

BASE DE DATOS NO RELACIONAL

ADMINISTRACION DE SERVICIOS DE RED BAJO LINUX

MANUAL TECNICO

Profesora: Msig. Adriana E. Collaguazo Jaramillo

Grupo #3

Elaborado por:

- Luis Illapa Guamán
- Darwin Albán Marcillo
- Javier Fierro Paladines

Fecha de inicio: lunes, 9 de diciembre de 2019

Fecha de fin: viernes, 31 de enero de 2020

Resumen ejecutivo

Con las apariciones de las primeras computadoras el mundo se ha visto por evolucionar en el área tecnológico haciendo que su avance tecnológico es cada vez sea más eficiente, el almacenamiento y procesamiento de información es una de las tareas fundamentales. En la actualidad existen gestores de bases de datos donde tienen el grado de satisfacer las tareas previstas por el usuario, también abarca su complejidad de la información por medio del tratado de la misma con sus diferentes propósitos y búsquedas.

Por medio de este proyecto se involucra en el tema del gestor de base de dato, pero en este caso es de una base no relacional de datos en el cual el que hace posible esto es el servidor MongoDB donde se podrá crear las colecciones indicadas donde tendrá clave-valor para su respectiva manipulación de sus datos.

Para alcanzar tales fines, se ha realizado por la creación de sitio web que por medio del sitio se dará cuenta que se trata del propósito de un gimnasio de calidad donde abarcara y podrá interactuar con la persona que estaría al mando del sitio en este caso los administradores y por otro lado también los empleados con ciertas limitaciones de acceso con la finalidad de hacer solo lo necesario teniendo en cuenta el cumplimiento del trabajo propuesto.

Aclarando un poco más esté presente proyecto, se ha elaborado con la finalidad de llevar un monitoreo de la información que se maneja en un Gimnasio, cuyo lugar se registra una afluencia masiva de personas cada día, por lo tanto, es necesario la implementación de un sistema de gestor de base de datos para almacenar la información detallada de los clientes. Adicionalmente para una mejor interacción y manipulación de la información, los registros se las realizara mediante una página web interactiva que tendrán acceso solo los empleados. En esta página web básicamente se podrá realizar acciones como registrar o eliminar usuarios, ingresar y renovar membresía del cliente, y por supuesto realizar consultas según se requiera.

Descripción del problema

El proyecto se desarrolla en la implementación de una base de datos no relacional, que detalle la estructura del manejo de datos de un Gimnasio, que muestre la información detallada de todos los clientes y empleados. Cada usuario tendrá un rol con lo cual podrá realizar diferentes acciones para gestionar y manipular los datos. Mediante una página web el usuario podrá iniciar sesión con su respectivo privilegio para realizar el registro de los datos, y además verificar la búsqueda del contenido en específico que desee investigar, esta a su vez se reflejará en el sistema de gestor de datos. Esto se lo desarrollara, implementando MongoDB, para la creación de la base de datos, y node.js para la presentación de la base de datos gráficamente en la página web.

Objetivos específicos

- A base de la teoría base datos no relacional se tendrá como propósito la creación de una base que se pueda gestionar de forma rápida con la funcionalidad de poder modificaciones de los posibles clientes del sistema.
- Diseñar una página web para el registro y control de la información de los usuarios del Gimnasio.
- Implementar un sistema de gestor de base de datos no relacional para almacenar los datos en formato JSON.

Funcionamiento

El funcionamiento del proyecto se genera, con la implementación de una base de datos no relacional, que se desarrollara en base a un diagrama de árbol para cada objeto que intervenga en el Gimnasio.

Estos datos se ingresan y se presentan en una página web, que con la ayuda de node server express de node.js, escribiendo la ruta en la aplicación de Express para extraer los datos que se necesita para ingresar a su página desde la base de datos y responder con los datos.

Se emplea una plantilla para construir una página con los datos respectivos, que permitirá la presentación grafica de esta estructura de la base de datos, y permitirá realizar búsquedas de los campos o parámetros solicitados por el usuario, con el fin de proporcionar una mejor interacción de búsqueda de información para el usuario, en cuanto a la información detallada que necesita buscar.

