**Cell Phone Mart**

CSCI 586 Project Report

Adarsh Rajanikanth, Malatesha Somasundar Anantha, Neelima Vangipuram, Sahil Wadhwa

**Group 1 Wednesday**

University of Southern California

1. **Background**

“There is a revolution occurring and it’s all about making the Web meaningful, understandable, and machine-readable, whether it is based on an intranet, extranet or, Internet. This is called Semantic Web and it will transition us towards a knowledge-centric viewpoint of everything.” The Semantic Web provides an infrastructure that enables not just Web pages, but databases, services, programs, sensors, personal devices and even household appliances to both consume and produce data on the Web [1]. URIs are a fundamental component of the current Web and are in turn a foundation for the Semantic Web. URIs, a fundamental component of Semantic Web provides the ability for uniquely identifying resources, RDF leverages URIs and the Web to provide a powerful means of expressing and representing the relationships and their meaning and Ontologies can be used to power advanced services such as more accurate Web search, intelligent software agents and knowledge management.

In our project titled Cell Phone Mart, we apply the concepts of Semantic Web and, by using Ontologies, integrate data from various sources to provide a unified view to the user. The data can be queried and used to extract information required by the user. We have provided queries that analyses various cellphone brands.

1. **Problem Statement**

There are many sources on the Internet that provide diverse set of disjoint information about Mobile Phones. **Kaggle**, one of the most comprehensive platforms that provides public datasets. The datasets on Kaggle are consistently accessible, and the forums helps to understand the nuances of the data. The data that we acquired from Kaggle had the information about unlocked mobile phones reviews from Amazon with attributes like – product name, brand name, price, rating, reviews, and review votes.

Similarly, **DeviceSpecifications & GSM Arena,** a comprehensive source that provides information about full specifications and features of various mobile devices such as – standard mobile phones and smartphones with attributes such as – dimensions, weight, SoC, CPU, Cores, RAM, Storage, Display, Battery, OS, Camera, SIM Card, Wi-Fi, USB, Bluetooth and positioning. **Wikipedia**, the most common and extensively used online source for anything contributed to the third data source to extract information about the Brand with attributes such as – type, founded in, headquarters, area served, other products, number of employees and its website link.

With data scattered around at multiple data sources, the user can’t get an integrated piece of information about a mobile phone of his choice at one place. For example, if a user wants to know which mobile phone had the highest rating with a certain screen size, he would have to sieve through all the sources. Likewise, identifying which mobile brand had the highest selling rate based on a key feature a user is interested in, becomes time-consuming otherwise.

In our project, we try to implement a few such key performance indices. We gather information about mobile phones from various data sources, unite and shape them, and provide a SPARQL endpoint. The results of the SPARQL queries are displayed on the browser using HTML and JavaScript. A provision for visualization is also provided which uses High charts to display charts and tables to display the query results.

Company Details

Queries

Mobile Handsets

**RESULTS**

**RDF**

Comments

Revenue

Figure 1. Workflow

1. **Scope**

In our project, we are focusing on data available from the following websites. The attributes crawled from each page are listed in Table 1:

1. [Kaggle](https://www.kaggle.com/PromptCloudHQ/amazon-reviews-unlocked-mobile-phones)
2. [Device Specifications](https://www.devicespecifications.com)
3. [Wikipedia](https://en.wikipedia.org/wiki/Main_Page)
4. [GSM Arena](https://www.gsmarena.com/)

We have also extracted the yearly revenue of Brands starting from 2014 to 2016 by web scrapping Wikipedia and the scrapping individual Brand pages which were not present in Wikipedia’s revenue data.

|  |  |
| --- | --- |
| **Website** | **Extracted Information** |
| Kaggle | product name, brand name, price, rating, reviews, review votes |
| Device Specification, GSM Arena | dimensions, weight, SoC, CPU, Cores, RAM, Storage, Display, Battery, OS, Camera, SIM Card, Wi-Fi, USB, Bluetooth and positioning |
| Wikipedia | type, founded in, headquarters, area served, other products, number of employees and its website link |
| Revenue Dataset | Profit\_Id, Profit\_2014, Profit\_2015, Profit\_2016 |

Table 1. Information extracted from websites

1. **Approach**

In this section, we discuss different phases of the development and execution of our project. The project can be divided into four phases.

* 1. Data Acquisition
  2. Data Cleaning
  3. Data Modeling & Integration
  4. Data Linking
  5. Querying

4.1 **Data Acquisition**

We implemented Java-based crawlers to extract data from the above-mentioned data sources by

1. Using jsoup library that provides an API to extract and manipulate data from the web pages we used this to extract data from various websites like Device specifications, GSM Arena, Wikipedia to obtain Cell phone’s technical specs, Detailed features, Brand info respectively.

2. For analysis of user reviews on individual models we got the users comments data set for cell phones on amazon from Kaggle which helped us to analyze each model based on user rating and comments in amazon.

|  |  |
| --- | --- |
| **Website** | **No. of records generated** |
| Kaggle | 4,13,000 records were generated for comments and review |
| Device Specification, GSM Arena | 8,632 models were captured for various brands of mobile phones |
| Wikipedia | 108 different brand information was captured |
| Revenue Data | 25896 records pertaining to the profits through years 2014 to 2016 |

Table 2. Statistics about generated data

Data Acquisition Challenges: Web Scrapping may sound simple but is accompanied with its own challenges. We faced the following:

* 9,000 records are from various above-mentioned Data Sources.
* Non-uniform structure and hence it took long time for data refinement making it available for mapping.
* Navigation through elements and attributes in an XML document

4.2 **Data Cleaning**

* Organized scrapped data by removing the duplicates rows from the CSV file by writing generic Java code.
* Removed some special characters and supported some by using specific data structures.
* Used specific datatype for handling website links/image URLs.
* Enforced UTF-Encoding for proper conversion of data to RDF.

4.3 **Data Modeling & Integration**

We created an Ontology for our model as shown in Figure 2. There are total of 4 classes namely – Brand, Model, Comments, and Profits. A total of 4 Object Properties – has\_brand, has\_comment, has\_model, profit\_has\_model. A total of 31 data properties. The object properties link different data sources.

has\_brand links the model data source with the brand data source.

has\_model links the comments data source with the model data source.

profit\_has\_model links the profit data source with the model data source.

There are 31 data properties from 4 data sources, each of these can be considered as the values associated with a particular data source. Figure 2 gives the complete details regarding both object and data properties.

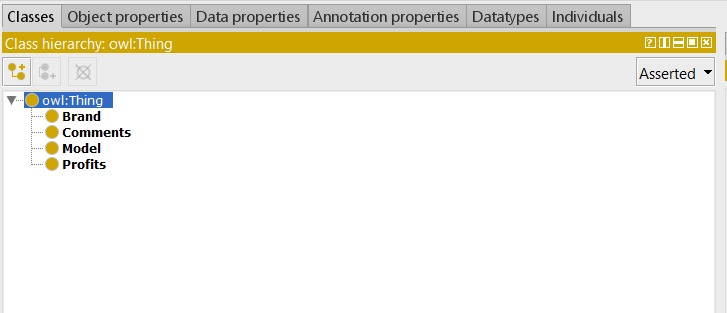
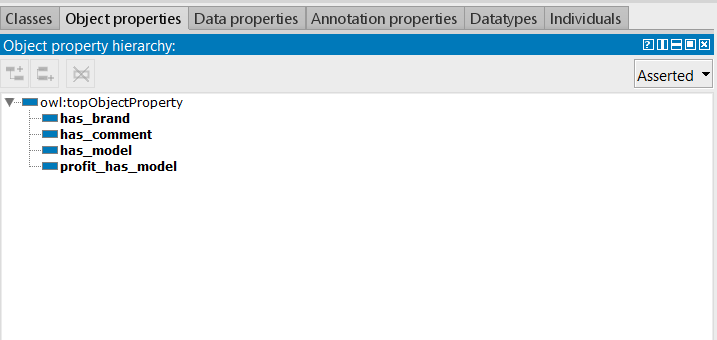
 

