

MF0223_3 Sistemas operativos y aplicaciones informáticas

ENUNCIADO

Configurar un sistema informático con un sistema de gestión de bases de datos que permita actualizar una base de datos de comidas y de las calorías que contienen.

Para ello el programa deberá implementar una base de datos de los distintos tipos de platos, con al menos estar tres tablas: platos, con información de los distintos platos como lentejas patatas a la riojana, cuscús; otra tabla con la procedencia del plato: comida tradicional, asiática, moderna; y un tercero con la categoría de los platos: primeros, segundos, entrantes,... La base de datos no estará fijada sola a estas tres tablas, ya que al hacer el diseño relacional puede surgir alguna más.

El programa a realizar deberá permitirnos introducir nuevas recetas, y visualizar la lista de dichos platos. Se accederá a estos elementos a través de un menú principal con dos botones, uno para cada función.

Así mismo, también deberá documentarse el proyecto haciendo uso de javadoc y sacar el archivo HTML de la documentación

Estructura de esta documentación:

UF1465

- Configuración de componente de equipo gestión BD
- Descripción de la aplicación Web donde se maneja la base de datos.

UF1466

- Diseño del diagrama entidad relación
- Diseño del diagrama relacional.
- Implementación de la base de datos.

UF1467

- Comentarios del código de la aplicación elaborada en UF1465
- Generación de la documentación Javadoc
- Documentación Javadoc.

Cod: 287895	Intel Core i5-10400 2.90 GHz 1 unds. x 159,9 €	Precio: 159,90 €	Unds.: 1	Total: 159,90 €
Cod: 288963	Gigabyte Z490M 1 unds. x 125,9 €	Precio: 125,90 €	Unds.: 1	Total: 125,90 €
Cod: 308989	Crucial CT8G4DFS8266 DDR4 2666MHz PC4-21300 8GB CL19 2 unds. x 37,99 €	Precio: 37,99 €	Unds.: 2	Total: 75,98 €
Cod: 170503	Tacens Mars Gaming MCPU1V2 Ventilador CPU 1 unds. x 8,24 €	Precio: 8,24 €	Unds.: 1	Total: 8,24 €
Cod: 147002	Crucial MX500 SSD 500GB SATA 1 unds. x 67 €	Precio: 67 €	Unds.: 1	Total: 67 €
Cod: 182592	Seagate BarraCuda 3.5" 2TB SATA 3 1 unds. x 51,99 €	Precio: 51,99 €	Unds.: 1	Total: 51,99 €
Cod: 192065	Tacens Anima AC4 USB 3.0 Negro 1 unds. x 18,5 €	Precio: 18,50 €	Unds.: 1	Total: 18,50 €
Cod: 304187	Cooler Master Elite V3 600W PFC Activo 1 unds. x 42,99 €	Precio: 42,99 €	Unds.: 1	Total: 42,99 €
Cod: 358300	Asus DRW-24D5MT Grabadora DVD 24X Reacondicionado 1 unds. x 13,85 €	Precio: 13,85 €	Unds.: 1	Total: 13,85 €
Cod: 106018	LG 22MP68VQ-P 21.5 LED IPS Reacondicionado 1 unds. x 69,1 €	Precio: 69,10 €	Unds.: 1	Total: 69,10 €
Cod: 47278	Logitech Wireless Combo MK220 Teclado + Ratón 1 unds. x 19,99 €	Precio: 19,99 €	Unds.: 1	Total: 19,99 €
Cod: 85813	Microsoft Windows 10 Pro 64Bits OEM 1 unds. x 144,95 €	Precio: 144,95 €	Unds.: 1	Total: 144,95 €

798,39 €

*Se incluyen los materiales necesarios para un montaje de calidad profesional, como bridas de sujeción, pasta térmica para los procesadores, etc.

*Los accesorios de cada componente y sus embalajes son enviados junto con el equipo montado.

*La compatibilidad podría no estar asegurada al 100%, en caso de dudas contacte con nosotros en el centro de soporte.

UF1465

Descripción de la aplicación Web donde se maneja la base de datos.

La página inicial de nuestra aplicación es la siguiente:

The screenshot shows a web page with a teal header bar. In the top left corner of the bar, it says 'MF0223_3'. To its right are three links: 'Inicio', 'Listado platos', and 'Agregar plato'. Below the header, the main content area has a title 'Restaurar base de datos'. Underneath the title is a button labeled 'Seleccionar archivo' with the sub-label 'Ningún archivo seleccionado' to its right. At the bottom of this section are two buttons: a blue one labeled 'Cargar base de datos' and a white one labeled 'Ir al listado de platos'. At the very bottom of the page, there is a teal footer bar containing the copyright notice '© 2021 Arturo Montañez'.

