database for storing the job data. Initially, it was assumed that the system would be able to use the UCT ICTS (Information and Communication Technology Services) cluster to perform the data mining jobs. Unfortunately, the staff in charge of maintaining the cluster communicated that they would not be able to assist us. This led to the system having to adapt to utilize Amazon Web Service Elastic Mapreduce Clusters to run the data mining jobs instead. Though Amazon granted us some academic credit to use for the project, this would eventually start costing money. The system can therefore be improved in future by performing the necessary steps to make the UCT ICTS cluster usable and making the necessary adjustments to the system to utilize the local cluster instead of Amazon's rented clusters.

REFERENCES

[1] Pritam Gundecha and Huan Liu. 2014. *Mining Social Media: A Brief Introduction.* In INFORMS Tutorials in Operations Research. Published online: 14 Oct 2014; 1-17. DOI: https://doi.org/10.1287/educ.1120.0105

[2] Lu Zhang, Zhiang Wu, Zhan Bu, Ye Jiang, Jie Cao. 2017. *A pattern-based topic detection and analysis system on Chinese tweets*, Journal of Computational Science, 2017, ISSN 1877-7503, https://doi.org/10.1016/j.jocs.2017.08.016.

[3] Hao Wang, Dogan Can, Abe Kazemzadeh, FranÃğois Bar, and Shrikanth Narayanan. *A system for real-time Twitter Sentiment Analysis of 2012 U.S. Presidential Election Cycle.* Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 115-120, Jeju, Republic of Korea, 8-14 July 2012.

[4] Olivier Couturier, Tarek Hamrouni, Sadok Ben Yahia, Engelbert Mephu Nguifo. *A scalable association rule visualization towards displaying large amounts of knowledge.* 11th International Conference Information Visualization (IV'07) 0-7695-2900-3/07, 2017

[5] Michael Hahsler. *arulesViz: Interactive Visualization of Association Rules with R.* The R Journal Vol. 9/2, December 2017, pages 163-185. ISSN 2073-4859

[6] P Beynon-Davies, C Carne, H Mackay,and D Tudhope.*Rapid application development (RAD): an empirical review.* European Journal of Information Systems (1999) 8 , 211-223

[7] C Abras, D Maloney-Krichmar, J Preece. *User-Centered Design.* In Bainbridge, W. Encyclopedia of Human-Computer Interaction (2004).

[8] Browser Market Share Worldwide. Retrieved August 27, 2018 from http://gs.statcounter.com/browser-market-share#monthly-201707-201707-map

[9] Jakob Nielsen, Heuristic evaluation, Usability inspection methods, John Wiley & Sons, Inc. New York, NY, 1994

[10] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, Ian H. Witten. *The WEKA Data Mining Software: An Update.* in SIGKDD Explorations, Volume 11, issue 1, pages 10-18

[11] Domenico Talia, Paolo Trunfio, and Oreste Verta. 2008. The Weka4WS framework for distributed data mining in service-oriented Grids. Concurrency and Computation: Practice and Experience 20, 16 (2008), 1933âĂŞ1951. DOI:http://dx.doi.org/10.1002/cpe.1311

[12] Viktor Medvedev, Olga Kurasova, Jolita BernataviĂŋienĂŮ, Povilas Treigys, Virginijus MarcinkeviĂŋius, and Gintautas Dzemyda. 2017. A new web-based solution for modelling data mining processes. Simulation Modelling Practice and Theory 76 (2017), 34âĂŞ46. DOI:http://dx.doi.org/10.1016/j.simpat.2017.03.001

[13] Reh, F. John. "Pareto's principle-The 80-20 rule." BUSINESS CREDIT-NEW YORK THEN COLUMBIA MD- 107(7) (2005): p 76.

