CS3204 | Cloud Infrastructure and Services

Lab2: Web Services in the Cloud

Content

- Goals & Objectives
- Benefits of the AWS

Lab Work Content

- Task 1: Cloud Web Instance
    - AWS & Its €1 Fee
    - Rest-API & Fake Data
    - Finding Salvation in Mining
- Task 2: Cloud Web Database
    - Form Submission in PHP
    - The choice of PHP & working with MyAdmin
- Experience gained from using, testing and dubbing new software

Goals & Objectives

My goal for this 2ish week lab is to put into practice the theoretical knowledge I have acquired from lectures in relation to working with the Cloud environment and how different companies use this service to speed up initial development of their projects.

I will accomplish this by documenting my progression through the set-up of one of Amazon's Web Services (AWS) called Elastic Beanstalk, extracting data from an API source, creating my own database and providing a web-based method to upload files to it.

By the end of this Lab work, I hope to have enhanced my understanding of tools that can be used to speed up development of website building as well as to familiarise myself with services such as AWS & Elastic Beanstalk.

All code can be found in my GitHub repository by following this link:

https://github.com/Passe-Sleeper/UNI-Public/tree/main/Year3/CS3204/Lab02

Benefits of AWS

Amazon Web Services is a nextensive provider of cloud based tools that allow various people to work on projects and web applications without the need for having their own server. This leads to many benefits which include: Scalability, Flexibility, Cost-effectiveness, Reliability, etc.

A bit of detail on each of these:

Scalability: AWS allows you to easily scale your infrastructure up or down based on your needs. You can quickly add or remove resources as your business demands change, without any major disruptions.

Flexibility: AWS offers a wide range of services that cater to various business requirements. Whether you need computing power, storage, databases, machine learning, or analytics, AWS has a service to fulfil your needs.

Cost-effectiveness: With AWS, you only pay for what you use. There are no long-term contracts or upfront costs. This pay-as-you-go model can help you save costs by eliminating the need for investing in expensive hardware and infrastructure.

Reliability: AWS has a global infrastructure that is designed to provide high availability and reliability. They offer multiple data centres worldwide, ensuring that your applications and data remain accessible even in the event of a failure.

There are also a couple of disadvantages that are also associated with similar benefits, such as: Complexity, Dependency, Cost, Security, etc.

A bit on each one of these:

Complexity: AWS is a complex platform with a steep learning curve. It requires a solid understanding of various services, configurations, and management practices, which can be overwhelming for beginners or small organisations with limited technical expertise.

Dependency: By using AWS, you become dependent on their services and infrastructure. This can be a disadvantage if AWS experiences outages or disruptions, which could impact your business operations. It's important to have contingency plans in place to mitigate such risks.

Cost: While AWS offers a pay-as-you-go pricing model, costs can add up quickly. If you don't carefully monitor and manage your usage, you may end up with unexpectedly high bills. Additionally, certain services and features may incur additional charges, leading to hidden costs.

Security: As with any cloud service provider, there are security concerns when using AWS.

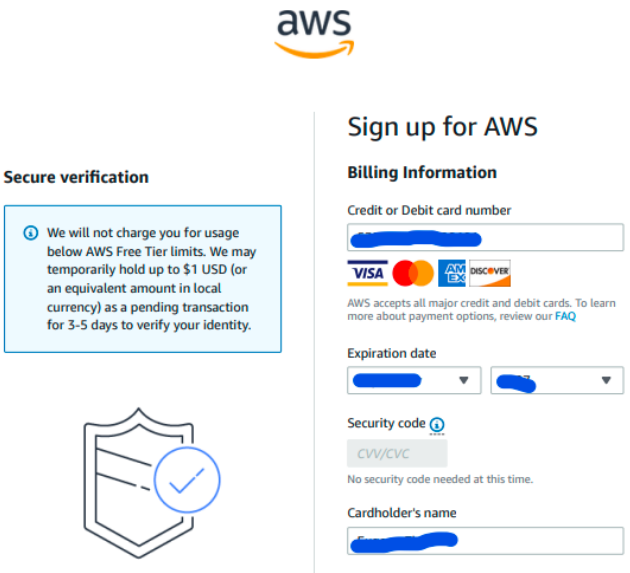
Overall, AWS offers a very wide branched service that takes time to master, is great for start-ups, but once the company grows, it is probably in their interest to start up their own websites & hosting services for security and cost reasons.