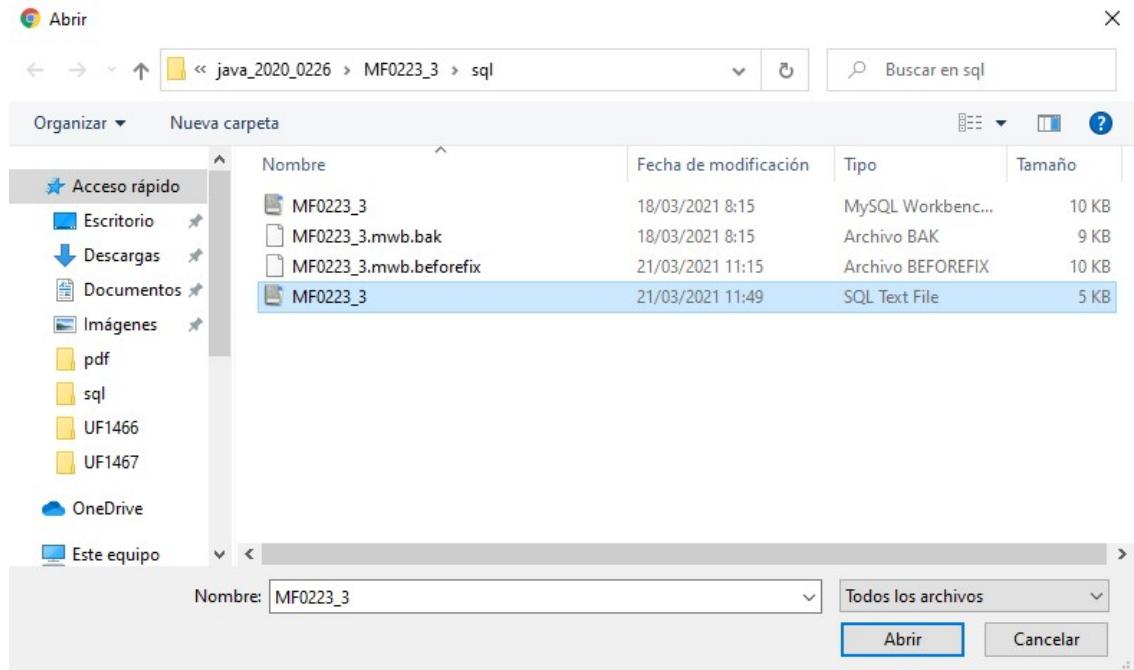
En esta pantalla podeos ver un menú superior donde tenemos las opciones de ir a Inicio (Que es la pantalla actual) “Listado de Platos” y “Agregar plato”. En la pantalla tenemos los botones “Seleccionar archivo”, “Cargar base de datos” e “Ir a listado de platos”.

Si tuviéramos ya cargada la base de datos podemos acceder directamente a la lista de platos pulsando al botón “Ir a listado de platos”, o a la opción del menú “Listado platos”. Dado que queremos ver el funcionamiento completo de la aplicación procederemos como vamos a contar a continuación.

Comprobamos las bases de datos que tenemos en nuestro sistema:

The screenshot shows a list of database schemas. At the top left, it says 'SCHEMAS'. Below that is a search bar with the placeholder 'Filter objects'. The list itself consists of a series of entries, each preceded by a small circular icon with a vertical line. The schemas listed are: backup_gestiondocente, ejemplojpa, ejercicios_sql, empadronamiento, gestiondocente, mydb, prueba, salon_masajes, supermercado, supermercadonuevo, tututu, uf1465_2, uf1466_2, uf2176, and usuarioservlets.

En nuestra aplicación pulsaremos sobre el botón “Seleccionar un archivo” para recuperar la copia de seguridad de nuestra base de datos. Nos saldrá una pantalla donde podremos seleccionar el archivo que queremos cargar:



Con lo cual se vería en la pantalla principal el archivo seleccionado:

MF0223_3 Inicio Listado platos Agregar plato

Restaurar base de datos

Seleccionar archivo MF0223_3.sql

Cargar base de datos Ir al listado de platos

© 2021 Arturo Montañez

A continuación pulsamos a “Cargar base de datos” y nos sale la pantalla con el listado de los platos:

MF0223_3 Inicio Listado platos Agregar plato

Base de datos cargada correctamente

Listado de platos

ID	Nombre	Calorías	Elaboración	Dificultad	Categoría	Origen
1	Patatas a la riojana	456	Patatas, pimientos, chorizo, sal, ajo	Media	Primero	Tradicional
2	Cuscus	345	Semola de trigo, verduras, pollo	Baja	Primero	Arabe
3	Lentejas	567	Lentejas, aceite, pimiento, cebolla, patata, zanahoria, chorizo, panceta	Media	Primero	Tradicional
4	Pizza	456	Masa de trigo, tomate, mozzarella, anchoas, orégano, alcacarras y aceite.	Baja	Segundo	Italiana
5	Chuleton ternera	987	Chuleton ternera, aceite, sal	Media	Segundo	Tradicional
6	Sushi	345	Pescado, arroz, algas	Alta	Entrante	Asiatica
7	Ensalada verde	123	Lechuga, tomate, cebolla, aceite, sal, aceitunas	Alta	Primero	Tradicional
8	Arroz con leche	789	Arroz, leche, azucar, canela	Baja	Postre	Tradicional
9	Tacos	789	Tortitas de maiz, pollo, verduras, salsa de guacamole	Baja	Segundo	Mexicana

Agregar plato

© 2021 Arturo Montañez

En caso de que al pulsar el botón “Cargar la base de datos” no hubiéramos seleccionado un archivo de respaldo, o hubiéramos cogido uno que no tuviera el formato correcto no saldría el siguiente mensaje de error:

The screenshot shows a web page titled "MF0223_3" with a navigation bar including "Inicio", "Listado platos", and "Agregar plato". A red error message box states: "Ha habido algún error con la carga de la base de datos. Debe seleccionar un archivo de backup correcto". Below this, a section titled "Restaurar base de datos" has a "Seleccionar archivo" input field containing "Ningún archivo seleccionado", a "Cargar base de datos" button, and a "Ir al listado de platos" button. At the bottom, a copyright notice reads "© 2021 Arturo Montañez".

