

Base de Datos y Aplicación Cultivo Hidropónico



Índice

- Presentación Base de Datos Cultivos
- Tablas
- Triggers
- Procedimientos
- Aplicación web de Cultivos

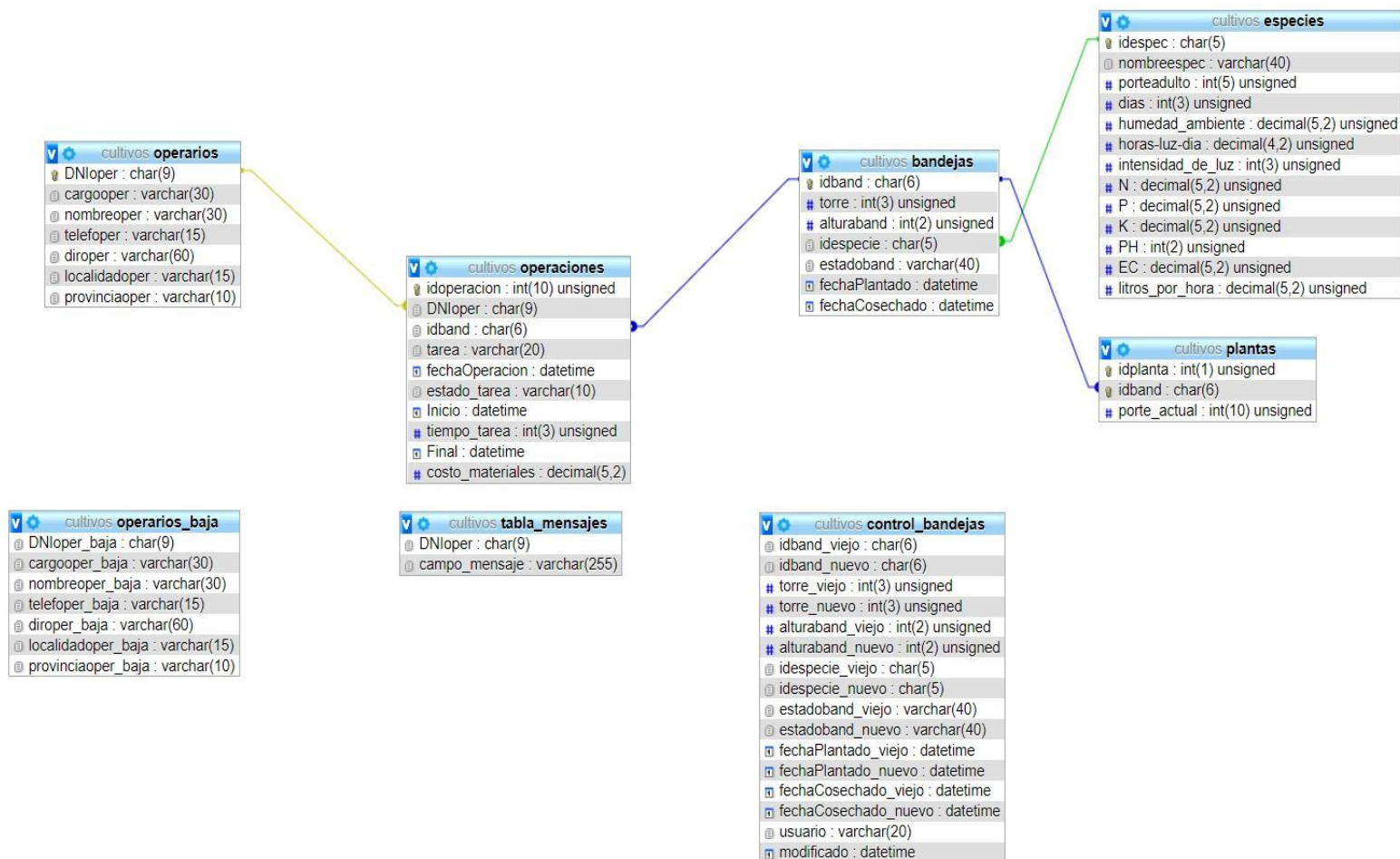
PRESENTACIÓN BASE DE DATOS CULTIVOS

Esta base de datos, junto con su aplicación web, que hemos llamado Cultivos, permite la gestión de una granja hidropónica de interior. Se trata de un modelo de agricultura que permite controlar todos los parámetros de crecimiento de las plantas. Las raíces de estas se encuentran sumergidas en agua, a través de la que se les pasan disueltos los minerales que necesitan para su crecimiento. La iluminación que reciben es artificial mediante lámparas LED, lo que permite que la luz llegue por igual a todas las plantas, aunque se encuentren apiladas (reduciendo así el espacio necesario para los cultivos).

Los operarios siembran las plantas en bandejas, que se apilan en torres, y a su vez las torres se encuentran en una nave industrial climatizada. Cada bandeja tiene encima su correspondiente luz LED y su bomba de agua, así como un sistema que permite ajustar la humedad ambiente.

Los sensores y actuadores incorporados en las bandejas, permiten leer y cambiar parámetros necesarios para la óptima producción de cada especie. Sin embargo los datos son tomados por operarios y los sensores calibrados por un técnico especializado. La recolección también la hacen los operarios.

La estructura que hemos elegido es la siguiente:



Trabajo Final BDD

José Eduardo R. B. – Víctor Daniel D. M.

TABLAS

- Tenemos 5 tablas principales que definen la estructura de la base de datos:

-**Operarios** (*DNloper, cargooper, nombreoper, telefooper, diroper, localidadoper, provinciaoper*):

En esta tabla se ha almacenado los trabajadores de la explotación, junto a sus datos más significativos, además se ha configurado DNloper como clave primaria, ya que este campo nunca se repetirá.

-**Operaciones** (*idoperacion, DNloper, idband, tarea, tiempo_tarea, costo_materiales, fechaOperacion*):

Está formada por un diario de todas las operaciones que los operarios realizan sobre la explotación con el objetivo de monitorizarlas, la clave primaria es idoperacion. Como las tareas se realizan siempre en las bandejas o sobre ellas, idband siempre tendrá que corresponder con una bandeja que tengamos, así como el operario.

-**Bandejas** (*idband, torre, alturaband, idespecie, estadoband, fechaplantado*):

Monitoriza el estado y posición de cada bandeja así como la especie plantada en la misma (una bandeja solo puede tener una especie de plantas)

-**Plantas** (*idplanta, idband, porte_actual*):

Muestra el porte actual de cada planta en cada bandeja. (cada bandeja puede tener hasta 6 plantas), la clave primaria está formada por idplanta e idband, y el porte está expresado en milímetros.

-**Especies** (*idespec, nombreespec, porteadulto, días, humedad_ambiente, horas_luz_dia, intensidad_de_luz, N, P, K, PH, EC, litros_por_hora*):

Muestra cuales son los parámetros ambientales óptimos para el crecimiento de la una determinada especie.

Relaciones entre tablas principales:

- Un Operario puede realizar varios tipos de Operaciones (relación 1 a ∞), pero una Operación sólo puede ser realizada por un Operario.
- Una Bandeja admite varios tipos Operaciones (relación 1 a ∞), pero una Operación sólo puede ser asignada a una Bandeja concreta.
- En una Bandeja se pueden plantar varias Plantas (relación 1 a ∞), de hecho nosotros hemos decidido que sean 6 plantas por bandeja. Pero una Planta sólo puede ser plantada en una Bandeja.
- Se puede plantar una Especie en varias Bandejas (relación 1 a ∞). Pero una Bandeja sólo puede albergar una Especie.

- También tenemos 3 tablas que sirven de soporte para borrados, actualizaciones y mensajes:

-Operarios_baja (*DNloper_baja, cargooper_baja, nombreoper_baja, telefooper_baja, diroper_baja, localidadoper_baja, provinciaoper_baja*):

Se utilizará para almacenar los operarios que demos de baja del sistema, es decir, para almacenar los operarios que son eliminados de la tabla Operarios. Cuando se elimina un operario en la tabla Operarios, el Trigger Before Delete **operarios_bd** almacena los valores de los campos del operario eliminado en unas variables globales con valores OLD; después, el Trigger After Delete **operarios_baja_ad** se encarga de almacenar los valores de las variables globales en los campos de un registro en la tabla Operarios_baja.

Cuando queramos volver a dar de alta al operario, bastará con eliminarlo de la tabla Operarios_baja, y el Trigger Before Delete **operarios_baja_bd** enviará los valores de sus campos de cuando estaba de baja a unas variables globales con valores OLD; después, el Trigger After Delete **operarios_ad** se encargará de almacenar los valores de las variables globales en los campos de un registro en la tabla Operarios.

-Control_bandejas (*idband_viejo, idband_nuevo, torre_viejo, torre_nuevo, alturaband_viejo, alturaband_nuevo, idespecie_viejo, idespecienuevo, estadoband_viejo, estadoband_nuevo, fechaplantado_viejo, fechaplantado_nuevo*):

Se utiliza para almacenar cualquier actualización que hagamos sobre los campos de un registro de la tabla Bandejas. Primero, el Trigger Before Update **bandejas_bu** se encarga de preservar en variables globales, cargadas con valores OLD, los valores de los campos antes de una actualización en un registro de la tabla Bandejas. Después, el Trigger After Update **bandejas_au** guardará en la tabla Control_bandejas los valores de las variables globales cargadas a valor OLD, pero también guardará en la tabla Control_bandejas los valores de las variables globales cargadas a valor NEW (el valor de una actualización). Así, la tabla Control_bandejas servirá de registro comparativo de los valores viejos y los cambios realizados en tabla Bandejas por un usuario en la fecha de modificación.

-Tabla_mensajes (*DNloper, campo_mensaje*):

Será usada únicamente para guardar los mensajes que lance un trigger muy peculiar: el Trigger After Insert **operaciones_ai**, del que hablaremos en la sección de triggers. En esta tabla se almacena un mensaje en *campo_mensaje*, que está asociado al identificador del operario *DNloper*, una persona que es anotada (insertada) por su jefe en la tabla Operaciones para que haga una tarea concreta. Pero si al jefe se le olvida asignar un valor de *costo_materiales* a los materiales necesarios para la tarea, nuestro trigger le recordará (después de la inserción) que es necesario que rellene ese campo en la tabla Operaciones.

TRIGGERS

Una de las nuevas características incluidas en MySQL son los triggers o disparadores, que son “oyentes”, que se mantienen a la escucha de los eventos que puedan producirse en una tabla (**insert** , **update** y **delete**) y ejecutan un código antes (**before**) o después (**after**) de que se produzca este evento.

Para referenciar las columnas de antes y después de que el evento se haya disparado, se usan las palabras clave **OLD** y **NEW**. Con la sentencia INSERT solo se permite NEW, con UPDATE se permiten ambas y con DELETE solo OLD.

Nosotros hemos querido hacer un trigger de cada tipo, es decir:

- Trigger Before Insert y Trigger After Insert.
- Trigger Before Update y Trigger After Update.
- Trigger Before Delete y Trigger After Delete.

- Trigger Before Insert

Editar disparador

Detalles

Nombre del disparador	plantas_bi
Tabla	plantas
Tiempo	BEFORE
Evento	INSERT
Definición	<pre>1 BEGIN 2 3 DECLARE errorMessage VARCHAR(255); 4 SET errorMessage = CONCAT('La planta sólo puede tener un id numérico 5 del 1 al 6, incluidos'); 6 7 IF NEW.idplanta<1 OR NEW.idplanta>6 THEN 8 SIGNAL SQLSTATE '45000' 9 SET MESSAGE_TEXT = errorMessage; 10 END IF; 11 END</pre>
Definidor	root@localhost

Se trata de un disparador que se activa antes de que se produzca el guardado de la inserción de un registro, en nuestro caso en la tabla Plantas. En concreto, nuestro trigger lanza un mensaje de error a modo de aviso que detiene la inserción de una planta cuando su id numérico es mayor que 6 (pues en una bandeja sólo puede haber seis plantas), o cuando su id numérico es menor que 1 (pues en una bandeja no puede tener plantas negativas o una planta cero, para eso tenemos NULL plantas por defecto).

```
CREATE TRIGGER `plantas_bi` BEFORE INSERT ON `plantas`  
  
FOR EACH ROW BEGIN  
  
    DECLARE errorMessage VARCHAR(255);  
  
    SET errorMessage = CONCAT('La planta sólo puede tener un id numérico del  
1 al 6, incluidos');  
  
    IF NEW.idplanta<1 OR NEW.idplanta>6 THEN  
  
        SIGNAL SQLSTATE '45000'  
  
        SET MESSAGE_TEXT = errorMessage;  
  
    END IF;  
  
END
```

- Trigger After Insert

Es un peculiar disparador que se activa después de que se produzca el guardado de la inserción de un registro, en nuestro caso en la tabla Operaciones.

El problema que presenta un trigger After Insert es, que al estar en ejecución después del guardado de la tabla que lo activa, ya no es posible para el trigger actuar sobre dicha tabla. Es decir, el código del trigger After Insert no podrá insertar, actualizar, ni borrar registros de dicha tabla. Y precisamente esas son las acciones que podrían interesarnos realizar si por ejemplo nos equivocáramos con un valor de un campo del registro insertado, o si se nos olvidara rellenar un campo.

