MVC: Modelo Vista Controlador

Ejercicio práctico

Nombre: EmDevApps

**ÍNDICE**

1. **VISTA**

* 1. index.php

2.**CONTROLADOR**

2.1 controlador\_index.php

2.2 Función pintaNoticias()

3. **MODELO**

* 1. model\_index.php

3.2 getNoticiasModel()

3.3 getUsuarioModel($email, $contrasena)

3.4 agregarNoticiaModel($usuario\_id, $titulo, $cuerpo, $fecha)

3.5 editarNoticiaModel($id, $titulo, $cuerpo, $fecha)

3.6 Función borrarNoticiaModel($id)

**4. FUNCIONAMIENTO DE LA APLICACIÓN**

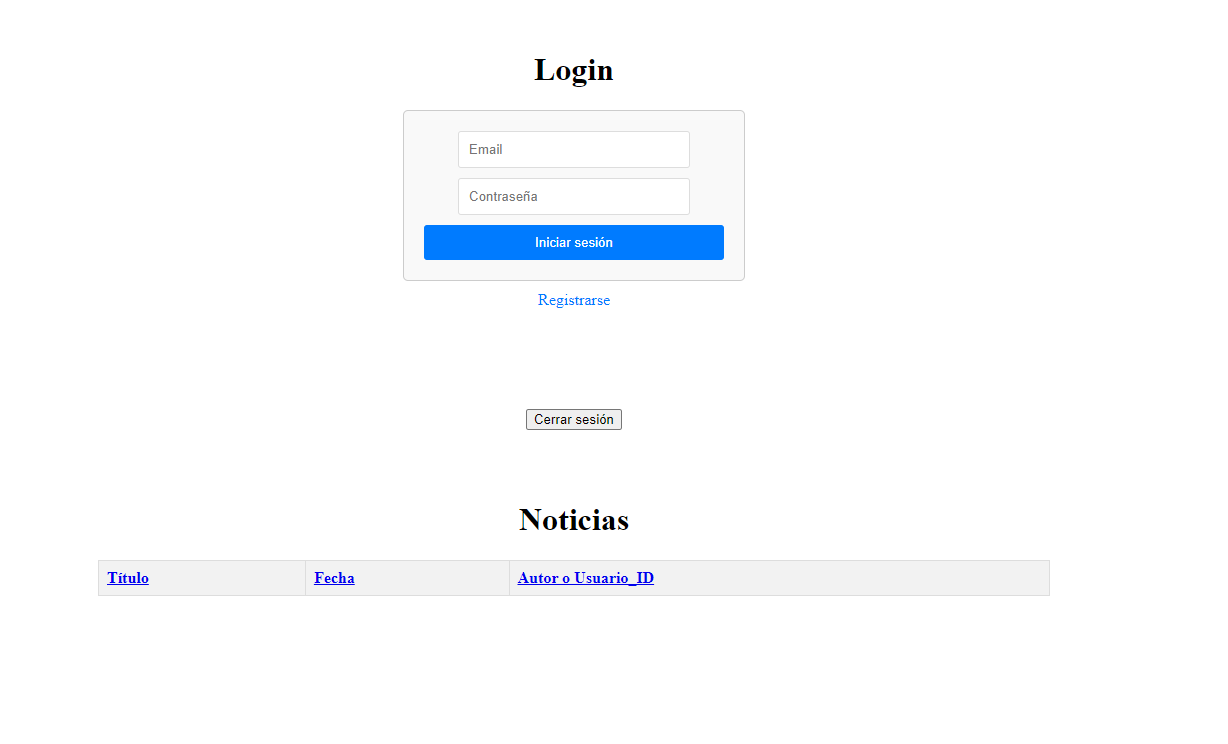
**5. CONCLUSIONES Y AUTOEVALUACIÓN**

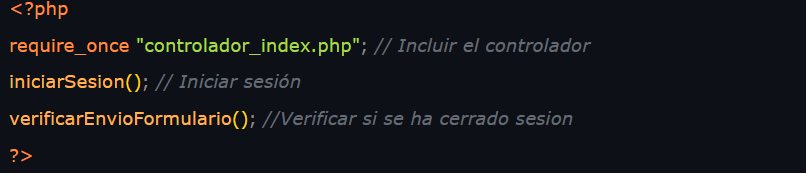
**1.VISTA**

Realmente tenemos 4 vistas en nuestra aplicación:

* vista\_ver\_noticia.php
* vista\_registro.php
* vista\_agregar\_noticia.php
* index.php **(vista estudiada en esta memoria)**

Pero debido a que el modelo vista controlador (MVC) es un concepto, lo estudiaremos a fondo desglosando el documento **index.php,** acompañado de su controlador y su modelo.

