**RPOJ2**

PART A - Implement

1. Web service that supports at least four (4) unique GET and four (4) unique POST actions
2. Build a correctly formed Database to implement the specifications of PROJ1
3. Integrates with a backend database for at least two (2) CRUD actions
4. Logging feature that accounts for every request with IP, browser, timestamp and action

It saves into a apache\_access.log

A close up of a device

Description automatically generated

1. Rate limit Web Service to one request per second per user session

A screenshot of a cell phone

Description automatically generated

1. Limit per session request to 1,000 in a 24hour period

A screenshot of a cell phone

Description automatically generated

1. Domain lock web service to a whitelist of referrers

A screenshot of a cell phone

Description automatically generated

PART B - Structure

1. One database object with all SQL actions
2. One session object with all user, logging and rate limiting actions
3. A default JSON error if request signature incorrect
4. Validation of GET and POST data before it is processed by any object

A screenshot of a cell phone

Description automatically generated

1. All output from Web Service is JSON
2. HTTP Error response codes three (3) implemented

PART C - Code Comments

1. Denote where and explain why you instantiated the database and session objects in that location

Session object instantiated before Database to check whether user excessed access limit first. If not, connect to database, giving less chance to user connect to database when we don’t want them to.

1. Explain the mathematics of either of your two rate limiting code.

For 1000 request in 24hrs code. I made three variables in session. One counting every access($accessCount), one record the current time($currentTime) and one remember the first-time server received a request($firstAccess).

If $accessCount goes up to 1000, it die() and echo a json\_encode(error);

If $currentTime are 1 day or more later than $firstAccess, it unset $firstAccess and set $accessCount to 0.

1. Note where you are checking if a session pre-exists, what are you doing if it does.

I have a variable checking every time when server get a request($accessCount) as above question.

If the $accessCount goes up means session pre-exist.

If I have to reset the whole session, I will comment out the session\_destory() code;

1. Explain the code structure that checks all of the GET/POST structures

The API will connect to a single end point ‘http://localhost:8888/PROJ2/api/ws.php’.

Different GET/POST action manage by a method parameter and an id parameter

For example, create recipe will be ‘http://localhost:8888/PROJ2/api/ws.php?method=crecipe’

Update recipe will be ‘http://localhost:8888/PROJ2/api/ws.php?method=urecipe&id=1’.

1. Write a README file that explains how to setup and configure Web Service
2. Write a test script that interacts with the web service to test all the known GET and POST requests as a part of Unit testing

A screenshot of a social media post

Description automatically generated

A screenshot of a cell phone

Description automatically generated