Nuestro Trigger After Insert **operaciones_ai** se activa después del guardado de un registro, pero en el que se nos olvida o no quisimos rellenar, un campo llamado **costo_materiales** (que se queda a NULL por defecto):

```
CREATE TRIGGER `operaciones_ai` AFTER INSERT ON `operaciones`

FOR EACH ROW BEGIN

IF NEW.costo_materiales IS NULL THEN

INSERT INTO Tabla_mensajes ( DNIoper, campo_mensaje )

VALUES( NEW.DNIoper, CONCAT('El operario tiene costo_materiales a NULL' ) );

END IF;

END
```

Trabajo Final BDD

José Eduardo R. B. – Víctor Daniel D. M.

No podemos actuar sobre la tabla Operaciones, pero podemos enviarnos un mensaje recordatorio que nos sirva para tener presente que hay que dar un valor numérico a costo_materiales. Lo hacemos insertando el mensaje en un campo, pero será de un registro en otra tabla a la que llamaremos Tabla_mensajes, que tiene dos campos para sus registros (DNloper, campo_mensaje).

Como se puede ver en el código, si se cumple la condición del IF se hace la inserción de un mensaje CONCAT en campo_mensaje, que alude al operario identificado mediante NEW.DNIloper en la tabla Operaciones (dicho operario rellenó a valores NEW todos los campos del último registro añadido en Operaciones, menos el campo costo_materiales que se quedó a NULL).

Antes de insertar un registro (con costo_materiales a NULL) en tabla Operaciones:

The image consists of two side-by-side screenshots of the phpMyAdmin interface. The left screenshot shows the 'Estructura' (Structure) tab for the 'operaciones' table. The right screenshot shows the 'Estructura' tab for the 'tabla_mensajes' table.

Left Screenshot: phpMyAdmin - Base de datos: cultivos - Tabla: operaciones

Columna	Tipo	Función	Nulo	Valor
idoperacion	int(10) unsigned			11
DNIloper	char(9)			93567223J - 93567223J
idband	char(6)			001-01 - 001-01
tarea	varchar(20)			plantar
fechaOperacion	datetime			2022-11-03 02:00:00
estado_tarea	varchar(10)			Por_hacer
Inicio	datetime			0000-00-00 00:00:00
tiempo_tarea	int(3) unsigned			10
Final	datetime			0000-00-00 00:00:00
costo_materiales	decimal(5,2)		<input checked="" type="checkbox"/>	

Right Screenshot: phpMyAdmin - Base de datos: cultivos - Tabla: tabla_mensajes

MySQL ha devuelto un conjunto de valores vacío (es decir: cero columnas). (La consulta tardó 0,00 segundos)

```
SELECT * FROM `tabla_mensajes`
```

DNloper campo_mensaje

Operaciones sobre los resultados de la consulta

Crear vista

Guardar esta consulta en favoritos

Etiqueta: ☐ Permitir que todo usuario pueda acceder a este favorito

Después de insertar, obtenemos un mensaje del trigger en Tabla_mensajes:

The first screenshot shows the 'operaciones' table in the 'cultivos' database. It displays two rows of data with columns: idoperacion, DNloper, idband, tarea, fechaOperacion, estado_tarea, Inicio, tiempo_tarea, Final, and costo_materiales.

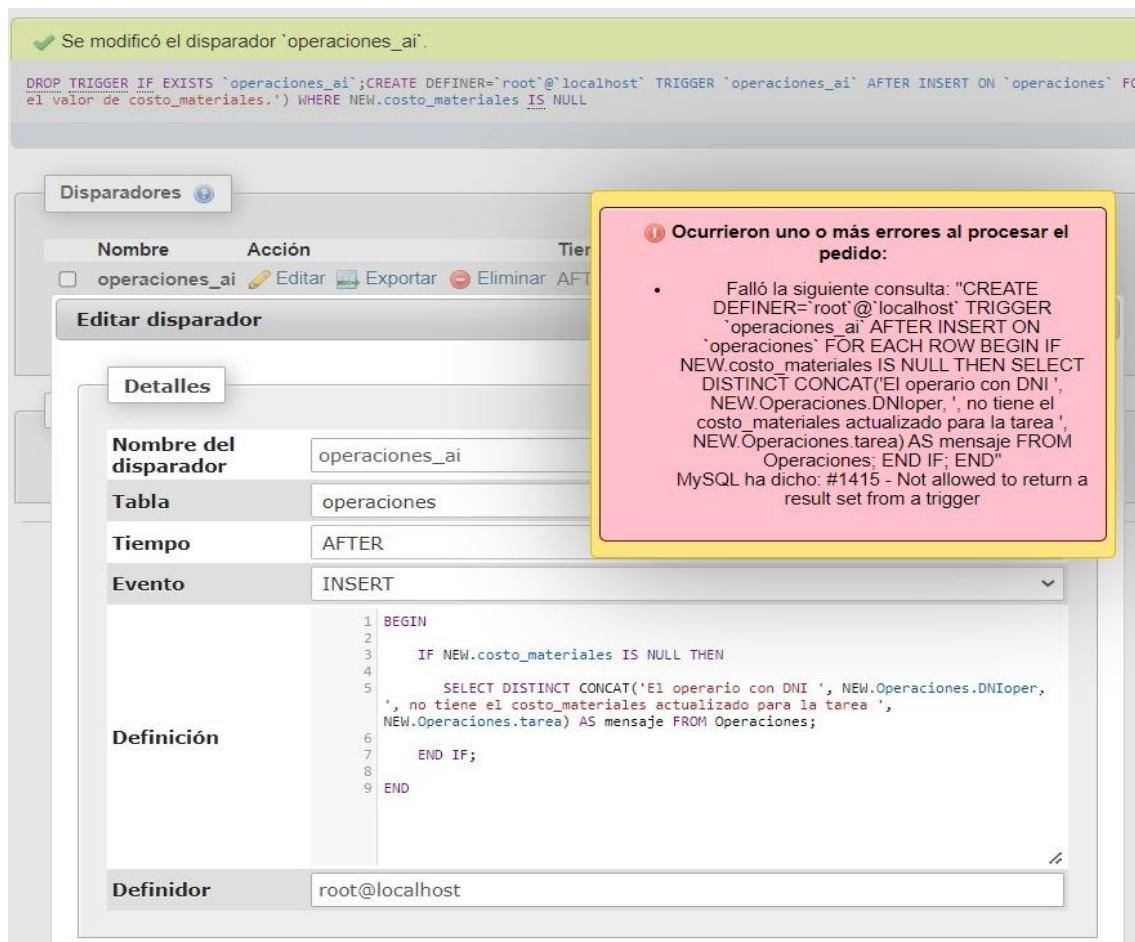
idoperacion	DNloper	idband	tarea	fechaOperacion	estado_tarea	Inicio	tiempo_tarea	Final	costo_materiales
11	93557223J	001-01	plantar	2022-11-03 02:00:00	Por_hacer	0000-00-00 00:00:00	10	0000-00-00 00:00:00	NULL
12	93557223J	001-02	plantar	2022-11-03 02:00:00	Por_hacer	0000-00-00 00:00:00	10	0000-00-00 00:00:00	2.25

The second screenshot shows the 'tabla_mensajes' table, which contains one message from a trigger. The message text is: '93557223J El operario tiene costo_materiales a NULL'.

mensaje
93557223J El operario tiene costo_materiales a NULL

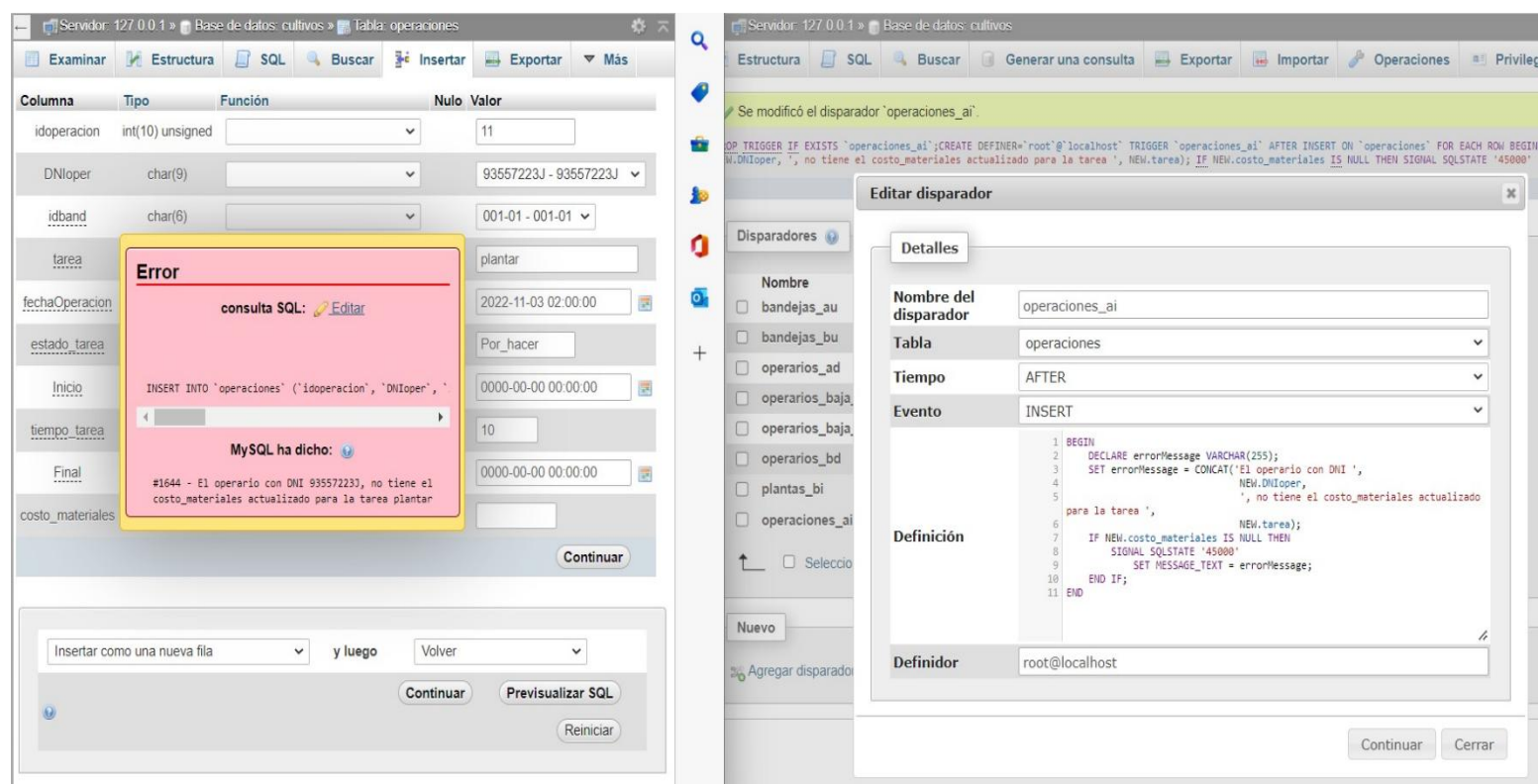
Viendo la función de nuestro trigger, podríamos pensar para qué insertar el mensaje en una tabla nueva “Tabla_mensajes” que encima hemos tenido que crear previamente y, lo que es peor, ha pasado a formar parte de nuestro diseño de la base de datos. ¿No sería mejor que el trigger, cuando se cumpliera la condición del IF, ejecutara una consulta SELECT que nos imprimiera un mensaje?

Por supuesto que sería mejor, pero como se ve en la imagen siguiente, los trigger no tienen permitido hacer consultas ni nada que se parezca a retornar un resultado por pantalla:



Y es entonces al ver esta imagen cuando se nos ocurre una idea, ¿acaso no está el trigger aquí, porque no puede hacer el SELECT, llamando a una ventana de error rosa que nos muestra un mensaje (de error) ? ¿No sería posible modificar el mensaje de error para que muestre el mensaje que nos interesa? .

