Universitat Oberta de Catalunya
uoc.edu

UNIVERSIDAD OBERTA DE CATALUNYA (UOC)

WWW.UOC.EDU

GESTIÓN DE GRANJAS PORCINAS: SALUPIG

TRABAJO DE FIN DE GRADO

GRADO DE INGENIERÍA INFORMÁTICA

(JEE)

Autor: JAVIER RODRÍGUEZ ALBARRÁN

Consultor: VICENÇ FONT SAGRISTA

PRA: SANTI CABALLE LLOBET

JUNIO, 2018

Aplicación Web de Gestión de Granjas Porcinas

Javier Rodríguez Albarrán
Grado en Ingeniería Informática
JEE

Vicenç Font Sagrista
Santi Caballe Llobet

13 de Junio de 2018

FICHA DEL TRABAJO FINAL

Título del trabajo:	*SaluPig: Gestión de granjas*
Nombre del autor:	*Javier Rodríguez Albarrán*
Nombre del consultor/a:	*Vicenç Font Sagrista*
Nombre del PRA:	*Santi Caballe Llobet*
Fecha de entrega (mm/aaaa):	06/2018
Titulación::	*Grado Ingeniería Informática*
Área del Trabajo Final:	*Java EE*
Idioma del trabajo:	*Castellano*
Palabras clave	*Java, JavaEE, Patrones de diseño*

Resumen del Trabajo (máximo 250 palabras): *desarrollo de un sistema de información web para la gestión que realizan las granjas con las que trabaja nuestro cliente. Para ello, utilizando un modelo de desarrollo de software en cascada, pasaremos por las fases de análisis de requerimientos, diseño del sistema, implementación y pruebas, elaborando un producto que cumpla las necesidades del cliente final*

Abstract: develop of a web information system for the management of the farms, that our client usually work.For this reason, we will use a software development by cascade. We will go through the phases of requirement analysis, system design, implementation and testing, developing a product that meets the needs of our final client.

Índice

Lista de figuras

Lista de tablas

1. Introducción

1.1 Contexto y justificación del Trabajo

A raíz de un reciente programa de televisión en el que se denunciaban las condiciones de en las granjas de la industria cárnica se ha puesto de manifiesto la falta de control en este sector. El programa “Salvados”, emitido el día cuatro de febrero de 2018, puso en jaque el mercado cárnico porcino (uno de los principales en nuestra industria), poniendo en duda la sostenibilidad de dicho sector, siendo altamente perjudicial tanto para la salud pública, así como para los animales que se crían en ella y sus trabajadores.

Tras el revuelo generado a raíz de la emisión de unas imágenes en las que se podían apreciar cerdos en un estado lamentable de desnutrición, enfermos y hacinados en un espacio a la vista insuficiente, se ha puesto de manifiesto una falta de control total de las empresas cárnicas para con las granjas que crían a los animales que ellos venden.

Así, nuestro cliente “SaluPig”, que está empezando en el sector, nos ha solicitado una aplicación web para poder controlar determinada información de cada una de las granjas con las que trabajan: número de animales, metros cuadrados de los que disponen, número de animales enfermos, número de trabajadores, bajas laborales acumuladas, gasto en veterinarios, número de revisiones por parte de profesionales, gasto en alimentación, etc.

Su objetivo es poder analizar toda esta información de forma sencilla, clara y concisa, para poder tomar decisiones oportunas y evitar llegar a la situación que ha denunciado el programa de televisión Salvados. De esta forma, podrán tomar decisiones que garanticen que el producto que les llega sea adecuado para la venta. Como añadido, y entendiendo la importancia que tienen las personas que cuidan de los animales, también se pretende tener un control de las condiciones laborales de los mismos.

Para llevarlo a cabo, “SaluPig” quiere realizar revisiones periódicas de cada una de las granjas, almacenando información relevante en su sistema de información. El sistema debe estar preparado para realizar tantas revisiones como el cliente necesite y mantener un histórico de todas las revisiones realizadas ya que desea comprobar la evolución de las mismas. Además, se pretende que la aplicación sea accesible vía web (a través de alguno de los navegadores más utilizados actualmente: Internet Explorer, Mozilla Firefox, Chrome), de forma que cuando se hace una visita a una de las granjas, se pueda rellenar in situ la información de la misma pudiendo haber varias personas visitando varias granjas a la vez.

1.2 Objetivos del Trabajo

Dividiremos los objetivos del TFG en dos grandes grupos: objetivos personales y objetivos del cliente.

1.2.1 Objetivos Personales

- El principal objetivo es obvio: obtener la titulación de Grado en Ingeniería Informática para la que me he preparado.

- Poner en práctica los conceptos obtenidos a lo largo de las asignaturas que componen el grado.

- Actualizar en la medida de lo posible mis conocimientos acerca de JEE, con el uso de tecnologías más o menos actuales: JQuery, JPA sobre Hibernate para acceso a base de datos Postgres SQL, Spring Controller.

- Elaborar un producto software siguiendo el patrón arquitectónico en capas y el patrón modelo vista controlador. Deberemos completar las distintas fases que componen el mismo: análisis, diseño, programación y pruebas.

1.2.2 Objetivos del Cliente Final

- Disponer de un sistema de información que permita almacenar y explotar información acerca de cada una de las granjas con las que trabajan.

- Desean disponer de una aplicación sencilla, sin una excesiva navegación de pantallas ya que sus conocimientos son muy limitados y no se sienten muy cómodos trabajando con dispositivos electrónicos: tabletas, pc, móviles, etc.

- Debe ser una aplicación web ya que tienen que poder acceder con un navegador instalado en sus portátiles o tabletas.

1.3 Enfoque y método seguido

Una metodología es un conjunto de técnicas y herramientas que ayudan a las personas que forman parte de equipo de proyecto a desarrollar un producto software y establecen los pasos a seguir para la correcta ejecución del mismo.

En el mercado existen varias metodologías aplicables al desarrollo de productos software: Jackson, SSADM, METRICA, RAD, Scrum, Extrem Programming, AUG, etc. Para nuestro producto he seleccionado seguir los pasos que define Métrica. Se ha seleccionado esta metodología por su flexibilidad, su sencillez de uso y su adaptabilidad a distintos procesos y etapas. Además, al ser una metodología promovida por el Ministerio de Hacienda y Administraciones Públicas será de utilidad en mi trabajo actual.

Los principales pasos que seguiremos y que define esta metodología son:

1. Planificación.

Se establece el plan de tareas a seguir para conseguir los objetivos del proyecto en tiempo y forma. Se correspondería con las tareas a realizar en PEC1

2. Desarrollo.
   Recoge todas las tareas que se deben llevar a cabo para desarrollar la aplicación, cubriendo desde el análisis de requisitos hasta la instalación del software desarrollado. Se correspondería con las tareas de la PEC2 y 3. Los procesos definidos en Métrica están estrechamente relacionados con las distintas etapas del ciclo de vida y son:

   a. Análisis del sistema
   b. Diseño del sistema
   c. Construcción del sistema
   d. Implantación y aceptación

3. Mantenimiento del sistema (queda fuera del alcance de este proyecto)
   Cubre posibles peticiones de mantenimiento de los usuarios finales del producto

Cada metodología define en cierto modo su ciclo de vida. En este caso seguiremos un ciclo de vida convencional pero no por ello menos práctico: modelo en cascada con vuelta atrás. Con este enfoque, se pretende completar cada paso antes de empezar el siguiente con el mayor grado de exactitud.

La razón fundamental que ha llevado a la elección de este tipo de ciclo de vida consiste en la dificultad de conocer todos los requisitos del sistema y sus funcionalidades desde un primer momento, con lo cual es necesario un modelo que permita volver atrás sin una repercusión excesiva para el desarrollo del mismo. En la siguiente figura podemos apreciar el funcionamiento de nuestro ciclo de vida:

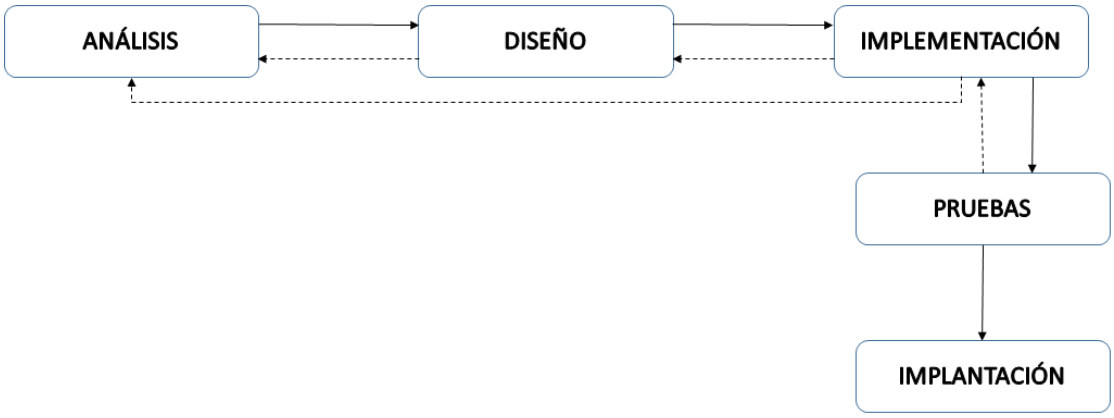


Ilustración 1: Ciclo de vida en cascada con vuelta atrás

1.4 Planificación del Trabajo

La planificación de tareas que forman parte del proyecto representadas mediante un diagrama de Gantt se puede apreciar en la Ilustración 1:

Diagrama de Gantt

	Nombre de tarea	Duración	Comienzo	Fin	Predecesoras
1	**TFG-Jave EE**	**81 días**	**mié 21/02/18**	**mié 13/06/18**	
2	**PEC1- Plan de trabajo**	**11 días**	**mié 21/02/18**	**mié 07/03/18**	
3	Requerimientos	7 días	mié 21/02/18	jue 01/03/18	
4	Planificación	2 días	vie 02/03/18	lun 05/03/18	3
5	Herramientos y frameworks	2 días	mar 06/03/18	mié 07/03/18	4
6	**PEC2- Análisis, Diseño**	**25 días**	**jue 08/03/18**	**mié 11/04/18**	**5**
7	**Análisis**	**18 días**	**jue 08/03/18**	**lun 02/04/18**	
8	Casos de uso y actores	5 días	jue 08/03/18	mié 14/03/18	5
9	Fichas de casos de uso	11 días	jue 15/03/18	jue 29/03/18	8
10	Prototipo	2 días	vie 30/03/18	lun 02/04/18	9
11	**Diseño**	**7 días**	**mar 03/04/18**	**mié 11/04/18**	
12	Diseño de base de datos	2 días	mar 03/04/18	mié 04/04/18	10
13	Diagrama de clases	3 días	jue 05/04/18	lun 09/04/18	12
14	Diagrama de arquitectura	2 días	mar 10/04/18	mié 11/04/18	13
15	**PEC3- Implementación**	**35 días**	**jue 12/04/18**	**mié 30/05/18**	
16	Instalación entorno de trabajo	1 día	jue 12/04/18	jue 12/04/18	14
17	Desarrollo de aplicación	25 días	vie 13/04/18	jue 17/05/18	16
18	Pruebas	7 días	vie 18/05/18	lun 28/05/18	17
19	Manual de usuario	1 día	mar 29/05/18	mar 29/05/18	18
20	Formación a usuarios	1 día	mié 30/05/18	mié 30/05/18	19
21	**Memoria y Presentación**	**10 días**	**jue 31/05/18**	**mié 13/06/18**	
22	Elaboración de memoria	6 días	jue 31/05/18	jue 07/06/18	20
23	Elaboración de presentación	3 días	vie 08/06/18	mar 12/06/18	22
24	Producto Final	1 día	mié 13/06/18	mié 13/06/18	23

Ilustración 2: Diagrama de Gantt

En dicho diagrama, se pueden apreciar todas las tareas que se han tenido en cuenta, su duración en días, fechas inicial y final estimadas y la actividad predecesora.

Destacar que en muchos casos serán necesarias planificaciones intermedias y revisiones debidas posibles fallos en la misma planificación y, muy posiblemente, cambios solicitados por el cliente a medida que se avanza en el desarrollo del producto.

Las fechas clave del desarrollo del proyecto serán las siguientes:

Hitos destacados	
Inicio	21/02/2018
PEC1: Plan de trabajo	21/02/2018 - 07/03/2018
PEC2: Análisis y diseño	08/03/2018 - 11/04/2018
PEC3: Implementación	12/04/2018 - 30/05/2018
Memoria y Presentación	31/05/2018 - 13/06/2018
Finalización	14/06/2018

Tabla 1. Hitos destacados

Para conseguir los objetivos descritos en el punto anterior siguiendo la planificación detallada necesitamos el siguiente software:

1. **Base de datos: PostgreSQL v. 10.2-1**

   Base de datos relacional distribuida bajo licencia BSD. Se trata de una base de datos estable, potente, robusta y fácil de administrar, lo que hace que sea una buena elección para el almacenamiento de nuestros datos

2. **Persistencia: Hibernate 4.3**

   Framework que gestiona la capa de persistencia a través de ficheros XML o anotaciones. Implementa, entre otras cosas, la especificación JPA (Java Persistence API), estándar de Java (JSR 220)

3. **Negocio: Servlets 4.0.6**

   La implementación de la lógica de negocio se llevará a cabo mediante la especificación java servlets, que recogerá las peticiones realizadas por el cliente, enviando o solicitando datos a la base de datos y respondiendo al cliente a su petición.

4. **Vista: JSP/JQuery**

   Mediante el uso de Java Server Pages y la biblioteca de clases JQuery, generaremos la vista de la aplicación. Además, nos ayudaremos de las librerías proporcionadas por w2ui (http://w2ui.com/web/). Estas librerías,

basadas en JQuery, proporcionan funciones para el desarrollo de aplicaciones front-end donde existe intercambio de datos.

5. **Controlador: Spring 4.3**

La capa controller se implementará utilizando el framework de Spring, ampliamente conocido en el desarrollo de software

6. **Servidor de aplicaciones: Apache Tomcat 9.0.5**

El hecho de no utilizar EJB nos permite utilizar como servidor de aplicaciones Apache Tomcat. Es un servidor ampliamente utilizado en el mercado tanto por su sencillez como su robustez. Se trabajará con la última versión que hay en el mercado.

7. **Gestión de proyectos: Maven 3.2**

Es una herramienta de software para la gestión y construcción de proyectos Java ampliamente utilizada en el desarrollo de software y proporcionada por Apache Software Foundation. Facilita la construcción del software, sus dependencias de otros módulos y componentes del mismo, y el orden de construcción de los elementos.

8. **IDE: Eclipse Oxygen**

Se utilizará como entorno de desarrollo Eclipse Oxygen, siendo uno de los principales IDE existentes en el mercado. Será muy útil la utilización de plugins de código libre que existen.

9. **Desarrollo: Java 1.8**

El desarrollo de las clases Java se hará utilizando la versión 1.8 para 64bits. Se trata de la última versión que existe en la actualidad, aunque no tardará en quedarse obsoleta debido al ritmo de desarrollo.

10. **Trazabilidad: Log4J**

Se utilizará la herramienta log4J para almacenar la trazabilidad (ficheros de log) que todas las acciones que se lleven a cabo en la aplicación.

11. Exportación de datos a Excel: Apache POI 3.15

Para realizar una exportación de datos a formato Excel se utilizarán las librerías proporcionadas por Apache para leer y escribir ficheros en formatos de Microsoft Office. Con la utilización de PIO podremos obtener un fichero con los datos que nos interese para poder imprimir, generar fichas, etc.

1.5 Breve sumario de productos obtenidos

Los productos obtenidos como resultado final han sido:

- Memoria en formato pdf (incluye manual de usuario de los productos necesarios)
- Presentación: video explicativo con power point como guía
- Fichero war con el despliegue de la aplicación (incluye fichero de propiedades previamente configurado).
- Fichero zip con el código fuente de la aplicación. Este fichero contiene la carpeta script. Se encuentra en CodigoFuente.zip\tfgSaluPig\src\main\resources\script y contiene varios ficheros, En scriptBBDD.txt están las sentencias necesarias para crear el modelo de datos; en tfg_usuario, tfg_granja y tfg_visita se han incluído datos para poder hacer pruebas sin tener que introducir registro a registro desde la aplicación. Estos últimos habrá que importarlos utilizando la herramienta Pgadmin4.

1.6 Breve descripción de los otros capítulos de la memoria

En los siguientes capítulos se abordarán las fases comentadas para el desarrollo del producto.

En el capítulo 2 se realizará un análisis funcional del producto a desarrollar, incluyendo un modelo de casos de uso con su descripción, las fichas de los casos de uso más importantes.

En el capítulo 3 se abordará el diseño del producto. Aquí se mostrará un prototipo de la aplicación, seguido de un diagrama de clases y de base de datos necesarios para conseguir los objetivos perseguidos. Para finalizar este

capítulo se incluye un breve análisis de la arquitectura seleccionada y las razones principales de su elección.

En el capítulo 4 se recogen una serie de conclusiones elaboradas tras la finalización del trabajo

En el capítulo 5 y 6 se incluye el glosario de términos utilizado y la bibliografía que se ha necesitado para elaborar el proyecto.

Finalmente, en el capítulo 7 (Anexos) se incluye los manuales de instalación de la base de datos PostgreSQL y Tomcat. Además de incluye un breve guía de cómo hacer el despliegue de la aplicación comentando los puntos más importantes (fichero de configuración).

