Project 4 Task 2 – Web Service Logging and Analysis Dashboard
By Qinzhi Peng

1.Log useful information
My web server can log information from android app by using the following snippet of code:

```
String search = request.getParameter("searchWord");

// connect to MongoDB
String uri = "mongodb://qinzhip:pqz12311@cluster0-shard-00-
00.lhwqj.mongodb.net:27017,cluster0-shard-00-
01.lhwqj.mongodb.net:27017,cluster0-shard-00-
02.lhwqj.mongodb.net:27017/InterestingPictureDB?w=majority&retryWrites=true&t
ls=true&authMechanism=SCRAM-SHA-1";
MongoClient mongoClient = MongoClients.create(uri);
MongoDatabase database = mongoClient.getDatabase("InterestingPictureDB");
MongoCollection<Document> collection = database.getCollection("logs");

if (request.getServletPath().equals("/getAnInterestingPicture")) {
    if (search != null) {
        id++;

        long start = System.currentTimeMillis();
        // use model to do the search and choose the result view
        pictureURL = ipm.doPexelSearch(search);
        long end = System.currentTimeMillis();

        latency = (end - start) / 1000d;

        // set the picture url
        request.setAttribute("pictureURL", pictureURL);

        // insert data
        // Source: https://www.mongodb.com/docs/drivers/java/sync/v4.3/usage-
examples/findOne/
        InsertOneResult log = collection.insertOne(new Document()
                .append("_id", new ObjectId())
                .append("id", id)
                .append("searchTerm", search)
                .append("requestFromPhone", "https://damp-fortress-
92242.herokuapp.com/getAnInterestingPicture?searchWord=" + search)
                .append("requestToAPI", ipm.request.toString())
                .append("responseFromAPI", ipm.IPresponse.toString())
                .append("pictureURL", pictureURL)
                .append("latency", latency));
        System.out.println("Success! Inserted document id: " +
log.getInsertedId());

        nextView ="result.jsp";

    } else {
        // no search parameter so choose the prompt view
        nextView = "prompt.jsp";
    }
}
```

2. Store the log information in a database
The web service can connect, store, and retrieve information from a MongoDB database in the cloud.
Database name is: InterestingPictureDB
Table name is: logs

```
// connect to MongoDB
String uri = "mongodb://qinzhip:pqz12311@cluster0-shard-00-
00.lhwqj.mongodb.net:27017,cluster0-shard-00-
01.lhwqj.mongodb.net:27017,cluster0-shard-00-
02.lhwqj.mongodb.net:27017/InterestingPictureDB?w=majority&retryWrites=true&t
ls=true&authMechanism=SCRAM-SHA-1";
MongoClient mongoClient = MongoClients.create(uri);
MongoDatabase database = mongoClient.getDatabase("InterestingPictureDB");
MongoCollection<Document> collection = database.getCollection("logs");

if (request.getServletPath().equals("/getAnInterestingPicture")) {
    if (search != null) {
        id++;

        long start = System.currentTimeMillis();
        // use model to do the search and choose the result view
        pictureURL = ipm.doPexelSearch(search);
        long end = System.currentTimeMillis();

        latency = (end - start) / 1000d;

        // set the picture url
        request.setAttribute("pictureURL", pictureURL);

        // insert data
        // Source: https://www.mongodb.com/docs/drivers/java/sync/v4.3/usage-
examples/findOne/
        InsertOneResult log = collection.insertOne(new Document()
                .append("_id", new ObjectId())
                .append("id", id)
                .append("searchTerm", search)
                .append("requestFromPhone", "https://damp-fortress-
92242.herokuapp.com/getAnInterestingPicture?searchWord=" + search)
                .append("requestToAPI", ipm.request.toString())
                .append("responseFromAPI", ipm.IPresponse.toString())
                .append("pictureURL", pictureURL)
                .append("latency", latency));
        System.out.println("Success! Inserted document id: " +
log.getInsertedId());

        nextView ="result.jsp";

    } else {
        // no search parameter so choose the prompt view
        nextView = "prompt.jsp";
    }
} else if (request.getServletPath().equals("/getDashboard")) {
    // retrieve data
    List<Document> logDoc = collection.find(gte("id", 1)).into(new
```

```
ArrayList<>());
    System.out.println("log list: ");
    for (Document eachLog : logDoc) {
        System.out.println(eachLog.toJson());
        Gson gson = new Gson();
        IPLog ipLog = gson.fromJson(eachLog.toJson(), IPLog.class);
        //System.out.println(ipLog.id);
        //System.out.println(ipLog.requestFromPhone);

        // add new logged information to logList
        logList.add(ipLog);

        // add data to related list
        pictureURLList.add(ipLog.pictureURL);
        pictureTagList.add(ipLog.searchTerm);
        latencyList.add(ipLog.latency);
    }

    request.setAttribute("pictureURL", pictureURL);
    request.setAttribute("loggedImg", logList);
    request.setAttribute("prevPicture", pictureURLList);
    request.setAttribute("topSearchTerm",
ipm.getTopSearchTerm(pictureTagList));
    //System.out.println(ipm.getTopSearchTerm(pictureTagList).get(0));
    request.setAttribute("avgLatency", ipm.getAvgLatency(latencyList));
    //System.out.println("avgLatency: "+ipm.getAvgLatency(latencyList));

    // choose dashboard view
    nextView = "dashboard.jsp";
}
```

