Ejercicio práctico creación de servicios web

1. Para realizar el siguiente ejercicio, se necesitan los siguientes programas:
instalar XAMPP, se lo descarga de manera gratuita en:

https://www.apachefriends.org/es/download.html

2. Después de la correcta instalación, se revisa que se creó la carpeta htdocs en la siguiente ruta **C:\xampp\htdocs**

3. Ahora, se crea la base de datos en XAMPP.
Una vez ejecutado XAMPP, se procede a crear la base datos con el nombre agenda, mediante los pasos que están en la imagen.

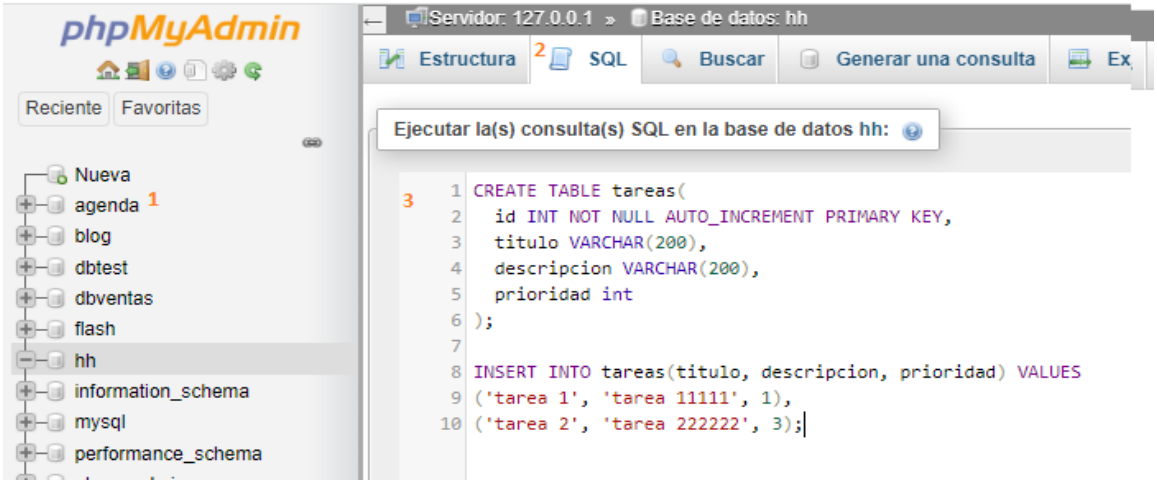
Figura 1

Base de datos 1

4. Ahora, se sigue la numeración y se pega el siguiente código.

Figura 2

Crear tabla




```
1
2
3
4
5
6
7
8
9
```

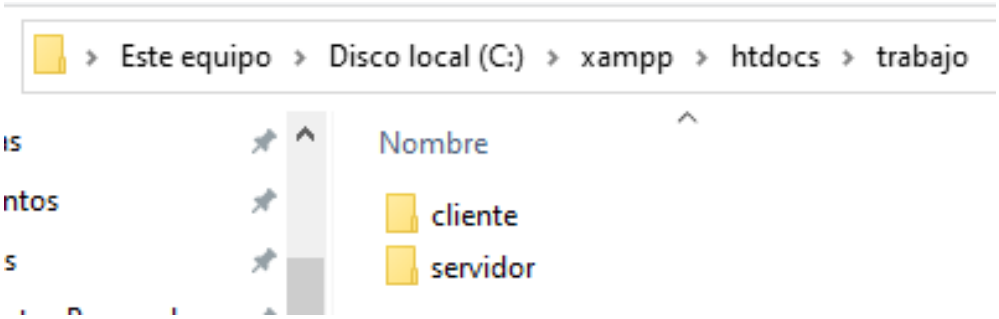
```
CREATE TABLE tareas(
  id INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
  titulo VARCHAR(200),
  descripcion VARCHAR(200),
  prioridad int
);

INSERT INTO tareas(titulo, descripcion, prioridad) VALUES
('tarea 1', 'tarea 11111', 1),
('tarea 2', 'tarea 222222', 3);
```