Figure 3. Classes Figure 4. Object Properties

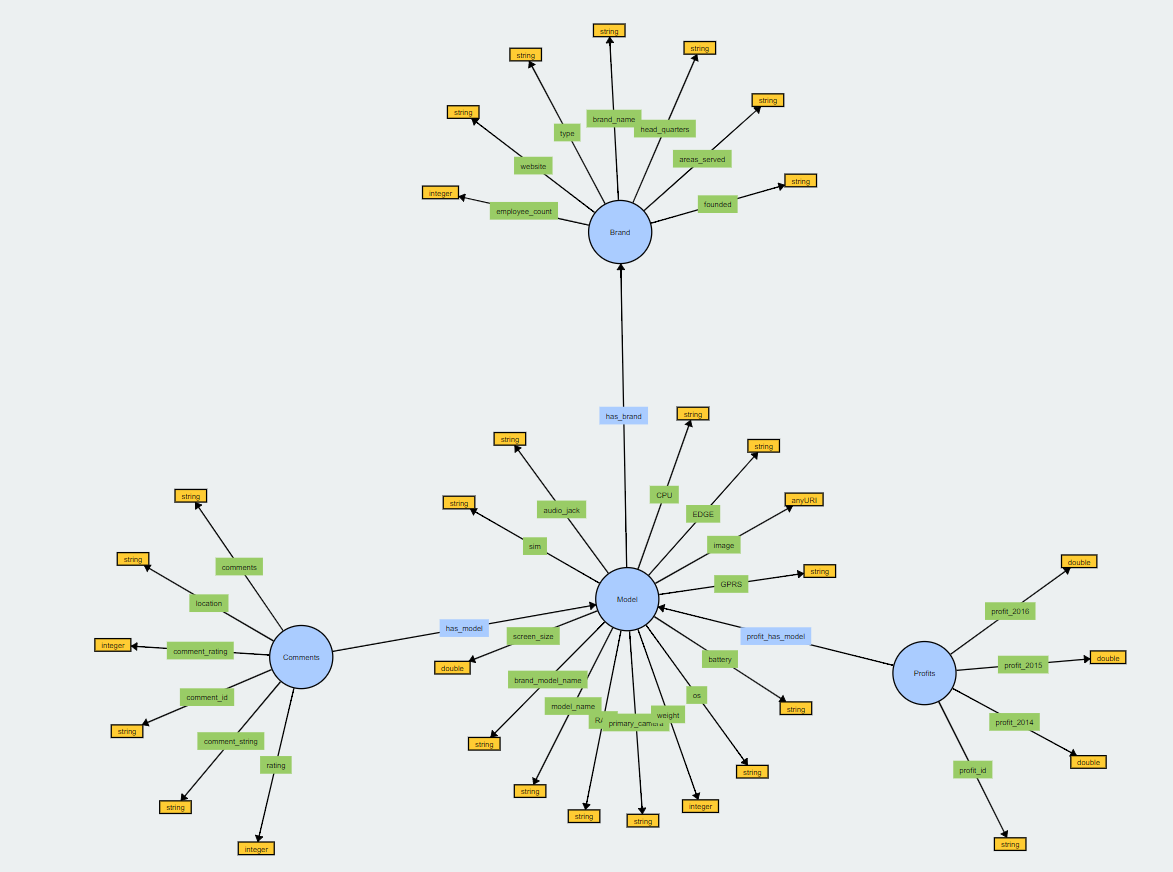


Figure 2. Ontology Graph

4.3 **Data Linking**

Data linking played a most prominent role in this application because the data was scrapped from various data resources and hence there was a requirement for linking or mapping of data in some way such that it makes sense. In order to do so we created a unique primary key in individual data set and then we linked them in owl using their URI so that we can obtain interlinked owl file containing mapped data. This was done by utilizing Protégé tool.

In below images we can notice that we have created unique id for individual data set and we have mapped them by utilizing their common attributes so that we can obtain N:1 and 1:N mapping between entities of mapped classes of different data sets.

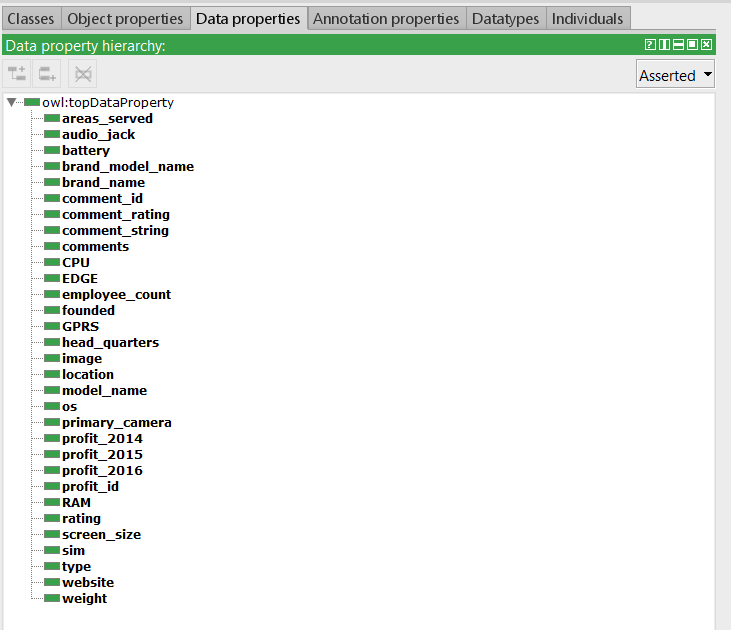
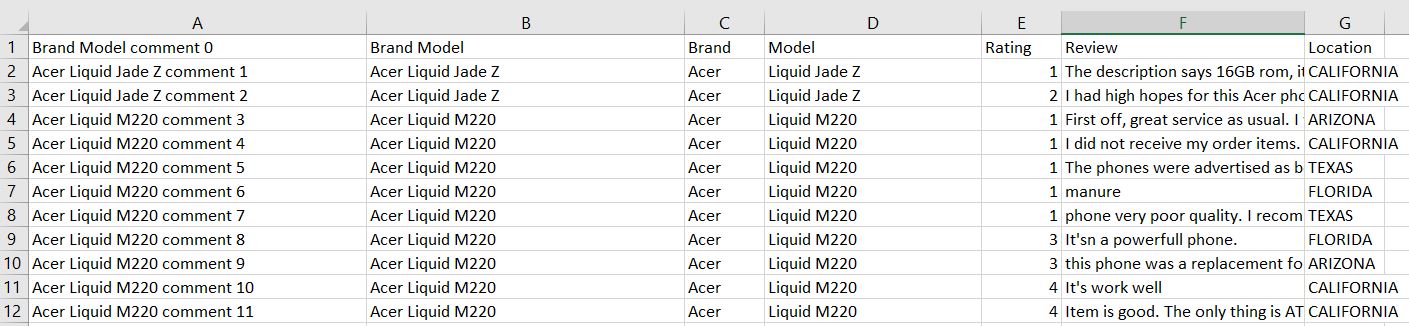
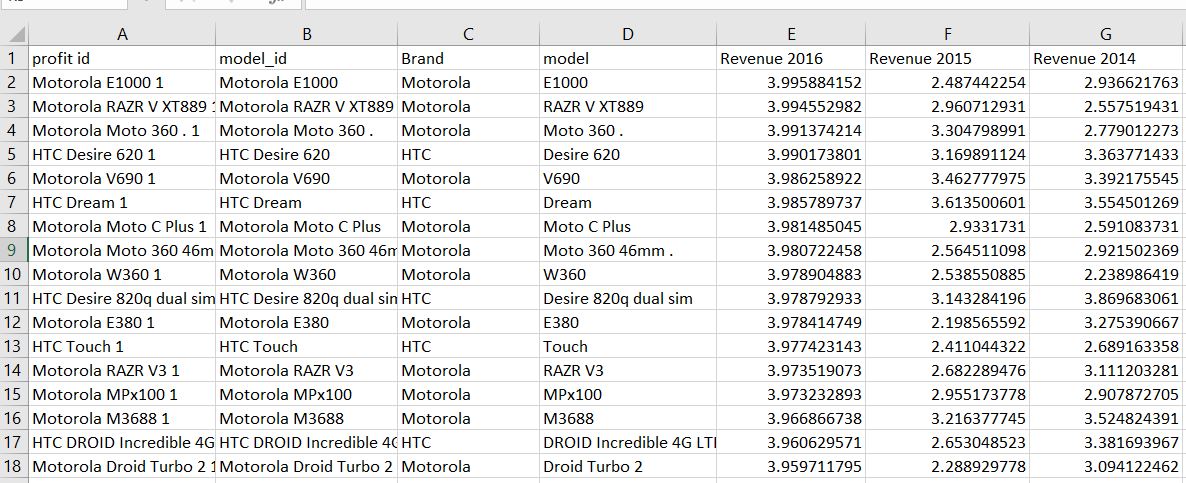