Una vez cargada correctamente la base de datos:

The screenshot shows a database schema browser titled "SCHEMAS". It lists several schemas: "backup_gestiondocente", "ejemplojpa", "ejercicios_sql", "empadronamiento", "gestiondocente", and "mf0223_3". The "mf0223_3" schema is expanded to show its tables: "categorias", "origenes", and "platos". The "categorias" table has columns "id" and "nombre_categoria". The "origenes" table has columns "id" and "nombre_origen". The "platos" table has columns "id", "nombre_plato", "calorías", "elaboración", "dificultad", "categorias_id", and "origenes_id". There are also sections for "Indexes", "Foreign Keys", and "Triggers" for each table.

Indicaremos como introducir un nuevo plato en nuestra base de datos.

En la pantalla de listados pulsaremos el botón “Aregar plato”:

The screenshot shows a table titled "Listado de platos" with columns: Id, Nombre, Calorías, Elaboración, Dificultad, Categoría, and Origen. The table contains 9 rows of dish information. Below the table is a blue button labeled "Agregar plato".

Id	Nombre	Calorías	Elaboración	Dificultad	Categoría	Origen
1	Patatas a la riojana	456	Patatas, pimientos, chorizo, sal, ajo	Media	Primerº	Tradicional
2	Cuscus	345	Semola de trigo, verduras, pollo	Baja	Primerº	Arabe
3	Lentejas	567	Lentejas, aceite, pimiento, cebolla, patata, zanahoria, chorizo, panceta	Media	Primerº	Tradicional
4	Pizza	456	Masa de trigo, tomate, mozzarella, anchoas, orégano, alcacarras y aceite.	Baja	Segundo	Italiana
5	Chuleton ternera	987	Chuleton ternera, aceite, sal	Media	Segundo	Tradicional
6	Sushi	345	Pescado, arroz, algas	Alta	Entrante	Asiatica
7	Ensalada verde	123	Lechuga, tomate, cebolla, aceite, sal, aceitunas	Alta	Primerº	Tradicional
8	Arroz con leche	789	Arroz, leche, azucar, canela	Baja	Postre	Tradicional
9	Tacos	789	Tortillas de maíz, pollo, verduras, salsa de guacamole	Baja	Segundo	Mexicana

Y nos saldrá la siguiente pantalla:

The screenshot shows a form titled "Nuevo plato" with fields: Plato (text input), Calorías (text input), Elaboración (text area), Dificultad (dropdown: Alta), Categoría (dropdown: Primerº), and Origen (dropdown: Tradicional). Below the form is a blue button labeled "Agregar plato".

Introduciremos los datos, y para terminar pulsaremos al botón “Aregar plato”. En caso de que alguno de los campos sea obligatorio y no esté relleno, o que no tenga un formato correcto nos sacara un mensaje de error, y tendremos que introducir los datos correctamente para poder añadir el nuevo plato a la base de datos:

The screenshot shows the "Nuevo plato" form with validation errors. The "Plato" field has a red border and a tooltip "Completa este campo". The "Calorías" field has a red border and a tooltip "Utiliza un formato que coincida con el solicitado Las calorías deben ser un numero entero".

Finalmente una vez introducidos los datos correctamente iremos a al listado de platos donde nos saldrá el nuevo plato que hemos añadido a nuestra base de datos:

MF0223_3 Inicio Listado platos Agregar plato

Plato agregado correctamente

Listado de platos

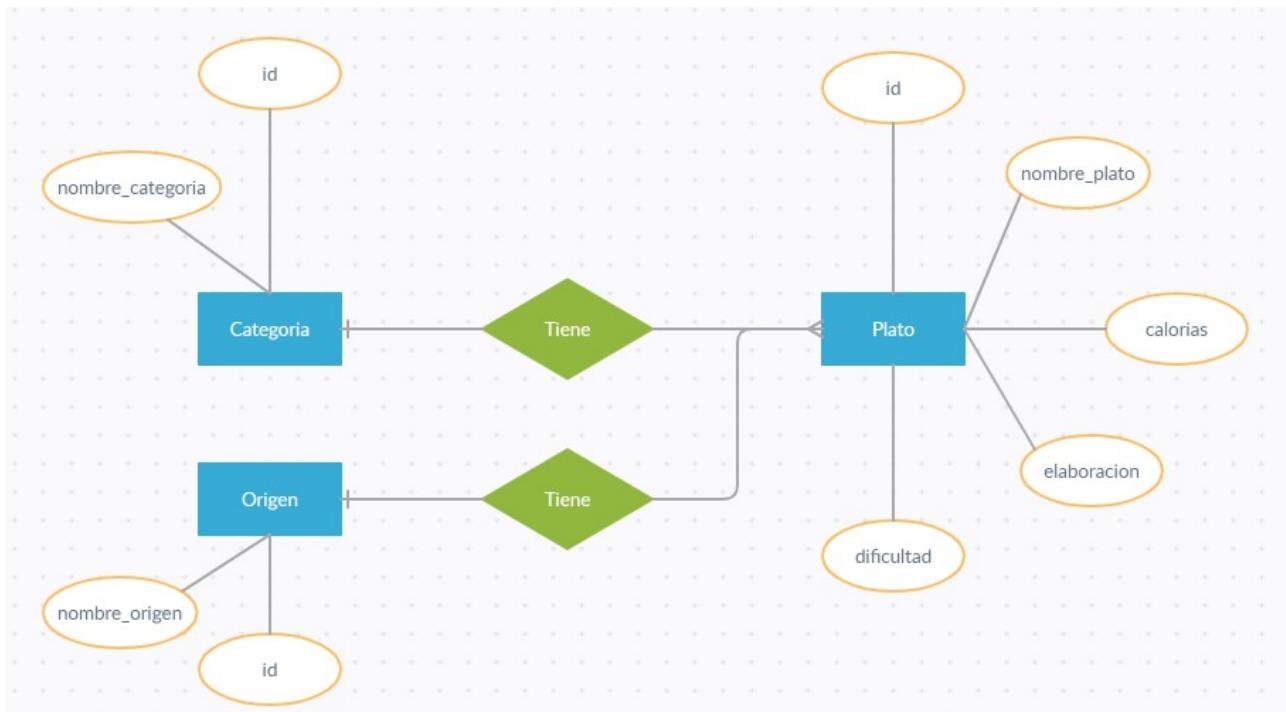
Id	Nombre	Calorías	Elaboración	Dificultad	Categoría	Origen
1	Patatas a la riojana	456	Patatas, pimientos, chorizo, sal, ajo	Media	Primero	Tradicional
2	Cuscus	345	Semola de trigo, verduras, pollo	Baja	Primero	Arabe
3	Lentejas	567	Lentejas, aceite, pimiento, cebolla, patata, zanahoria, chorizo, panceta	Media	Primero	Tradicional
4	Pizza	456	Masa de trigo,tomate, mozzarella, anchoas, orégano, alcacarras y aceite.	Baja	Segundo	Italiana
5	Chuleton ternera	987	Chuleton ternera, aceite, sal	Media	Segundo	Tradicional
6	Sushi	345	Pescado, arroz, algas	Alta	Entrante	Asiatica
7	Ensalada verde	123	Lechuga, tomate, cebolla, aceite, sal, aceitunas	Alta	Primero	Tradicional
8	Arroz con leche	789	Arroz, leche, azucar, canela	Baja	Postre	Tradicional
9	Tacos	789	Torotas de maiz, pollo, verduras, salsa de guacamole	Baja	Segundo	Mexicana
10	Sushi	210	Bonito, arroz, algas, huevo, salsa soja.	Media	Primero	Asiatica

Agregar plato

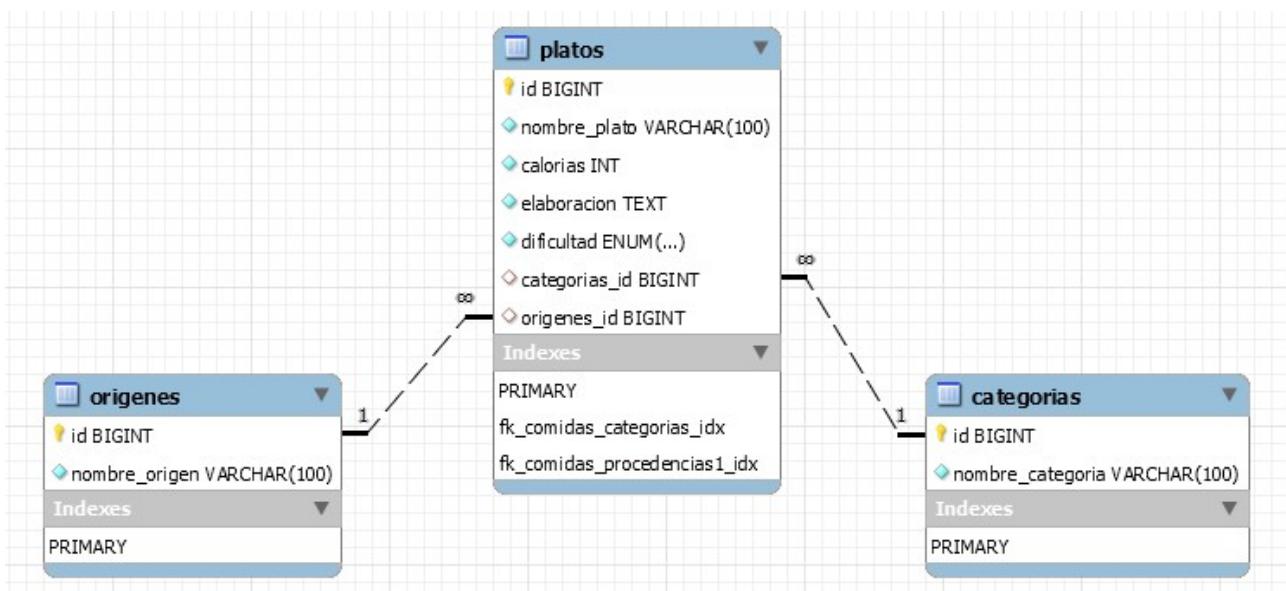
© 2021 Arturo Montañez

UF1466

Diseño del diagrama entidad relación.



Diseño del diagrama relacional.



Implementación de la base de datos.