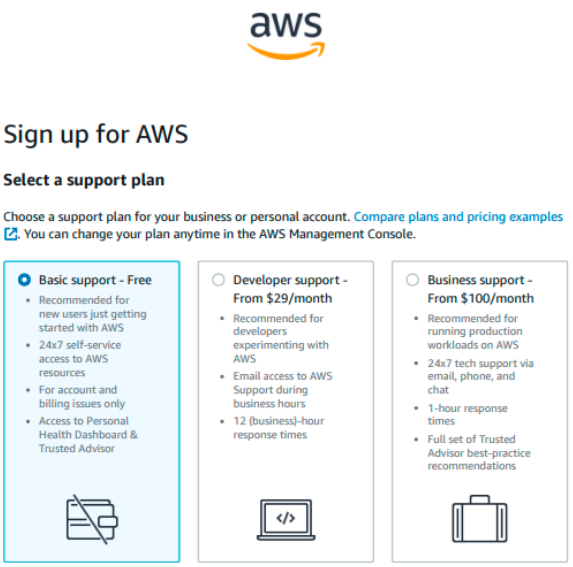
Lab Work Content

Task 1: Cloud Web Instance

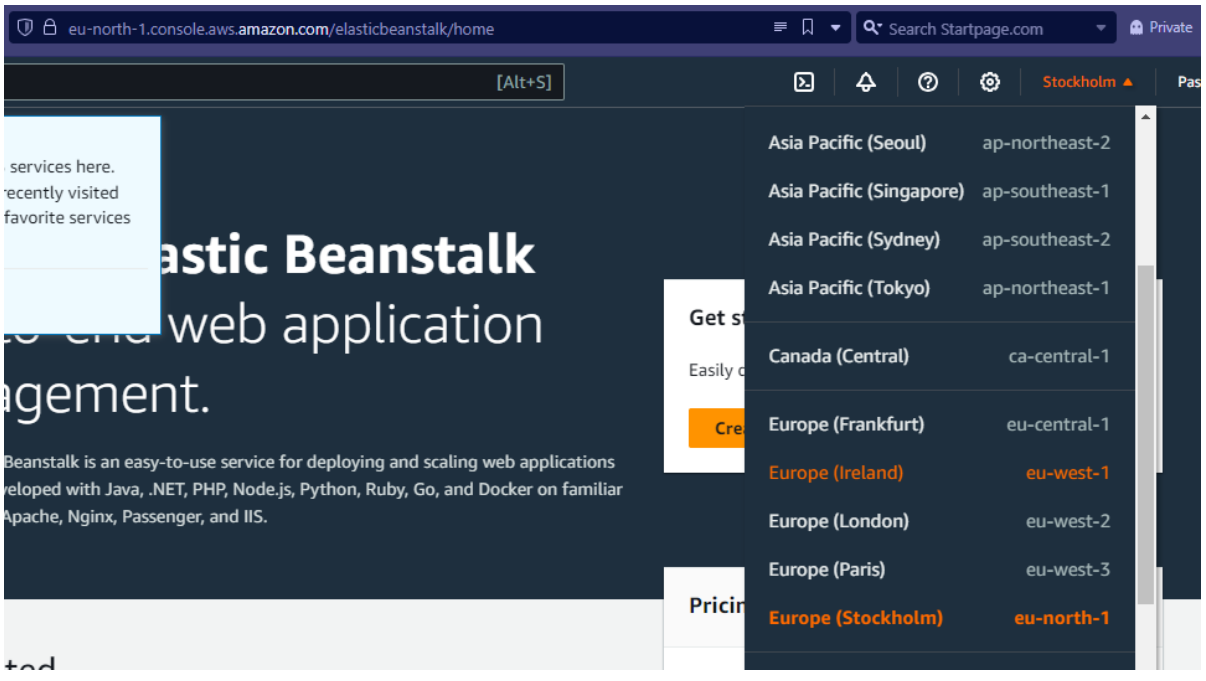
AWS & Its €1 Fee

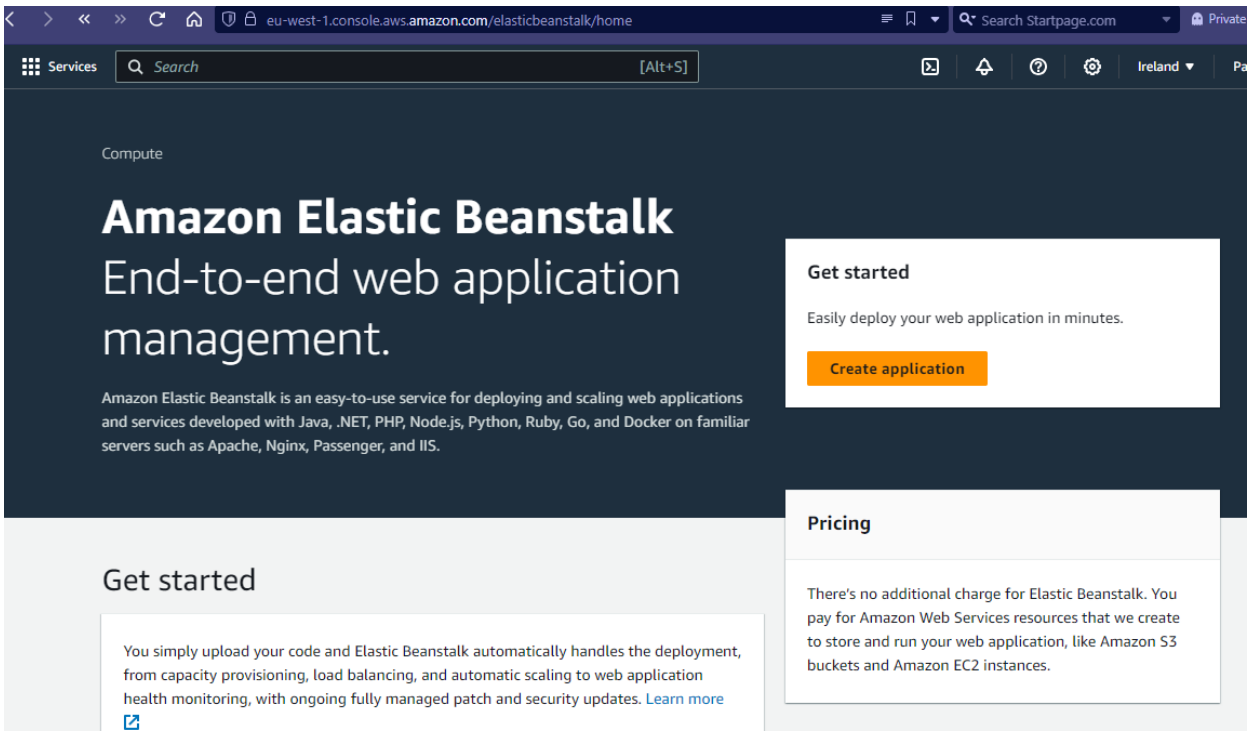
When initially following the lab instructions of signing up to AWS services, I was forced to create an account and even pay a deposit of €1! I was quite surprised at this, but, since it was just €1, it should be fine…