Figure 5. Data Properties

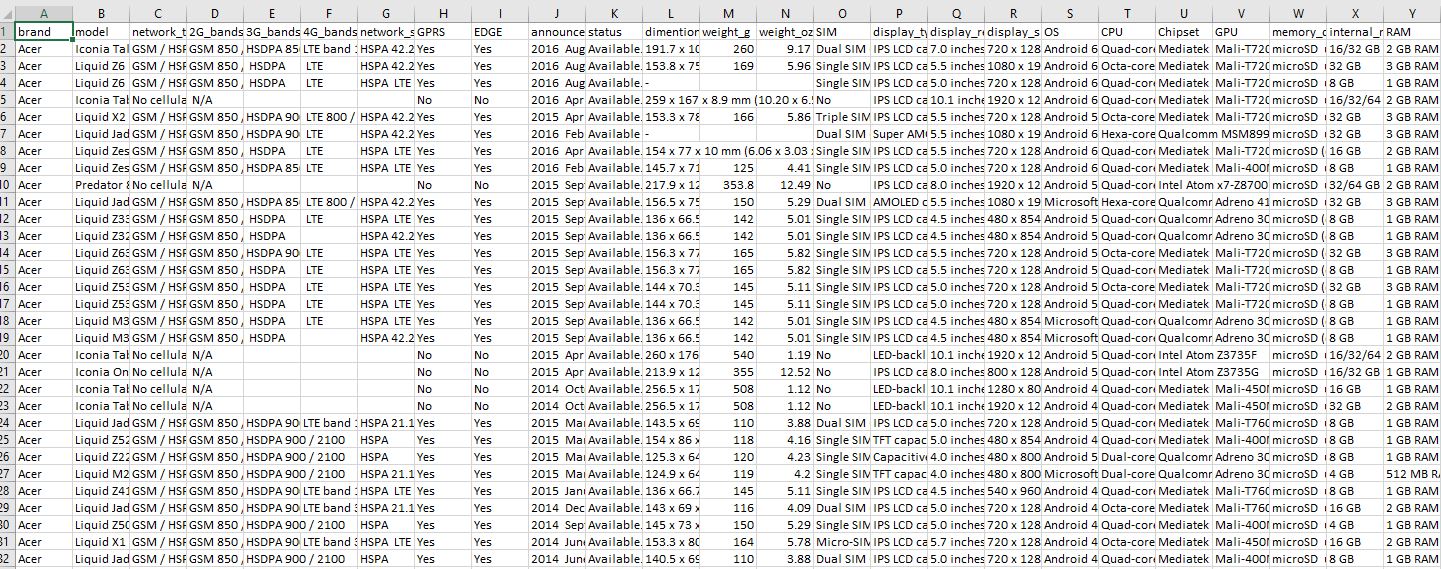
COMMENTS



REVENUE



SPECIFICATIONS



BRANDS



Figure 6. Generating a consolidated dataset for Protégé

By integrating all the above data sets we got a consolidated data set with mapped key value pairs, now we can use single query to get the resquired results from the consolidated dataset.

**Challenges in Data Linking** – Due to the huge heterogeneity in data, we faced a few challenges while linking these datasets. The dataset acquired for Comments and reviews about mobile phones had redundant entries for comments, we resolved this ambiguity by deleting the duplicates. Then we had to assign a unique id for each of the rows of different datasets for linking. We carefully chose the brand name as the unique key for the brand xls sheet, the concatenation of brand name and the model name was the unique for the model sheet. For revenue and comments we assigned a unique id by hashing all the column values of each row. We also observed that a brand can have multiple models, for instance Samsung could produce hundreds of models under their brand. This problem of data linking with cardinality of n:1 was resolved by applying transformation rules(Figure 8). The below screenshot shows the process of data linking.

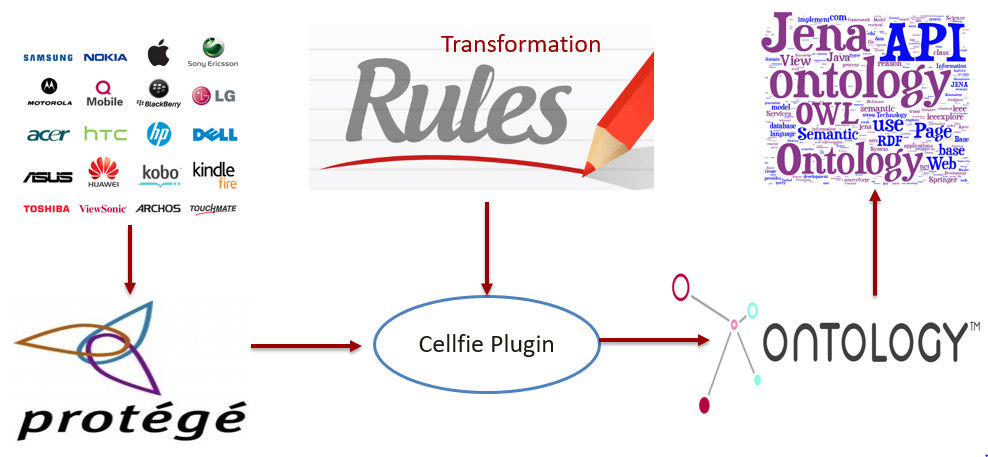
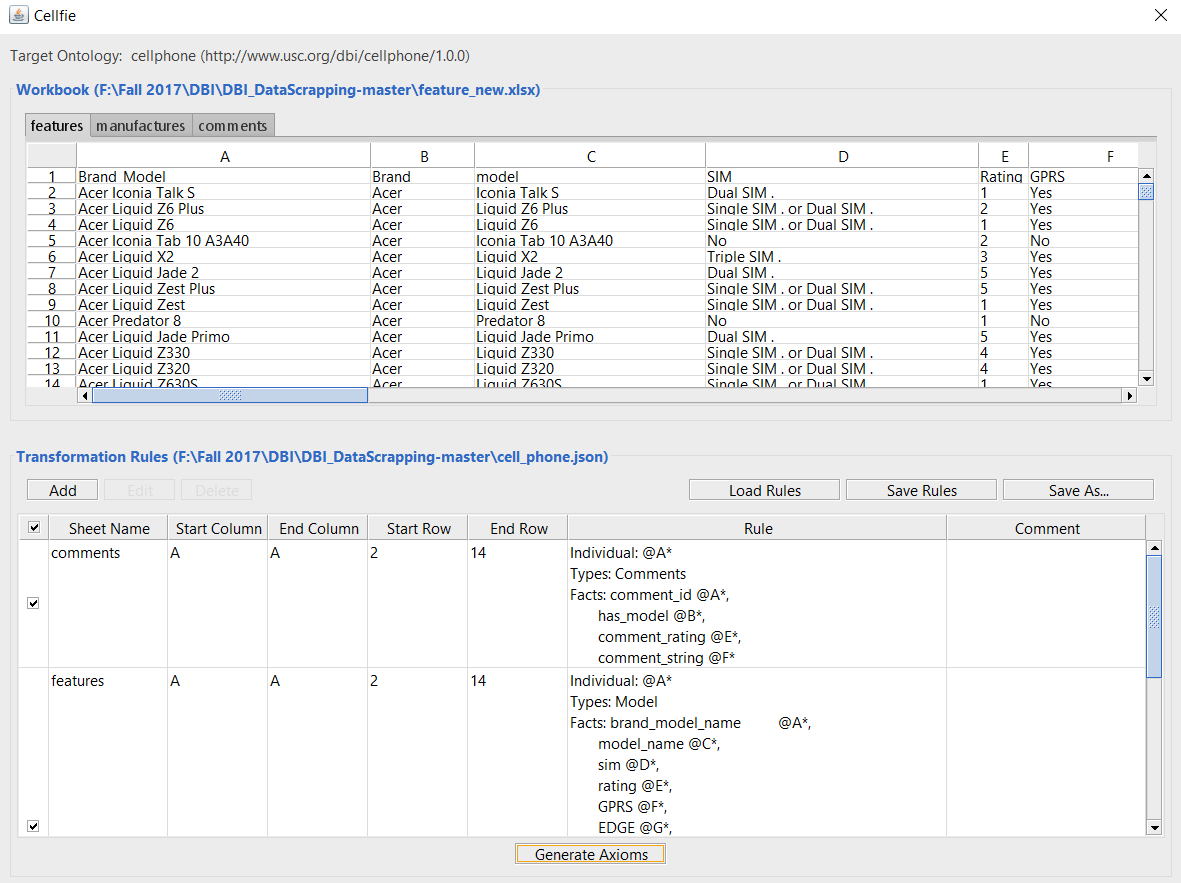


