Technical Questionnaire

1. Explain the difference between single-quoted and double-quoted strings in PHP. Provide examples of when you would use each.

   The main difference is that in double-quoted strings variables and special characters are interpreted and replaced with their actual values while in single quoted strings they are treated as regular characters.

   Example:

   Single-quoted:

```php
<?php
$language = 'php';
echo 'I love ' . $php .'s';
```

   Double-quoted:

```php
<?php
$language = 'php';
echo "I love $language!\n";
```

2. Describe the principles of Object-Oriented Programming (OOP) in PHP. How do you define a class and create objects in PHP? Provide an example of a class and its instantiation.
   - **Encapsulation**
     - **It bundles data and methods within a class, providing a protective barrier around the internal workings of an object.**
   - **Inheritance**
     - **Allows class to inherit properties and methods from another class to promote reuse.**
   - **Polymorphism**
     - **Offers the ability of objects to take on different forms and exhibit different behaviours while sharing a common interface.**
   - **Abstraction**
     - **used to simplify complex functionalities by focusing on essential details and hiding unnecessary implementation specifics.**

   Defining a Class and Creating Objects in PHP:

```php
<?php
class User {
    public $name;
    public $email;

    public function __construct($name, $email) {
        $this->name = $name;
        $this->email = $email;
    }

    public function getUserDetails() {
```

```
        return "Name: {$this->name}, Email: {$this->email}";
    }
}

$user1 = new User('John', 'John@email.com');
$user2 = new User('Jane', 'jane@email.com');

echo $user1->getUserDetails() . "\n";
echo $user2->getUserDetails() . "\n";
?>
```

3. Explain the purpose of exception handling in PHP. How do you catch and handle exceptions in your code? Provide an example of how you would use try-catch blocks.

   Exception handling in PHP helps to gracefully manage and respond to errors that may occur during execution of code.

   The most common way I catch and handle exceptions is through try-catch blocks as shown below:

```
<?php
function divide($dividend, $divisor) {
    try {
        if ($divisor == 0) {
            throw new Exception("Cannot divide by zero.");
        }
        $result = $dividend / $divisor;
        return $result;
    } catch (Exception $e) {
        echo "Error: " . $e->getMessage() . "\n";
    }
}
```

4. Discuss different methods for connecting to a database in PHP. Describe the differences between MySQLi and PDO. Provide an example of how to perform a basic database query using one of these methods.

   The two commonly used methods are MySQLi and PDO(Php Data Objects). The differences include:

   - **PDO supports multiple database system while MySQLi only supports MySQL databases.**
   - **PDO only offers object oriented interface only while MySQLi offers both procedural and object-oriented interfaces.**
   - **MySQLi supports multiple statements in a single query while PDO only supports it by using multiple queries within a transaction.**

   Example basic database query using PDO:

```
<?php
$host = "localhost";
$username = "root";
```

```php
$password = "";
$database = "test";

try {
    $conn = new PDO("mysql:host=$host;dbname=$database", $username,
$password);
    $conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);

    $sql = "SELECT id, name, email FROM users where deleted_at is null";
    $result = $conn->query($sql);
    if ($result->rowCount() > 0) {
        echo "User Data:\n";
        while ($row = $result->fetch(PDO::FETCH_ASSOC)) {
            echo "User ID: " . $row['id'] . ", Name: " . $row['name'] . ",
Email: " . $row['email'] . "\n";
        }
    } else {
        echo "No users found.\n";
    }
} catch (PDOException $e) {
    echo "Error: " . $e->getMessage() . "\n";
}
?>
```

5. How would you protect a PHP application from common security vulnerabilities such as SQL injection and cross-site scripting (XSS)? Provide code examples or best practices for mitigating these threats.

SQL Injection

- **Use prepared statements – prevent user input from being treated as executable code:**

```php
- <?php
    $email = $_POST['email'];
    $password = $_POST['password'];

    $stmt = $pdo->prepare("SELECT * FROM users WHERE email = :email
    AND password = :password");
    $stmt->bindParam(':email', $email);
    $stmt->bindParam(':password', $password);
    $stmt->execute();
```

- **Input Validation and Sanitization:**

```php
<?php
$email = filter_var($_POST['email'], FILTER_SANITIZE_STRING);
$password = filter_var($_POST['password'], FILTER_SANITIZE_STRING);
```

Cross-Site Scripting (XSS):

- **Output Encoding:**

```php
- <?php
    echo "Welcome, " . htmlspecialchars($name) . "!";
```

- **Input Validation:**

```php
<?php
$branch = filter_var($_GET['branch'], FILTER_SANITIZE_STRING);
echo "Filtered input: " . $branch;
```

- **HTTP only Cookies:**

```php
- <?php
   setcookie('session_cookie', 'value', time() + 3600, '/', '',
   false, HttpOnly:true);
```

6. Compare and contrast the major cloud service providers (e.g., AWS, Azure, Google Cloud). Describe the advantages and use cases for each. If you were to deploy a PHP application, which cloud provider would you choose, and why?
   - **AWS**
     - **Advantages**
       - **Has a vast number of services**
       - **Most popular with a well-established community**
       - **Has large number of data centres globally with low latency**
     - **Use cases**
       - **Web hosting, scalable applications, data storage and processing, machine learning, Internet of Things (IoT), and serverless computing.**
   - **Azure**
     - **Advantages**
       - **Integrated well with Microsoft products**
       - **Strong support for hybrid scenarios**
       - **Enterprise focus**
     - **Use cases**
       - **Enterprise applications, Windows-based workloads, AI and machine learning, hybrid cloud solutions, and development with .NET technologies.**
   - **Google Cloud**
     - **Advantages**
       - **Strengths in data analytics, machine learning, and big data services.**
       - **Innovation in Containers and Kubernetes.**
       - **Fast global network infrastructure.**
     - **Use cases**
       - **Big data analytics, machine learning, containerized applications, and scenarios where a strong focus on data and analytics is required.**

I would choose AWS to host my PHP application because of its strong focus on Linux, its cloudfront service to act as CDN to distribute static and dynamic contents easily and S3 for easy storage of files.

