Project 4 Task 2 - Find Activity App

My application takes a username string from the user, and to give user recommend activity by clicking“Find Activity”Button. User is able to choose“Do it”or“ Try another activity”by clicking the corresponding button and also able to return to find activity just by clicking“Back to find activity”button.

And my web application will record all operations from clients. Those records will be processed by analysis functions. User will first see three tables of analysis, they are“High frequency Activity Types TOP 10”,“Just DO IT Activities TOP 10”and“DISLIKE Activities

TOP 10”.

The dashboard URL is:

<https://frozen-castle-25997.herokuapp.com/>

Or

<https://frozen-castle-25997.herokuapp.com/index.jsp>

1. Log useful information

In my web application, I logged useful information as the below table shows:

|  |  |  |
| --- | --- | --- |
| **Name** | **Meaning** | **Example** |
| Action  Name | The name of the operation. | Client Request, Request API |
| User | The name of the user from android client input. | John |
| Action Time | The exactly date and time when this action performed. | 2022-03-30T04:50:07.161 |
| Activity ID | The id of an activity. | 6553978 |
| Client | The name of the client. | Mozilla/5.0, okhttp/4.9.3 |
| Client  Request | HTTP request method and target route. | GET /activity/doit |
| Client Reply | The content of reply to client. | REPLY: name: Look at pictures and videos of cute animals, id: 2565076,  type: relaxation, result: true, message: "" |
| 3rd Request | HTTP request method and URL to 3rd party. | GET  <http://www.boredapi.com/api/activity/> |
| 3rd Reply | The content of reply from 3rd party | REPLY: activity "Uninstall unusedparty.  apps from your devices" with type  "busywork" KEY(2850593) price: 0.0,  participants: 1, accessibility: 0.0, link: |

ll ti i t ( i f )

|  |  |  |
| --- | --- | --- |
|  | party. |  |

2. Store the log information in a database

I use InfoStore class to implement all operations with mongodb in InfoStore.java class Module.

I use POJO classes to store and retrieve data from mongodb.

I created to POJO classes: BoringActivity and ActivityLog.

Reference link: <https://www.mongodb.com/docs/drivers/java/sync/v4.3/quick-start/>

Both these two classes have empty constructor methods that will ensure mongodb lib being able to create POJOs.

Then all operations with mongodb will be easier and better understanding.

An example of POJO class:

|  |
| --- |
| public class BoringActivity {  private String activityName;  private String activityType;  private String activityId;  public String getActivityName() { return activityName; }  public void setActivityName(String value) { activityName = value; }  public String getActivityType() { return activityType; }  public void setActivityType(String value) { activityType = value; }  public String getActivityId() { return activityId; }  public void setActivityId(String value) { activityId = value; }  public BoringActivity() { }  } |

An example of inserting to collections:

|  |
| --- |
| // get infos collection  MongoCollection<ActivityLog> collection = db.getCollection("infos", ActivityLog.class);  // insert log record  co ec on. nser One \_ n o ; |

3. Display operations analytics and full

logs on a web-based dashboard

Here is a screenshot of how they being displayed to user:



**3.1** **A** **unique** **URL** **addresses** **a** **web** **interface**

**dashboard** **for** **the** **web** **service.**

The unique URL address is:

<https://frozen-castle-25997.herokuapp.com/index.jsp>

**3.2** **The** **dashboard** **displays** **at** **least** **3** **interesting**

**operations** **analytics.**

“High frequency Activity Types TOP 10”,“Just DO IT Activities TOP 10”and“DISLIKE Activities TOP 10”are my 3 interesting operations analytics.

High frequency Activity Types TOP 10:

Among those random activities I grouped them by the types of activities. And counted the getting times of type of activity. Sorted by getting times by descending order.

Just DO IT Activities TOP 10: The top 10 activities user clicked“DO IT!”button to. DISLIKE Activities TOP 10: The top 10 activities user clicked“TRY ANOTHER”button to.

**3.3** **The** **dashboard** **displays** **formatted** **full** **logs.**

Considering a mess of things will show in dashboard page, I moved full logs to another url which is <https://frozen-castle-25997.herokuapp.com/logs.jsp> and there also a link in dashboard page.

|  |
| --- |
| <a href=”logs.jsp”>View Full Logs</a> |

The formatted full logs table was implemented by tags in jsp looping:

|  |
| --- |
| <% for (ActivityLog log : logs) { %>  <tr>  <td><%=log.getActionName()%></td>  <td><%=log.getUserName()%></td>  <td><%=log.getActionTime()%></td>  <td><%=log.getActivityId()%></td>  <td><%=log.getClientInfo()%></td>  <td><%=log.getClientRequestInfo()%></td>  <td><%=log.getClientReplyInfo()%></td>  <td><%=log.getApiRequestInfo()%></td>  <td><%=log.getApiReplyInfo()%></td>  </tr>  <% } %> |

Here is a screenshot of full logs page:



4. Deploy the web service to Heroku

I used a tomcat docker image to deploy to Heroku.

ROOT.war package and tomcat\_starter.sh script are built into this image.

After my docker image being pushed to Heroku web application“frozen-castle-25997”. My web service is available at: <https://frozen-castle-25997.herokuapp.com/>