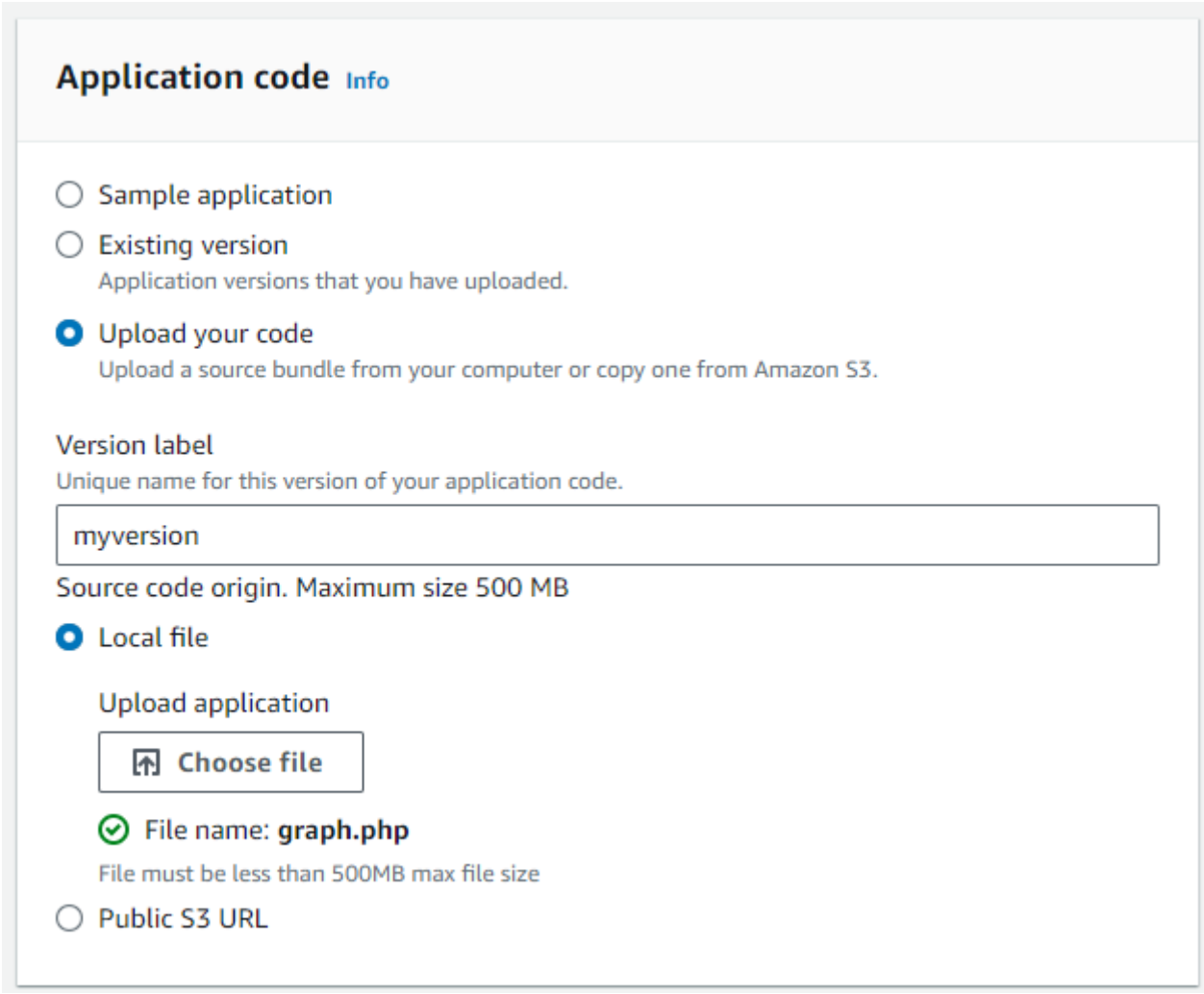




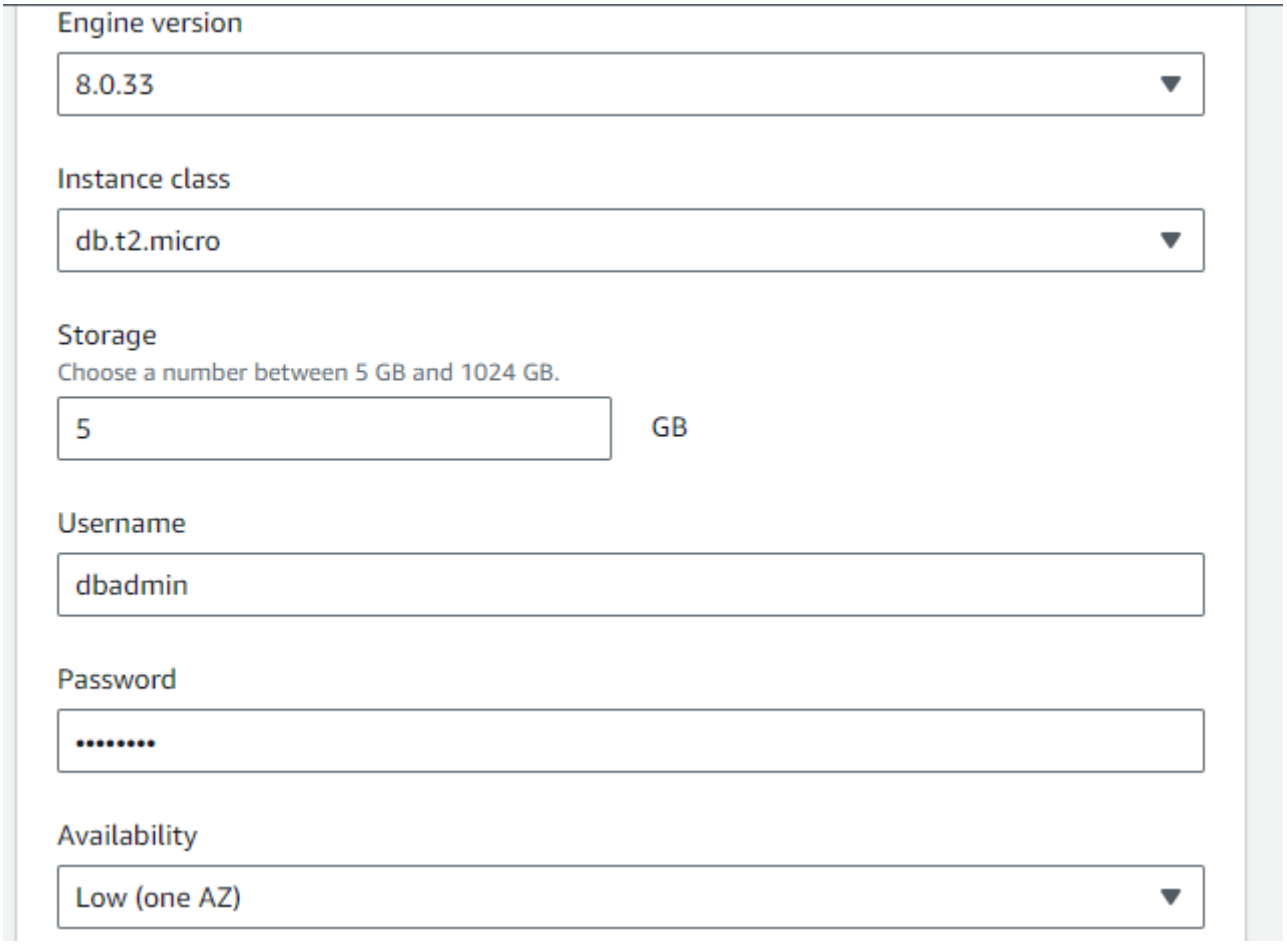
Then I continued to follow the same procedures as stated in the notes..



