Calorie Calculator Application using Google Cloud Platform

Dublin City University
School of Computing
CA675: Cloud Technologies
Assignment 2 Report
Date of submission - 19th April 2019
Group J
Course – MCM

Student Number	Name	Mail ID
18210298	Aishwarya Gupta	aishwarya.gupta3@mail.dcu.ie
18210193	Monishri Rajendran	monisri.rajendran2@mail.dcu.ie
18210902	Shashank Ravishankar	shashank.agrahararavishankar2@mail.dcu.ie
17210335	Vignesh Chandran	vignesh.chandran3@mail.dcu.ie

Introduction

In today's world where food is readily available, we forget to take the right amount of nutrition our body needs. It is highly important to make sure that body gets the right number of calories each day. Obesity and malnutrition can increase the risk of developing several diseases in both children and adults. The need to maintain a healthy lifestyle has become utmost important in the modern world. Tracking the number of calories that we consume each day can tell us a lot about our nutrition and diet. This type of self-monitoring can help manage weight and track our nutrition needs. Our application system allows a person to track the number of calories and nutrition that he/she intakes. The system includes a web interface to search for the food and select multiple items. The application then displays a visualisation based on the foods selected informing the person about the macros that have been consumed and total calories intake.

Cloud Technologies Used

We have used various technologies for our application.

1. **Spark (Databricks):** We have used spark on Data bricks cluster for cleaning and processing our data.
2. **Google Cloud SQL:** Since we wanted to integrate our data with java application and our data being relational in nature, we decided to use a cloud MySQL database for storing our data which could be easily accessed by our application.
3. **Tableau online:** We connected Tableau to our Cloud based data source and have used it to create a dynamic visualisation to show the various macros and calories intake.
4. **Google App Engine:** We have used Google App engine to host our application on Cloud.

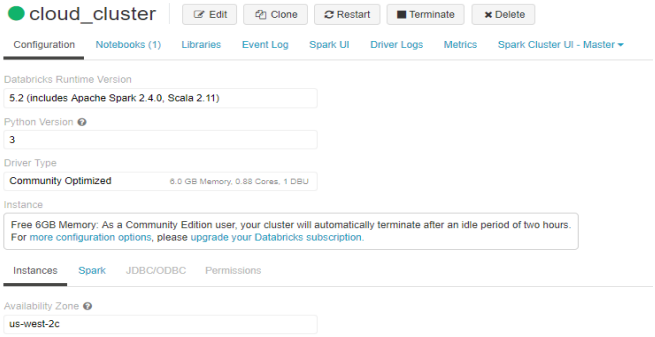
Dataset

We have used a nutritional dataset from Open Food Facts Engine which contains 356,166 rows and 163 columns. This data is publicly available on Kaggle (https://www.kaggle.com/openfoodfacts/world-food-facts). The dataset consists information about various food products from different brands, the energy per 100 grams, macros like fat, protein, fibre, sugar etc., For our question, we needed the nutritional information about the various food items hence we proceeded with the data cleaning.

Data Cleaning

Apache Spark

We had a large sized dataset, hence we decided to use Apache Spark for quick cleaning and processing of data. Apache Spark is an open-source distributed general-purpose cluster-computing framework. To take full advantage of this cluster-based technology, we decided to use it on Databricks Unified Analytics Platform. Databricks is a paid service, however they have a community edition which provides free access to Databricks with limited resources.



Data cleaning environment set up:

1. Create a login for Databricks community edition.
2. Create a cluster to run spark on.
3. Upload the data onto the cluster for access in the notebook.
4. Create a notebook and attach to cluster.

Steps for data cleaning:

After the environment has been set up, we proceed with data cleaning activities

1. Reduce the number of columns to one that are needed for our analysis namely, product_name, brands, energy_100g, energy-from-fat_100g, fat_100g, trans-fat_100g, cholesterol_100g, carbohydrates_100g, sugars_100g, fiber_100g, proteins_100g.
2. Removing duplicates from the dataset.
3. Replacing NA values with 0 in numerical columns.
4. Trimming the values in fields to remove additional spaces
5. Removing data which is erroneous e.g. Energy = 0 but other macros have values
6. Removing products where product name is blank.