A continuación se va a mostrar la estructura de nuestra aplicación. Comenzaremos por **index.php**

El archivo index.php, es enteramente una VISTA. Que realiza un **require\_once** de su controlador. **controlador\_index.php**

El archivo **index.php,** por tratarse de una VISTA. Contendrá código html, y llamadas a funciones que se encuentran en el controlador. En este caso, **controlador\_index.php**

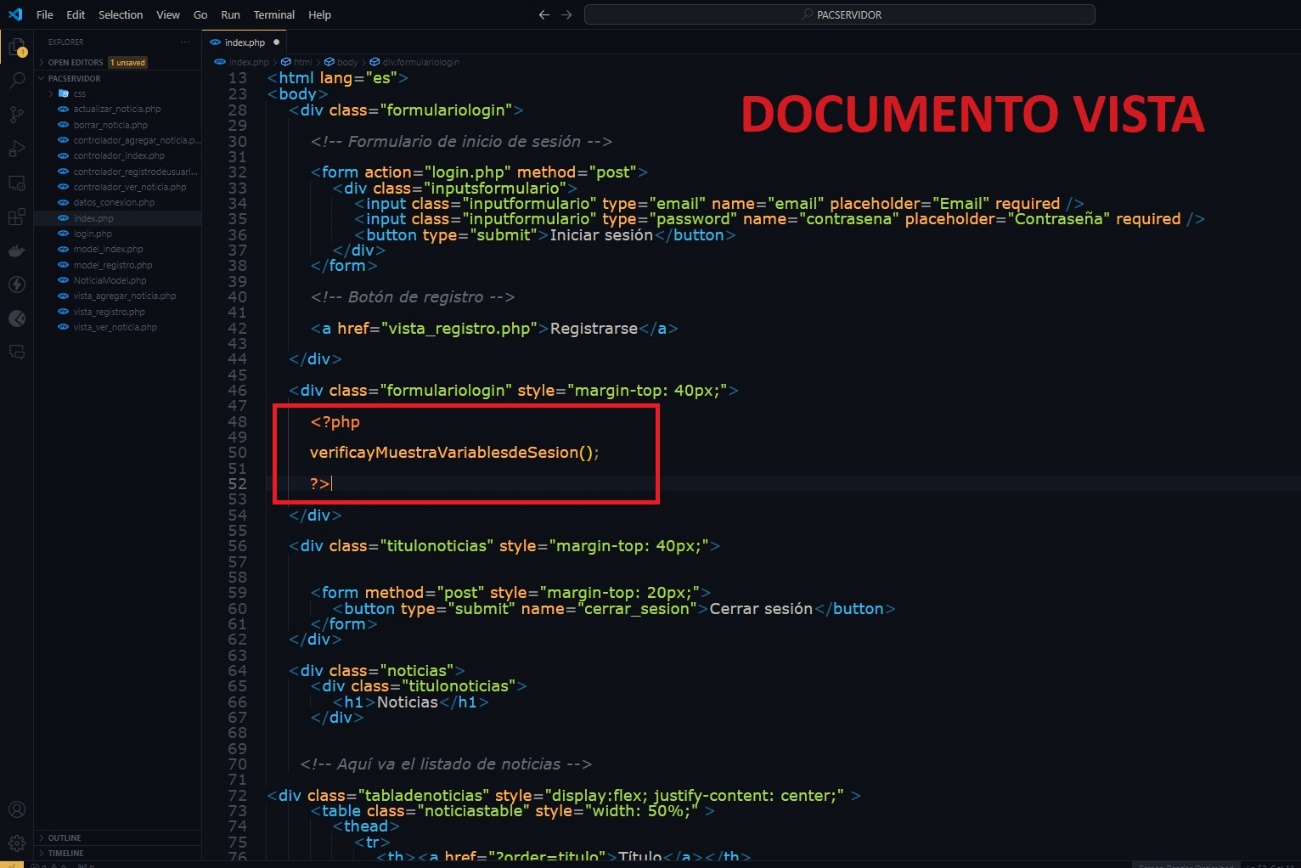
El documento **index.php**, por lo tanto, es enteramente un documento HTML, pero que requiere tener extensión **.php** porque en su interior van a existir llamadas a funciones php, que van a retornar contenido **HTML + datos integrados.**

