Recuperatorio PRIMER PARCIAL – PROGRAMACION III – 2 cuat. 2020

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. 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.

Parte 1 (hasta un 5)

Cocinero.php. Crear, en **./clases**, la clase **Cocinero** con atributos privados (especialidad, email y clave), constructor (que inicialice los atributos), un método de instancia ToJSON(), que retornará los datos de la instancia (en una cadena con formato **JSON**).
Agregar:
Método de instancia GuardarEnArchivo(), que agregará al cocinero en **./archivos**/cocinero.json. Retornará un **JSON** que contendrá: éxito(bool) y mensaje(string) indicando lo acontecido.
Método de clase TraerTodos(), que retornará un array de objetos de tipo Cocinero.
Método de clase VerificarExistencia($cocinero), que recorrerá el array (invocar a TraerTodos) y retornará un **JSON** que contendrá: existe(bool) y mensaje(string).
Si el cocinero está registrado (email y clave), retornará ***true*** y el mensaje indicará cuantos cocineros están registrados con la misma especialidad del cocinero recibido por parámetro. Caso contrario, retornará ***false***, y el/los nombres de la/las especialidades más populares (mayor cantidad de apariciones).

AltaCocinero.php: Se recibe por POST la ***especialidad***, el ***email*** y la ***clave***. Invocar al método GuardarEnArchivo.

VerificarCocinero.php: Se recibe por POST el ***email*** y la ***clave***, si coinciden con algún registro del archivo JSON (VerificarExistencia), crear una COOKIE nombrada con el email y la especialidad del cocinero, separado con un guión bajo (maru_botana@gmail.com_pastelero) que guardará la fecha actual (con horas, minutos y segundos) más el retorno del mensaje del método VerificarExistencia.
Retornar un **JSON** que contendrá: éxito(bool) y mensaje(string) indicando lo acontecido (agregar el mensaje obtenido del método de clase).

ListadoCocineros.php: (GET) Se mostrará el listado de todos los cocineros en formato JSON.

MostrarCookie.php: Se recibe por GET la ***especialidad*** y el ***email*** del cocinero y se verificará si existe una cookie con el mismo nombre, de ser así, retornará un **JSON** que contendrá: éxito(bool) y mensaje(string), dónde se mostrará el contenido de la cookie. Caso contrario, false y el mensaje indicando lo acontecido.
Nota: *Reemplazar los puntos por guiones bajos en el email (en caso de ser necesario).*

Receta.php. Crear, en **./clases**, la clase **Receta** con atributos públicos (id, nombre, ingredientes, tipo y pathFoto), constructor (con todos sus parámetros opcionales), un método de instancia ToJSON(), que retornará los datos de la instancia (en una cadena con formato JSON).

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

- **Agregar**: agrega, a partir de la instancia actual, un nuevo registro en la tabla **recetas** (id, nombre, ingredientes, tipo, foto), de la base de datos **recetas_bd**. Retorna **true**, si se pudo agregar, **false**, caso contrario.
- **Traer**: retorna un array de objetos de tipo Receta, recuperados de la base de datos.

Implementar la interface en la clase Receta.

AgregarRecetaSinFoto.php: Se recibe por POST el ***nombre***, los ***ingredientes*** y el ***tipo***. Se invocará al método Agregar.
Se retornará un **JSON** que contendrá: éxito(bool) y mensaje(string) indicando lo acontecido.

ListadoRecetas.php: (GET) Se mostrará el listado **completo** de las recetas (obtenidas de la base de datos) en una tabla (HTML con cabecera). Invocar al método Traer.
Nota*: preparar la tabla (HTML) con una columna extra para que muestre la imagen de la foto (si es que la tiene).*

Parte 2 (hasta un 6)

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

- **Existe**: retorna **true**, si la instancia actual está en el array de objetos de tipo Receta que recibe como parámetro (comparar por nombre y tipo). Caso contrario retorna **false**.
- **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.