A continuación pondremos el Script que no servirá para crear la base de datos:

```
CREATE DATABASE IF NOT EXISTS `mf0223_3` /*!40100 DEFAULT CHARACTER SET utf8mb4 COLLATE utf8mb4_0900_ai_ci */ /*!80016 DEFAULT ENCRYPTION='N' */;
USE `mf0223_3`;
-- MySQL dump 10.13 Distrib 8.0.22, for Win64 (x86_64)
--
-- Host: localhost Database: mf0223_3
-----
-- Server version 8.0.22

/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!50503 SET NAMES utf8 */;
/*!40103 SET @OLD_TIME_ZONE=@@TIME_ZONE */;
/*!40103 SET TIME_ZONE='+00:00' */;
/*!40014 SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0 */;
/*!40014 SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0 */;
/*!40101 SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='NO_AUTO_VALUE_ON_ZERO' */;
/*!40111 SET @OLD_SQL_NOTES=@@SQL_NOTES, SQL_NOTES=0 */;

--
-- Table structure for table `categorias`
--

DROP TABLE IF EXISTS `categorias`;
/*!40101 SET @saved_cs_client = @@character_set_client */;
/*!50503 SET character_set_client = utf8mb4 */;
CREATE TABLE `categorias` (
  `id` bigint NOT NULL AUTO_INCREMENT,
  `nombre_categoria` varchar(100) NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=8 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
/*!40101 SET character_set_client = @saved_cs_client */;

--
-- Dumping data for table `categorias`
--

LOCK TABLES `categorias` WRITE;
/*!40000 ALTER TABLE `categorias` DISABLE KEYS */;
INSERT INTO `categorias` VALUES
(1,'Primero'),(2,'Segundo'),(3,'Postre'),(4,'Entrante'),(5,'Aperitivo'),(6,'Merienda'),(7,'Otros');
/*!40000 ALTER TABLE `categorias` ENABLE KEYS */;
UNLOCK TABLES;

--
-- Table structure for table `origenes`
--

DROP TABLE IF EXISTS `origenes`;
/*!40101 SET @saved_cs_client = @@character_set_client */;
/*!50503 SET character_set_client = utf8mb4 */;
CREATE TABLE `origenes` (
  `id` bigint NOT NULL AUTO_INCREMENT,
```

```

`nombre_origen` varchar(100) NOT NULL,
PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=10 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
/*!40101 SET character_set_client = @saved_cs_client */;

-- 
-- Dumping data for table `origenes`
-- 

LOCK TABLES `origenes` WRITE;
/*!40000 ALTER TABLE `origenes` DISABLE KEYS */;
INSERT INTO `origenes` VALUES
(1,'Tradicional'),(2,'Asiatica'),(3,'Moderna'),(4,'Mexicana'),(5,'Arabe'),(6,'Italiana'),(7,'Peruana'),(8,'Centro Europa'),(9,'Otros');
/*!40000 ALTER TABLE `origenes` ENABLE KEYS */;
UNLOCK TABLES;

-- 
-- Table structure for table `platos`
-- 

DROP TABLE IF EXISTS `platos`;
/*!40101 SET @saved_cs_client    = @@character_set_client */;
/*!40101 SET character_set_client = utf8mb4 */;
CREATE TABLE `platos` (
`id` bigint NOT NULL AUTO_INCREMENT,
`nombre_plato` varchar(100) NOT NULL,
`calorias` int NOT NULL,
`elaboracion` text NOT NULL,
`dificultad` enum('Alta','Media','Baja') NOT NULL,
`categorias_id` bigint DEFAULT NULL,
`origenes_id` bigint DEFAULT NULL,
PRIMARY KEY (`id`),
KEY `fk_comidas_categorias_idx` (`categorias_id`),
KEY `fk_comidas_procedencias1_idx` (`origenes_id`),
CONSTRAINT `fk_comidas_categorias` FOREIGN KEY (`categorias_id`) REFERENCES `categorias` (`id`),
CONSTRAINT `fk_comidas_procedencias1` FOREIGN KEY (`origenes_id`) REFERENCES `origenes` (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=11 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
/*!40101 SET character_set_client = @saved_cs_client */;

-- 
-- Dumping data for table `platos`
-- 

LOCK TABLES `platos` WRITE;
/*!40000 ALTER TABLE `platos` DISABLE KEYS */;
INSERT INTO `platos` VALUES (1,'Patatas a la riojana',456,'Patatas, pimientos, chorizo, sal, ajo','Media',1,1),(2,'Cuscus',345,'Semola de trigo, verduras, pollo ','Baja',1,5),(3,'Lentejas',567,'Lentejas, aceite, pimiento, cebolla, patata, zanahoria, chorizo, panceta','Media',1,1),(4,'Pizza',456,'Masa de trigo,tomate, mozzarella, anchoas, orégano, alcarras y aceite','Baja',2,6),(5,'Chuleton ternera',987,'Chuleton ternera, aceite, sal','Media',2,1),(6,'Sushi',345,'Pescado, arroz, algas','Alta',4,2),(7,'Ensalada verde',123,'Lechuga, tomate, cebolla, aceite, sal, aceitunas','Alta',1,1),(8,'Arroz con leche',789,'Arroz, leche, azucar, canela','Baja',3,1),(9,'Tacos',789,'Tortas de maiz, pollo, verduras, salsa de guacamole','Baja',2,4),(10,'Sushi',210,'Bonito, arroz, algas, huevo, salsa soja','Media',1,2);
/*!40000 ALTER TABLE `platos` ENABLE KEYS */;
UNLOCK TABLES;

-- 
-- Dumping events for database 'mf0223_3'

```

```
--  
--  
-- Dumping routines for database 'mf0223_3'  
--  
/*!40103 SET TIME_ZONE=@OLD_TIME_ZONE */;  
  
/*!40101 SET SQL_MODE=@OLD_SQL_MODE */;  
/*!40014 SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS */;  
/*!40014 SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS */;  
/*!40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;  
/*!40101 SET CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */;  
/*!40101 SET COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */;  
/*!40111 SET SQL_NOTES=@OLD_SQL_NOTES */;
```

```
-- Dump completed on 2021-03-22 9:06:03
```

UF1467

Comentarios del código de la aplicación elaborada en UF1465

Dada la cantidad de archivos que tiene el proyecto, se van a mostrar solo algunos ejemplos representativos de los comentarios introducidos en el código

Acceso a datos.

