# Instalación

## Base de datos (MySQL)

1. Crear una base de datos limpia llamada ***shop*** con cotejamiento ***utf8\_general\_ci***. Para este test de pruebas, utilizaremos el usuario **root** sin contraseña (se podría proteger creando un usuario y password específico).
2. Importar el fichero **shop.sql** en MySQL para importar las tablas, procedimientos almacenados y datos de prueba para la base de datos ***shop***.
3. La base de datos quedará constituida por:
   1. **Tablas:**
      1. ***categories*:** Para las categorías.
      2. ***products*:** Para los productos.
      3. ***products\_categories*:** Para relacionar productos y categorías.
   2. **Procedimientos almacenados:**
      1. ***GET\_CATEGORIES*:** Para obtener el listado de todas las categorías (ordenadas por nombre).
      2. ***GET\_CATEGORY*:** Para obtener el detalle de una categoría determinada.
      3. ***GET\_CATEGORY\_PRODUCTS*:** Para obtener los productos (ordenados por nombre) de una categoría determinada.
      4. ***GET\_PRODUCTS*:** Para obtener el listado de todos los productos (ordenados por nombre).
      5. ***GET\_PRODUCT*:** Para obtener el detalle de un producto determinado.
      6. ***GET\_PRODUCT\_CATEGORIES*:** Para obtener las categorías (ordenadas por nombre) de un producto determinado.
      7. ***GET\_PRODUCTS\_BY\_CATEGORY*:** Para obtener todos los productos (ordenados por nombre) de una categoría determinada.
      8. ***INSERT\_PRODUCT:*** Inserta un producto en una categoría determinada.

## Proyecto web

1. Crear en el servidor una carpeta de proyecto llamada ***shop*** e incluir el proyecto CodeIgniter.

# Web (API REST y Front)

## Framework PHP (CodeIgniter)

1. El proyecto se ha realizado con el framework PHP **CodeIgniter 3.1.6** (<https://codeigniter.com/>).
2. Se ha integrado un API Rest para CodeIgniter a través de la librería **CodeIgniter Rest Server** (<https://github.com/chriskacerguis/codeigniter-restserver>). Para este test de pruebas el API Rest no tiene autentificación (se podría proteger con una autorización básica con usuario y password).
3. Para probar las peticiones del API Rest, se puede utilizar por ejemplo la aplicación para Chrome [Advanced Rest Client](https://chrome.google.com/webstore/detail/advanced-rest-client/hgmloofddffdnphfgcellkdfbfbjeloo). El API Rest cuenta con 2 endpoints:
   1. Productos (GET):
      1. <http://[endpoint]/product/products>
      2. [http://[endpoint]/product/product\_detail/{prodcut\_id}](http://[endpoint]/product/product_detail/%7bprodcut_id%7d)
      3. [http://[endpoint]/product/products\_by\_category/{category\_id}](http://[endpoint]/product/products_by_category/%7bcategory_id%7d)
   2. Categorías (GET):
      1. <http://[endpoint]/category/categories>
      2. [http://[endpoint]/category/category\_detail/{category\_id}](http://[endpoint]/category/category_detail/%7bcategory_id%7d)
4. Configuración:
   1. En el fichero **application/config/**[**database.ph**](http://database.ph/)**p** ya están configurados los datos de acceso a la base de datos ***shop*** (servidor **localhost**, usuario **root** y **sin contraseña**).
   2. En el fichero **application/config/rest.php** ya están configurados los datos de acceso al API REST (sin autentificación, aquí es donde se pude configurar algún tipo de autentificación como por ejemplo autentificación básica).
   3. En el fichero **application/config/routes.php** ya están configurados las rutas personalizadas y reglas de enrutado.
5. Modelos:
   1. Los modelos se encuentran dentro del directorio **application/models**.
6. Vistas:
   1. Las vistas se encuentran dentro del directorio **application/views**.
7. Controladores:
   1. Los controladores se encuentran dentro del directorio **application/controllers**.

## Bootstrap

Se ha utilizado el framework [Bootstrap v4.0.0-beta.2](https://getbootstrap.com/docs/4.0/getting-started/download/) para hacer el diseño adaptado a cualquier pantalla de la web de productos.

## JQuery y JavaScript

Se ha utilizado la librería JQuery y JavaScript para realizar llamadas asíncronas tanto para la obtención de productos como para visualizar el detalle de un producto. Las funciones generadas se pueden consultar en el fichero **assets/js/shop.js**.

## CSS

Se ha creado una pequeña hoja de estilo para la web en el fichero **assets/css/shop.less**. Se ha utilizado preprocesador [Less 2.7.2](http://lesscss.org/) ya que es el mismo que utiliza Bootstrap.

## Generador de datos de prueba

Se han incluido 10 productos de Decathlon como ejemplo en la base de datos. Para generar más productos aleatorios se ha desarrollado el siguiente método (GET) para generar X productos aleatorios:

[http://[endpoint]/product/generate\_fake\_products/{quantity}](http://[endpoint]/product/generate_fake_products/%7bquantity%7d)

Cada uno de estos nuevos productos, es asignado de forma aleatoria a una o dos categorías de las ya predefinidas en la base de datos.

## Gestor de tareas

No se ha utilizado ningún gestor de tareas.

## Testing

No se ha utilizado ningún framework de testing.