Figure 7. Data Linking Steps

In the figure 6, we can see that data was collected from heterogenous data sources and a consolidated dataset was generated to be fed to Protégé accompanied by Cellfie plugin to which we issued transformation rules to create a resulting OWL file. This OWL file was then given as an input to Apache Fuseki Jena that was our application’s SPARQL endpoint.

 Figure 8. Adding data from CSV to RDF/OWL file by applying transformation rules

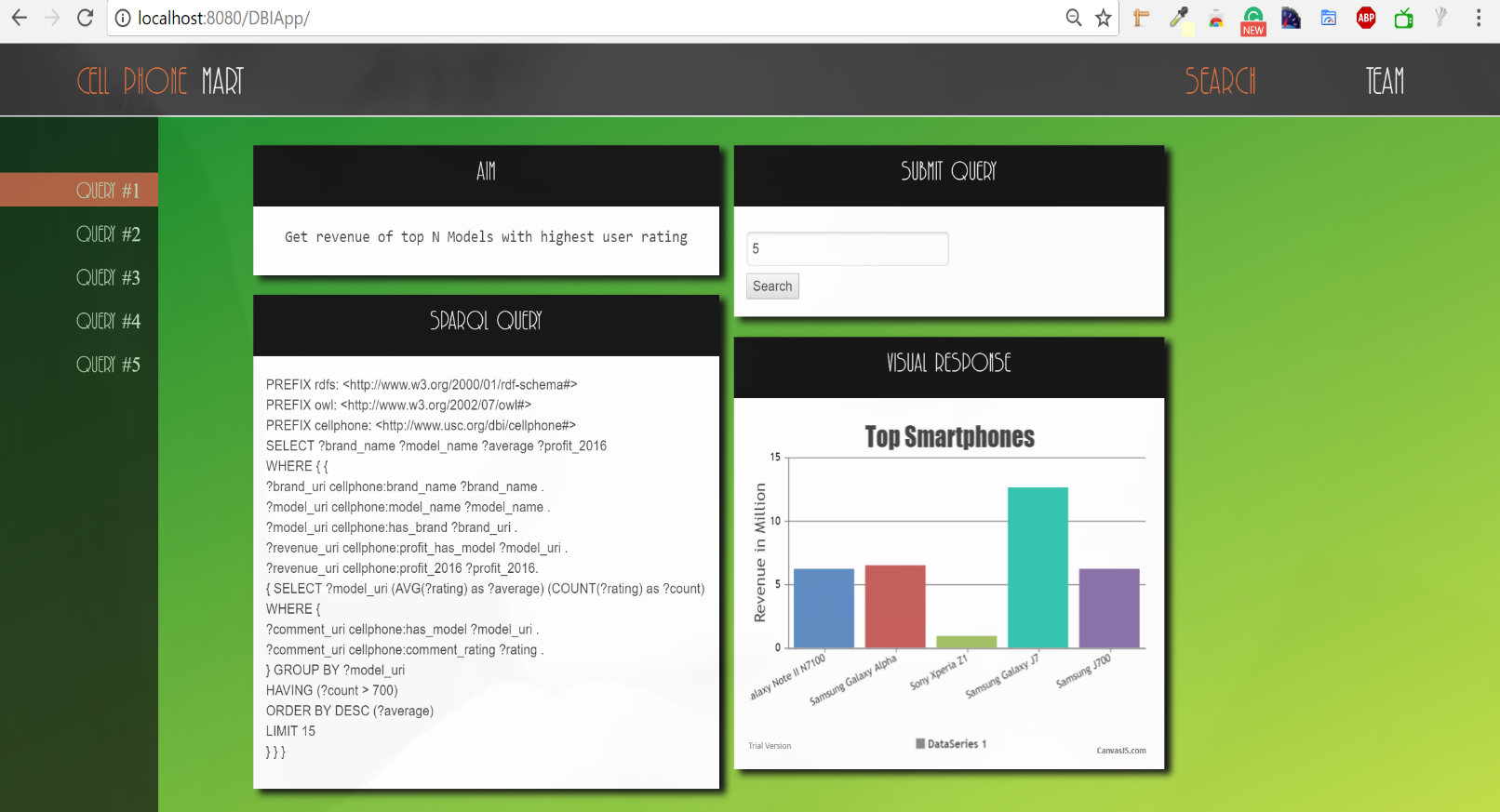
4.4 **Querying**

After data integration and linking, the RDF dataset had roughly 14,00,000 triples. We hosted the RDF dataset in Apache Jena Fuseki server. This server acted as a SPARQL end point. The repository was then queried using Apache Jena API. A user-interface was developed using HTML, jQuery and JavaScript to display the results of the query. We also used High charts, a JavaScript library to implement bar charts and histograms for the results of the queries. The user interface is shown in Figure 9. The queries are pre-defined and provisions for users to change the parameters of the query is provided in the UI, the functionality of the user issuing their own queries from the UI is in scope for future work. When user chooses to run a pre-configured query, the actual SPARQL query is also shown in the text area, which can be used as a template for custom queries. The list of pre-configured queries and their descriptions is given in Table 3.