Hardware

Entorno virtualizado

Dispositivo	Cantidad
Servidor de virtualización Centos 7	1

Tabla 1. Hardware requerido para solución virtualizada

Software

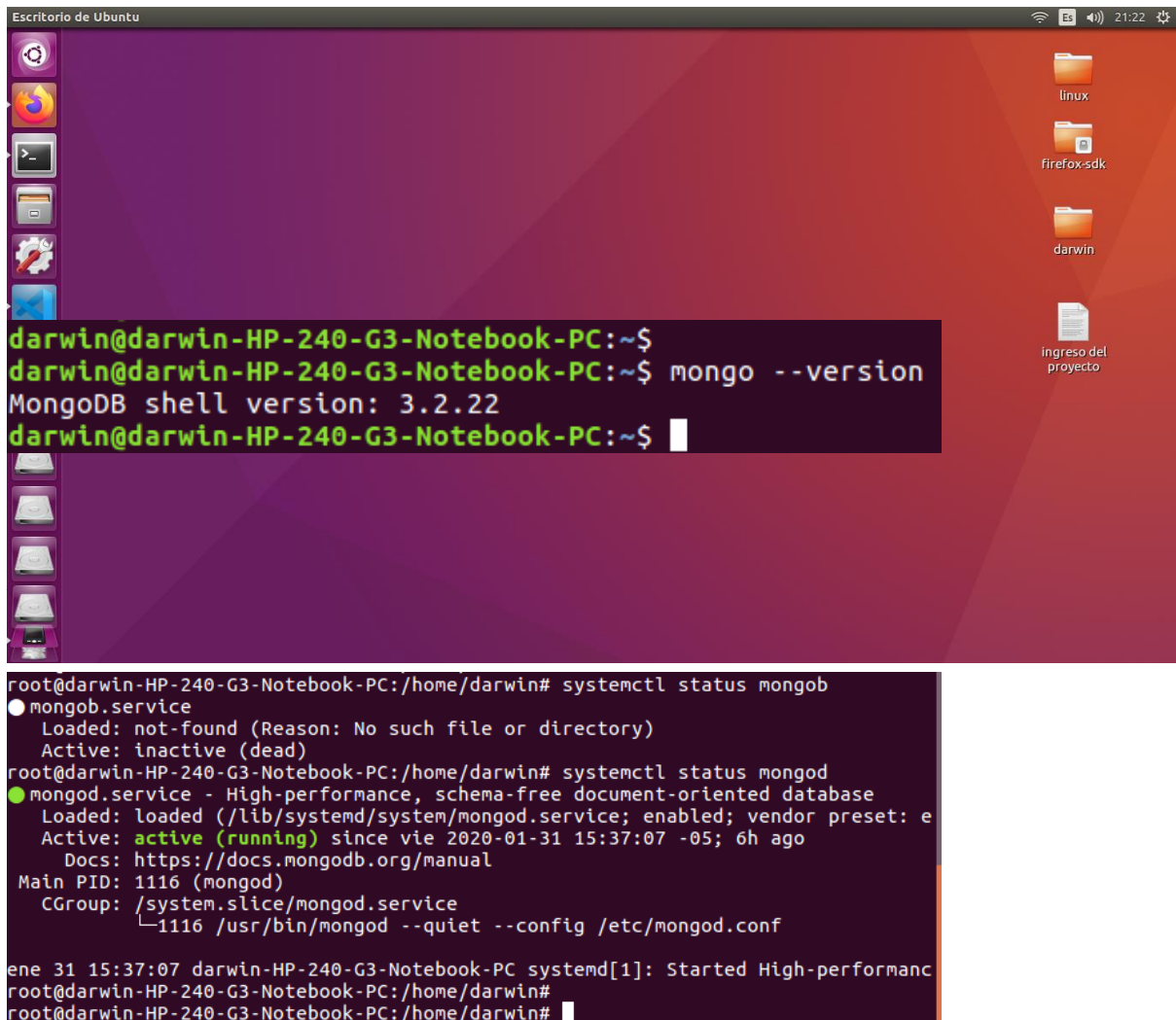
- Ubuntu 18.04 LTS
- Visual Studio
- MongoDB
- Node.js
- Bootstrap

Se utilizará MongoDB ya que es un sistema de base de datos NoSQL orientado a documento de código abierto. Guarda los datos con un esquema más dinámico haciendo que la integración de datos en ciertas aplicaciones sea más fácil y rápida.

Para la creación de la página web en la parte de Backend se usará Node js donde es un entorno en tiempo de ejecución multiplataforma. Y para la parte frontend se usará Vue js donde es un marco JavaScript de código abierto Modelo-vista-modelo de vista para construir interfaces de usuario y aplicaciones de una sola página.

