

UNIVERSIDAD CENTRAL DEL ECUADOR



**FACULTAD DE INGENIERÍA, CIENCIAS FÍSICAS Y MATEMÁTICA
CARRERA DE INGENIERÍA INFORMÁTICA**

**SISTEMA DE REGISTRO DE ANIMALES Y SUS DERIVADOS DEL
CENTRO EXPERIMENTAL UYUMBICHO DE LA FACULTAD DE
VETERINARIA DE LA UNIVERSIDAD CENTRAL DEL ECUADOR**

**TRABAJO DE GRADUACIÓN PREVIO A LA OBTENCIÓN DEL TÍTULO
DE INGENIERO INFORMÁTICO**

AUTORES:

NARCISA DEL PILAR CHISAGUANO CAIZAPANTA

ANA YADIRA NARANJO QUINGAÍZA

TUTOR: ING. JÉFFERSON BELTRÁN MSc

QUITO – ECUADOR

2012



UNIVERSIDAD CENTRAL DEL ECUADOR

Sistema de Registro de Animales y sus Derivados del Centro Experimental Uyumbicho

CERTIFICACIÓN

Certificamos que el trabajo de graduación: “**SISTEMA DE REGISTRO DE ANIMALES Y SUS DERIVADOS DEL CENTRO EXPERIMENTAL UYUMBICHO DE LA FACULTAD DE VETERINARIA DE LA UNIVERSIDAD CENTRAL DEL ECUADOR**”, es original y ha sido desarrollado por las señoritas Narcisa Chisaguano y Yadira Naranjo, bajo nuestra dirección y conforme a todas las revisiones realizadas.

Ing. Jéfferson Beltrán MSc

DIRECTOR DE TESIS

Ing. YasminaAtarihuanaMSc

MIEMBRO DEL TRIBUNAL

Ing. Santiago Morales MSc

MIEMBRO DEL TRIBUNAL

Quito.- UC. Facultad de Ingeniería Ciencias Físicas y Matemática.



AUTORIZACIÓN DE LA AUTORÍA INTELECTUAL

Yo, CHISAGUANO CAIZAPANTA NARCISA DEL PILAR, NARANJO QUINGAIZA ANA YADIRA en calidad de autoras del trabajo de investigación o tesis realizada sobre: **SISTEMA DE REGISTRO DE ANIMALES Y SUS DERIVADOS DEL CENTRO EXPERIMENTAL UYUMBICHO DE LA FACULTAD DE VETERINARIA DE LA UNIVERSIDAD CENTRAL**, por la presente autorizamos a la UNIVERSIDAD CENTRAL DEL ECUADOR, hacer uso de todos los contenidos que me pertenecen o de parte de los que contiene esta obra, con fines estrictamente académicos o de investigación.

Los derechos que como autor me corresponden, con excepción de la presente autorización, seguirán vigentes a mi favor, de conformidad con lo establecido en los artículos 5, 6, 8, 19 y demás pertinentes de la Ley de Propiedad Intelectual y su Reglamento.

Quito, 04 de abril de 2012

Narcisa ChisaguanoCaizapanta

C.I: 171556394-4

Ana Yadira Naranjo

C.I: 171542080-6



DEDICATORIA

Esta tesis la dedico a mi hija, ya que quiero que ella valore mi esfuerzo en alcanzar una carrera profesional, la misma que permitirá mejorar mi calidad de vida familiar, realizarme como persona y contribuir al desarrollo del país.

Yadira



DEDICATORIA

Mi tesis la dedico, a ti Dios que me diste la oportunidad de vivir y regalarme una familia maravillosa.

Con mucho cariño a mis Padres que me dieron la vida y han estado conmigo en todo momento. Gracias por darme una profesión para mi futuro y por creer en mí, aunque hemos pasado momentos difíciles siempre han estado apoyándome y brindándome todo su amor, por todo esto les agradezco de todo corazón el que estén a mi lado.

A mi hijita Brithany quien es la fuerza y el motivo para seguir superándome y luchando por mis metas.

Narcisa



AGRADECIMIENTO

A Dios quien me proporciona sabiduría, inteligencia y fuerza para desempeñarme en mi trabajo diario y el desarrollo de esta tesis, a mis padres, esposo y hermanos que me brindaron su apoyo incondicional, amor y compresión, a mi hija que constituye la razón de mi superación y el motivo por alcanzar mis metas.

