**Taller práctico.**

**Actividad N 25.**

**MVC\_CRUD parte 3.**

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

**Paso 12.**

Continuando con el ejercicio, se especifican los controladores del proyecto, este código debe ubicarse en la carpeta Controller del proyecto. Creamos proveedor.controller.php el nombre debe guardarse de esta forma.

|  |
| --- |
| proveedor.controller.php |
| <?php  //Se incluye el modelo donde conectará el controlador.  require\_once "model/proveedor.php";  class ProveedorController{  private $model;  //Creación del modelo  public function \_\_CONSTRUCT(){  $this->model = new proveedor();  }  //Llamado plantilla principal  public function Index(){  require\_once 'view/header.php';  require\_once 'view/proveedor/proveedor.php';  require\_once 'view/footer.php';  }  //Llamado a la vista proveedor-editar  public function Crud( ){  $pvd = new proveedor();  //Se obtienen los datos del proveedor a editar.  if(isset($\_REQUEST['nit'])){  $pvd = $this->model->Obtener($\_REQUEST['nit']);  }  //Llamado de las vistas.  require\_once 'view/header.php';  require\_once 'view/proveedor/proveedor-editar.php';  require\_once 'view/footer.php';  } |

**Paso 13.**

Ahora se construye el llamado a la vista proveedor-nuevo

|  |
| --- |
| proveedor.controller.php (continuación código paso 12) |
| //Llamado a la vista proveedor-nuevo  public function Nuevo(){  $pvd = new proveedor();  //Llamado de las vistas.  require\_once 'view/header.php';  require\_once 'view/proveedor/proveedor-nuevo.php';  require\_once 'view/footer.php';  } |

**Paso 14.**

Este código complementa la acción solicitada anteriormente, es un método que registrar al modelo un nuevo proveedor.

|  |
| --- |
| proveedor.controller.php (continuación código paso 13) |
| //Método que registrar al modelo un nuevo proveedor.  public function Guardar(){  $pvd = new proveedor();  //Captura de los datos del formulario (vista)  $pvd->nit = $\_REQUEST['nit'];  $pvd->razonS = $\_REQUEST['razonS'];  $pvd->dir = $\_REQUEST['dir'];  $pvd->tel = $\_REQUEST['tel'];  //Registro al modelo proveedor.  $this->model->Registrar($pvd);  //header() es usado para enviar encabezados HTTP sin formato.  //”Location:” No solamente envía el encabezado al navegador, sino que  //también devuelve el código de status (302) REDIRECT al  //navegador  header('Location: index.php');  } |

**Paso 15.**

Este código complementa la acción solicitada anteriormente, es un método que modifica el modelo de un proveedor.

|  |
| --- |
| proveedor.controller.php (continuación código paso 14) |
| //Método que modifica el modelo de un proveedor.  public function Editar(){  $pvd = new proveedor();  $pvd->nit = $\_REQUEST['nit'];  $pvd->razonS = $\_REQUEST['razonS'];  $pvd->dir = $\_REQUEST['dir'];  $pvd->tel = $\_REQUEST['tel'];  $this->model->Actualizar($pvd);  header('Location: index.php');  } |

**Paso 16.**

Este código complementa la acción solicitada anteriormente, es un método que elimina la tupla proveedor con el NIT dado.

|  |
| --- |
| proveedor.controller.php (continuación código paso 15) |
| //Método que elimina la tupla proveedor con el NIT dado.  public function Eliminar(){  $this->model->Eliminar($\_REQUEST[‘nit’]);  header(‘Location: index.php’);  }  }//no borrar ya que es la llave que cierra el class. |

Como actividad debes:

1. Crear los elementos descritos desde el paso 6, hasta el paso 11. Has captura de pantalla según la siguiente tabla: (recuerda hacer la descripción de cada paso empleado una buena redacción)

|  |
| --- |
| Captura de pantalla paso12 |
|  |
| Captura de pantalla paso13 |
|  |
| Captura de pantalla paso14 |
|  |
| Captura de pantalla paso15 |
|  |
| Captura de pantalla paso16 |
|  |

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

R/ Esta clase de PHP contiene todos los métodos que tendrá proveedor.

1. ¿para que se usa header()? Explica en un párrafo de 57 palabras.

R/ Redirecciona al archivo quye el indiquemos. Ej: tenemos header(‘Location: perfil.php’); Alli nos esta redirigiendo al contenido de perfil.php

1. ¿Qué es el composer? Explica en un párrafo de 500 palabras según sea el caso. Emplea un gráfico para entender el mismo.

R/En términos simples, Composer es una herramienta que permite a los desarrolladores de PHP descargar y administrar fácilmente las bibliotecas de código PHP de terceros que necesitan para sus proyectos.

En lugar de descargar manualmente cada biblioteca y mantenerlas actualizadas, Composer automatiza el proceso de gestión de dependencias, lo que ahorra tiempo y reduce el riesgo de errores. Composer utiliza un archivo llamado "composer.json" para definir las bibliotecas que se necesitan y sus versiones específicas.