ANEXO:
PUESTA EN MARCHA DEL SERVIDOR WEB

Las aplicaciones web desarrolladas en el lenguaje PHP necesitan 3 elementos para poder ser ejecutadas:

1) Servidor web: Apache, Internet Information Server...
2) Intérprete de PHP
3) Sistema Gestor de Bases de Datos: MySQL, Oracle, Microsoft SQL Server...

Aunque el propósito final de estas aplicaciones es ejecutarse en un servidor web remoto, conviene tener instalado un servidor local sobre el que realizar las pruebas por diversas razones:

- Poder controlar todos los parámetros de configuración del servidor.
- No exponer la aplicación al público mientras se está desarrollando.
- Poder ejecutar y probar la aplicación aunque no se disponga de conexión a Internet.

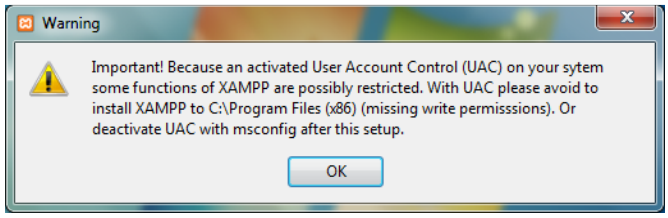
Existen herramientas de tipo Todo-En-Uno como Xampp o Wamp Server que nos facilitan el trabajo de instalación y puesta en marcha.

Nosotros vamos a utilizar Xampp que incluye Apache, PHP y MariaDB (con gestor phpMyAdmin). Tiene versiones para Windows, Linux y OS X, y al ser un proyecto con una trayectoria muy larga, cuenta con abundante documentación y numerosos complementos para ser utilizado con diferentes tipos de CMS, blogs, e-commerce, etc.

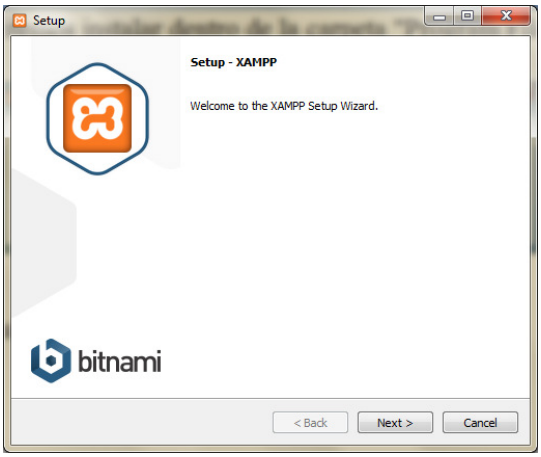
INSTALACIÓN Y PUESTA EN MARCHA DEL SERVIDOR

La instalación es tan simple como ejecutar el archivo de instalación y seguir los pasos:

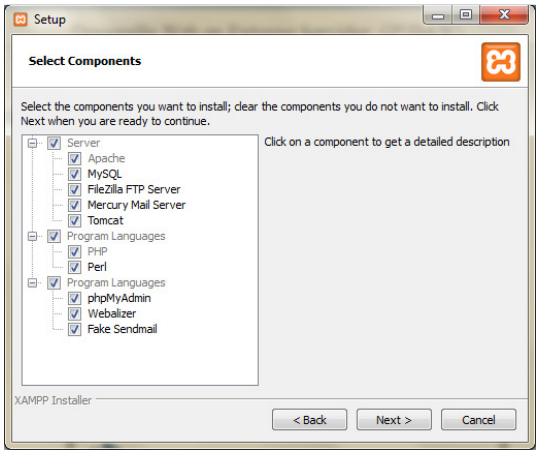
1) En primer lugar el programa de instalación comprobará si tenemos un antivirus en ejecución, en cuyo caso nos advertirá de que la instalación puede resultar muy lenta. Así que podemos optar por parar el antivirus durante la instalación.
2) A continuación si tenemos activado el sistema de Control de Cuentas de Usuario (UAC) se nos advierte que algunas de las funciones pueden estar restringidas y que en ese caso evitemos instalar dentro de la carpeta "*Program Files*".