A la Universidad Central del Ecuador templo del saber y de grandes virtudes por abrirme sus puertas, a los maestros que me brindaron una formación integral para la vida.

A la Facultad de Veterinaria y Zootecnia y en especial al Centro de Experimental Uyumbicho quienes nos proporcionaron la información necesaria para el desarrollo del sistema.

Yadira



AGRADECIMIENTO

En primer lugar a Dios y a la Virgen del Quinche por haberme guiado durante estos años de carrera; en segundo lugar a mis PADRES, mis hermanos y a mi Esposo; por siempre haberme dado su fuerza y apoyo incondicional que me han ayudado a llegar hasta donde estoy ahora.

Agradezco a mis profesores por haberme impartido el conocimiento, a las personas quienes nos brindaron todo su apoyo para el desarrollo de esta tesis.

Por último a la Universidad Central del Ecuador y en especial a la Facultad de Ingeniería que me supo abrir las puertas para seguir mi carrera profesional.

Narcisa



CONTENIDO

CAPÍTULO I

	INTRODUCCIÓN	1
1.1	PLANTEAMIENTO DEL PROBLEMA.	2
1.1.1	ÁREA BOVINA	3
1.1.2	ÁREA DE CUYES	4
1.1.3	ÁREA AVÍCOLA	5
1.1.4	ÁREA PORCINA	6
1.1.5	AREA DE APICULTURA	7
1.2	INTERROGANTES DE LA INVESTIGACIÓN	8
1.3.	OBJETIVOS	8
1.3.1	OBJETIVO GENERAL	8
1.3.2	OBJETIVOS ESPECÍFICOS	9
1.4	JUSTIFICACIÓN	9

CAPÍTULO II

2.	REVISIÓN BIBLIOGRÁFICO	11
2.1	ANTECEDENTES	11
2.2	FUNDAMENTACIÓN TEÓRICA	11
2.2.1	DEFINICIÓN DE JAVA	11
2.3.	ARQUITECTURA JAVA ENTERPRISE JEE6	12
2.3.1	TECNOLOGÍA WEB JSF	15
2.3.2	PRIMEFACES	17
2.3.3.	SERVIDOR DE APLICACIONES JBOSS	17
2.3.3.1	JBOSS 7	18
2.3.4	JAVA PERSISTENCE API	19
2.3.5	HIBERNATE	20
2.3.6	CONTENEDOR ENTERPRISE JAVA BEAN	21
2.3.7	BASE DE DATOS	22
2.3.7.1	POSTGRESQL	22
2.3.8	JASPER REPORTS	23



UNIVERSIDAD CENTRAL DEL ECUADOR
Sistema de Registro de Animales y sus Derivados del Centro Experimental Uyumbicho

2.4	MODELO DE UML	23
2.4.1	TIPOS DE DIAGRAMAS UML	24
2.4.1.1	DIAGRAMA DE CASO DE USO	25
2.4.1.2	DIAGRAMA DE CLASES	25
2.4.1.3	DIAGRAMA DE OBJETOS	26
2.4.1.4	DIAGRAMA DE ESTADOS	25
2.4.1.5	DIAGRAMA DE SECUENCIAS	26
2.4.1.6	DIAGRAMA DE COLABORACIÓN	26
2.4.1.7	DIAGRAMA DE ACTIVIDAD	27
2.4.1.8	DIAGRAMA DE COMPONENTES	27
2.4.1.9	DIAGRAMA DE DISTRIBUCIÓN	27
2.5	METODOLOGÍA	28
2.5.1.	MODELO EN ESPIRAL	28
2.5.2	PARADIGMA DEL MODELO EN ESPIRAL	29
2.5.3	FASES DEL MODELO EN ESPIRAL	29