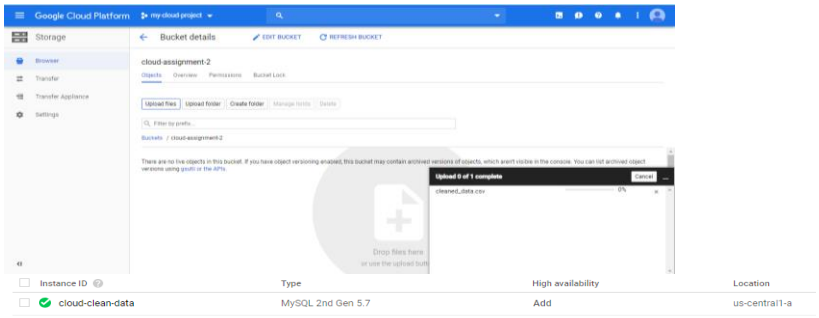
After performing, we finally had the clean data which we proceeded with storing on the Google Cloud SQL database.

Data Storage

Google Cloud SQL

Cloud SQL is a fully-managed database service that makes it easy to set up, maintain, manage, and administer your relational databases on Google Cloud Platform. It allows us to create either MySQL or PostgreSQL database. For our Application, we decided to use MySQL database.

Database setup:



1. Create a Google cloud platform account.
2. Create a bucket on cloud storage and upload the csv file generated after data cleaning.
3. Create a Cloud SQL instance with MySQL
4. Post creation of instance, from the cloud active shell create a database and table where the data will be imported.
5. From the main page of cloud SQL instance, select import csv bucket where file is uploaded file, enter the name of the table which is nutri_data in our case. Now, our database has been set up and is ready to be used.

Web Application Development

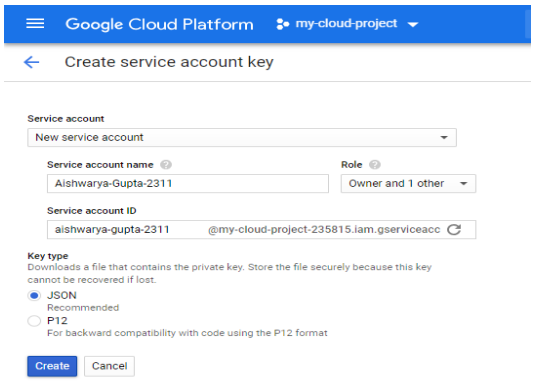
After the data storage has been set up, the next phase of the application development part was focussed in such a way that the web application should be able to communicate with the dataset stored in the cloud storage.

The tools and Technologies used to develop the application is listed as follows:

- Application Development: Eclipse IDE for Enterprise Java Developers
- Eclipse Version 2019-03 (4.11.0)
- Server: Apache Tomcat Server
- Front End: HTML,CSS, JSP, JavaScript
- Programming Language(Middleware): Java 8
- Back End: MySQL JDBC driver / HikariCP
- Database: Cloud MySQL database (RDBMS)

The JDBC (Java Database Connectivity) which is an API of J2EE which helps java program in interact with RDBMS (Relational Database Management System). HikariCP is a very lightweight (at roughly 130Kb) and lightning fast JDBC connection pooling framework developed by Brett Wooldridge around 2012. Web Servers acts as a mediator that helps to accept the request that comes from web server and transverse it to the web application. Servlets are the API of J2EE that helps to develop dynamic web resources. The package representation of servlet API is javax.servlet.*

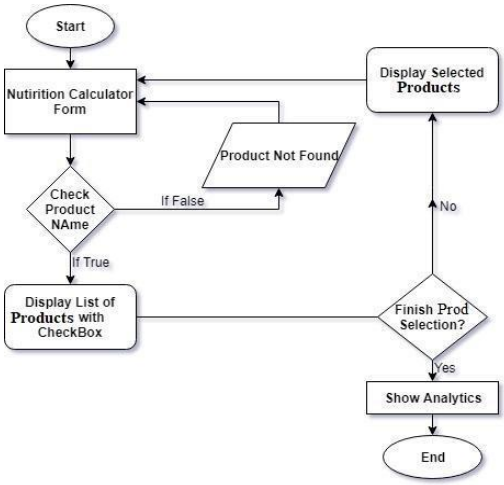
Database connectivity with Web Application (Local App to Cloud Database)