A continuación vemos llamadas a funciones como por ejemplo:

**<?php**

**verificayMuestraVariablesdeSesion();**

**?>**

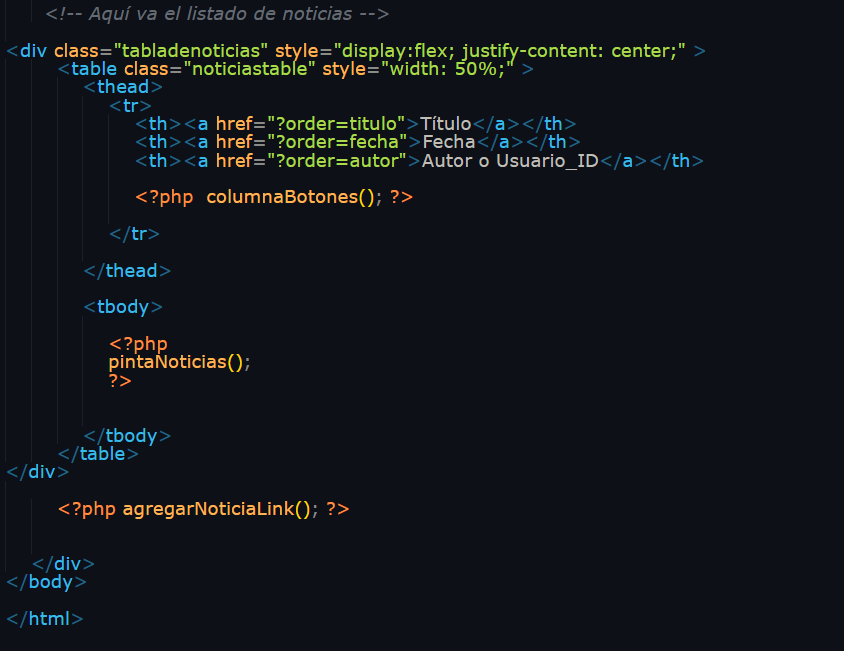


Otro ejemplo de llamada a función dentro de nuestra vista sería la siguiente:

**<?php**

**pintaNoticias();**

**?>**



2.**CONTROLADOR**

Todas estas funciones (**verificayMuestraVariablesdeSesion(); pintaNoticias(); …)**  se encuentran dentro del controlador. **controlador\_index.php**



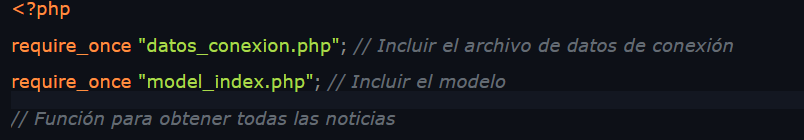
**ESTUDIO DE pintaNoticias()**

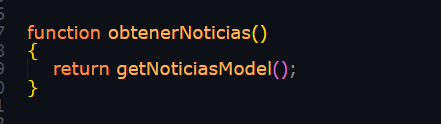
**pintaNoticias(); por ejemplo:**

Es una función que necesita primero el array asociativo de **$noticias.** El cual contiene aquello que retorne la función **obtenerNoticias().** Esta última función también se encuentra dentro del controlador, pero trabaja con una función del modelo llamada **getNoticiasModel()**.

El modelo es aquel documento que recoge datos de la base de datos, y se los brinda al controlador para que este aplique la lógica, y mapee o “pinte” la información en formato HTML. Escribiendo con **“ echo”** en el documento u otras formas que nos permita el lenguaje de programación empleado.