2. Análisis funcional

En este capítulo se abordará el análisis funcional del sistema que se pretende desarrollar. Se empezará con un posible diseño del modelo de casos de uso explicando sus partes más relevantes. A continuación, se detallarán las fichas de casos de uso que sean relevantes, saltando las que resulten obvias o repetitivas. Seguidamente se incluirá un prototipo de pantallas, lo cual orientará al usuario final acerca de cómo quedará la aplicación. También se incluirán en este punto tanto el diagrama de clases principales, así como el diagrama relacional de base de datos sobre el que trabajará nuestro sistema. Para finalizar, se incluye una descripción detallada de la arquitectura del sistema.

2.1 Modelo de casos de uso. Descripción

La aplicación que se va a construir en este proyecto pretende facilitar la gestión de las granjas con las que puede trabajar una empresa. Se pretende que, de una manera sencilla, se albergue una serie de información acerca de una granja que será visitada por alguien de nuestra empresa. Durante esta visita, la persona encargada guardará en el sistema una serie de datos que serán de utilidad para saber el estado de la granja, así como su evolución a lo largo de las visitas que podrá realizar.

Se plantea la existencia de dos perfiles:

1. Administrador. Responsable de la administración del sistema. Sus principales funciones serán la de administración de usuarios que puedan utilizar el sistema y de las granjas que se puedan visitar. También podrá realizar visitas a las granjas.
   El administrador también deberá estar logado en el sistema. Por esta razón será necesario que exista al menos un usuario cuando se haga la implantación del sistema. Este primer usuario será el que deberá crear el usuario administrador de nuestro cliente.

2. Visitante. El visitante, previamente dado de alta en el sistema, deberá logarse en la aplicación para poder rellenar los datos de su visita.

Para conseguir los objetivos marcados, se define el siguiente modelo de casos de uso:

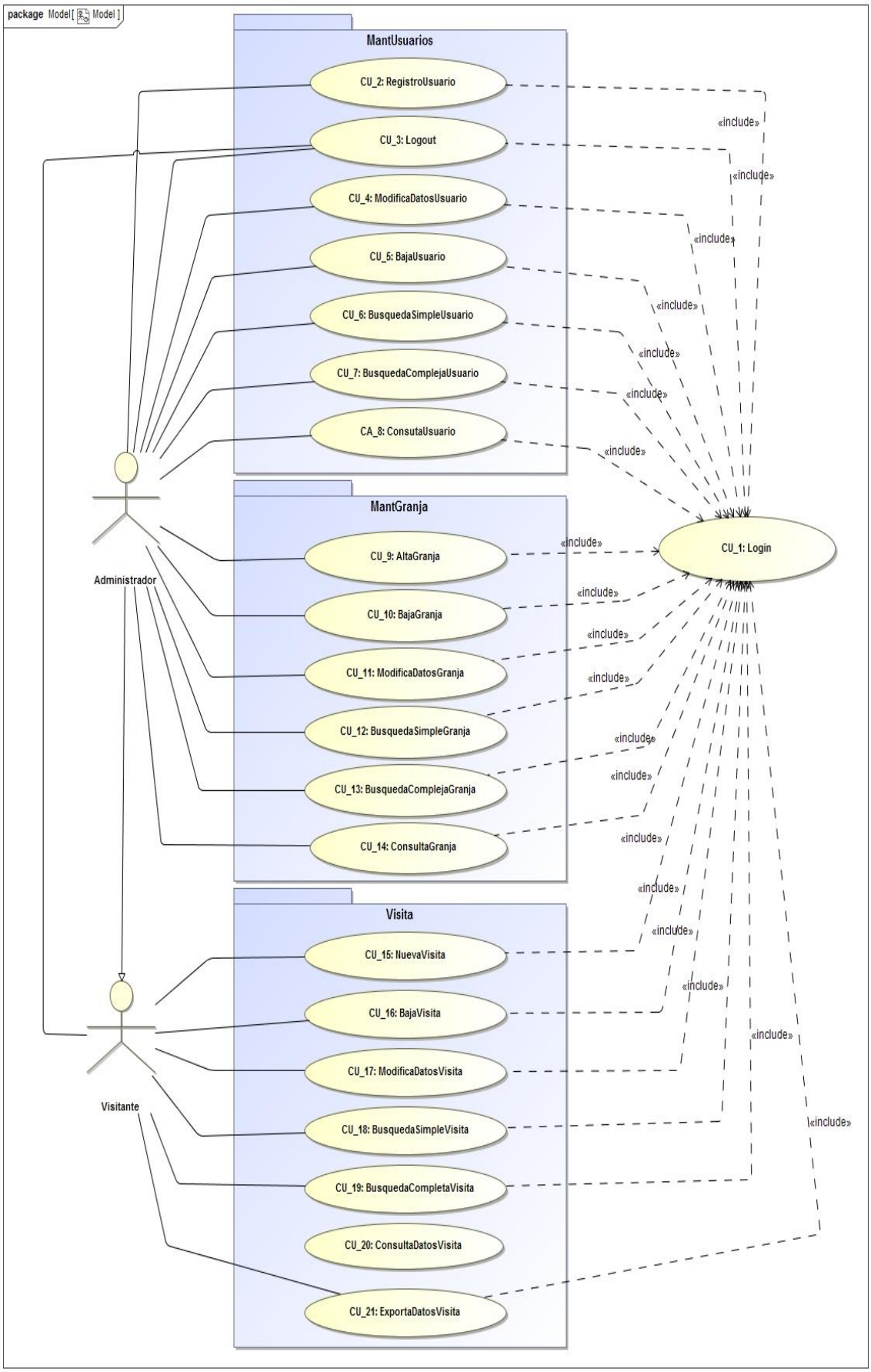


Ilustración 3. Modelo de casos de uso

En el modelo propuesto se distinguen tres grandes paquetes, en base a la funcionalidad que describen:

1. MantUsuario: acciones necesarias para el mantenimiento de los usuarios que podrán acceder al sistema. Alta, baja, modificación y consulta de los usuarios existentes.

2. MantGranja: acciones referidas al mantenimiento de las distintas granjas que se podrán visitar. Se tratará de mantenimiento de datos básicos de la granja: dirección, persona responsable, etc.

3. Visita: en este paquete se recogen todas las acciones que se podrán realizar con una visita: nuevas visitas, eliminar visitas, modificación de datos de una visita y búsqueda simples y complejas para analizar datos conjuntos.

Para la realización de todos los casos de uso será necesario estar logado en el sistema. Por esta razón, y cómo sólo el usuario administrador podrá generar nuevos usuarios, es indispensable que el sistema inicialmente cuente con un usuario administrador creado directamente en base de datos. A partir de este usuario, se podrán crear tantos administradores y/o visitantes como sea necesario.

En el punto siguiente se abordará la descripción de cada uno de los casos de uso.

2.2 Fichas de casos de uso

CU_1: Login	
Descripción	Permite a un usuario del sistema acceder al mismo. En función de sus permisos se le mostrará

	las opciones que le correspondan
Actores	Administrador, visitante
C. U. relacionados	--
Pre-condiciones	No estar identificado en el sistema
Post-condiciones	--
Proceso	Entrar a la url principal de la aplicación Introducir usuario y password Pulsar entrar
Alternativas/Excepciones	Si ya está logado en el sistema se le mostrará directamente la aplicación Si introduce usuario/password erróneo o que no existe se denegará el acceso

Tabla 2. Caso de uso: login

CU2_RegistroUsuario	
Descripción	Permite a un administrador del sistema dar de alta nuevos usuarios para acceder a la aplicación.
Actores	Administrador
C. U. relacionados	CU_1
Pre-condiciones	Debe estar logado en el sistema y ser administrador El nuevo usuario debe tener un nombre_usuario distinto de los existentes
Post-condiciones	--
Proceso	Seleccionar pestaña mantenimiento de usuarios Pulsar Agregar Nuevo Introducir datos del usuario: - Nombre completo - Usuario (login) - Password - Email - Teléfono - Rol

	Pulsar Guardar
Alternativas/Excepciones	En caso de querer guardar sin introducir todos los datos obligatorios se avisará al usuario del problema y se impedirá guardar

Tabla 3. Caso de uso: registro usuario

CU_3: Logout	
Descripción	Liberará la sesión del usuario borrando los datos relativos a él almacenados en sesión
Actores	Administrador, visitante
C. U. relacionados	CU_1
Pre-condiciones	Estar logado en el sistema
Post-condiciones	--
Proceso	Pulsar el botón “Salir”
Alternativas/Excepciones	

Tabla 4. Caso de uso: logout

CU_4: ModificaUsuario	
Descripción	Modifica los datos de un usuario
Actores	Administrador
C. U. relacionados	CU_1, CU6, CU_7
Pre-condiciones	Estar logado en el sistema. Tener el perfil Administrador
Post-condiciones	--
Proceso	Entrar en pestaña de mantenimiento de usuarios Buscar el usuario que se desea modificar Seleccionar la columna que se desea modificar haciendo doble click en la misma Introducir datos necesarios Salir del campo modificado haciendo click en cualquier parte de la pantalla o con el uso de tabulador. Con esta acción los campos serán

	guardados automáticamente
Alternativas/Excepciones	Si el usuario intenta introducir algún campo con formato erróneo (email sin formato adecuado, por ejemplo), se avisará del error y no permitirá guardar.

Tabla 5. Caso de uso: modificar usuario

CU_5: BajaUsuario	
Descripción	Realiza la baja lógica del usuario que seleccione.
Actores	Administrador
C. U. relacionados	CU_1, CU_6, CU_7
Pre-condiciones	El usuario debe estar conectado y tener el perfil Administrador. El usuario que se quiere modificar debe existir y estar dado de alta
Post-condiciones	El usuario dado de baja no podrá conectarse a la aplicación.
Proceso	Entrar en pestaña de mantenimiento de usuarios Buscar el usuario que se desea dar de baja Seleccionar la columna “estado” en el grid de datos y establecer valor “Baja” Salir del campo “Estado” haciendo click en cualquier parte de la pantalla o pulsando la tecla tabulador. Con esta acción los campos se guardarán automáticamente sin necesidad de pulsar guardar.
Alternativas/Excepciones	

Tabla 6. Caso de uso: baja usuario

CU_6: BusquedaSimpleUsuario	
Descripción	Realizará una búsqueda rápida de usuarios
Actores	Administrador
C. U. relacionados	CU_1

Pre-condiciones	El usuario debe estar logado y tener perfil Administrador
Post-condiciones	--
Proceso	Entrar en pestaña de mantenimiento de usuarios Introducir texto en recuadro de búsqueda habilitado a tal efecto Pulsar buscar o la tecla intro Aparecerá la lista de usuarios en los que alguno de sus datos contenga la cadena introducida. Se hace una búsqueda “or” del dato introducido en todos los campos del usuario.
Alternativas/Excepciones	--

Tabla 7. Caso de uso: búsqueda simple usuario

CU_7: BusquedaComplejaUsuario	
Descripción	Realizará una búsqueda más compleja que el caso anterior, ya que permite seleccionar que campo específico de los existentes estamos buscando. Si se seleccionan varios campos realizará una búsqueda “And” entre todos esos campos. Se establecen distintas posibilidades de búsqueda: que un campo empiece por una cadena, que finalice por ella, que la contenga, etc… Si es una fecha se permitirá buscar fechas anteriores y posteriores o coincidentes. Si es un campo numérico se permitirá buscar valores mayores, menores o coincidentes…
Actores	Administrador
C. U. relacionados	CU_1
Pre-condiciones	El usuario debe estar logado en el sistema y tener perfil Administrador
Post-condiciones	---
Proceso	Entrar en pestaña de mantenimiento de usuarios.

	Pulsar el botón “buscar” En la ventana que se abre introducir el/los criterios que se deseen aplicar: - nombre_completo - Login - Estado - Email - Teléfono - Rol Pulsar el botón “Buscar”
Alternativas/Excepciones	Si se quisieran inicializar todos los valores del cuadro de búsqueda, se podrá pulsar “Limpiar”.

Tabla 8. Caso de uso: búsqueda completa usuario

CU_8: ConsultaUsuario	
Descripción	Permite consultar los datos de un usuario
Actores	Administrador
C. U. relacionados	CU_1, CU_6, CU_7
Pre-condiciones	El usuario que realiza la acción debe estar logado y tener perfil Administrador
Post-condiciones	--
Proceso	Entrar en pestaña de mantenimiento de usuarios Realizar la búsqueda del usuario. Seleccionar el usuario que se quiere consultar haciendo click en el mismo En la parte inferior de la pantalla se mostrarán los datos del usuario que se ha seleccionado
Alternativas/Excepciones	

Tabla 9. Caso de uso: consulta de usuario

CU9_AltaGranja	
Descripción	Permite a un administrador del sistema dar de alta una nueva granja objeto de visita.

Actores	Administrador
C. U. relacionados	CU_1
Pre-condiciones	Debe estar logado en el sistema y tener perfil Administrador
Post-condiciones	--
Proceso	Seleccionar pestaña mantenimiento de granjas Pulsar Agregar Nuevo Introducir datos de la granja: - Nombre - Dirección - Responsable - Teléfono - Descripción Pulsar Guardar
Alternativas/Excepciones	En caso de querer guardar sin introducir todos los datos obligatorios se avisará al usuario del problema y se impedirá guardar También podrá salir de la ventana de nueva granja sin guardar los datos.

Tabla 10. Caso de uso: alta de granja

CU_10: BajaGranja	
Descripción	Realiza la baja lógica de una de las granjas del sistema.
Actores	Administrador
C. U. relacionados	CU_1, CU_12, CU_13
Pre-condiciones	El usuario debe estar conectado y tener el perfil Administrador. La granja que se quiere modificar debe existir y estar dada de alta
Post-condiciones	La granja dada de baja no podrá ser objeto de visitas.

Proceso	Entrar en pestaña de mantenimiento de granjas Buscar la granja que se desea dar de baja Seleccionar la columna “estado” en el grid de datos y establecer valor “Baja” Salir del campo “Estado” haciendo click en cualquier parte de la pantalla o pulsando la tecla tabulador. Con esta acción los campos se guardarán automáticamente sin necesidad de pulsar guardar.
Alternativas/Excepciones	

Tabla 11. Caso de uso: baja de granja

CU_11: ModificaDatosGranja	
Descripción	Modifica los datos básicos de una granja
Actores	Administrador
C. U. relacionados	CU_1, CU12, CU_13
Pre-condiciones	Estar logado en el sistema con perfil administrador
Post-condiciones	--
Proceso	Entrar en pestaña de mantenimiento de granjas Buscar la granja que se desea modificar Seleccionar la columna que se desea modificar haciendo doble click en la misma Introducir datos necesarios Salir del campo modificado haciendo click en cualquier parte de la pantalla o con el uso de tabulador. Con esta acción los campos serán guardados automáticamente
Alternativas/Excepciones	Si el usuario intenta introducir algún campo con formato erróneo se avisará del error y no permitirá guardar. Si quisiera cancelar el proceso, podrá hacerlo pulsando la tecla ESC.

Tabla 12. Caso de uso: modificar granja

CU_12: BusquedaSimpleGranja	
Descripción	Realizará una búsqueda rápida de las granjas del sistema
Actores	Administrador
C. U. relacionados	CU_1
Pre-condiciones	El usuario debe estar logado y tener perfil Administrador
Post-condiciones	--
Proceso	Entrar en pestaña de mantenimiento de granjas Introducir texto en recuadro de búsqueda habilitado a tal efecto Pulsar buscar o la tecla intro Aparecerá la lista de granjas en las que alguno de sus datos contenga la cadena introducida. Se hace una búsqueda “or” del dato introducido en todos los campos de la granja.
Alternativas/Excepciones	--

Tabla 13. Caso de uso: búsqueda simple de granja

CU_13: BusquedaComplejaGranja	
Descripción	Realizará una búsqueda más compleja que el caso anterior (CU_12), ya que permite seleccionar que campo específico de los existentes estamos buscando. Si se seleccionan varios campos realizará una búsqueda “And” entre todos esos campos. Se establecen distintas posibilidades de búsqueda: que un campo empiece por una cadena, que finalice por ella, que la contenga, etc… Si es una fecha se permitirá buscar fechas anteriores y posteriores o coincidentes. Si es un campo numérico se permitirá buscar valores mayores, menores o coincidentes…

Actores	Administrador
C. U. relacionados	CU_1
Pre-condiciones	El usuario debe estar logado en el sistema y tener perfil Administrador
Post-condiciones	---
Proceso	Entrar en pestaña de mantenimiento de granjas. Pulsar el botón “buscar” En la ventana que se abre introducir el/los criterios que se deseen aplicar: - Nombre - Dirección - Responsable - Teléfono - Estado Pulsar el botón “Buscar”
Alternativas/Excepciones	Si se quisieran inicializar todos los valores del cuadro de búsqueda, se podrá pulsar “Limpiar”.