La respuesta es que sí se puede, y además, como veremos en la siguiente imagen, podemos aprovechar el buen tamaño de la ventana de error para escribir un mensaje largo porque es más visualizable que un mensaje almacenado en el campo de un registro (éste puede contener mensajes largos de hasta 21845 caracteres, pero la ventana máxima de visualización es de 50 caracteres):



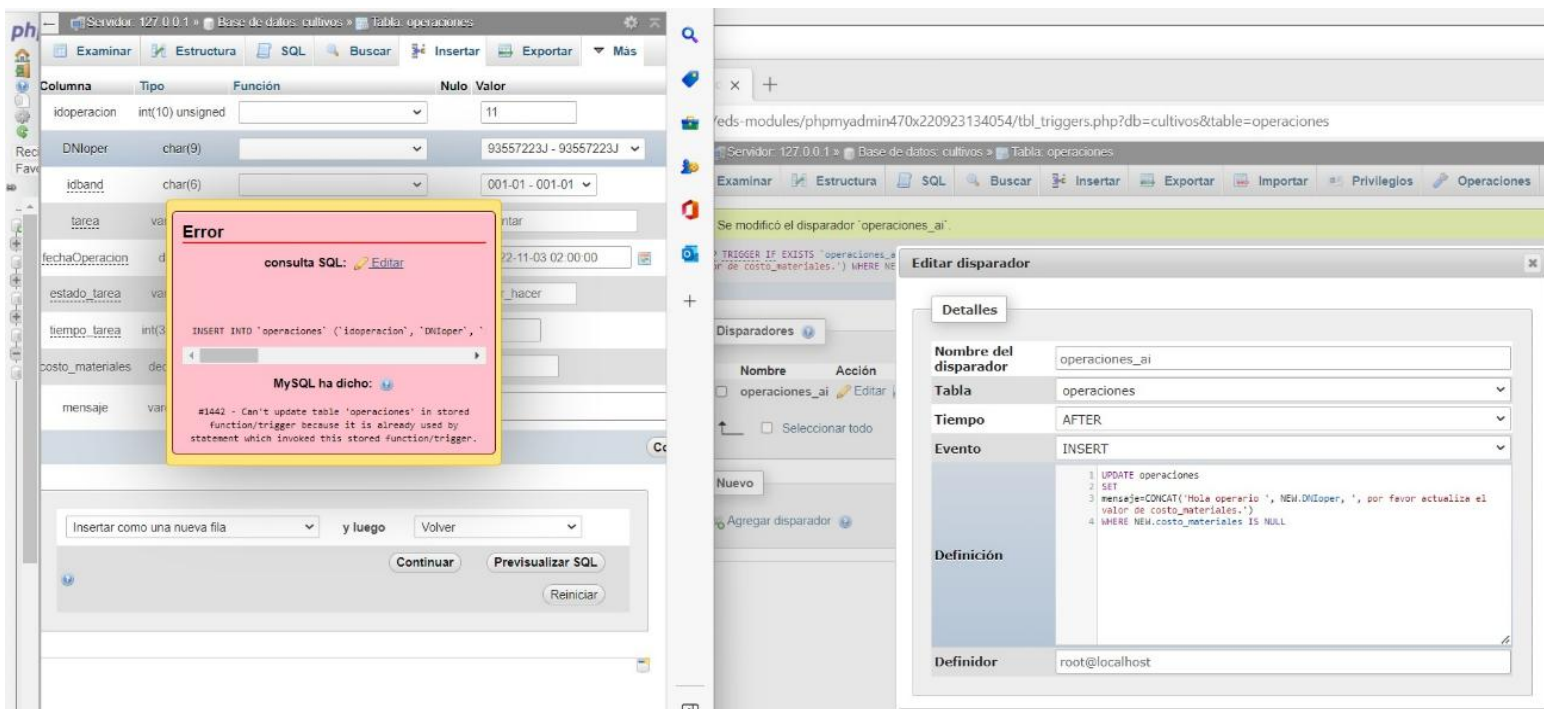
Como podemos observar, al intentar insertar un registro (con costo_materiales a NULL) en tabla Operaciones, el trigger ha forzado a sacar una ventana de error con el mensaje que nos interesaba. Pero démonos cuenta de que el mensaje de error ha emergido antes de que el registro pudiera ser insertado y guardado en la tabla Operaciones, interrumpiendo el proceso como hacen los mensajes de error del sistema.

La funcionalidad es totalmente correcta y deseada, pero es el tipo de funcionalidad de un trigger Before Insert, y no lo que intentábamos conseguir para el caso que nos ocupa: un trigger After Insert que muestre el mensaje de aviso después de insertar un registro.

Por lo que hemos visto, la única forma de aprovechar la funcionalidad de un trigger After Insert es volcar el mensaje a un campo de registro de una tabla diferente a la tabla llamadora del trigger.

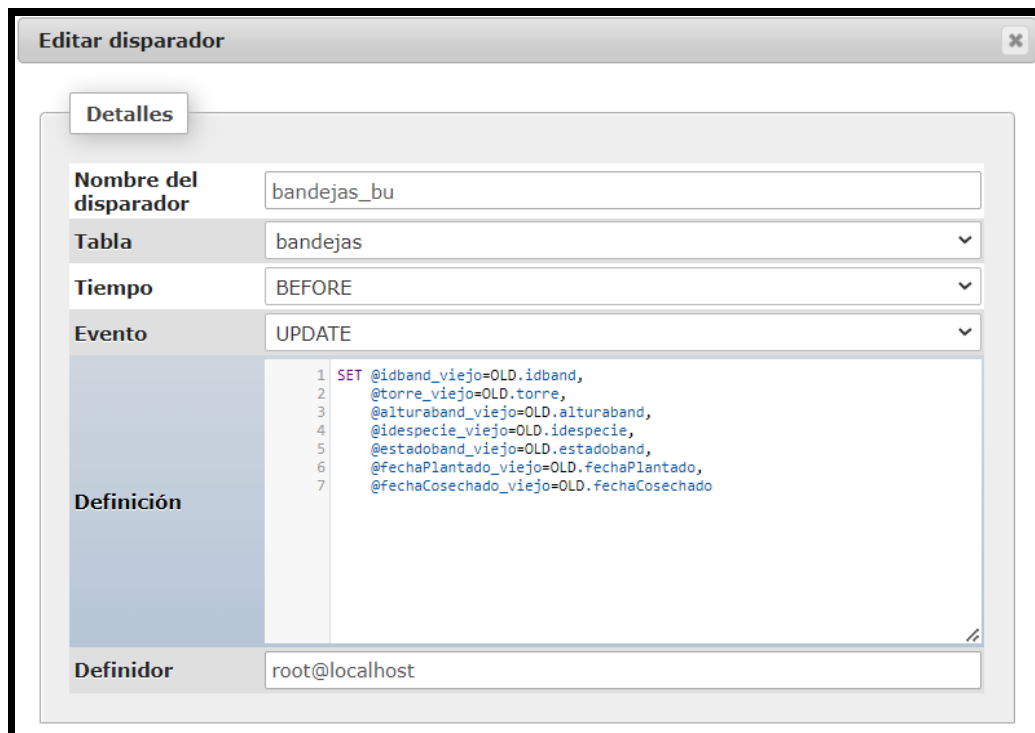
Tema aparte sería que la funcionalidad de un trigger After Insert nos resulte conveniente o no. Pues un trigger que obliga a un usuario mediante un mensaje a corregir manualmente un problema que ni siquiera fue capaz de detener, parece resultar poco útil y práctico.

Para acabar nuestro análisis de este trigger, aprovechamos para demostrar en la imagen siguiente que el trigger After Insert no permite hacer algo que sí lo haría bastante útil, ejecutar una consulta de actualización del registro incorrecto que se hiciera efectiva después de insertar y dejara al registro corregido:



- Trigger Before Update

El trigger Before Update **bandejas_bu** se utiliza para preservar en variables globales **_viejo** el valor OLD de los campos de los registros actualizados/modificados en la tabla Bandejas. Justo al hacer click en el Editar de un registro (para empezar a modificar valores de campos), se llama a este trigger y se ejecuta la orden SET que le hemos programado. A continuación se llama automáticamente al trigger After Update **bandejas_au**.

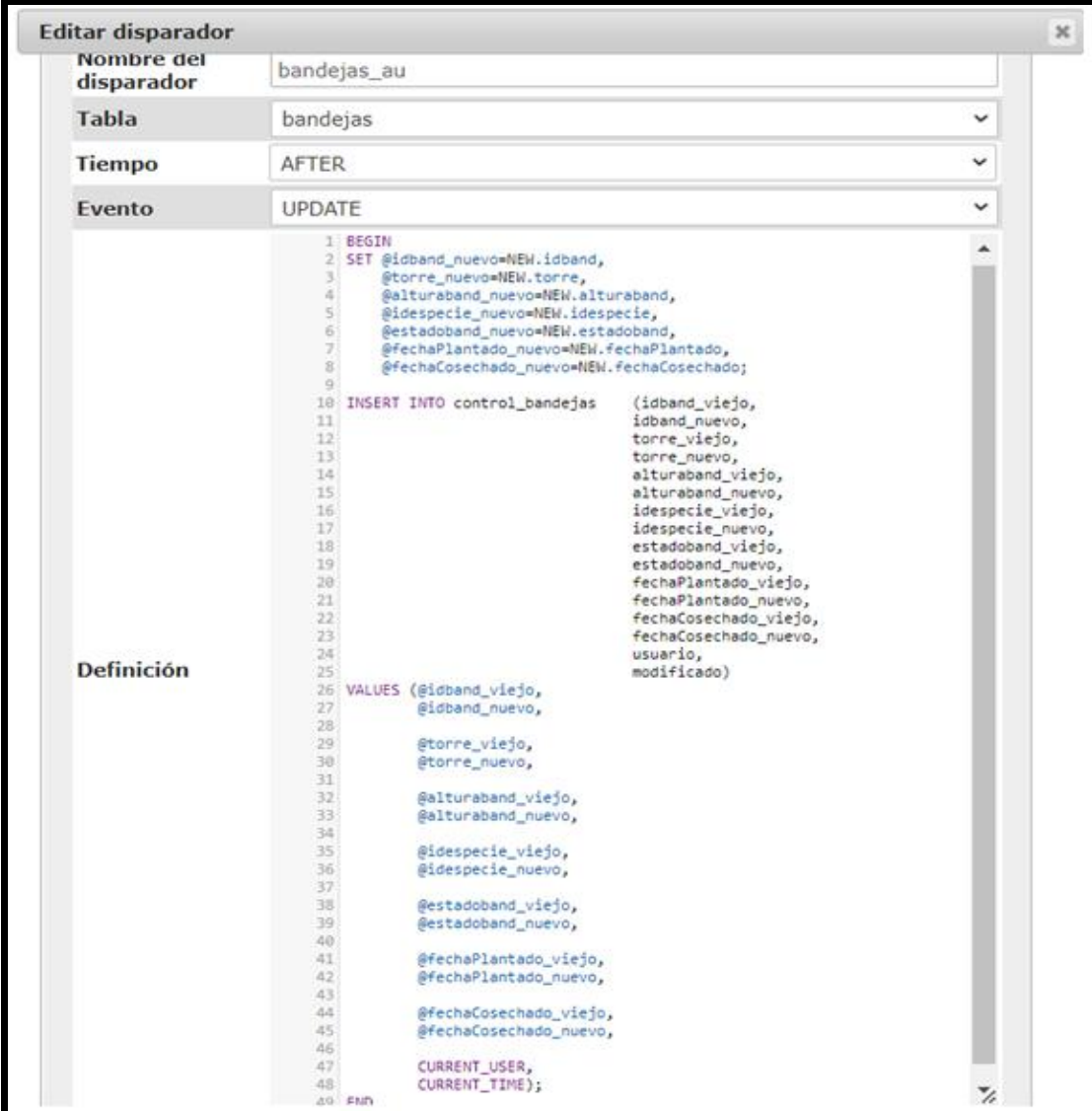


Prácticamente actúa como un trigger Before Delete, pero en vez de activarse antes de borrar un registro, se activa antes de actualizar/editar un registro.

- Trigger After Update

Gracias al trigger Before Update **bandejas_bu**, habíamos preservado en variables globales *_viejo* los valores de campos de un registro a editar en la tabla Bandejas. El trigger After Update **bandejas_au** se va a utilizar para insertar/guardar los valores a OLD de las variables globales *_viejo* en campos de un registro de la tabla Control_bandejas, pero también para insertar/guardar los valores a NEW de las variables globales *_nuevo*. Estas últimas se habrían rellenado al principio del código (en la orden SET) con los nuevos valores introducidos en los campos del registro que estemos actualizando/modificando.

Así, la tabla Control_bandejas actuará como un almacén de los valores OLD y New de registros editados en la tabla Bandejas.



The screenshot shows a window titled "Editar disparador" (Edit trigger). It contains the following configuration:

- Nombre del disparador: bandejas_au
- Tabla: bandejas
- Tiempo: AFTER
- Evento: UPDATE

The "Definición" (Definition) tab shows the following SQL code:

```
1 BEGIN
2 SET @idband_nuevo=NEW.idband,
3     @torre_nuevo=NEW.torre,
4     @alturaband_nuevo=NEW.alturaband,
5     @idespecie_nuevo=NEW.idespecie,
6     @estadoband_nuevo=NEW.estadoband,
7     @fechaPlantado_nuevo=NEW.fechaPlantado,
8     @fechaCosechado_nuevo=NEW.fechaCosechado;
9
10 INSERT INTO control_bandejas (idband_viejo,
11                               idband_nuevo,
12                               torre_viejo,
13                               torre_nuevo,
14                               alturaband_viejo,
15                               alturaband_nuevo,
16                               idespecie_viejo,
17                               idespecie_nuevo,
18                               estadoband_viejo,
19                               estadoband_nuevo,
20                               fechaPlantado_viejo,
21                               fechaPlantado_nuevo,
22                               fechaCosechado_viejo,
23                               fechaCosechado_nuevo,
24                               usuario,
25                               modificado)
26 VALUES (@idband_viejo,
27          @idband_nuevo,
28          @torre_viejo,
29          @torre_nuevo,
30          @alturaband_viejo,
31          @alturaband_nuevo,
32          @idespecie_viejo,
33          @idespecie_nuevo,
34          @estadoband_viejo,
35          @estadoband_nuevo,
36          @fechaPlantado_viejo,
37          @fechaPlantado_nuevo,
38          @fechaCosechado_viejo,
39          @fechaCosechado_nuevo,
40          CURRENT_USER,
41          CURRENT_TIME);
42
43 END
```

Este trigger prácticamente actúa como Un trigger After Insert, pero en vez de activarse después de insertar y guardar un registro, se activa después de guardar un registro en el que se ha estado editando/corrigiendo alguno de sus campos.