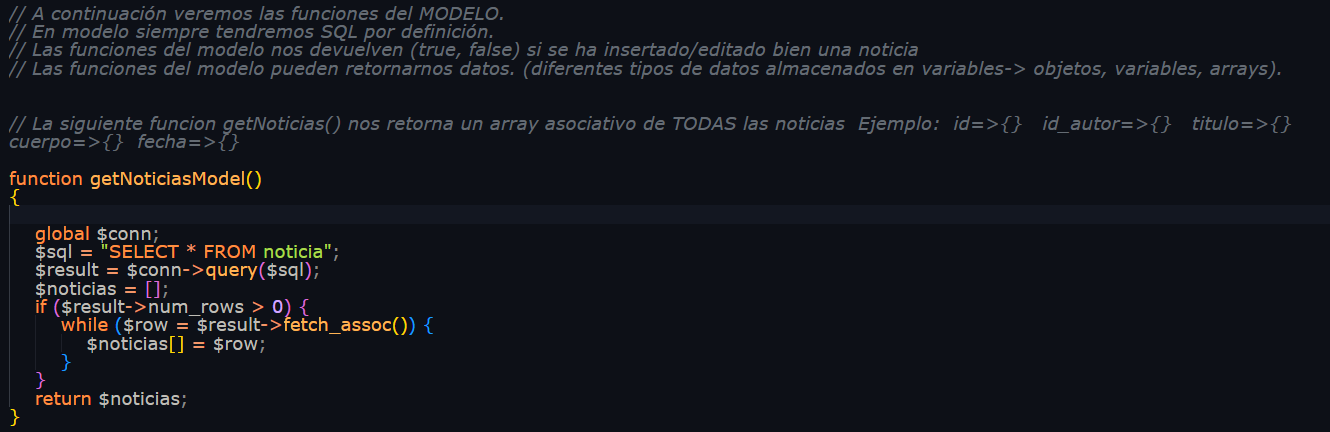
El controlador, debe de requerir o incluir, el documento modelo.





3. **MODELO**

Vemos entonces la función **getNoticiasModel()**, de nuestro documento **model\_index.php**



Y en resumen, esa es la estructura del modelo vista controlador, que se aplicó al resto de vistas.

Resto de vistas:

* vista\_ver\_noticia.php
* vista\_registro.php
* vista\_agregar\_noticia.php
* index.php **(vista estudiada en esta memoria)**

Para profundizar en las funciones del modelo, Veamos un resumen de funciones importantes del **model\_index.php.**

**Función getNoticiasModel()**

Esta función devuelve un array asociativo que contiene todas las noticias de la base de datos. Cada noticia está representada por un conjunto de campos, incluyendo ID, ID del autor, título, cuerpo y fecha.

**Función getUsuarioModel($email, $contrasena)**

Esta función recupera un usuario de la base de datos utilizando su dirección de correo electrónico y contraseña como parámetros. Retorna un array asociativo con la información del usuario, incluyendo ID, nombre, correo electrónico y contraseña.

**Función agregarNoticiaModel($usuario\_id, $titulo, $cuerpo, $fecha)**

Esta función agrega una nueva noticia a la base de datos con los parámetros proporcionados, incluyendo el ID del autor, título, cuerpo y fecha de la noticia. Retorna true si la inserción fue exitosa y false si no lo fue.