Tabla 14. Caso de uso: búsqueda compleja granja

	CU_14: ConsultaGranja
Descripción	Permite consultar los datos de un usuario
Actores	Administrador
C. U. relacionados	CU_1, CU_12, CU_13
Pre-condiciones	El usuario que realiza la acción debe estar logado y tener perfil Administrador
Post-condiciones	--
Proceso	Entrar en pestaña de mantenimiento de granjas Realizar la búsqueda de la granja. Seleccionar la granja que se quiere consultar haciendo click en la misma En la parte inferior de la pantalla se mostrarán los datos de la granja que se ha seleccionado
Alternativas/Excepciones	

Tabla 15. Caso de uso: consulta de granja

CU_15: NuevaVisita	
Descripción	Permitirá almacenar los datos extraídos de una visita realizada a la granja
Actores	Administrador, Visitante
C. U. relacionados	CU1
Pre-condiciones	El usuario que realiza la acción debe estar logado en el sistema. El usuario debe tener perfil Administrador o Visitante
Post-condiciones	
Proceso	Seleccionar pestaña mantenimiento de visitas Pulsar Agregar Nuevo Introducir datos de la visita: - Granja - Fecha visita - Número de trabajadores - Número de bajas SS - Grado satisfacción - Número animales macho - Número animales hembra - Número nacimientos - Número fallecimientos - Diseño - Metros cuadrados - Observaciones Será obligatorio introducir, al menos, granja y fecha de visita. Pulsar Guardar
Alternativas/Excepciones	En caso de querer guardar sin introducir todos los datos obligatorios se avisará al usuario del problema y se impedirá guardar

	También podrá salir de la ventana de la visita a la granja sin guardar los datos.

Tabla 16. Caso de uso: crear visita

CU_16: BajaVisita	
Descripción	Realiza la baja lógica de una de las visitas que se ha realizado
Actores	Administrador, Visitante
C. U. relacionados	CU_1, CU_18, CU_19
Pre-condiciones	El usuario debe estar conectado y tener el perfil Administrador o visitante La visita que se quiere modificar debe existir y estar dada de alta
Post-condiciones	--
Proceso	Entrar en pestaña de mantenimiento de visitas Buscar la visita que se desea dar de baja Seleccionar la columna “estado” en el grid de datos y establecer valor “Baja” Salir del campo “Estado” haciendo click en cualquier parte de la pantalla o pulsando la tecla tabuladora. Con esta acción los campos se guardarán automáticamente sin necesidad de pulsar guardar.
Alternativas/Excepciones	

Tabla 17. Caso de uso: baja de visita

CU_17: ModificaDatosVisita	
Descripción	Modifica los datos básicos de una visita realizada
Actores	Administrador, Visitante
C. U. relacionados	CU_1, CU18, CU_19
Pre-condiciones	Estar logado en el sistema. Tener el perfil Administrador o visitante

Post-condiciones	--
Proceso	Entrar en pestaña de mantenimiento de visitas Buscar la visita que se desea modificar Seleccionar la columna que se desea modificar haciendo doble click en la misma Introducir datos necesarios Salir del campo modificado haciendo click en cualquier parte de la pantalla o con el uso de tabulador. Con esta acción los campos serán guardados automáticamente
Alternativas/Excepciones	Si el usuario intenta introducir algún campo con formato erróneo se avisará del error y no permitirá guardar. Si quisiera cancelar el proceso, podrá hacerlo pulsando la tecla ESC.

Tabla 18. Caso de uso: modifica visita

CU_18: BusquedaSimpleVisita	
Descripción	Realizará una búsqueda rápida de las visitas dadas de alta en el sistema
Actores	Administrador, visitante
C. U. relacionados	CU_1
Pre-condiciones	El usuario debe estar logado y tener perfil Administrador o visitante
Post-condiciones	--
Proceso	Entrar en pestaña de mantenimiento de visitas Introducir texto en recuadro de búsqueda habilitado a tal efecto Pulsar buscar o la tecla intro Aparecerá la lista de visitas en las que alguno de sus datos contenga la cadena introducida. Se hace una búsqueda “or” del dato introducido en todos los campos de la visita.

Alternativas/Excepciones	--

Tabla 19. Caso de uso: búsqueda simple visita

CU_19: BusquedaComplejaVisita	
Descripción	Realizará una búsqueda más compleja que el caso anterior (CU_18), ya que permite seleccionar que campo específico de los existentes estamos buscando. Si se seleccionan varios campos realizará una búsqueda "And" entre todos esos campos. Se establecen distintas posibilidades de búsqueda: que un campo empiece por una cadena, que finalice por ella, que la contenga, etc… Si es una fecha se permitirá buscar fechas anteriores y posteriores o coincidentes. Si es un campo numérico se permitirá buscar valores mayores, menores o coincidentes…
Actores	Administrador, Visitante
C. U. relacionados	CU_1
Pre-condiciones	El usuario debe estar logado en el sistema y tener perfil Administrador
Post-condiciones	---
Proceso	Entrar en pestaña de mantenimiento de granjas. Pulsar el botón "buscar" En la ventana que se abre introducir el/los criterios que se deseen aplicar Pulsar el botón "Buscar"
Alternativas/Excepciones	Si se quisieran inicializar todos los valores del cuadro de búsqueda, se podrá pulsar "Limpiar".

Tabla 20. Caso de uso: búsqueda compleja visita

CU_20: ConsultaDatosVisita	
Descripción	Permite consultar los datos de una visita
Actores	Administrador, visitante

C. U. relacionados	CU_1, CU_18, CU_19
Pre-condiciones	El usuario que realiza la acción debe estar logado y tener perfil Administrador o visitante
Post-condiciones	--
Proceso	Entrar en pestaña de mantenimiento de visitas Realizar la búsqueda de la visita. Seleccionar la visita que se quiere consultar haciendo click en la misma En la parte inferior de la pantalla se mostrarán los datos de la visita que se ha seleccionado
Alternativas/Excepciones	

Tabla 21. Caso de uso: consulta datos de visita

CU_21: ExportarDatosVisita	
Descripción	Permite exportar a un fichero Excel los datos de las visitas. No será necesario disponer de la aplicación Microsoft Excel ya que existen alternativas gratuitas en el mercado: xls viewer, LibreOffice, OpenOffice, GoogleDocs, etc…
Actores	Administrador, visitante
C. U. relacionados	CU_1, CU_18, CU_19
Pre-condiciones	El usuario que realiza la acción debe estar logado y tener perfil Administrador o visitante
Post-condiciones	--
Proceso	Entrar en pestaña de mantenimiento de visitas Pulsar el botón Listados. Seleccionar “Todas las visitas realizadas” Se iniciará la descarga de un fichero .xls con todas las visitas existentes en la base de datos
Alternativas/Excepciones	Existe la opción de imprimir solamente las visitas que se encuentran en el filtro. Para ello, tendrá que pulsar “Visitas filtradas realizadas”

Tabla 22. Caso de uso: exportar visitas a Excel

3. Diseño de la aplicación

En el presente capítulo se muestra una aproximación a lo que será el diseño de la aplicación.

3.1 Prototipo de pantallas

Un prototipo es una maqueta del diseño de la aplicación sin funcionalidad implementada. Este prototipo es un paso inicial hacia la implementación de la aplicación con el que se conseguirá que el usuario final sea capaz de entender cómo funcionará la aplicación. Como su nombre indica, es una maqueta y, por tanto, la versión final podría variar.

3.1.1 Pantalla de inicio

Para poder entrar en la aplicación todos los usuarios deberán estar identificados en la misma. De esta forma, para poder empezar a trabajar, deberá existir al menos un usuario creado en base de datos (mediante la ejecución de un script). La pantalla de inicio será parecida a la siguiente:

Ilustración 4. Pantalla login

Si los datos introducidos no son correctos se mostrará mensaje informando de ello. En caso contrario, se comprobará que rol tiene asignado el usuario y se mostrarán las opciones oportunas. Estas son:

Usuario administrador: mantenimiento de usuarios, mantenimiento de granjas y mantenimiento de visitas.

Ilustración 5. Pantalla usuario administrador

Usuario visitante: únicamente tendrá la opción de mantenimiento de visitas.

Ilustración 6. Pantalla usuario visitante

En todos los casos, siempre se mantendrá visible la opción “Salir” para abandonar la aplicación limpiando los datos de sesión. De esta forma se evitará accesos no deseados por otros usuarios. En el caso de que se abandone la aplicación sin pulsar la opción de “Salir”, si el usuario vuelve a acceder a la misma ya no se le mostrará la pantalla de login.

3.1.2 Mantenimiento de usuarios

Las pantallas para el mantenimiento de usuarios se parecerán a la siguiente:

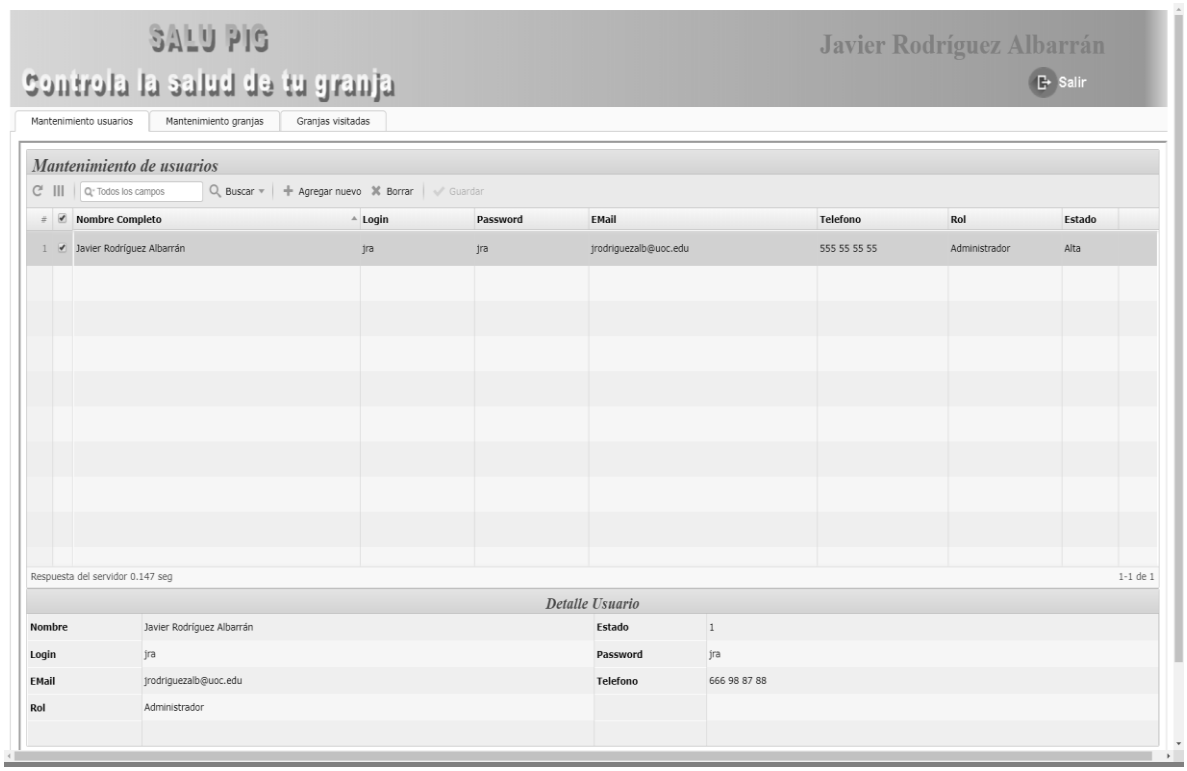


Ilustración 7. Pantalla mantenimiento usuarios

En ésta, se podrán ver los usuarios que están introducidos en la aplicación.

Desde esta misma ventana, se podrá hacer una búsqueda rápida de usuarios introduciendo un texto libre en el recuadro habilitado para ello y pulsando buscar.

También se podrá realizar una búsqueda más compleja. Estando en blanco el recuadro de búsqueda, si se pulsa buscar se desplegará una ventana con los datos sobre los que podremos consultar:

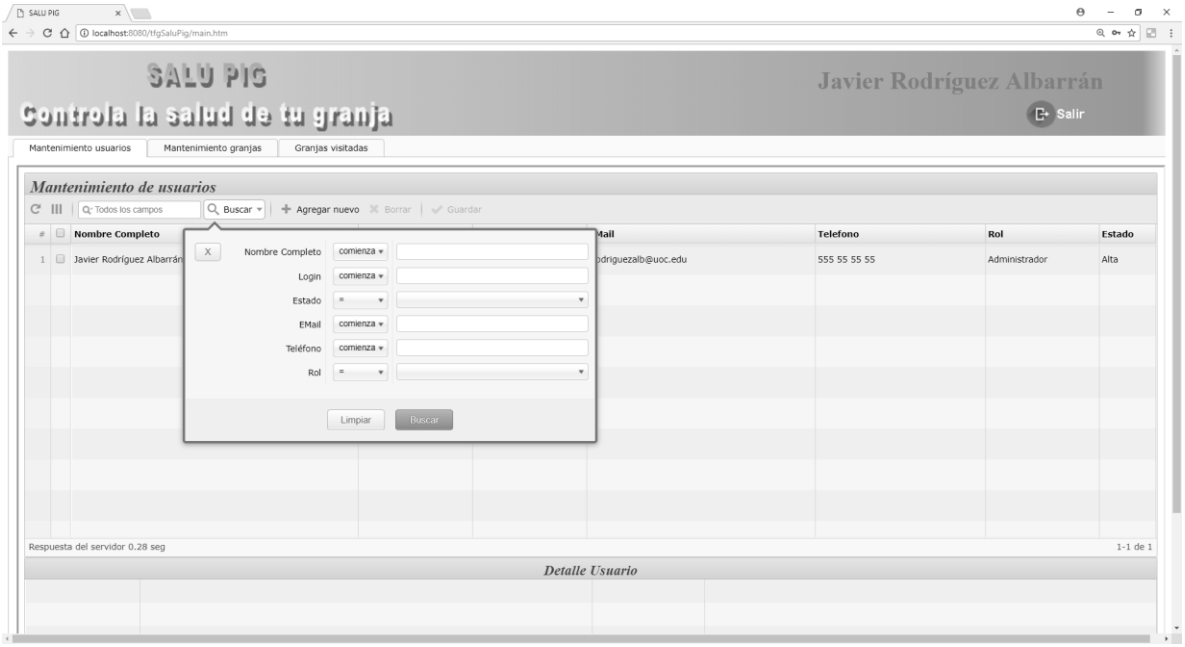


Ilustración 8. Pantalla búsqueda de usuarios

Para dar de alta nuevos usuarios, habrá que pulsar la opción “Agregar Nuevo”. La pantalla para introducir nuevos usuarios es la siguiente:

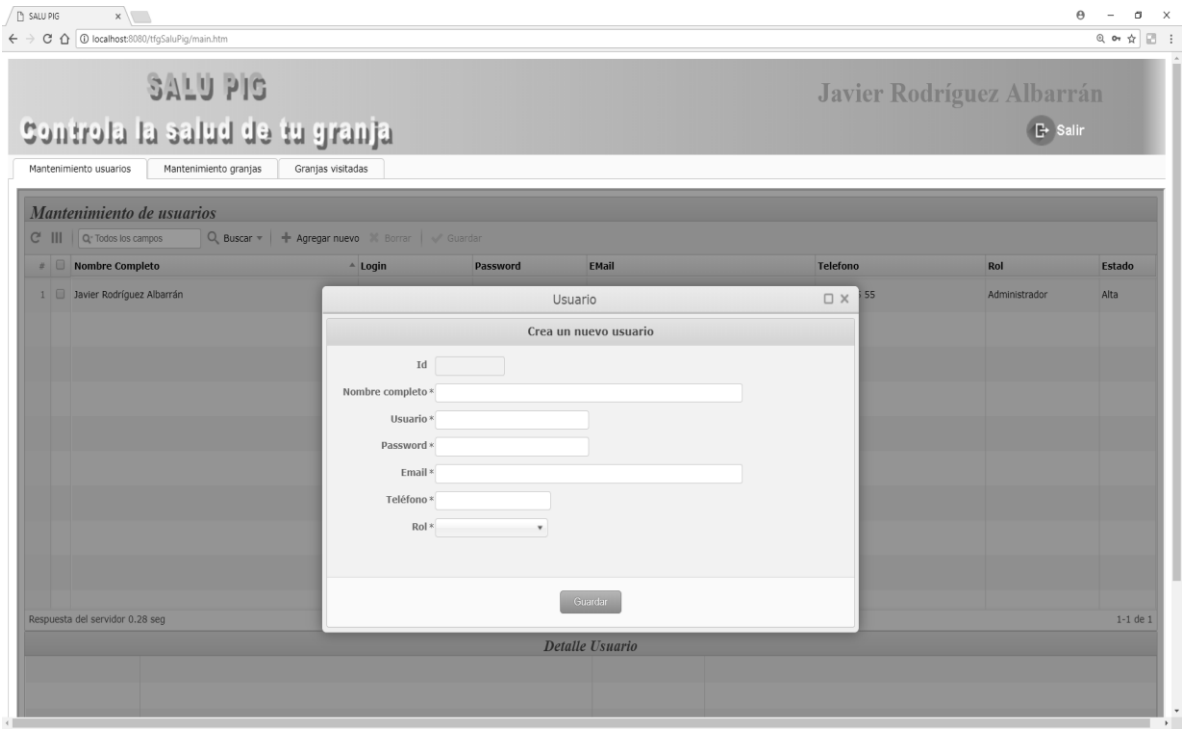


Ilustración 9. Pantalla alta de usuarios

Para consular los datos de un usuario simplemente habrá que seleccionarlo y sus datos aparecerán en la parte inferior de la ventana:

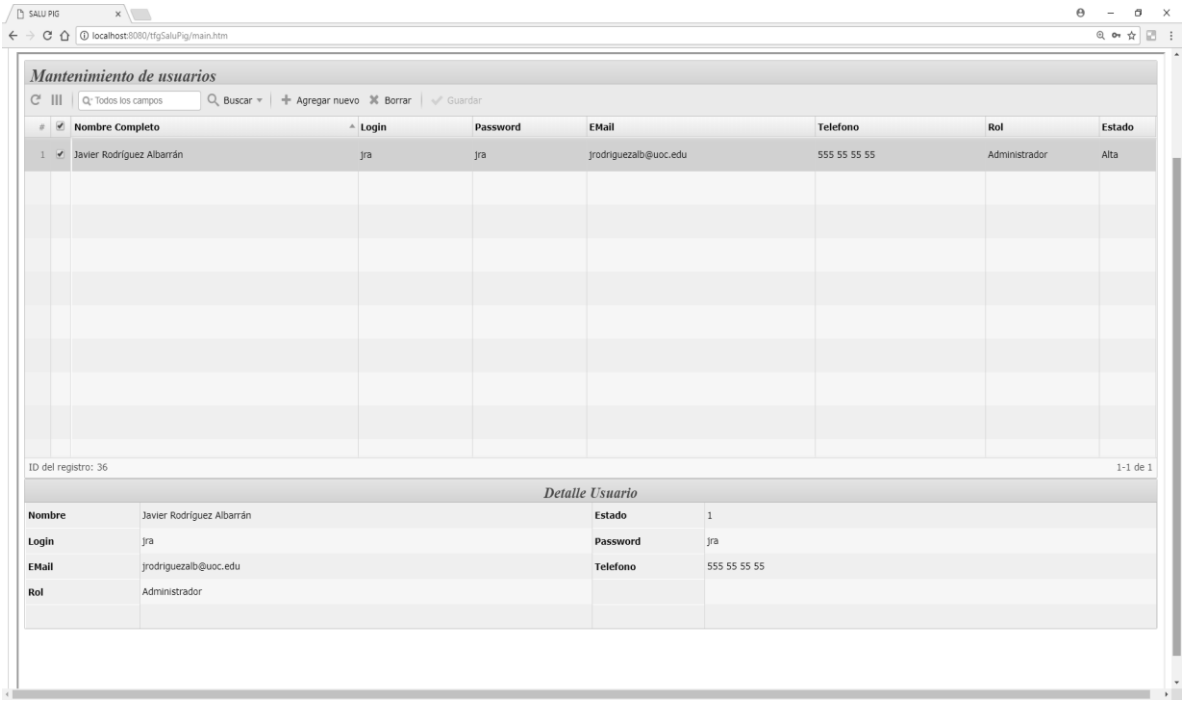


Ilustración 10. Pantalla consulta de usuarios

Además, los datos que se muestran en este filtro será configurables. Pudiendo seleccionar el orden de las columnas (arrastrando el título de la columna), se podrá cambiar el tamaño de cada una y se podrán seleccionar cuáles son visibles y cuáles no. Para esto seleccionaremos en icono con las tres rayas paralelas verticales situado al lado del cuadro de búsqueda:

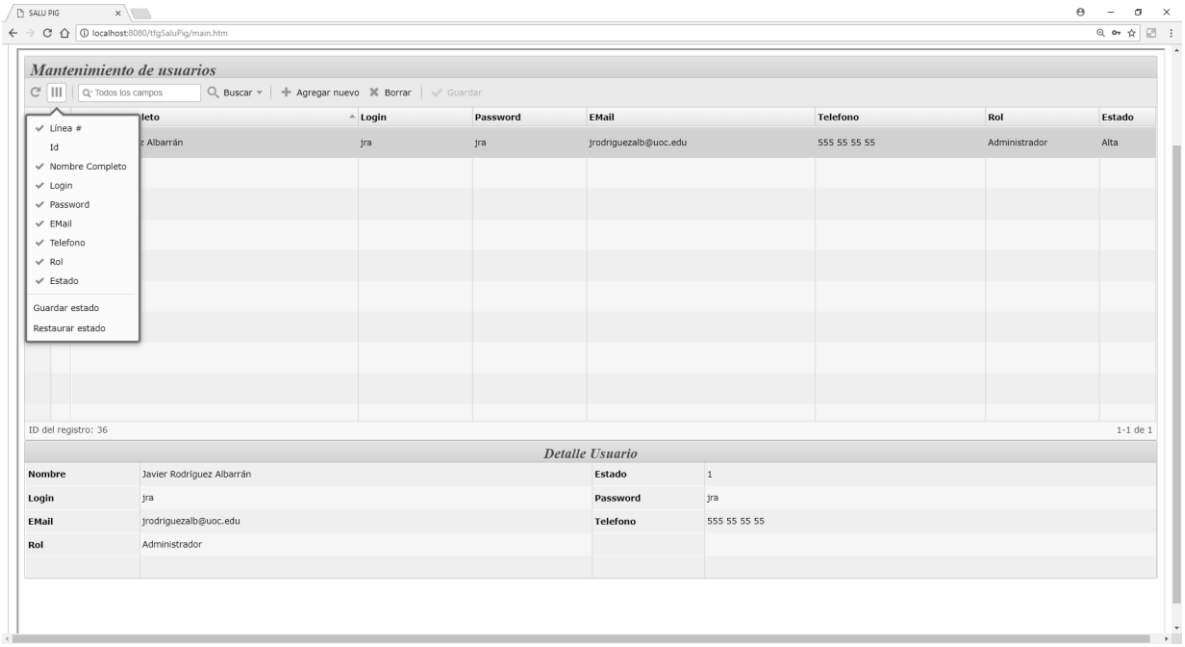


Ilustración 11. Pantalla configuración mant. usuarios

3.1.3 Mantenimiento de granjas

Las pantallas para el mantenimiento de granjas serán bastante parecidas a las del mantenimiento de usuario. De esta forma conseguiremos que el periodo de aprendizaje por parte del usuario final sea mínimo. En la pantalla inicial se podrán ver las granjas que están introducidas en la aplicación.

Ilustración 12. Pantalla mantenimiento de granjas

Al igual que sucede en al mantenimiento de usuarios, desde esta ventana, se podrá hacer una búsqueda rápida de granjas introduciendo un texto libre en el recuadro habilitado para ello y pulsando buscar.

También se podrá realizar una búsqueda más compleja. Estando en blanco el recuadro de búsqueda, si se pulsa buscar se desplegará una ventana con los datos sobre los que podremos consultar:

Ilustración 13. Pantalla búsqueda de granjas

Para dar de alta nuevas granjas, habrá que pulsar la opción “Agregar Nuevo”. La pantalla para introducir nuevas granjas será la siguiente:

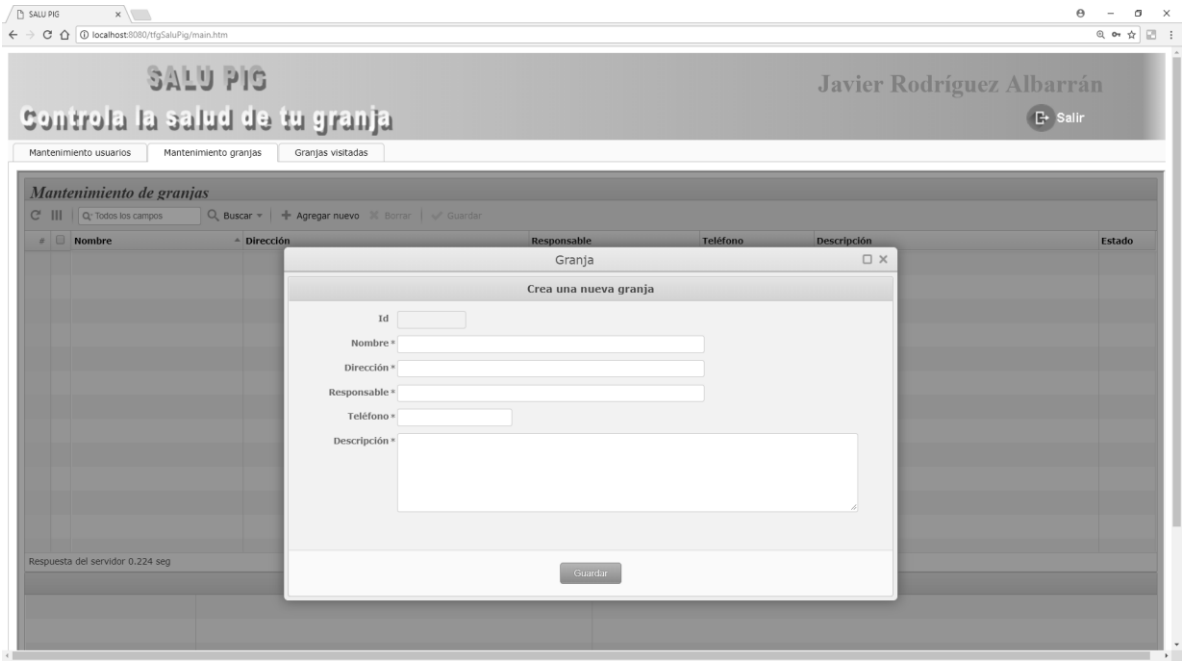


Ilustración 14. Pantalla alta de nueva granja

Para consultar los datos de una granja simplemente habrá que seleccionarla y sus datos aparecerán en la parte inferior de la ventana:

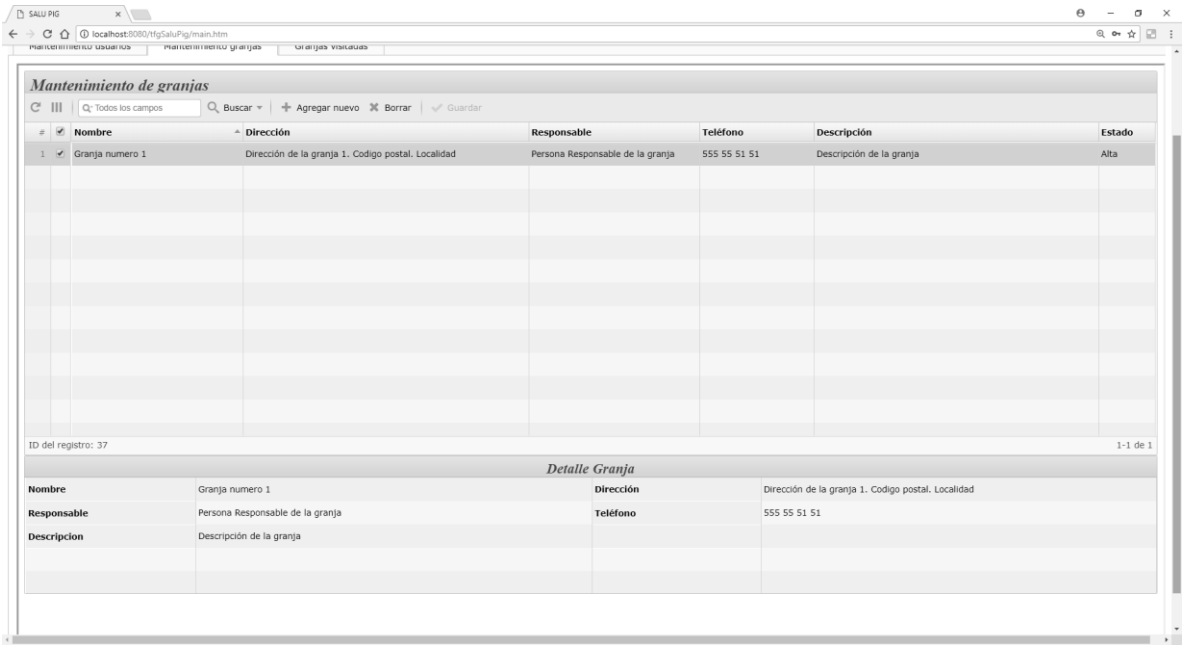


Ilustración 15. Pantalla consulta de granjas

En este caso (al igual que en el mantenimiento de usuarios), las columnas que se muestran también serán igualmente configurables.

3.1.4 Mantenimiento de visitas

Persiguiendo un diseño uniforme de la aplicación, las pantallas para el mantenimiento de las visitas que se realicen también serán parecidas a las dos anteriores. Obviamente cada una tendrá sus propios datos. En la pantalla inicial se podrán ver las visitas que se han realizado a las granjas con las que trabajan:

Ilustración 16. Pantalla mantenimiento de visitas

Al igual que sucede en al mantenimiento de usuarios y granjas, desde esta ventana, se podrá hacer una búsqueda rápida de visitas introduciendo un texto libre en el recuadro habilitado para ello y pulsando buscar.

También se podrá realizar una búsqueda más compleja. Estando en blanco el recuadro de búsqueda, si se pulsa buscar se desplegará una ventana con los datos sobre los que podremos consultar:

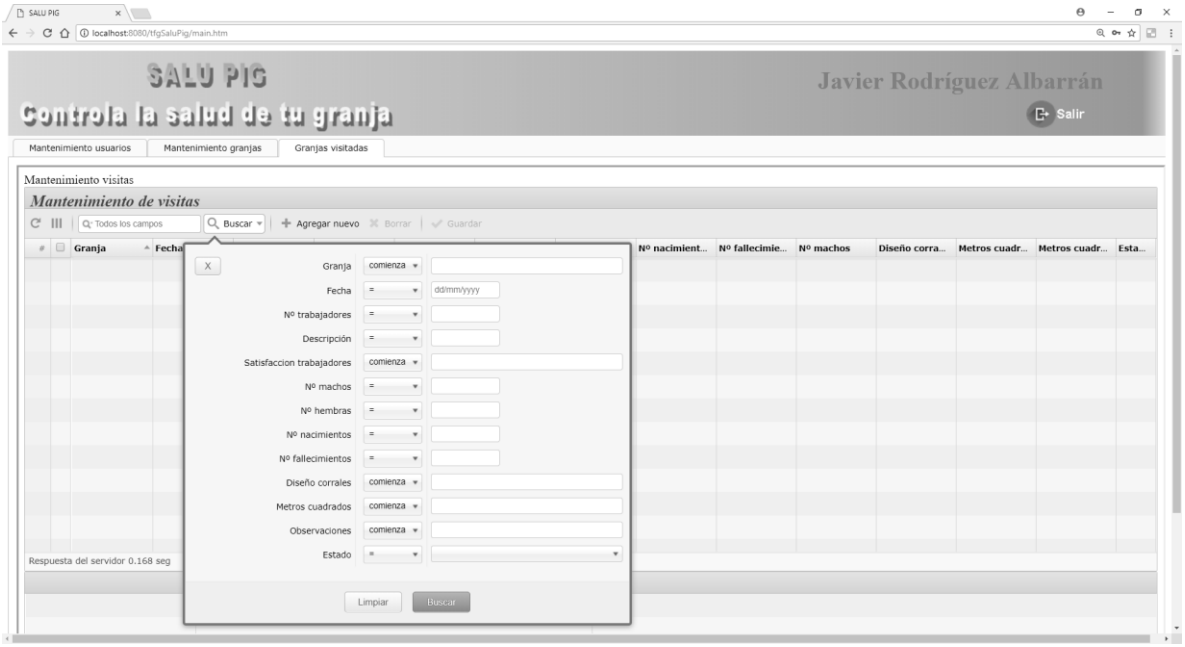


Ilustración 17. Pantalla de búsqueda de visitas

Para dar de alta una nueva visita habrá que pulsar la opción “Agregar Nuevo”. La pantalla para introducir nuevas granjas será la siguiente:

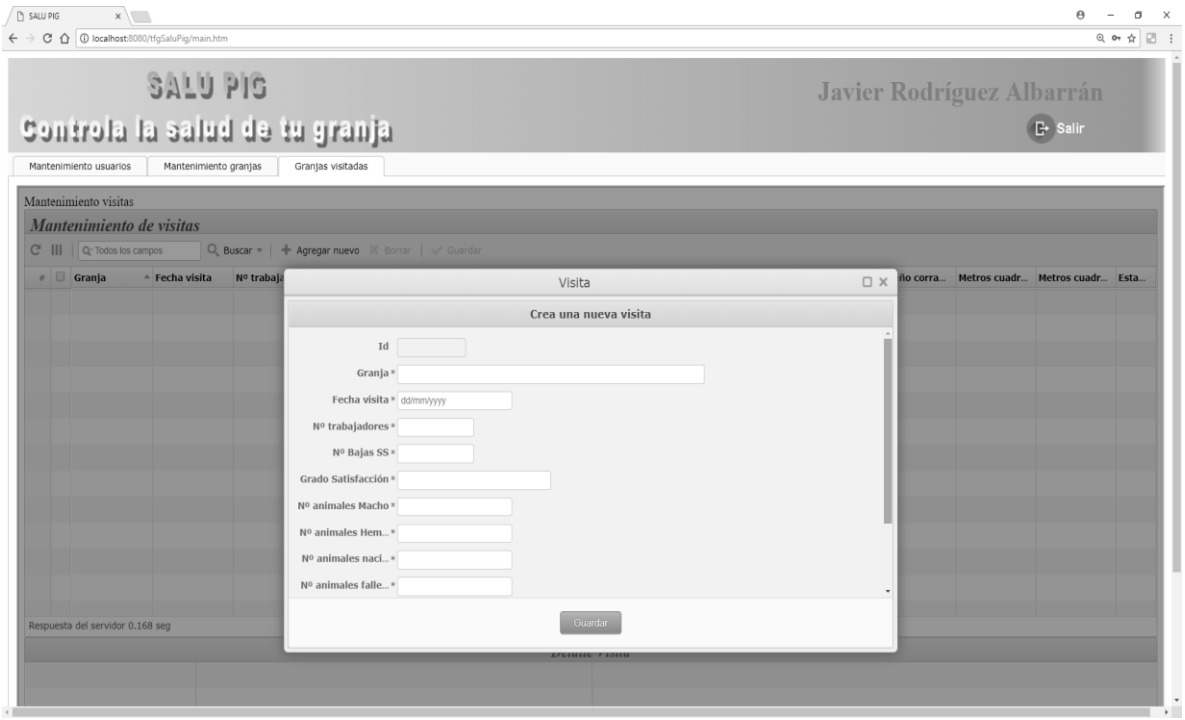


Ilustración 18. Pantalla alta de nueva visita

Para consular el detalle de una visita simplemente habrá que seleccionarla y sus datos aparecerán en la parte inferior de la ventana