CAPÍTULO III

3.1	ACTORES QUE INTERVIENEN EN EL NEGOCIO	31
3.2	ACTORES QUE INTERVIENEN EN EL SISTEMA	31
3.3	DIAGRAMA GENERAL DE LOS CASOS DE USO DEL ADMINISTRADOR	32
3.4.	CASO DE USO DEL ADMINISTRADOR	34
3.4.1	DIAGRAMA DE SECUENCIA DEL ADMINISTRADOR	34
3.4.2	CASO DE USO DEL MANTENIMIENTO DE LA INFORMACIÓN	36
3.4.2.1	DIAGRAMA DE SECUENCIA MANTENIMIENTO DE LA INFORMACIÓN	36
3.4.3	CASO DE USO GESTIONAR USUARIOS, PERFILES, PERMISOS	38
3.4.3.1	DIAGRAMA DE SECUENCIA GESTIONAR USUARIOS	38
3.4.4	CASO DE USO DE REGISTRO DE PRODUCCIÓN DE HUEVOS	40
3.4.4.1	DIAGRAMA DE SECUENCIA DE PRODUCCIÓN DE HUEVOS	40
3.4.5	CASO DE USO DEL REGISTRO DE PRODUCCIÓN LECHERA	42
3.4.5.1	DIAGRAMA DE SECUENCIA DE PRODUCCIÓN LECHERA	42
3.4.6	CASO DE USO DEL REGISTRO DE PRODUCCIÓN MIEL Y POLEN	44
3.4.6.1	DIAGRAMA DE SECUENCIA DEL REGISTRO DE PRODUCCIÓN MIEL Y POLEN	44
3.4.7	CASO DE USO DEL REGISTRO DE PRODUCTOS DERIVADOS	46



UNIVERSIDAD CENTRAL DEL ECUADOR
Sistema de Registro de Animales y sus Derivados del Centro Experimental Uyumbicho

3.4.7.1	DIAGRAMA DE SECUENCIA DE PRODUCTOS DERIVADOS	46
3.4.8	CASO DE USO DEL REGISTRO DE VENTA DE ANIMALES	48
3.4.8.1	DIAGRAMA DE SECUENCIA DEL REGISTRO DE VENTA DE ANIMALES	48
3.4.9	CASO DE USO DE REGISTRO DE CONSUMO DE LA PRODUCCIÓN	50
3.4.9.1	DIAGRAMA DE SECUENCIA DEL REGISTRO DE CONSUMO DE LA PRODUCCIÓN	50
3.4.10	CASO DE USO DEL REGISTRO DE LA VENTA DEL PRODUCTO	52
3.4.10.1	DIAGRAMA DE SECUENCIA DE LA VENTA DEL PRODUCTO	52
3.4.11	CASO DE USO REALIZAR REPORTES	54
3.4.11.1	DIAGRAMA DE SECUENCIA REALIZAR REPORTES	54
3.5	CASO DE USO GENERAL ANIMALES INDIVIDUALES	56
3.5.1	DIAGRAMA DE SECUENCIA GENERAL ANIMALES INDIVIDUALES	57
3.5.2	CASO DE USO DE REGISTRO DE INFORMACIÓN DE ANIMALES INDIVIDUALES	59
3.5.2.1	DIAGRAMA DE SECUENCIA DE REGISTRO DE INFORMACIÓN ANIMALES INDIVIDUALES	59
3.5.3	CASO DE USO DEL REGISTRO DE INSEMINACIÓN DE ANIMALES INDIVIDUALES	61
3.5.3.1	DIAGRAMA DE SECUENCIA DEL REGISTRO DE INSEMINACIÓN ANIMALES INDIVIDUALES	61
3.5.4	CASO DE USO DEL REGISTRO SANITARIO ANIMALES INDIVIDUALES	63
3.5.4.1	DIAGRAMA DE SECUENCIA DEL REGISTRO SANITARIO ANIMALES INDIVIDUALES	63
3.5.5	CASO DE USO DEL REGISTRO DE VACUNAS ANIMALES INDIVIDUALES	65



UNIVERSIDAD CENTRAL DEL ECUADOR
Sistema de Registro de Animales y sus Derivados del Centro Experimental Uyumbicho