Implementación del proyecto

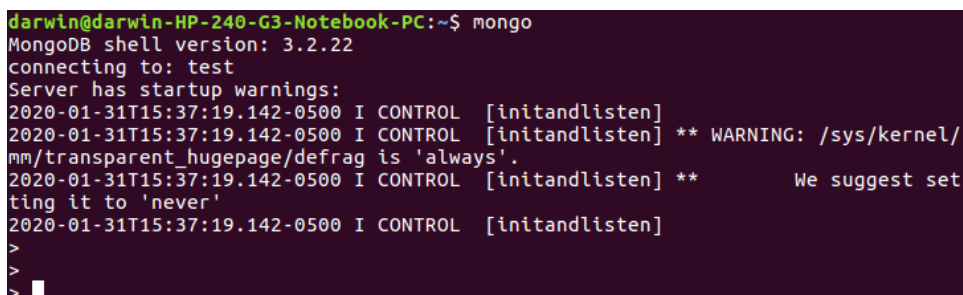
Distribución Linux: Ubuntu 16.04



The screenshot shows an Ubuntu 16.04 desktop with a red and purple background. A terminal window is open, displaying the following commands and output:

```
darwin@darwin-HP-240-G3-Notebook-PC:~$  
darwin@darwin-HP-240-G3-Notebook-PC:~$ mongo --version  
MongoDB shell version: 3.2.22  
darwin@darwin-HP-240-G3-Notebook-PC:~$  
  
root@darwin-HP-240-G3-Notebook-PC:/home/darwin# systemctl status mongob  
● mongob.service  
   Loaded: not-found (Reason: No such file or directory)  
   Active: inactive (dead)  
root@darwin-HP-240-G3-Notebook-PC:/home/darwin# systemctl status mongod  
● mongod.service - High-performance, schema-free document-oriented database  
   Loaded: loaded (/lib/systemd/system/mongod.service; enabled; vendor preset: e  
   Active: active (running) since vie 2020-01-31 15:37:07 -05; 6h ago  
     Docs: https://docs.mongodb.org/manual  
   Main PID: 1116 (mongod)  
    CGroup: /system.slice/mongod.service  
            └─1116 /usr/bin/mongod --quiet --config /etc/mongod.conf  
  
ene 31 15:37:07 darwin-HP-240-G3-Notebook-PC systemd[1]: Started High-performanc  
root@darwin-HP-240-G3-Notebook-PC:/home/darwin#  
root@darwin-HP-240-G3-Notebook-PC:/home/darwin#
```

Por medio de esto se comprueba que la instalación fue exitosa. Una vez realizado este paso importante, ingresamos a mongo para hacer sus creaciones correspondientes.



The screenshot shows a terminal window with the following output:

```
darwin@darwin-HP-240-G3-Notebook-PC:~$ mongo  
MongoDB shell version: 3.2.22  
connecting to: test  
Server has startup warnings:  
2020-01-31T15:37:19.142-0500 I CONTROL [initandlisten]  
2020-01-31T15:37:19.142-0500 I CONTROL [initandlisten] ** WARNING: /sys/kernel/  
mm/transparent_hugepage/defrag is 'always'.  
2020-01-31T15:37:19.142-0500 I CONTROL [initandlisten] ** We suggest set  
ting it to 'never'  
2020-01-31T15:37:19.142-0500 I CONTROL [initandlisten]  
>  
>  
>
```

Por siguiente dentro de mongo, comenzaremos la creación de las colecciones posibles de nuestro proyecto con la finalidad de poder gestionar sus datos de manera rápida. Aquí se crea la base de dato no relacional denominado con el nombre “GYM_DB”

```
> use GYM_DB
switched to db GYM_DB
> █
```

Se crea y se muestra de las creaciones de las colecciones correspondiente al proyecto que se usaran

```
>
> db.createCollection('clientes');█
```

```
>
> db.createCollection('users');█
```

```
>
> show collections
clientes
users
>
> █
```

Luego se insertará datos de cada colección realizada, con los campos apropiados para poder la gestión del mismo.

```
>
> db.Clientes.insert({"Id_cliente":"001","Cedula":"0897545845","Nombre":"jose","Apellido":"arreaga","Correo":"jese145@mail.com","Num_telefono":"0245787916"});█
```

Se observa la base de dato en formato json ya que se mostrará como listado todo lo que se ha ingresado

```
> db.clientes.find().pretty()
{
  "_id" : ObjectId("5e32b6bce9879c0e7a4b391a"),
  "Cedula" : "0950968973",
  "Nombre" : "mario",
  "Apellido" : "castro",
  "Correo" : "mcastroq@espol.edu.ec",
  "Num_telef" : "0981049925",
  "__v" : 0
}
{
  "_id" : ObjectId("5e32c3d8e2cf920e962e5dd6"),
  "Cedula" : "09515146389",
  "Nombre" : "jaime",
  "Apellido" : "alban",
  "Correo" : "jaime@gmail.com",
  "Num_telef" : "972384738932",
  "__v" : 0
}
```

```

}
> db.users.find().pretty()
{
  "_id" : ObjectId("5e3252bdc5c50f21bd42d875"),
  "username" : "root",
  "email" : "root@rootmail.com",
  "password" : "$2a$10$GVl9fW7G0zQpjiEhZpK4umIT7o888LVzjT4rzyH5KxlvXr9Q8N
9m",
  "date" : ISODate("2020-01-30T03:51:25.534Z"),
  "rol" : "ADMIN",
  "nombre" : "Root",
  "apellido" : "Root",
  "telefono" : "012456789",
  "cedula" : "123456789",
  "__v" : 0
}

```

Se comprueba si que hay conexión con el localhost:8080 en el navegador



Para poder visualizar la página. Se deberá en el software Visual Studio Code se ejecuta en el terminal el siguiente comando “npm run dev” donde permitirá levantar el ambiente del proyecto y que la página web funciones correctamente.

