REC. PRIMER PARCIAL – PROGRAMACIÓN III – 2 cuat. 2023

Aclaración:
Las partes se corregirán de manera secuencial (ascendentemente). Si están bien todos los puntos de una parte, habilita la corrección de la parte posterior.
Se debe crear un archivo por cada entidad de PHP.
Todas las entidades deben estar dentro de un namespace (Apellido\Nombre del alumno).
Todos los métodos deben estar declarados dentro de clases. Se tendrá encuentra la aplicación de PSR-1, el tipado de atributos, parámetros y valores de retorno.
Todos los métodos deben estar declarados dentro de clases.
PDO es requerido para interactuar con la base de datos.
Se deben respetar los nombres de los archivos, de las clases, de los métodos y de los parámetros.
Todas las referencias a archivos y/o recursos, deben estar de manera relativa.

Parte 1 (hasta un 5)

Ciudadano.php. Crear, en **./clases**, la clase **Ciudadano** con atributos públicos (ciudad, email y clave), un constructor (que inicialice los atributos, la *ciudad* será opcional) y un método de instancia **toJSON():string**, que retornará los datos de la instancia (en una cadena con formato **JSON**).
Agregar:
Método de instancia **guardarEnArchivo($path)**, que agregará al ciudadano en el path recibido por parámetro. Retornará un **JSON** que contendrá: éxito(bool) y mensaje(string) indicando lo acontecido.
Método de clase **traerTodos($path)**, que retornará un array de objetos de tipo Ciudadano.
Método de clase **verificarExistencia($ciudadano, $path)**, que invocará al método *traerTodos* y retornará un **JSON** que contendrá: existe(bool) y mensaje(string).
Si el ciudadano está registrado (email y clave), retornará ***true***. Caso contrario, retornará ***false***, y el mensaje indicará que el ciudadano no está registrado.

AltaCiudadano.php: Se recibe por POST la ***ciudad***, el ***email*** y la ***clave***. Invocar al método *guardarEnArchivo*. El path será ***"./archivos/ciudadanos.json"***.

VerificarCiudadano.php: Se recibe por POST el ***email*** y la ***clave***, se verificará si coinciden con algún registro del archivo JSON. Invocar al método *verificarExistencia*. El path será ***"./archivos/ciudadanos.json"***.
Retornar un **JSON** que contendrá: éxito(bool) y mensaje(string) indicando lo acontecido.

ListadoCiudadanos.php: (GET) Se mostrará el listado de todos los ciudadanos en formato JSON. Invocar al método *traerTodos*.

Ciudad.php. Crear, en **./clases**, la clase **Ciudad** con atributos públicos (id, nombre, poblacion, pais y pathFoto), un constructor (con todos sus parámetros opcionales) y un método de instancia **toJSON():string**, que retornará los datos de la instancia (en una cadena con formato **JSON**).

Crear, en **./clases**, la interfaz ***IParte1***. Esta interfaz poseerá los métodos:

- **agregar()**: agrega, a partir de la instancia actual, un nuevo registro en la tabla **ciudades** (id, nombre, poblacion, pais, foto), de la base de datos **ciudades_bd**. Retorna **true**, si se pudo agregar, **false**, caso contrario.
- **traer()**: este método **estático**, retorna un array de objetos de tipo Ciudad, recuperados de la base de datos.
- **existe($ciudades)**: retorna **true**, si la instancia actual está en el array de objetos de tipo Ciudad que recibe como parámetro (comparar por nombre y país). Caso contrario retorna **false**.

Implementar la interfaz en la clase Ciudad.

ListadoCiudades.php: (GET) Se mostrará el listado **completo** de las ciudades (obtenidas de la base de datos) en una tabla (**HTML** con cabecera) o en formato **JSON**. Invocar al método estático *traer*.
Nota: *Si se recibe el parámetro **tabla** con el valor **mostrar**, retornará los datos en una tabla (HTML con cabecera), preparar la tabla para que muestre la imagen, si es que la tiene.*
*Si el parámetro no es pasado o no contiene el valor **mostrar**, retornará el array de objetos con formato JSON.*

AgregarCiudad.php: Se recibirán por POST los valores: ***nombre***, ***poblacion***, ***pais*** y ***foto*** para registrar una ciudad en la base de datos.
Verificar la previa existencia de la ciudad invocando al método ***existe***. Se le pasará como parámetro el array que retorna el método estático ***traer***.
Si la ciudad ya existe en la base de datos, se retornará un mensaje que indique lo acontecido.
Si la ciudad no existe, se invocará al método ***agregar***. La foto guardarla en ***"./ciudades/fotos/"***, con el nombre formado por: el **nombre** punto **pais** punto hora, minutos y segundos del alta (***Ejemplo: bernal.argentina.105905.jpg***).
Se retornará un **JSON** que contendrá: éxito(bool) y mensaje(string) indicando lo acontecido.

