

El presente documento pretende servir de brújula para poder explicar como fue resuelto este conjunto de ejercicios:

Sistema de control de inventario el cual únicamente se podrá mostrar y agregar vehículos al sistema:

Pantalla de Administración.

Cerrar sesión

Catalogo vehiculos

Agregar vehiculo

ID	Tipo vehículo	Número de llantas	Potencia (HP)
1	SEDAN	4	100
2	MOTOCICLETA	2	150

Formulario de captura de vehículo:

Agregar vehículo

Tipo vehículo

Potencia de motor (HP)

CancelarGuardar

Selección de tipo de vehículo

Agregar vehículo

Tipo vehículo

Sedan

Sedan

Motocicleta

Cancelar

Guardar

Captura de Potencia en HP (caballos de Fuerza):

Agregar vehículo

Tipo vehículo

Sedan

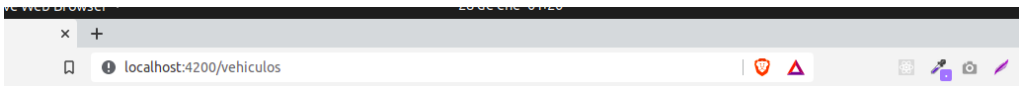
Potencia de motor (HP)

300

Cancelar

Guardar

Una vez capturado, se muestra en pantalla.



Cerrar sesión

Catalogo vehiculos

Agregar vehiculo

ID	Tipo vehículo	Número de llantas	Potencia (HP)
1	SEDAN	4	100
2	MOTOCICLETA	2	150
3	SEDAN	4	300

Ejemplo de formulario con nuevos registros.

Actividades Brave Web Browser 28 de ene 01:27

Frontend localhost:4200/vehiculos

[Cerrar sesión](#)

Catalogo vehiculos

[Agregar vehículo](#)

ID	Tipo vehículo	Número de llantas	Potencia (HP)
1	SEDAN	4	100
2	MOTOCICLETA	2	150
3	SEDAN	4	300
4	MOTOCICLETA	2	500
5	SEDAN	4	5000
6	MOTOCICLETA	2	3500

En caso de que los datos no estén completos o correctos se muestra mensaje de warning.

Agregar vehículo

Tipo vehículo

Sedan

Potencia de motor (HP)

¡Es requerido capturar todos los datos correctamente!

Cancelar

Guardar

A nivel de programación hay dos secciones relevantes para esta sección:

1. Frontend

El archivo `frontend/app/templates/vehiculos.hbs` muestra el template handlebars del listado

```

        <th scope="col">Número de llantas</th>
        <th scope="col">Potencia (HP)</th>
    </tr>
</thead>
<tbody>
    {{#each this.vehiculosData as |vehiculo|}}
        <tr>
            <th scope="row">{{vehiculo.vehiculo_k}}</th>
            <td>{{vehiculo.tipo_vehiculo}}</td>
            <td>{{vehiculo.numero_llantas}}</td>
            <td>{{vehiculo.potencia_motor}}</td>
        </tr>
    {{/each}}
</tbody>
</table>
</div>
```

El archivo `frontend/app/routes/vehiculos.js` se incluye el controlador del listado de datos.

```

21
22     setupController = (controller, model) => {
23         super.setupController(controller, model);
24         this.reloadData(this);
25         var _scope = this;
26         window.reloadData = () => {
27             _scope.reloadData(_scope);
28         };
29     };
30
31     reloadData = (_scope) => {
32         $.ajax({
33             type: 'POST',
34             url: 'http://localhost:8080/index.php/gilasw/vehiculo/listar',
35             success: function (response) {
36                 var data = JSON.parse(response);
37                 _scope.controller.set('vehiculosData', data.data);
38             },
39         });
40     };
41 }
42
```

El archivo `frontend/app/components/agregar-vehiculo-component.hbs` incluye el handlebars del formulario.

```
17
18     <form>
19         <div class="form-group">
20             <label for="tipoFormControlSelect">Tipo vehículo</label>
21             <select class="form-control" id="tipoFormControlSelect">
22                 <option value=null></option>
23                 <option value='SEDAN'>Sedan</option>
24                 <option value='MOTOCICLETA'>Motocicleta</option>
25             </select>
26         </div>
27         <div class="form-group">
28             <label for="potenciaMotorFormControlText">Potencia de motor (HP)</label>
29             <input type="number" class="form-control" id="potenciaMotorFormControlText" min="1" max="10000" />
30         </div>
31
32         {{#if this.hasError}}
33
34         <div class="form-group">
35             <br/>
36             <div class="alert alert-warning" role="alert">
37                 ¡Es requerido capturar todos los datos correctamente!
38             </div>
39         </div>
40
41         {{/if}}
42     </form>
43
44 </div>
```

