Cloud Computing
Assignment 2(Building an Application for Task Management amongst multiple Users)

By-Ankit Gour
MSc-Big Data and analysis (September 2019-2020)
Student ID-2982713

Table of Contents

1. About the Assignment 3

2. Files in this Application 4

A. YAML Files (2 files): 4

- app. yaml: 4
- Index. yaml: 4

B. Python Files(4 Files): 5

- myuser.py: 5
- TaskBoarddata.py: 5
- TaskListdata.py: 5
- main.py: 5

C. HTML FILES(7FILES) : 8

1. main.html –..... 8
2. Mainpage.html –..... 8
3. create.html- 8
4. view.html-..... 8
5. dash.html-..... 8
6. taskedit.html..... 8
7. boardedit.html..... 8

1. About the Assignment

This application is based on Building an Application for Task Management amongst multiple Users. The main purpose of this application is to give assess to each and every user to add a task board into the database. Each and every task board can have multiple task and many participants in it. 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. In this Application there is no use of Key Property.

This application is having 7 brackets in developing but I have done 5 of brackets. With this application there is a git repository attached (Note: .git file might be hidden so Please check once.)

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
Two attribute which is email id and i both are 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.

- Emailed: To store user email.
- I: To store all the keys of dashboard in a list in which user is permitted.

- TaskBoarddata.py:

This is a file which is used to connect with ndb datastore on the google cloud engine to store the data of dashboard. In this we import ndb module from google app engine.
This file consist of class named taskboarddata in which we are having four attributes with StringProperties. In this key is used in the combination of taskboard_owner_name and taskboardname which is further saved in myuser in I attribute.

- taskboard_owner_name: The name of the creator of the board.
- taskboard_name:Name of the dashboard.
- taskboard_participant: List of participant in the dashboard.
- task:List of key of task present on the dashboard.

- TaskListdata.py:

This is a file which is used to connect with ndb datastore on google cloud engine to store data of all task. In this we are having one class named tasklistdata. In this we import ndb module from google app engine. This class consist of 7 attribute.

- taskboard_name: name of the dashboard.
- task_creater: Name of the task creator.
- task_name: Name of the task.
- task_allocated_user: email id of the user to whome task is assigned.
- task_Status: Status of the task completed or not.
- task_due_date: Due date of the task
- task_completed_date_time: Date and Time of task completion.

- main.py:
  - This is the main file of this whole assignment which consist of all the classes and functions.
  - In this class we are creating multiple files and importing them.
  - These files are webapp2, os, jinja2, userndb which are inbuilt functions.
  - Except for these we are using the 3 datastores for loading and fetching data.
  - Here we are creating a jinja2 environment for our application.

- *Mainpage class:*
- Here we are giving the access to the user by providing him the login-logout functionality.
- This the main page from where all the functionalities are being performed and all the access are provided to the user.
- Here we defining the key which is the email id. This initializes the application.
- And also we are checking if the user is logged in or logged out of the system.
- If the user is logged in he would be able to access the application else he would be sent to the logged in page.
- In this we pass various template variables in the jinja environment of our front end.

- *Boardadd class:*
- Here the user is creating a new dashboard and the outputs are saved in task board data which is one of the datastores.
- Along with this we create a key for performing this functionality.
- In this we call the current user of the task board in the application, as he is creating the dashboards.
- While adding a dashboard we generate a key based on current users email id and dashboard name.
- We also save the current user in the task member field of the database of the particular dashboard he is in.
- In this we are not passing any template values as it ain't required.

- *Boarddisplay class:*
- Here we are displaying all the dashboards which are with the current user.
- This is done from the database with the help of a key property.
- This is done by calling the current user's key and fetching the i attribute of the current user from the datastore.
- The list in I consists of all the keys of the dashboards in which user is permitted.
- Using that key we fetch all the dashboards using for loop.

- *Vi class:*
- Here all the tasks which are insides the dashboards via which a new html page is opened.
- In this we are fetching all the tasks by calling task board field tasks.
- in the attribute task all the keys of all tasks is present.
- In this we are using the same function as we have called our dashboards in the upper class.

- *Dashboard class:*
- In this class we are adding the user and adding new functions are defined.
- In the first part, we are creating a task

- While doing so, we are creating a key using task board name and class and we are also ensuring that there should not be same name task in a particular board.
- While saving task board data, we also mark some required things, like a user who's adding a task board should give a name and deadline to a task board as its required.
- While assign we also save that particular task key to that particular board task section.
- We also perform another part In this section we add a user to a board.
- While adding a user we insure that user is created if he's not present on the application and the key of the task board should be saved in the users I attribute.
- And also we save the user id in the dashboard participants list.

- *Stat class:*
- In this class we are changing the status of the task if it's not completed along with this the completion time and the date is marked.
- Here we are firstly checking the if the task completed or not.
- if the task is not completed then its status would be changed to completed and the current time and the date will be stored in the task list data in the attribute task completion date and time.

- *Delet class:*
- Here we are deleting the task and deleting the keys of that particular task from the board.
- In this class while deleting a task we insure that we remove the task key from task board data and we also ensure that data is completely removed using key properties from the task list data.

- *Edittask class:*
- Here the entire task is edited with all its entities and replacing the keys if the name is found to be similar.
- In this class we are editing a task.
- First we are fetching a task detail from the key property and displaying them to the user.
- If a user modifies the task and changes the task name, the key property of task will be changed and stored in the task board data.
- If a user tried to edit a deadline of the task and will be stored in the task list data and the changes will be visible.
- In this we are giving user an option to edit.

- *Modifyboard class:*
- Here we are modifying a particular board.
- Along with removal of all the users and a particular user is being removed performing a common functionality.
- On this we can edit a dashboard

C. HTML FILES(7FILES) :

1. main.html –
   - It is the main html file.
   - Here the users login and logout is made.

2. Mainpage.html –
   - Here we have created a new option, one is adding a dashboard and the other is viewing a task board.
3. create.html-
   - Here we are creating a board and is the front end page of board creation.
   - On this page we create a dashboard.

4. view.html-
   - This is a front end page of the board display and here we are displaying all the task boards.
   - On this page we can view all the dashboard and we can have the option for view dashboard and modify dashboard.

5. dash.html-
   - This is the front end page for dashboards, where all the tasks inside the dashboards are displayed like task name, allocation id and status.
   - On this page we can add a member to the dashboard.
   - We can also create a task on this page .

6. taskedit.html
- This is the front end of task edit and here we edit the task.
- On this we can edit a task on the basis of task name, status and deadline

7. boardedit.html
- This is the dashboard editing front end page.