Server Inventory Management System (SIMS)

SIMS provides basic inventory management services for maintaining the records of a large number of Servers for a company. SIMS is implemented using PHP in a Service Oriented Architecture style.

Prerequisites

The Prerequisites for SIMS are as follows:

- PHP>=7.3
- OpenSSL >=1.0.1
- ReallySimpleJWT
- Web Server
- PHPUnit9 for test code execution.

Source Files Description

1. Client.php - Validates the requests and securely sends it to server inventory service API implemented in ServerInventoryServiceApi.php.
2. ServerInventoryServiceApi.php - Accepts requests from Client.php and invokes server inventory management service implemented in ServerInventoryManagementService.php for performing server inventory management operation.
3. ServerInventoryManagementService.php - Implements the server inventory management service.
4. DatabaseService.php - Abstract Class providing service for the database related operation. It provides code reusability in case of implementing any other services that use database operations.
5. DBServiceInterface.php - Defines the contract for all the services to adhere to that are using database service.
6. database.sql - File containing sample SQL statements for server inventory database.
7. drivercode.php - Contains the sample implementation of server inventory management operations by using Client object.

Installing and Usage

1. Configure the web server to host ServerInventoryServiceApi.php.
2. Change the url variable found in Client.php to point to ServerInventoryServiceApi.php.
3. Change the database connection parameters if needed in ServerInventoryManagementService.php file.
4. DBServiceInterface.php, DatabaseService.php, ServerInventoryManagementService.php should be accessible by ServerInventoryServiceApi.php.
5. ReallySimpleJWT library should be accessible for both Client.php and ServerInventoryServiceApi.php.

drivercode.php provides usage examples. One of the example is listed below.

```php
<?php
//include Client.php
require("Client.php");

/**
* creating Client Object to be used for calling its functions.
*/
$cliObj = new Client();



/**
* Example for findServers call.
* Obtain server detail for IP="178.34.234.44"
*/
$data = array();
$data['columnName'] = "IP";
$data['value'] = "178.34.234.44";
$result = $cliObj->findServers($data);
echo print_r($result,true);
```

Running the tests

The test for Client.php and ServerInventoryManagementService.php are provided in ClientTest.php and ServerInventoryManagementServiceTest.php respectively. The test files are located in tests folder. To run tests, following command can be used

```
./PHPUnit9/bin/phpunit tests
//PHPUnit9 is the root directory of the testing framework.
```

Built With

- PHP 7.3
- Nginx 1.4.6
- MySql 5.5.62
- OpenSSL 1.0.1
- ReallySimpleJWT
- PHPUnit9 testing framework
- Ubuntu 14.04

Securing the data transmitted

To securely transmit the data between Client and server inventory service API, *signed JSON Web Tokens (JWT)* are used. ReallySimpleJWT PHP library is used to create and verify JWT tokens. The created token is again *encrypted* using *OpenSSL* for further increasing the security of the data being transmitted. The token contains necessary claims for the server API and Client to identify each other. The request and response data

are sent as payload with the claims in the token.

Things to improve

1. Adding or updating multiple servers with a single request can be done. Currently supports only multiple delete with a single request.
2. Encrypted JSON web tokens can be combined with OAuth protocol to further strengthen the data transmission.
3. Server inventory service API implementation should validate the Client request parameters profoundly and check for database violations (like unique constraint) before adding or updating the server details to avoid database errors.
4. Overall code structuring and organizing can be further improved.

Author

Mohamed Abdulla

LinkedIn: Abdulla

Acknowledgments

- supratims
- funkytaco
- RobDWaller
- Stack OverFlow Community