**Technical Questionnaire**

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

**single-quoted:**

they are Enclosed in single quotes, escape sequences are treated as literal characters and are not interprated, there's no variable interpolation.

efficient because there's no parsing of variables or escape sequences.

**$name = 'Test';**

**echo 'Morning, $name!'; // Output: Morning, $name!**

**echo 'Morning, '.$name.'!'; // Output: Morning, Test!**

**double quoted:**

they are enclosed in double quotes,escape sequences are interpreted.

there's variable interpolation, meaning you can embed variables directly within the string.

**$address = 'Nairobi';**

**echo "your address is, $address!"; // Output: address, Nairobi!**

**echo "your address is, {$address}!"; // Output: your address is, Nairobi!**

**echo "your address is, " . $address . "!"; // Output: your address is, Nairobi!**

**Uses:**

single-quoted strings are used when a string is simple without variables or escape sequences, and you want to maximize on efficiency.

double-quoted strings are used when variables or escape sequences are included within the string. useful when constructing dynamic strings or embedding variables directly.

1. 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.

it's a programming paradigm that revolves around the concept of **"objects,"** which can encapsulate data and behavior. OOP in PHP allows you to structure your code in a more modular and reusable way.

**<?php**

**// Define a simple class 'Kingdoms'**

**class Kingdoms {**

**// Properties (attributes)**

**public $Kingdom;**

**public $name;**

**public function defKingdom() {**

**echo "This is Kindgom, {$this->Kingdom} and it's a {$this->name}";**

**}**

**}**

**// Create an instance (object) of the 'Kingdoms' class**

**$kingdoms1 = new Kingdoms();**

**// Set values for the object's properties**

**$kingdoms1->Kingdom = "animalia";**

**$kingdoms1->name = "cat";**

**// Call the 'defKingdom' method**

**$kingdoms1->defKingdom(); // Output: This is Kindgom, animalia and it's a cat.**

**?>**

1. 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.

The purpose of exception handling is to improve the robustness and reliability of code by allowing the handling of errors in a more structured manner. It enables one to separate error-handling code from the main logic, making code more maintainable and readable.

you can use the try, catch, and finally blocks for exception handling:

try: Contains the code that might throw an exception.

catch: Catches and handles exceptions thrown in the corresponding try block.

finally: Contains code that will be executed regardless of whether an exception was thrown or not. It is optional.

**<?php**

**function divide($numerator, $denominator) {**

**if ($denominator == 0) {**

**throw new Exception("Division by zero is not allowed.");**

**}**

**return $numerator / $denominator;**

**}**

**try {**

**// Attempt to divide 10 by 0**

**$result = divide(10, 0);**

**echo "Result: $result";**

**} catch (Exception $e) {**

**echo "Exception caught: " . $e->getMessage();**

**} finally {**

**echo " Finally block executed.";**

**}**

**?>**

1. 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.

There are several methods for connecting to a database in PHP, two commonly used ones are MySQLi (MySQL Improved) and PDO (PHP Data Objects). Both MySQLi and PDO provide secure and efficient ways to interact with databases,

MySQLi is a PHP extension specifically developed to enhance the MySQL database connectivity. It offers both a procedural and an object-oriented interface.

PDO is a database access layer providing a uniform method of access to multiple databases. It supports various database management systems, not just MySQL, making it more versatile than MySQLi.

**procedural connection example using msqli**

**<?php**

**// Connection parameters**

**$hostname = 'localhost';**

**$username = 'username';**

**$password = 'password';**

**$database = 'database';**

**// Create a connection**

**$conn = mysqli\_connect($hostname, $username, $password, $database);**

**// Check the connection**

**if (!$conn) {**

**die("Connection failed: " . mysqli\_connect\_error());**

**}**

**// Perform a basic query**

**$query = "SELECT \* FROM users";**

**$result = mysqli\_query($conn, $query);**

**// Process the result**

**while ($row = mysqli\_fetch\_assoc($result)) {**

**echo "User ID: " . $row['user\_id'] . ", Name: " . $row['name'] . "<br>";**

**}**

**// Close the connection**

**mysqli\_close($conn);**

**?>**

1. 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 Prevention**

by utilizing prepared statements and parameterized queries to ensure that user input is treated as data, not executable code.