5. Se instala XAMPP, se descarga de manera gratuita de https://www.apachefriends.org/es/download.html
6. Después de la correcta instalación, se revisa que se creó la carpeta htdocs en la siguiente ruta **C:\xampp\htdocs**
   En esta carpeta, se crea la carpeta trabajo.
   Dentro de la carpeta de trabajo, se crean dos carpetas: “cliente” y “servidor”

Figura 3

Creación de carpetas

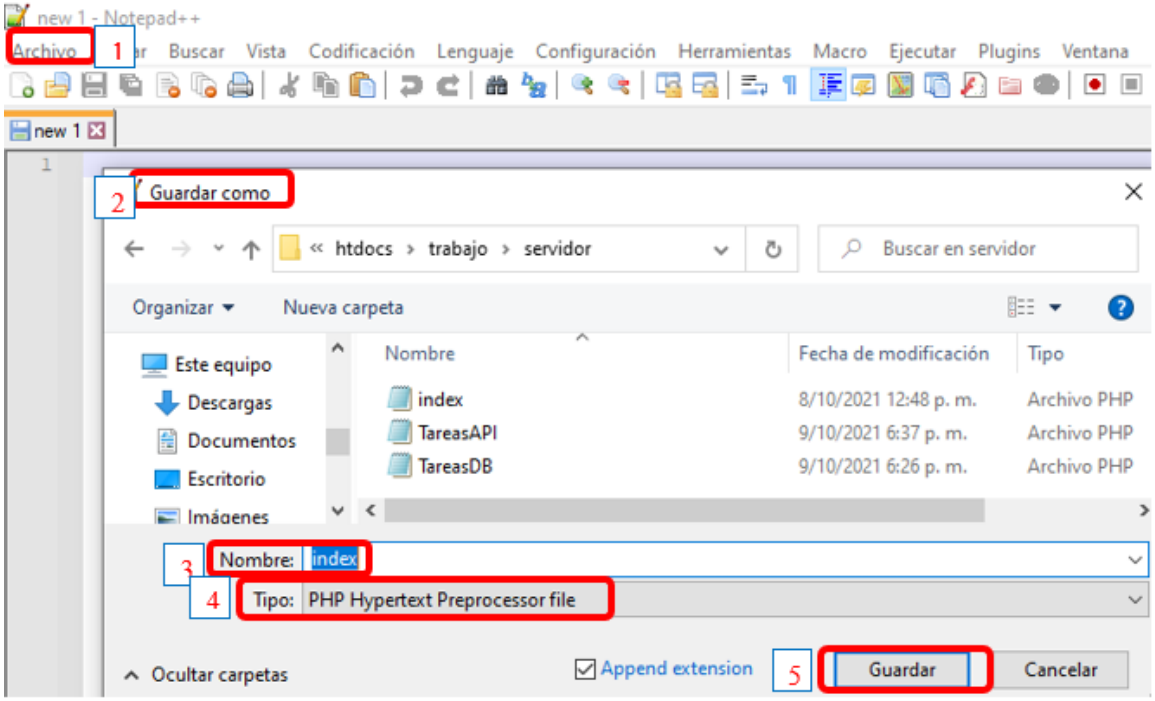


7. Se instala el programa ***Notepad++,*** que es de uso gratuito. Se puede descargar de la página oficial, dependiendo del sistema operativo que se utilice:
   https://notepad-plus-plus.org/downloads/

   - Para crear los archivos, se abre el programa *Notepad++,* se empieza creando los archivos que van dentro de la carpeta *servidor*
   - Por defecto, abre un archivo sin nombre. Se da clic en *archivo* y clic en *Guardar como* y se busca la localización de la carpeta C:\xampp\htdocs\trabajo\servidor
   - Se crea un archivo con el nombre *index*
   - Donde dice tipo de archivo, se selecciona PHP Hypertext Preprocessor file “Esta es la extensión”