3.5.5.1	DIAGRAMA DE SECUENCIA DEL REGISTRO DE VACUNAS ANIMALES INDIVIDUALES	65
3.5.6	CASO DE USO DEL REGISTRO DE MUERTE ANIMALES INDIVIDUALES	67
3.5.6.1	DIAGRAMA DE SECUENCIA DEL REGISTRO DE MUERTE ANIMALES INDIVIDUALES	67
3.5.7	CASO DE USO DEL REGISTRO DE PARTOS Y CRÍAS	69
3.5.7.1	DIAGRAMA DE SECUENCIA DE PARTOS Y CRÍAS ANIMALES INDIVIDUALES	69
3.5.8	CASO DE USO DEL REGISTRO DE PESOS ANIMALES INDIVIDUALES	71
3.5.8.1	DIAGRAMA DE SECUENCIA DE PESOS ANIMALES INDIVIDUALES	71
3.5.9	CASO DE USO DEL REGISTRO DE SEMEN PORCINOS (ANIMALES INDIVIDUALES)	73
3.5.9.1	DIAGRAMA DE SECUENCIA DEL REGISTRO DE SEMEN ANIMALES INDIVIDUALES	73
3.5.10	CASO DE USO DE EVENTOS ANORMALES ANIMALES INDIVIDUALES	75
3.5.10.1	DIAGRAMA DE SECUENCIA DEL REGISTRO DE EVENTOS ANORMALES ANIMALES INDIVIDUALES	75
3.6	CASO DE USO REGISTRO GENERAL ANIMALES GRUPALES	77
3.6.1	DIAGRAMA DE SECUENCIA GENERAL ANIMALES GRUPALES	77
3.6.2	CASO DE USO DEL REGISTRO DE INFORMACIÓN ANIMALES GRUPALES	79
3.6.2.1	DIAGRAMA DE SECUENCIA DEL REGISTRO DE INFORMACIÓN ANIMALES GRUPALES	79
3.6.3	CASO DE USO DEL REGISTRO DE CONSUMO DE ALIMENTO ANIMALES GRUPALES	81



3.6.3.1	DIAGRAMA DE SECUENCIA REGISTRO DE CONSUMO DE ALIMENTO ANIMALES GRUPALES	81
3.6.4	CASO DE USO REGISTRO DE MUERTE DE ANIMALES GRUPALES	83
3.6.4.1	DIAGRAMA DE SECUENCIA DEL REGISTRO DE MUERTE DE ANIMALES GRUPALES	83
3.6.5	CASO DE USO REGISTRO DE VACUNAS ANIMALES GRUPALES	85
3.6.5.1	DIAGRAMA DE SECUENCIA DEL REGISTRO DE VACUNAS ANIMALES GRUPALES	85

CAPÍTULO IV

4	ESPECIFICACIONES FUNCIONALES	87
4.1	MODELO DE LA SOLUCIÓN DE LA PROPUESTA (ARQUITECTURA DEL PROYECTO CEU)	87
4.1.1	CAPA DEL CLIENTE	88
4.1.2	CAPA WEB	88
4.1.2.1	CONTENIDO	88
4.1.2.2	CSS E IMÁGENES	89
4.1.2.3	PÁGINAS	89
4.1.2.4	REPORTES	89
4.1.2.5	TEMPLATE	89
4.1.3	CAPA LÓGICA DEL NEGOCIO	89
4.1.3.1	SESSION BEANS	90



UNIVERSIDAD CENTRAL DEL ECUADOR
Sistema de Registro de Animales y sus Derivados del Centro Experimental Uyumbicho

4.1.3.2	MANAGER BEANS	91
4.1.4	CAPA DE PERSISTENCIA	91
4.1.5	CAPA DE BASE DE DATOS	92
4.2	MODELO CONCEPTUAL DE LA BASE DE DATOS DEL PROYECTO CEU	93
4.3	MODELO FÍSICO DE LA BASE DE DATOS DEL PROYECTO CEU	94
4.4	MODELO DE OBJETOS, CLASES DE LA BASE DE DATOS DEL PROYECTO CEU	95

CAPÍTULO V

5.1	CONCLUSIONES	96
5.2	RECOMENDACIONES	97
	BIBLIOGRAFÍA	98

ANEXOS

ANEXO 1	INSTALACIÓN PROYECTO CEU	100
ANEXO 2	MANUAL DE USUARIO	119
ANEXO 3	MANUAL TÉCNICO	155
ANEXO 4	DICCIONARIO DE DATOS	211
ANEXO 5	GLOSARIO DE TÉRMINOS	230
ANEXO 6	FOTOS	232



LISTA DE ILUSTRACIONES

ILUSTRACIÓN 1 (ÁREA BOVINA)	3
ILUSTRACIÓN 2 (ÁREA DE CUYES)	4
ILUSTRACIÓN 3 (ÁREA AVÍCOLA)	5
ILUSTRACIÓN 4 (ÁREA PORCINA)	6
ILUSTRACIÓN 5 (ÁREA PORCINA) IDENTIFICACIÓN PARA MONTAS	7
ILUSTRACIÓN 6 (ÁREA APICULTURA)	8
ILUSTRACIÓN 7 MODELO ESPIRAL	30