Dao.java

```
package com.ipartek.formacion.mf0223_3.accesodatos;

import java.util.Set;

/**
 * Interfaz donde se declaran los métodos con los datos
 *
 * @author Arturo Montañez
 * @version 1.0
 * @param <T> Es el objeto que le vamos a pasar al Dao
 */
public interface Dao<T> {

    default Iterable<T> obtenerTodos(){
        throw new AccesoDatosException("NO IMPLEMENTADO");
    }
    default T obtenerPorId(long id){
        throw new AccesoDatosException("NO IMPLEMENTADO");
    }
    default T insertar(T objeto){
        throw new AccesoDatosException("NO IMPLEMENTADO");
    }
    default T modificar (T objeto){
        throw new AccesoDatosException("NO IMPLEMENTADO");
    }
    default void borrar(Long id){
        throw new AccesoDatosException("NO IMPLEMENTADO");
    }
}
```

DaoFabrica.java

```
package com.ipartek.formacion.mf0223_3.accesodatos;

import com.ipartek.formacion.mf0223_3.entidades.Categoría;

/**
 * Selecciona que acceso a datos vamos a usar
 *
 * @author Arturo Montañez
 * @version 1.0
 *
 */
public class DaoFabrica {
    private DaoFabrica() {}

    private static final Dao<Plato> daoPlato = new PlatoDaoMySql();
    private static final Dao<Categoría> daoCategoría = new CategoríaDaoMySql();
    private static final Dao<Origen> daoOrigen = new OrigenDaoMySql();

    /**
     * Método que devuelve el Dao del plato
     *
     * @return the daoplato
     */
    public static Dao<Plato> getDaoPlato() {
        return daoPlato;
    }
}
```

PlatoDaoMySql.java

```
package com.ipartek.formacion.mf0223_3.accesodatos;

import java.sql.Connection;

/**
 * Implementa los métodos de Dao para Plato
 *
 * @author Arturo Montañez
 * @version 1.0
 */

public class PlatoDaoMySql implements Dao<Plato>{

    private static final String SQL_SELECT = "select p.id,
        + "from platos p \r\n"
        + "left join categorias c on p.categorias_id =
        + "left join origenes o on p.origenes_id = o.i
    private static final String SQL_INSERT = "INSERT INTO |

    private DataSource dataSource = null;

    /**
     * Obtiene todos los platos de la base de datos
     *
     * @return platos
     */
    @Override
    public Iterable<Plato> obtenerTodos() {
        try (Connection con = Config.dataSource.getConnection();
            Statement s = con.createStatement();
            ResultSet rs = s.executeQuery(SQL_SELECT)) {
            List<Plato> platos = new ArrayList<>();
            while (rs.next()) {
                Plato p = new Plato();
                p.setId(rs.getInt("id"));
                p.setNombre(rs.getString("nombre"));
                p.setCategoría(rs.getString("categorias"));
                p.setOrigen(rs.getString("origenes"));
                platos.add(p);
            }
            return platos;
        } catch (SQLException e) {
            throw new RuntimeException(e);
        }
    }

    public void insertar(Plato plato) {
        try (Connection con = Config.dataSource.getConnection();
            PreparedStatement ps = con.prepareStatement(SQL_INSERT)) {
            ps.setString(1, plato.getNombre());
            ps.setString(2, plato.getCategoría());
            ps.setString(3, plato.getOrigen());
            ps.executeUpdate();
        } catch (SQLException e) {
            throw new RuntimeException(e);
        }
    }

    public void modificar(Plato plato) {
        try (Connection con = Config.dataSource.getConnection();
            PreparedStatement ps = con.prepareStatement(SQL_UPDATE)) {
            ps.setString(1, plato.getNombre());
            ps.setString(2, plato.getCategoría());
            ps.setString(3, plato.getOrigen());
            ps.setInt(4, plato.getId());
            ps.executeUpdate();
        } catch (SQLException e) {
            throw new RuntimeException(e);
        }
    }

    public void borrar(Plato plato) {
        try (Connection con = Config.dataSource.getConnection();
            PreparedStatement ps = con.prepareStatement(SQL_DELETE)) {
            ps.setInt(1, plato.getId());
            ps.executeUpdate();
        } catch (SQLException e) {
            throw new RuntimeException(e);
        }
    }
}
```

Controladores.

ListadoServlet.java

```
package com.ipartek.formacion.mf0223_3.controladores;

import java.io.IOException;

/**
 * Controlador que muestra el listado de los platos
 *
 * @author Arturo Montañez
 * @version 1.0
 */

@WebServlet("/listado")
public class ListadoServlet extends HttpServlet {
    private static final long serialVersionUID = 1L;
```

Entidades.

Plato.java

```
package com.ipartek.formacion.mf0223_3.entidades;

import java.io.Serializable;

/**
 * Representa los platos
 *
 * @author Arturo Montañez
 * @version 1.0
 */

@Data @NoArgsConstructor @AllArgsConstructor
public class Plato implements Serializable{
```

Lógica de negocio.