El archivo `frontend/app/components/agregar-vehiculo-component.js` contiene el los comportamientos del formulario.

```
agregar-vehiculo-component.js - gilasw - Visual Studio Code

Run Terminal Help

agregar-vehiculo-component.hbs JS agregar-vehiculo-component.js 9+ X

frontend > app > components > JS agregar-vehiculo-component.js > AgregarVehiculoComponentComponent > closeModal

33     }
34     this.hasError = true;
35     return;
36 }
37
38 var params = {
39     tipo_vehiculo: $('#tipoFormControlSelect').val(),
40     potencia_motor: $('#potenciaMotorFormControlText').val(),
41 };
42
43 var self = this;
44
45 $.ajax({
46     type: 'POST',
47     url: 'http://localhost:8080/index.php/gilasw/vehiculo/agregar',
48     data: params,
49     success: function (response) {
50         var responseObject = JSON.parse(response);
51         if (responseObject.success != true) {
52             self.hasError = true;
53             return;
54         }
55         window.reloadData();
56         $('#vehiculoModal').modal('hide');
57     },
58 });
59
60 }
61
```

2. Backend

El backend esta administrado en tres capas:

EL controlador backend/app/Controllers/Gilasw/Vehiculo.php es el responsable de recibir la petición del frontend y hacer las validaciones necesarias.

```
Vehiculo.php
backend > app > Controllers > Gilasw > Vehiculo.php
1  <?php
2
3  namespace App\Controllers\Gilasw;
4
5
6  use App\Controllers\MC_Controller;
7
8  class Vehiculo extends MC_Controller{
9
10     public function __construct(){
11
12         parent::__construct();
13
14         $this->bi = new \App\Libraries\Gilasw\VehiculoBI();
15     }
16
17
18     public function listar(){
19         $responseObject = $this->bi->listar();
20         echo $this->getSuccess( $responseObject );
21         exit;
22     }
23
24     public function agregar(){
25
26         $tipo_vehiculo = $this->request->getPost('tipo_vehiculo');
27         $potencia_motor= $this->sanitizeInt( $this->request->getPost('potencia_motor') );
28         $numero_llantas= 0;
29
30         /* Validaciones en backend */
```

Una segunda capa es la capa de Bussisnes (BI). Esta capa cuando es necesario. Incluye implementación de reglas de negocio.

```
backend > app > Libraries > Gilasw > VehiculoBI.php
1  <?PHP
2
3  namespace App\Libraries\Gilasw;
4
5  class VehiculoBi{
6
7      public function __construct(){
8          $this->dao = new \App\Models\Gilasw\VehiculoDAO();
9      }
10
11     public function listar(){
12
13         $responseObject = $this->dao->listarVehiculos();
14         return $responseObject;
15     }
16
17     public function insertRecordIntoTableVehiculos( $insertRecord ){
18
19         $vehiculo_k = $this->dao->insertRecordIntoTableVehiculos( $insertRecord );
20
21         $responseObject = (object) array(
22             'success' => true,
23             'data'    => (object) array( 'vehiculo_k' => $vehiculo_k )
24         );
25
26         return $responseObject;
27     }
28
29 }
30
31
```

Partial support, click to resolve

Ln 30, Col 1 Spaces: 4 UT

La capa final el Model backend/app/Models/Gilasw/VehiculoDAO.php es el archivo que tiene acceso a la BD.

```
Run Terminal Help
Vehiculo.php • VehiculoBI.php • VehiculoDAO.php X
backend > app > Models > Gilasw > VehiculoDAO.php
3 namespace App\Models\Gilasw;
4
5 use App\Models\MC_Model;
6
7 class VehiculoDAO extends MC_Model{
8
9     function listarVehiculos(){
10
11         $db      = \Config\Database::connect();
12         $builder = $db->table( 'gilasw_vehiculos' );
13         $query   = $builder->get();
14
15         $recordObject = (object) array(
16             'numFilas' => $builder->countAll(),
17             'data'     => $query->getResult()
18         );
19
20         return $recordObject;
21     }
22
23     public function insertRecordIntoTableVehiculos( $recordObject ){
24
25         $vehiculo_k = $this->insertRecordIntoTable( $recordObject, 'gilasw_vehiculos' );
26
27         return $vehiculo_k;
28     }
29
30 }
31
32
33
```

Partial support, click to resolve Ln 1, Col 1 Spaces: 4 UTF-8 LF php

En Base de datos se almacenan todos estos datos.