LISTA DE GRÁFICOS

GRÁFICO 1 ARQUITECTURA JEE6	12
GRÁFICO 2 (TECNOLOGÍA WEB JSF)	15
GRÁFICO 3 ARQUITECTURA DEL PROYECTO CEU	87
GRÁFICO 4 (CAPA LÓGICA DEL NEGOCIO)	89



LISTA DE TABLAS

TABLA 1	EL ADMINISTRADOR	35
TABLA 2	MANTENIMIENTO DE LA INFORMACIÓN DEL CENTRO EXPERIMENTAL UYUMBICHO	37
TABLA 3	GESTIÓN USUARIOS, PERFILES, PERMISOS	39
TABLA4	REGISTRO DE PRODUCCIÓN DE HUEVOS	41
TABLA5	REGISTRO DE PRODUCCIÓN LECHERA	43
TABLA6	REGISTRO DE LA PRODUCCIÓN DE MIEL Y POLEN	45
TABLA 7	REGISTRO DE PRODUCTOS DERIVADOS	47
TABLA 8	REGISTRO DE VENTA DE ANIMALES	49
TABLA 9	REGISTRO DEL CONSUMO DE LA PRODUCCIÓN	51
TABLA10	REGISTRO DE VENTA DE PRODUCTOS	53
TABLA 11	REALIZAR REPORTES	55
TABLA 12	REGISTRO GENERAL DE ANIMALES INDIVIDUALES	58
TABLA 13	REGISTRO DE INFORMACIÓN DE ANIMALES INDIVIDUALES	60



UNIVERSIDAD CENTRAL DEL ECUADOR
Sistema de Registro de Animales y sus Derivados del Centro Experimental Uyumbicho

TABLA 14	REGISTRO INSEMINACIÓN DE ANIMALES INDIVIDUALES	62
TABLA 15	REGISTRO SANITARIO ANIMALES INDIVIDUALES	64
TABLA 16	REGISTRO DE VACUNAS ANIMALES INDIVIDUALES	66
TABLA 17	REGISTRO DE MUERTE ANIMALES INDIVIDUALES	68
TABLA 18	REGISTRO DE PARTOS Y CRÍAS ANIMALES INDIVIDUALES	70
TABLA 19	REGISTRO DE PESOS ANIMALES INDIVIDUALES	72
TABLA 20	REGISTRO DE SEMEN PORCINOS (ANIMALES INDIVIDUALES)	74
TABLA 21	REGISTRO DE EVENTOS ANORMALES ANIMALES INDIVIDUALES	76
TABLA 22	REGISTRO GENERAL DE ANIMALES GRUPALES	78
TABLA 23	REGISTRO DE INFORMACIÓN ANIMALES GRUPALES	80
TABLA 24	REGISTRO DE CONSUMO DE ALIMENTOS ANIMALES GRUPALES	82
TABLA 25	REGISTRO DE MUERTE ANIMALES GRUPALES	84
TABLA 26	REGISTRO DE VACUNAS ANIMALES GRUPALES	86



RESUMEN

El **CEU** (Centro Experimental Uyumbicho) está ubicado en la parroquia de Uyumbicho cantón Mejía es parte de la Facultad de Veterinaria y Zootecnia de la Universidad Central del Ecuador, en la actualidad sus registros diarios lo hacen en forma manual y utiliza la herramienta de Excel, por lo que se debe automatizar, a través de un sistema informático que cumpla con los requerimientos necesarios.

La realización del Sistema facilitará la administración y la toma de decisiones oportunas en el Centro, y la Facultad de Veterinaria tendrá acceso al Sistema mediante la web y de esta manera se podrá obtener en forma rápida, reportes diarios de toda la información que se maneja en el Centro Experimental Uyumbicho

El sistema se entregará en la Facultad de Veterinaria y Zootecnia de la Universidad Central del Ecuador.

DESCRIPTORES:

REGISTRO DE ANIMALES/CENTRO EXPERIMENTAL

UYUMBICHO/FACULTAD DE VETERINARIA/WEB JSF/ UML/ BASE DE DATOS /TIPOS DE DIAGRAMAS DE DATOS



ABSTRACT

The CEU (Centro Experimental Uyumbicho) is ubicated in the Parroquia of Uyumbicho Canton Mejia is part of the Faculty of Veterinary and Zootechnics of the Central University of Ecuador, in the present they make their daily records manually and utilize the Exel tool, which is why it has to be automize through an informatic system that complies with the nessessary requirements.