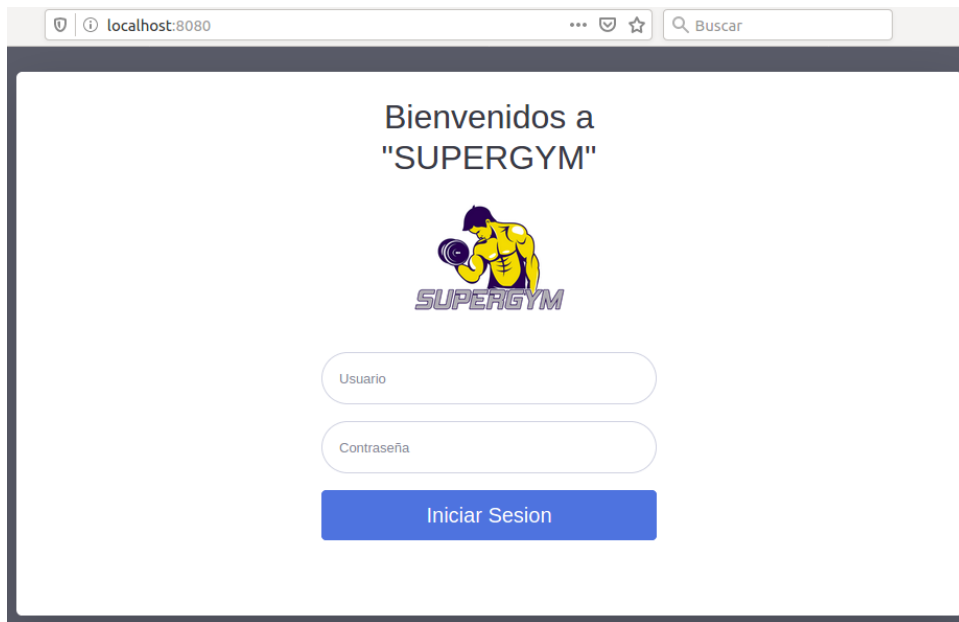
```

darwin@darwin-HP-240-G3-Notebook-PC:~/GYM_Proyec$ npm run dev
> GYM Proyec@1.0.0 dev /home/darwin/GYM_Proyec
> nodemon src/index.js

[nodemon] 2.0.2
[nodemon] to restart at any time, enter `rs`
[nodemon] watching dir(s): *.*
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node src/index.js`
Server on port 8080
Conexion satisfactoria

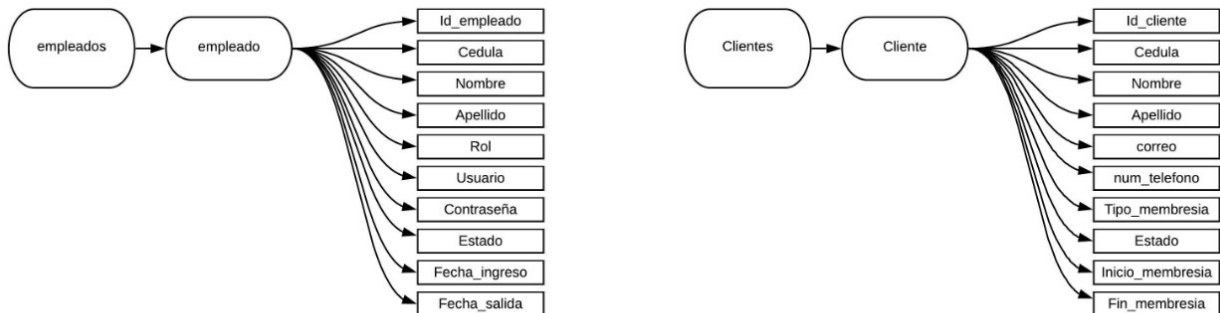
```

Ahora si podemos volver al navegador y volver a actualizarlo y nos mostrara la página principal de nuestro proyecto.



Ahí solo toca ingresar su usuario “root” y contraseña “root1234” y luego ya es llenar los campos proporcionado por la página.

Diagramas de diseño



Descripción de los campos de la base de datos no relacional

En MongoDB la información se almacena en documentos, agrupados en colecciones, la cual es un conjunto de datos que representan una determinada información de un objeto.