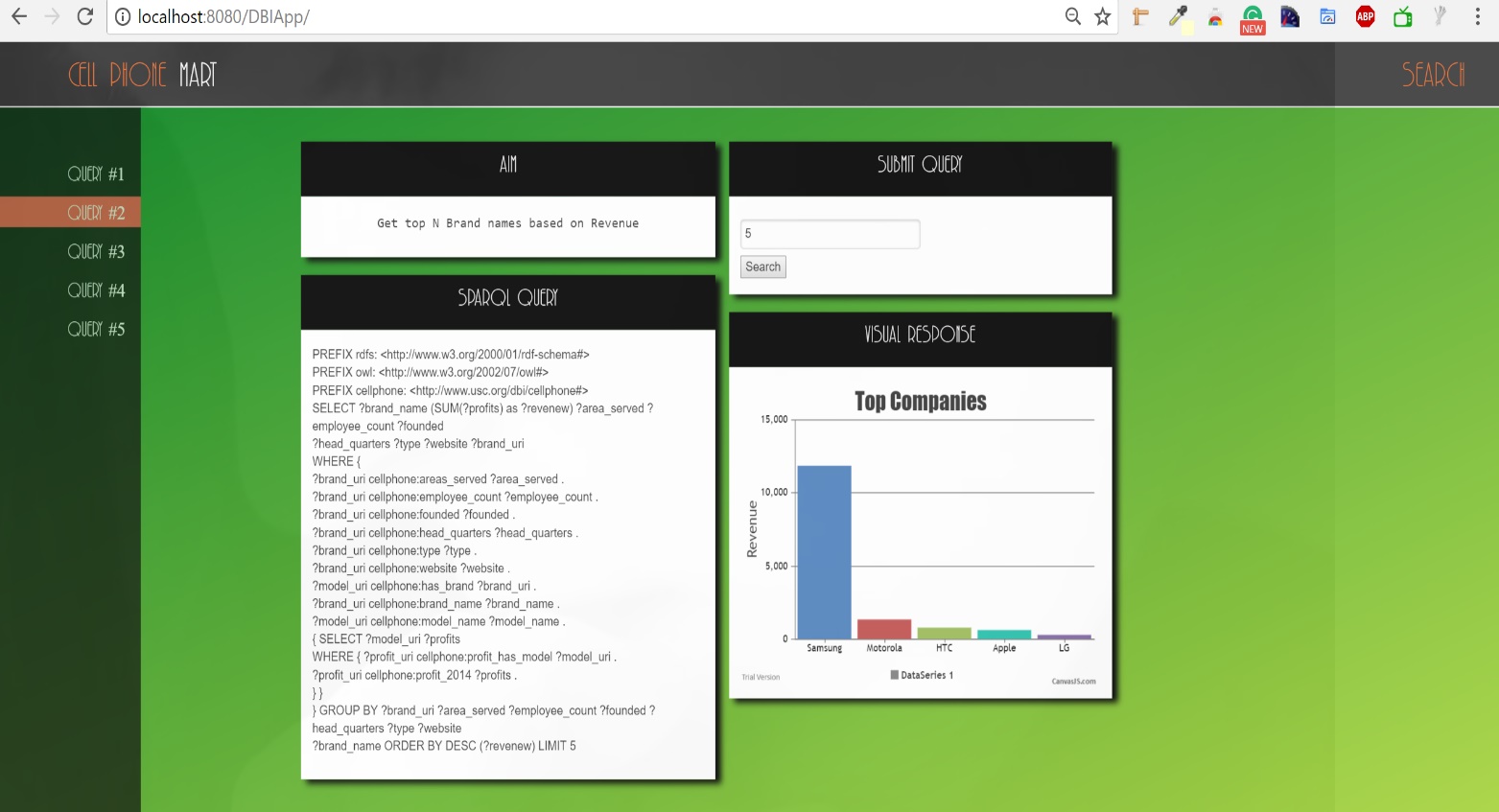
|  |  |
| --- | --- |
| **Query** | **Query Description** |
| Query #1 | **Get the revenue of top N models with highest user rating**: In this query, we are finding the cell phone models which has the best reviews according to Amazon customers, then we get the profits obtained by the companies by selling each of these phone models. This helps in understanding the general trend of the revenue generated by the best rated cell phones. We have also given the bar chart representation for visualizing the trend. |
| Query #2 | **Get top N models based on their revenue:**  In this query we are retrieving top N models, which are having highest gross profit among all the available brands. This helps user in make better choice of a model while buying a cell phone. |
| Query #3 | **Get all the details of a cell phone based on brand and model:** Helps user to get all the details of a cell phone by entering the brand name and model name. This query returns feature data, comments, ratings and profits. |
| Query #4 | **Get all the phones manufactured by reputed brands whose weight is less than a given weight:**  Here users are asked to enter the maximum weight of a cell phone which they wish to buy. The query returns all the cell phones whose weights is below the user entered weights. |
| Query #5 | **Get all the cell phones with screen size greater than X and has best user rating:**  In this queryusers enters the size of cell phone in inches, then we will retrieve all the top-rated cell phones greater than or equal to the user specified size. |

Table 3. Query Description

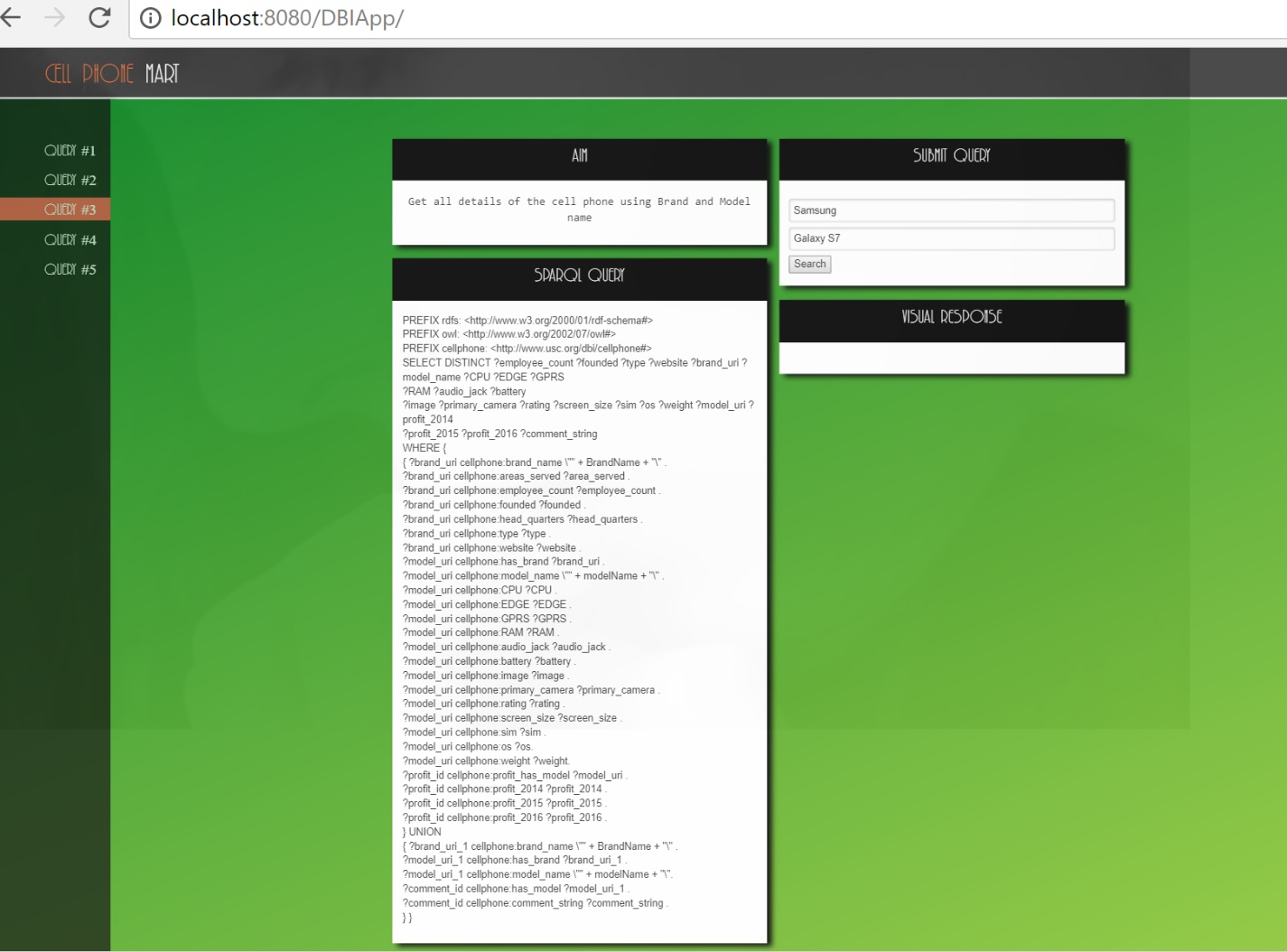
**1.Get the revenue of top N models with highest user rating:**