Figura 4

Creación archivos PHP servidor

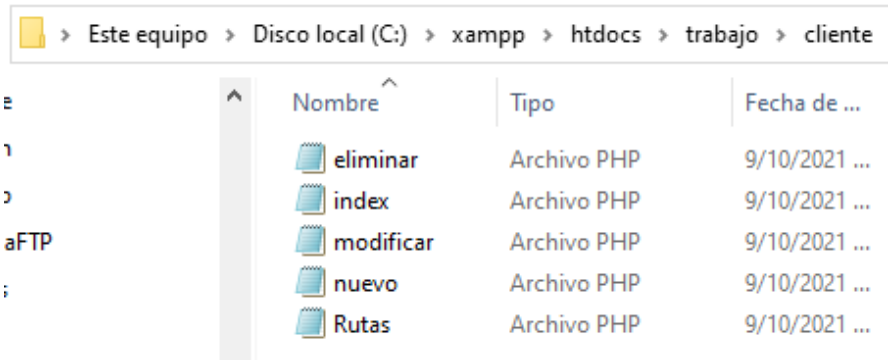


8. Se hace lo mismo para cada archivo
   Index
   tareasAPI
   TareasDB
   Nota: hay que colocar los nombres del archivo como están estipulados para que la programación se ejecute correctamente.

9. Dentro de la carpeta *cliente,* se crean los siguientes archivos “5”
   eliminar
   index
   modificar
   nuevo
   Rutas

Figura 5

Creación archivos PHP cliente



10. Ahora, se empieza a elaborar el código fuente desde el lado servidor o el proveedor de servicios, se abre el archivo index de la carpeta servidor y se escribe el siguiente código:

```
<?php
    require_once "TareasAPI.php";
    $tareasAPI = new TareasAPI();
    $tareasAPI->API();
?>
```

11. En el servidor, se crea el archivo 2 *TareasDB*

```
<?php

class TareasDB {
    protected $mysqli;
    const LOCALHOST = 'localhost'; // 127.0.0.1
    const USER = 'root';
    const PASSWORD = '';
    const DATABASE = 'agenda';

    /**
     * Constructor de clase Inicializa la variable mysqli
     */
```

```
13      public function __construct() {
14          try{
15              $this->mysqli = new mysqli(self::LOCALHOST,
16      self::USER, self::PASSWORD, self::DATABASE);
17          }catch (mysqli_sql_exception $e){
18              http_response_code(500);
19              exit;
20          }
21      }
22
23      public function dameUnoPorId($id=0){      //función que
24      retorna un registro por medio de una id
25          $stmt = $this->mysqli->prepare("SELECT * FROM tareas
26      WHERE id=? ; "); // se prepara la consulta con prepare por
27      medio de la conexión que tenemos
28          $stmt->bind_param('i', $id); // en lugar de la interrogación,
29      coloque el valor de la variable id
30          $stmt->execute();
31          $result = $stmt->get_result();
32          $tarea = $result->fetch_all(MYSQLI_ASSOC);
33          $stmt->close();
34          return $tarea;
35      }
36
37      public function dameLista(){    //esta función retorna una lista
38          $result = $this->mysqli->query('SELECT * FROM tareas');
39          $tareas = $result->fetch_all(MYSQLI_ASSOC); //aquí se
40      ejecuta la consulta
41          $result->close();
42          return $tareas;
43      }
44
45      public function guarda($titulo, $descripcion, $prioridad){ //esta
46      función guarda un registro
47          $stmt = $this->mysqli->prepare("INSERT INTO
48      tareas(titulo, descripcion, prioridad) VALUES(?, ?, ?)");
49          $stmt->bind_param('ssi', $titulo, $descripcion, $prioridad);
50          $r = $stmt->execute();
51          $stmt->close();
52          return $r;
53      }
54
55      public function elimina($id=0) { //esta función elimina un
56      registro
57          $stmt = $this->mysqli->prepare("DELETE FROM tareas
58      WHERE id = ?");
59          $stmt->bind_param('i', $id);
60          $r = $stmt->execute();
61          $stmt->close();
62          return $r;
63      }
```

```
64
65      public function actualiza($id, $titulo, $descripcion, $prioridad){
66  //esta función actualiza un registro
67          if($this->verificaExistenciaPorId($id)){
68              $stmt = $this->mysqli->prepare("UPDATE tareas SET
69  titulo=?, descripcion=?, prioridad=? WHERE id = ?");
70              $stmt->bind_param('ssii', $titulo, $descripcion,
71  $prioridad, $id);
72              $r = $stmt->execute();
73              $stmt->close();
74              return $r;
75          }
76          return false;
77      }
78
      public function verificaExistenciaPorId($id){//esta función
  verifica que exista un registro por id
          $stmt = $this->mysqli->prepare("SELECT * FROM tareas
  WHERE ID=?");
          $stmt->bind_param("i", $id);
          if($stmt->execute()){
              $stmt->store_result();
              if ($stmt->num_rows == 1){
                  return true;
              }
          }
          return false;
      }
  }
```

