Desarrollo de Aplicaciones I – FIUBA. Ejercicios Servicios Web

Ejercicio 1

1) Crear un sitio web e incluir el framework Fat-Free-Framework.

2) Definir la clase “User” la cual reciba id,nombre e e-mail en su constructor. Hacer getters y setters.

3) Routear la URL /users a el método “get()” de la clase ControllerUsers.

4) Dentro del método “get()”, crear 4 objetos “User” y devolverlos en formato JSON.

5) Probar el servicio web entrando con el browser a la direccion “localhost:8080/users”

Ejercicio 2

1) Agregar la clase ControllerManager al ejercicio anterior, y hacer que ControllerUsers herede ésta.

2) Escribir el método “beforeRoute()” en ControllerManager y escribir un mensaje dentro, para verificar que este está ejecutando.

Ejercicio 3

1) Tomar el ejercicio anterior y crear la clase ControllerUser (sin “s”)

2) Rutear la URL /user/@id al método “get” de esta clase.

3) Dentro del mismo crear un objeto User y devolverlo en formato JSON.

4) Probar el servicio web entrando con el browser a la direccion “localhost:8080/user/27”

Ejercicio 4

En este ejercicio se incluirá al sitio “smart home” el backend para poder obtener la lista de dispositivos que se muestra.

1) Tomar el ejercicio 12 de typescript resuelto, y crear la carpeta “ws” dentro de su raíz.

2) Incluir dentro de “ws”: Fat-Free-Framework (carpeta “lib”), la carpeta “Controllers” y el archivo index.php que hará el ruteo de los servicios web.

3) Agregar la clase “ControllerManager” y “ControllerDevices”.

4) Rutear la URL localhost:8080/ws/devices a la clase “ControllerDevices”

5) Devolver en el método “get” de la clase, una lista de dispositivos fija (en formato JSON) según los campos definidos en los ejercicios de typescript.

6) Cambiar el código de Typescript para consultar este web service.

Ejercicio 5

1) Partiendo del ejercicio 4, agregar el servicio web que reciba el request POST que realiza el sitio web al modificar un switch de un dispositivo.

2) Crear la clase ControllerDevice (sin “s” al final) y hacer que herede de ControllerManager.

3) Rutear la URL localhost:8080/ws/device a la clase ControllerDevice.

4) Se deberá agregar el método “post” en la clase “ControllerDevice”. Este método se ejecutará al recibir un request post a la URL mencionada.

5) Enviar un mensaje dentro del método “post” para debuggear el funcionamiento, incluyendo la variable global $_POST para verificar que hayan llegado los datos enviados desde el frontend.

6) Cambiar el código de Typescript para consultar este web service.

Ejercicio 6

1) Agregar la conexión a la base de datos en la clase ControllerManager y dejar disponible el atributo “db” que representa la conexión, para que pueda ser utilizado desde las clases hijo.

2) Reemplazar el código del método “get” de la clase ControllerDevices y realizar la consulta a la base de datos.

3) Devolver el mismo contenido en formato JSON.

Ejercicio 7

1) Modificar el método “post” de la clase ControllerDevice, para que el estado seteado del dispositivo se escriba en la base de datos. Tener en cuenta que:

- La variable global $_POST tendrá los campos:
  - “state” con los valores de texto “true” o “false”
  - “id” con el valor de texto “id_1”, “id_2”, etc.
- Se deberán chequear errores (por ejemplo que no vengan dichos datos o sean incorrectos)

2) El ws deberá devolver el resultado de la operación en formato JSON respetando el siguiente formato:

En caso exitoso:
```
{“response”:true,”idDevice”:”99”,”state”:”true|false”}
```

En caso de error:
```
{“response”:false,”msg”:”Mensaje de error”}
```

3) Verificar el funcionamiento chequeando los valores de estado de los dispositivos en la base con phpmyadmin.