

Tecnologías de almacenamiento de datos: Actividad 1

X

BASES DE DATOS DOCUMENTALES:



mongoDB®

UNIVERSIDAD EUROPEA MIGUEL DE CERVANTES

empezar_

CONTENIDOS



1. DESCRIPCIÓN DEL
CASO



4. CONSULTA DE
DOCUMENTOS



7. ELIMINAR COLECCIÓN



2. CREACIÓN DE LA
BASE DE DATOS



5. CREACIÓN DE ÍNDICES



8. ELIMINAR LA BASE DE
DATOS



3. INSERCIÓN DE
DOCUMENTOS EN COLECCIONES



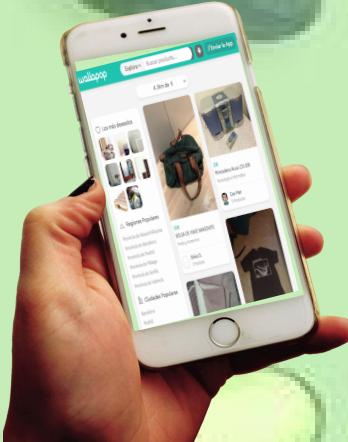
6. MODIFICACIÓN DE
DOCUMENTOS



9. REFERENCIAS



1. DESCRIPCIÓN DEL CASO

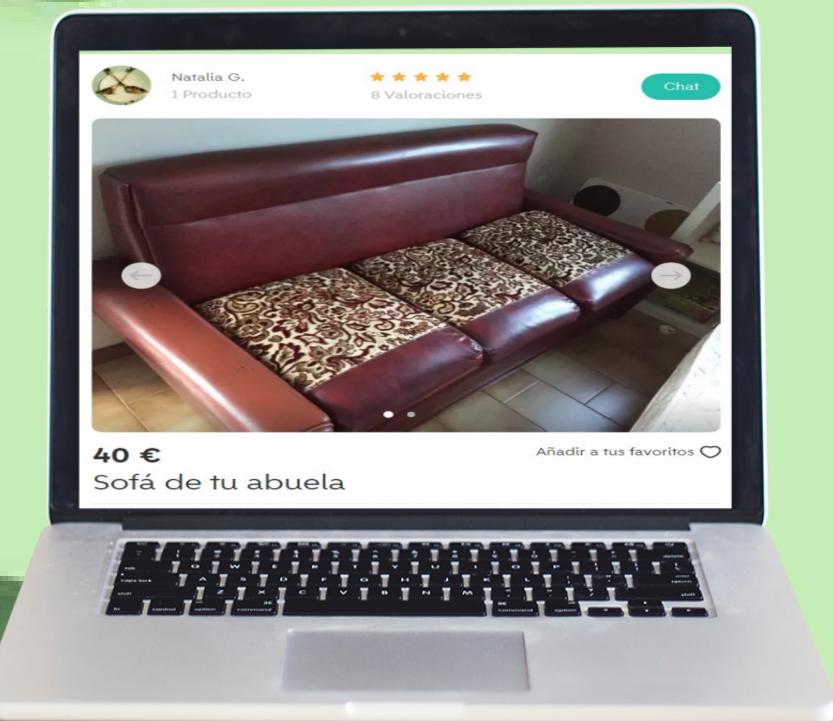


Una pequeña empresa de emprendedores ha concebido una aplicación para dispositivos móviles que permita la compra-venta de cualquier tipo de artículo entre particulares.

Para que un usuario pueda hacer uso de la aplicación debe registrarse con un nombre de usuario, una contraseña, aportar una dirección de correo electrónico y la dirección de un punto de venta preferente donde entregar los artículos.



1. DESCRIPCIÓN DEL CASO



Para poner un ítem a la venta el vendedor debe indicar una breve descripción del artículo, el precio de salida y la lista de etiquetas que lo describen. La aplicación incluirá también la fecha en la que el artículo sale a la venta.

A fin de que los usuarios de la aplicación puedan buscar artículos que se vendan cerca de su zona, la aplicación también guarda la longitud y latitud del punto de venta introducido por el vendedor del objeto. La dirección definitiva de entrega puede variar, ya que comprador y vendedor se pueden poner de acuerdo mediante correo electrónico para acordar el lugar que mejor les venga a ambos.

1. DESCRIPCIÓN DEL CASO



Los compradores interesados en un determinado artículo pueden hacer contraofertas, indicando su precio. La aplicación debe también guardar de cada contraoferta cuándo se realizó y quién la realizó.

A efectos de visualización de resultados de las búsquedas, es importante reflejar el estado de un artículo: disponible, vendido o descatalogado. Pasan a este último estado aquellos artículos que hayan sido puestos a la venta hace más de 12 meses y no hayan sido vendidos.