1. Include cloud connector jars in the pom.xml of the project
2. Create a service account key in the IAM page of Google cloud
3. provide access to connect to the Cloud SQL database.
4. Download the json file having authentication information and set up a new environment variable as GOOGLE_SECURITY_CREDENTIALS which points to the file.
5. Enable API to connect to the Cloud SQL from external applications.
6. Use JDBC driver with the cloud connector jars to access the Cloud SQL database.

Design of the Application

The first page of the application will request the user to enter the name of the product which will be searched from the database. The fed name in the first page of the application is searched from the database and the list of searched products is displayed in the second page along with a check box. User can check box the food which they wish to have.



Flowchart of the Application working design

Then, the number of foods will be show in the next page along with the message asking like if the user wish to select more products or else to finish the selection and to show the analytics for the selected products.

If the user finishes the food selection, then nutrition facts analysis for all the selected products will be shown by the graphical visualization.

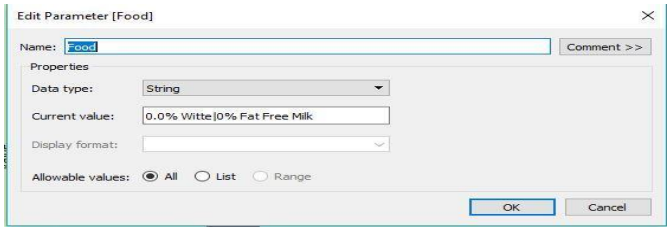
On the other hand, If the user wants to select more products, then the application will be redirected to the first page along with the product name which is already selected. Now, the user can enter the product name again which will be search from the database.

Now, all the current and previous selected products will be now stored with respective ID number and product name which is sent to next page for analytics (i.e. Visualization).

Data Visualisation:

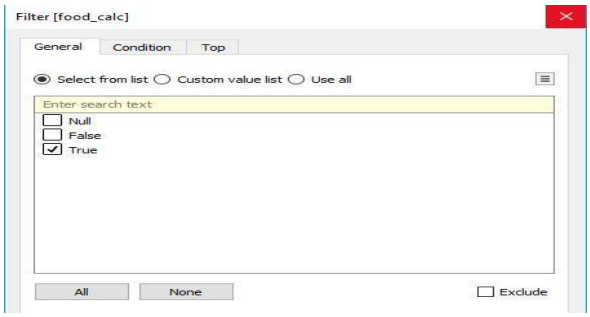
The food items selected by the user which is sent to Tableau by using its respective product name and ID number is visualized dynamically by using Parameters and calculated fields. Tableau provides an opportunity to visualize dynamic content by using Parameters.

Visualization setup and steps involved,



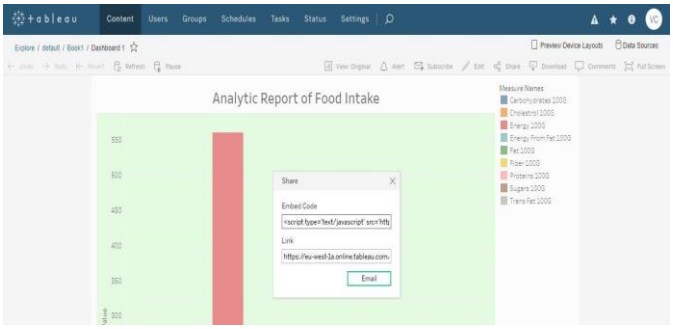
- Tableau workbook is connected to the database on cloud sql.
- The required Dimensions and measures from the database are used to create the visualization.
- For the visualization, the measures used in the viz are marked with different colours.

- To make the visualization dynamic, a parameter field named “food” is created which takes the input(Product Name) from the front end and sends it to the Product field in the visualization.



