**Angular 7|6 with PHP and MySQL RESTful CRUD Example & Tutorial**

Receive new PHP tutorials.

Top of Form





Bottom of Form

In this tutorial, you'll create an example REST API CRUD Angular 7 application with PHP and MySQL back-end.

You will be creating a simple RESTful API that supports GET, POST, PUT and DELETE requests and allow you to perform CRUD operations against a MySQL database to create, read, update and delete records from a database.

For the application design, It's a simple interface for working with vehicle insurance policies. For the sake of simplicity, you are only going to add the following attributes to the policies database table:

* number which stores to the insurance policy number,
* amount which stores the insurance amount.

This is of course far from being a complete database design for a fully working insurance system. Because at least you need to add other tables like employees, clients, coverage, vehicles and drivers etc. And also the relationships between all these entities.

**Prerequisites**

In this tutorial we assume you have the following prerequisites:

* The MySQL database management system installed on your development machine,
* PHP installed on your system (both these first requirements are required by the back-end project),
* Node.js 8.9+ and NPM installed in your system. This is only required by your Angular project.

You also need to have a working experience with PHP and the different functions that will be used to create the SQL connection, getting the GET and POST data and returning JSON data in your code.

You need to be familiar with TypeScript, a superset of JavaScript that's used with Angular.

A basic knowledge of Angular is preferable but not required since you'll go from the first step until your create a project that communicates with a PHP server.

*Also read*[*PHP Image/File Upload Tutorial and Example [FormData and Angular 7 Front-End]*](https://www.techiediaries.com/php-file-upload-tutorial)

**Creating the PHP Application**

Let's start by creating a simple PHP script that connects to a MySQL database and listens to API requests then responds accordingly by either fetching and returning data from the SQL table or insert, update and delete data from the database.

Create a folder for your project:

$ mkdir angular-php-app

$ cd angular-php-app

$ mkdir backend

You create the angular-php-app that will contain the full front-end and back-end projects. Next, you navigate inside it and create the backend folder that will contain a simple PHP script that implements a simple CRUD REST API against a MySQL database.

Next, navigate into your backend project and create an api folder.

$ cd backend

$ mkdir api

Inside the api folder, create the following files:

$ cd api

$ touch database.php

$ touch read.php

$ touch create.php

$ touch update.php

$ touch delete.php

Open the backend/api/database.php file and add the following PHP code step by step:

<?php

header("Access-Control-Allow-Origin: \*");

header("Access-Control-Allow-Methods: PUT, GET, POST, DELETE");

header("Access-Control-Allow-Headers: Origin, X-Requested-With, Content-Type, Accept");

These lines are used to add response headers such as CORS and the allowed methods (PUT, GET, DELETE and POST).

Setting CORS to \* will allow your PHP server to accept requests from another domain where the Angular 7 server is running from without getting blocked by the browser by reason of the **Same Origin Policy**. In development, you'll be running the PHP server from localhost:8080 port and Angular from localhost:4200 which are considered as two distinct domains.

Next, add:

define('DB\_HOST', 'localhost');

define('DB\_USER', 'root');

define('DB\_PASS', 'YOUR\_PASSWORD');

define('DB\_NAME', 'mydb');

These variables hold the credentials that will be used to connect to the MySQL database and the name of the database.

***Note****: Make sure you change them to your actual MySQL credentials. Also make sure you have created a database with a policies table that has two number and amount columns.*

Next, add:

function connect()

{

$connect = mysqli\_connect(DB\_HOST ,DB\_USER ,DB\_PASS ,DB\_NAME);

if (mysqli\_connect\_errno($connect)) {

die("Failed to connect:" . mysqli\_connect\_error());

}

mysqli\_set\_charset($connect, "utf8");

return $connect;

}

$con = connect();

This will allow you to create a connection to the MySQL database using the mysqli extension.

That's all for the database.php file

**Implementing the Read Operation**

Now, let's implement the read operation. Open the backend/api/read.php file and add the following code:

<?php

/\*\*

\* Returns the list of policies.

