NOMBRES COMPLETOS: JUAN SEBASTIÁN RONDÓN LÓPEZ

CEDULA: 1005753422

CONOCIMIENTOS EN BASES DE DATOS

**1. SQL PRODUCTOS (20%):**

Tabla productos con los siguientes campos:

**id\_fabricante:** id del fabricante del producto.

**id\_producto:** Código que utiliza el fabricante para codificar el producto.

**descripcion:** nombre del producto.

**precio:** precio del producto.

**existencias:** Nº de unidades del producto que tenemos en almacén

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Id\_fabricante | Id\_producto | Descripción | precio | existencia |
| Aci | 41001 | Aguja | 58 | 227 |
| Aci | 41002 | Micropore | 80 | 150 |
| Aci | 41003 | Gasa | 112 | 80 |
| Aci | 41004 | Equipo macrogoteo | 110 | 50 |
| Bic | 41003 | Curas | 120 | 20 |
| Inc | 41089 | Canaleta | 500 | 30 |
| Qsa | Xk47 | Compresa | 150 | 200 |
| Bic | Xk47 | Compresa | 200 | 200 |

* Obtenga una lista de todos los productos utilizando sentencias SQL, indicando para cada uno su id\_fabricante, id\_producto, descripción, precio y precio con I.V.A. incluido (es el precio anterior aumentado en un 10%).
* Obtenga la cantidad total de existencias por producto.
* Obtenga el promedio de precio por fabricante.
* Obtenga el producto con mayor precio.
* Cargar en el sistema un nuevo pedido de 500 Curas que envió el fabricante Bic.
* El fabricante Osa ya no es nuestro proveedor y se deben eliminar sus productos de la BD.

R:// El archivo scripts\_productos.sql en el repositorio GIT contiene todos las sentencias del ejercicio.

**2. CONOCIMIENTOS BÁSICOSDE PROGRAMACIÓN (10%).**

1. ¿Qué es una variable?

R:// Es un espacio en memoria designado para poder guardar un valor, el cual puede cambiar durante la ejecución y ser utilizado posteriormente.

1. ¿Qué es un ciclo?, ¿mencione dos tipos de ciclos que conozca?

R:// Es una estructura de control que permite la iteración de un bloque de código o lógica de programación repetidas veces según como lo requiramos, esto varia según el tipo el tipo de ciclo que vayamos a utilizar, ya que algunos nos permiten realizar la iteración mientras que se cumpla una condición, o un número definido de repeticiones.

Dos tipos de ciclos que conozco: for y while.

1. ¿Qué es un condicional?

R:// Es una estructura de control la cual por medio de una validación lógica de verdadero o falso, nos permite en los programas ejecutar una secuencia de código u otra, según si se cumple o no una condición.

1. En el caso de un estudiante, cuando mencionamos su nombre, edad, sexo, dirección y teléfono, estamos mencionando:
   1. Los atributos de una clase
   2. Las operaciones de una clase
   3. Los objetos de una clase
   4. Ninguna de las anteriores

R:// a.

1. ¿Qué es la programación Orientada a Objetos (POO)?

R:// Es un modelo de programación que nos permite organizar el código en objetos, los cuales cada uno cuenta con atributos y métodos. Todo con la finalidad de poder hacer código más limpio y manejable.

1. ¿Que son patrones de diseño y para que se pueden utilizar?

R:// Son como esquemas o guías que nos ayudan a estructurar el código de un proyecto según las necesidades. Todo con la finalidad de poder realizar nuestro código más eficiente, mantenible y escalable.

1. ¿Cuál operador condicional de PHP evalúa que los valores son iguales y del mismo tipo de datos? (5%)
   * ==
   * ||
   * ===
   * ==?

R:// C - ===

1. Desde un punto de vista Booleano, cada valor de cero o string vacío en PHP se considera: (5%)
   * True
   * False
   * Error
   * Null

R:// B – False

1. De la siguiente estructura escriba el resultado final de la variable $c:

**$a = 1;**

**$b = 9;**

**for ( $i = $a; $i <= $b; $i = $i+1){**

**$c = $a + $i;**

**}**

**echo $c;**

**R:// $c = 10;**

1. De la siguiente estructura escriba el resultado final de la variable $c:

**$b = 1;**

**$c = 0;**

**while $b < 11 {**

**$c = $c + 1;**

**$b=$b + 1;**

**}**

**echo $c;**

**R:// Error de sintaxis por falta de paréntesis en el ciclo while.**

**En cuyo caso tuviera los paréntesis, la respuesta sería: $c = 10.**

1. De la siguiente estructura escriba el resultado final de la variable $print:

**$a=5;**

**$b=6;**

**$x=8;**

**$c=20;**

**$print=””;**

**if ($a <= $b and $b >= $c){**

**if ($x <= $c and $c >= 10){**

**$print=”Feliz Navidad”;**

**}**

**}else{**

**if ($c > $x or $x <= $b){**

**$print=”Feliz Año”;**

**}**

**}**

**echo $print;**

**R:// $print = Feliz año.**

**3. CONOCIMIENTOS DE LÓGICA DE PROGRAMACIÓN (70%).**

**Desarrolle la matriz CRUD para el listado de pacientes (se puede subir una foto del paciente / Opcional).**

**Para el desarrollo de la matriz CRUD del paciente debe de tener en cuenta las siguientes tablas y sus relaciones por llaves foráneas que considere necesaria.**