Los objetos para el Gimnasio son:

- Empleados
- Clientes

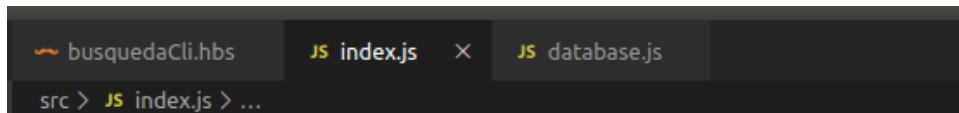
```
> db.clientes.find().pretty()
{
  "_id" : ObjectId("5e32b6bce9879c0e7a4b391a"),
  "Cedula" : "0950968973",
  "Nombre" : "mario",
  "Apellido" : "castro",
  "Correo" : "mcastroq@espol.edu.ec",
  "Num_telef" : "0981049925",
  "__v" : 0
}
{
  "_id" : ObjectId("5e32c3d8e2cf920e962e5dd6"),
  "Cedula" : "09515146389",
  "Nombre" : "jaime",
  "Apellido" : "alban",
  "Correo" : "jaime@gmail.com",
  "Num_telef" : "972384738932",
  "__v" : 0
}

> db.users.find().pretty()
{
  "_id" : ObjectId("5e3252bdc5c50f21bd42d875"),
  "username" : "root",
  "email" : "root@rootmail.com",
  "password" : "$2a$10$GVl9fw7G0zQpj0iEhzipK4umIT7o888LVzjT4rzyH5KxlvXr9Q8N9m",
  "date" : ISODate("2020-01-30T03:51:25.534Z"),
  "rol" : "ADMIN",
  "nombre" : "Root",
  "apellido" : "Root",
  "telefono" : "012456789",
  "cedula" : "123456789",
  "__v" : 0
}
```

Explicación del código fuente completo desarrollado con los comentarios correspondientes

index.js

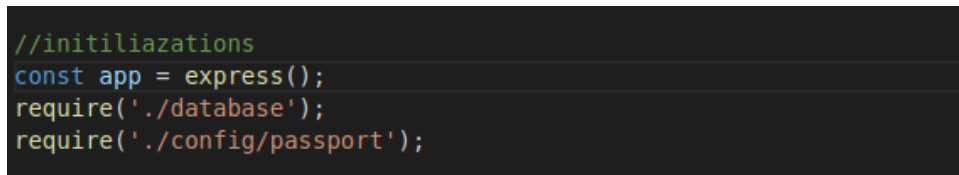
Para el sitio web lo primero fue la creación del index.js como se ve en la imagen.



El index.js contiene cada una de las librerías donde se lo denomina modulo que tiene el node.js, donde node.js fue donde se hizo posible la creación del proyecto.



Ya por otro lado está la inicialización son java script donde permite la conexión correspondiente



Ya a esto se hace la conexión con la base de dato creado anteriormente y también para ingresar a la configuración de los usuarios. Donde su respectiva configuración es en passport.js

```

onfig > JS passport.js > ...
const passport = require('passport');
const LocalStrategy = require('passport-local').Strategy;

const User = require('../models/User');

passport.use(new LocalStrategy({
  }, async (username, password, done) => {
    // Match Email's User
    const user = await User.findOne({username: username});
    if (!user) {
      return done(null, false, { message: 'Usuario no registrado.' });
    } else {
      // Match Password's User
      const match = await user.matchPassword(password);
      if (match) {
        return done(null, user);
      } else {
        return done(null, false, { message: 'Contraseña incorrecta.' });
      }
    }
  }
}));

passport.serializeUser((user, done) => {
  done(null, user.id);
});

passport.deserializeUser((id, done) => {
  User.findById(id, (err, user) => {
    done(err, user);
  });
});

```

Mediante el passport.js se hace posible ver si en verdad se ha registrado o que la contraseña está mal ingresada con la finalidad de que el usuario vea que se ha cometido un error.

Ahora para poder configurar el servicio o setear el servicio, donde se observará que está configurada en localhost:8080 para nuestra página inicio de sección en el GYM tanto el administrador como los trabajadores del mismo sistema.

```

//Setting
app.set('port', process.env.PORT || 8080);
app.set('views', path.join(__dirname, 'views'));
app.engine('.hbs', exphbs({
  defaultLayout: 'main',
  layoutsDir: path.join(app.get('views'), 'layouts'),
  partialsDir: path.join(app.get('views'), 'partials'),
  extname: '.hbs'
}));
app.set('view engine', '.hbs');

```

En la imagen anterior se observa que está configurado en el puerto 8080 con la finalidad de poder habilitarlo. Para observar que en verdad se está ejecutando en ese puerto asignado se muestra a continuación

```
darwin@darwin-HP-240-G3-Notebook-PC:~/GYM_Proyec$ npm run dev
> GYM Proyec@1.0.0 dev /home/darwin/GYM_Proyec
> nodemon src/index.js

[nodemon] 2.0.2
[nodemon] to restart at any time, enter `rs`
[nodemon] watching dir(s): *.*
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node src/index.js`
Server on port 8080
Conexion satisfactoria
```

Se nota que la conexión es satisfactoria donde se dirige al navegador y se coloca en el buscador “localhost:8080” y nos mostrara la página de inicio del GYM

Para las vistas de la pagina

Se ha utilizado los handlebars donde son cada una de las configuraciones de ciertos módulos como se observará continuación

