Case 2: Creating Data as a Service

Instructions:

- You can use either Python or R to work on this case.
- No sharing of work. You can work in your teams only
- You are expected to submit a report that summarizes the key steps in your implementation as a flow chart and submit fully functional code.
- Deadline: 07/07/2017 11.59 PM. Late submissions lose 10% points per day.

Working with large datasets:

We are going to expand the housing problem we worked in the class and work with real Zillow data.

Review https://www.continuum.io/blog/developer-blog/productionizing-and-deploying-data-science-projects **for motivation on why we are working on this assignment**

Review the following to understand the problem:

- https://www.kaggle.com/c/zillow-prize-1
- https://www.kaggle.com/philippsp/exploratory-analysis-zillow
- https://www.kaggle.com/sudalairajkumar/simple-exploration-notebook-zillow-prize
- https://www.kaggle.com/captcalculator/a-very-extensive-zillow-exploratory-analysis

1. Data ingestion, EDA, Wrangling:

- Download the data from Zillow. (https://www.kaggle.com/c/zillow-prize-1)
- Create an IPYB notebook and Conduct an in-depth EDA (See below for ideas; Note: Your code should be original. You are welcome to use ideas but with attribution).
- Put together a note on what data cleansing is required for automation
- Clean up the data and take care of missing data values using a Python/R script
- Programmatically write the data to a S3 bucket named “ZillowData”. (Use a configuration file to put your Amazon keys)

```
{  "AWSAccess":"access key",

 "AWSSecret":"secret key"}
```

- Dockeize this image: Your folder structure should be something like this (assuming all data you are uploading is in the data folder):
- Assignment2
    - config.json
    - run.sh
    - dataIngestion.py
    - rawDataEDA.ipyb
    - data
    - ddmmyyMMSS.log
- Submit your image to dockerHub and upload all files to github.

2. Create a DBaas (Database as a service)

- Using the DBaas assigned to you, move the clean data to a cloud-based database.
- Note: You should use command line to do this. Create a script that will take the clean data you created earlier and create a DBaas. You will have to review the apis for your assigned DBaas to do this.
- Write a Jupyter notebook illustrating how you can run sample queries connecting to the cloud DBaas and getting back the data.

Data storage:

1. For this exercise, you will be saving the clean data in the system assigned to you
    a. Team 1: Amazon RDS
    b. Team 2 : Amazon S3 – Use Athena
    c. Team 3 : Amazon SimpleDB
    d. Team 4 : Google BigQuery
    e. Team 5 : IBM Cloudant
    f. Team 6 : Microsoft SQL Server db on the cloud
    g. Team 7 : Mongodb Atlas
    h. Team 8 : Mongodb on an EC2 instance
    i. Team 9 : Postgres SQL on an EC2 instance
    j. Team 10 : SQLite on an EC2 instance

3. Create a Rest API to serve the data:

- Your next task is to create a Web services REST API to serve data
- Preferably use FLASK to do this.
- Your web service must be hosted on the cloud (Choose any)
- Here are a few example articles you should review to get a good idea on how you can go about doing this:
    - https://medium.com/@joeclark.phd/creating-a-data-product-with-flask-mongodb-and-ibm-bluemix-7deada61c573
    - http://blog.luisrei.com/articles/flaskrest.html
    - https://blog.threatstack.com/writing-a-web-service-using-python-flask
    - https://blog.miguelgrinberg.com/post/designing-a-restful-api-with-python-and-flask
    - http://blog.thedataincubator.com/2015/09/painlessly-deploying-data-apps-with-bokeh-flask-and-heroku/
    - https://www.codementor.io/jadianes/building-a-web-service-with-apache-spark-flask-example-app-part2-du1083854
    - https://content.pivotal.io/blog/data-science-how-to-text-analytics-as-a-service
- Create a Jupyter notebook to illustrate how to use your REST API

4. Enhancing your REST API: Geospatial search

Note that each record has a Latitude and Longitude. Your goal is to create a REST API that given a Lat and Long, should return the 10 closest homes.

Write a Jupyter notebook and illustrate using this REST API. You should present results something like this

```
+----------------+--------+-------+--------+
| hotel_name     | lat    | lon   | dist   |
+----------------+--------+-------+--------+
| Hotel Astori.. | 122.41 | 37.79 | 0.0054 |
| Juliana Hote.. | 122.41 | 37.79 | 0.0069 |
| Orchard Gard.. | 122.41 | 37.79 | 0.0345 |
| Orchard Gard.. | 122.41 | 37.79 | 0.0345 |
...
+----------------+--------+-------+--------+
10 rows in set (4.10 sec)
```

..

Review these articles for the algorithm and how to use SQL to get these results:

- http://www.arubin.org/files/geo_search.pdf
- https://www.percona.com/blog/2014/06/19/using-udfs-for-geo-distance-search-in-mysql/
- https://www.percona.com/blog/2013/10/21/using-the-new-mysql-spatial-functions-5-6-for-geo-enabled-applications/