3. Display operations analytics and full logs on a web-based dashboard
a. A unique URL addresses a web interface dashboard for the web service.
https://damp-fortress-92242.herokuapp.com/getDashboard

b. The dashboard displays at least 3 interesting operations analytics.
The dashboard can show previous searched images, top5 picture search terms and their counts, and the average latency.

Previously searched images (5)

Top 5 picture search terms(&counts)

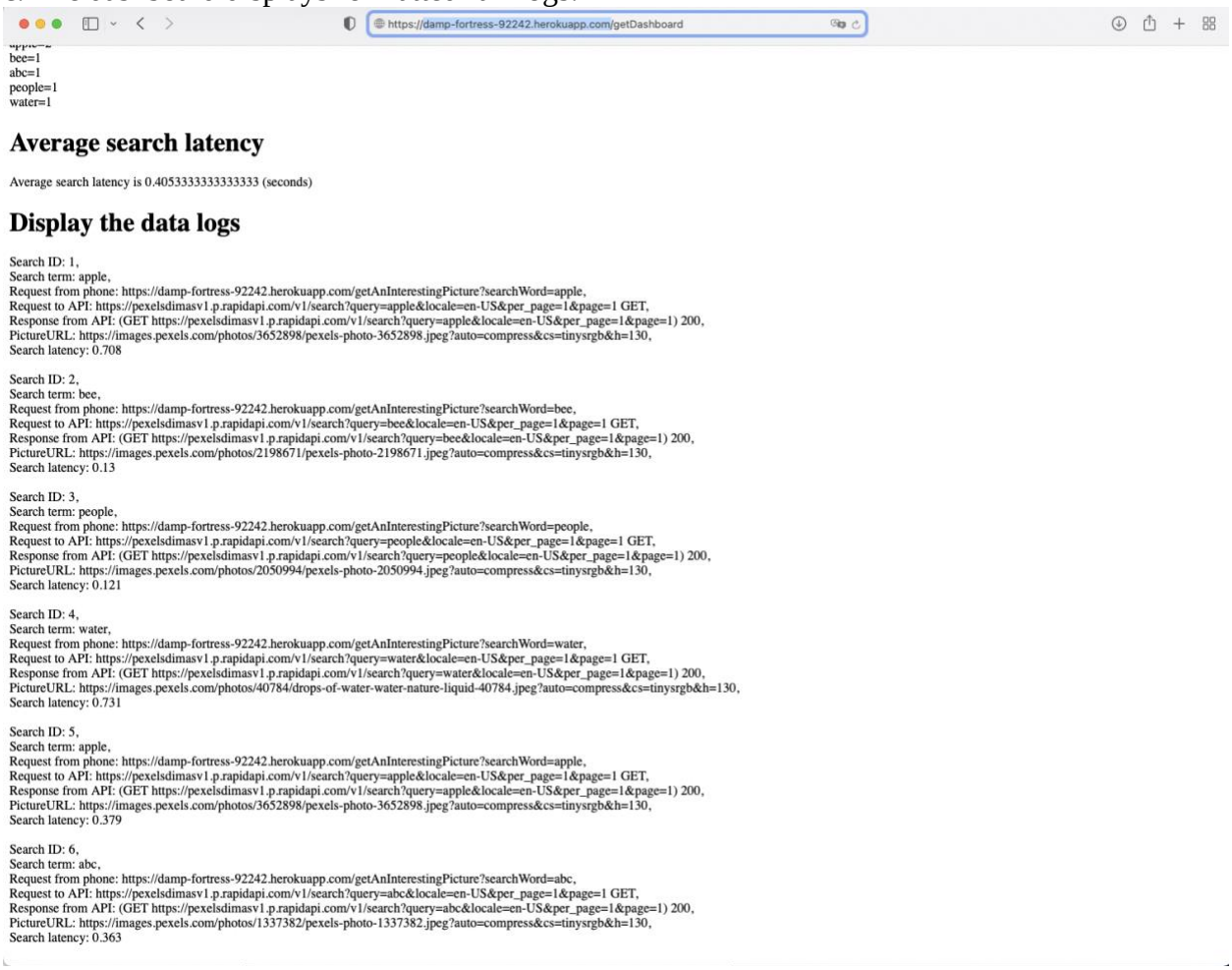
apple=2
bee=1
abc=1
people=1
water=1

Average search latency

Average search latency is 0.4053333333333333 (seconds)

Display the data logs

c. The dashboard displays formatted full logs.



4. Deploy the web service to Heroku
Url: damp-fortress-92242