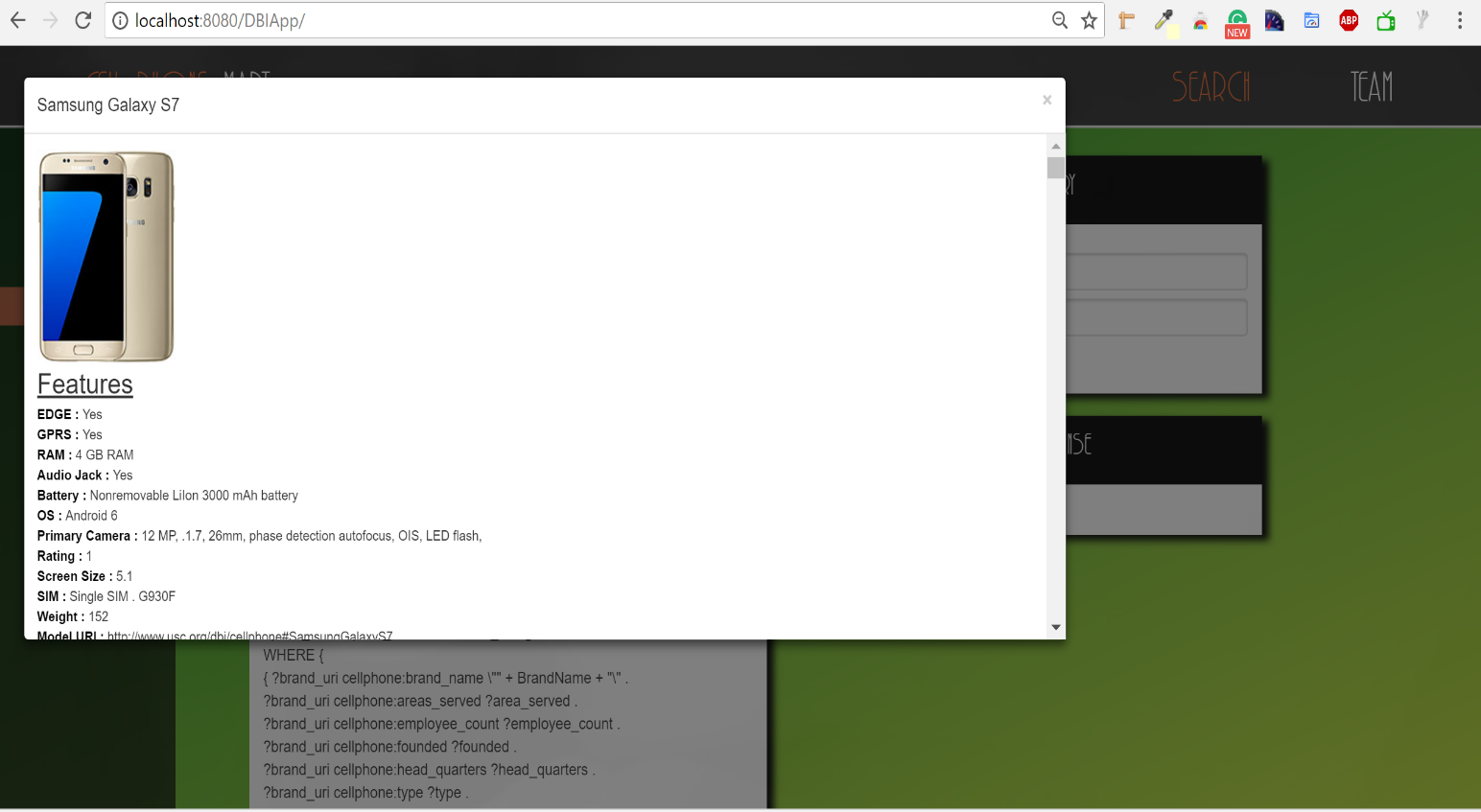
**2. Get top N models based on their revenue:**



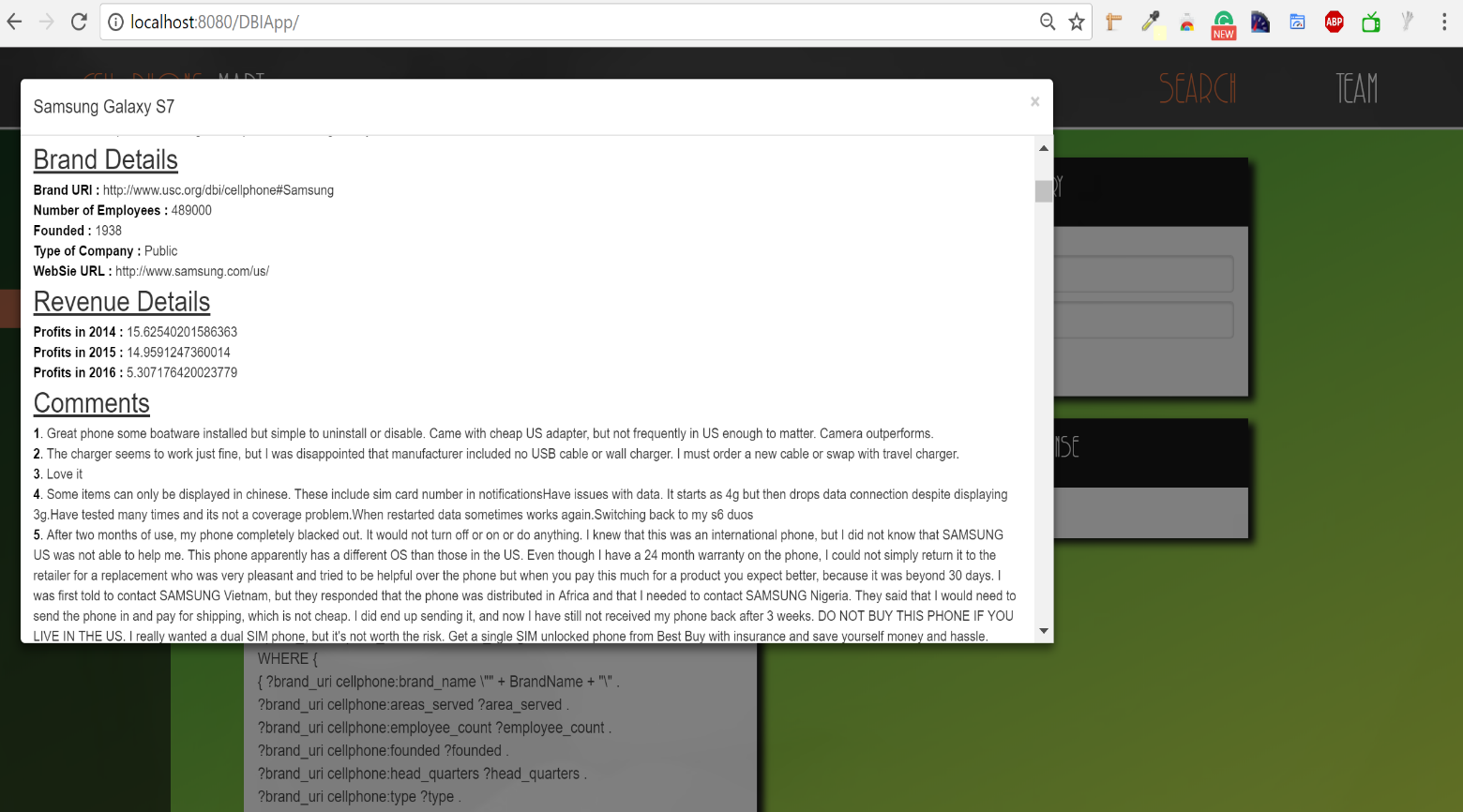
**3. Get all the details of a cell phone based given brand and model:**



