HW04: Complete ActivityLog microservice and integrate with Wolfit

Submit Assignment

Due Tuesday by 10pm **Points** 100 **Submitting** a text entry box

The purpose of this homework is to

- Complete the ActivityLog microservice by establishing a new data store used exclusively for activity logging.
- Learn about NoSQL databases such as MongoDB.
- Factor out activity logging from Wolfit and integrate with the microservice.

Completing the microservice

Last week you built the shell for the ActivityLog microservice. To complete it, you need to move from sample / dummy data to using a real data store. Once we introduce a "real" database, we need to also introduce environment management (dev and test in this case) so that we can safely develop, test, and deploy the service.

Why MongoDB

MongoDB is a NoSQL database that is ideal for storing lightly-structured data like activity logs, especially when the data comes in as JSON. It is considered "NoSQL" because it does not require a firm schema to be established up front, instead storing the data in flexible, JSON-like documents. This means that fields can vary from document to document and data structure can be changed over time. We'll see this happen in a week or two as we introduce another "activity logger" to simulate another app in our enterprise using the same microservice.

The service API

The service API and message structure will be identical to the shell you built last week, with a few qualifications:

- GET: `/api/activities/` -- will return the most recent "n" activity entries, where "n" is a service-defined variable that should be part of the environment configuration. Default this to 10.
- POST: `/api/activities` -- create a new activity entry. It will return the same JSON document that was provided, with the following additional requirements:
  - It should return a **status code of 201 (https://httpstatuses.com/201)** if the logging is successful.
  - It will add the element id to the document and fill in the native MongoDB id for the new log entry.
  - It will add the element location to the document and fill in the relative URI for the newly created entry. It would look something like `/api/activities/507f191e810c19729de860ea`

How to finish your microservice

- Continue working in your hw03 repository.
- Create a `tests` subdirectory and add Pytest to your environment by executing `pipenv install -d pytest`. You will also want to install the Python Coverage tool to evaluate test coverage.
- Either install MongoDB to your system (using the usual suspects like `brew` and `apt-get`), or use a free cloud MongoDB server. I recommend using a **MongoDB Atlas (https://www.mongodb.com/cloud/atlas)** free sandbox environment.
- Modify the three REST API endpoints to query and save (accordingly) from the MongoDB server, using a test-driven development approach with Pytest. While not required, I suggest you declare a MongoDB document that describes the schema for your log entries. It should look familiar:

```
class ActivityLog(Document):
    user_id = IntField(required=True)
    username = StringField(required=True, max_length=64)
    timestamp = DateTimeField(default=datetime.utcnow)
    details = StringField(required=True)
```

- Modify the Wolfit app to stop logging entries to the local database and instead invoke the microservice as a REST client. You can delete the latest_entry method and clean up the tests that call it. This is because latest_entry was just a hook to allow for testing of the activity log.
- The URL root (e.g., http://0.0.0.0:5001) must be a configurable element that is part of the environment loading from `dev.settings` and `test.settings`.

Turning in your work

- Use the repository that was generated for you for hw03 (will look something like hw03-*-<username>). This is the 1st URL to turn in.
- I should also have access to your Wolfit repository initiated in hw02. This is the 2nd URL.
- Turn in nothing here but the above two URLs, separated with a line break!
- Create a new subdirectory under evidence in your hw03 repo: evidence/hw4. In this folder, include:
  - Screenshot or similar evidence showing your test coverage.
  - Screenshot or similar evidence showing a query or dump from MongoDB showing that entries are flowing there. Generate entries by playing around in your dev sandbox Wolfit app (login, post, comment, vote, logout, etc.).

Wolfit integration rubric

Criteria	Ratings			Pts
Assignment submitted on time By deadline.	**5 pts** **On time**		**0 pts** **Late**	5 pts
Clean project structure	**10 pts** **Clean structure, no extra files**	**5 pts** **Some missing or extraneous files**	**0 pts** **Poor structure, many missing or extraneous files**	10 pts
Quality of commit messages Do the commit messages exhibit a high level of quality per the assignment guidelines?	**10 pts** **Excellent quality of commit messages, which sufficient volume**	**5 pts** **Quality issues with commit messages** Typos, unclear instructions, or limited volume.	**0 pts** **Very few commits with poor messages**	10 pts
Proper implementation of GET /activities	**10 pts** **Proper implementation**	**5 pts** **Partial implementation**	**0 pts** **No implementation**	10 pts
Proper implementation of GET /activities/<id>	**10 pts** **Proper implementation**	**5 pts** **Partial implementation**	**0 pts** **No implementation**	10 pts

Criteria	Ratings				Pts
Proper implementation of POST /activities	**10 pts** **Proper implementation**		**5 pts** **Partial implementation**	**0 pts** **No implementation**	10 pts
Proper implementation of tests for ActivityLogger microservice	**10 pts** **Complete and proper implementation**		**5 pts** **Partial implementation**	**0 pts** **No implementation**	10 pts
Correct modification of Wolfit to call ActivityLogger	**25 pts** **Completely functional integration**	**20 pts** **Integrated but without test modifications**	**15 pts** **Implemented but wthout error checking / handling**	**0 pts** **No implementation**	25 pts
Structure of submitted evidence	**10 pts** **Well organized and complete**		**5 pts** **Evidence missing or poorly organized**	**0 pts** **No evidence**	10 pts
					Total Points: 100