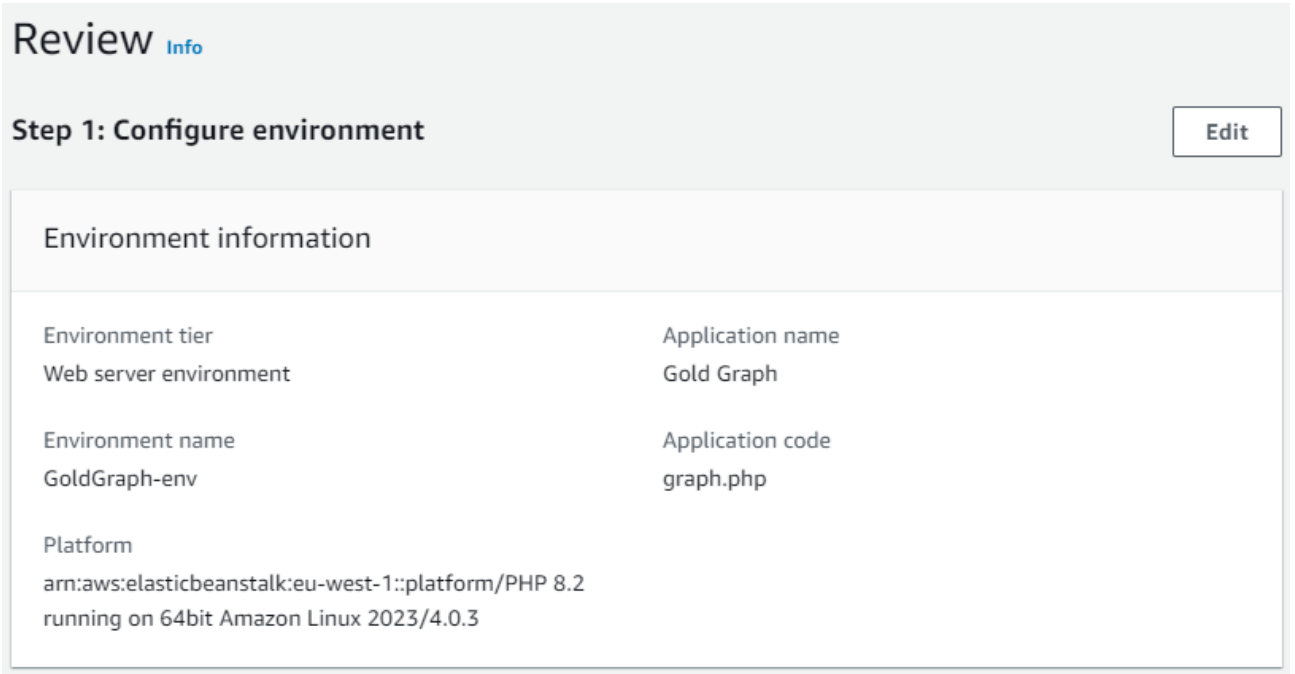

eu-west-1.console.aws.amazon.com/elasticbeanstalk/home
Search Startpage.com
Private
Services
Search
[Alt+S]
Ireland
Compute
Amazon Elastic Beanstalk
End-to-end web application management.
Amazon Elastic Beanstalk is an easy-to-use service for deploying and scaling web applications and services developed with Java, .NET, PHP, Node.js, Python, Ruby, Go, and Docker on familiar servers such as Apache, Nginx, Passenger, and IIS.
Get started
Easily deploy your web application in minutes.
Create application
Pricing
There's no additional charge for Elastic Beanstalk. You pay for Amazon Web Services resources that we create to store and run your web application, like Amazon S3 buckets and Amazon EC2 instances.
Get started
You simply upload your code and Elastic Beanstalk automatically handles the deployment, from capacity provisioning, load balancing, and automatic scaling to web application health monitoring, with ongoing fully managed patch and security updates. Learn more


Application code Info
Sample application
Existing version
Application versions that you have uploaded.
Upload your code
Upload a source bundle from your computer or copy one from Amazon S3.
Version label
Unique name for this version of your application code.
myversion
Source code origin. Maximum size 500 MB
Local file
Upload application
Choose file
File name: graph.php
File must be less than 500MB max file size
Public S3 URL

I added the Database that I would be using. Created a Username & Password as required of me.



—----------



Networking, database, and tags Info

Configure VPC settings, and subnets for your environment's EC2 instances and load balancer. Set up an Amazon RDS database that's integrated with your environment.

Database

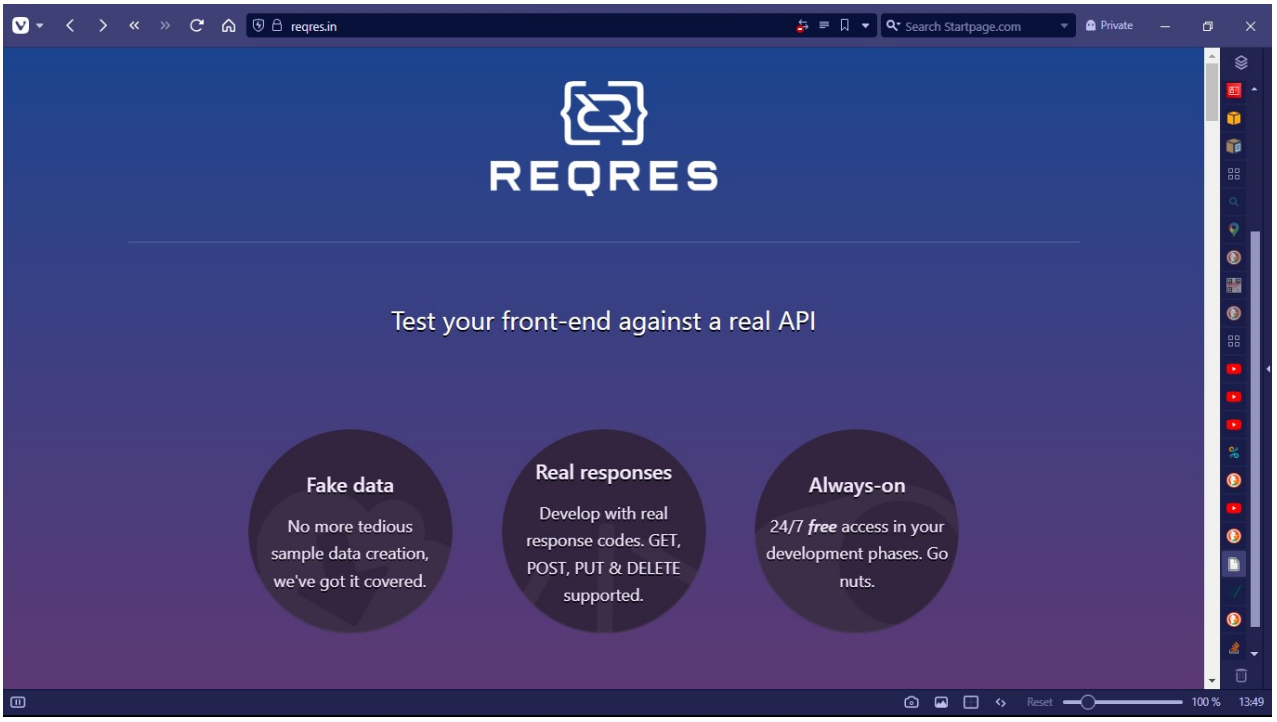
Database availability	Has coupled database	Database deletion policy
false	true	Delete
Database engine	**Database engine version**	**Database instance class**
mysql	8.0.33	db.t2.micro
Database password	**Database storage**	**Database username**
******	5	dbadmin