A FIRST AND SECOND PROTOTYPE DESIGNS

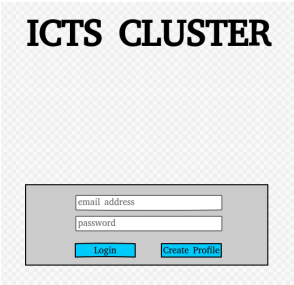

Figure 16: 1st Prototype: Login page

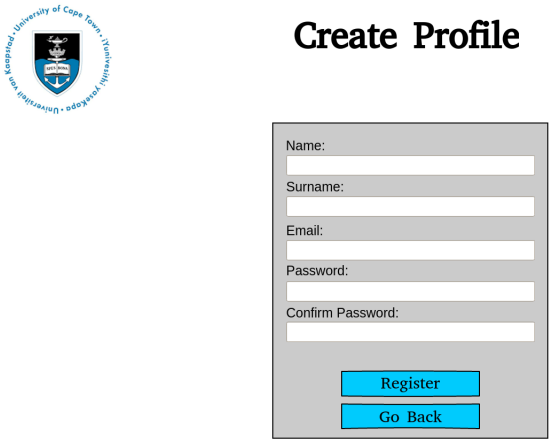

Figure 17: 1st Prototype: Create profile page

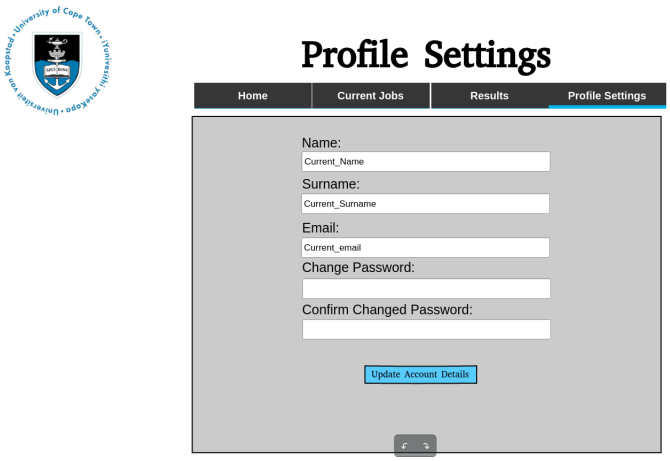

Figure 18: 1st Prototype: Change profile settings page

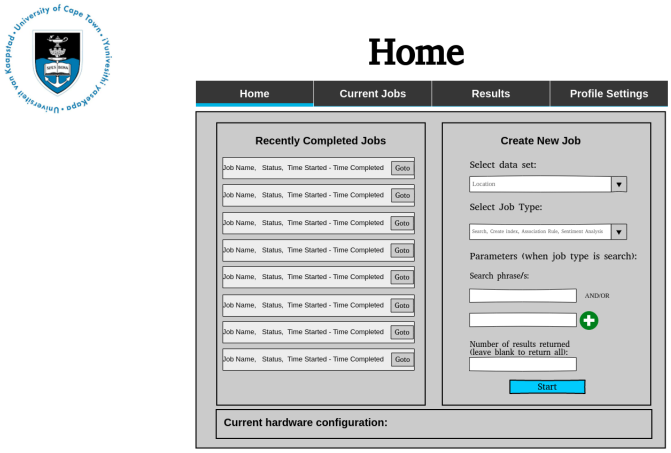

Figure 19: 1st Prototype: Home page

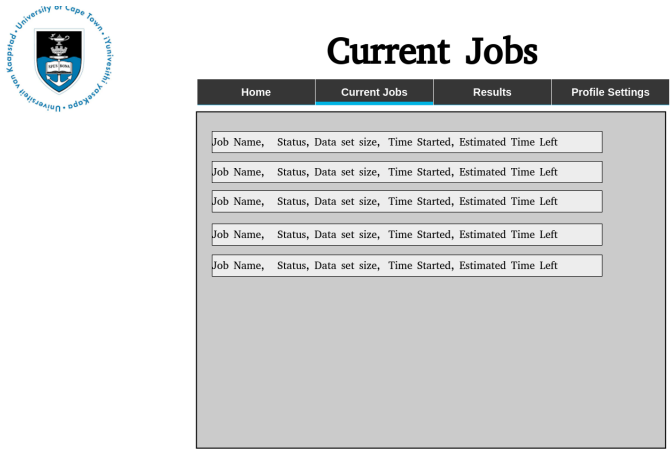

Figure 20: 1st Prototype: Current jobs page