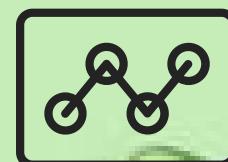
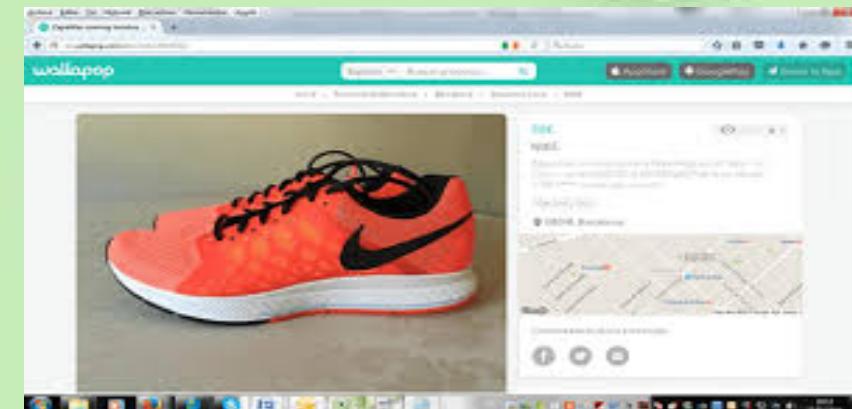
1. ESTRUCTURAS DEL AGREGADO USUARIO E ITEM:



Alex

★★★★★ (1)

The user profile card displays the name "Alex" in a large, bold, black font. Below it is a 5-star rating icon followed by the number "(1)". To the left of the card is a white arrow pointing left. Above the card is a red-bordered icon of two people, and to its right is a scissors icon. Below the card is a circular profile picture showing a blue sky and a green landscape.



2. CREACIÓN DE LA BASE DE DATOS:



Arrancar el servicio de MongoDB:

Abrimos un símbolo de sistema y escribimos mongod.

Seguidamente, para iniciar el shell de mongo, abrimos otro símbolo de sistema y escribimos mongo .

Comprobar la correcta instalación:

Revisamos en el símbolo de sistema donde escribimos mongod que aparece el puerto correcto (27017) y la dirección IP correcta.

Parar el servicio de MongoDB:

En el símbolo de sistema donde escribimos mongod ahora usamos los comandos Control+C.

Conexión de MongoDB a través del shell:

Abrimos un nuevo símbolo de sistema para escribir mongo.

Visualizar las bases de datos existentes:

show dbs

Consultar la base de datos actual:

db

3. INSERCIÓN DE DOCUMENTOS EN COLECCIONES:

Crear la base de datos actual:

Llamaremos a la base de datos compraventa, y tendremos dos colecciones: usuarios y articulos.

Para ello escribimos `use compraventa`. Como la base de datos no tiene ninguna colección, esta no aparecerá en el registro de base de datos.

Creamos la siguiente colección: `db.createCollection("usuarios")`

Finalmente, al mostrar las bases de datos con `show dbs` observaremos que aparece la nueva base de datos creada.

Para indicar la base de datos sobre la que vamos a trabajar:

Basta con teclear `db`.

3. INSERCIÓN DE DOCUMENTOS EN COLECCIONES:



Existen dos formas de añadir documentos a una colección:

1º Crear una variable con el documento y posteriormente añadirla a la colección con la función insert.

use compraventa

definimos la variable usuarios , siendo un array de documentos:

```
var usuario=[{documento 1},{documento 2},{documento 3},...];
```

```
db.createCollection("usuarios")
```

```
db
```

```
db.usuarios.insert(usuario)
```

2º Insertar directamente el documento en la función insert

```
db.usuarios.insert({documento})
```

De forma análoga definimos la colección articulos e insertamos los productos.

4 . CONSULTA DE DOCUMENTOS



Mostrar todos los documentos de una colección:

`db.usuarios.find().pretty()` (Si usamos pretty() aparecerán en el shell los documentos en formato JSON)

`db.articulos.find().pretty()`

Búsqueda de Arrays:

-**Mostrar la descripción de todos los artículos que estén etiquetados como “consolas”.**

`db.articulos.find({"etiquetas": "consolas"}, {"descripcion": 1, "_id": 0})`

-**Mostrar la descripción y precio de los artículos que estén etiquetados como “consolas” y “wii”.**

`db.articulos.find({"etiquetas": "consolas", "etiquetas": "wii"}, {"descripcion": 1, "_id": 0, "precio_inicial": 1})`

-**Mostrar la descripción y precio de los artículos que estén etiquetados como “consolas” o “wii”.**

`db.articulos.find({"etiquetas": {$in: ["consolas", "wii"]}}, {"descripcion": 1, "_id": 0, "precio_inicial": 1})`

4 . CONSULTA DE DOCUMENTOS



Búsqueda de subagregados:

- Mostrar la descripción de todos los elementos etiquetados como “consolas” que tengan contraofertas realizadas con posterioridad al 1 de enero de 2014.

```
db.articulos.find({"etiquetas":"consolas","contraoferta.fecha_oferta":{$gt: new Date("2014-01-01")}}, {"descripcion":1,"_id":0})
```

Ordenación de resultados:

- Mostrar todos los documentos de la colección items ordenados de manera que se muestren primero los que tienen contraofertas más recientes:

```
db.articulos.find().sort({"contraoferta.fecha_oferta":-1})
```

- Mostrar los usuarios en orden alfabético por su ciudad de residencia

```
db.usuarios.find({{},"nombre":1,"_id":0}).sort({"ciudad":1})
```

Agregación

Existe un framework de agregación que permite resumir los resultados de una búsqueda.