**$stmt = $conn->prepare("SELECT \* FROM users WHERE username = ?");**

**$stmt->bind\_param("s", $username);**

**$stmt->execute();**

**$result = $stmt->get\_result();**

by using password hashing algorithms (e.g., bcrypt) to securely store passwords.

**$hashed\_password = password\_hash($password, PASSWORD\_BCRYPT);**

by Keeping your PHP version, web server, and other software up-to-date to patch security vulnerabilities.

**Cross-Site Scripting (XSS) Prevention:**

by implementing a Content Security Policy header to define which resources are allowed to be loaded, reducing the risk of XSS attacks.

**header("Content-Security-Policy: default-src 'self'");**

by sanitizing user input before displaying it in HTML to prevent XSS attacks. make use of functions like htmlspecialchars.

**$user\_input = '<script>alert("XSS");</script>';**

echo htmlspecialchars($user\_input, ENT\_QUOTES, 'UTF-8');

1. 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?

The major cloud service providers are AWS (Amazon Web Services), Azure (Microsoft Azure), and Google Cloud

**Amazon Web Services (AWS):**

**Advantages:**

* Comprehensive Service Offering: AWS provides a vast array of services, including computing power (EC2), databases (RDS), storage (S3), machine learning, and more.
* Global Presence: AWS has a broad global network with data centers in various regions, allowing for low-latency access worldwide.
* Mature Ecosystem: As one of the first major cloud providers, AWS has a mature and extensive ecosystem with a large community and a wide range of third-party integrations.

**Use Cases:**

* Scalable web applications.
* Big data and analytics.
* Machine learning and AI.
* DevOps and CI/CD pipelines.

**Microsoft Azure:**

**Advantages:**

* Integration with Microsoft Technologies: Azure seamlessly integrates with Microsoft technologies, making it an attractive option for organizations already using Microsoft products.
* Hybrid Cloud Solutions: Azure provides strong support for hybrid cloud scenarios, allowing organizations to integrate on-premises infrastructure with the cloud.

**Use Cases:**

* Enterprises with existing Microsoft infrastructure.
* .NET applications.
* Hybrid cloud solutions.
* AI and IoT applications.

**Google Cloud Platform (GCP):**

**Advantages:**

* Data and Analytics Services: GCP is known for its strong data and analytics services, including BigQuery and Dataflow.
* Open Source and Kubernetes Focus: Google has a strong focus on open source technologies and is a key contributor to Kubernetes, making it suitable for containerized applications.

**Use Cases:**

* Data analytics and machine learning.
* Containerized applications with Kubernetes.
* Serverless computing.
* Global application deployment.
* Web applications and mobile backends.

**AWS for PHP:**

Elastic Beanstalk is a Platform as a Service (PaaS) offering that simplifies the deployment of PHP applications.

EC2 instances can be used for more fine-grained control and customization.

**Azure for PHP:**

Azure App Service offers PaaS for PHP applications.

Azure Virtual Machines allow more control over the infrastructure.

**GCP for PHP:**

App Engine is a fully managed PaaS offering for PHP applications.Compute Engine provides more control and flexibility.

The choice of cloud provider would depend on various factors, including specific requirements for my PHP application, preferences, and existing infrastructure. However, each of the major cloud providers supports PHP applications, so the decision may come down to my familiarity with the platform, pricing considerations, and any additional services I may need.**( I have worked with AWS more, so when it comes to familiarity, I’d choose AWS, also because of the Scalable web applications attribute)**

1. 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.

Infrastructure as Code(Iac) is a concept in cloud computing that involves managing and provisioning computing infrastructure through machine-readable script files, rather than through physical hardware configuration or interactive configuration tools. With IaC, infrastructure components such as virtual machines, networks, storage, and other resources are defined, configured, and managed using code.