- To input multiple selected products to the visualization through parameters, a calculated field is created which enables the parameter to take multiple items through a regex function.
- This Calculated field is then enabled as a filter and set true.

Tableau Online



The created workbook is published to the server using Tableau online. The embed URL from the Tableau Online is used to publish the visualization on the webapp.

Visualisation Integration with the Application:

The visualization is integrated with the WebApp by using JavaScript API. A javascript embed code with functions to dynamically alter the parameters by taking the user selections as inputs is created and embedded within the front end code. The function is connected to the parameter name and the filter name in the Tableau workbook.

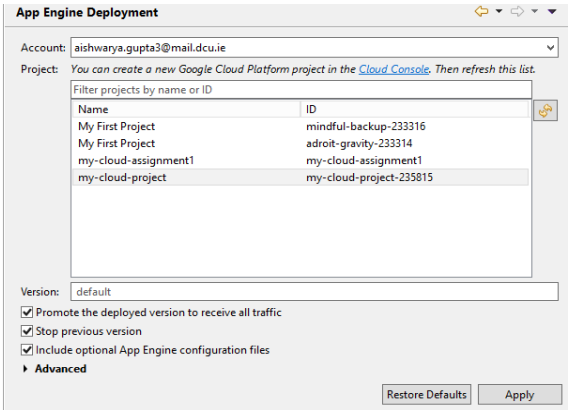
Web APP Hosting

We decided to host our application on Google App Engine as our data store was also present on Google cloud platform.

Google App Engine:

Google App Engine is a web framework and cloud computing platform for developing and hosting web applications in Google-managed data centres.

Hosting Application on Cloud



1. Create an instance of App Engine on Google cloud for a Google cloud project.
2. Download the google cloud sdk on eclipse.
3. Convert the project into a Google cloud project.
4. Connect the project with the google cloud App Engine instance.
5. Right click on the java project select ‘Deploy to App Engine Standard’
6. The application will get deployed and a link would be provided to access the application.

Database connection to application (From App Engine to Google Cloud SQL)

1. Include dependency in pom.xml file of the project to have google client api jars.
2. In IAM page of Google Cloud Platform, provide access to the service account which is connected to the App Engine to connect to the Cloud SQL database.
3. Change the connection code in the application to use HikariCP.
4. In the connections tab of Cloud SQL instance, enable connection from Applications deployed in App Engine under same project.

Challenges Faced and lessons learnt

The main challenge that we faced was connecting the various parts of our application together and the integration. Our application had various components – Database – Application – Tableau which had to be connected such that application with database, Tableau with database, application with tableau. Understanding how to make these connections and enable access helped us learn a lot about the security standards followed by cloud platforms. While hosting the application onto Google App Engine we had to again make changes in the way our application is connecting with the database as this was happening on another server and not on local machine. We also learnt about application development on cloud and using various cloud services. Learning about Spark and using Databricks for data cleaning was another gem that we would take from this project.

Response to Feedback

As suggested by the review team, we are trying to answer multiple questions. We are showing not only the total calories consumed by the user but also the amount of fat, proteins, carbohydrates, fibre, sugar, etc consumed by the user. We have not included other suggestions e.g. Allergen advices as this would have made our application more complicated although we can include it if we perform any future work on our application.

Responsibility Statement

Team Member	Role	Tasks Performed
Aishwarya Gupta	Resource Investigator, Implementer, Coordinator	Idea, Data cleaning, Cloud database Set up and connecting it with various components (Tableau, Application, App engine), Hosting application on Google App Engine.
Monisri Rajendran	Team worker, implementer	Data Cleaning, Data visualisation, Tableau Server (online) connection, Dynamic embedding using Javascript API
Vignesh Chandran	Team worker, implementer	Web Application Development (Coding for Application)
Shashank Ravishankar	Team worker, implementer	Data visualisation

References:

1. https://cloud.google.com/
2. https://cloud.google.com/sql/docs/
3. https://www.baeldung.com/hikaricp
4. https://www.javatpoint.com/java-jdbc
5. https://cloud.google.com/appengine/