12. En el servidor, se crea el Archivo 3 *TareasAPI*

```
1   <?php
2   require_once "TareasDB.php";
3   class TareasAPI {
4       public function API(){
5           header('Content-Type: application/JSON');
6           $method = $_SERVER['REQUEST_METHOD'];
7           switch ($method) {
8           case 'GET':
9               $this->procesaListar();// son funciones creadas en la
10  parte de abajo de este archivo
11              break;
12          case 'POST':
13              $this->procesaGuardar();// son funciones creadas en la
14  parte de abajo de este archivo
```

```
            break;
        case 'PUT':
            $this->procesaActualizar();// son funciones creadas en
la parte de abajo de este archivo
            break;
        case 'DELETE':
            $this->procesaEliminar();// son funciones creadas en la
parte de abajo de este archivo
            break;
        default:
            echo 'MÉTODO NO SOPORTADO';
            break;
        }
    }

    function response($code=200, $status="", $message="") {
        http_response_code($code);
        if( !empty($status) && !empty($message) ){
            $response = array("status" => $status
,"message"=>$message);
            echo json_encode($response, JSON_PRETTY_PRINT);
        }
    }

    function procesaListar(){
        if($_GET['action']=='tareas'){   //se verifica la acción y se
verifica que actúe sobre la tabla tareas
            $tareasDB = new TareasDB();// aquí se instancia un
objeto de la clase tareasdb
            if(isset($_GET['id'])){       // se solicita un registro por id
                $response = $tareasDB->dameUnoPorId($_GET['id']);
                echo json_encode($response,
JSON_PRETTY_PRINT);// aquí se muestra la información en
formato json un registro por id
            }else{
                $response = $tareasDB->dameLista();        // de lo
contrario, manda la lista completa
                echo json_encode($response,
JSON_PRETTY_PRINT); // muestra la lista en formato json
            }
        }else{
             $this->response(400);
        }
    }

    function procesaGuardar(){
        if($_GET['action']=='tareas'){   // se comprueba que trabaja
en la tabla tareas
            //Decodifica un string de JSON
            $obj = json_decode( file_get_contents('php://input') );
            $objArr = (array)$obj;
```

```
66
67              if (empty($objArr)){
68                $this->response(422,"error","Nothing to add. Check
69      json");
70              }else if(isset($obj->titulo)){
71                  $tareasDB = new TareasDB();
72                  $tareasDB->guarda( $obj->titulo, $obj->descripcion,
73      $obj->prioridad );
74                  $this->response(200,"success","new record added");
75              }else{
76                $this->response(422,"error","The property is not
77      defined");
78              }
79          } else{
80            $this->response(400);
81          }
82      }
83
84      function procesaActualizar() {
85          if( isset($_GET['action']) && isset($_GET['id']) ){
86              if($_GET['action']=='tareas'){
87                  $obj = json_decode( file_get_contents('php://input') );
88                  $objArr = (array)$obj;
89                  if (empty($objArr)){
90                      $this->response(422,"error","Nothing to add. Check
91      json");
92                  }else if(isset($obj->titulo)){
93                      $tareasDB = new TareasDB();
94                      $tareasDB->actualiza($_GET['id'], $obj->titulo,
95      $obj->descripcion, $obj->prioridad );
96                      $this->response(200,"success","Record updated");
97                  }else{
98                      $this->response(422,"error","The property is not
99      defined");
                    }
                    exit;
                }
            }
            $this->response(400);
        }

        function procesaEliminar(){
            if( isset($_GET['action']) && isset($_GET['id']) ){
                if($_GET['action']=='tareas'){
                    $tareasDB = new TareasDB();
                    $tareasDB->elimina($_GET['id']);
                    $this->response(204);
                    exit;
                }
            }
            $this->response(400);
```

```
    }
}
```