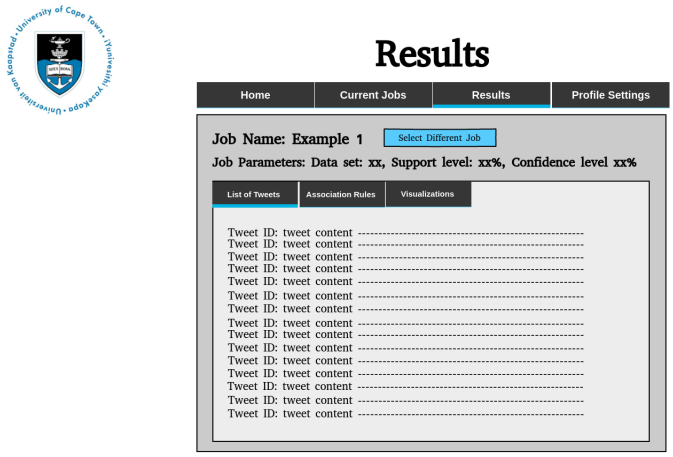

Figure 21: 1st Prototype: Results page

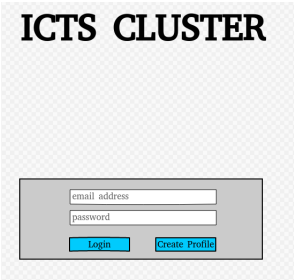

Figure 22: 2nd Prototype: Login page

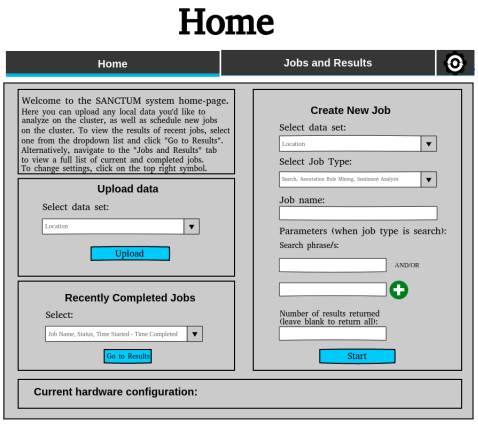

Figure 25: 2nd Prototype: Home page

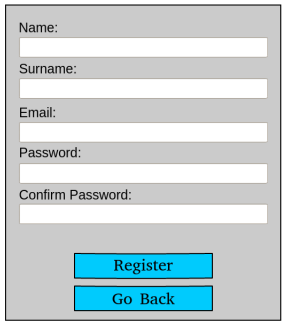

Figure 23: 2nd Prototype: Create profile page

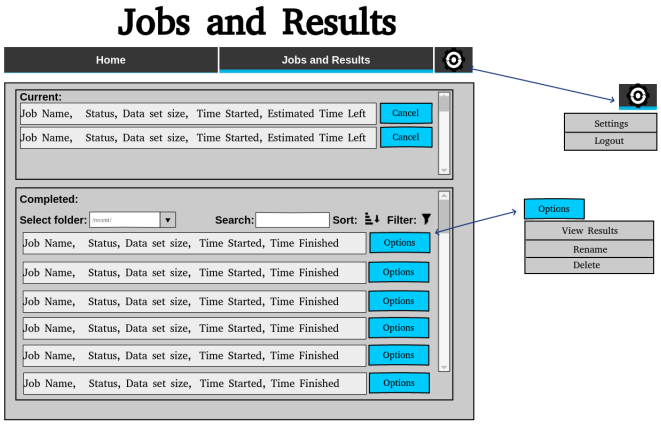

Figure 26: 2nd Prototype: Jobs and Results page

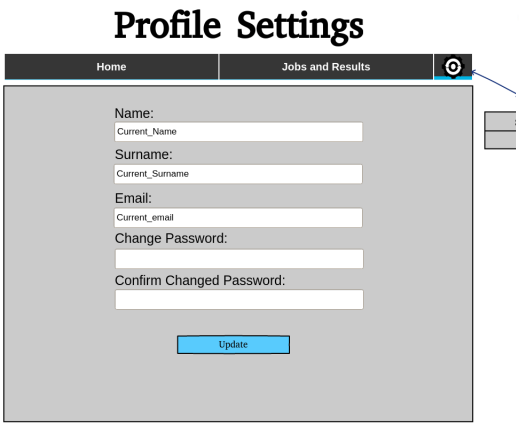

Figure 24: 2nd Prototype: Change profile settings page

