Senior BackEnd Developer

Exercise

The candidate will implement a RESTful tracking system using a MongoDB database.

The API endpoints will be the following:

1. /track/<`ActivityID`>/<`UserID`>/<`Event`>
2. /activity/<`ActivityID`>
3. /user/<`UserID`>

`ActivityID` will identify the activity on which the event was performed. It will be accepted only if verifies the following conditions:

1. Contains capital case letters.
2. Contains lower case letters.
3. Contains numbers.
4. Is between 4 and 9 characters long.

The `UserID` will identify the user that generates the event for the specified activity. It will be a numeric identifier.

`Event` will identify the type of event generated. Possible values are `enter`, `click`, `exit`.

If either the `ActivityID`, the `UserID` or the `Event` fails the validation the request will reply with a `400 Invalid Request` error code.

If the request completes successfully a `200 OK` reply will be generated.

PUT `/track/<ActivityID>/<UserID>/<event>`

On invocation this API will:

1. Store, in a collection called Track_<`ActivityID`>, a document containing the following fields:
   - `ActivityID`
   - `UserID`
   - `Event`
   - Current timestamp
   - User Agent
   - IP address
2. Update, in a collection called Activities, a document identified through the `ActivityID` as follows:
   - Incrementing a separate counter for each type of event.
   - Setting, on document creation, a creation time field to the current timestamp.
   - Setting a last update field to the current timestamp.
3. Update in a collection called Users, a document identified through the `UserID` as follows:

- For each `ActivityID`, increment a separate counter for each type of event.
- Setting, on document creation, a creation time field to the current timestamp.
- Setting a last update field to the current timestamp.

GET `/activity/<ActivityID>`
On invocation this API will return, from the Activities collection, the document identified through the `ActivityID`.

GET `/user/<UserID>`
On invocation this API will return, from the Users collection, the document identified through the `UserID`.

Database
The Mongo Database is already set up and accessible through the following connection string:

```
mongodb://squaretestuser:squaretestpassword@ds045948.mongolab.com:45948/squaretestdb
```

Optional requests

Tests
Tests will be created for each API endpoint

Profiling
A simple profiling app will invoke every single API, a specifiable number of times, returning the minimum, maximum and average request time.

Profiling+
The profiling app will use a multithread approach to simulate the normal usage from multiple sources and will return the same statistics as from the previous point.