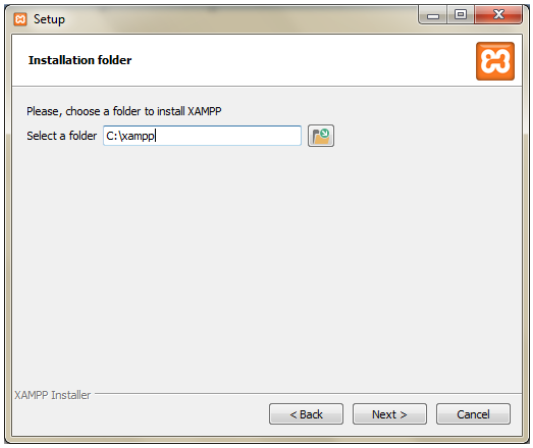
3) A partir de aquí comienza el proceso de instalación:



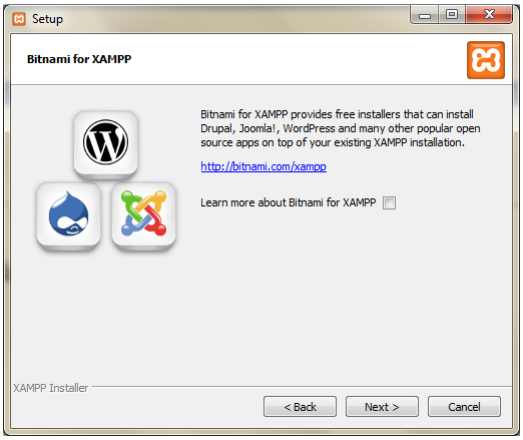
4) Seleccionamos los componentes que deseamos instalar:



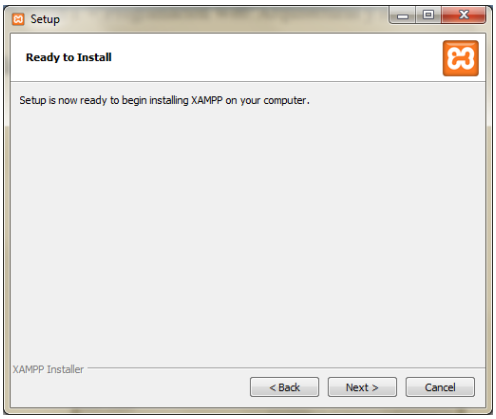
5) Y la ruta de instalación:

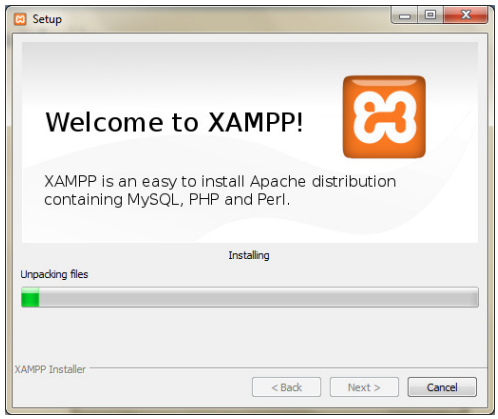


6) Por último se nos ofrece la posibilidad de acceder a la página de Bitnami para descargar complementos. Como no necesitamos ninguno desmarcamos la casilla:



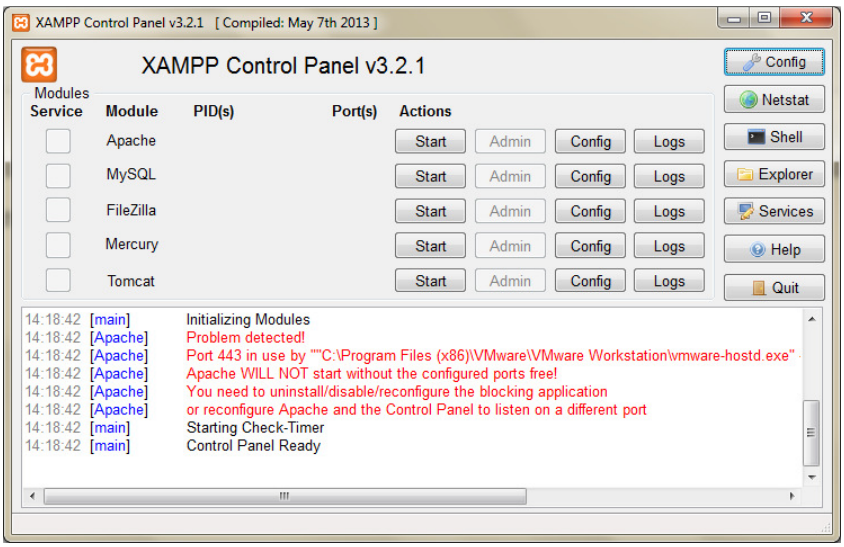
7) Comienza la instalación:



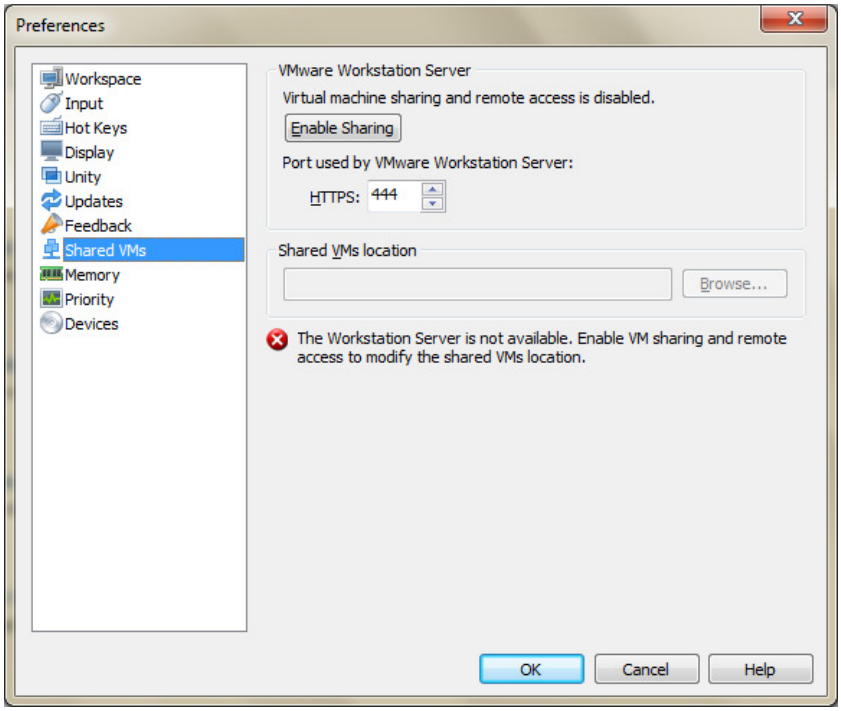


8) Una vez finalizada la instalación tendremos acceso al panel de control de Xampp desde el que podremos iniciar y detener los diferentes servicios, acceder a los archivos de configuración, ver los archivo de logs, etc.

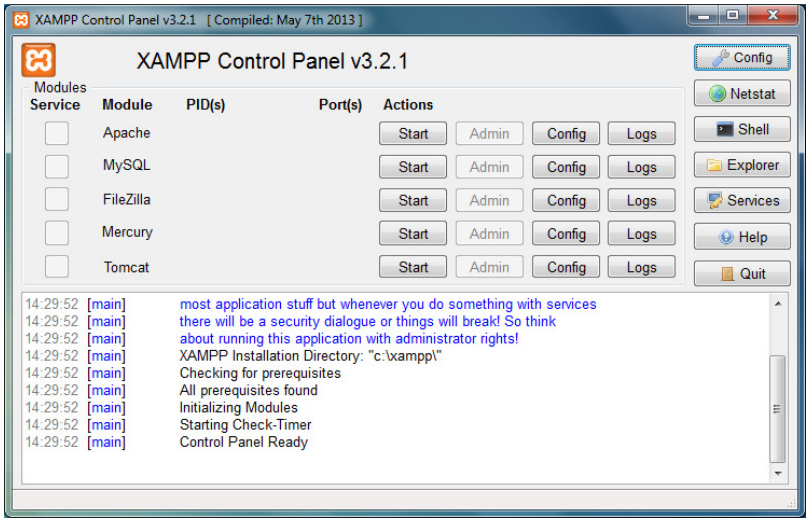
Puede ocurrir que tengamos instalada alguna aplicación que ya esté utilizando los puertos utilizados por Apache (80, 443), en cuyo caso deberemos modificar la configuración de esa aplicación o de Apache para que escuchen en puertos distintos.