PlatoNegocio.java

```
package com.ipartek.formacion.mf0223_3.logicanegocio;

import java.util.Set;

/**
 * Interfaz donde se declaran los métodos de PlatoNegocio
 *
 * @author Arturo Montañez
 * @version 1.0
 */

public interface PlatoNegocio {

    Iterable<Plato> listadoPlatos();

    Plato agregarPlato(Plato plato);

}
```

PlatoNegocioImpl.java

```
package com.ipartek.formacion.mf0223_3.logicanegocio;

import com.ipartek.formacion.mf0223_3.accesodatos.Dao;

/**
 * Implementación de los métodos de la interfaz PlatoNegocio
 *
 * @author Arturo Montañez
 * @version 1.0
 */

public class PlatoNegocioImpl implements PlatoNegocio{
    private Dao<Plato> daoPlato = DaoFabrica.getDaoPlato();

    /**
     * Método que saca el listado de todos los platos
     *
     * @return platos
     */
    @Override
    public Iterable<Plato> listadoPlatos() {
        Iterable<Plato> platos = daoPlato.obtenerTodos();
        return platos;
    }

    /**
     * Método donde inserto un nuevo plato en la base de datos
     *
     * @param plato Es el nuevo plato que quiero agregar
     * @return Plato
     */
    @Override
    public Plato agregarPlato(Plato plato) {
        return daoPlato.insertar(plato);
    }
}
```

Generación de la documentación Javadoc

Al general el Javadoc comprobamos que no hay ningún error:

Documentación Javadoc.

Dada la cantidad de archivos html que tiene el javadoc, se van a mostrar solo algunos ejemplos representativos de los comentarios introducidos en el código.

En la página inicial de la documentación de Javadoc tenemos lo siguiente:



MF0223_3

The screenshot shows a dark blue header bar with white text. The first item 'OVERVIEW' is highlighted with an orange background. Other items include 'PACKAGE', 'CLASS', 'USE', 'TREE', 'DEPRECATED', 'INDEX', and 'HELP'. Below the header is a light gray content area. A section titled 'Packages' is shown, with 'com.ipartek.formacion.mf0223_3.accesodatos' listed.

Packages
com.ipartek.formacion.mf0223_3.accesodatos
com.ipartek.formacion.mf0223_3.controladores
com.ipartek.formacion.mf0223_3.entidades
com.ipartek.formacion.mf0223_3.logicanegocio

Vamos a ver algunos ejemplos de cada una de las capas.

Acceso a datos.

The screenshot shows a dark blue header bar with white text. The first item 'OVERVIEW' is highlighted with an orange background. Other items include 'PACKAGE', 'CLASS', 'USE', 'TREE', 'DEPRECATED', 'INDEX', and 'HELP'. Below the header is a light gray content area. A section titled 'Interface Summary' is shown, listing 'Dao<T>' with a description 'Interfaz donde se declaran los métodos con los datos'. A section titled 'Class Summary' is shown, listing 'CategoriaDaoMySql', 'DaoFabrica', 'OrigenDaoMySql', and 'PlatoDaoMySql' with their respective descriptions. A section titled 'Exception Summary' is shown, listing 'AccesoDatosException' with a description 'Captura las excepciones y lanza el mensaje'.

Interface Summary	
Interface	Description
Dao<T>	Interfaz donde se declaran los métodos con los datos

Class Summary	
Class	Description
CategoriaDaoMySql	Implementa los métodos de Dao para Categoria
DaoFabrica	Selecciona que acceso a datos vamos a usar
OrigenDaoMySql	Implementa los métodos de Dao para Origen
PlatoDaoMySql	Implementa los métodos de Dao para Plato

Exception Summary	
Exception	Description
AccesoDatosException	Captura las excepciones y lanza el mensaje



PlatoDaoMySql.java

OVERVIEW PACKAGE CLASS USE TREE DEPRECATED INDEX HELP

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAILED: FIELD | CONSTR | METHOD

Package com.ipartek.formacion.mf0223_3.accesodatos

Class PlatoDaoMySql

java.lang.Object
com.ipartek.formacion.mf0223_3.accesodatos.PlatoDaoMySql

All Implemented Interfaces:

Dao<Plato>

public class PlatoDaoMySql
extends java.lang.Object
implements Dao<Plato>

Implementa los métodos de Dao para Plato

Version:
1.0

Author:
Arturo Montañez

Constructor Summary

Constructors	Description
Constructor	
PlatoDaoMySql()	

Method Summary

All Methods	Instance Methods	Concrete Methods
Modifier and Type	Method	Description
Plato	insertar(Plato plato)	Inserta un nuevo plato en la base de datos
java.lang.Iterable<Plato>	obtenerTodos()	Obtiene todos los platos de la base de datos
Methods inherited from class java.lang.Object		
equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait		
Methods inherited from interface com.ipartek.formacion.mf0223_3.accesodatos.Dao		
borrar, modificar, obtenerPorId		

Constructor Details

PlatoDaoMySql

Controladores.