7. Explain the concept of Infrastructure as Code and its importance in cloud infrastructure management. Provide an example of how you would define infrastructure components using a tool like Terraform or AWS CloudFormation.

   IaC allows developers and operations teams to define and automate the creation, deployment, and management of infrastructure using code.

   It enables the recreation of entire environments consistently.

   Infrastructure code can be stored in version control systems like Git, allowing for versioning, collaboration, and the ability to roll back to previous states.

   It serves as documentation, providing a clear and documented representation of the infrastructure's configuration and architecture.

   Example using Terraform to spin an EC2:

```
provider "aws" {
region = "us-east-1"
}
variable "key_name" {}

resource "aws_instance" "web-server" {
    ami           = "ami-0c55b159cbfafe1f0"
    instance_type = "t2.micro"
    key_name      = var.key_name

    tags = {
    Name = "web-server"
    }
}
```

8. Write a PHP function that takes an array of integers and returns the sum of all even numbers in the array.

```
<?php
function sumOfEvenIntegers($integers) {
    return array_sum(array_filter($integers, fn($integer) => $integer
% 2 === 0));
}
```

9. Create a PHP script that reads a text file, counts the number of words in the file, and displays the result. Ensure that your code handles file open and read errors gracefully.

```
<?php
function countWordsInFile($filename) {
    try {
        $fileHandle = fopen($filename, 'r');
        if ($fileHandle === false) {
            throw new Exception("Unable to open the file:
$filename");
```

```
        }
        $fileContents = fread($fileHandle, filesize($filename));
        if ($fileContents === false) {
            throw new Exception("Error reading file: $filename");
        }
        fclose($fileHandle);
        $wordCount = str_word_count($fileContents);
        return $wordCount;
    } catch (Exception $e) {
        echo "Error: " . $e->getMessage() . "\n";
        return false;
    }
}
$filename = 'test.txt';
$wordCount = countWordsInFile('test.txt');

if ($wordCount !== false) {
    echo "Number of words in the file '$filename': $wordCount\n";
} else {
    echo "Failed to count words in the file.\n";
}
```

10. Using PHP, make a GET request to a sample REST API (e.g., JSONPlaceholder) to retrieve a list of users. Parse the JSON response and display the user's name and email address.

```
<?php
$apiEndpoint = 'https://jsonplaceholder.typicode.com/users';

try {
    $jsonResponse = file_get_contents($apiEndpoint);
    if ($jsonResponse === false) {
        throw new Exception("Error fetching data from the API.");
    }
    $users = json_decode($jsonResponse, true);
    echo "List of Users:\n";
    foreach ($users as $user) {
        echo "Name: {$user['name']}, Email: {$user['email']}\n";
    }
} catch (Exception $e) {
    echo "Error: " . $e->getMessage() . "\n";
}
```

11. Describe how you would design an auto-scaling setup in AWS to handle a PHP application with fluctuating traffic. What services and features would you use, and provide a high-level architecture diagram if possible.

- **Use EC2 instances to host your PHP application in different availability zones.**
- **Set up an Auto Scaling Group to automatically adjust the number of EC2 instances based on CPU utilization and network metrics.**
- **Place an Elastic Load Balancer in front of the Auto Scaling Group.**
- **Use Amazon RDS for hosting the database and automatically scale using read replicas.a**

Advanced questions:

12. Write a PHP script that performs asynchronous processing using a message queue system like RabbitMQ or Redis. The script should receive a task (e.g., an email sending request) and process it in the background without blocking the main application. Demonstrate how you would set up the message queue and create a worker script to handle the tasks.

    The solution can be found in the attached zip file in a folder named: task12

    Or in directly from this github url. https://github.com/Kibedickson/test/tree/main/task12

13. Write a PHP script that serializes a large data structure (e.g., an array or object), compresses it, saves it to a file, and then unserializes and decompresses the data from the file. You can use standard PHP functions for serialization and a compression library like zlib to achieve this.

    The solution can be found in the attached zip file in a folder named: task13

    Or in directly from this github url. https://github.com/Kibedickson/test/tree/main/task13

14. Write a PHP script that integrates with a REST API protected by OAuth 2.0 authentication. Implement the OAuth 2.0 authorization code flow to obtain an access token and use that token to make authenticated requests to the API. Provide a code example that demonstrates the complete authentication and data retrieval process.

    The solution can be found in the attached zip file in a folder named: task14

    Or in directly from this github url. https://github.com/Kibedickson/test/tree/main/task14

15. Develop a PHP application that connects to an MS SQL Server database, retrieves data from multiple tables, performs a complex SQL query to join and aggregate data, and then returns the results as JSON. Demonstrate proper error handling and security measures in your code.

The solution can be found in the attached zip file in a folder named: task15

Or in directly from this github url. https://github.com/Kibedickson/test/tree/main/task15