* **departamentos (id, nombre)**
* **municipios (id, departamento\_id, nombre),**
* **tipos\_documento (id, nombre)**
* **genero(id, nombre)**
* **paciente (id, tipo\_documento\_id, numero\_documento, nombre1, nombre2, apellido1, apellido2, genero\_id, departamento\_id, municipio\_id, correo)**

**Se debe tener en cuenta lo siguiente para la elaboración del proyecto:**

**Incluir todas las migraciones o scripts de creación de la base de datos necesarias para dicho proyecto.**

**Incluir los seeder o registros de datos necesarios para las siguientes tablas:**

* **Departamentos: 5 registros**
* **Municipios: 2 registros por cada departamento**
* **Tipos de documento: 2 registros**
* **Users: El registro del usuario administrador con contraseña 1234567890**
* **Paciente: 5 registros de prueba para la tabla paciente.**

**3.1. Desarrollo Backend (PHP)**

**Objetivo:** Evaluar conocimientos en la lógica de programación, estructuras de datos, patrones de diseño y manejo de bases de datos.

**Ejercicio:**

* Crear una API RESTful con PHP que permita gestionar **pacientes** (crear, leer, actualizar y eliminar).
* Implementar validación de datos (por ejemplo, validación de correo electrónico, etc.).
* Implementar autenticación mediante JWT (JSON Web Tokens) o sesiones.
* Conectar la API con una base de datos **MySQL** para almacenar los datos de los pacientes.

**Tareas específicas:**

* Crear un archivo UserController.php con los métodos correspondientes.
* Escribir pruebas unitarias utilizando PHPUnit.
* Incluir una capa de acceso a la base de datos utilizando PDO.
* Validación de entradas y protección contra inyecciones SQL.

**3.2. Desarrollo Frontend (HTML, CSS, JavaScript)**

**Objetivo:** Evaluar la capacidad para desarrollar una interfaz interactiva y funcional usando tecnologías de frontend.

**Ejercicio:**

* Crear un formulario que interactúe con la API previamente desarrollada para gestionar los pacientes (por ejemplo, un formulario para registrar nuevos pacientes).
* La interfaz debe ser intuitiva, y se deben utilizar técnicas como el manejo de eventos en JavaScript (por ejemplo, con fetch o Axios) para interactuar con la API.

**Tareas específicas:**

* Crear una interfaz de usuario que permita enviar una solicitud POST a la API para agregar un nuevo paciente.
* Mostrar una lista de pacientes obtenida mediante una solicitud GET de la API.
* Usar **CSS** o **frameworks como Bootstrap o laravel** para el diseño de la interfaz.
* Manejar errores en el frontend, como mostrar mensajes si la API devuelve un error o si los datos no son válidos.

**3.3. Integración y Funcionalidades Adicionales**

**Objetivo:** Evaluar la capacidad para integrar ambas partes (backend y frontend) y la comprensión de buenas prácticas en el desarrollo web.

**Ejercicio:**

* Completar la funcionalidad de edición y eliminación de paciente tanto en la API como en la interfaz de usuario.
* Implementar una página de inicio que muestre todos los pacientes en una tabla o lista.
* Asegurarse de que las peticiones de la API y las respuestas se manejen de manera eficiente (por ejemplo, utilizando **Ajax** para evitar recargas de página).

**Tareas específicas:**

* Crear las funcionalidades de **actualización** y **eliminación** de registros de paciente mediante formularios y botones en la interfaz.
* Mostrar respuestas dinámicas en el frontend, como "paciente creado correctamente" o "Error al eliminar paciente".

**3.4. Desafío Extra (Opcional)**

**Objetivo:** Evaluar el conocimiento en herramientas o buenas prácticas adicionales.

**Ejercicio adicional:**

* Integrar un sistema de **paginación** en la lista de pacientes.
* Implementar un sistema de **búsqueda** de paciente por nombre o correo electrónico en la interfaz.
* Añadir **validaciones del lado del cliente** (en JavaScript) para evitar el envío de datos incorrectos a la API.

**Evaluación**

A la hora de evaluar la prueba, se evaluarán los siguientes aspectos:

* **Calidad del código**: ¿Es limpio, comprensible y sigue las buenas prácticas?
* **Uso de MVC**: ¿Está organizado el código de forma modular?
* **Seguridad**: ¿Está bien gestionada la seguridad, especialmente en el manejo de formularios y datos sensibles?
* **Interactividad y diseño frontend**: ¿Es la interfaz amigable y funcional?
* **Desempeño y optimización**: ¿La aplicación funciona correctamente incluso con múltiples solicitudes?

**NOTA: Subir el proyecto a un repositorio en** [**https://github.com/**](https://github.com/) **y compartir la url del mismo.**

**NOTA: En el archivo scripts\_crud\_pacientes.sql del respositorio GIT, contiene la creación de la base de datos, para la creación de las tablas y sus respectivos registros están asociados en los archivos de migrate y seeders del API realizado en el Framework de Laravel.**

**Para inicializar el proyecto en react, ejecutar primero el comando npm install y seguido npm run dev para poner en marcha el sistema.**

**Link repositorio:** [**https://github.com/MrJu4an/PruebaTecnica\_TSinergia**](https://github.com/MrJu4an/PruebaTecnica_TSinergia)