**Función editarNoticiaModel($id, $titulo, $cuerpo, $fecha)**

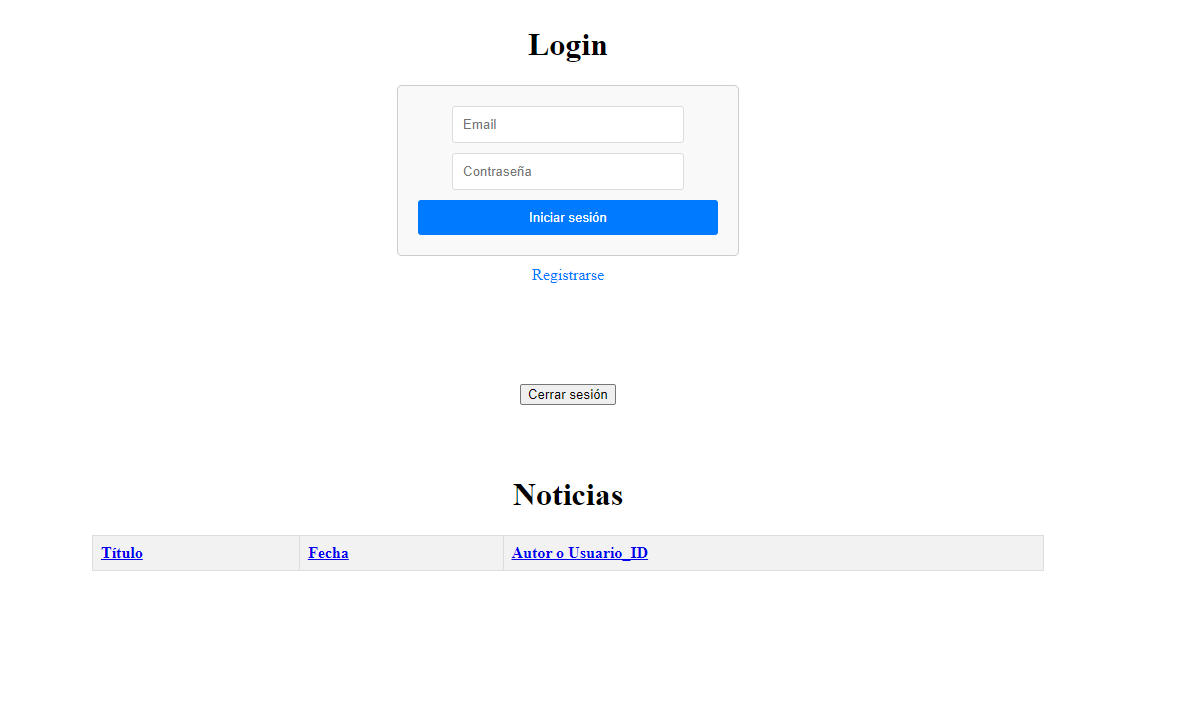
Esta función actualiza una noticia existente en la base de datos con los nuevos valores proporcionados para el título, cuerpo y fecha. Se identifica la noticia a actualizar mediante su ID. Retorna true si la actualización fue exitosa y false si no lo fue.