- Actualización del cliente

```
views > partials > actualizarCliente.hbs > body.text-gray-900.bg-dark
{{> messages}}
{{> errors}}
<div class="card float-none container" style="width: 95%;">
  <div class="pt-5 pb-3 text-center">
    <h2>Actualizar datos del cliente</h2>
  </div>

  <div class="row justify-content-center">
    <div class="col-md-5 justify-content-center">
      <form action="/actualiza" method="POST">
        <input type="text" placeholder="Ingrese Identificación" class="input-control input">
        <button class="btn btn-outline-success ml-2" id="btnConsultar" type="submit">Consu
      </form>
    </div>
  </div>
  <hr style="width: 100%;">
  <form action="/clientes/actualizar" method="POST" >
    <div class="row justify-content-center">
      <div class="col-md-2">
        <label for="firstName" class="mt-2">Nombre</label>
        <input for="" class="form-control" id="firstName" name="nombre" value="{{nomb
      </div>
      <div class="col-md-2">
        <label for="lastName" class="mt-2">Apellidos</label>
        <input for="" class="form-control" id="lastName" name="apellido" value="{{ape
      </div>
      <div class="col-md-2">
        <label for="cedula" class="mt-2">Cedula</label>
        <!-- <label class="form-control" id="cedula" name="cedula" value="{{cedula}}":
        <input class="form-control" id="Cedula" name="Cedula" value="{{cedula}}>
      </div>
      <div class="col-md-3">
        <label for="email" class="mt-2">Correo</label>
        <input for="" class="form-control" id="email" name="correo" value="{{correo}}>
```


- Actualización del usuario

```
views > partials > actualizarUsuario.hbs > body.text-gray-900.bg-dark

<div class="row justify-content-center mt-3 mb-3">
  <div class="col-md-2">
    <button class="btn btn-primary" type="submit">Actualizar</button>
  </div>
  <div class="col-md-2">
    <button class="btn btn-danger" type="button">Cancelar</button>
  </div>
</div>
</form>
</div>
</body>
<script src="../../js/jquery-3.4.1.min.js"></script>
<script src="../../js/popper.min.js"></script>
<script src="../../js/bootstrap.min.js"></script>

<script>
  $('.input-number').on('input', function () {
    this.value = this.value.replace(/^[^0-9]/g, '');
  });
</script>

</html>
```

- Búsqueda de clientes

```
views > partials > 🔍 busquedaCli.hbs > 📄 body.text-gray-900.bg-dark
<body class="text-gray-900 bg-dark">
  {{> messages}}
  {{> errors}}
  <div class="card float-none container" style="width: 95%;">
    <div class="pt-5 pb-3 text-center">
      <h2>Activación de Membresías</h2>
    </div>

    <div class="row justify-content-center">
      <div class="col-md-5 justify-content-center">
        <form action="/busqueda" method="POST">
          <input type="text" placeholder="Ingrese Identificación" class="input-control input">
          <button class="btn btn-outline-success ml-2" id="btnConsultar" type="submit">Consu
        </form>
      </div>
    </div>
    <hr style="width: 100%;">
    <div class="row justify-content-center">
      <div class="col-md-3">
        <button class="btn btn-primary" type="button" onclick="btnAddMembresia()">Nueva Memb
      </div>
      <div class="col-md-3">
        <button class="btn btn-primary" type="button" onclick="btnRenovarMemb()">Renovar Mem
      </div>
    </div>
    <hr style="width: 100%;">
    <form action="/cliente/actMembresia" method="POST">
      <div class="row justify-content-center">
        <div class="col-md-2">
          <label for="firstName" class="mt-2">Nombre</label>
          <input for="" class="form-control" id="firstName" value="{{nombre}}"></input>
        </div>
        <div class="col-md-2">
          <label for="lastName" class="mt-2">Apellidos</label>
          <input for="" class="form-control" id="lastName" value="{{apellido}}"></input>
        </div>
      </div>
    </form>
  </div>
</body>
```

- Eliminación de clientes

views > partials > eliminaCliente.hbs > body.text-gray-900.bg-dark

```
body class="text-gray-900 bg-dark"
  {{> messages}}
  {{> errors}}

<div class="card float-none container" style="width: 95%;">
  <div class="pt-5 pb-3 text-center">
    <h2>Eliminar cliente</h2>
  </div>

  <div class="row justify-content-center">
    <div class="col-md-5 justify-content-center">
      <form action="/elimina" method="POST">
        <input type="text" class="form-control" value="{{nombre}}"/>
        <button class="btn btn-danger" type="submit">Eliminar</button>
      </form>
    </div>
  </div>

  <hr style="width: 100%;">
  <form action="/clientes/eliminar" method="POST">
    <div class="row justify-content-center">
      <div class="col-md-2">
        <label for="firstName" class="mt-2">Nombre</label>
        <input for="" class="form-control" id="firstName" value="{{nombre}}"/>
      </div>
      <div class="col-md-2">
        <label for="lastName" class="mt-2">Apellidos</label>
        <input for="" class="form-control" id="lastName" value="{{apellido}}"/>
      </div>
      <div class="col-md-3">
        <label for="cedula" class="mt-2">Cedula</label>
        <!-- <label class="form-control" id="cedula" name="cedula" value="{{cedula}}"/>
        <input class="form-control" id="Cedula" name="Cedula" value="{{cedula}}"/>
      </div>
      <div class="col-md-3">
        <label for="email" class="mt-2">Correo</label>
        <input for="" class="form-control" id="email" value="{{correo}}"/>
      </div>
    </div>
  </form>
```