También en este caso las columnas que se muestran también serán igualmente configurables libremente

3.2 Diagrama de clases principales

El diagrama de las clases principales que compondrán la aplicación será el siguiente:

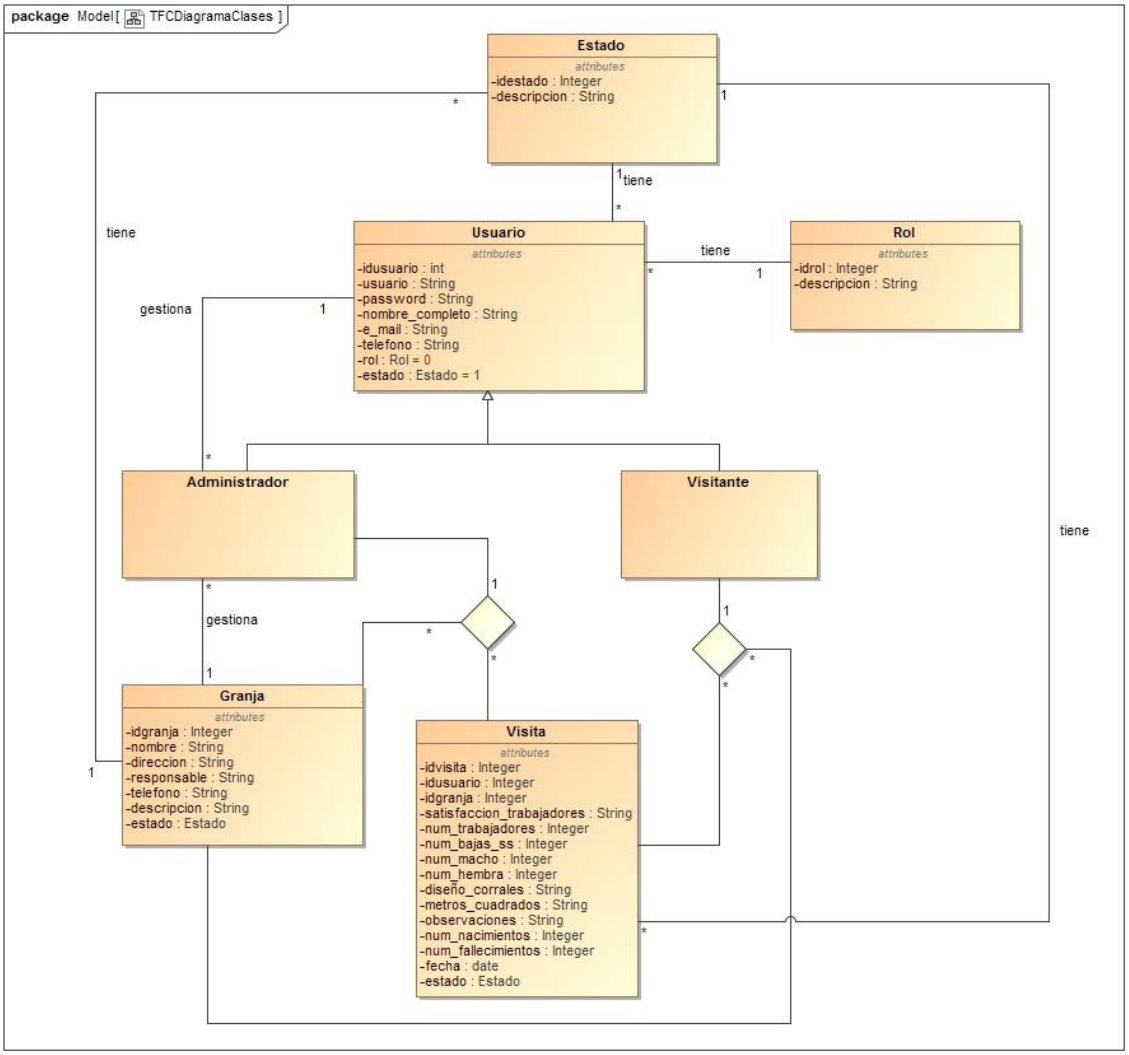


Ilustración 19. Diagrama de clases

En el diagrama puede observarse que todas las clases tienen un estado. Esto es por considerarse común el estado (alta o baja) en todos los casos. Para la implementación de la misma, estos valores se han metido en un fichero de propiedades. Algo parecido ocurre con la clase Rol. Al tener únicamente dos valores, se ha decidido incluir estos dos valores en el mismo fichero de propiedades, de forma que su tratamiento será mucho más sencillo.

La clase usuario tendrá dos posibles tipos: Administrador y Visitante. Ambas tienen los mismos datos. Haciéndolo así, se deja abierta la posibilidad de guardar información distinta para cada uno de ellos. Además, se puede apreciar que cada uno de ellos podrá hacer operaciones distintas. En este caso, el usuario Administrador puede gestionar otros usuarios, gestionar granjas y realizar visitas. El usuario Visitante sólo podrá realizar visitas. Hay que comentar también aquí que la relación Administrador-Granja-Visita y Visitante-Granja-Visita se podría haber llevado al nivel de Usuario. Se deja abierta la posibilidad de sólo el perfil Visitante puedan realizar visitas.

3.3 Diagrama relacional de base de datos

Para cumplir los objetivos perseguidos, se ha modelado la base de datos de la siguiente forma:

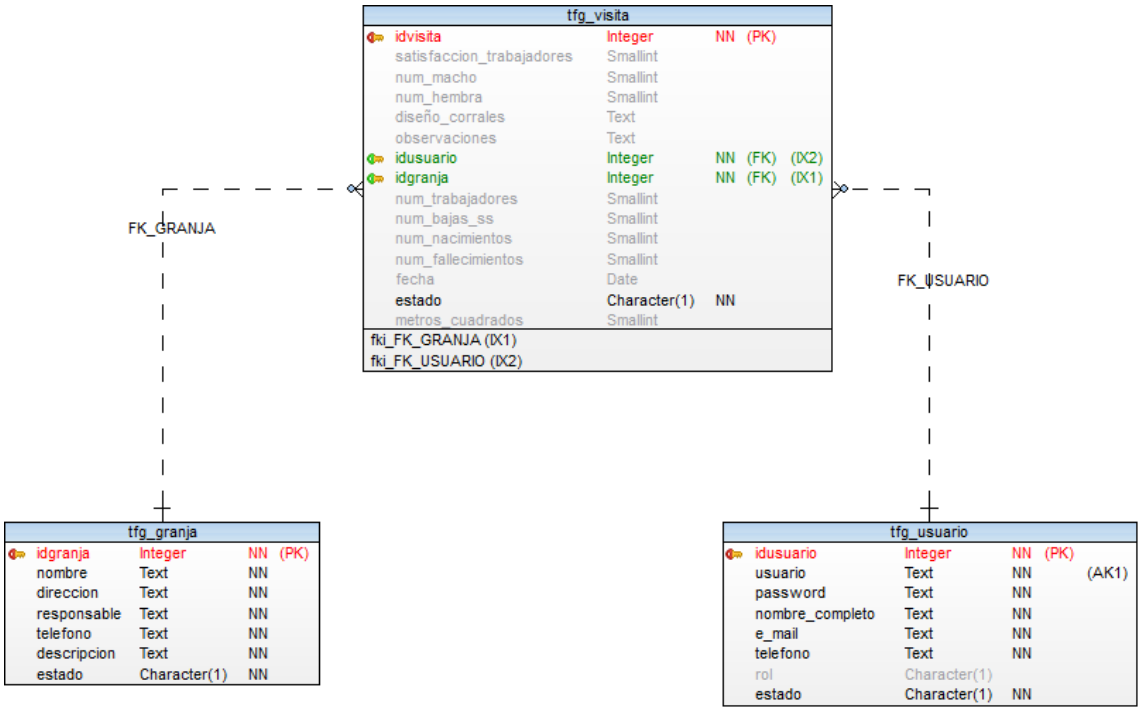


Ilustración 20. Modelo de datos

En el diagrama se reflejan los datos obligatorios con letra negrita, los no obligatorios con letra más clara, en rojo las columnas que son clave primaria de la tabla, y en verde se pueden apreciar las columnas que referencian a otras tablas.

Los datos de los usuarios que podrán conectarse a la aplicación se guardarán en la tabla tfg_usuario. Cuando se crea la base de datos debe crearse un usuario administrador, que será el usuario inicial con el que se trabajará. Esta tabla contiene:

- Idusuario: clave primaria de la tabla. Será un campo autocalculado mediante secuencia.
- Usuario: login de usuario. Debe ser único.
- Password: password del usuario. No se ha establecido longitud mínima
- ni validaciones de caracteres.
- Nombre_completo: nombre completo del usuario
- E_mail: email de contacto del usuario. Se valida en la aplicación que sea una cadena de email válida

- Teléfono: número de teléfono de contacto.
- Rol: rol con el que se conectará a la aplicación. Dos posibles valores: 0 (administrador) y 1 (visitante)
- Estado: estado del usuario. Dos posibles valores: 0 (baja) y 1 (alta)

Los datos de cada una de las granjas que podrán ser visitadas se almacenarán en la tabla tfg_granja. Estaba tabla contiene:

- Idgranja: clave primaria de la tabla.
- Nombre: nombre identificativo de la granja. Será un campo autocalculado mediante secuencia.
- Dirección: ubicación de la granja
- Responsable: datos de la persona responsable de la granja
- Teléfono: teléfono de contacto del responsable de la granja
- Descripción: texto libre para describir la granja
- Estado: estado del usuario. Dos posibles valores: 0 (baja) y 1 (alta)

Los datos de las visitas que se realicen se guardarán en la tabla tfg_visita. Contendrá los siguientes datos:

- Idvisita: clave primaria de la tabla. Será un campo autocalculado mediante secuencia.
- Satisfacción_trabajadores: grado de satisfacción de los trabajadores en un intervalo de 0 a 10, siendo 0 el grado más instafisfactorio y 10 la mejor valoración posible.
- Num_macho: número de animales macho que tiene la granja
- Num_hembra: número de animales hembra que tiene la granja
- Diseño_corrales: texto relativo al diseño que presentan los corrales. Cualquier dato relativo a éstos se incluirá aquí
- Metros_cuadrados: número de metros cuadrados que tiene la granja. Servirá para valorar si hay espacio suficiente para la cantidad de animales existentes.

- Observaciones: observaciones que se deseen hacer constar de la visita realizada
- Idusuario: identificador del usuario que realiza la visita. Es clave foránea a la tabla tfg_usuario
- Idgranja: identificador de la granja que se visita. Es clave foránea a la tabla tfg_granja.
- Num_trabajadores: número de trabajadores que tiene la granja. Servirá para valorar si hay suficiente mano de obra para un correcto mantenimiento de la granja
- Num_bajas_ss: número de bajas que se han producido en los trabajadores de la granja. Podría dar indicios de las condiciones en las que trabajan los trabajadores. Un elevado número de bajas podría ser señal de malas condiciones laborales.
- Num_nacimientos: número de animales nacidos desde la última visita a la granja.
- Num_fallecimientos: número de animales fallecidos (no sacrificados) desde la última visita. Podría ser una señal de malas condiciones de la granja
- Fecha: fecha en la que se realiza la visita.
- Estado: estado de la visita. Dos posibles valores: 0 (baja) y 1 (alta)

Para controlar los estados de cada una de las entidades se podría haber incluido una tabla tfg_estado con dos columnas: id (de tipo int) y descripción (de tipo Text) con los posibles valores alta y baja (o los que fueran necesarios). En este caso se ha decidido establecer estos valores en el fichero de configuración de la aplicación. Este fichero, denominado tfgSaluPig.properties, contendrá entre otros los valores posibles para los estados de la visita, de usuario y de granja, y los valores de conexión a la base de datos. Al utilizar maven como herramienta, se ha incluído este fichero de configuración en el despliegue. De esta forma, si quisiera cambiarse algún parámetro, habría que descomprimir el fichero war generado, modificar el fichero properties, y volver a generar el war que se desplegará en el servidor correspondiente.

Lo mismo ocurre con el campo rol de la tabla tfg_usuario. Al existir únicamente dos perfiles posibles, éstos se han añadido en el fichero de configuración de la aplicación.

También se podría haber incluido en la tabla tfg_granja un campo idusuario como clave foránea a la tabla tfg_usuario. De esta forma se controlaría el usuario que ha creado/modificado la granja.

3.4 Arquitectura de la aplicación

Las aplicaciones distribuidas contemporáneas presentan diversos estilos de arquitectura, normalmente clasificadas en función de dónde estén situados los datos a los que se pretende acceder. Surgen así arquitecturas a 2, 3, …, n niveles, sistemas de flujo de datos, arquitecturas cliente/servidor, arquitecturas basadas en eventos, arquitecturas orientadas a servicios, etc. Según hemos visto en la asignatura Ingeniería del Software de Componentes y Sistemas Distribuidos, se podría definir un estilo arquitectónico como: “Los estilos arquitectónicos representan formas diferentes de estructurar un sistema usando componentes y conectores, de acuerdo con decisiones esenciales sobre los elementos arquitectónicos y estableciendo restricciones importantes sobre tales elementos y sus posibles relaciones.”

Como se pretende que los usuarios puedan acceder a la aplicación sin tener que instalar ningún software especial, únicamente un navegador, se ha seleccionado una arquitectura en 3 capas: presentación, negocio e integración. El uso de tres capas presenta varias ventas. Entre ellas:

- Sistema altamente flexible
- Sistema muy escalable
- Sistema poco acoplado, ya que la dependencia de unas capas con respecto a otras es mínima. No resultaría nada complicado cambiar alguna de las capas.
- Tolerante a fallos
- Fácilmente mantenible

3.4.1 Capa de presentación

Define la interfaz gráfica de usuario y las interacciones del mismo con la aplicación. Se hará a través de un navegador web.

Para el diseño de la vista se ha decidido utilizar Ajax (Asynchronous Javascript and XML). Según la página https://platzi.com/blog/ajax-con-jquery/ Ajax) "es una técnica de desarrollo web que, al combinar una serie de tecnologías independientes, nos permite intercambiar información entre el servidor y el cliente (un navegador web) de forma asíncrona. Como resultado, obtenemos una navegación ágil, rápida y dinámica; y también la posibilidad de realizar cambios sobre una web sin necesidad de actualizarla". Las tecnologías que lo posibilitan son, entre otras: javascript, xml, json, dom y XMLHttpRequest. El principal inconveniente de utilitzar ajax se encuentra en la necesidad de escribir código distinto para cada uno de los tipos de navegadores que utilicemos. Para solventar este problema y no tener que hacer un desarrollo distinto para cada posible navegador se ha utilizado jQuery. Según su página web (https://jquery.com/), jQuery "es una biblioteca de Javascript, rápida, pequeña y rica en funciones. Hace cosas como el recorrido y manipulación de documentos HTML, manejo de eventos, animación, y un Ajax mucho más simple, con una API fácil de usar que funciona en todos los navegadores actuales más utilizados."

Implementado con jQuery, se ha utilizado una librería denominada "w2ui" (http://w2ui.com/web/). Consiste en un conjunto de plugins de jQuery para desarrollos front-end que necesitan acceso a datos almacenados en una base de datos, por ejemplo. Mediante la utilización de ésta potente pero ligera herramienta, se ha facilitado la implementación de las pestañas para manejar los distintos mantenimientos, así como los grids de mantenimiento de datos.

Las peticiones que se realizarán desde cada una de las vistas se harán a través de la utilización del controlador de Spring. Spring es una implementación del patrón de diseño "front controller". Todas las peticiones que se hagan pasarán por este controlador, que será el encargado de gestionar las operaciones a realizar, llamando a la capa de negocio, y de buscar la vista definida para devolver los resultados.

3.4.2 Capa de negocio

Representa la lógica de la aplicación. Recoge peticiones de la capa cliente, las procesa (accediendo a la siguiente capa si fuera necesario) y devuelve una respuesta al cliente.

El modelo de negocio se ha desarrollado con servlets, que reciben petición, se conectan a base de datos si es necesario para realizar las operaciones oportunas, y devuelve los resultados.

3.4.3 Capa de integración o de administración de datos

Gestiona la información localmente persistente de la aplicación. Debe gestionar toda la información necesaria para poder proporcionar las funcionalidades descritas en los casos de uso anteriores. Para llevar a cabo las labores de persistencia se ha optado por utilizar JPA (Java Persistence API). JPA "es un documento en el cual se especifican los principios básicos de gestión de la capa de persistencia en el mundo JEE" (https://www.genbetadev.com/frameworks/jpa-vs-hibernate). Para implementar JPA se ha optado por utilizar el framework Hibernate. Se han creado tres entidades, correspondientes con las tres tablas que se han tratado en la aplicación: TfgUsuario, TfgGranja y TfgVisita. Con este framework se han mapeado fácilmente los campos de las tres tablas de base de datos en nuestro modelo de objetos. Para realizar las operaciones sobre base de datos (inserciones, modificaciones y borrados) se ha generado para cada tabla, una clase propia (GranjaDAO, UsuarioDAO y VisitaDAO), cada una de ellas con su correspondiente interfaz. En caso de querer cambiar la persistencia de datos y utilizar acceso a datos tradicional (utilizando el API JDBC por ejemplo), únicamente habría que cambiar la implementación de estas clases.