Query 1

Limit to 1000 rows

1 • select * from gilasw_vehiculos;

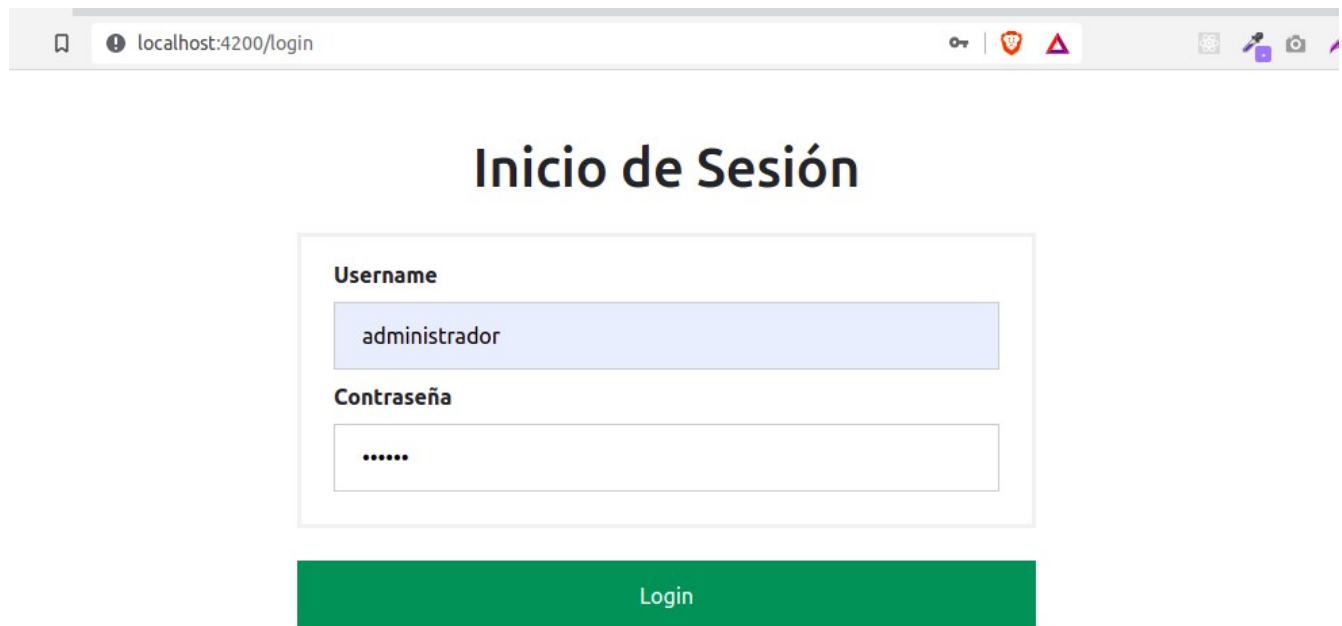
Result Grid

#	vehiculo_k	tipo_vehiculo	numero_llanta	potencia_moto	activo	usu
1	1	SEDAN	4	100	1	0
2	2	MOTOCICLETA	2	150	1	0
3	3	SEDAN	4	300	1	0
4	4	MOTOCICLETA	2	500	1	0
5	5	SEDAN	4	5000	1	0
6	6	MOTOCICLETA	2	3500	1	0
*	NULL	NULL	NULL	NULL	NULL	NULL

gilasw_vehiculos 3

Action Output

Incluye pantalla de inicio de sesión.



The screenshot shows a web browser window with the address bar displaying 'localhost:4200/login'. The page has a white background and a large, bold, black title 'Inicio de Sesión' centered at the top. Below the title is a light gray rectangular box containing two input fields. The first field is labeled 'Username' and contains the text 'administrador'. The second field is labeled 'Contraseña' and contains six dots, indicating a password. Below these fields is a solid green rectangular button with the text 'Login' in white.

localhost:4200/login

Inicio de Sesión

Username

administrador

Contraseña

.....