Al igual que con el trigger After Insert, el código que programemos en un trigger After Update no podrá hacer cambios en la tabla que lo llama. Es lógico, si ya hemos estado editando en un registro de la tabla Bandejas, para qué activar una segunda edición del mismo registro. El que esta funcionalidad no sea necesaria en el trigger After Update pero sí hubiera sido útil en el trigger After Insert, hace que el primero sea más práctico que el segundo.

- Trigger Before Delete

Hemos realizado dos triggers Before Delete para preservar los datos de dos tablas:

- El trigger Before Delete **operarios_bd** se utiliza para preservar en variables globales *_baja* el valor de campos de un registro borrado en la tabla Operarios. A continuación se llama automáticamente al trigger After Delete **operarios_baja_ad** (que guardará los datos de las variables globales en un registro de la tabla Operarios_baja).

Editar disparador

Detalles

Nombre del disparador	operarios_bd
Tabla	operarios
Tiempo	BEFORE
Evento	DELETE

Definición

```
1 SET @DNIoper_baja=OLD.DNIoper,  
2   @cargoooper_baja=OLD.cargoooper,  
3   @nombreoper_baja=OLD.nombreoper,  
4   @telefooper_baja=OLD.telefooper,  
5   @dioper_baja=OLD.dioper,  
6   @localidadoper_baja=OLD.localidadoper,  
7   @provinciaoper_baja=OLD.provinciaoper
```

Definidor

root@localhost

- El trigger Before Delete **operarios_baja_bd** se utiliza para preservar en variables globales *_baja* el valor de campos de un registro borrado en la tabla Operarios_baja. A continuación se llama automáticamente al trigger After Delete **operarios_ad** (que guardará los datos de las variables globales en un registro de la tabla Operarios).

Editar disparador

Detalles

Nombre del disparador

operarios_baja_bd

Tabla

operarios_baja

Tiempo

BEFORE

Evento

DELETE

Definición

1 SET @DNIoper=OLD.DNIoper_baja,

2 @cargoooper=OLD.cargoooper_baja,

3 @nombreoper=OLD.nombreoper_baja,

4 @telefooper=OLD.telefooper_baja,

5 @dirooper=OLD.dirooper_baja,

6 @localidadoper=OLD.localidadoper_baja,

7 @provinciaoper=OLD.provinciaoper_baja

Definidor

root@localhost

- Trigger After Delete

Hemos realizado dos triggers Before Delete para guardar en dos Tablas los datos preservados en variables globales:

- Gracias al trigger Before Delete **operarios_bd** habíamos preservado en variables globales el valor de campos de un registro borrado en la tabla Operarios. El trigger After Delete **operarios_baja_ad** se utiliza para insertar/guardar los valores de las variables globales en campos de un registro de la tabla Operarios_baja, poniendo de baja al operario.

Editar disparador

Detalles

Nombre del disparador	operarios_baja_ad
Tabla	operarios_baja
Tiempo	AFTER
Evento	DELETE

Definición

```
1 INSERT INTO operarios
  (DNIoper,cargoooper,nombreoper,telefooper,diroper,localidadoper,provinciaoper)
2 VALUES(@DNIoper,@cargoooper,@nombreoper,@telefooper,@diroper,@localidadoper,
  @provinciaoper)
```

Definidor root@localhost

- Gracias al trigger Before Delete **operarios_baja_bd** habíamos preservado en variables globales el valor de campos de un registro borrado en la tabla Operarios_baja. El trigger After Delete **operarios_ad** se utiliza para insertar/guardar los valores de las variables globales en campos de un registro de la tabla Operarios, poniendo de alta al operario nuevamente.

Editar disparador

Detalles

Nombre del disparador

operarios_ad

Tabla

operarios

Tiempo

AFTER

Evento

DELETE

Definición

```
1 INSERT INTO operarios_baja
(DNIoper_baja,cargooper_baja,nombreoper_baja,telefooper_baja,diooper_baja,l
ocalidadoper_baja,provinciaoper_baja)
2 VALUES(@DNIoper_baja,@cargooper_baja,@nombreoper_baja,@telefooper_baja,@dir
oper_baja,@localidadoper_baja,@provinciaoper_baja)
```

Definidor

root@localhost

PROCEDIMIENTOS

Hemos implementado tres Procedimientos:

- Procedimiento **actualizar_porte_actual_Plantas** :

Como su nombre indica, está pensado para que un operario pueda actualizar en la tabla **Plantas** el valor del **porte_actual** (que será igual a *nuevo_porte*) de una planta **idplanta** (que será igual a *n_idplanta*) situada en una bandeja **idbandeja** (que será igual a *n_idbandeja*) .

Editar rutina

Detalles

Nombre de rutina

actualizar_porte_actual_Plantas

Tipo

PROCEDURE

Parámetros

	Dirección	Nombre	Tipo	Longitud/Valores	Opciones	
↑	IN	nuevo_porte	INT	10	UNSIGNED	Eliminar
↑	IN	n_idplanta	INT	1	UNSIGNED	Eliminar
↑	IN	n_idband	CHAR	6	utf8mb4	Eliminar

Definición

Agregar parámetro

```
1 UPDATE Plantas SET
2
3 Plantas.porte_actual=nuevo_porte WHERE ( Plantas.idplanta=n_idplanta AND Plantas.idband=n_idband)
```

Es determinístico

☐

Ajustar privilegios

☒

Definidor

`root`@`localhost`

Tipo de seguridad

DEFINER

Acceso de datos SQL

NO SQL

Comentario

- Procedimiento **actualizar_estado_tarea_Operaciones**:

Como su nombre indica, está pensado para que un operario pueda actualizar en la tabla Operaciones el valor del **estado_tarea** (que será igual a *nuevo_estado*) de una operación **idoperacion** (que será igual a *n_idoperacion*) llevada a cabo por el operario con identificador **DNloper** (que será igual a *n_DNloper*) .

A mayores:

Para el caso cuando el valor introducido *nuevo_estado* = 'Haciendo', entonces se actualiza la fecha **Inicio** al valor del CURRENT_TIME.

Para el caso cuando el valor introducido *nuevo_estado* = 'Hecha', entonces se actualiza la fecha **Final** al valor del CURRENT_TIME.

Editar rutina

Detalles

Nombre de rutina

actualizar_estado_tarea_Operaciones

Tipo

PROCEDURE

Parámetros

Dirección	Nombre	Tipo	Longitud/Valores	Opciones
IN	nuevo_estado	VARCHAR	20	utf8mb4 Eliminar
IN	n_idoperacion	INT	10	UNSIGNED Eliminar
IN	n_DNloper	CHAR	9	utf8mb4 Eliminar

Agregar parámetro

Definición

```

1 BEGIN
2   SET @nuevoestado=nuevo_estado;
3
4   UPDATE Operaciones
5   SET
6     Operaciones.estado_tarea=nuevo_estado,
7     Operaciones.Inicio=CASE WHEN @nuevoestado = 'Haciendo' THEN CURRENT_TIME ELSE
8     Operaciones.Inicio END,
9     Operaciones.Final=CASE WHEN @nuevoestado = 'Hecha' THEN CURRENT_TIME ELSE Operaciones.Final
10  END
11  WHERE ( Operaciones.idoperacion=n_idoperacion AND Operaciones.DNloper=n_DNloper);
12 END

```

Es determinístico

☐

Ajustar privilegios

☒

Definidor

`root`@`localhost`

Tipo de seguridad

DEFINER

Acceso de datos SQL

NO SQL

Comentario

- Procedimiento **actualizar_estadoband_Bandejas**:

Como su nombre indica, está pensado para que un operario pueda actualizar en la tabla **Bandejas** el valor del **estadoband** (que será igual a **nuevo_estadoband**) de una bandeja **idband** (que será igual a **n_idband**) que tiene plantada la especie **idespecie** (que será igual a **n_idespecie**).

A mayores:

Para el caso cuando el valor introducido **nuevo_estadoband** = 'Plantando', entonces se actualiza la **fechaPlantado** al valor del **CURRENT_TIME**.

Para el caso cuando el valor introducido **nuevo_estadoband** = 'Cosechando', entonces se actualiza la **fechaCosechado** al valor del **CURRENT_TIME**.

Editar rutina

Detalles

Nombre de rutina
actualizar_estadoband_Bandejas

Tipo
PROCEDURE

Parámetros

Dirección	Nombre	Tipo	Longitud/Valores	Opciones
IN	nuevo_estadoband	VARCHAR	40	utf8mb4 Eliminar
IN	n_idband	CHAR	6	utf8mb4 Eliminar
IN	n_idespecie	CHAR	5	utf8mb4 Eliminar

Agregar parámetro

Definición

```

1 BEGIN
2   SET @nuevoestado=nuevo_estadoband;
3
4   UPDATE Bandejas
5   SET
6     Bandejas.estadoband=nuevo_estadoband,
7     Bandejas.fechaPlantado=CASE WHEN @nuevoestado = 'Plantando' THEN CURRENT_TIME ELSE Bandejas.fechaPlantado
8   END,
9     Bandejas.fechaCosechado=CASE WHEN @nuevoestado = 'Cosechando' THEN CURRENT_TIME ELSE Bandejas.fechaCosechado
10  END
11  WHERE ( Bandejas.idband=n_idband AND Bandejas.idespecie=n_idespecie);
12 END

```

Es determinístico
☐

Ajustar privilegios
☒

Definidor
`root`@`localhost`

Tipo de seguridad
DEFINER

Acceso de datos SQL
NO SQL

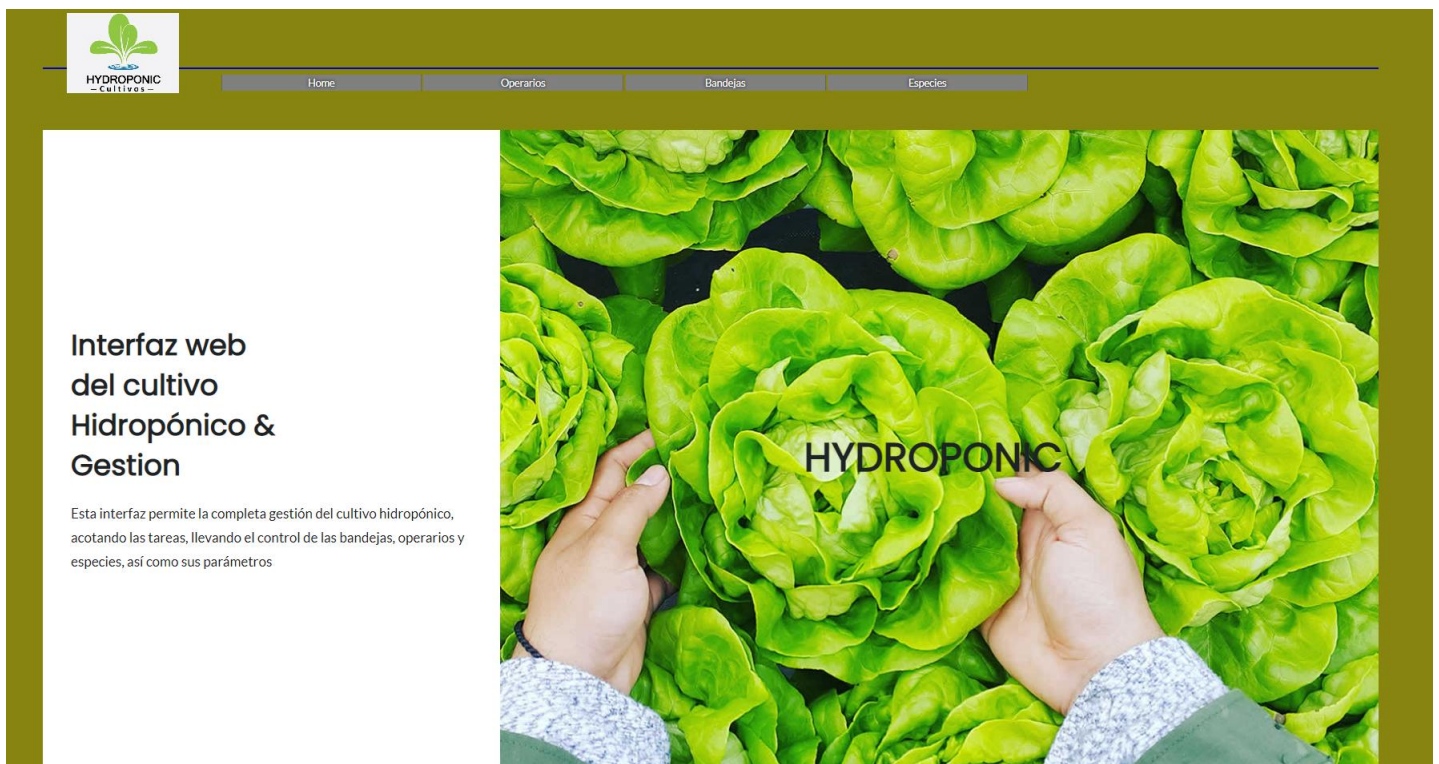
Comentario

Aplicación web de cultivos

De cara al uso de la base de datos por parte de un usuario, implementamos una aplicación web a modo de interfaz. Está diseñada principalmente en lenguajes PHP y HTML5, aunque también tiene CSS y Javascript en algunos archivos. A continuación presentamos cada una de las páginas de las que se compone la aplicación.