13. Para profundizar en la creación *web service*, se puede hacer con este video de apoyo https://youtu.be/_2VPq7lX8Cs

14. Ahora, se van a crear los archivos que van dentro de la carpeta *cliente*

    En la carpeta cliente, se crea el archivo eliminar

```
<?php
    require_once "Rutas.php";
    $rutas = new Rutas();

    $id = $_GET['id'];
    $url = $rutas->dameUrlBase()."/servidor/tareas/".$id;

    $ch = curl_init();
    curl_setopt($ch, CURLOPT_URL, $url);
    curl_setopt($ch, CURLOPT_CUSTOMREQUEST,
"DELETE");
    $result = curl_exec($ch);
    $httpCode = curl_getinfo($ch, CURLINFO_HTTP_CODE);
    curl_close($ch);

    $resultado = "No fue posible eliminar el registro";
    if($result) {
      $resultado = "Registro eliminado exitosamente";
    }
?>

<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
    </head>
    <body>
        <?php echo
$rutas->dameMenuInicio()."    ".$rutas
->dameMenuNuevo(); ?>
        <br>
        <?php echo $resultado; ?>
    </body>
</html>
```

15. En la carpeta cliente, se crea el archivo index

```
<?php
    require_once "Rutas.php";
    $rutas = new Rutas();

    $registros = json_decode(file_get_contents($url =
$rutas->dameUrlBase().'/servidor/tareas'), true);// aquí se decodifica
o lee la respuesta archivo json
?>

<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
    </head>
    <body>
        <?php echo
$rutas->dameMenuInicio()."    ".$rutas->da
meMenuNuevo(); ?>
        <br>
        <table border="1">
            <thead>
                <tr>
                    <th>ID</th>
                    <th>TÍTULO</th>
                    <th>DESCRIPCIÓN</th>
                    <th>PRIORIDAD</th>
                    <th>OPCIONES</th>
                </tr>
            </thead>
            <tbody>
                <?php
                    foreach ($registros as $registro) {
                        echo "<tr>";
                        echo "<td>".$registro['id']."</td>";
                        echo "<td>".$registro['titulo']."</td>";
                        echo "<td>".$registro['descripcion']."</td>";
                        echo "<td>".$registro['prioridad']."</td>";
                        echo
"<td>".$rutas->dameMenuModificar($registro['id'])."  &n
bsp; ".$rutas->dameMenuEliminar($registro['id'])."</td>";
                        echo "</tr>";
                    }
                ?>

            </tbody>
        </table>
    </body>
</html>