Login

El login se implemento en el archivo frontend/app/templates/login.hbs

```
login.hbs M
frontend > app > templates > login.hbs > html > body > div.container > div.row > div.col-6
109 | </body>
110 | </html>
111 |
112 |
113 | <script>
114 |
115 |     function dologin() {
116 |
117 |         $("input[name=username]").val();
118 |         $("input[name=password]").val();
119 |
120 |         var params = {
121 |             username: $("input[name=username]").val(),
122 |             password: $("input[name=password]").val()
123 |         };
124 |
125 |         $.ajax({
126 |             type: 'POST',
127 |             url: 'http://localhost:8080/index.php/gilasw/autenticacion/login',
128 |             data: params,
129 |             success: function (response) {
130 |
131 |                 var data = JSON.parse(response);
132 |
133 |                 if (data.success == true) {
134 |                     window.location.href = data.redirect;
135 |                 }
136 |             },
137 |         });
138 |     };
139 |
140 | }
```

partial support, click to resolve

Ln 99, Col 1 Spaces: 2

En el backend el controlador backend/app/Controllers/Gilasw/Autenticacion.php es el responsable de administrar el inicio de sesión y fin de sesión. Codeigniter ofrece herramientas para gestionar de manera muy sencilla el uso de la sesión.

```
Go Run Terminal Help
... login.hbs M Autenticacion.php M
backend > app > Controllers > Gilasw > Autenticacion.php
7 class Autenticacion extends Controller{
8
9     protected $session;
10
11     public function __construct(){
12         $this->session = session();
13     }
14
15     public function login(){
16
17         $username      = $this->request->getPost('username');
18         $password       = $this->request->getPost('password');
19
20         if( $username !== 'administrador' || $password !== 'gilasw' ){
21             echo json_encode( (object) array( 'success' => false, 'error' => 'Credenciales incorrectas' ));
22             exit;
23         }
24         $newdata = [
25             'username' => $username,
26             'isLoggedIn' => TRUE,
27         ];
28         $this->session->set($newdata);
29         echo json_encode( (object) array(
30             'success' => true,
31             'redirect' => 'http://localhost:4200/vehiculos'
32         ));
33         exit;
34     }
35
36     public function logout(){
37         $this->session->destroy();
38     }
39 }
```

Online: [Partial support, click to resolve](#)

Comentario: para simplificar la solución se validó directamente el username y contraseña en el controlador. Sin embargo para un entorno productivo se debe hacer tres cambios importantes:

1. Buscar en Base de datos que la combinación de username y contraseña estén guardado para un único usuario.
2. La contraseña debe estar guardada en base de datos con algún algoritmo de encriptación.
3. Por seguridad se debe utilizar un valor de SALT para mantener mas segura la contraseña.

Igualmente se implementó la API publica tanto para listar como para agregar elementos.

The screenshot shows the Postman interface with a GET request to `http://localhost:8080/index.php/gilasw/publicAPI/listarvehiculos...` returning a 200 OK status. The response body is a JSON array of vehicle records.