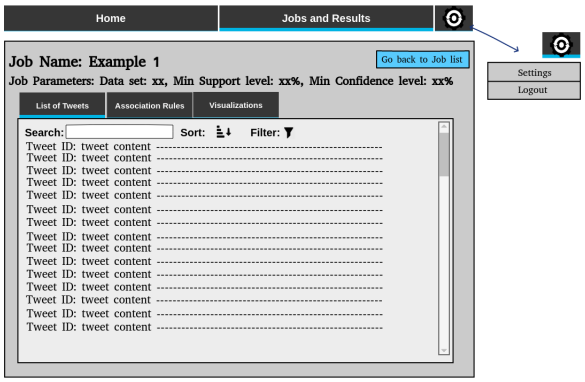

Figure 27: 2nd Prototype: Jobs and Results page, Information Retrieval results

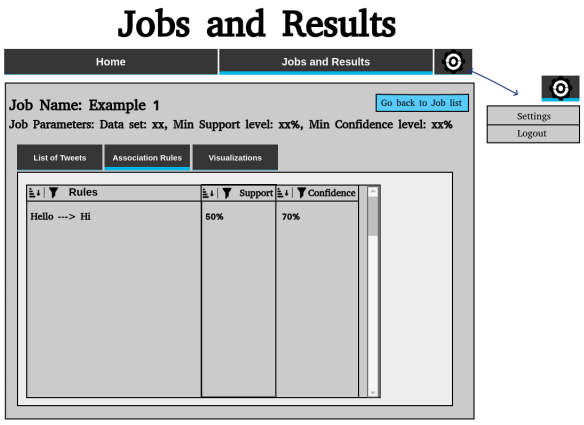

Figure 28: 2nd Prototype: Jobs and Results page, Association Rule results

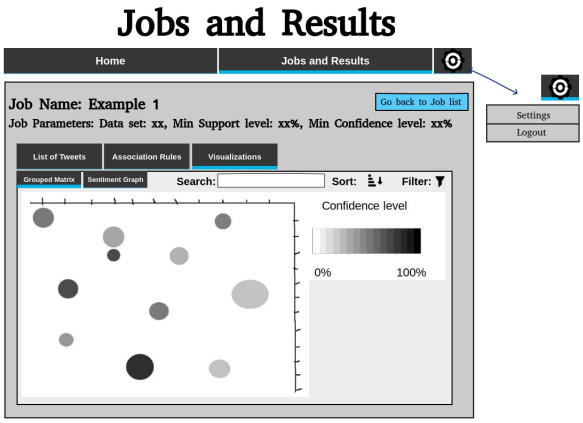

Figure 29: 2nd Prototype: Jobs and Results page, Visualizations

B BEFORE AND AFTER REDESIGN PHASE

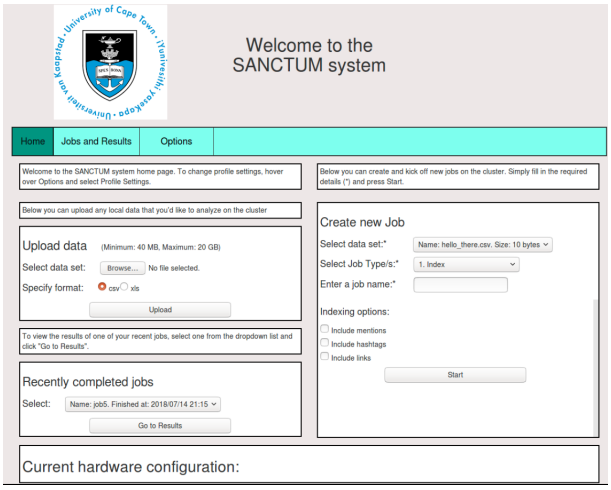

Figure 30: Before redesign: Home page