Tags

Key	Value
No tags	
There are no tags defined	

Rest-API & Fake Data

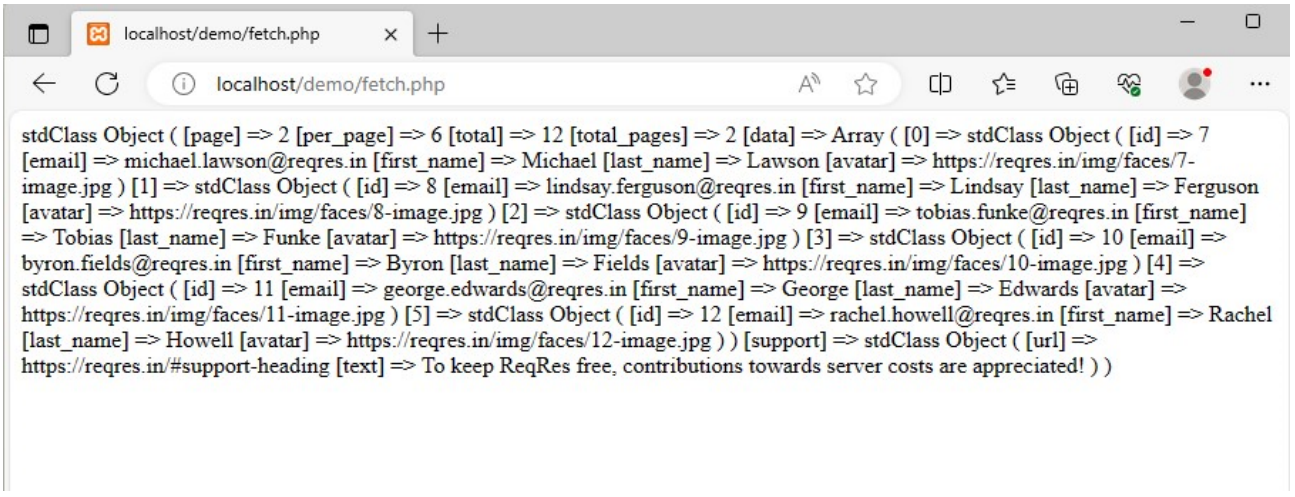
Initially I wished to import data from a website called Reqres.in, and in order to do so, I follow some online tutorials on how the Rest-API worked. I even coded up a small application to extract the code from the database and to write out everything it collected. The result was consistent with what others had done too!



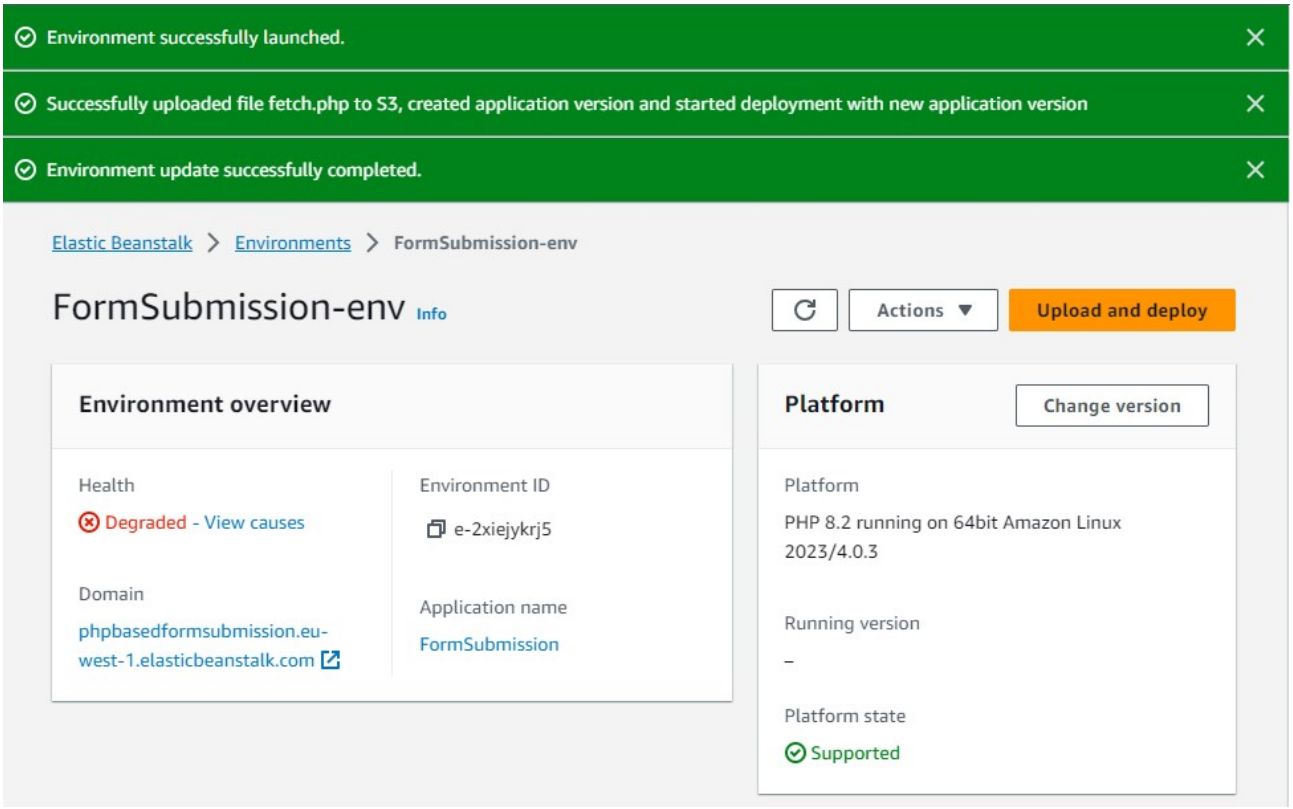
fetch.php

```php
<?php

$ch = curl_init();

$url = "https://reqres.in/api/users?page=2";

curl_setopt($ch, CURLOPT_URL, $url);
curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);

$resp = curl_exec($ch);

if ($e = curl_error($ch)) {
    echo $e;
}
else {
    $decoded = json_decode($resp);
    print_r($decoded);
}

curl_close($ch);

?>
```