Implementar la interface en la clase Receta.

VerificarReceta.php: Se recibe por POST el parámetro ***receta***, que será una cadena **JSON** (nombre y tipo), si coincide con algún registro de la base de datos (invocar al método Traer) retornar los datos del objeto (invocar al ToJSON). Caso contrario informar: si no coincide el nombre o el tipo o ambos.

AgregarReceta.php: Se recibirán por POST todos los valores: ***nombre***, ***ingredientes***, ***tipo*** y ***foto*** para registrar una receta en la base de datos.
Verificar la previa existencia de la receta invocando al método Existe. Se le pasará como parámetro el array que retorna el método Traer.
Si la receta ya existe en la base de datos, se retornará un mensaje que indique lo acontecido.
Si la receta no existe, se invocará al método Agregar. La imagen guardarla en "./recetas/imagenes/", con el nombre formado por el **nombre** punto **tipo** punto hora, minutos y segundos del alta (***Ejemplo: chocotorta.pasteleria.105905.jpg***).
Se retornará un **JSON** que contendrá: éxito(bool) y mensaje(string) indicando lo acontecido.

ModificarReceta.php: Se recibirán por POST los siguientes valores: ***receta_json*** (id, nombre, ingredientes, tipo y pathFoto, en formato de cadena JSON) y ***foto*** (para modificar una receta en la base de datos. Invocar al método Modificar.

Nota: El valor del id, será el id de la receta 'original', mientras que el resto de los valores serán los de la receta modificada.

Si se pudo modificar en la base de datos, la foto modificada se moverá al subdirectorio "./recetasModificadas/", con el nombre formado por el **nombre** punto **tipo** punto '***modificado***' punto hora, minutos y segundos de la modificación (***Ejemplo: puchero.bodegon.modificado.105905.jpg***).
Se retornará un **JSON** que contendrá: éxito(bool) y mensaje(string) indicando lo acontecido.

Parte 3

Crear, en **./clases**, la interface ***IParte3***. Esta interface poseerá los métodos:

- **Eliminar**: elimina de la base de datos el registro coincidente con la instancia actual (comparar por nombre y tipo). Retorna **true**, si se pudo eliminar, **false**, caso contrario.
- **GuardarEnArchivo**: escribirá en un archivo de texto (*recetas_borradas.txt*) toda la información de la receta más la nueva ubicación de la foto. La foto se moverá al subdirectorio "./recetasBorradas/", con el nombre formado por el **id** punto **nombre** punto '***borrado***' punto hora, minutos y segundos del borrado (***Ejemplo: 123.paella.borrado.105905.jpg***).

Implementar la interface en la clase Receta.

EliminarReceta.php: Si recibe un ***nombre*** por GET, retorna si la receta 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 recetas borradas y sus respectivas imagenes.
Si recibe el parámetro ***receta_json*** (id, nombre y tipo, en formato de cadena JSON) por POST, más el parámetro ***accion*** con valor "**borrar**", se deberá borrar la receta (invocando al método Eliminar).
Si se pudo borrar en la base de datos, invocar al método GuardarEnArchivo.
Retornar un JSON que contendrá: éxito(bool) y mensaje(string) indicando lo acontecido.

Parte 4

FiltrarReceta.php: Se recibe por POST el ***tipo***, se mostrarán en una tabla (HTML) las recetas cuyo tipo coincidan con el pasado por parámetro.
Si se recibe por POST el ***nombre***, se mostrarán en una tabla (HTML) las recetas cuyo nombre coincida con el pasado por parámetro.
Si se recibe por POST el ***nombre*** y el ***tipo***, se mostrarán en una tabla (HTML) las recetas cuyo nombre y tipo coincidan con los pasados por parámetro.

MostrarBorrados.php: Muestra todo lo registrado en el archivo de texto "recetas_borradas.txt". Para ello, agregar un método estático (en Receta), llamado **MostrarBorrados**.