The realization of the system will facilitate the administration and opportune decision making in the Center, and the Faculty of Veterinary will have access to the system through the web, and will be able to obtain daily reports in a faster way of all the information that is manage at the Centro Experimental Uyumbicho.

The system will be deliver to the Faculty of Veterinary and Zootechnics of the Central University of Ecuador.

DESCRIPTOR:

**REGISTRATION OF ANIMALS/ CENTRO EXPERIMENTAL
UYUMBICHO/COLLEGE OF VETERINARY SCIENCIE/WEB JSF/UML/
DATABASE /PLOT TYPES OF DATA**



CAPÍTULO I

INTRODUCCIÓN.

Con el avance de la ciencia y tecnología, el mundo globalizado en que vivimos a través de las redes sociales podemos recibir información que nos brindan los medios de comunicación, que nos permiten crear sistemas óptimos para brindar una información verás con calidad y eficiencia. En el mundo actual con la implantación de las nuevas tecnologías nos facilitan optimizar el trabajo, ahorrar tiempo y recursos, aumentar la productividad, y mejorar el rendimiento de las diferentes áreas y entidades, para lo cual se requiere utilizar herramientas que nos permitan mejorar los procesos como es el caso del software libre, que sin tener que pagar una licencia y se puede modificar el producto podemos adaptarlo a nuestras necesidades.

El Centro Experimental Uyumbicho de la Facultad de Veterinaria de la Universidad Central del Ecuador, no cuenta con un sistema informático que le permita llevar un registro de control de los animales y sus derivados, de las cinco áreas (Avicultura, Bovinos, Apicultura, Cuyes, Porcina) y de la producción que tiene el Centro, el mismo que lo ha venido realizando hasta la actualidad en forma tradicional o manual.

El desarrollo de esta tesis permitirá optimizar, simplificar el trabajo y el manejo del Centro, aumentando así su rendimiento. Para que esto se lleve a efecto se realizará la investigación necesaria, realizando entrevistas y discusiones con usuarios potenciales, analizando procesos, realizando diagramas, recolectando datos, estableciendo interfaces de usuarios y prototipos técnicos y priorizando requerimientos; una vez que se disponga de la información necesaria procederemos a la construcción y posterior entrega del sistema.

El sistema integrará varios procesos que se enlazan en la Web, permitiendo así que todos puedan acceder a la información del servicio del Centro y sus ofertas, y un mayor control de los recursos a nivel interno.

Las personas encargadas del Centro deberán actualizar diariamente y continuamente la producción y el movimiento de las cinco áreas.



El servicio que facilita el sistema es informar a las autoridades de la Universidad Central del Ecuador especialmente a la Facultad de Veterinaria de la Universidad Central del Ecuador, los reportes diarios que se realizan en el Centro Experimental Uyumbicho sin necesidad que el encargado se traslade a la misma.

1.1. PLANTEAMIENTO DEL PROBLEMA.

El Centro Experimental Uyumbicho tiene cinco áreas de que son: Avicultura, Bovinos, Cuyes, Porcina, Apicultura y sus derivados, cada una de las cuales llevan su registro manual, diario y continuo, el mismo que es guardado en los archivos del Centro, cada cierto tiempo tienen la colaboración de pasantes de la Facultad de Veterinaria de la Universidad Central del Ecuador que de igual manera llevan su registro manual.

Este conteo es reportado a diario al Administrador en formatos preparados por ellos, por lo que se ve la necesidad de aprovechar los recursos tecnológicos para la implementación de un Sistema que permita recolectar la información en forma óptima y actual, la misma que esté al alcance de autoridades y clientes brindando de esta manera un servicio de calidad y competitividad.

Con nuestra sistematización el Centro Experimental Uyumbicho de la Facultad de Veterinaria de la Universidad Central del Ecuador, va ha mantener un estándar de toda la información, obteniendo como resultado una mejor organización y rendimiento en el entorno funcional.

Una de las ventajas de la sistematización es tener un sistema de control y seguimiento de toda la parte administrativa, operativa, reproductiva y comercializada del Centro.

Con la aplicación de este sistema mejoraremos la calidad de información, servicios, requerimientos y problemas que se presenten en el Centro tomando en cuenta la ayuda gerencial que el sistema proporciona al momento de generar reportes, planificar, organizar y tomar decisiones acertadas que beneficien al Centro Experimental Uyumbicho de la Facultad de Veterinaria de la Universidad Central del Ecuador, demostrando calidad y eficiencia en los servicios que prestan.