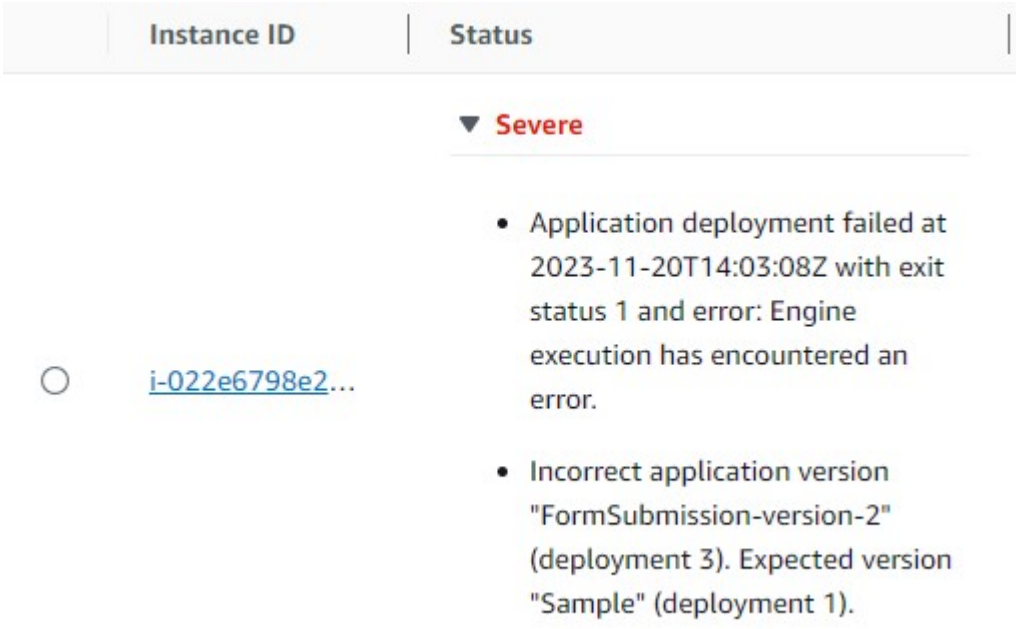
Using XAMPP, I was able to test on the local host before running it on the Cloud.



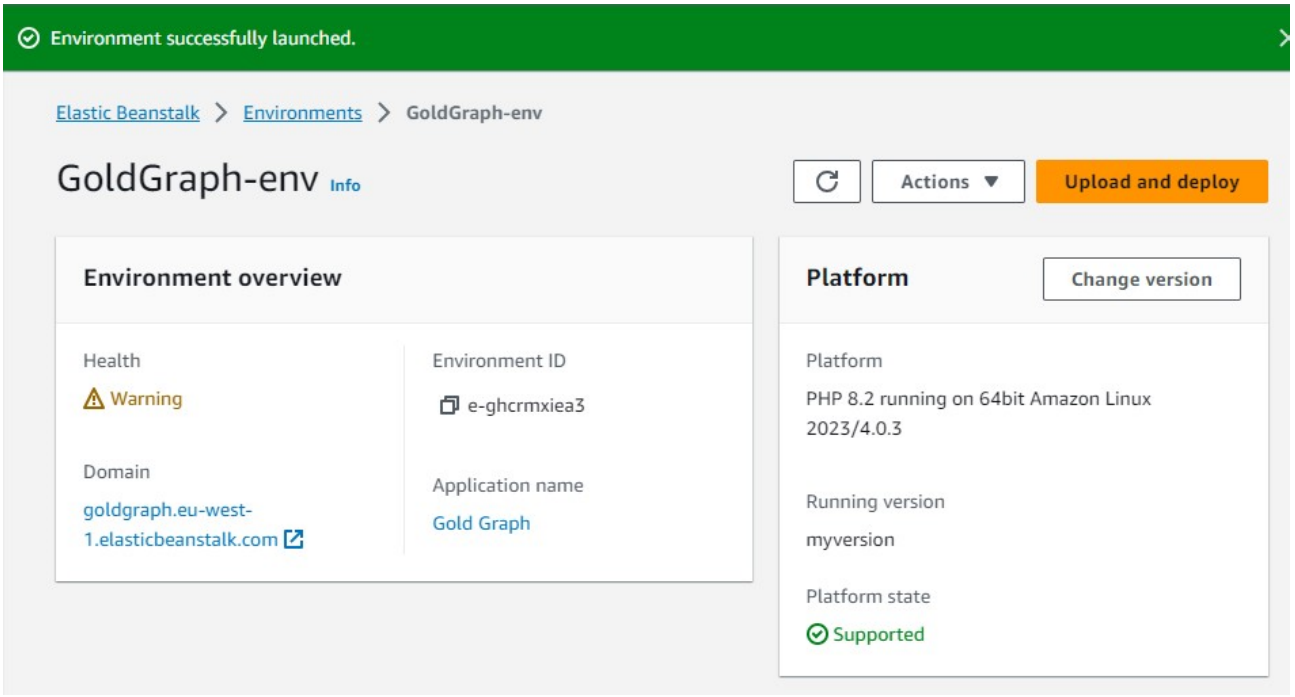
However, when attempting to run it on the AWS servers, I encountered a lot of errors in relation to the file health being Degraded cloud