La base de datos utilizada es PostgreSQL. Como se comentó en capítulos anteriores, se trata de una base de datos estable, potente, robusta y fácil de administrar, lo que hace que sea una buena elección para el almacenamiento de nuestros datos. En la sección de Anexos de esta memoria se incluye un breve manual de instalación de esta base de datos.

4. Conclusiones

La principal conclusión extraída de la implementación de un producto software pasa por la necesidad de un buen análisis de requisitos. Conocer el grado de alcance del proyecto es fundamental para poder realizar una buena planificación. Siempre es bueno mantener una cierta flexibilidad en todas las fases ya que será prácticamente imposible definir al 100% el producto con los requisitos iniciales y, por tanto, habrá modificaciones que habrá que ir acoplando en el tiempo de la forma menos traumática posible y siempre sin perder el objetivo final.

Para nuestro caso, se ha conseguido mantener la planificación realizada inicialmente. No era un objetivo complicado ya que a lo largo del desarrollo no ha habido cambios en los requisitos ni nos hemos encontrados problemas como productos incompatibles que nos hicieran tener que valorar otras opciones.

Además, el proyecto me ha servido para afianzar determinados conocimientos acerca de los productos instalados. A lo largo de las asignaturas que he cursado del grado se han utilizado Hibernate, Spring, etc. de forma somera. Me gustaría comentar aquí que en mi caso concreto no he realizado el grado completo, si no que he hecho una adaptación de Ing. Técnica en Informática de Gestión a Grado de Informática. De esta forma, sólo he tenido que cursar 6 asignaturas ya que el resto eran convalidaciones. Estoy seguro que si hubiera cursado todas las asignaturas habría profundizado mucho más en varios frameworks, librerías, etc. En cualquier caso, estoy bastante satisfecho con la elaboración de un proyecto completo, dónde he podido poner en práctica e, incluso, aumentar los conocimientos de las herramientas utilizadas.

Finalmente, comentar que el producto desarrollado podría ser utilizado como base para trabajos futuros y adaptaciones a otros modelos de negocio sin ser cambios muy traumáticos. Sin mucho esfuerzo podría utilizarse la aplicación desarrollada para generar otras nuevas: generador de encuestas, control de personal de una empresa, etc.

5. Glosario

- AJAX: Asynchronous Javascript and XML
- API: Application Programming Interface
- AUG: Agile Unified Process
- CU_XX: caso de uso XX
- DOM: Document Object Model
- HTML: Hipertext Mark-up lenguaje
- HTTP: Hypert Text Transfer Protocol
- JEE: Java Enterprise Edition
- JDBC: Java Database Connectivity
- JPA: Java Persistence API
- JRE: Java Runtime Environment
- JSON: JavaScript Object Notation
- JSP: Java Server Pages
- JVM: Java Virtual Machine
- Log4J: Log for Java
- RAD: Rapid Application Development
- SSADM Structured System Analysis and Design Method
- URL: Uniform Resource Locator
- WAR: Web Application Archive
- WWW: World Wide Web
- XML: Extensible Markup Languaje

6. Bibliografía

6.1 Bibliografía clásica

- [Mohammad Akif, Steven Brodhead, Andrei Cioroianu, James Hart, Eric Jung, Dave Writz: 2001]: Java y XML. Referencia para programadores. Ed. Anaya
- [Antonio Martín Sierra, Ramón Egido García: 2016]: Curso avanzado de programación en Java EE. Syncrom Informática
- [Eric van der Vlist, Danny Ayers, Erik Bruchez, Joe Fawcett, Alessandro Vernet: 2007]: Programación Web 2.0. Ed. Anaya
- [Alex Rodríguez Vidal: 2015]: Programación en Java. Uso de Spring Framework. Versión Kindle

6.2 Bibliografía electrónica

https://platzi.com/blog/ajax-con-jquery/
https://jquery.com/
http://w2ui.com/web/
https://poi.apache.org/
https://www.genbetadev.com/frameworks/jpa-vs-hibernate

No se incluye fecha de visita a las páginas mencionadas por haberse tenido que visitar constantemente a lo largo del desarrollo del proyecto. Además, esta documentación electrónica ha sido ampliamente visitada no tanto para el desarrollo de la memoria como a la fase de implementación.

7. Anexos

Se incluyen en este apartado los manuales de instalación de la base de datos y del servidor de aplicaciones.

7.1. Instalación de base de datos PostgreSQL

En este apartado comentaremos los pasos necesarios para la instalación del servidor de base de datos utilizado: PostgreSQL

7.1.1 Descarga del software

Para descargar el software necesario nos hemos conectado a la página del fabricante (www.postgresql.org). En la sección de descargas y una vez registrados hemos descargado la versión 10 del mismo.

7.1.2 Instalación

Después de descargar el software (postgresql-10.2-1-windows-x64.exe), ejecutamos el mismo. Es importante hacerlo con permisos de administrador.

Las principales pantallas que nos encontraremos mientras estamos haciendo la instalación serán las siguientes:

Ilustración 21. Instalación Postgresql. Pantalla inicial

A continuación, seleccionaremos el directorio en el que haremos la instalación. En nuestro caso hemos dejado el directorio que viene por defecto.

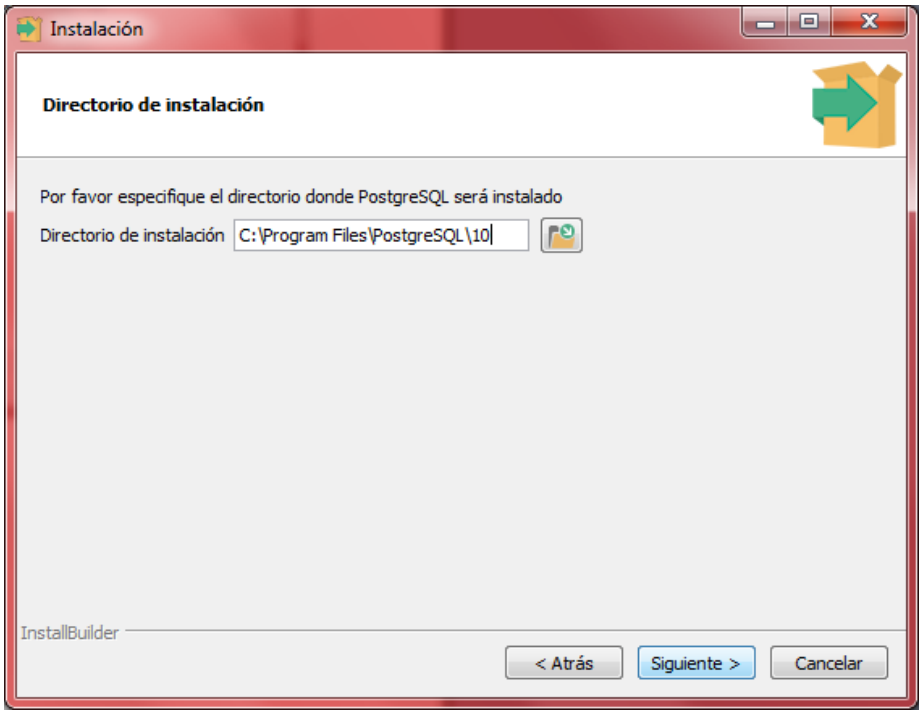


Ilustración 22. Instalación Postgresql. Directorio

Una vez seleccionado el directorio, debemos seleccionar los componentes a instalar. En nuestro caso hemos seleccionado todos. Destacar la opción pgAdmin4 que nos será de mucha utilidad a la hora de configurar la base de datos.

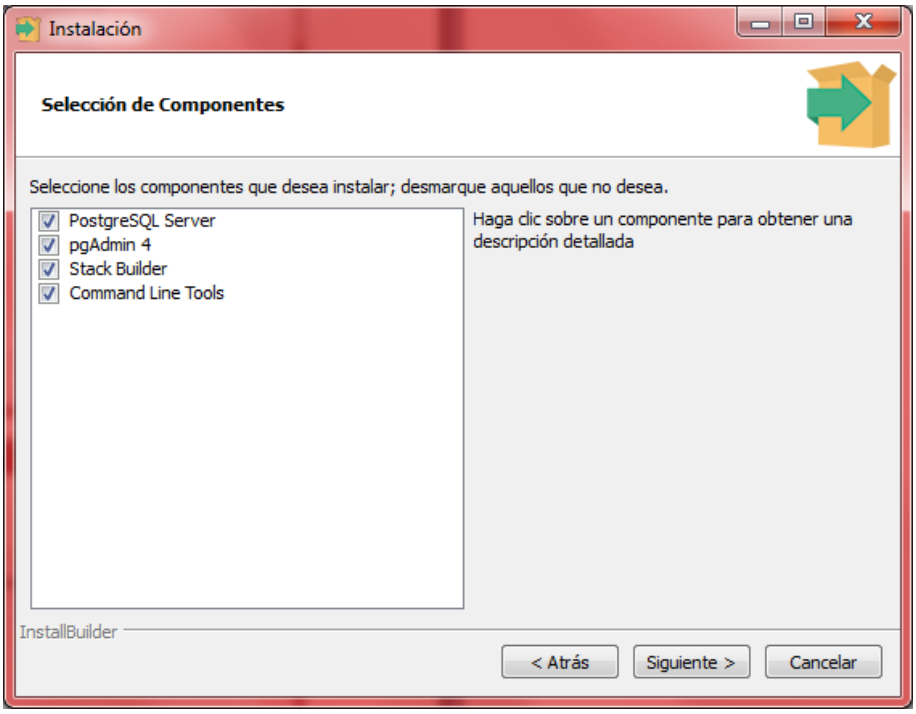


Ilustración 23. Instalación Postgresql. Selección componentes

A continuación, seleccionaremos el directorio en el que el servidor almacenará los datos de la base de datos. En este caso, también hemos dejado la opción por defecto.

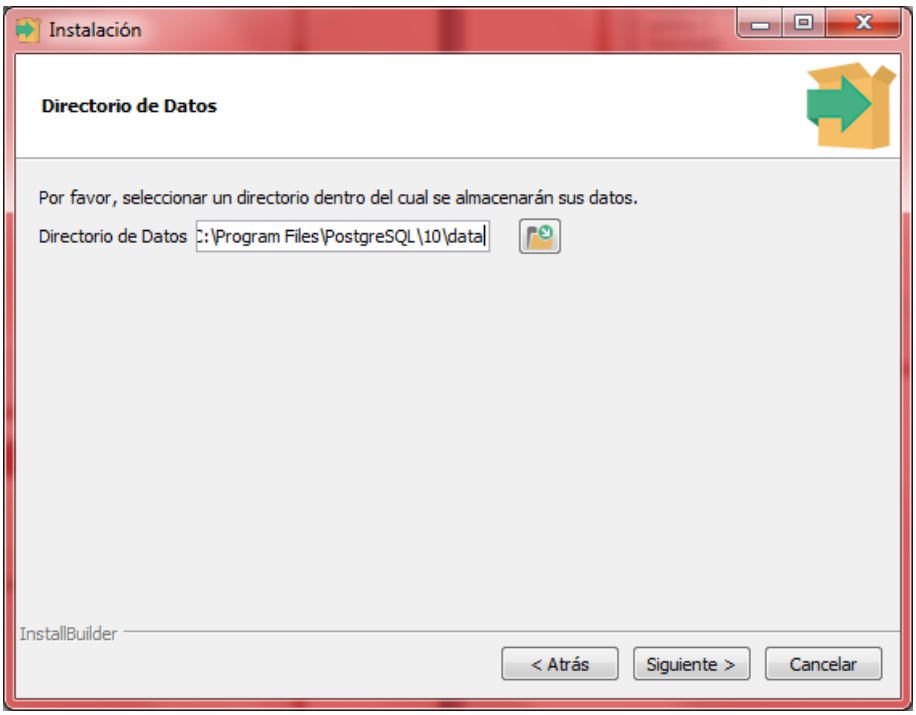


Ilustración 24. Instalación Postgresql. Directorio datos

Seguidamente tendremos que introducir un nombre de usuario y contraseña para conectarnos al servidor PostgreSQL. En nuestro caso hemos utilizado la combinación uoc/uoc, aunque sería conveniente utilizar una password más segura.

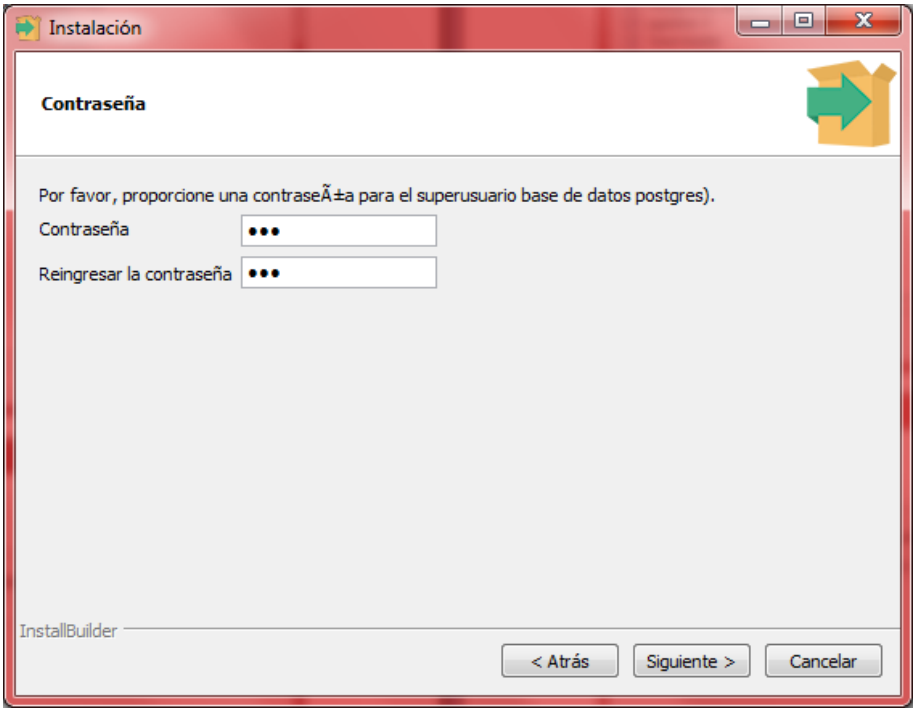


Ilustración 25. Instalación Postgresql. Usuario/Password

En las siguientes dos pantallas podremos seleccionar el puerto en el que tendremos que conectarnos para acceder al servidor y la configuración regional utilizada. En ambos casos dejamos las que nos propone la aplicación.

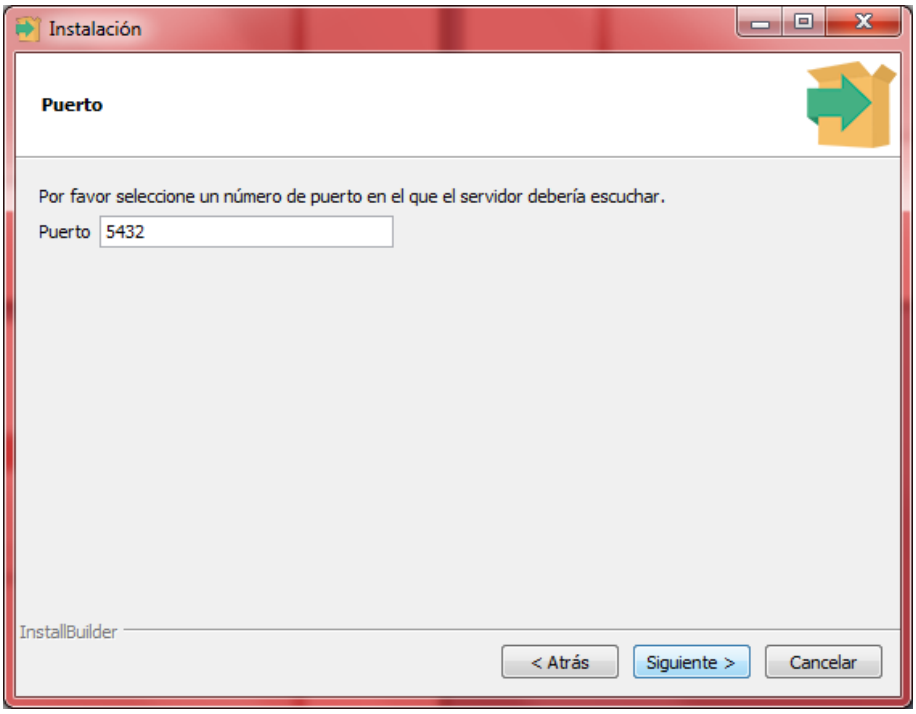


Ilustración 26. Instalación Postgresql. Puerto

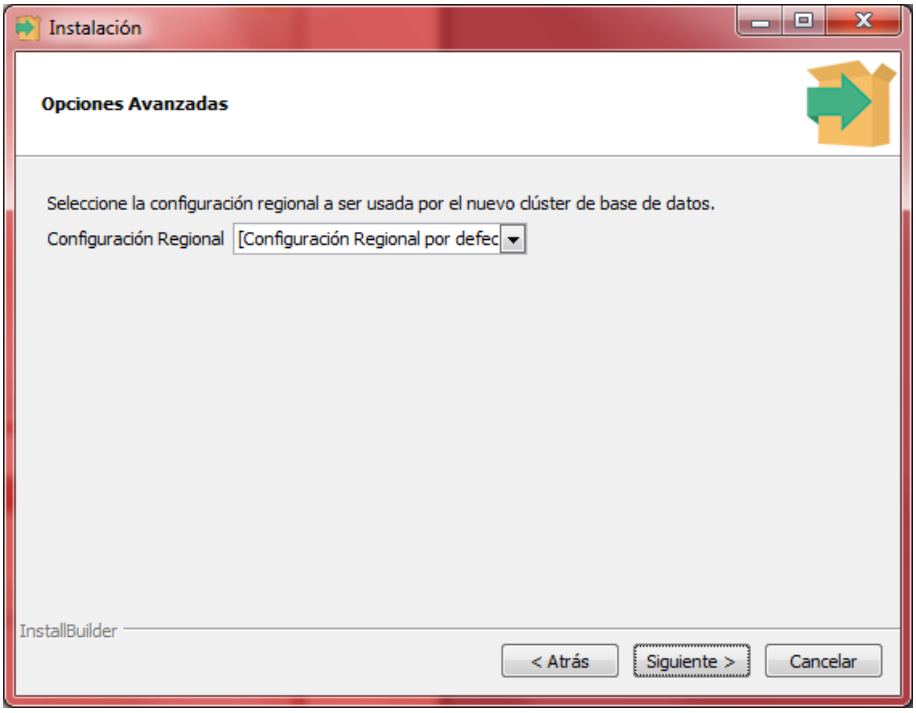


Ilustración 27. Instalación Postgresql. Configuración regional

Después de seleccionar las opciones anteriores, empezará la instalación del producto. Si todo ha ido correctamente se mostrará una pantalla informando de ello.