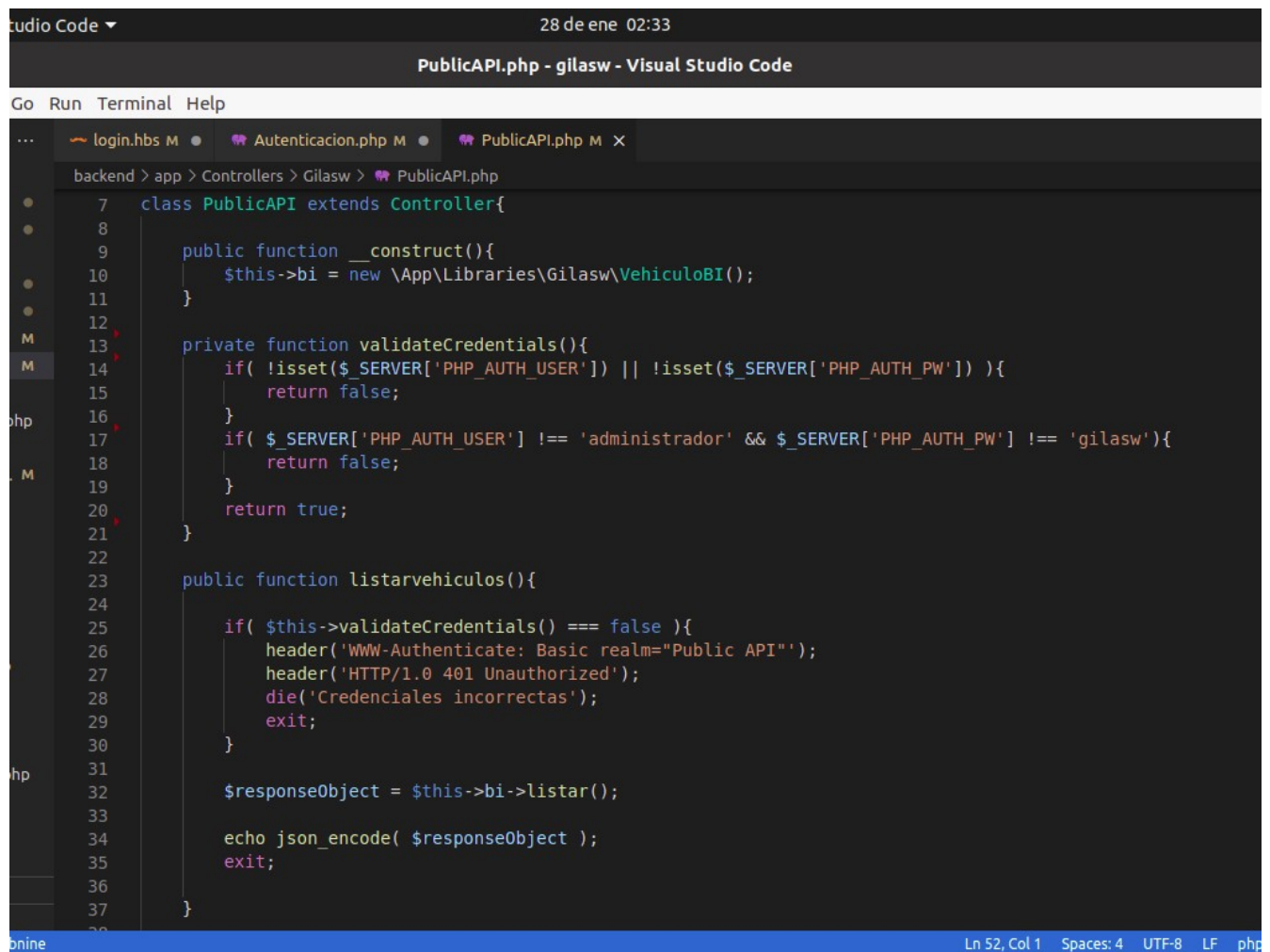
Query Params

KEY	VALUE	DESCRIPTION
Key	Value	Description

Body (Status Code: 200 OK)

```
1 [{"numFilas":19,"data":[{"vehiculo_k":"1","tipo_vehiculo":"MOTOCICLETA","numero_llantas":"2","potencia_motor":"2","activo":"1","usuario_creacion":"0","fecha_hora_creacion":null,"usuario_modificacion":null,"fecha_hora_modificacion":null,"usuario_eliminacion":null,"fecha_hora_eliminacion":null,"vehiculo_k":"2","tipo_vehiculo":"SEDAN","numero_llantas":"4","potencia_motor":"100","activo":"1","usuario_creacion":"0","fecha_hora_creacion":null,"usuario_modificacion":null,"fecha_hora_modificacion":null,"usuario_eliminacion":null,"fecha_hora_eliminacion":null,"vehiculo_k":"3","tipo_vehiculo":"SEDAN","numero_llantas":"4","potencia_motor":"15000","activo":"1","usuario_creacion":"0","fecha_hora_creacion":null,"usuario_modificacion":null,"fecha_hora_modificacion":null,"usuario_eliminacion":null,"fecha_hora_eliminacion":null,"vehiculo_k":"4","tipo_vehiculo":"MOTOCICLETA","numero_llantas":"2","potencia_motor":"1500","activo":"1","usuario_creacion":"0","fecha_hora_creacion":null,"usuario_modificacion":null,"fecha_hora_modificacion":null,"usuario_eliminacion":null,"fecha_hora_eliminacion":null,"vehiculo_k":"5","tipo_vehiculo":"MOTOCICLETA","numero_llantas":"2",
```

Para la implementación de la API publica, se utilizó el método de Auth Basic.



```
7 class PublicAPI extends Controller{
8
9     public function __construct(){
10         $this->bi = new \App\Libraries\Gilasw\VehiculoBI();
11     }
12
13     private function validateCredentials(){
14         if( !isset($_SERVER['PHP_AUTH_USER']) || !isset($_SERVER['PHP_AUTH_PW']) ){
15             return false;
16         }
17         if( $_SERVER['PHP_AUTH_USER'] !== 'administrador' && $_SERVER['PHP_AUTH_PW'] !== 'gilasw' ){
18             return false;
19         }
20         return true;
21     }
22
23     public function listarvehiculos(){
24
25         if( $this->validateCredentials() === false ){
26             header('WWW-Authenticate: Basic realm="Public API"');
27             header('HTTP/1.0 401 Unauthorized');
28             die('Credenciales incorrectas');
29             exit;
30         }
31
32         $responseObject = $this->bi->listar();
33
34         echo json_encode( $responseObject );
35         exit;
36     }
37 }
```

Ln 52, Col 1 Spaces: 4 UTF-8 LF php