Para la obtención de esta información se realizaran entrevistas, investigación de campo con usuarios potenciales, analizando procesos y recolectando datos para la construcción del sistema.

El Centro Experimental Uyumbicho consta de las siguientes áreas:

- Bovina.
- Porcina.
- Cuyes.
- Avícola.
- Apicultura.

Además cuenta con el personal humano como:

- Administrador del Centro.
- Jefes de Áreas.
- 16 trabajadores.
- Pasantes de la Facultad de Veterinaria de la Universidad Central del Ecuador.

A continuación detallaremos los datos obtenidos en las diferentes áreas que se investigó en el Centro.

1.1.1. ÁREA BOVINA



Ilustración 1

En esta área los empleados encargados llevan un registro de los animales dividiéndolos por categorías, los animales al nacer son identificados por un arete



que se los coloca en la oreja a los 8 o 15 días de nacido, se registra el arete, la raza, fecha de nacimiento, raza del padre, código de la madre, raza de la madre.

Durante dos meses son alimentados con heno y balanceado permaneciendo en el establo, luego son alimentados con hierba y sales minerales.

Luego son clasificados en hembras y machos, a estos últimos los crían para luego ser vendidos y las hembras pasan a ser vaconas medio, son las que tienen 7 a 12 meses de vida en donde presentan su primer celo.

Vaconas Vientre: Son las que tienen de 13 a 18 meses de vida y son las que están gestando por primera vez.

Vacas Rejo: Son las consideradas, mayores de 19 meses en adelante, se las somete a periodos de inseminación y si en la tercera inseminación es negativa se descarta la vaca para la venta.

Vacas Lechando: A partir de que paren tienen un período de 10 meses.

Vacas Secas: Son las que no están dando leche, en un período de 2 meses.

Toda ésta información se lleva en los libros en donde se anota toda la información del animal como la fecha del parto, número de parto, número de arete de la cría, sexo, estado del animal y la producción de leche diaria.

1.1.2. ÁREA DE CUYES



Ilustración 2



En esta área tienen dos galpones que son:

Galpón de Madres: Está formado por 50 cajones las mismas que se encuentran en período preñez y dura de 44 a 68 días. Las crías son destetadas entre 4 y 21 días de nacido y luego separadas de acuerdo al sexo que presentan.

Galpón de venta: Se encuentran los cuyes que no cuentan con las características de la raza escogida para reproducción y las madres que ya han cumplido su ciclo de reproducción.

Alimentación.

La alimentación de los cuyes consiste en materia verde (alfalfa, picuyo, tréboles, etc), balanceado preparado en el CEU, a base de afrechillo, maíz, soya, melaza, vitaminas. Las porciones dependen de la edad.

El registro del movimiento diario de los cuyes se lleva en registros manuales, los mismos que son enviados periódicamente a la persona encargada de archivar la información, sobre el total madres, padres, hembras y machos de reproducción, de engorde.

1.1.3. ÁREA AVÍCOLA.



Ilustración 3

Los polluelos son comprados a la primera semana de vida, esta área se dividen en las siguientes categorías.

- a. Pollos: Son las aves que comprenden hasta las 17 semanas.
- b. Gallinas Ponedoras: Son aquellas aves que empiezan su ciclo de postura a partir de la semana 18.
- c. Gallinas de Descarte: Son las gallinas que terminarán su ciclo de postura y se destinan para la venta.



El área avícola cuenta con dos galpones, el proceso de recolección de huevos se realiza diariamente, se alimentan con balacedado elaborado en el Centro; calculando la cantidad de alimento según la categoría que se encuentren las aves y la cantidad de alimento que consume cada una, por el tiempo que va ha consumir.

1.1.4.ÁREA PORCINA.



Ilustración 4

En el área porcina se registran los datos de todo el ciclo de vida de los cerdos desde su concepción, nacimiento proceso de crecimiento y engorde hasta que es vendido o faenado. Para identificarlos se utiliza el método del muesqueo proporciona una clave numérica que puede tener tres diferentes aplicaciones:

Identificación del individuo y camada.

Este sistema tiene la limitante de sólo poder ser utilizado en explotaciones pequeñas, debido a que de forma natural sólo se puede llegar a identificar a 161 camadas, aunque haciendo marcas como agujeros al centro de la oreja se puede aumentar a dos o tres veces la capacidad de marcaje.