```

```
3
1
3
2
3
3
3
4
3
5
3
6
3
7
3
8
3
9
4
0
4
1
4
2
```

16. En la carpeta cliente, se crea el archivo modificar

```
<?php
    require_once "Rutas.php";
    $rutas = new Rutas();

    $id = $_GET['id'];
    $registro = json_decode(file_get_contents($url =
$rutas->dameUrlBase().'/servidor/tareas/'.$id), true);

    if(isset($_POST['titulo'])) {
      $url = $rutas->dameUrlBase().'/servidor/tareas/'.$id;
      $data = array(
          'titulo' => $_POST['titulo'],
          'descripcion' => $_POST['descripcion'],
          'prioridad' => $_POST['prioridad']
      );

      $postdata = json_encode($data);
```

```
18
19          $ch = curl_init($url);
20          curl_setopt($ch, CURLOPT_CUSTOMREQUEST, "PUT");
21          curl_setopt($ch, CURLOPT_POSTFIELDS, $postdata);
22          curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);
23          curl_setopt($ch, CURLOPT_HTTPHEADER,
24      array('Content-Type: application/json'));
25          $result = curl_exec($ch);
26          curl_close($ch);
27        }
28      ?>
29      <!DOCTYPE html>
30      <html>
31        <head>
32          <meta charset="UTF-8">
33          <title></title>
34        </head>
35        <body>
36          <?php echo
37      $rutas->dameMenuInicio()."    ".$rutas
38      ->dameMenuNuevo(); ?>
39          <br>
40          <?php if(!isset($_POST['titulo'])) {
41            echo '<form action="" method="post" id="form1">
42            <label for="titulo">Título:</label><br>
43            <input type="text" id="titulo" name="titulo"
44      value="'.$registro[0]['titulo'].'"><br>
45            <label for="descripcion">Descripción:</label><br>
46            <input type="text" id="descripcion" name="descripcion"
47      value="'.$registro[0]['descripcion'].'"><br>
48            <label for="prioridad">Prioridad:</label><br>
49            <select name="prioridad" id="prioridad"
50      value="'.$registro[0]['prioridad'].'">
51              <option value="1">1</option>
52              <option value="2" selected>2</option>
53              <option value="3">3</option>
54            </select>
55            <br>
            <button type="submit" form="form1"
      value="Submit">Guardar</button>
          </form>';
           } else {
             echo "Registro modificado";
           }
         ?>
      </body>
      </html>
```

17. En la carpeta cliente, se crea el archivo nuevo

```
<?php
    require_once "Rutas.php";
    $rutas = new Rutas();

    if(isset($_POST['titulo'])) {
      $url = $rutas->dameUrlBase().'/servidor/tareas';
      $data = array(
          'titulo' => $_POST['titulo'],
          'descripcion' => $_POST['descripcion'],
          'prioridad' => $_POST['prioridad']
      );

      $postdata = json_encode($data);

      $ch = curl_init($url);
      curl_setopt($ch, CURLOPT_POST, 1);
      curl_setopt($ch, CURLOPT_POSTFIELDS, $postdata);
      curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);
      curl_setopt($ch, CURLOPT_HTTPHEADER,
array('Content-Type: application/json'));
      $result = curl_exec($ch);
      curl_close($ch);
    }
?>
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
    </head>
    <body>
        <?php echo
$rutas->dameMenuInicio()."    ".$rutas
->dameMenuNuevo(); ?>
        <br>
        <?php if(!isset($_POST['titulo'])) {
          echo '<form action="" method="post" id="form1">
            <label for="titulo">Título:</label><br>
            <input type="text" id="titulo" name="titulo"><br>
            <label for="descripcion">Descripción:</label><br>
            <input type="text" id="descripcion"
name="descripcion"><br>
            <label for="prioridad">Prioridad:</label><br>
            <select name="prioridad" id="prioridad">
              <option value="1">1</option>
              <option value="2" selected>2</option>
              <option value="3">3</option>
            </select>
            <br>
            <button type="submit" form="form1"
value="Submit">Guardar</button>
```

```
          </form>';
        } else {
          echo "Registro guardado";
        }
        ?>
    </body>
</html>
```

18. En la carpeta cliente, se crea el archivo Rutas

```
<?php

class Rutas {

    protected $urlBase = "http://localhost/trabajo";

    public function __construct() {

    }

    public function dameUrlBase() {
        return $this->urlBase;
    }

    public function dameMenuInicio() {
        return '<a
href="'.$this->urlBase.'/cliente/index.php">Inicio</a>';
    }

    public function dameMenuNuevo() {
        return "<a
href="'.$this->urlBase."/cliente/nuevo.php'>Nuevo</a>";
    }

    public function dameMenuModificar($id) {
        return "<a
href="'.$this->urlBase."/cliente/modificar.php?id=".$id."'>Modific
ar</a>";
    }

    public function dameMenuEliminar($id) {
        return "<a
href="'.$this->urlBase."/cliente/eliminar.php?id=".$id."'>Eliminar
</a>";
    }
}
```