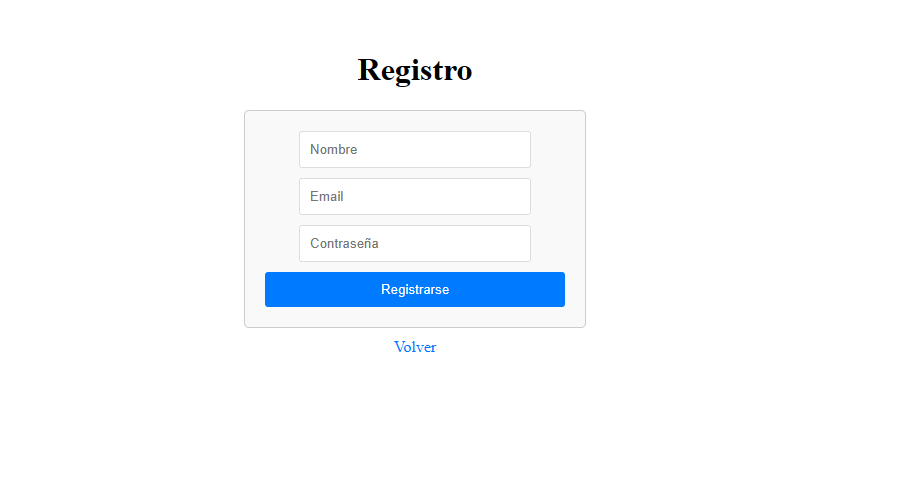
**Función borrarNoticiaModel($id)**

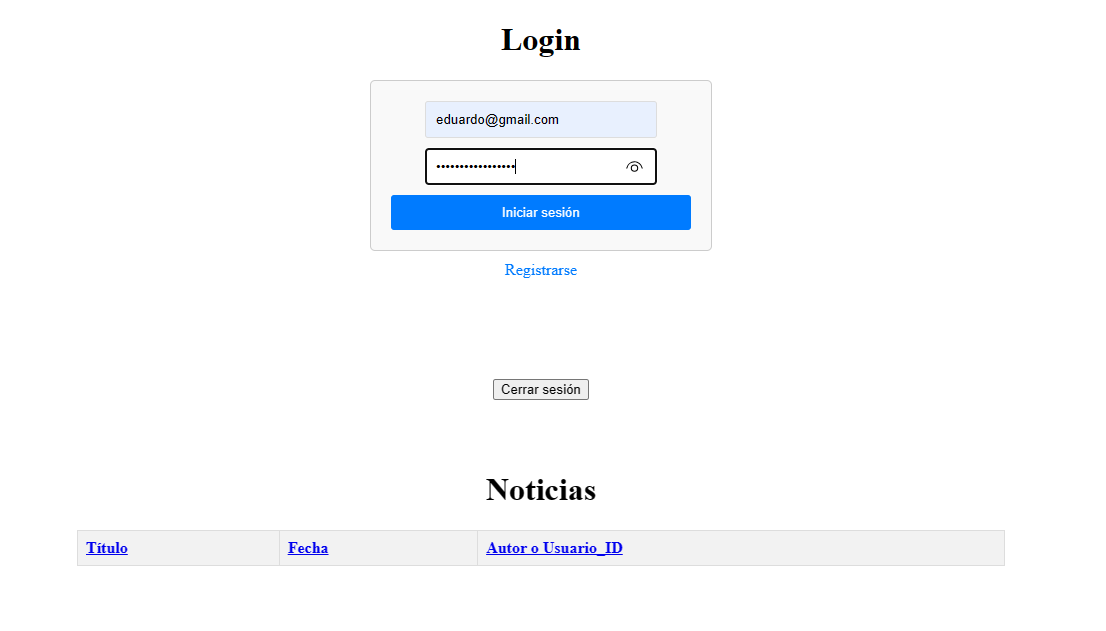
Esta función elimina una noticia de la base de datos utilizando su ID como parámetro. Retorna true si la eliminación fue exitosa y false si no lo fue.

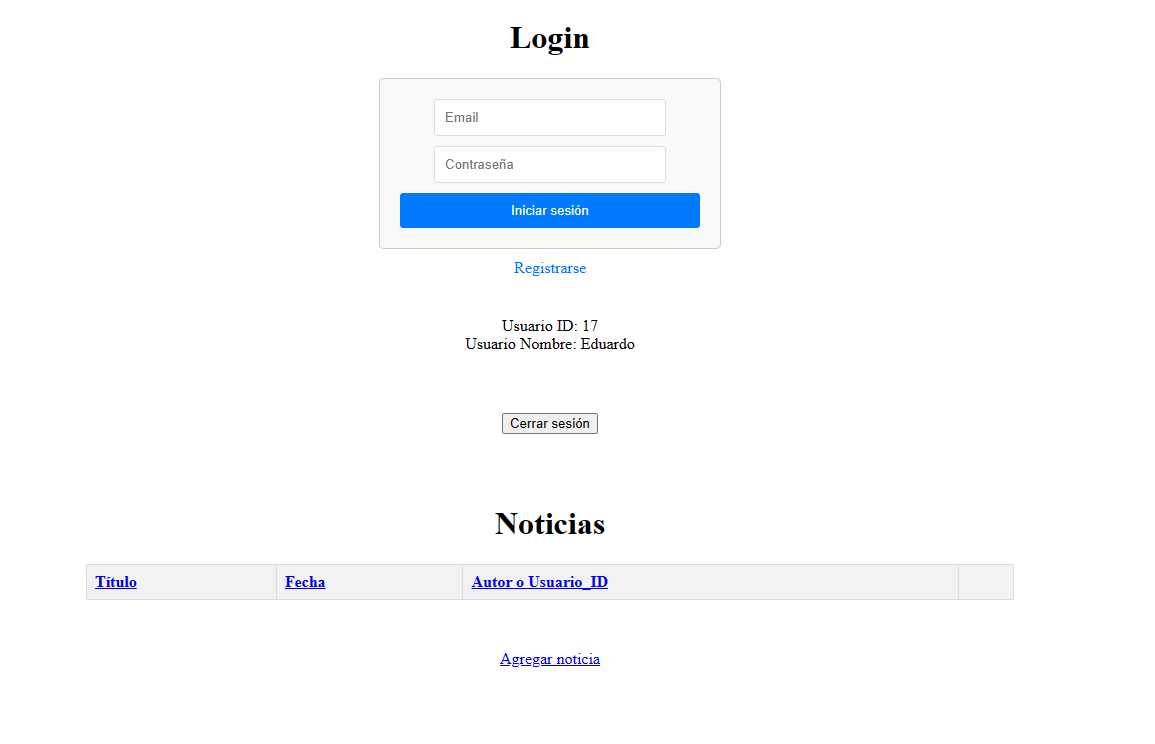
Cada función del modelo está diseñada para interactuar con la base de datos y realizar operaciones específicas relacionadas con la gestión de noticias y usuarios.