OVERVIEW PACKAGE CLASS USE TREE DEPRECATED INDEX HELP

Package com.ipartek.formacion.mf0223_3.controladores

Class Summary	
Class	Description
AgregarServlet	Controlador que se hace agrega un plato a la base de datos
Config	Configuración gestión de la fábrica
IndexServlet	Controlador que se hace la carga de una base de datos desde el fichero seleccionado
ListadoServlet	Controlador que muestra el listado de los platos

OVERVIEW PACKAGE CLASS USE TREE DEPRECATED INDEX HELP

IndexServlet.java

OVERVIEW PACKAGE CLASS USE TREE DEPRECATED INDEX HELP

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD

Package com.ipartek.formacion.mf0223_3.controladores

Class IndexServlet

```
java.lang.Object
    javax.servlet.GenericServlet
        javax.servlet.http.HttpServlet
            com.ipartek.formacion.mf0223_3.controladores.IndexServlet
```

All Implemented Interfaces:

```
java.io.Serializable, javax.servlet.Servlet, javax.servlet.ServletConfig
```

```
@WebServlet("/index")
public class IndexServlet
extends javax.servlet.http.HttpServlet
```

Controlador que se hace la carga de una base de datos desde el fichero seleccionado

Version:

1.0

Author:

Arturo Montañez

See Also:

Serialized Form

Constructor Summary

Constructors

Constructor
IndexServlet()

Method Summary

Methods inherited from class javax.servlet.http.HttpServlet

```
service
```

Methods inherited from class javax.servlet.GenericServlet

```
destroy, getInitParameter, getInitParameterNames, getServletConfig, getServletContext, getServletInfo, getServletName, init, init, log, log
```

Methods inherited from class java.lang.Object

```
equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait
```

Constructor Details

IndexServlet

public IndexServlet()

Entidades.

OVERVIEW PACKAGE CLASS USE TREE DEPRECATED INDEX HELP

Package com.ipartek.formacion.mf0223_3.entidades

Class Summary

Class	Description
Alerta	Representa las alertas de la aplicación cuando cargamos la base de datos
Categoría	Representa las categorías
Origen	Representa los orígenes
Plato	Representa los platos

OVERVIEW PACKAGE CLASS USE TREE DEPRECATED INDEX HELP

Plato.java

OVERVIEW PACKAGE CLASS USE TREE DEPRECATED INDEX HELP

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD

Package com.ipartek.formacion.mf0223_3.entidades

Class Plato

java.lang.Object
com.ipartek.formacion.mf0223_3.entidades.Plato

All Implemented Interfaces:
java.io.Serializable

```
public class Plato
extends java.lang.Object
implements java.io.Serializable
```

Represents the plates

Version:
1.0

Author:
Arturo Montañez

See Also:
Serialized Form

Constructor Summary

Constructors

Constructor
Plato()

Method Summary

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructor Details

Plato

public Plato()

Lógica de negocio.

OVERVIEW PACKAGE CLASS USE TREE DEPRECATED INDEX HELP

Package com.ipartek.formacion.mf0223_3.logicanegocio

Interface Summary

Interface	Description
CategoríaNegocio	Interface where the methods of CategoríaNegocio are declared
OrigenNegocio	Interface where the methods for OrigenNegocio are declared
PlatoNegocio	Interface where the methods of PlatoNegocio are declared

Class Summary

Class	Description
CategoríaNegocioImpl	Class where the methods of the CategoríaNegocio interface are implemented
FabricaNegocio	Class where the logic of the negocio is called
OrigenNegocioImpl	Class where the methods of the OrigenNegocio interface are implemented
PlatoNegocioImpl	Implementation of the methods of the PlatoNegocio interface

OVERVIEW PACKAGE CLASS USE TREE DEPRECATED INDEX HELP

PlatoNegocio.java

The screenshot shows a Java class documentation page for `PlatoNegocio`. The page includes the package declaration, interface summary, method summary, and method details.

Interface PlatoNegocio

All Known Implementing Classes:
`PlatoNegocioImpl`

public interface `PlatoNegocio`

Interfaz donde se declaran los métodos de `PlatoNegocio`

Version:
1.0

Author:
Arturo Montañez

Method Summary

All Methods **Instance Methods** **Abstract Methods**

Modifier and Type	Method	Description
<code>Plato</code>	<code>agregarPlato(Plato plato)</code>	
<code>java.lang.Iterable<Plato></code>	<code>listadoPlatos()</code>	

Method Details

`listadoPlatos`
`java.lang.Iterable<Plato> listadoPlatos()`

`agregarPlato`
`Plato agregarPlato(Plato plato)`

PlatoNegocioImpl.java

The screenshot shows a Java class documentation page for `PlatoNegocioImpl`. The page includes the package declaration, class summary, constructor summary, method summary, and constructor details.

Class PlatoNegocioImpl

`java.lang.Object`
`com.ipartek.formacion.mnf0223_3.logicanegocio.PlatoNegocioImpl`

All Implemented Interfaces:
`PlatoNegocio`

public class `PlatoNegocioImpl`
extends `java.lang.Object`
implements `PlatoNegocio`

Implementación de los métodos de la interfaz `PlatoNegocio`

Version:
1.0

Author:
Arturo Montañez

Constructor Summary

Constructors

Constructor	Description
<code>PlatoNegocioImpl()</code>	

Method Summary

All Methods **Instance Methods** **Concrete Methods**

Modifier and Type	Method	Description
<code>Plato</code>	<code>agregarPlato(Plato plato)</code>	Método donde inserto un nuevo plato en la base de datos
<code>java.lang.Iterable<Plato></code>	<code>listadoPlatos()</code>	Método que saca el listado de todos los platos

Methods inherited from class `java.lang.Object`

`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Constructor Details

`PlatoNegocioImpl`

```
public PlatoNegocioImpl()
```