**The benefits of Infrastructure as Code include:**

* Reproducibility: IaC allows you to recreate infrastructure consistently and reliably across different environments, reducing the chances of configuration drift.
* Version Control: Infrastructure code can be version-controlled, enabling teams to track changes, collaborate, and roll back to previous versions if needed.
* Automation: IaC automates the process of provisioning and configuring infrastructure, making it faster, more efficient, and less error-prone compared to manual methods.
* Scalability: Infrastructure can be easily scaled up or down by adjusting the code, facilitating the management of dynamic workloads.

**example using AWS cloud formation**

AWS CloudFormation is an infrastructure as code service provided by AWS. It allows you to define and provision AWS infrastructure using a declarative JSON or YAML template.

a.Create a CloudFormation Template (**ec2-template.yaml**):

**Resources:**

**testEC2Instance:**

**Type: AWS::EC2::Instance**

**Properties:**

**ImageId: ami-0c55b159cbfafe1f0**

**InstanceType: t2.micro**

b. Deploy the Stack:

You can deploy the CloudFormation stack using the AWS Management Console, AWS CLI, or an SDK.

(run the commamd below if using cli) .It creates a CloudFormation stack named testEC2Stack based on the ec2-template.yaml file.

**aws cloudformation create-stack --stack-name testEC2Stack --template-body file://ec2-template.yaml**

After deployment, you can check the AWS Management Console to verify that the EC2 instance has been created.

When you're done with the resources, you can delete the CloudFormation stack using the AWS CLI: **aws cloudformation delete-stack --stack-name MyEC2Stack**

Confirm the deletion by typing **yes** when prompted.

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

Function implemetation on file **sum\_even\_numbers.php**

1. 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.

Script implemetation on file **text\_file.php**

1. 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.

Script implemetation on file **users.php**

1. 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.

Find the tw high level diagrams on file diagram.sql

Designing an auto-scaling setup in AWS for a PHP application with fluctuating traffic involves leveraging on various AWS services to dynamically adjust the infrastructure based on demand.

* **Amazon EC2 Instances:**

Use EC2 instances to host the application. These instances will form the core of the application infrastructure.

* **Amazon RDS (Relational Database Service):**

Deploy MySQL or PostgreSQL database on Amazon RDS to ensure high availability and scalability for the database layer.

* **Amazon Elastic Load Balancer (ELB):**

Set up an Application Load Balancer (ALB) to distribute incoming traffic across multiple EC2 instances. This helps in load balancing and ensures that traffic is directed to healthy instances.

* **Auto Scaling Groups:**

Create Auto Scaling Groups for the EC2 instances. This allows one to automatically adjust the number of instances based on traffic patterns, ensuring that they have enough resources during peak times and scaling down during low traffic periods.

* **Amazon CloudWatch:**

Use CloudWatch to set up alarms and triggers based on specific metrics such as CPU utilization, network traffic, or custom application metrics. These alarms will trigger Auto Scaling actions to add or remove instances dynamically.

* **Amazon S3 (Simple Storage Service):**

Store static assets, such as images, stylesheets, and JavaScript files, in S3 to offload some of the load from the web servers and improve overall performance.

* **Amazon ElastiCache:**

Implement ElastiCache to cache frequently accessed data and reduce the load on the database. This can significantly improve the performance of the application.

* **AWS Identity and Access Management (IAM):**

Configure IAM roles with the minimum required permissions for EC2 instances to interact with other AWS services securely.

* **Elastic Beanstalk (Optional):**

Consider using Elastic Beanstalk for a Platform as a Service (PaaS) approach. It abstracts away some of the complexity of managing infrastructure and allows one to focus more on the application code.

**Advanced questions:**

1. 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.

Install required libraries **composer require php-amqplib/php-amqplib**

setup RabbitMQ

Script implemetation on file **processmessage.php**

Script implemetation fro worker on file **worker.php**

start the worker in one termianal **php worker.php**

In another terminal, send a task to the message queue using the **processmessage.php** script: **php processmessage.php** "Sending a test email"

1. 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.

Script implemetation on file **serialization.php**

1. 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.

Script implemetation on file **intergration.php**

1. 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.

Script implemetation on file **fechdata.php**