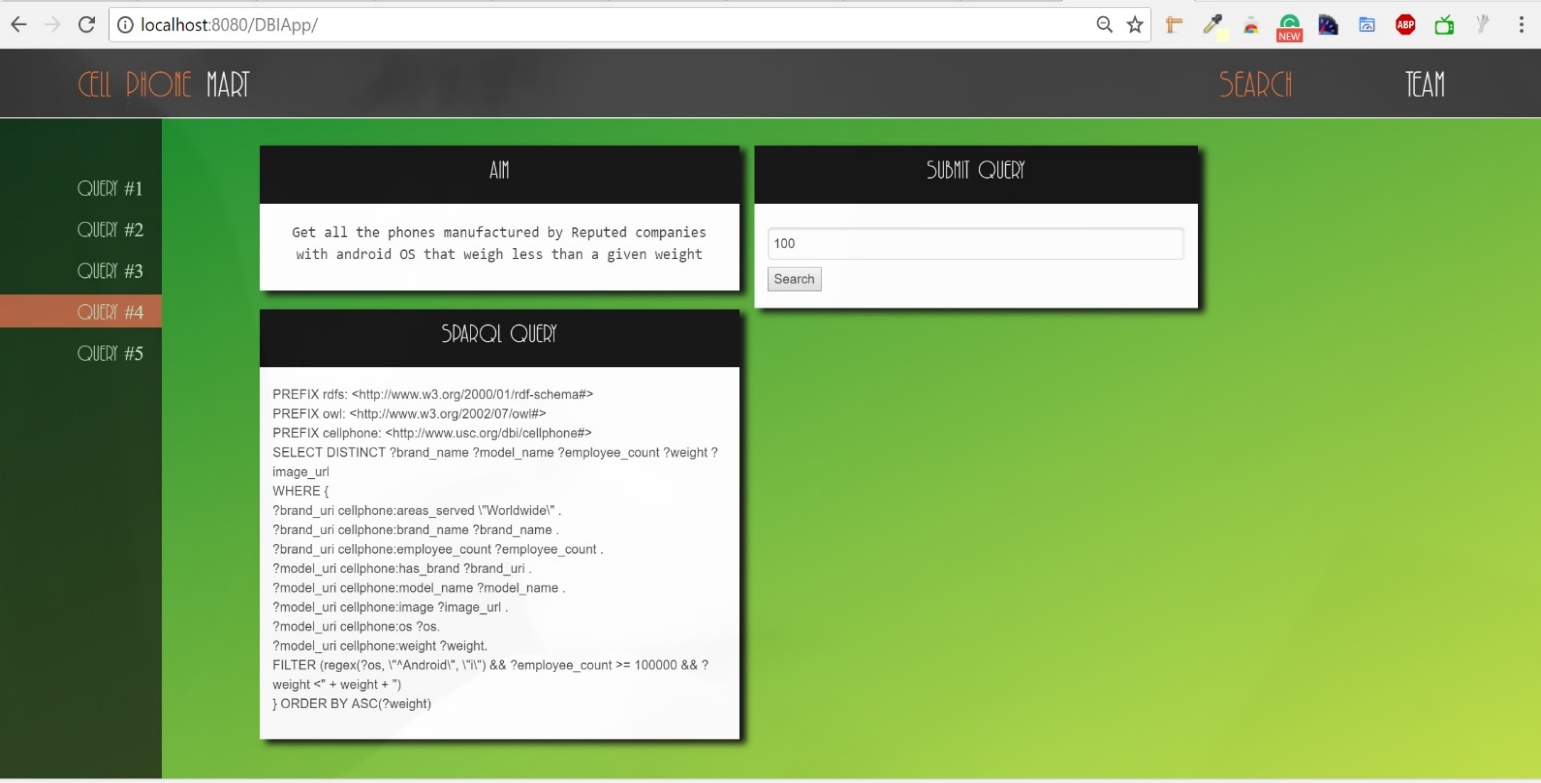
**Device Specs as part of result:**

****

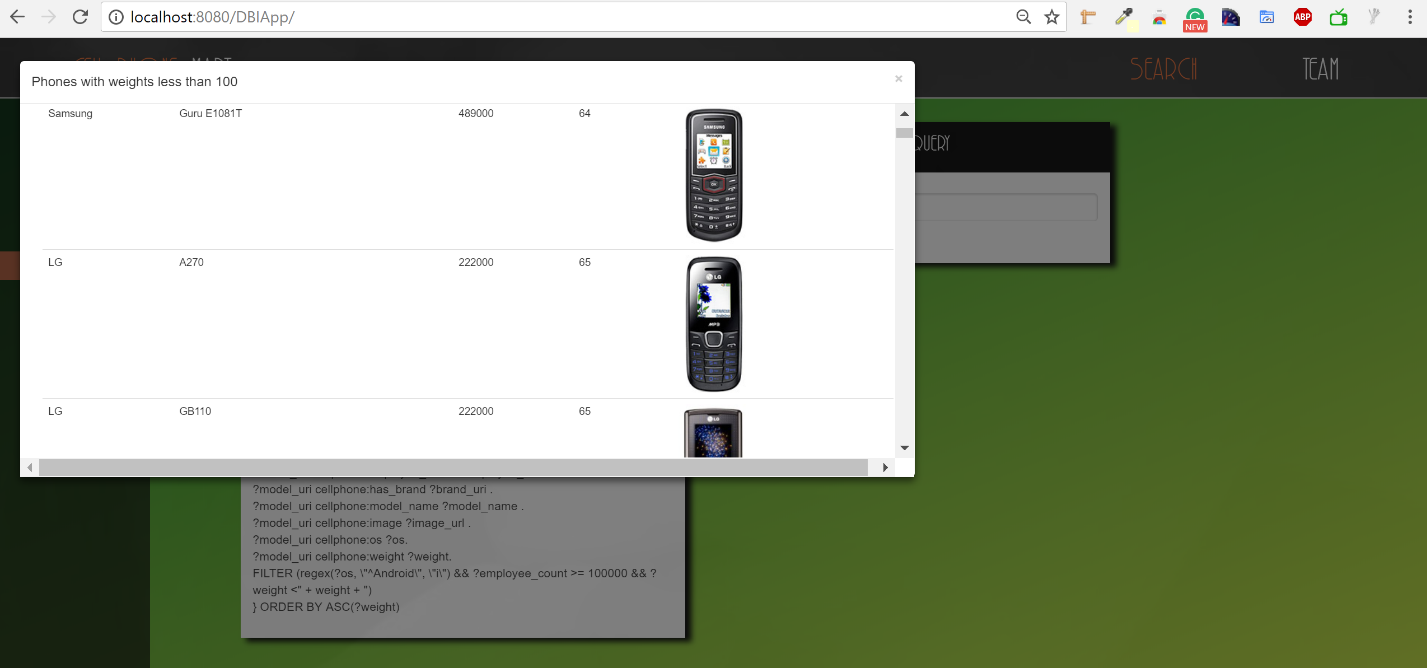
**Brand, revenue, rating as part of results:**