Parte 2 (hasta un 6)

Crear, en **./clases**, la interfaz ***IParte2***. Esta interfaz poseerá los métodos:

- **modificar():** Modifica en la base de datos el registro coincidente con la instancia actual (comparar por id). Retorna **true**, si se pudo modificar, **false**, caso contrario.
- **eliminar()**: elimina de la base de datos el registro coincidente con la instancia actual (comparar por nombre y pais). Retorna **true**, si se pudo eliminar, **false**, caso contrario.
- **guardarEnArchivo(Ciudad $ciudad)**: este método de **clase**, escribirá en el archivo de texto ***"./archivos/ciudades_borradas.txt"*** toda la información de la ciudad recibida cómo parámetro. Retorna **true**, si se pudo guardar, **false**, caso contrario.

Implementar la interfaz en la clase Ciudad.

ModificarCiudad.php: Se recibirán por POST los siguientes valores: ***ciudad_json*** (id, nombre, poblacion, pais y pathFoto, en formato de cadena JSON) y ***foto*** para modificar una ciudad en la base de datos. Invocar al método *modificar*.
Nota: *El valor del id y el de pathFoto, serán el id y el pathFoto de la ciudad 'original', mientras que el resto de los valores serán los de la ciudad a ser modificada.*

Si se pudo modificar en la base de datos, la foto 'original' se moverá al subdirectorio ***"./ciudades/modificadas/"***.
Mientras que la nueva foto, se guardará siguiendo las mismas indicaciones de la página AgregarCiudad.php
Se retornará un JSON que contendrá: éxito(bool) y mensaje(string) indicando lo acontecido.

EliminarCiudad.php: Si recibe el parámetro ***nombre*** por GET, retorna si la ciudad está en la base o no (mostrar mensaje).
Si recibe por GET (sin parámetros), se mostrarán en una tabla (HTML) la información de todas las ciudades borradas del archivo ***"./archivos/ciudades_borradas.txt"***.
Si recibe el parámetro ***ciudad_json*** (id, nombre y pais, en formato de cadena JSON) por POST, más el parámetro ***accion*** con valor "**borrar**", se deberá borrar la ciudad (invocando al método *eliminar*).
Si se pudo borrar en la base de datos, invocar al método *guardarEnArchivo*.
Si no se pasa el parámetro ***accion*** con el valor "**borrar**" o no se pasa el parámetro, se deberá enviar un mensaje alusivo.
Retornar un JSON que contendrá: éxito(bool) y mensaje(string) indicando lo acontecido.

Parte 3 (hasta un 7)

FiltrarCiudad.php: Si se recibe por POST el ***nombre***, se mostrarán en una tabla (HTML) las ciudades cuyo nombre coincidan con el pasado por parámetro.
Si se recibe por POST el ***pais***, se mostrarán en una tabla (HTML) las ciudades cuyo país coincida con el pasado por parámetro.
Si se recibe por POST el ***nombre*** y el ***pais***, se mostrarán en una tabla (HTML) las ciudades cuyo nombre y país coincidan con los pasados por parámetro.

MostrarFotosModificadas.php: Muestra todas las fotos de ***"./ciudades/modificadas/"*** (en una tabla HTML). Para ello, agregar un método estático (en Ciudad), llamado **MostrarModificadas**.

Parte 4 (hasta un 8)

En el directorio raíz del proyecto, agregar la siguiente página:
listadoCiudadesPDF.php: (GET) Generar un listado de las ciudades de la base de datos y mostrarlo con las siguientes
características:
- Encabezado (apellido y nombre del alumno a la izquierda y número de página a la derecha).
- Cuerpo (Título del listado, listado completo de las ciudades con sus respectivas fotos).
- Pie de página (fecha actual, centrada).