**4.FUNCIONAMIENTO DE LA APLICACIÓN**

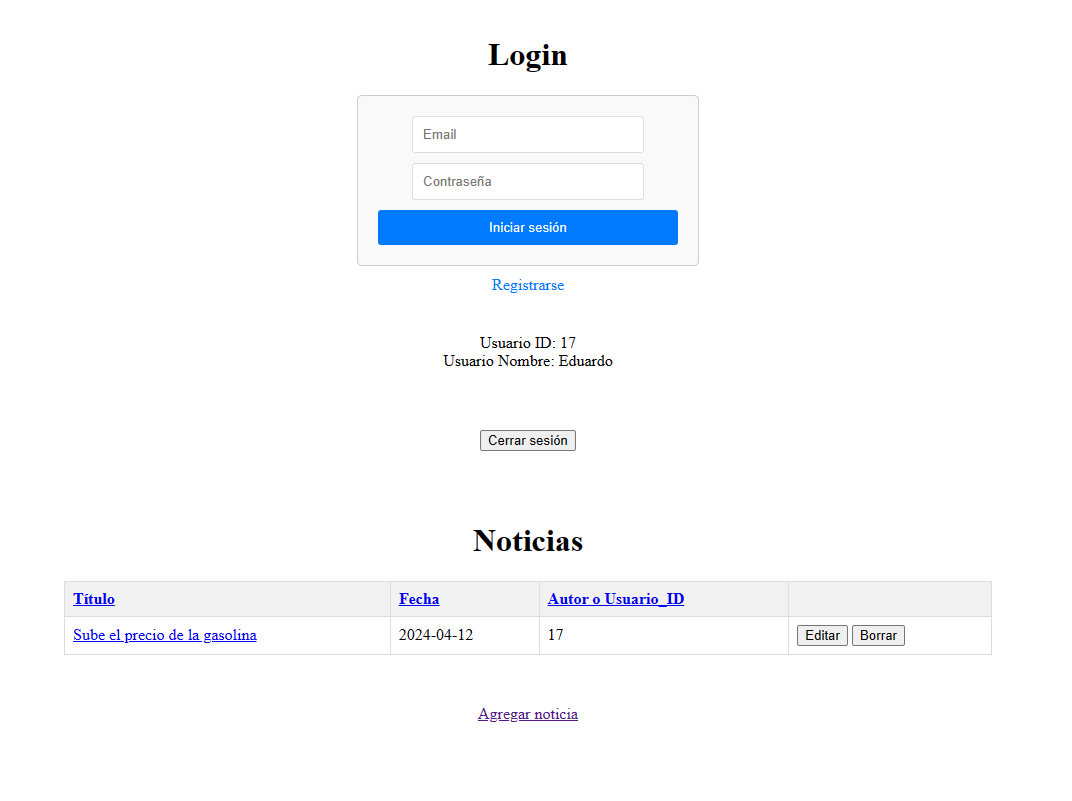




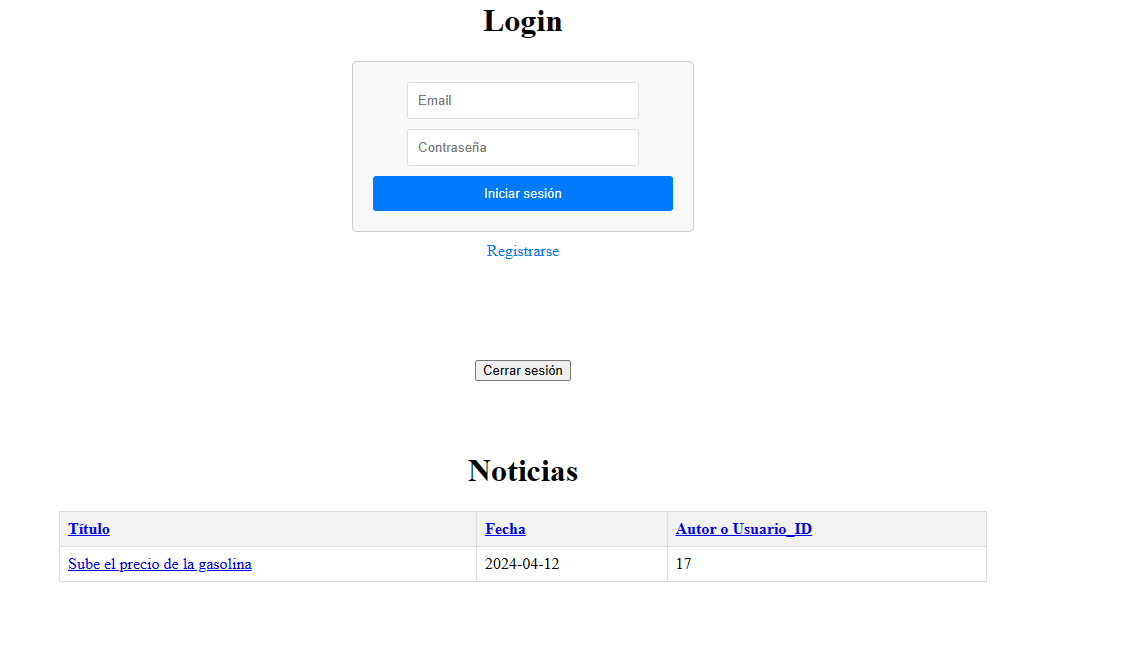