Home

Para la página principal **Index.php** se usó una plantilla que utiliza el framework Bootstrap, el cual facilita el diseño de sitios web, ya que incluye plantillas de tipografía, formularios... etc.



La principal ventaja de usar Bootstrap es que nos permite aplicar animaciones, en nuestro caso para hacer más visuales cosas como la transición de carga de la página, o la colocación de referencias de clientes a nuestra web (en la parte inferior). También permite hacer más atractiva la información dispuesta, en nuestro caso nos ayudó a elaborar de forma sencilla una disposición en bloques de texto e imágenes que alternan las posiciones izquierda y derecha.

Se decidió no usar animaciones Bootstrap en el resto de las páginas para que quedaran códigos más sencillos, dado que si las añadíamos, el servidor MySQL hacía que se cargaran muy lentamente las páginas. Incluir el framework Bootstrap también empeoraba el rendimiento de la descarga de datos a los listados.

Trabajo Final BDD

José Eduardo R. B. – Víctor Daniel D. M.

Para movernos a las páginas que contienen los listados de la base de datos, disponemos del siguiente menú situado en la parte superior de cada página.

[Home](#)[Operarios](#)[Bandejas](#)[Especies](#)

Comentaremos ahora un poco cada uno de los enlaces a los que nos redirigen las opciones del menú, y también otros botones de cada página:

Operarios

Cuando hacemos clic en el menú sobre Operarios, se carga la página **operarios_listado.php**

DNI	Cargo del operario	Nombre del operario	Teléfono del operario	Dirección del operario	Localidad Operario	Provincia del Operario
12345678A	técnico de sensores	Antonio Ferreras López	682 235 689	Ojo del Halcón 7, 8º A	Grijota	Palencia
13520623J	peón	Rosa Gómez García	657 892 345	Alcalde Pepe 6, 3ºD	Villaumbrales	Palencia
13530623J	Encargado	Juan López Gómez	689 237 585	Antonio Machado 40, 3º D	Santander	Cantabria
23569782X	peón	Pedro Trapero Sáiz	689 895 623	Felipe II 12, 1º A	Torrelavega	Cantabria
28569782X	peón	Ana Martín Martín	639 857 412	Ercilla 22, 1º A	Lugo	Lugo
43869750K	peón	Isabel Puig Sáiz	689 562 323	Camilo José Cela 40, 1º A	Unquera	Cantabria
93557223J	peón	Anabelle West González	628 818 838	Castillo 5, 4º Dcha	Valdeomillos	Palencia
93558223J	peón	Pepito Gómez Saiz	675 896 362	Juan Ramón Jiménez 45, 3º D	Villalobón	Palencia
93582223J	peón	John Hendrix Pérez	623 275 831	Plaza Mayor 5, 3º Dcha	Cabárceno	Cantabria

[Añadir nuevo operario](#)

Que muestra una lista de los operarios que haya registrados en la base de datos, ordenados por su DNI, y con otros datos personales de interés.

Si hacemos click sobre el botón inferior 'Añadir nuevo operario', se nos redirige a **operarios_insertar.php**.


Allí se nos permite rellenar los campos del nuevo operario.


The screenshot shows a web application interface for 'HYDROPONIC - Cultivos -'. It features a navigation bar with links to 'Home', 'Operarios', 'Bandejas', and 'Especies'. The main content area is titled 'Operarios: insertar' and contains a form with the following fields: 'DNI*' (with a placeholder '800000000A'), 'Cargo*', 'Nombre*', 'Teléfono*' (with a placeholder 'XXX XXX XXX'), 'Dirección*', 'Localidad*', and 'Provincia*'. Below the form are 'Insertar' and 'Cancelar' buttons. A small note at the bottom left states '* Campos obligatorios'.

La página limita los datos que puede introducir el usuario. Por ejemplo, el DNI tiene que tener necesariamente 8 números y acabar con una letra mayúscula. Esto está restringido así mediante un pattern definido en el input del DNI que programamos en código HTML5.


También el número de teléfono tiene que estar escrito en un formato espaciado cada tres dígitos: XXX XXX XXX

Si estamos conformes con los datos que queremos introducir, le damos al botón 'Insertar'. Si no, podemos pulsar 'Cancelar' (o pulsamos Operarios en el menú) y volveríamos a la página del listado de operarios.

En caso de que no estemos conformes con algún dato de un operario del listado, tenemos la opción de editar la información del operario pulsando el icono . Se nos redirigirá a la página **operarios_editar.php**, a la cual se le pasa el DNIOper del operario para que su información pueda ser tomada de la base de datos y visualizada en la página en unas cajitas input como las de la página de insertar (de hecho tienen los mismos formatos para introducir datos).




Una vez escribamos la información nueva para el operario, la operación de editar finaliza al hacer clic sobre 'Modificar'. Si la información nueva no incluye el DNI de otro operario ni incumple los formatos en que deben estar escritos el DNI y el teléfono, se nos devolverá un mensaje conforme la operación de modificación del operario se ha realizado con éxito.

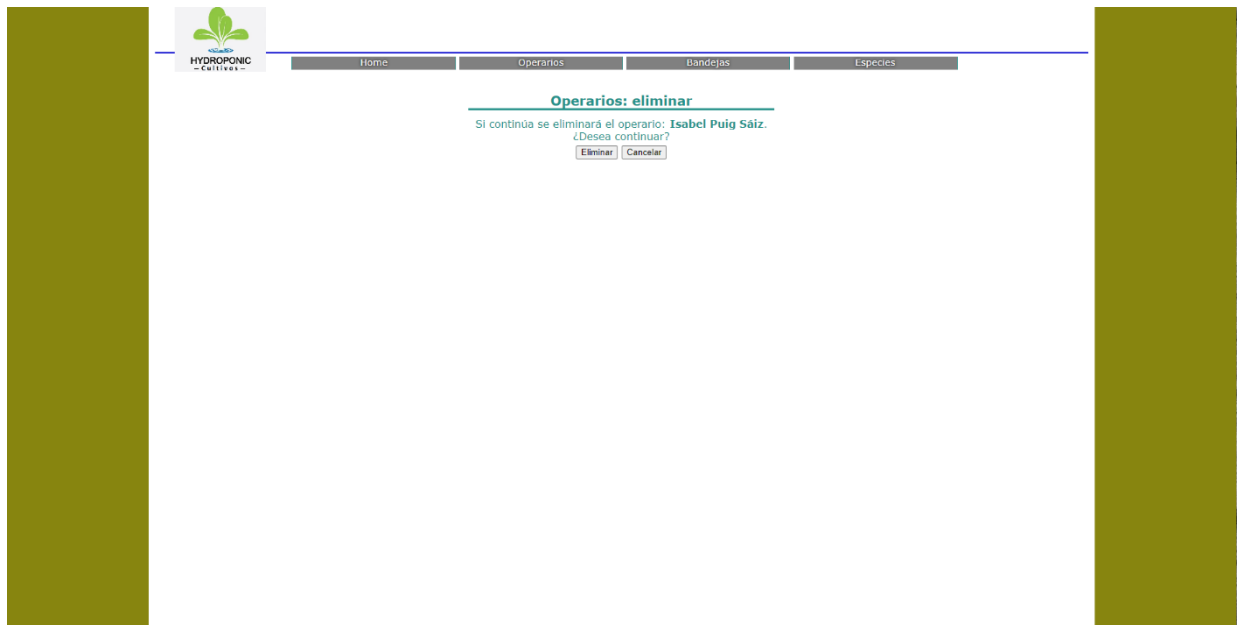


Pulsando sobre el botón 'Cancelar' o en el enlace Operarios del menú, volvemos al listado.

Trabajo Final BDD

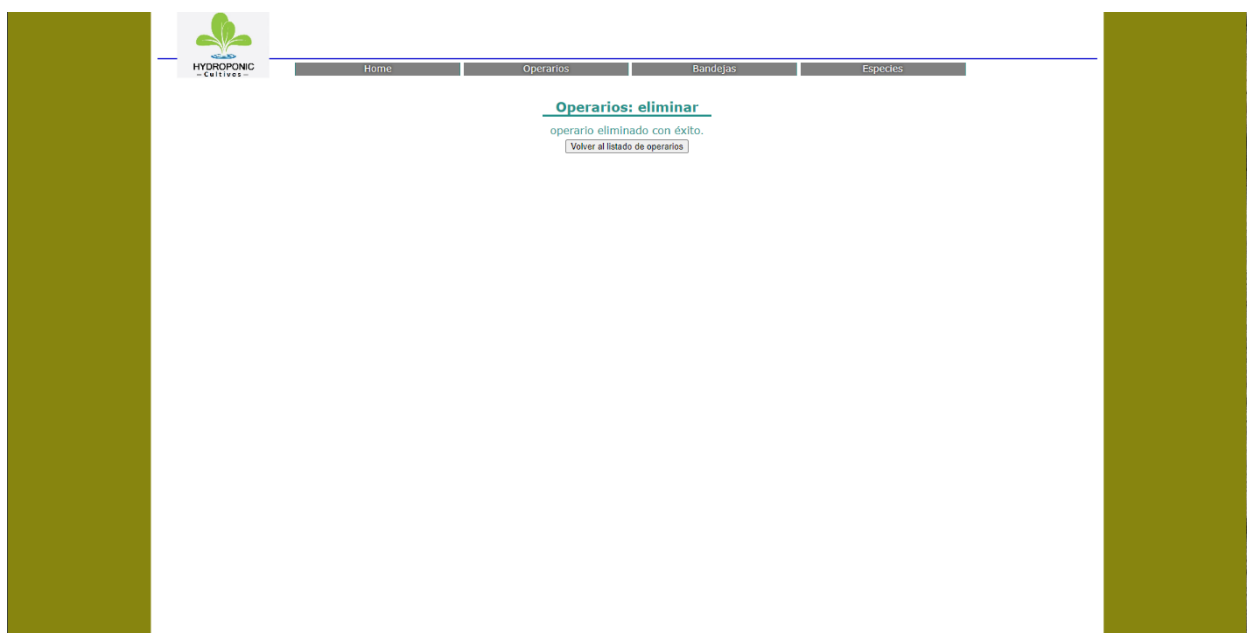
José Eduardo R. B. – Víctor Daniel D. M.

Si se desea eliminar un operario, hacemos clic sobre su correspondiente icono  que cada operario tiene en su margen derecho. Esto cargará la página **operarios_eliminar.php**, a la que se le habrá pasado el DNloper correspondiente para que pueda encontrar en la base de datos al operario que se desea eliminar. Una vez encontrado, en la página nos aparecerá un mensaje de consulta sobre si deseamos proceder con la eliminación.



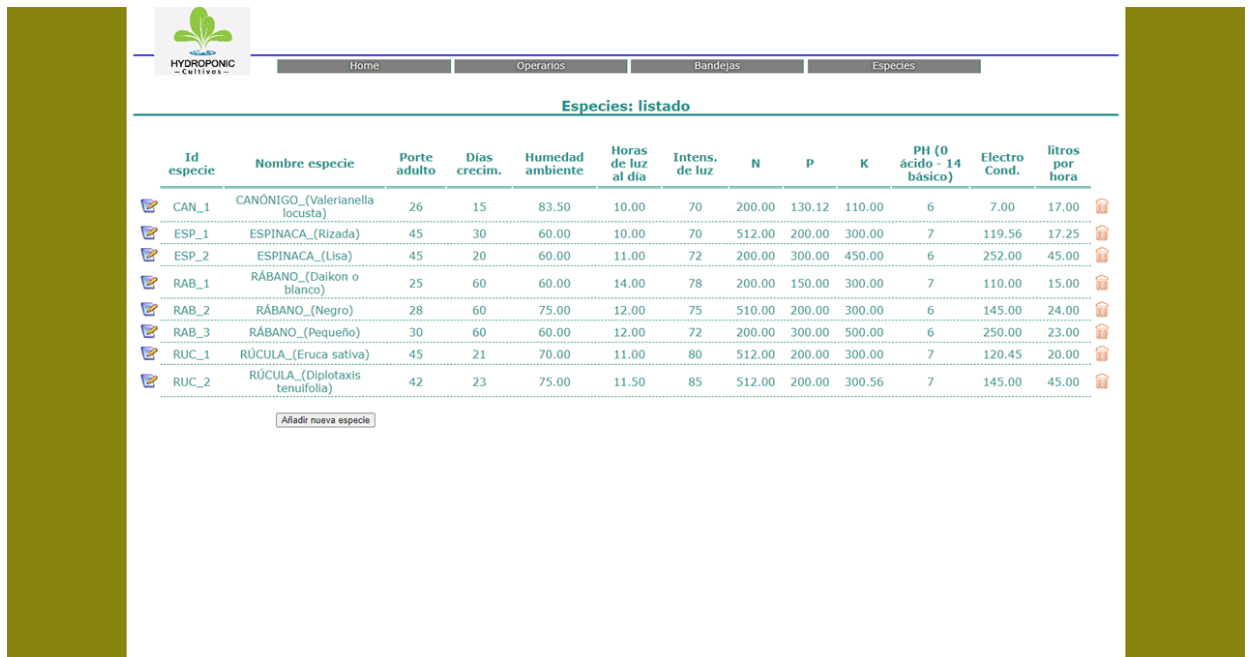
Si no estamos conformes con la eliminación podemos hacer click en el botón 'Cancelar' y volveríamos a la página del listado de operarios.