-Mostrar los emails de los usuarios que tienen algún artículo a la venta (sin mostrar repetidos):

```
db.articulos.aggregate([{$match:{estado:"en venta"}},{$lookup:  
{from:"usuarios",localField:"venta.vendedor",foreignField:"_id",as:"seller"}},{$unwind:"$seller"},  
{$group:{'_id':'$seller.e_mail'}}])
```

Mostrar cuántos usuarios viven en Valencia
(equivalente a db.usuario.find({"direccion.ciudad":"Valencia"}).count())

```
db.usuarios.find({"ciudad":"Valencia"}).count()
```

Mostrar los artículos que tiene o tuvo a la venta cada usuario

```
db.articulos.aggregate([{$lookup:  
{from:"usuarios",localField:"nombre",foreignField:"producto",as:"en_venta"}},{$project:  
{"venta.vendedor":1,"producto":1,"_id":0}}])
```

5. CREACIÓN DE ÍNDICES



Crear un índice sobre el campo descripción

```
db.articulos.createIndex({descripcion:1},{name:"indice_descripcion"})
```

Creación de un índice sobre el campo email del subagregado vendedor

```
db.articulos.createIndex({"venta.vendedor.e_mail":1},{name:"indice_email"})
```

Creación de índice sobre el subagregado vendedor:

```
db.articulos.createIndex({"venta.vendedor":1},{name: "indice_vendedor"})
```

Creación de un índice compuesto formado por los campos descripción y estado:

```
db.articulos.createIndex({"descripcion":1, "estado": 1},{name:"indice_descripcion_estado"})
```

Creación de índices sobre localizaciones geográficas:

```
db.articulos.createIndex({"venta.vendedor.loc": "2dsphere"})
```

Una vez creado este tipo de índice podríamos hacer búsquedas sobre el campo indexado(localización) utilizando criterios de geolocalización:

```
db.articulos.find({"venta.punto_de_venta":{$near:{$geometry:{type:"Point",coordinates:[10,20]},$minDistance: 0,$maxDistance:1000}}})
```

6 . MODIFICACIÓN DE DOCUMENTOS



Modificar el documento con descripción “Mando xBox negro” para que pase a estar vendido al usuario con email llopez@gmail.com y de nombre Luis

```
db.articulos.update({"descripcion":"Mando xBox Negro"},{$set:  
{"estado":"llopez@gmail.com","contraoferta.comprador":"Luis","contraoferta.fecha": new Date()}})
```

Actualizar la colección de usuarios para añadir al usuario con los datos:email:ffernandez@gmail.com,psw:"ffernandez2",via:"C/Escalona",num:8,ciudad:"Barcelona" si no existiera:

```
db.usuarios.updateOne({"e_mail":"ffernandez@gmail.com"},{$set:{ "nombre":"Fernando  
Fernandez","contraseña":"ffernandez2","e_mail":"ffernandez@gmail.com","ciudad":"Barcelona","vi  
a":"C/Escalona","num":8}})
```

Actualizar la colección de usuarios modificando el nombre de la C/Escalona por C/Ancha

```
db.usuarios.update({"e_mail":"ffernandez@gmail.com"},{$set:{ "via":"C/Ancha"}})
```

6 . MODIFICACIÓN DE DOCUMENTOS



Tecnologías de almacenamiento de datos

Añadir una contraoferta al artículo Thermomix a cargo del usuario con email ggomez@gmail.com y nombre “Gabriel” por 73 euros, si no existiera previamente:

```
db articulos.updateOne ({"product": "Thermomix"},{$push:{"contraoferta":  
{"ofertante":db.usuarios.findOne ({"nombre": "Gabriel Ruiz Sarosa"},  
{_id:1})._id}, "precio_oferta":73,"fecha_oferta": new Date()}}, {upsert:true})
```

Actualizar el document cuya descripción es Thermomix vendiéndoselo a la oferta anterior:

```
db articulos.updateOne ({"product":"Thermomix"},{$set:  
{"estado":"vendido","venta.$vendedor":db.usuarios .findOne({"nombre":"Gabriel Ruiz Sarosa"},  
{_id:1})._id}})
```

Eliminar el tag “entretenimiento” de todos los documentos de la colección items, siempre que exista:

```
db.articulos.update({ "etiquetas": "entretenimiento"}, {$pull:{ "etiquetas": "entretenimiento" }})
```

7. ELIMINAR DOCUMENTOS

Eliminar documentos

```
db.articulos.remove({"producto":"wii U"})
```

Eliminar colección
db.articulos.drop()

Eliminar la base de datos

```
db  
db.dropDatabase()  
show dbs
```

8. ELIMINAR COLECCIÓN

9. ELIMINAR BASE DE DATOS

