Blinkfire Analytics Code Challenge

Welcome to Blinkfire Analytics Code Challenge (BACC). In this BACC, you will have 3 days to complete the task described in this document. For any questions regarding the description or requirements of this challenge, please send them via email to nan@blinkfire.com.

We prefer that you complete this coding challenge in Python but if you are more comfortable working in Java that will also be acceptable for this challenge.

Feel free to pick the database and front-end language that you are most comfortable working with.

[Description of the task]

Assume that we are managing the backend of an image sharing social platform. Its name is InstSnap. User posts on InstSnap on the backend contain various information including:

- User id
- Post creation time
- Post text
- Post image url
- Post expiration time

Please design a data structure to properly store this user post meta data, and implement a REST API to enable querying the user posts with various filters.

Creating a webpage/GUI to conduct the queries is a bonus. In the web version, the user should be able to select or input one or multiple search criterion. Based on the search criteria specified by the user the resulting posts should be returned and displayed in the frontend, the posts should include and display each posts image.

[Task details]

Together with this BACC task description, an xlsx file will be provided. In the file, a list of sample post meta data is provided. Each of the columns in the xlsx file represents an attribute of the post. And they are explained below:

- User_id
    - A unique id to identify a user in system
- Post creation time

    - Datetime to indicate the time when the post was created
- Post text
    - A text description that was posted with the image
- Post image url
    - The url to the image embedded in the post
- Post expiration time
    - An integer that indicates the number hours the post is valid after the post's creation time. A post should only be returned if it is active (not expired) based on the applied searchable date range.

The user should be able to input and search posts in the system according to any combination of the below:

- By user id
- By filtering all the valid posts given start and end time
- By keywords in text is a bonus

- **Task 1**
    - **Design proper data structure to store the provided post meta data**
- **Task 2**
    - **Enable post search**
- **Task 3**
    - **Implement a REST API to enable post queries via API**
- **Task 4**
    - **Webpage / GUI to define and run searches**
- **Task 5**
    - **Design / formatted display of the returned results**
    - **The formatted image display on webpage**

Please note, the completion of all the 5 tasks is not mandatory. Task 1-2 are basics, and Task 4-5 are a bonus.

[A simplified example]

A simplified example of the given meta meta data in the table below:

user_id	creat_time	text	image_url	expiration

00001	2019-03-01-8:00:00	What a wonderful day	http://instsnap/test_image_1.jpg	24
00002	2019-03-02-8:00:00	Happy friday	http://instsnap/test_image_2.jpg	48
00001	2019-03-03-8:00:00	Have a nice weekend	http://instsnap/test_image_3.jpg	96

[search_1]

Inquiry:

```
{
   ‘user_id’ : 00001,
   ‘Start_time’: 2019-03-04-8:00:00,
   ‘End_time’: 2019-03-05-8:00:00,
   ‘Search_fields’: [‘user_id’, ‘text’]
}
```

Return:

```
{
   {
         ‘user_id’ : 00001,
         ‘Text’ : ‘Have a nice weekend’
   }
}
```

Note: the post from user_1 on 2019-03-01 is not returned due to the post expired during the search date range.

[Submission]

Please submit the answer, your code, and a brief description of your solution.