Pero si estamos conformes, pulsamos el botón 'Eliminar', y la operación termina con un mensaje que confirma que la eliminación fue exitosa.



Especies

Cuando hacemos clic en el menú sobre Especies, se carga la página **especies_listado.php**



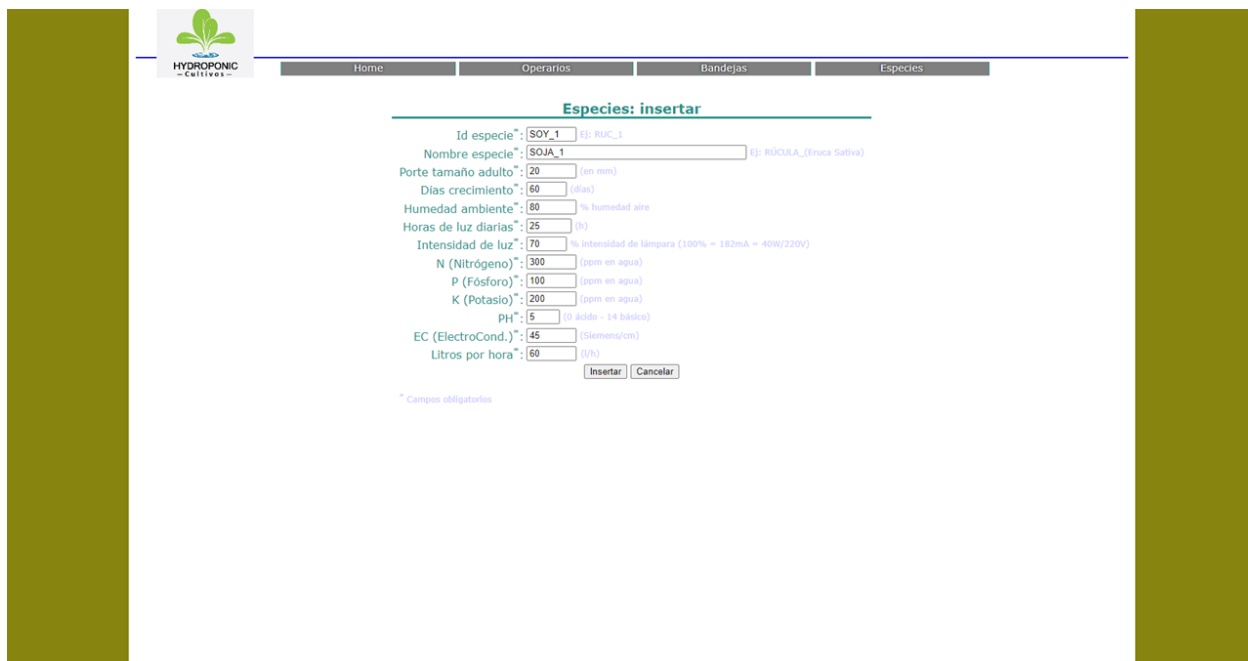
Id especie	Nombre especie	Porte adulto	Dias crecim.	Humedad ambiente	Horas de luz al día	Intens. de luz	N	P	K	PH (0 ácido - 14 básico)	Electro Cond.	litros por hora
CAN_1	CANÓNIGO_(Valerianella locusta)	26	15	83.50	10.00	70	200.00	130.12	110.00	6	7.00	17.00
ESP_1	ESPINACA_(Rizada)	45	30	60.00	10.00	70	512.00	200.00	300.00	7	119.56	17.25
ESP_2	ESPINACA_(Lisa)	45	20	60.00	11.00	72	200.00	300.00	450.00	6	252.00	45.00
RAB_1	RÁBANO_(Daikon o blanco)	25	60	60.00	14.00	78	200.00	150.00	300.00	7	110.00	15.00
RAB_2	RÁBANO_(Negro)	28	60	75.00	12.00	75	510.00	200.00	300.00	6	145.00	24.00
RAB_3	RÁBANO_(Pequeño)	30	60	60.00	12.00	72	200.00	300.00	500.00	6	250.00	23.00
RUC_1	RÚCULA_(Eruca sativa)	45	21	70.00	11.00	80	512.00	200.00	300.00	7	120.45	20.00
RUC_2	RÚCULA_(Diplotaxis tenuifolia)	42	23	75.00	11.50	85	512.00	200.00	300.56	7	145.00	45.00

[Añadir nueva especie](#)

Que muestra una lista de las especies que haya registradas en la base de datos, ordenadas por su Id especie, y con otros datos de interés.

Si hacemos clic sobre el botón inferior 'Añadir nueva especie', se nos redirige a **especies_insertar.php**.

Allí se nos permite rellenar los campos de una nueva especie.



Especies: insertar

Id especie*: Ej: RUC_1

Nombre especie*: Ej: RÚCULA_(Eruca Sativa)

Porte tamaño adulto*: (en mm)

Dias crecimiento*: (días)

Humedad ambiente*: % humedad aire

Horas de luz diarias*: (h)

Intensidad de luz*: % intensidad de lámpara (100% = 182mA = 40W/220V)

N (Nitrógeno)*: (ppm en agua)

P (Fósforo)*: (ppm en agua)

K (Potasio)*: (ppm en agua)

PH*: (0 ácido - 14 básico)


EC (ElectroCond.): (Siemens/cm)

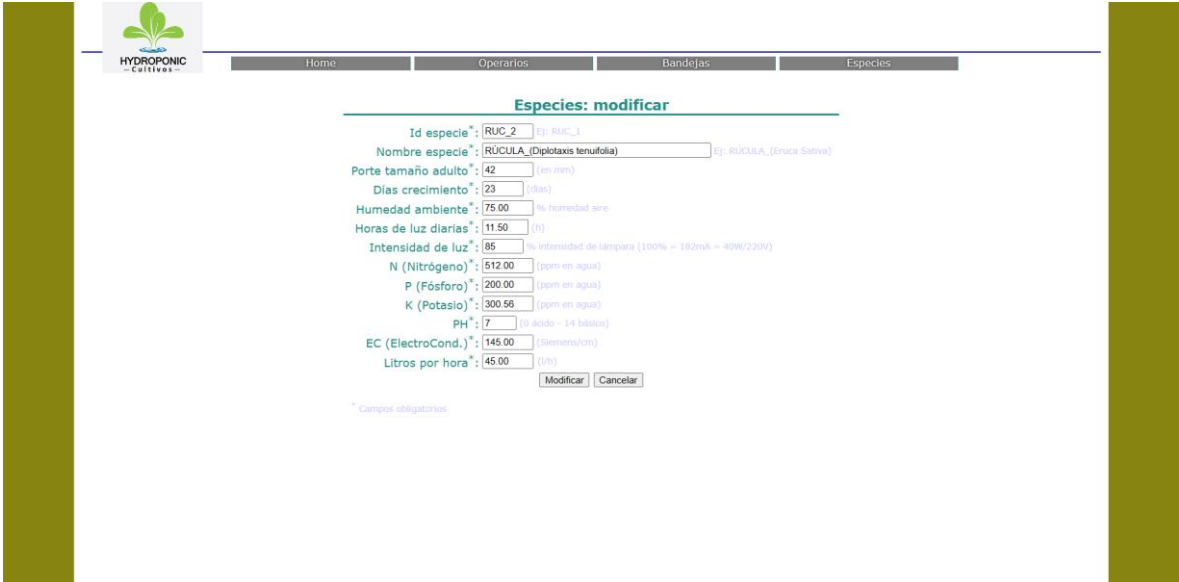
Litros por hora*: (l/h)

* Campos obligatorios


Esta página también tiene limitaciones a la hora de introducir los datos, ya que Id especie solo podrá insertarse si concuerda con el formato XXX_1, siendo los 3 primeros caracteres letras mayúsculas y el último un número; en cuanto al PH, este sólo podrá tener valores enteros entre 0 y 14.

Si estamos conformes con los datos que queremos introducir, le damos al botón 'Insertar'. Si no, podemos pulsar 'Cancelar' (o pulsamos Especies en el menú) y volveríamos a la página del listado de especies.

En caso de que no estemos conformes con algún dato de una especie del listado, tenemos la opción de editar la información de la especie pulsando el icono . Se nos redirigirá a la página **especies_editar.php**, a la cual se le pasa el id especie correspondiente para que la información de la especie seleccionada pueda ser tomada de la base de datos y visualizada en la página en unas cajitas input como las de la página de insertar.



Una vez escribamos la información nueva para la especie, la operación de editar finaliza al hacer clic sobre 'Modificar'. Si la información nueva no incluye el id de otra especie ni incumple el formato en que debe estar escrito el id, y también respeta el rango de 0 a 14 del PH; se nos devolverá un mensaje conforme la operación de modificación de la especie se ha realizado con éxito.



[Home](#)
[Operarios](#)
[Bandejas](#)
[Especies](#)

Especie modificada con éxito.

Especies: modificar

Id especie*: Ej: RUC_1

Nombre especie*: Ej: RÚCULA_(Eruca Sativa)

Porte tamaño adulto*: (en mm)

Días crecimiento*: (días)

Humedad ambiente*: % humedad aire

Horas de luz diarias*: (h)

Intensidad de luz*: % intensidad de lámpara (100% = 182mA = 40W/220V)

N (Nitrógeno)*: (ppm en agua)

P (Fósforo)*: (ppm en agua)

K (Potasio)*: (ppm en agua)


PH*: (0 ácido - 14 básico)


EC (ElectroCond.)*: (Siemens/cm)

Litros por hora*: (l/h)

* Campos obligatorios

Pulsando sobre el botón 'Cancelar' o en el enlace Especies del menú, volvemos al listado.

Si se desea eliminar una especie, hacemos clic sobre su correspondiente icono  que cada especie tiene en su margen derecho. Esto cargará la página **especies_eliminar.php**, a la que se le habrá pasado el id especie correspondiente para que pueda encontrar en la base de datos a la especie que se desea eliminar. Una vez encontrada, en la página nos aparecerá un mensaje de consulta sobre si deseamos proceder con la eliminación.



[Home](#)
[Operarios](#)
[Bandejas](#)
[Especies](#)

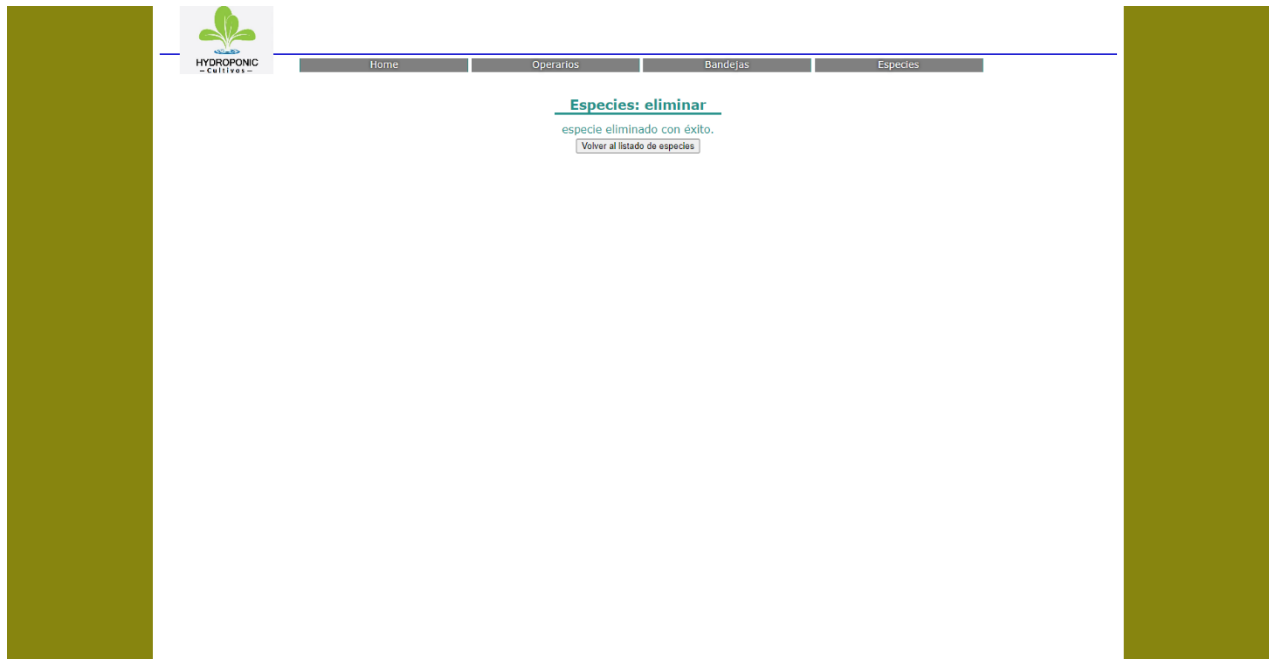
Especies: eliminar

Si continúa se eliminará la especie: **SOJA_(soja Monsanto 1)**.

¿Desea continuar?

Si no estamos conformes con la eliminación podemos hacer clic en el botón 'Cancelar' y volveríamos a la página del listado de especies.

Pero si estamos conformes, pulsamos el botón 'Eliminar', y la operación termina con un mensaje que confirma que la eliminación fue exitosa.