****

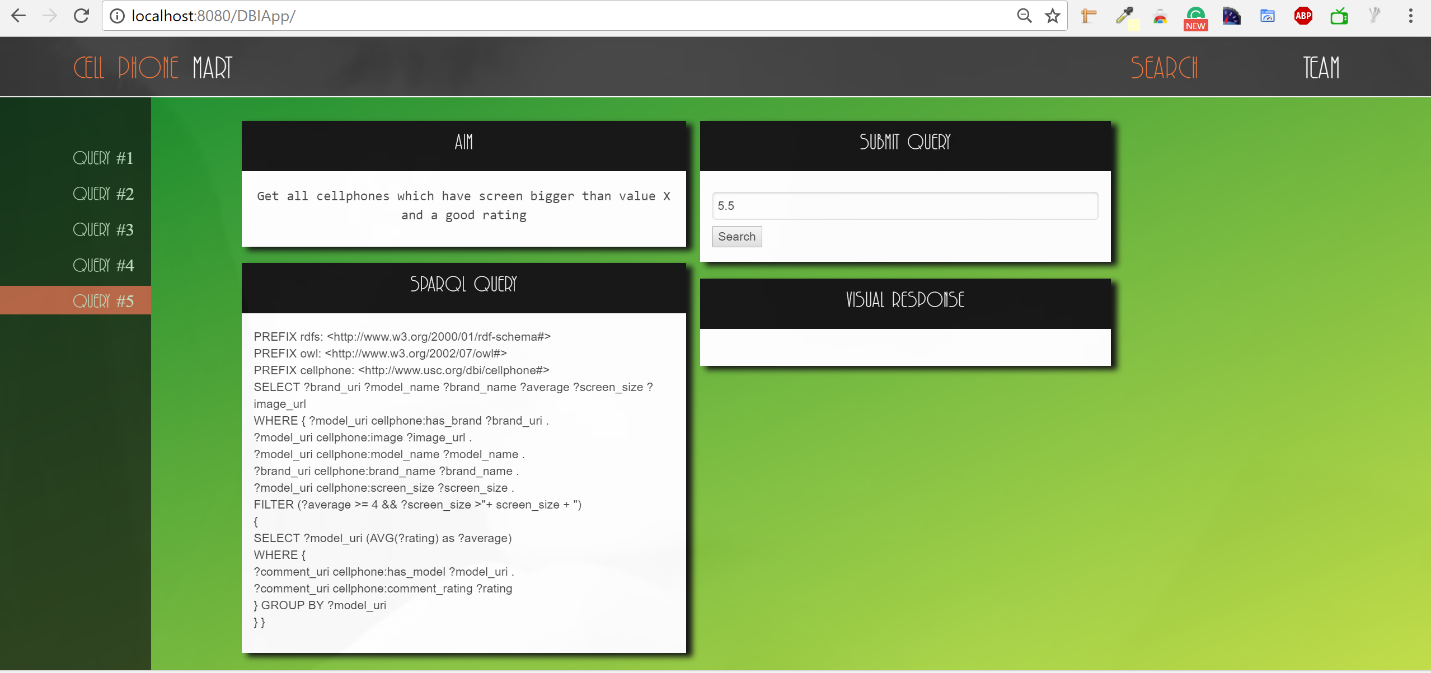
**4. Get all the phones manufactured by reputed brands whose weight is less than a given weight:**



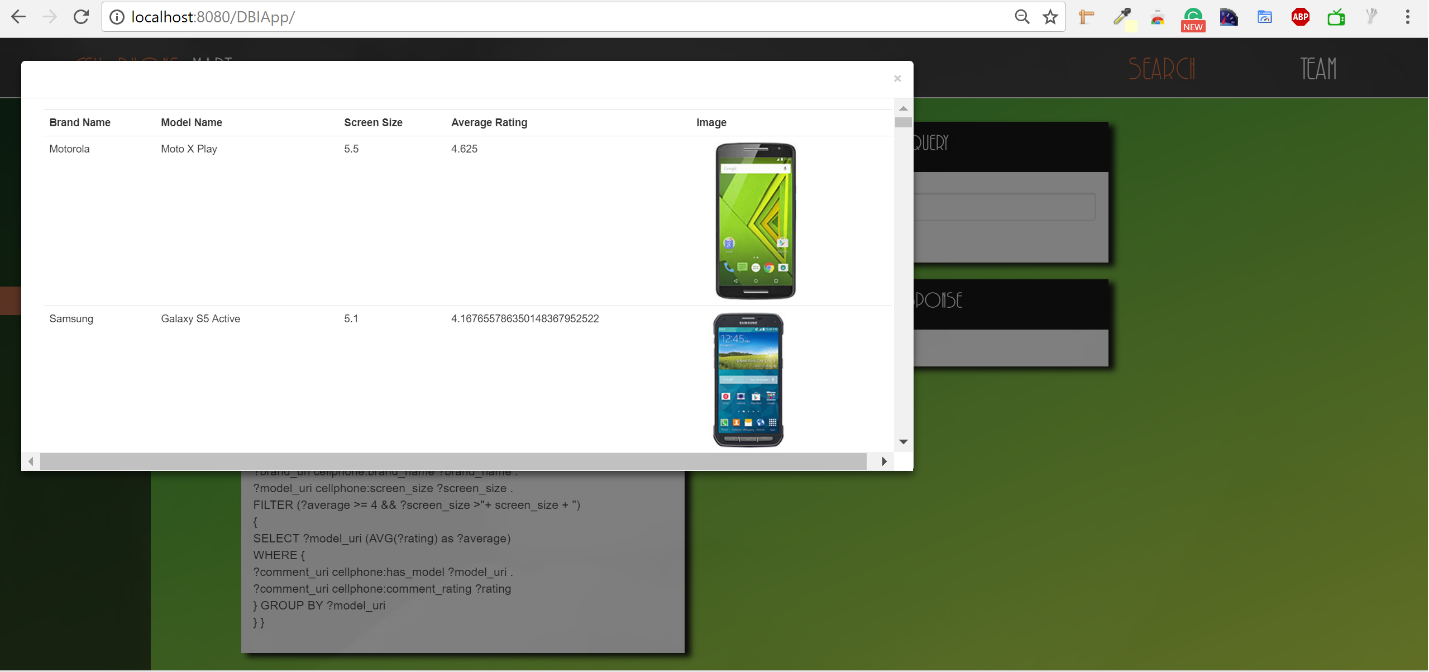
**Result with all models along with weight:**



**5. Get all the cell phones with screen size greater than X and has best user rating:**



**Result has all the phones whose screen size is above X (5.5 inches) and with their average rating:**



1. **Conclusion & Future Work**

Use of Ontology is progressively gaining popularity to unify data in various domains. In Cell Phone Mart, we showed one of its many uses in mobile phone industry by answering queries which would have required exploring multiple web pages to answer.

For future work, we could have an endpoint for more generalized queries for the user to key in. More datasets could be added to extend the current schema and the ontology for a better data management and more complicated queries. For instance, data related to stocks of a company. The current scope of Cell Phone Mart included a set of predefined queries covering a few key performance indices a user would be interested in; however, this could be extended with various analytical tools, and statistical concepts. We used high charts, a JavaScript charting library, in future high stocks could be used to show trends of various company stocks.

1. **References**
   1. Integrating Applications on the Semantic Web, Hendler, James, Berners-Lee, Tim and Miller, Eric "Integrating Applications on the Semantic Web," Journal of the Institute of Electrical Engineers of Japan, Vol 122(10), October 2002
   2. Integrated Movie Database - CSCI 586 Project Report by Muhammad Rizwan Saeed, Santhoshi Priyanka Gooty Agraharam, Ran Ao
   3. <https://www.youtube.com/watch?v=R9ERlUgvgwM>
   4. <https://github.com/protegeproject/cellfie-plugin/wiki/Grocery-Tutorial>
   5. <https://jena.apache.org/documentation/serving_data/>
   6. <https://protegewiki.stanford.edu/wiki/Excel_Import>