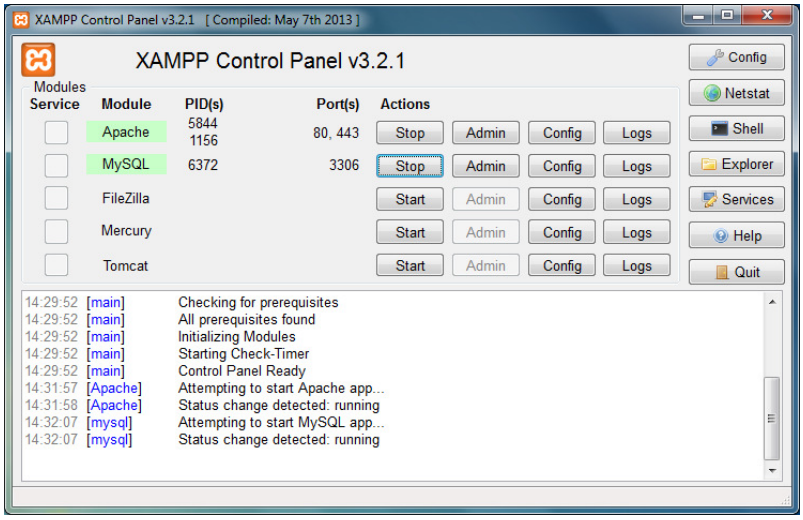


Aquí puede verse que el conflicto lo tenemos con VMware, así que accedemos a su configuración y modificamos el puerto en "*Workstation Preferences*", "*Shared VMs*".



9) Tras resolver el conflicto volvemos a abrir el panel de control e iniciamos Apache y MySQL:

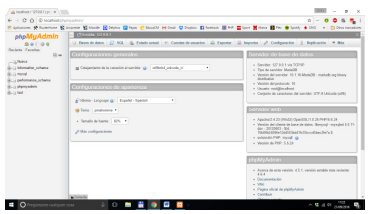




10) Ya tenemos nuestro servidor local en marcha:

http://localhost/dashboard

http://localhost/dashboard/phpinfo.php



http://localhost/phpmyadmin/

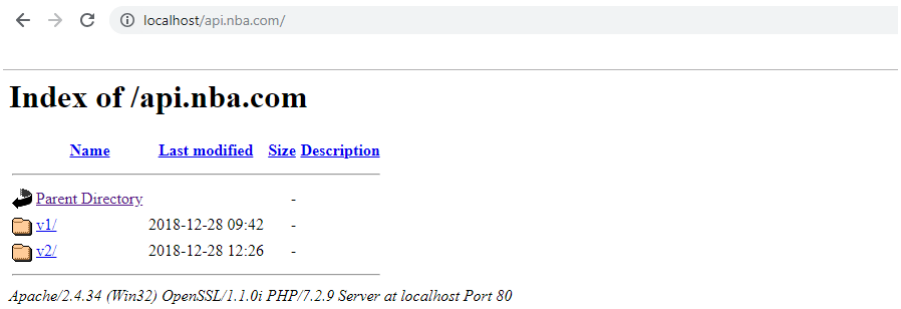
CONFIGURACIÓN DEL SERVIDOR

1) **Carpeta de ejecución**: deberemos alojar los archivos "php" que queramos ejecutar en el interior de la carpeta "C:\xampp\htdocs" que es la carpeta a la que se accede directamente a través de la ruta "http://localhost/" y dentro de ella se abrirá de forma automática el archivo denominado "index" que ahora mismo es el archivo de configuración de Xampp.

   Dentro de "htdocs", vamos a crear una carpeta para nuestras aplicaciones denominada "api.nba.com", y dentro de ella otras dos subcarpetas “v1” y “v2” para cada una de las versiones de nuestra “API”.