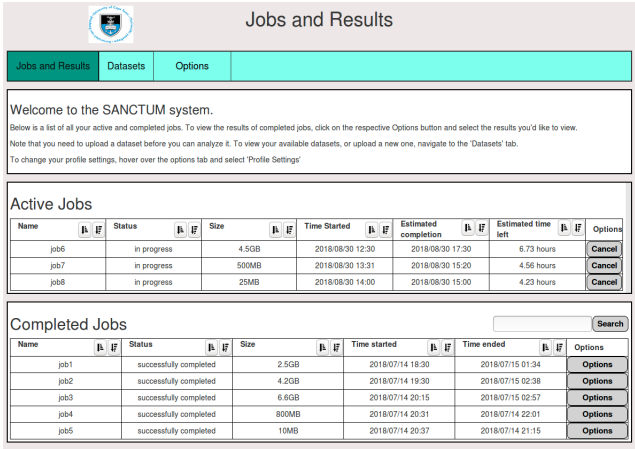

Figure 32: After redesign: Jobs and Results page

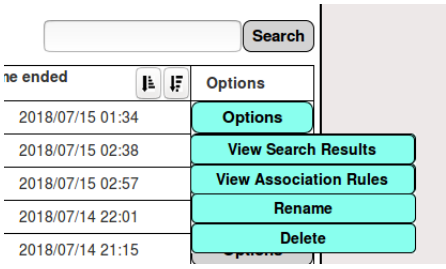

Figure 33: After redesign: Jobs and Results dropdown

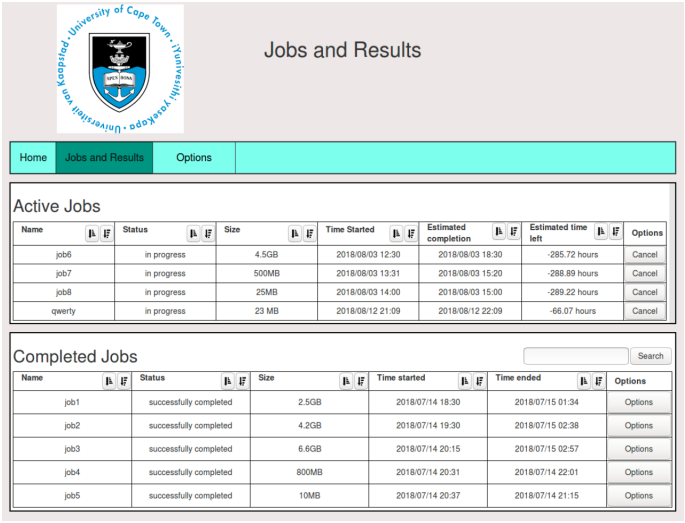

Figure 31: Before redesign: Jobs and results page

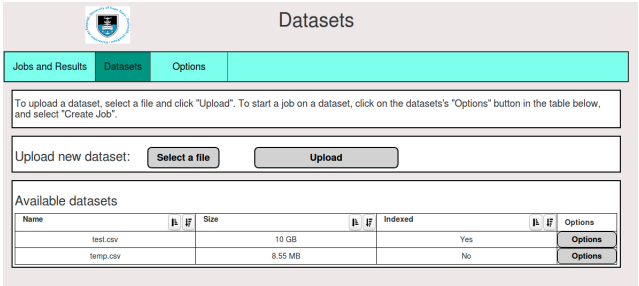

Figure 34: After redesign: Datasets page

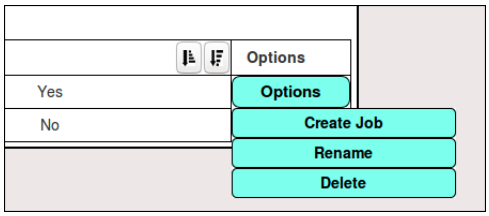

Figure 35: After redesign: Datasets dropdown

Figure 36: After redesign: Form for starting a job