Cloud Computing
Assignment 1(Building an EV Database)

By-Ankit Gour

MSc-Big Data and analysis (September 2019-2020)

Student ID-2982713

Table of Contents

1. About the Assignment

This application is based on an Ev database management. The main purpose of this application is to give assess to each and every user to add a car into the database and can search a particular car, compare two or more car and can give review to a car. This application is build using python, html, CSS and java script. This application is also using google app engine to save data on a no SQL database.

This application was built in seven steps from data adding to giving reviews by the users.in this application all the classes and html files were created in atom and a git repository is been created for the same.

Application is consisting of total 13 file, which consist of 2 YAML file, 4 python file and 7 html file

2. Files in this Application

A. YAML Files (2 files):

- app.yaml :

This YAML file informs about runtimes and libraries needed for this application to run to the google app engine. It is also responsible for all the request to be routed among the application.

In this YAML file we declare which language we are using and what is its version like in this we ae using python and its version is 2.7. we also declare it is a thread safe or not. We also declare which is our main file.

- Index.yaml:

In this file we usually make some index on the property of entity. But in this application we are not creating any index. So, in this case all the index will be auto generated.

B. Python Files(4 Files):

- myuser.py:

This is the file which is used to connect with ndb datastore on the google cloud engine to store the data of user on it. This file consist of one class Myuser and only one attribute which is email id which is having String property. In this we first of all import ndb module from google app engine. By using this file and its class we will save our data on the app engine.

- data.py:

This is another file used to connect with ndb datastore on google cloud engine to store data about all the ev we are going to create. In this file we have a class named data and in this file first we will import ndb module from google app engine.

In this the class data consist of 7 attribute:

1. Name : This attribute is having String property. This will save the name of the car in the database.
2. Manufacturer: This is having string Property. This will save the manufacturer name of the car in the database.
3. Year: This attribute is having integer property. This will save the year of manufacture of the car in the database.
4. Battery size: This attribute is having integer property. This will save the battery size of the car in the database.

5. Wltp range: This attribute is having integer property. This will save the wltp range of the car in the database.
6. Cost: This attribute is having integer property. This will save the price of the car in. the database.
7. Power: This attribute is having integer property. This will save the power of the car in the database.

- Reve.py:

This is another file used to connect with ndb datastore on google cloud engine to store data of customer review on each car. This file consist of a class named reve and we have import ndb module from google app engine.

The class reve consist of 4 attribute:

1. Limit: This attribute is having string property. This will store user id who have given the review in the database.
2. Name: This attribute is having string property. This will store name of the car whose review is given in the database.
3. Customer review : This is having text property. This will store customer review for the car in the database.
4. Rating: This attribute is having string property. This will store the rating of a customer for the car ranging between 0-10 in the database.

- A1.py:

This is the main python file which consist of all the classes and responsible for all the functionality of the application.

In this file we first import all the module which are required. So in this python file we have imported 8 module:

1. webapp2: We have imported weapp2.
2. Jinja2: : We have imported jinja2.
3. Users: We have imported module users from google.appengine.api.
4. Ndb: We have imported module ndb from google.appengine.ext.
5. Os: : We have imported os.
6. Myuser: We have imported class myuser from myuser.py.
7. Data: We have imported class data from data.py.
8. Reve: We have imported class reve from reve.py.

There 7 classes in this file:

1. Mainpage:

This is the main loading class of the application. In this class we will pass all the required elements of the application

.1. It will be used to create login and logout URL for the user.
.2. In this all the logged in user will be given access to another path of application with no restriction. But a user is not logged in will be given another path with restriction.
.3. In this bracket we will allow user to view option page.

.4. On the main page a user can see all the review and rating by other user on each cars.