   El acceso a dicha carpeta será, por tanto:

   - Local: “localhost/api.nba.com”
   - Remoto: “ip/api.nba.com”



2) Configuración de Apache: aunque ya tenemos el servidor en marcha y no es necesario modificar nada puede que más adelante sea necesario modificar algún parámetro, para ello hemos de acceder al archivo "httpd.conf" que se encuentra en "c:\xampp\apache\conf". Algunos de los parámetros más importantes son:

   - ServerRoot: especifica la ubicación del directorio raíz donde se encuentra instalado el Apache.

     ```
     ServerRoot "C:/xampp/apache"
     ```

   - DocumentRoot: permite establecer la carpeta donde se alojarán los archivos que se quieren servir.

     ```
     DocumentRoot "C:/xampp/htdocs"
     ```

   - Listen: especifica que puerto se utilizará para atender las peticiones.

     ```
     Listen 80
     ```

   - ServerAdmin: indica la dirección de correo a la que se deberán reportar los errores que se produzcan en el servidor.

```
ServerAdmin postmaster@localhost
```

- ServerName: permite establecer el nombre de dominio.

```
ServerName localhost:80
```

- DirectoryIndex: indica el orden de búsqueda de archivos dentro del directorio raiz cuando no se especifica uno concreto.

```
DirectoryIndex index.php index.pl index.cgi index.asp /
  index.shtml index.html index.htm default.php /
  default.pl  default.cgi default.asp default.shtml /
  default.html default.htm home.php home.pl home.cgi /
  home.asp home.shtml home.html home.htm
```

- Timeout: define el tiempo que el servidor esperará por recibir y transmitir durante la comunicación en segundos. El valor por defecto es 300 segundos.
- KeepAlive: determina si el servidor permitirá más de una petición por conexión y se puede usar para prevenir que un cliente pueda consumir demasiados recursos del servidor. El valor por defecto es Off.

Hay que tener en cuenta que algunas de las configuraciones pueden ser sobrescritas por el archivo oculto ".htaccess" que se encuentra en la carpeta del sitio web.
Más información sobre estos archivos:

Archivo httpd.conf Archivo .httaccess

3) Configuración de PHP: se configura a través del archivo "php.ini" que se encuentra en la carpeta "c:\xampp\php". Tampoco vamos a necesitar modificar este archivo por el momento, aunque si conviene tener en cuenta algunas de sus directivas:
   - max_execution_time: establece el tiempo de ejecución máximo en segundos antes de que el analizador termine y sirve para prevenir que scripts mal escritos bloqueen el servidor. El valor por defecto es *30* salvo cuando se ejecuta PHP desde la línea de comandos en cuyo caso el valor por defecto es *0*.
   - memory_limit: establece el máximo de memoria en bytes que un script puede consumir. Ayuda a prevenir que scripts mal programados consuman toda la memoria disponible en el servidor. Para no tener límite de memoria, se ha de establecer esta directiva a *-1*.
   - upload_max_filesize: indica el tamaño máximo en bytes de los ficheros que pueden ser subidos al servidor. El tamaño máximo por defecto es "2M".

- display_errors: permite establecer si se mostrarán los errores del código en el navegador, el valor por defecto es "*off*". Puede ser recomendable modificarlo para que se muestren en el entorno de desarrollo pero nunca en el servidor que mostrará la página web al exterior ya podría ser fuente de información para posibles atacantes del sitio web.
- extension: permite añadir nuevas funcionalidades a PHP a través de módulos, para ello habría que activar la línea correspondiente y asegurarse de guardar el archivo asociado en el directorio indicado por la variable "*extensión_dir*".

Los cambios en el archivo php.ini pueden provocar que el sistema no funcione como se desea, por ello **es muy recomendable hacer una copia del archivo con su configuración inicial antes de modificarlo, y hacer los cambios de uno en uno** para probar su funcionamiento.
Cada vez que modifiquemos este archivo deberemos reiniciar el servidor Apache para que los cambios surtan efecto.