- Eliminación de usuarios

```
views > partials > eliminaUsuario.hbs > body.text-gray-900.bg-dark
<body class="text-gray-900 bg-dark">
  {{> messages}}
  {{> errors}}

  <div class="card float-none container" style="width: 95%;">
    <div class="pt-5 pb-3 text-center">
      <h2>Eliminar usuario</h2>
    </div>

    <div class="row justify-content-center">
      <div class="col-md-5 justify-content-center">
        <form action="/usuarios/buscarE" method="POST">
          <input type="text" placeholder="Ingrese Identificación" class="input-control input">
          <button class="btn btn-outline-success ml-2" id="btnConsultar" type="submit">Cons
        </form>
      </div>

      <hr style="width: 100%;">
      <form action="/usuarios/eliminar" method="POST">
        <div class="row justify-content-center">
          <div class="col-md-2">
            <label for="firstName" class="mt-2">Nombre</label>
            <input for="" class="form-control" id="firstName" value="{{nombre}}"></input>
          </div>
          <div class="col-md-2">
            <label for="lastName" class="mt-2">Apellidos</label>
            <input for="" class="form-control" id="lastName" value="{{apellido}}"></input>
          </div>
          <div class="col-md-3">
            <label for="cedula" class="mt-2">Cedula</label>
            <!-- <label class="form-control" id="cedula" name="cedula" value="{{cedula}}">
            <input class="form-control" id="Cedula" name="Cedula" value="{{cedula}}"></input>
          </div>
          <div class="col-md-3">
            <label for="email" class="mt-2">Correo</label>
            <input for="" class="form-control" id="email" value="{{correo}}"></input>
          </div>
        </div>
      </form>
    </div>
  </div>
</body>
```

- Errores

```
views > partials > errors.hbs > ...
{{#if error }}
<div class="alert alert-danger alert-dismissible fade show" role="alert">
  {{error}}
  <button type="button" class="close" data-dismiss="alert" aria-label="Close">
    <span aria-hidden="true">&times;</span>
  </button>
</div>
{{/if}}

{{#each errors }}
<div class="alert alert-danger alert-dismissible fade show" role="alert">
  {{text}}
  <button type="button" class="close" data-dismiss="alert" aria-label="Close">
    <span aria-hidden="true">&times;</span>
  </button>
</div>
{{/each}}
```

- Home

```
views > partials > home.hbs > body#page-top.bg-dark > div#wrapper > ul#accordionSidebar.navbar-nav.bg-gradient-pr
<body id="page-top" class="bg-dark">
  <!-- Page Wrapper -->
  <div id="wrapper">

    <!-- Sidebar -->
    <ul class="navbar-nav bg-gradient-primary sidebar sidebar-dark accordion" id="accordionS

      <!-- Sidebar - Brand -->
      <a class="sidebar-brand d-flex align-items-center justify-content-center" href="/home"

        <div class="sidebar-brand-icon rotate-n-15">
          <i class="fas"></i>
        </div>
        <div class="sidebar-brand-text mx-3">SUPERGYM</div>
      </a>

      <!-- Divider -->
      <hr class="sidebar-divider my-0">

      <!-- Nav Item - Dashboard -->
      <li class="nav-item active">
        <a class="nav-link" href="/cerrarsesion">
          <i class="fas fa-fw fa-tachometer-alt"></i>
          <span>Cerrar sesion</span></a>
        </li>

      <!-- Divider -->
      <hr class="sidebar-divider">

      <!-- Heading -->
      <div class="sidebar-heading text-center">
        Controles:
      </div>

      <!-- Nav Item - Pages Collapse Menu -->
```

- Ingreso

```
views > partials > login.hbs > body
<body>
  {{> errors}}

  <div class="container">
    <!-- Outer Row -->
    <div class="row justify-content-center">
      <div class="col-xl-10 col-lg-12 col-md-9">
        <div class="card o-hidden border-0 shadow-lg my-4">
          <div class="card-body p-4">
            <!-- Nested Row within Card Body -->
            <div class="row justify-content-center ">

              <div class="col-lg-6">
                <div class="pl-5 pr-5">
                  <div class="text-center">
                    <h1 class="h2 text-gray-900 mb-4">Bienvenidos a <br>
                  </div>
                  <div class="card container border-0" style="width: 12re
                    
                  <form class="user pt-3" action="/iniciarsesion" method=
                    <div class="text-center">
                      <br>
                    </div>
                    <div class="form-group">
                      <input class="form-control form-control-user"
                    </div>
                    <div class="form-group">
                      <input type="password" name="password" required
                    </div>
                    <button class="btn btn-primary btn-lg btn-bl
                  </form>
```