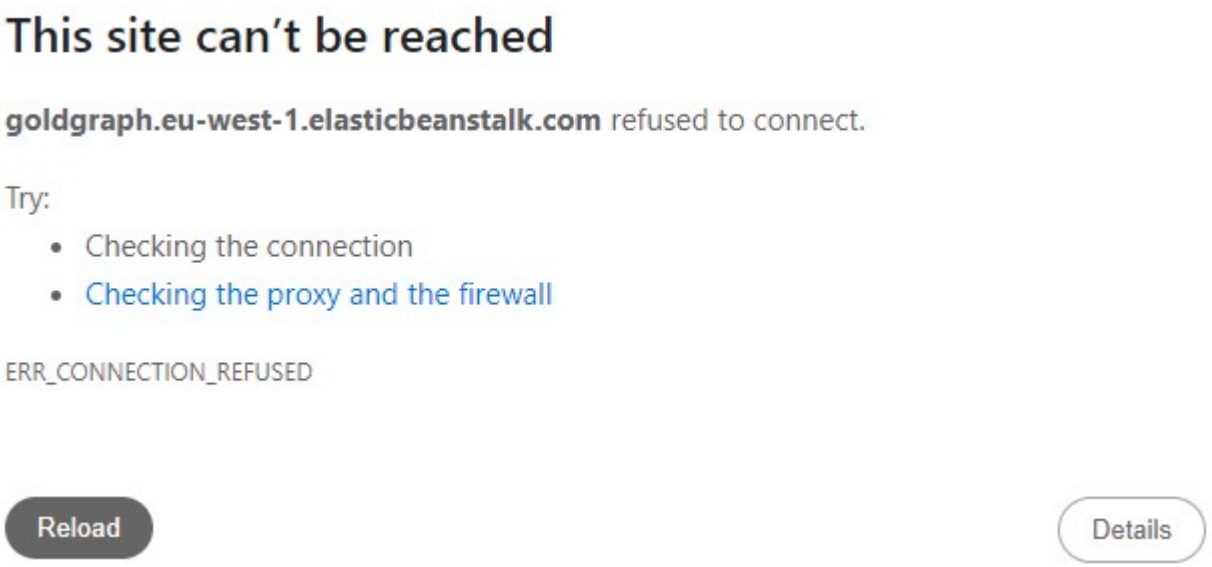
I tried to resolve the issue both via my own efforts and by googling excessively. Unfortunately, nothing worked.



I recreated the Environment and this time managed to bring down the Health Degradation to just a Warning status.



Unfortunately, when attempted to access my website I got an error stating:

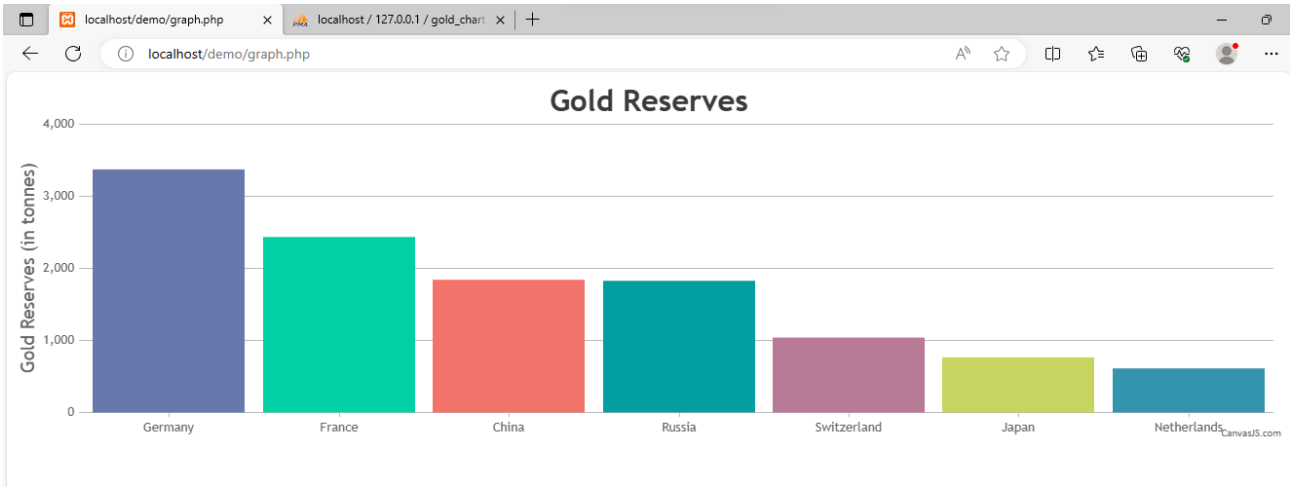


As none of this was working, I returned back to the LocalHost Apache server and continued working on my Project

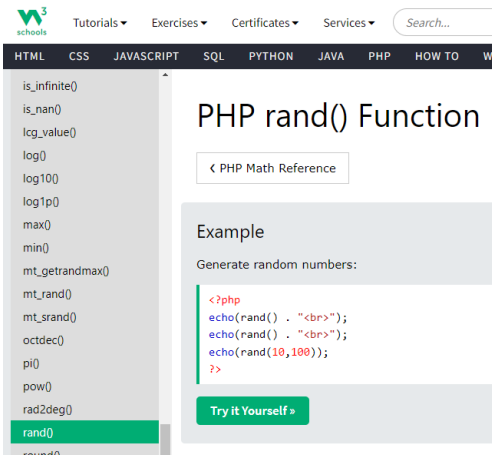
Finding Salvation in Mining

As AWS had given up on me, I decided to modify my website and make it more interactive and a bit more interesting to look at each time!

Initially I had built a static model using Canvas.js - a very handy tool for creating simple graphs for websites! This was the result:



It looked very neat & I liked the Result, however, it was quite boring to refresh the page and see the same result every single time! To make it more entertaining, I modified my data points to instead provide me with randomised results to my processes!