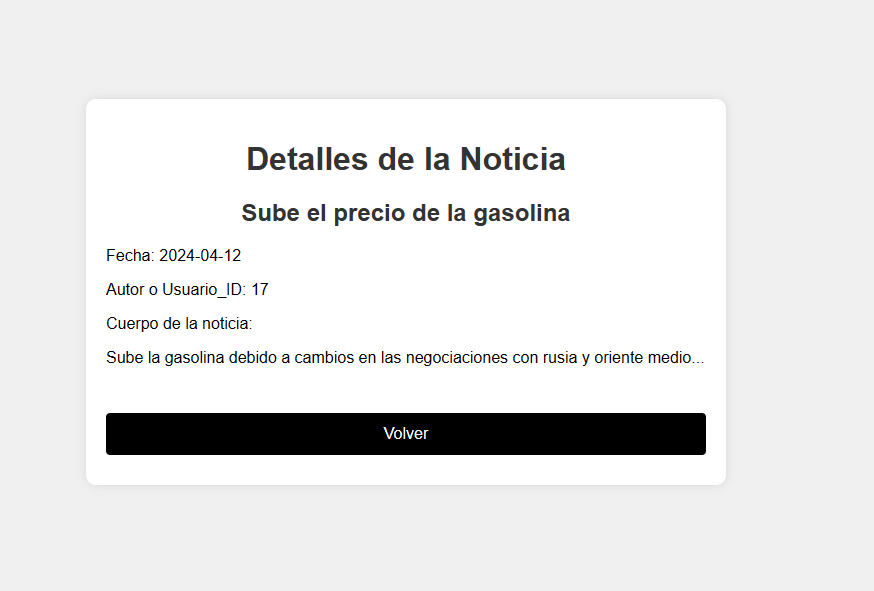






**Cerramos sesión**





**5.CONCLUSIONES**

La aplicación cumple con los requisitos especificados.