Más información sobre las directivas de php.ini:

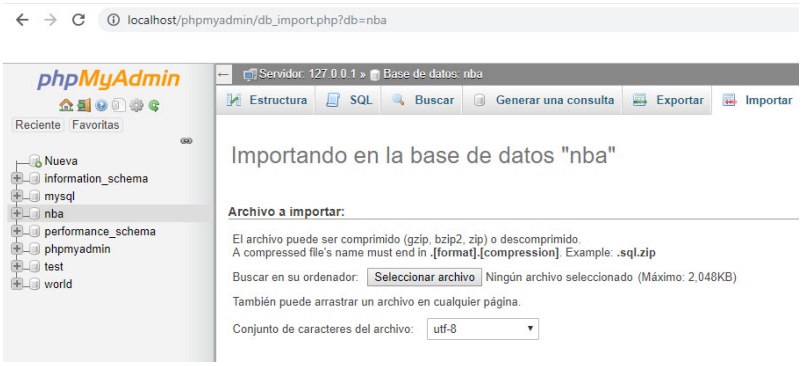
Directivas php.ini

INSTALACIÓN DE LA BASE DE DATOS

En estos ejercicios vamos a utilizar la base de datos “nba” (archivo zip suministrado) que deberemos importar desde phpmyadmin.

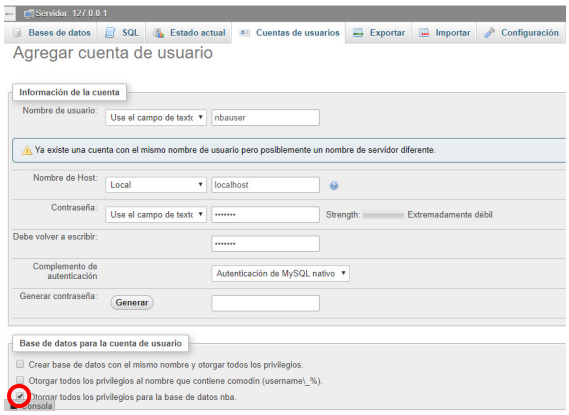
Observa que dentro del archivo zip hay varios archivos, uno contiene la base de datos completa, pero como se trata de un archivo muy grande y dependiendo del procesador de tu máquina se puede producir un *timeout* sin que finalice la importación de los datos, se suministra la misma base de datos fragmentada en varios archivos.

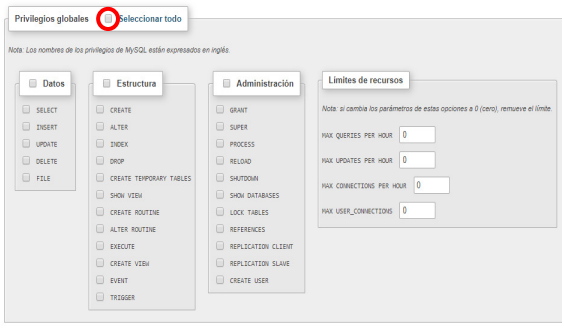
Tanto si lo importas con un único archivo, como si lo haces con los fragmentados, deberás crear en primer lugar la base de datos “nba” y tras seleccionarla importar la información de los archivos.



Tras crear la base de datos “nba” e importar los datos, crearemos un usuario con todos los permisos sobre dicha base de datos, cuyo nombre y contraseña serán “nbauser”.

Para ello iremos a la pestaña “Privilegios”, seleccionaremos “Agregar cuenta de usuario”, estableceremos su nombre y contraseña, y marcaremos la opción “otorgar todos los privilegios para la base de datos nba”, pero ninguno de los privilegios globales.





EDITAR Y EJECUTAR NUESTRO PRIMER PROGRAMA

Para la edición de código PHP se puede utilizar cualquier editor de texto, si bien es recomendable utilizar algún editor o IDE especializado como Aptana, PHPStorm, etc. ya que nos facilitarán la edición del código mediante ayudas, autocompletado, tabulaciones, etc.

Como prueba de nuestro servidor vamos a crear dentro de la carpeta "v1", un pequeño script denominado "obtenerEquipos.php" que se conecte a la base de datos y nos devuelva una lista con los nombres de todos los equipos:

```
<?php
   $host = "localhost";
   $bd = "nba";
   if (isset($_REQUEST['user'])) {
      $user = $_REQUEST['user'];
   } else {
      $user = "nbauser";
   }
   if (isset($_REQUEST['pass'])) {
      $pass = $_REQUEST['pass'];
   } else {
      $pass = "nbauser";
   }

   $con = mysqli_connect($host, $user, $pass, $bd)
      or trigger_error(mysqli_error($con), E_USER_ERROR);
   $query = "SELECT nombre FROM equipos";
   $res = mysqli_query($con, $query) or die(mysqli_error($con));
   while ($equipo = mysqli_fetch_assoc($res)) {
      echo $equipo['nombre']."/";
   }
   mysqli_close($con);
?>
```