Bandejas

Cuando hacemos clic en el menú sobre Bandejas, se carga la página **bandejas_listado.php**

The screenshot shows the 'Bandejas: listado' page. It features a table with 7 columns: id de la bandeja, torre, altura, id de la especie, estado de la bandeja, fecha de plantado, and fecha de cosechado. Each row represents a tray and includes icons for editing and deleting. The table lists 20 trays with various statuses like 'requiriendo_Plantar', 'Plantando', 'Normal', 'requiriendo_QuitarHierbas', 'QuitandoHierbas', 'requiriendo_MedirPorte', 'MidiendoPorte', 'requiriendo_Cosechar', and 'Cosechando'.

id de la bandeja	torre	altura	id de la especie	estado de la bandeja	fecha de plantado	fecha de cosechado
001-01	1	1	CAN_1	requiriendo_Plantar (bandeja vacía)		
001-02	1	2	CAN_1	requiriendo_Plantar (bandeja vacía)		
001-03	1	3	CAN_1	requiriendo_Plantar (bandeja vacía)		
001-04	1	4	CAN_1	Plantando	2022-11-03	
001-05	1	5	CAN_1	Normal	2022-11-03	
002-01	2	1	ESP_1	requiriendo_QuitarHierbas	2022-10-24	
002-02	2	2	ESP_1	requiriendo_QuitarHierbas	2022-10-24	
002-03	2	3	ESP_1	requiriendo_QuitarHierbas	2022-10-24	
002-04	2	4	ESP_1	QuitandoHierbas	2022-10-24	
002-05	2	5	ESP_1	Normal	2022-10-24	
003-01	3	1	ESP_2	requiriendo_MedirPorte	2022-10-17	
003-02	3	2	ESP_2	requiriendo_MedirPorte	2022-10-17	
003-03	3	3	ESP_2	requiriendo_MedirPorte	2022-10-17	
003-04	3	4	ESP_2	MidiendoPorte	2022-10-17	
003-05	3	5	ESP_2	Normal	2022-10-17	
004-01	4	1	RAB_1	requiriendo_Cosechar	2022-09-03	
004-02	4	2	RAB_1	requiriendo_Cosechar	2022-09-03	
004-03	4	3	RAB_1	requiriendo_Cosechar	2022-09-03	
004-04	4	4	RAB_1	Cosechando	2022-09-03	2022-11-03

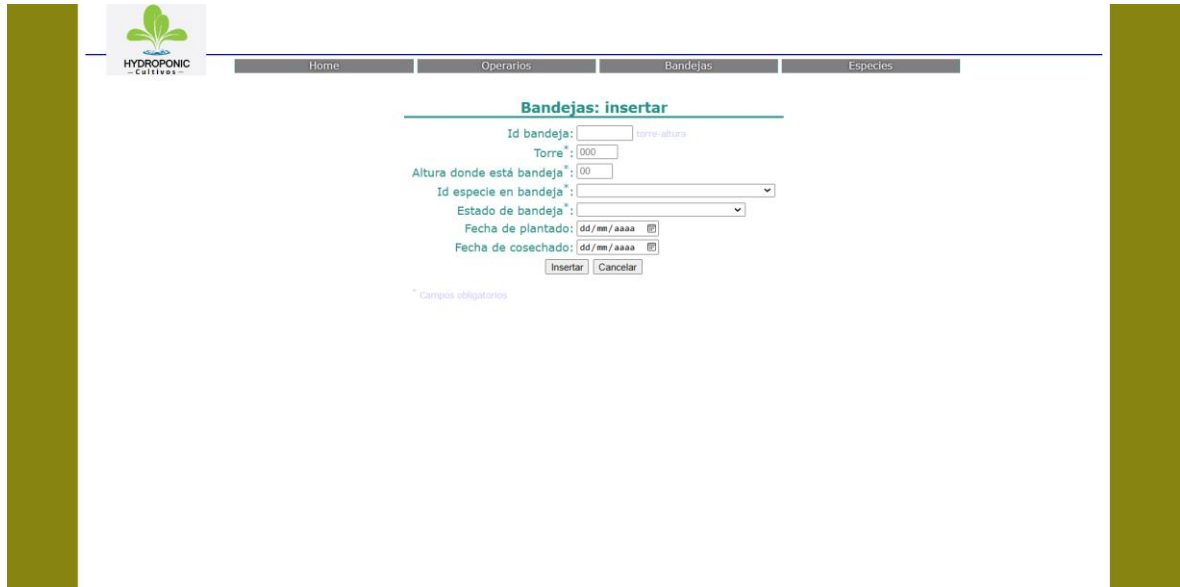
Que muestra una lista de las bandejas que haya registradas en la base de datos, ordenadas por su Id bandeja, y con otros datos de interés.

Trabajo Final BDD

José Eduardo R. B. – Víctor Daniel D. M.

Si hacemos clic sobre el botón inferior ‘Añadir nueva bandeja’, se nos redirige a **bandejas_insertar.php**.


Allí se nos permite rellenar los campos de una nueva bandeja.



The screenshot shows a web application interface for 'HYDROPONIC Cultivos'. At the top, there is a navigation bar with links: Home, Operarios, Bandejas, and Especies. The main content area is titled 'Bandejas: insertar'. It contains a form with the following fields: 'Id bandeja:' (disabled), 'Torre*' (text input, value: 000), 'Altura donde está bandeja*' (text input, value: 00), 'Id especie en bandeja*' (dropdown menu), 'Estado de bandeja*' (dropdown menu), 'Fecha de plantado:' (date picker, format: dd/mm/aaaa), and 'Fecha de cosechado:' (date picker, format: dd/mm/aaaa). There are 'Insertar' and 'Cancelar' buttons at the bottom. A small note at the bottom left says '* Campos obligatorios'.

Esta página presenta tiene como peculiaridad que el Id de bandeja no puede ser rellenado manualmente, pues se crea automáticamente a partir de los valores introducidos en los campos Torre y Altura. Además, cuando empezamos a rellenar los campos Torre y Altura, se nos avisa del número de caracteres que llevamos introducidos y de que deben ser 3 dígitos para Torre y 2 dígitos para Altura.

Si estamos conformes con los datos que queremos introducir, le damos al botón ‘Insertar’. Si no, podemos pulsar ‘Cancelar’ (o pulsamos Bandejas en el menú) y volveríamos a la página del listado de bandejas.

En caso de que no estemos conformes con algún dato de una bandeja del listado, tenemos la opción de editar la información de la bandeja pulsando el icono . Se nos redirigirá a la página **bandejas_editar.php**, a la cual se le pasa el id bandeja correspondiente para que la información de la bandeja pueda ser tomada de la base de datos y visualizada en la página en unas cajitas input como las de la página de insertar. Sin embargo, como veremos a continuación, también se visualizan dos listados con subdatos relativos a la bandeja: un listado de *Plantas* y otro de *Operaciones*.

Bandejas: modificar

Id bandeja: 004-05 torre-altura

Torre: 4

Altura donde está bandeja: 5

Id especie en bandeja: RAB_1, RABANO (Daikon o blanco)

Estado de bandeja: Normal

Fecha de plantado: 03/09/2022

Fecha de cosechado: 03/11/2022

* Campos obligatorios

Plantas

Nº planta	Tamaño actual	Tamaño adulto (especie)	Días crecimiento (especie)
1	27 mm	25 mm	60 días
2	27 mm	25 mm	60 días
3	27 mm	25 mm	60 días
4	27 mm	25 mm	60 días
5	27 mm	25 mm	60 días
6	27 mm	25 mm	60 días
+			

Operaciones

Nº operación	Operación	Operario	Fecha a realizar operación	Estado	Tiempo realización operación	Coste materiales necesarios
20	D) cosechar	135306231	2022-11-03 00:00:00	Hecha	13 min.	0.05 €
405	E) vaciar y limpiar	935822231	2022-11-04 00:00:00	Por_hacer	min.	0.10 €

La página se divide visualmente en tres áreas diferentes:

- Área superior izquierda: donde se modifican los campos de la bandeja cargada.
- Área superior derecha: donde se visualizan las plantas que tiene actualmente la bandeja.
- Área inferior: donde se visualizan las operaciones asociadas a la bandeja.

En el área superior izquierda, se cargan por defecto todos los campos de la bandeja a editar. Justo debajo de los campos, se encuentran dos botones: Modificar y Cancelar. Cualquier campo de bandeja que alteremos, deberá ser confirmado mediante una pulsación en el botón Modificar. Si pretendemos abandonar la edición y volver a la página bandejas_listado.php, bastaría con pulsar el botón Cancelar. También podemos salir de la página para editar pulsando alguno de los directorios del menú. Como peculiaridad, destacar que al igual que en bandejas_insertar.php, el id de bandeja se autorrellena automáticamente (gracias a un javascript) a partir de los valores que editemos en los campos Torre y Altura.


En el área superior derecha, se carga un listado de las plantas que crecen en la bandeja; y por si fuera necesario añadir una nueva planta, disponemos de un botón para insertar. Cada planta de la lista dispone de un botón de eliminar, por si se requiere eliminar rápidamente una planta, y un botón de editar para cuando se necesite actualizar la medida de tamaño actual. Al pulsar el botón Modificar se hace efectivo el cambio y se cargan también los valores nuevos en los inputs de edición, por si se desea seguir haciendo cambios en la planta. Si se quiere dejar de editar la planta, pulsamos el botón Cancelar.

En el área inferior, se carga un listado de las operaciones planificadas para la bandeja. Al igual que para las plantas, aquí también se dispone de un botón para insertar una nueva operación. Y cada operación de la lista tiene un botón para eliminar dicha operación, y otro para iniciar la edición de sus campos, que se confirma con el botón de Modificar, o se anula con Cancelar. Como peculiaridad, algunos inputs tienen asociado códigos javascript que guían al usuario durante los procesos de inserción o edición de una operación:

-El campo de Fecha a realizar operación, lanza un mensaje de aviso cuando la fecha introducida es la fecha actual, sugiriendo en el mensaje que la fecha debería ser al menos un día superior al de hoy para dar tiempo al operario a prepararse.

-El campo Estado, contiene un desplegable para elegir una de las tres opciones disponibles (Por_hacer, Haciendo, Hecha). Si se selecciona la opción Hecha sin haber elegido nunca la opción Haciendo, se lanza un mensaje de error que avisa de que primeramente se debería haber elegido la opción Haciendo, por lo menos alguna vez. Si se elige la opción Haciendo, se autorrellena el campo de Inicio con el valor de la fecha actual. Y si después se elige la opción Hecha, se autorrellena el campo Final con el valor de la fecha que esté en curso entonces. Por último, si decidiéramos que la operación tiene Estado a valor Por_hacer, los campos Inicio y Final se ponen automáticamente vacíos.

-Cuando los campos Inicio y Final estén rellenos con sus respectivas fechas (se les hayan pasado automáticamente o de forma manual), entonces se autorrellena el campo Tiempo realización operación con el valor de los minutos que hay de diferencia entre ambas fechas.

Al final de la página de bandejas_editar.php podemos ver que hay un botón para Imprimir informe . Tal como sugiere, si pulsamos el botón se abre una nueva pestaña en el navegador a la que se le pasa el id bandeja correspondiente, y que contiene un informe estructurado con todos los datos relevantes de la bandeja, sus plantas, y sus operaciones. Dicho informe está en formato PDF, listo para enviar a una impresora. A continuación enseñaremos un ejemplo de informe que se generaría para la bandeja 004-05 :



HYDROPONIC
- Cultivos -

Informe de la bandeja

Nº bandeja: 004-05
Torre: 4
Altura: 5
Estado bandeja: Normal
Fecha plantado: 2022-09-03
Fecha de Cosechado: 2022-11-03

	id planta	porte actual
Plantas:	1	27
	2	27
	3	27
	4	27
	5	27
	6	27

Especie


Id Especie: RAB_1 RÁBANO_(Daikon o blanco)
Porte adulto: 25
Días: 60
Humedad ambiente: 60.00
Horas de luz al día: 14.00
Intensidad de luz: 78 %
Nitrógeno (ppm): 200.00
Fósforo (ppm): 150.00
Nitrógeno (ppm): 300.00
PH: 7
Electroconductividad: 110.00
Litros por hora: 15.00

