**Taller práctico.**

**Actividad N 25.**

**MVC\_CRUD parte 4.**

**Objetivo de la actividad:** Codificar las interfaces gráficas y módulos usando PHP y el servidor apache aplicando el concepto de MVC\_CRUD

**Paso 1.**

Continuando con el ejercicio, se especifican los controladores del proyecto, este código debe ubicarse en la carpeta Controller del proyecto. Creamos producto.controller.php, que contiene una lógica similar al apartado anterior. (recuerda que es un solo codigo)

|  |
| --- |
| producto.controller.php |
| <?php  require\_once 'model/producto.php';  class ProductoController{  private $model;  public function \_\_CONSTRUCT(){  $this->model = new producto();  }  //Llamado plantilla principal  public function Index(){  require\_once 'view/header.php';  require\_once 'view/producto/producto.php';  require\_once 'view/footer.php';  } |

**Paso 2.**

Creamos el CRUD, recuerda que debes antes haber creado la tabla producto en la BD.

|  |
| --- |
| producto.controller.php (continuación código paso 1) |
| public function Crud(){  $prod = new producto();  if(isset($\_REQUEST['idProducto'])){  $prod = $this->model->Obtener($\_REQUEST['idProducto']);  }  require\_once 'view/header.php';  require\_once 'view/producto/producto-editar.php';  require\_once 'view/footer.php'; |

**Paso 3.**

Se crea la función nuevo producto de igual manera se hace el requerimiento a las vistas.

|  |
| --- |
| producto.controller.php (continuación código paso 2) |
| public function Nuevo(){  $prod = new producto();  require\_once 'view/header.php';  require\_once 'view/producto/producto-nuevo.php';  require\_once 'view/footer.php';  } |

**Paso 4.**

Se crea la funcion guardar del CRUD.

|  |
| --- |
| producto.controller.php (continuación código paso 3) |
| public function Guardar(){  $prod = new producto();  $prod->idProducto = $\_REQUEST['idProducto'];  $prod->nit = $\_REQUEST['nit'];  $prod->nomprod = $\_REQUEST['nomprod'];  $prod->precioU = $\_REQUEST['precioU'];  $prod->descrip = $\_REQUEST['descrip'];  $this->model->Registrar($prod);  header('Location: index.php?c=producto');  } |

**Paso 5.**

Se crea la función editar del CRUD.

|  |
| --- |
| producto.controller.php (continuación código paso 4) |
| public function Editar(){  $prod = new producto();  $prod->idProducto = $\_REQUEST['idProducto'];  $prod->nit = $\_REQUEST['nit'];  $prod->nomprod = $\_REQUEST['nomprod'];  $prod->precioU = $\_REQUEST['precioU'];  $prod->descrip = $\_REQUEST['descrip'];  $this->model->Actualizar($prod);  header('Location: index.php?c=producto');  } |

**Paso 6.**

Se crea la función elimiar del CRUD.

|  |
| --- |
| producto.controller.php (continuación código paso 5) |
| public function Eliminar(){  $this->model->Eliminar($\_REQUEST['idProducto']);  header('Location: index.php');  }  }//no eliminar esta llave ya que cierra el class. |

Como actividad debes:

1. Crear los elementos descritos desde el paso 1, hasta el paso 6. Has captura de pantalla según la siguiente tabla: (recuerda hacer la descripción de cada paso empleado una buena redacción)

|  |
| --- |
| Captura de pantalla paso1 |
| un controlador llamado ProductoController. se utiliza para interactuar con la tabla producto en de la base de datos.  El controlador ProductoController tiene dos métodos:  CONSTRUCT(): Este método es el constructor de la clase y se utiliza para crear un objeto de la clase producto.  Index(): Este método se utiliza para cargar la plantilla principal de la aplicación.  En el método Index(), se cargan las vistas necesarias para mostrar los datos de los productos en una tabla. |
| Captura de pantalla paso2 |
| un método llamado Crud. Este método se utiliza para cargar la vista producto-editar.php y obtener los datos del producto a editar.  En el método Crud(), se crea un objeto de la clase producto. Si se ha establecido el parámetro idProducto en la variable global $\_REQUEST, se obtienen los datos del producto a editar utilizando el método Obtener() de la clase producto. Luego, se cargan las vistas necesarias para mostrar los datos del producto en un formulario. |
| Captura de pantalla paso3 |
| un método llamado Nuevo. Este método se utiliza para cargar la vista producto-nuevo.php, que se utiliza para agregar un nuevo producto a la tabla producto en una base de datos.  En el método Nuevo(), se crea un objeto de la clase producto. Luego, se cargan las vistas necesarias para mostrar un formulario que permita al usuario ingresar los datos del nuevo producto. |
| Captura de pantalla paso4 |
| método llamado Guardar. Este método se utiliza para guardar los datos de un producto en la base de datos.  En el método Guardar(), se crea un objeto de la clase producto y se capturan los datos del formulario (vista) utilizando la variable global $\_REQUEST. Luego, se asignan los valores capturados a los atributos del objeto producto. A continuación, se registra el producto en la base de datos utilizando el método Registrar() del modelo producto. Finalmente, se redirige al usuario a la página index.php?c=producto utilizando la función header(). |
| Captura de pantalla paso5 |
| método llamado Editar().se utiliza para actualizar los datos de un producto en la tabla producto de una base de datos.  En el método Editar(), se crea un objeto de la clase producto y se capturan los datos del formulario (vista) utilizando la variable global $\_REQUEST. Luego, se asignan los valores capturados a los atributos del objeto producto. A continuación, se actualizan los datos del producto en la tabla producto utilizando el método Actualizar() del modelo producto. Finalmente, se redirige al usuario a la página index.php?c=producto utilizando la función header(). |
| Captura de pantalla paso6 |
| Por ultimo crear es un método llamado Eliminar(). se utiliza para eliminar un registro de la tabla producto en una base de datos.  En el método Eliminar(), se llama al método Eliminar() del modelo producto y se pasa el valor del parámetro idProducto obtenido de la variable global $\_REQUEST. Luego, se redirige al usuario a la página index.php utilizando la función header(). |

1. ¿para que se usa class ProductoController? Explica en un párrafo de 57 palabras.

La clase "ProductoController" se utiliza para controlar las acciones relacionadas con la gestión de productos. Este controlador maneja las solicitudes y las vistas relacionadas con los productos, como listar productos, editar productos, agregar nuevos productos, guardar cambios en la base de datos y eliminar productos. Se comunica con el modelo "producto" para acceder a los datos de la base de datos y realizar operaciones en función de las solicitudes del usuario.

1. ¿Qué es el larabel? Explica en un párrafo de 500 palabras según sea el caso.

Laravel es un marco de trabajo de código abierto para el lenguaje de programación PHP. Laravel se basa en el patrón de diseño modelo-vista-controlador (MVC) y proporciona una amplia gama de características y herramientas integradas, como validación de entrada, protección contra CSRF, prevención de inyección SQL y más. Laravel también ofrece una amplia gama de componentes y ayudantes integrados que facilitan tareas comunes, como el manejo de formularios, AJAX, autenticación, seguridad, caché y más. Además, Laravel tiene una comunidad activa y amigable que contribuye al proyecto y ofrece soporte.