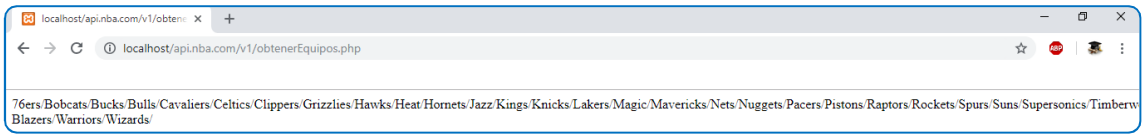
En primer lugar, se establecen los parámetros de la conexión: host y base de datos.
El usuario y su contraseña se recibirán desde el cliente que realice la petición, pero para poder realizar pruebas sin necesidad de cliente se han añadido de forma manual.

A continuación, creamos una conexión con los valores establecidos y en caso de error finalizamos la ejecución del script.

Una vez que tenemos creada la conexión establecemos y ejecutamos sobre ella la consulta, en este caso obtener el nombre de todos los equipos.
Por último, recorremos los valores obtenidos y los devolvemos al archivo de salida utilizando "/" como separador.

PRUEBAS

Podemos probar el script, y con ello comprobar que nuestro servidor de aplicaciones y el sistema gestor de bases de datos funcionan correctamente, poniendo la dirección del archivo directamente en el navegador, es decir:
http://localhost/api.nba.com/v1/obtenerEquipos.php



También podemos comprobar la conexión al script desde nuestro terminal virtual Android utilizando la aplicación de la práctica 3 del tema 4 "Conexión a internet", en la que se podía establecer una ruta de un archivo en la red y se mostraba su contenido.
En este caso a la hora de establecer la ruta, como el archivo no se encuentra dentro de la máquina virtual, deberemos sustituir "localhost" por la ip de la máquina en que se encuentra el servidor web:
http://ip/api.nba.com/v1/obtenerEquipos.php

Si todo funciona correctamente, nuestra aplicación Android solicitará el recurso "obtenerEquipos.php" al servidor web. Éste, al tratarse de un archivo php, lo enviará al servidor de aplicaciones que ejecutará el código php, se conectará al sistema gestor de base de datos, realizará la consulta y obtendrá los datos, para posteriormente generar un archivo de salida que contiene los datos solicitados.

EJECUTANDO LA APP EN ANDROID 9

Desde la versión 9 de Android (Pie) se ha restringido el tráfico HTTP de texto sin encriptar, por lo que si deseamos poder recoger ese tipo de contenido tendremos que habilitarlo en el Manifiesto, mediante la propiedad "*usesCleartextTraffic*" dentro del elemento "*Application*".

```
<application
    android:allowBackup="true"
    android:icon="@mipmap/ic_launcher"
    android:label="@string/app_name"
    android:roundIcon="@mipmap/ic_launcher_round"
    android:supportsRtl="true"
    android:theme="@style/AppTheme"
    android:usesCleartextTraffic="true">
```

Esta configuración permitirá el tráfico de texto plano para todas las comunicaciones.
Pero también podemos limitarlo sólo a los dominios en los que confiemos mediante un archivo de configuración de red que crearemos en la carpeta de recursos "res" y dentro de ella en "XML" al que podemos llamar "network_security_config.xml":

```
<?xml version="1.0" encoding="utf-8"?>
<network-security-config>
    <domain-config cleartextTrafficPermitted="true">
        <domain includeSubdomains="true">tuDominio/IP</domain>
    </domain-config>
</network-security-config>
```

Tras crear este archivo, indicando los dominios en los que se permite el tráfico de texto plano, deberemos añadir el archivo de configuración al Manifiesto, también dentro de "*application*".

```
<application
    android:allowBackup="true"
    android:icon="@mipmap/ic_launcher"
    android:label="@string/app_name"
    android:roundIcon="@mipmap/ic_launcher_round"
    android:supportsRtl="true"
    android:theme="@style/AppTheme"
    android:networkSecurityConfig="@xml/network_security_config">
```