- Mensajes

```
views > partials > messages.hbs > ...
{{#if success_msg}}
<div class="alert alert-success alert-dismissible fade show" role="alert">
  {{success_msg}}
  <button type="button" class="close" data-dismiss="alert" aria-label="Close">
    <span aria-hidden="true">&times;</span>
  </button>
</div>
{{/if}}
```

- Registro

```
views > partials > registro.hbs > body.bg-dark
<body class="bg-dark">
  {{> messages}}
  {{> errors}}
  <div class="card float-none container" style="width: 95%;">

    <div class="py-4 text-center">
      <h2>Registro de Clientes</h2>
    </div>

    <div class="row justify-content-center">
      <div class="col-md-8 order-md-1">
        <h4 class="mb-3">Datos personales</h4>
      </div>
    </div>
    <div class="pb-4 text-center">
      <form action="/clientes/nuevo" method="POST">
        <div class="row justify-content-center">
          <div class="col-md-4 order-md-1">
            <label for="Nombre">Nombre</label>
            <input type="text" class="form-control" name="Nombre" placeholder="" required="">
          </div>

          <div class="col-md-4 order-md-1">
            <label for="Apellido">Apellidos</label>
            <input type="text" class="form-control" name="Apellido" placeholder="" required="">
          </div>
        </div>

        <div class="row justify-content-center">
          <div class="col-md-4 mb-3 mt-3">
            <label for="Cedula">Cedula: <span class="text-muted"></span></label>
            <input type="text" class="form-control input-number" name="Cedula" placeholder="">
          </div>
        </div>
      </form>
    </div>
  </div>
</body>
```

- Registro de usuario

```
views > partials > registroDeUsuario.hbs > body.bg-dark
body class="bg-dark"
  {{> messages}}
  {{> errors}}

  <div class="card float-none container" style="width: 95%;">

    <div class="py-4 text-center">
      <h2>Registro de Usuarios</h2>
    </div>

    <div class="row justify-content-center">
      <div class="col-md-8 order-md-1">
        <h4 class="mb-3">Datos personales</h4>
      </div>
    </div>

    <div class="pb-4">
      <form action="/usuarios/registro" method="POST">
        <div class="row justify-content-center">
          <div class="col-md-3 mb-2 mt-2">
            <label for="Nombre">Nombre</label>
            <input type="text" class="form-control" name="nombre" placeholder="" required="yes">
          </div>

          <div class="col-md-3 mb-2 mt-2">
            <label for="Apellido">Apellidos</label>
            <input type="text" class="form-control" name="apellido" placeholder="" required="yes">
          </div>

          <div class="col-md-3 mb-2 mt-2">
            <label for="Num_telef">Telefono:<span class="text-muted"></span></label>
            <input type="telefono" class="form-control input-number" name="telefono" placeholder="" required="yes">
          </div>
        </div>
      </form>
    </div>
  </div>
```

Análisis de presupuesto

Para este proyecto realizado no se requiere inversión debido a que todos los softwares utilizados son de libre acceso. Pero en el caso hipotético de mente empresarial, podemos optar por ofrecer a los dueños de gimnasio una página web de rápido acceso y poder gestionar sus datos. Solo se analizaría el tiempo invertidos y los requerimientos que tan rigurosos sean de la empresa

Conclusiones

- Se concluye que el uso de sistema de gestor de base de datos no relacional no requiere servidores con gran cantidad de recursos para operar.
- Se concluye que mongoDB permite registrar un gran flujo de usuarios ya que al ser una base de datos distribuida puede escalar tanto en forma vertical (CPU Y RAM) y en forma horizontal (creando nodos).
- Usar una página web para el registro de los usuarios permite que la empresa sea más rentable y eficiente con el manejo de la información.
- Se puede concluir que la creación de la página web funciona correctamente porque se cumplió con los parámetros propuesto, siendo así que el empleado tenga la iteración con la misma página donde pueda actualizar membresía como eliminar clientes.

Referencias bibliográficas

- Sánchez, J. (2004). Principios sobre bases de datos relacionales. Informe, Creative Commons, 11-20.
- SK. (20 de agosto de 2014). *Setting Up DNS Server On CentOS 7*. Recuperado el 2 de agosto de 2019, de Unixmen: <https://www.unixmen.com/setting-dns-server-centos-7/>
- Quiroz, J. (2003). El modelo relacional de bases de datos. Boletín de Política Informática, 6, 53-61.