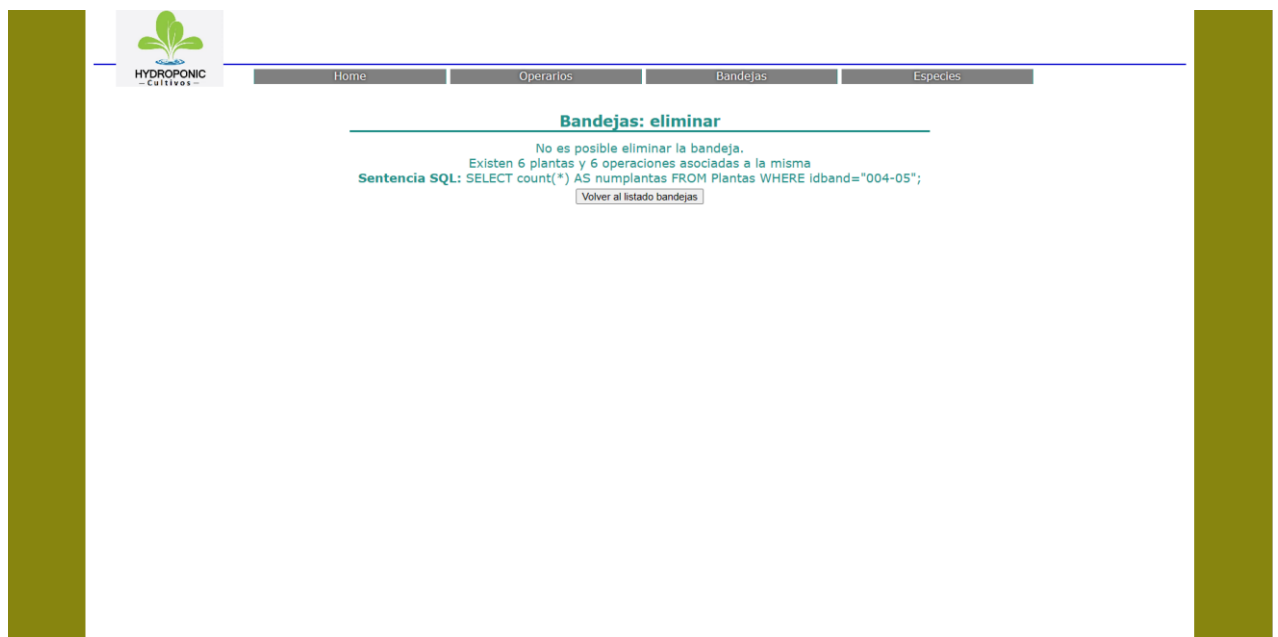
Operaciones:

id	DNloper	tarea	fecha operacion	estado	inicio	mins	final	costo
20	13530623J	D) cosechar	2022-11-03 00:00:00	Hecha	2022-11-03 00:00:00	13	2022-11-03 00:13:00	0.05eur
405	93582223J	E) vaciar y limpiar	2022-11-04 00:00:00	Por_hacer				0.10eur
410	12345678A	F) calibrar_sensores	2022-11-05 00:00:00	Por_hacer				0.20eur
415	93557223J	A) plantar	2022-11-06 00:00:00	Por_hacer				0.15eur
420	13530623J	B) quitar_hierbas	2022-12-16 00:00:00	Por_hacer				0.05eur
425	13530623J	C) medir_porte	2023-01-26 00:00:00	Por_hacer				0.00eur
					Total minutos:	13	Total costos:	0.55 €

Para realizar esta página se utilizó la librería fpdf17, la cual exporta a formato PDF texto y variables que manejemos en PHP.

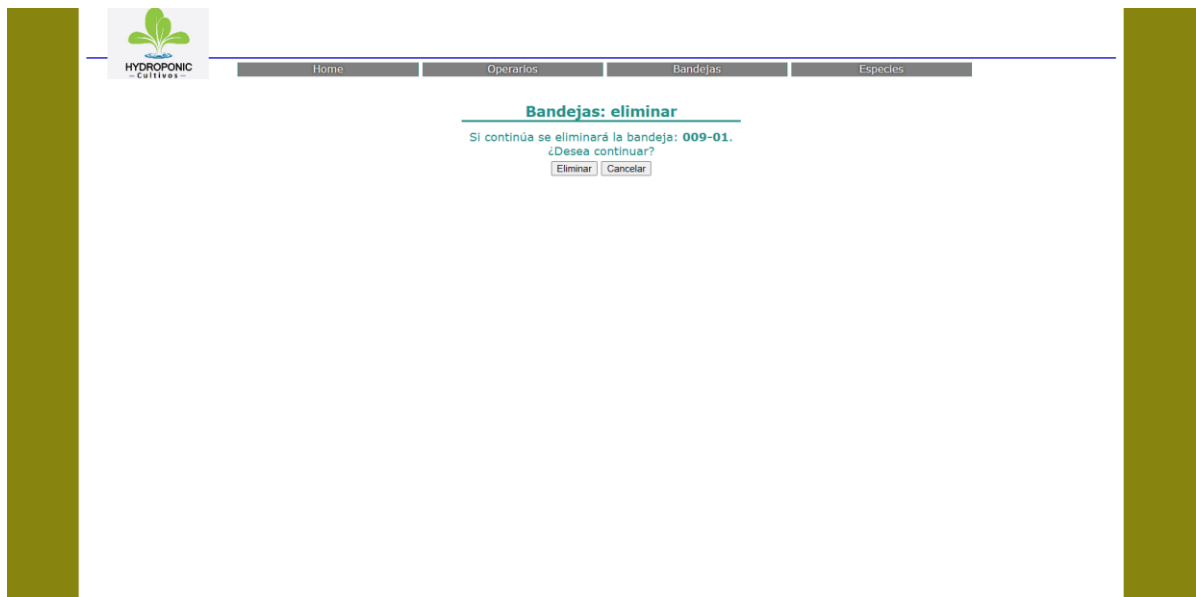
Para volver al listado de bandejas, pulsamos sobre el botón 'Cancelar' que hay bajo los campos de edición de una bandeja, o en el enlace Bandejas del menú.

Si se desea eliminar una bandeja, hacemos clic sobre su correspondiente icono  que cada bandeja tiene en su margen derecho. Esto cargará la página **bandejas_eliminar.php**, a la que se le habrá pasado el id bandeja correspondiente para que pueda encontrar en la base de datos a la bandeja que se desea eliminar. Una vez encontrada, en la página nos aparecerá un mensaje de aviso acerca del número de plantas y operaciones que haya en la bandeja. Por seguridad, para no perder la información de las plantas y las operaciones, se impide la eliminación de la bandeja a la que están asociadas.



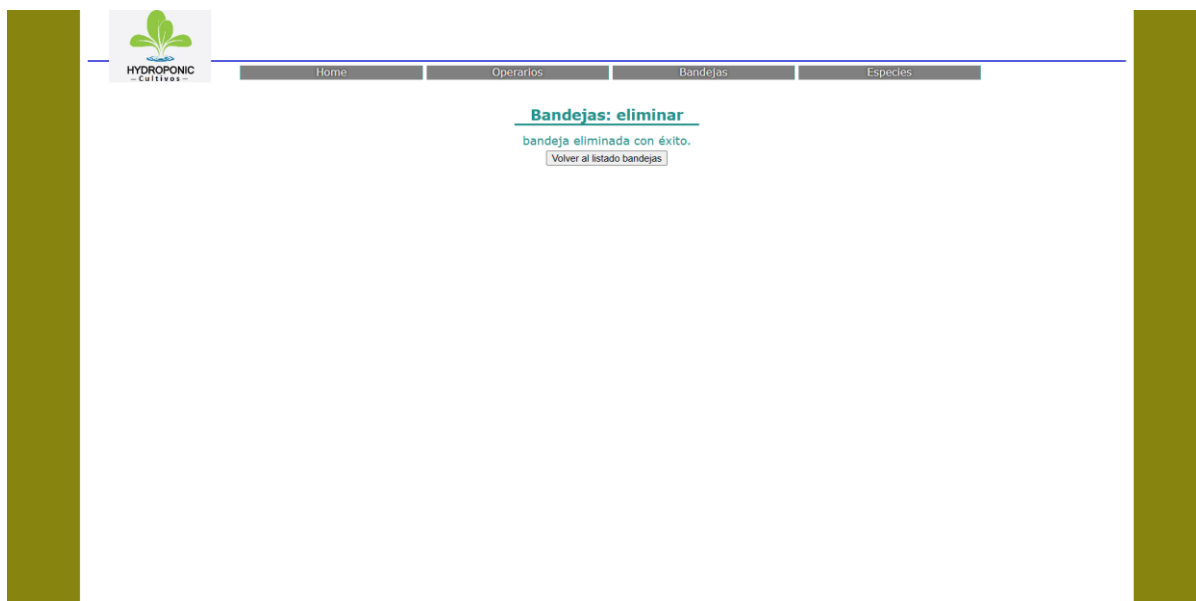
La única opción sería volver primero a la página de `bandejas_listado.php`, hacer clic en el icono de editar de la bandeja, y una vez se nos haya redirigido a la página de `bandejas_editar.php` pertinente, eliminar cada planta y operación que haya en la página.

Después volveríamos al listado de bandejas, haríamos clic en el icono de eliminar de la bandeja, y ya se nos descargaría en la página de eliminar un mensaje de consulta sobre si deseamos proceder con la eliminación.



Si no estamos conformes con la eliminación podemos hacer clic en el botón 'Cancelar' y volveríamos a la página del listado de bandejas.

Pero si estamos conformes, pulsamos el botón 'Eliminar', y la operación termina con un mensaje que confirma que la eliminación fue exitosa.



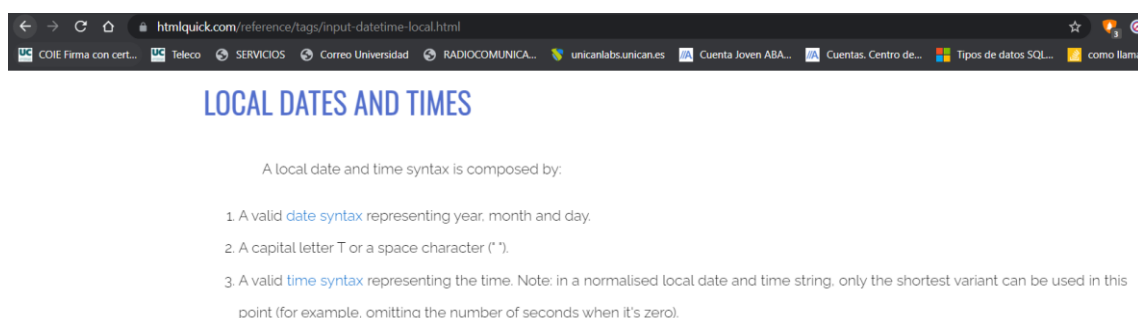
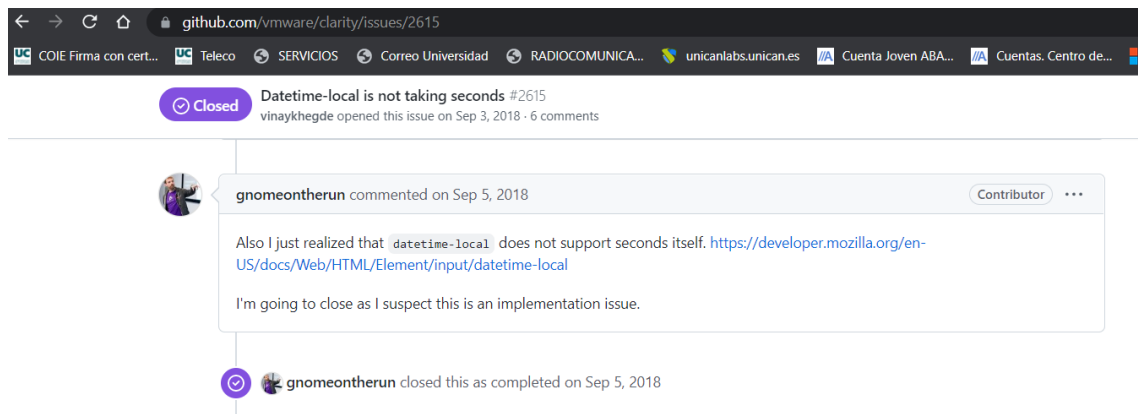
Problemas encontrados durante la realización de la aplicación web

El único problema serio que tiene nuestra aplicación es que en la página de `bandejas_editar.php`, en el listado de Operaciones, los inputs de las fechas (tipo `datetime`) no son capaces de capturar el valor 00 en el campo de los segundos. Irónicamente sí capturan otros valores de segundos (01, 02, ..., 59). En cambio sí pueden representar visualmente todos los segundos añadiendo `step=1` en la definición del input (por defecto está programado `step=60` segundos para que se visualice sólo hasta los minutos).

Tras mucho tiempo de pelear con el código y de buscar soluciones en internet, nos topamos con que el fallo no se debe a que nosotros hayamos hecho algo mal. El problema se debe a un fallo de diseño del input tipo `datetime`, que fue creado así por un grupo desarrollador de Mozilla, y que pensaron que si el valor de los segundos era 00 era mejor considerar que no había segundos y representar sólo hasta los minutos. Esto plantea un serio problema, pues los segundos 00 existen (marcan los segundos en punto) y no son lo mismo que segundos NULL.

Aparentemente, parece que podríamos haber tomado la decisión radical de representar nuestras fechas `datetime` hasta los minutos y así capear el problema. Sin embargo, el formato de fechas `datetime` de la base de datos SQL es tozudo y requiere un rango válido: "1000-01-01 00:00:00" a "9999-12-31 23:59:59", es decir, con segundos incluidos.

A continuación adjuntamos algunos comentarios de programadores que han tenido que enfrentar el mismo problema que hemos conocido nosotros:



Trabajo Final BDD

José Eduardo R. B. – Víctor Daniel D. M.

<https://developer.mozilla.org/en-US/docs/Web/HTML/Element/input/datetime-local>

Handling browser support

As mentioned above, non-supporting browsers gracefully degrade to a text input, but this creates problems both in terms of consistency of user interface (the presented control will be different), and data handling.

The second problem is the most serious; as we mentioned earlier, with a datetime-local input, the actual value is always normalized to the format YYYY-MM-DDThh:mm.