2. Option: This bracket consist of three button add, search and compare.
   - A user can go to any of the three pages from here.
   - This class is having a html front end named option.html
   - A logged in user is only allowed to add a data to ev. So if a user who is not logged in will be give error saying please login.
   - Search and compare are the two function which can be used by user who is not logged in.
3. Data:
   - This is the class responsible for adding data to the datastore.
   - This class is having html front end named dataupload.html.
   - And the data store is used is data.
   - This page is visible when a logged in user click on add button on option page.
   - This function is only accessible to logged in user.
   - In this class we are going to validate and verify the data input by the user.
   - A user cannot keep an empty field and store data to datastore.
   - A user can not re entre the data which is already present in the database.
   - In this the will be validate on the basis of name, manufacture and year using query.
   - First we will fetch all the present data from the database and store it in a list the we will compare all the data in the list with the data entered by the user to check that user has not entered the same data.
   - If a user has entered a ev data which already exist it will print error on the submit.
   - Else it will save the data to the database and print data has been successfully added.
4. Datret:
   - This is the class which handle all the function of search.
   - This class is having html front end named datasearch.html
   - The database used is data.
   - This page is visible to all the users even they are not logged in.
   - This page viewed when a user click on search button on option page.
   - In this class a user can search the ev database.
   - Now, the entire data is checked if all the fields are filled or not.
   - If all the fields are empty, it will show the entire data.
   - If any of the fields are filled, the search is performed.
   - For this I have used if else loop using ndb multikey.
   - If the data is not there in the ndb datastore, then it will throw an error.
   - For this I have created 7 new variables, for all numeric values attributes are compared with their lower and upper attributes.
   - When search is complete, the data will be visible in a list format and name attribute will be generated with the hyperlink
   - And this link would redirect it to the edit delete page.
5. Editdel:
   - This is class which handle all the function of edit and del.

- This class is having a front end named eidel.html.
- This is the page which will open when a user will click on the hyperlink present on name on search page or compare page.
- The main function this page is to edit the current data or delete it.
- There are 3 button on this page.
- The first button is edit. This will edit the data in the ev.
- When a user click on edit button a validation will check that user has edited data or not.
- If yes then it will check that the data is not already present in the database.
- This is working as the user click the button all the edit functionality will be executed.
- The method edit used here is that I created a new key which deletes the old variable first and then assigns a new value by appending it.
- The method delete works the same, just the difference that it deletes the data using the key.
- If a user clicks on the button of review, then its redirected to the review page with the name attribute of the current EV, for giving the review.

6. Comp:
   - This is the class which handle all the function of comparison.
   - This class is having html front end named compare.html.
   - This class uses data database to extract data from it to show.
   - This page is visible when a user click on compare button on option page.
   - This page is assessable to all users.
   - In this class we first of all display whole list of car to the user.
   - User can select which car they want to compare(minimum 2 at a time) using check boxes.
   - When user click on submit button a table below is formed showing all the details of the car to the user.
   - This table will show the car with their detail which user has selected.
   - In the table there will be hyperlink on the car name which will redirect it to the editdel page if user is logged in.

7. Rev:
   - This is the class which will allow user to give review for the particular car.
   - This class is having front end named rewiew.html
   - This class uses reve database to store data.
   - This class uses data and myuser database to retrieve information.
   - This page will open when a logged in user click on review button on editdel page.
   - When a user click on review button on editdel page a href link takes name of the current ev data present and open the review page.
   - In this class first we will get the user id.
   - Using user id and name of car we will generate a new string which we will save in limit attribute.
   - Using name of car we will save it in name
   - We will take input from the user as review and rating.

- On submit we will perform a check that user has not giver review for the same car before.
- If a user has given a review it will print an error.
- Else it will save the review and it will print review submitted.

C. HTML FILES(7FILES)

1. A1.html –

- It is the main html file.
- Here the users login and logout is made and the review is displayed.

2. option.html –

- This has 3 buttons, add, search and review.
- This is connected to the class option in A1.py

3. dataupload.html

- This file consists of each and every entity from name to power.
- It has a submit button.
- This file is front end to class data in A1.py.
- This will upload the data.

4. datasearch.html-

- This will perform the search.
- For name and manufacture it's just one entry.
- For all the other fields it has a lower and an upper entry.
- Then the search is performed.
- The search result is in a form of a list which is displayed below.

5. eidel.html

- This page is opened when we click on the name attribute from the table, which is a hyperlink.
- In this it has 3 buttons,. Edit, delete and review.
- Editdel class is connected to A1.py.

6. compare.html

- Here the data is showed in the beginning.
- The search button is searched with checkboxes.
- When we click on checkboxes, the only values for class data is shown.

7. rewiew.html

- Here we submit the review.
- The backend class is rev in A1.py.
- The reviews are submitted with the text box provided on the page.

- And a range bar for customer rating ranging from 0-10.
- On clicking the submit, the review is submitted.