Ilustración 28. Instalación Postgresql. Fin instalación

7.1.3 Configuración PostgreSQL

Para la configuración de PostgreSQL utilizaremos la herramienta pgAmin4 que hemos instalado previamente. Para que funcione correctamente es fundamental que la ejecutemos con permisos de administrador.

Una vez ejecutado se mostrará la siguiente pantalla:

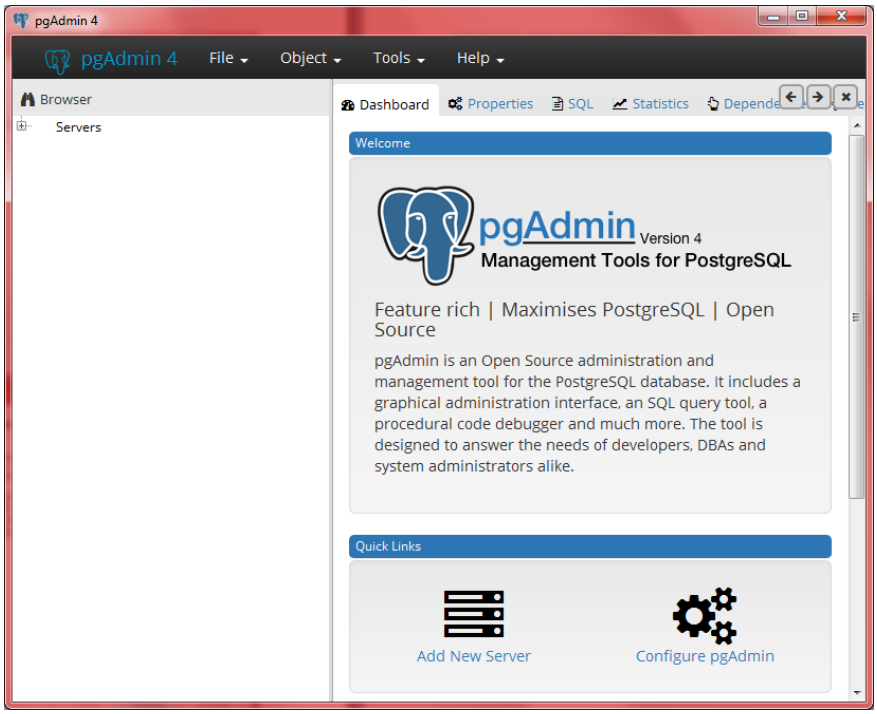


Ilustración 29. Configuración Postresql. PGAdmin4

Para conectarnos habrá que pinchar sobre la opción “Servers” e introducir el nombre de usuario y la password que se seleccionó cuando se instaló la aplicación.

El objetivo de este punto no es explicar pasa a paso la configuración de la misma. Para ello se adjunta en un fichero a parte el script necesario que hay que ejecutar para la creación de la base de datos: esquema, usuario, tablas, etc.

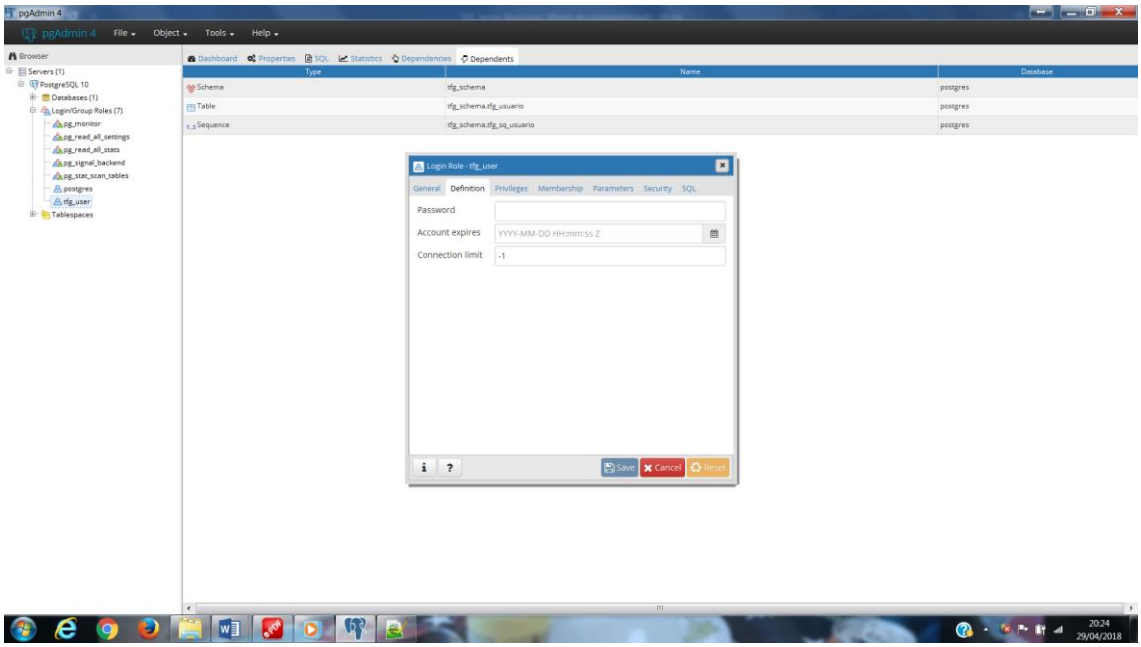


Ilustración 30. Configuración Postgresql. Configurar password usuario bbdd

7.2 Scripts de creación de base de datos

Para la creación de la base de datos, así como de los datos iniciales necesarios para que la aplicación funcione correctamente, se deberá ejecutar mediante scripts. Dichos script se han incluido en una carpeta con el mismo nombre (CodigoFuente.zip\tfgSaluPig\src\main\resources\script\) dentro del fichero adjunto que contiene el código fuente de la aplicación.

Podremos hacerlo desde la herramienta PGAdmin4. Una vez arrancada la aplicación, seleccionaremos la base de datos que viene creada por defecto “postgresql” y desde el submenú “Query Tools” que se encuentra en el menú “Tools” podremos copiar el script y pegarlo la ventana correspondiente.

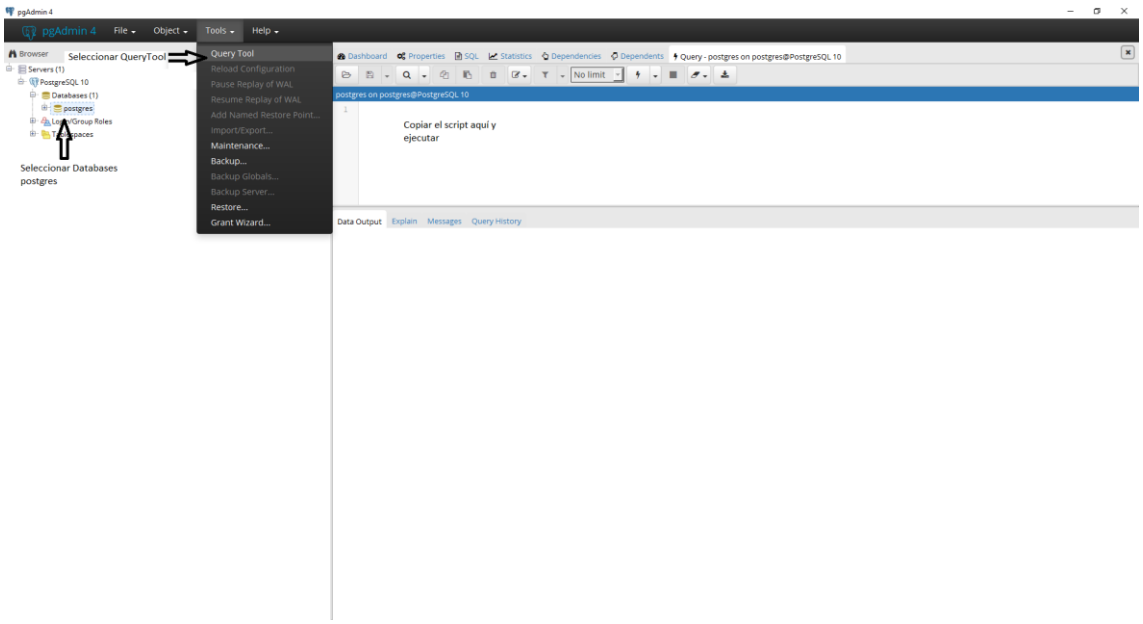


Ilustración 31. Ejecución script de base de datos

Este script se encarará de crear tanto usuario de base de datos, como esquema, tablas, etc…

Para la inserción de los datos iniciales de la aplicación (referentes a usuarios) habrá que importar el fichero tfg_usuario. Además, si queremos introducir una serie de datos con los que poder hacer pruebas, también deberemos importar el fichero tfg_granja (datos de granjas) y el fichero tfg_visita (datos de varias visitas creadas aleatoriamente utilizando la función random de postgresql).

7.3 Instalación de servidor Tomcat

En este apartado explicaremos los puntos principales para la instalación del servidor que albergará nuestra aplicación.

7.3.1 Descarga del software

La descarga necesaria del software se ha hecho desde la página del fabricante, en su opción de descargas (https://tomcat.apache.org/download-90.cgi). Una vez concluida la descarga se debe ejecutar con permisos de administrador apache-tomcat-9.0.5.exe.

7.3.2 Instalación del software

La instalación es muy sencilla y consta de los siguientes pasos:

Ilustración 32. Instalación Tomcat. Bienvenida

Pulsando “next” tendremos que aceptar la licencia de apache. Una vez hecho esto pasaremos a la selección de los componentes a instalar. Será suficiente con dejar los que vienen por defecto

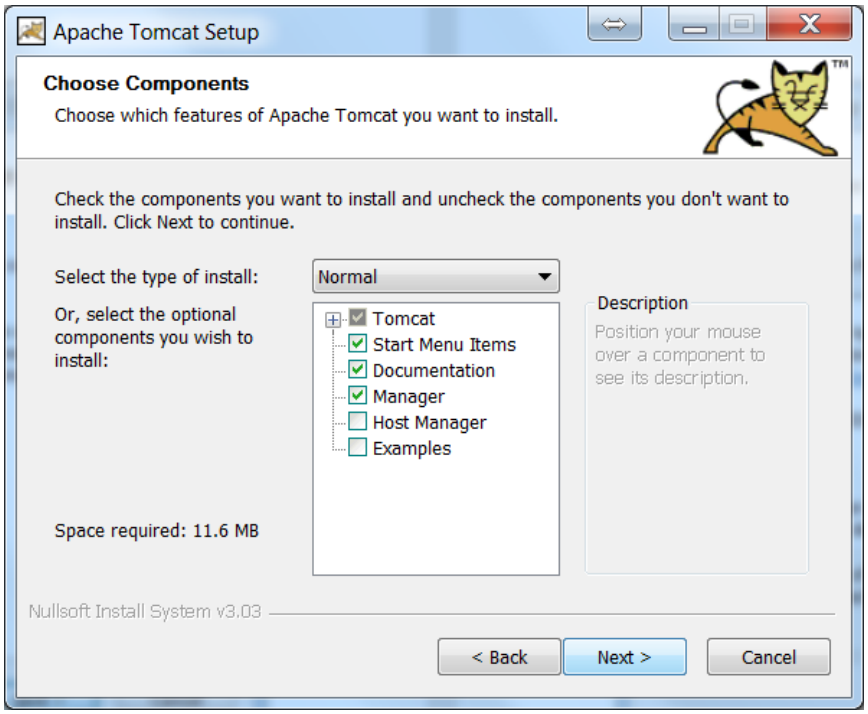


Ilustración 33. Instalación Tomcat. Selección componentes

A continuación, tendremos que seleccionar opciones tales como el puerto de escucha, nombre del servicio que se creará en el sistema operativo, nombre de usuario y password del administrador.

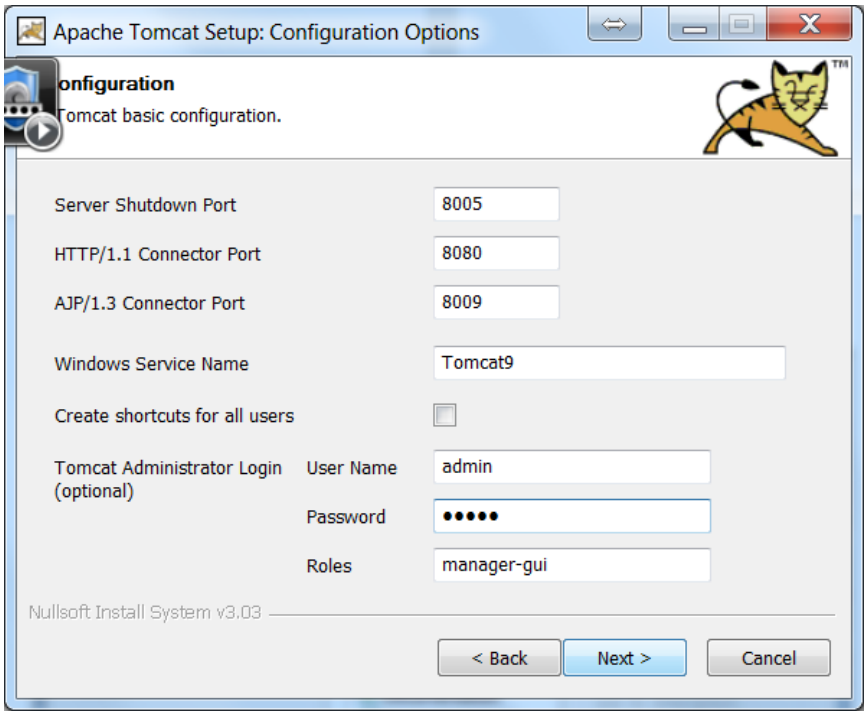


Ilustración 34. Instalación Tomcat. Configuración

Para la ejecución de Tomcat será necesario una máquina virtual de Java. Como será posible que existan varias instaladas, la instalación nos permite seleccionar con cuál se ejecutará.

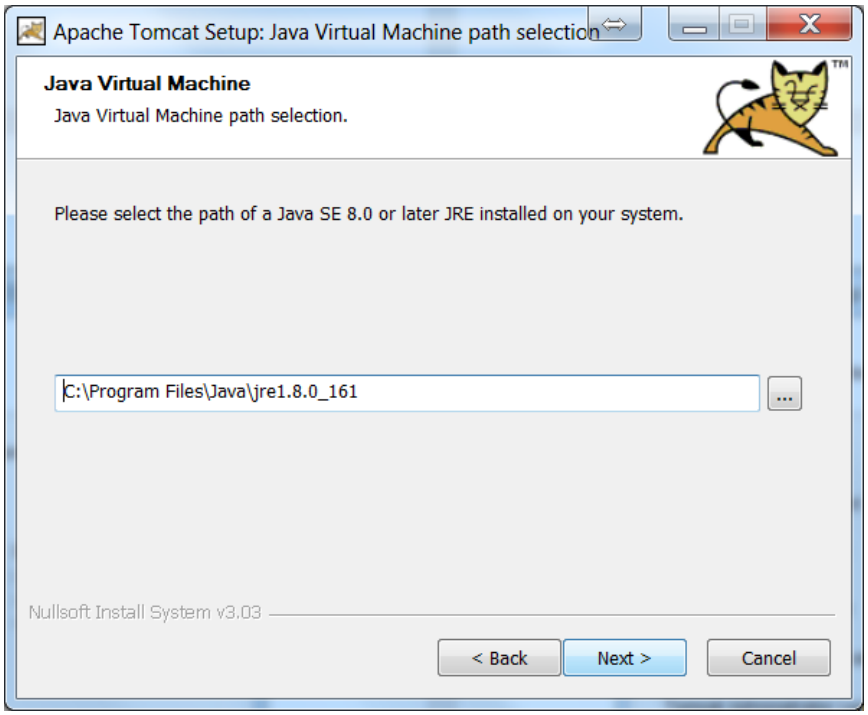


Ilustración 35. Instalación Tomcat. Máquina virtual Java

En el paso siguiente habrá que seleccionar el directorio en el que se realizará la instalación. Es importante acordarse del directorio introducido ya que será necesario para saber dónde tendremos que hacer el despliegue de la aplicación.