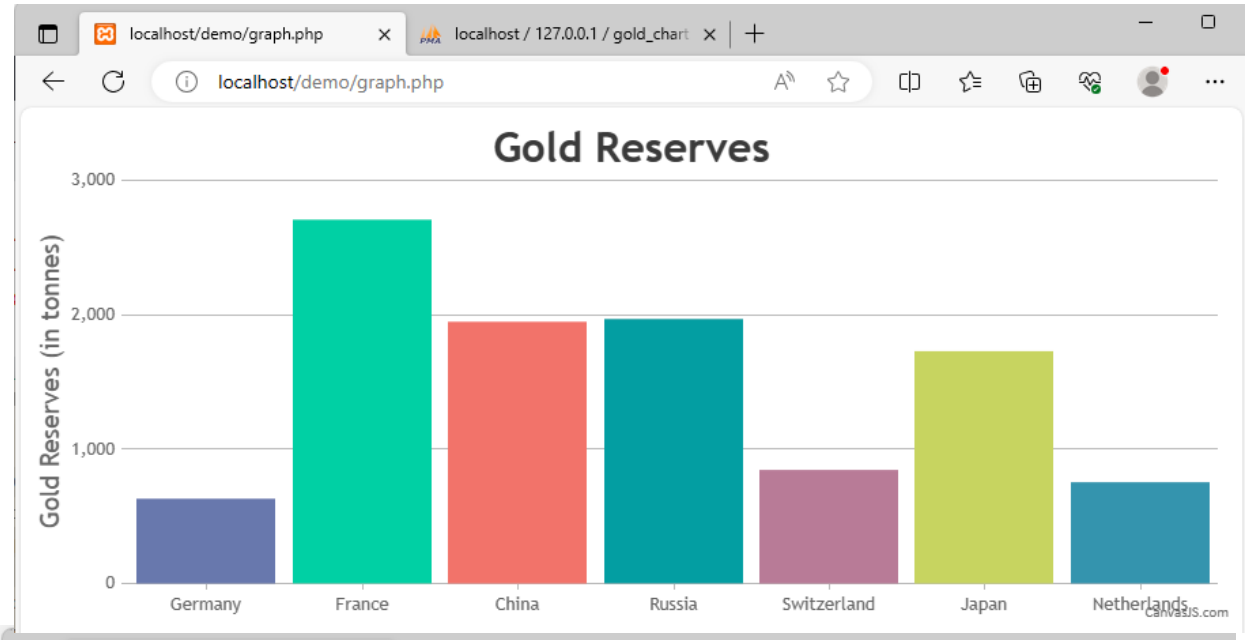

```php
$dataPointsRandom = array(
    array("y" => rand(123.47, 3691.43), "label" => "Germany" ),
    array("y" => rand(1435.21, 2739.43), "label" => "France" ),
    array("y" => rand(1942.43, 2042.23), "label" => "China" ),
    array("y" => rand(1942.43, 2042.23), "label" => "Russia" ),
    array("y" => rand(800.12, 1285.39), "label" => "Switzerland" ),
    array("y" => rand(1265.25, 2017.30), "label" => "Japan" ),
    array("y" => rand(412.453, 803.02), "label" => "Netherlands" )
);
```

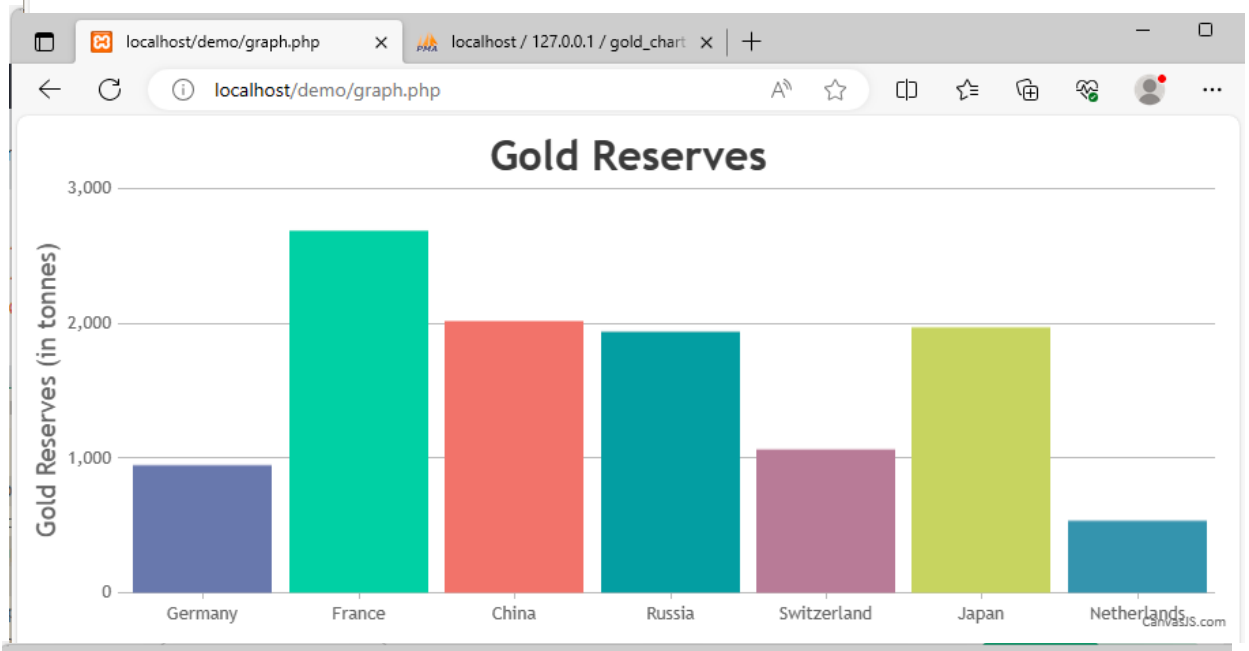
This allowed me to refresh the page and get a different result every time!

Take a look yourself! I have a few of them printed out there!

—————————

S———---


localhost/demo/graph.php
localhost / 127.0.0.1 / gold_chart
localhost/demo/graph.php
Gold Reserves
Gold Reserves (in tonnes)
3,000
2,000
1,000
0
Germany
France
China
Russia
Switzerland
Japan
Netherlands
CanvasJS.com


localhost/demo/graph.php
localhost / 127.0.0.1 / gold_chart
localhost/demo/graph.php
Gold Reserves
Gold Reserves (in tonnes)
3,000
2,000
1,000
0
Germany
France
China
Russia
Switzerland
Japan
Netherlands
CanvasJS.com


localhost/demo/graph.php
localhost / 127.0.0.1 / gold_chart
localhost/demo/graph.php
Gold Reserves
Gold Reserves (in tonnes)
3,000
2,000
1,000
0
Germany
France
China
Russia
Switzerland
Japan
Netherlands
CanvasJS.com