\*/

require 'database.php';

$policies = [];

$sql = "SELECT id, number, amount FROM policies";

if($result = mysqli\_query($con,$sql))

{

$i = 0;

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

{

$policies[$i]['id'] = $row['id'];

$policies[$i]['number'] = $row['number'];

$policies[$i]['amount'] = $row['amount'];

$i++;

}

echo json\_encode($policies);

}

else

{

http\_response\_code(404);

}

This will fetch the list of policies from the database and return them as a JSON response. If there is an error it will return a 404 error.

**Implementing the Create Operation**

Let's now implement the create operation. Open the backend/api/create.php file and add the following code:

<?php

require 'database.php';

// Get the posted data.

$postdata = file\_get\_contents("php://input");

if(isset($postdata) && !empty($postdata))

{

// Extract the data.

$request = json\_decode($postdata);

// Validate.

if(trim($request->number) === '' || (float)$request->amount < 0)

{

return http\_response\_code(400);

}

// Sanitize.

$number = mysqli\_real\_escape\_string($con, trim($request->number));

$amount = mysqli\_real\_escape\_string($con, (int)$request->amount);

// Create.

$sql = "INSERT INTO `policies`(`id`,`number`,`amount`) VALUES (null,'{$number}','{$amount}')";

if(mysqli\_query($con,$sql))

{

http\_response\_code(201);

$policy = [

'number' => $number,

'amount' => $amount,

'id' => mysqli\_insert\_id($con)

];

echo json\_encode($policy);

}

else

{

http\_response\_code(422);

}

}

**Implementing the Update Operation**

Open the backend/api/update.php file and add the following code:

<?php

require 'database.php';

// Get the posted data.

$postdata = file\_get\_contents("php://input");

if(isset($postdata) && !empty($postdata))

{

// Extract the data.

$request = json\_decode($postdata);

// Validate.

if ((int)$request->id < 1 || trim($request->number) == '' || (float)$request->amount < 0) {

return http\_response\_code(400);

}

// Sanitize.

$id = mysqli\_real\_escape\_string($con, (int)$request->id);

$number = mysqli\_real\_escape\_string($con, trim($request->number));

$amount = mysqli\_real\_escape\_string($con, (float)$request->amount);

// Update.

$sql = "UPDATE `policies` SET `number`='$number',`amount`='$amount' WHERE `id` = '{$id}' LIMIT 1";

if(mysqli\_query($con, $sql))

{

http\_response\_code(204);

}

else

{

return http\_response\_code(422);

}

}

**Implementing the Delete Operation**

Open the backend/api/delete.php file and add the following code:

<?php

require 'database.php';

// Extract, validate and sanitize the id.

$id = ($\_GET['id'] !== null && (int)$\_GET['id'] > 0)? mysqli\_real\_escape\_string($con, (int)$\_GET['id']) : false;

if(!$id)

{

return http\_response\_code(400);

}

// Delete.

$sql = "DELETE FROM `policies` WHERE `id` ='{$id}' LIMIT 1";

if(mysqli\_query($con, $sql))

{

http\_response\_code(204);

}

else

{

return http\_response\_code(422);

}

In all operations, we first require the database.php file for connecting to the MySQL database and then we implement the appropriate logic for the CRUD operation.

**Serving the PHP REST API Project**

You can next serve your PHP application using the built-in development server using the following command:

$ php -S 127.0.0.1:8080 -t ./angular7-php-app/backend

This will run a development server from the 127.0.0.1:8080 address.

**Creating the MySQL Database**

In your terminal, run the following command to start the mysql client:

$ mysql -u root -p

The client will prompt for the password that you configured when installing MySQL in your system.

Next, run this SQL query to create a mydb database:

mysql> create database mydb;

**Creating the policies SQL Table**

Next create the policies SQL table with two number and amount columns:

mysql> create table policies( id int not null auto\_increment, number varchar(20), amount float, primary key(id));

Now, you are ready to send GET, POST, PUT and DELETE requests to your PHP server running from the 127.0.0.1:8080 address.

For sending test requests, you can use REST clients such as Postman or cURL before creating the Angular UI.

Leave your server running and open a new terminal.