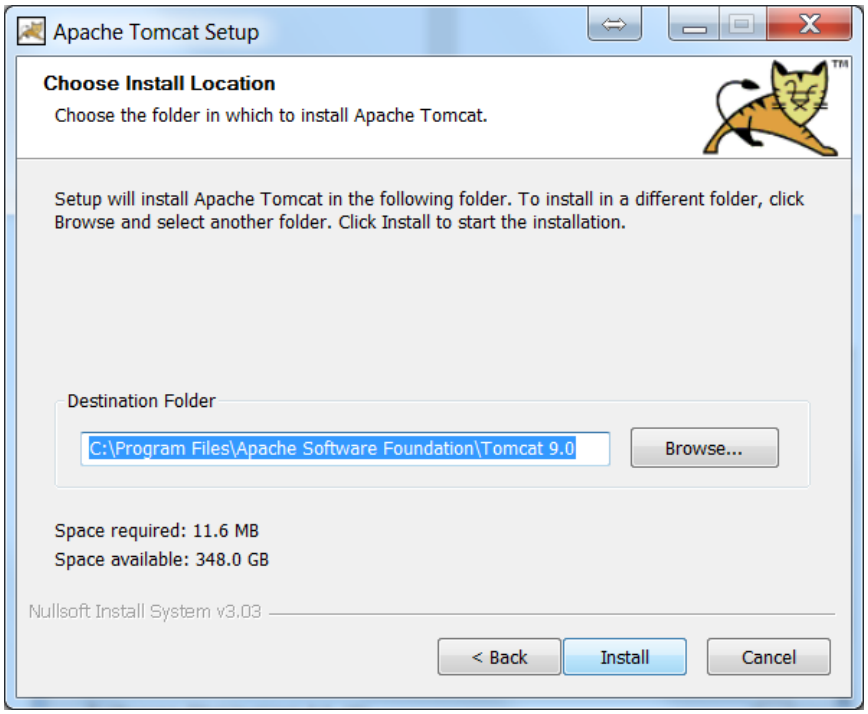


Ilustración 36. Instalación Tomcat. Directorio

Una vez seleccionado el directorio de instalación comenzará el proceso de instalación. Si todo ha ido correctamente se mostrará pantalla informando de ello:

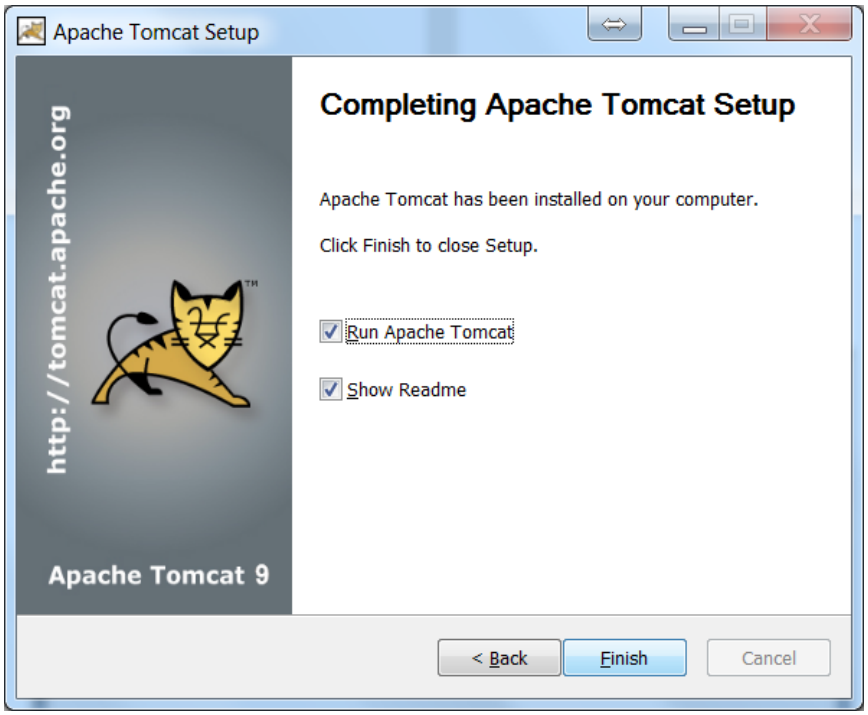


Ilustración 37. Instalación Tomcat. Fin instalación

7.3.3 Configuración básica

Tomcat es un servidor muy sencillo que para nuestro caso no ha necesitado configuración especial. Durante la instalación se han creado varios directorios:

- Bin: ficheros binarios necesarios para la ejecución del servidor. En este punto destacar los dos que nos harían falta para levantar el servidor (startup.bat) y para bajarlo (shutdown.bat). En cualquier caso, durante la instalación se han creado un servicio de sistema operativo
- Conf: archivos de configuración del servidor. Aquí podremos modificar opciones tales como máquina virtual utilizada, usuarios, roles, memoria utilizada por el servidor, puertos de escucha, etc. Es el primer directorio que lee al arrancar y se basa en su contenido para cargar el resto de opciones.
- Lib: contiene todos los jar necesarios para la ejecución del servidor
- Logs: directorio donde el servidor escribirá todo lo que le va pasando.
- Temp: directorio temporal inicialmente vacío
- Webapps: aplicaciones desplegadas en el servidor. Para el despliegue de nuestra aplicación será suficiente con copiar el fichero war generado con Maven en este directorio. Una vez copiado se iniciará automáticamente el despliegue de la misma. Es un proceso que no llevará más de un minuto y nuestra aplicación estará disponible para ser ejecutada.
- Work: directorio de trabajo del servidor. Lo usan los JSP para compilarse y convertirse en servlets.

7.4 Despliegue y configuración: TfgSaluPig.properties y log4j.properties

Como se ha comentado anteriormente para el despliegue de la aplicación únicamente será necesario copiar el war que se incluye en la entrega en la carpeta webapps del directorio de instalación de Tomcat.

En el fichero war incluido se han utilizado dos ficheros de configuración. En el fichero TfgSaluPig.properties tiene los siguientes valores:

```
#configuracion listado excel
xls_ruta_generado=c:\\temp\\tfgSaluPig\\
xls_nombre_generado=visitas_generado.xls
```

```
#base de datos
bbdd.url=jdbc:postgresql://localhost:5432/postgres
bbdd.user=tfg_user
bbdd.password=tfg_password
bbdd.driver=org.postgresql.Driver

#Estados posibles de los usuarios
NumEstadosUsuario=2
Estado_0=Baja
Estado_1=Alta

NumRoles=2
Rol_0=Administrador
Rol_1=Visitante

#Estados posibles de las granjas en base de datos
NumEstadosGranja=2
EstadoGranja_0=Baja
EstadoGranja_1=Alta

#Estados posibles de las visitas a una granja
NumEstadosVisita=2
EstadoVisita_0=Baja
EstadoVisita_1=Alta
```

En este fichero se definen los parámetros necesarios para la exportación de datos de visitas a un fichero Excel. Inicialmente se guardarán en un fichero en el servidor que, a continuación, se enviará al navegador del usuario que ha solicitado la información.

Además, en este fichero se recogen los parámetros necesarios para conectarse a la base de datos. Además, se han definido aquí los distintos roles que pueden existir en la aplicación, así como estados para usuarios (alta y baja), estados para granjas (alta y baja) y estados para visitas (alta o baja).

Con respecto al log de aplicaciones los parámetros predefinidos son los siguientes:

```
log4j.logger.org.hibernate=error, ARCHIVO
log4j.logger.com.ddp=debug,ARCHIVO

### Para dirigir mensajes a un archivo ###
log4j.appender.ARCHIVO=org.apache.log4j.RollingFileAppender
log4j.appender.ARCHIVO.File=c:/temp/tfgSaluPig/logs/saluPig.log
log4j.appender.ARCHIVO.MaxFileSize=2000KB
log4j.appender.ARCHIVO.MaxBackupIndex=10
log4j.appender.ARCHIVO.Append=true
log4j.appender.ARCHIVO.layout=org.apache.log4j.PatternLayout
log4j.appender.ARCHIVO.layout.ConversionPattern=%d %p %t %c - %m%n en
```

Como puede verse, los ficheros de log se alojarán en la carpeta c:\temp\tfgSaluPig\logs. En dicha carpeta se irá creando ficheros de log en modo debug para la aplicación y error para las clases de Hibernate. Hemos tenido en cuenta que se trata de un entorno de desarrollo y/o en una fase inicial de producción. Más adelante, habrá que cambiar este parámetro a modo error, de forma que reduciremos el espacio de almacenamiento necesario para log de aplicación.

En ambos casos pueden modificarse los ficheros sin necesidad de hacer un despliegue nuevo. Para ello, habrá que ir al directorio de instalación de Tomcat, y dentro de la carpeta webapps encontraremos un directorio tfgSaluPig. Si navegamos en este directorio entrando en WEB-INF\classes podremos encontrar los ficheros de configuración y modificarlos a nuestra necesidad, por ejemplo, para guardar los ficheros de log en otro directorio o cambiar el usuario y la contraseña con el que nos conectaremos a la base de datos.

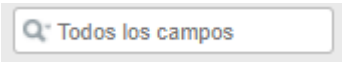
7.5 Implementación. Aspectos destacados

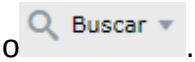
Como se ha comentado anteriormente, en la implementación de la aplicación se ha utilizado un framework denominado w2ui. Dicho framework ofrece diversas funcionalidades basadas en jQuery: pestañas, formularios, layouts, popups y especialmente grid de datos.

Me gustaría destacar en este punto, que el grid de datos proporciona una potente herramienta de búsqueda sobre los datos que tiene cargados. La comunicación entre el componente y nuestro modelo de datos para la carga de datos se hace a través de un servlet, el cuál se encarga de recoger las peticiones que se hacen en la vista (a través del grid), procesarlas según la acción solicitada (nuevo registro, baja de registro y consulta) y devolver los resultados al grid. Todo este intercambio de información se hace mediante intercambios de datos en formato JSON. Cada uno de los grid implementados en la aplicación, usuarios, granjas y visitas, tiene asociado un servlet (MantUsuariosServlet, MantGranjasServlet y MantVisitasServlet). Cada uno de estos servlet, recibe las peticiones, “descifra” el contenido JSON que recibe, realiza las acciones necesarias (inserción, consulta, modificación) de nuevos

usuarios, granjas y/ visitas, y “cifra” el resultado para enviarlo como respuesta a la vista donde se dibuja el grid.

Además, hay que destacar que el grid configurado ofrece muy diversas operaciones que nos permitirá hacer una explotación importante de los datos recogidos en ellos. Algunas de ellas:

1. Ordenación: se podrá ordenar por cada una de las columnas del grid haciendo click en el título de la columna. Se permite orden ascendente y descendente.
2. Mostrar/ocultar las columnas que se desee en cada momento. Para ello, habrá que seleccionar el icono  y se desplegará la lista de las columnas que se presentan.
3. Agregar nuevos registros. Pulsando sobre Agregar Nuevo se abrirá una ventana con los datos a rellenar
4. Búsqueda simple. Si se introduce cualquier texto en el recuadro 

, se hará una búsqueda OR en todas las columnas del grid. En caso de querer buscar únicamente sobre una de las columnas, habrá que pulsar la lupa que aparece dentro de este recuadro y a continuación introducir el texto buscado. Los resultados dónde coincida el texto buscado aparecerán sombreados en amarillo.
5. Búsqueda compleja. En muchas ocasiones será necesario afinar un poco más la búsqueda y únicamente filtrar por alguna o varias columnas pero no todas. También puede ser necesario buscar en un rango de valores o fechas. Para ello se presenta el icono 

. Este icono mostrará otra ventana con todos los campos del grid pudiendo introducir un texto a buscar para cada uno de ellos. En este caso la búsqueda será “AND” entre todas las columnas introducidas, resaltándose igualmente en amarillo el texto encontrado. Para campos de tipo texto, se permitirá búsquedas donde encuentra el literal, que comience, que termine o que contenga el texto introducido. Además busca tanto mayúsculas como minúsculas. Para campos numéricos, se podrá buscar por valores iguales, mayores,

menores o entre dos valores. No se permite meter valores no numéricos en este tipo de campos. Para campos de tipo fecha, se permiten búsquedas igual, mayor, menor o entre dos rangos de fechas. Si el valor de la fecha introducida no es correcto el campo se pondrá en blanco.

Gráficamente, la comunicación entre la vista del grid y el modelo de datos será:

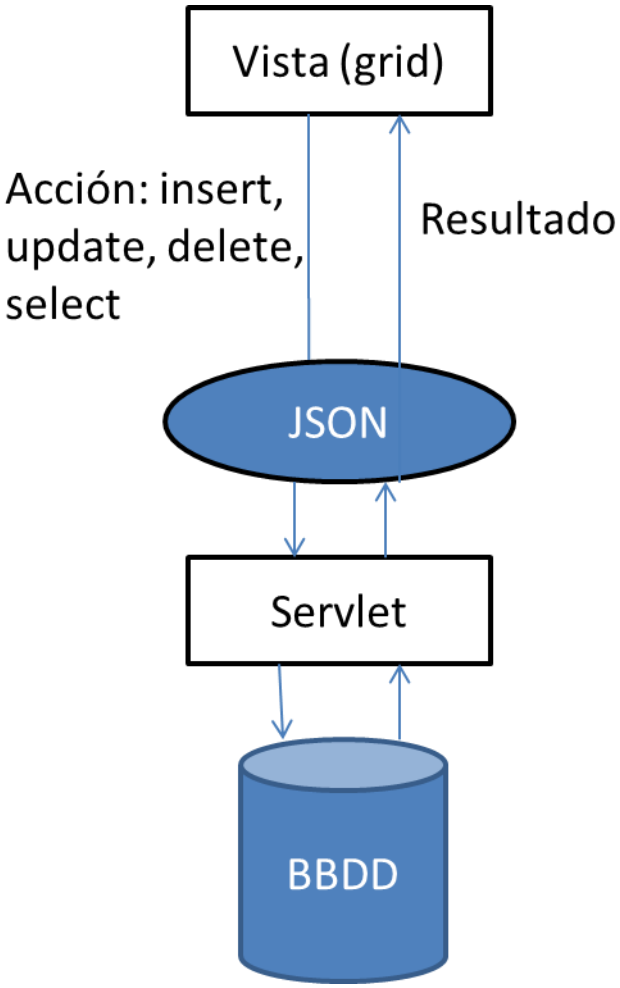


Ilustración 38. Implementación w2ui

Algunos ejemplos de intercambios JSON:

1. Petición de lista de usuarios registrados ordenados por nombre

```
{"offset":0,"limit":0,"cmd":"get","sort":[{"field":"nombreCompleto","directio
n":"ASC"}],"selected":[]}
```

2. Resultado de la consulta de de usuarios registrados ordenados por nombre


```
{"total":2,"records":[{"password":"jra","estado":"1","e_mail":"jrodriguezalb@uoc.edu","usuario":"jra","nombreCompleto":"Javier Rodríguez Albarrán","telefono":"666 66 66 66","recid":68,"rol":"0"},{"password":"visitante","estado":"1","e_mail":"jrodriguezalb@uoc.edu","usuario":"visitante","nombreCompleto":"Javier Rodríguez Albarrán Visitante","telefono":"777 77 77 77","recid":69,"rol":"1"}],"status":"success"}
```


3. Petición de lista de granjas visitadas desde el 1 de enero de 2018:


```
{"search":[{"field":"fecha","type":"date","value":"01/01/2018","operator":"more"}],"offset":0,"limit":100,"cmd":"get","searchLogic":"AND","sort":[{"field":"idgranja","direction":"ASC"}],"selected":[]}
```


4. Resultado de la lista de granjas visitadas en el año 2018:


```
{"total":1,"records":[{"estado":"0","numFallecimientos":6,"numNacimientos":2,"idgranja":"Nombre de la Granja","diseñoCorrales":"Tiene un buen diseño","numBajasSs":0,"numHembra":35,"fecha":"17/06/2018","satisfaccionTrabajadores":7,"numMacho":15,"observaciones":"Ojo. VAlorar que puede pasar. Hay muchos fallecidos. Mirar siguiente visita","numTrabajadores":3,"metrosCuadrados":1250,"recid":3550}],"status":"success"}
```


5. Modificación de una visita, cambiando el campo número de trabajadores:


```
{"search":[{"field":"fecha","type":"date","value":"01/01/2018","operator":"more"}],"offset":0,"limit":100,"changes":[{"numTrabajadores":34,"recid":3550}],"cmd":"save","searchLogic":"AND","sort":[{"field":"idgranja","direction":"ASC"}],"selected":[3550]}
```


6. Resultado de la operación de actualización anterior:


```
{"status":"success"}
```