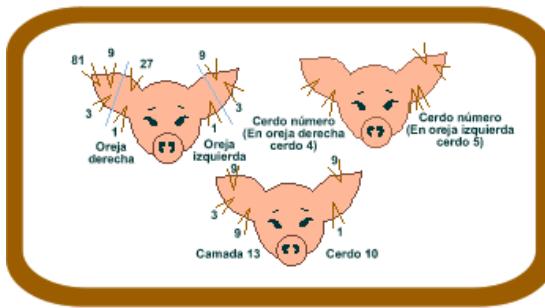


Ilustración 5

Con respecto a la alimentación se lo realiza a base de balanceado, preparado en el mismo Centro.

A continuación detallamos los cerdos que existen en ésta Área.

- Predestete 12.
- Gestación 4.
- Cerdos Reproductores Machos 9.
- Cerdos de reemplazo 3.
- Cerdos en lactación 3.
- Cerdos de recría 41.
- Cerdos en crecimiento 7.
- Cerdos en engorde 10.

En total existen= 86.

1.1.5. ÁREA DE APICULTURA.

En el área de Apicultura, la recolección de miel se la realiza en temporadas, tienen colmenas y el encargado del área, es quién entrega al administrador del Centro Experimental Uyumbicho la cantidad de miel y polen recolectado.



Ilustración 6

Como podemos observar al recolectar los datos se requiere de un sistema de control que permita mayor agilidad en la información de las áreas que contiene el Centro y de esta manera innovar los procesos utilizando los sistemas informáticos actualizados.

1.2. INTERROGANTES DE LA INVESTIGACIÓN.

Dentro de las interrogantes que nos hemos planteado al realizar nuestro trabajo de investigación se tiene lo siguiente:

- ¿La creación del sistema del CEU constituye una herramienta tecnológica que facilite el cuidado, manejo y producción de animales y sus derivados del Centro Experimental Uyumbicho?
- ¿La aplicación del Sistema CEU proporciona a los administradores y clientes datos actualizados con eficiencia y veracidad?
- ¿El sistema CEU puede ser adaptado a los avances tecnológicos para mejorar su estructura y lograr de esta manera una información más real y oportuna?

1.3. OBJETIVOS.

1.3.1. OBJETIVO GENERAL.

Desarrollar un sistema que permita llevar un registro de los animales y sus derivados vía Web del Centro Experimental Uyumbicho de la Facultad de Veterinaria de la Universidad Central del Ecuador.



1.3.2. OBJETIVOS ESPECÍFICOS.

- Examinar el problema al inicio de proyecto, establecer el problema, delimitarlo y definir las necesidades del usuario.
- Establecer estimaciones razonables del costo, una valoración efectiva del riesgo, una planificación del proyecto asequible que proporcione una indicación fiable del progreso y plantear el alcance del sistema dando un enfoque realista del mismo.
- Diseñar un sistema orientado a la Web para el registro de animales y sus derivados del Centro Experimental Uyumbicho.
- Comprobar que el software realice correctamente las tareas indicadas en la especificación del problema, documentar detalladamente el desarrollo del software y la gestión del proyecto.
- Presentar reportes, consultas e información en forma inmediata y eficaz, para toma oportuna y correcta de decisiones.

1.4. JUSTIFICACIÓN.

Actualmente en el Centro Experimental Uyumbicho de la Facultad de Veterinaria de la Universidad Central del Ecuador, lleva un registro de los animales y sus derivados en hojas Excel, datos proporcionados por cada uno de los empleados en las diferentes áreas. El Administrador del Centro tiene que llevar un reporte semanal sobre novedades que presentan los animales, ingresos y egresos del Centro a la Facultad de Veterinaria de la Universidad, al observar esta realidad, se ve la necesidad de crear un Sistema que registre la información de los animales y sus derivados, el mismo que permita agilitar la información de las áreas en forma oportuna, ahorrando tiempo y recursos.

Es por esta razón que se propone al Centro, la solución a este problema con la creación de un sistema, para conocer en forma exacta, oportuna y verás la situación del Centro Experimental Uyumbicho, el total de animales, la producción de los mismos y derivados: diariamente, semanalmente, mensualmente, anualmente, permitiendo al Administrador tener una visión global del Centro.



UNIVERSIDAD CENTRAL DEL ECUADOR

Sistema de Registro de Animales y sus Derivados del Centro Experimental Uyumbicho

De esta manera se optimizará la administración de recursos, arcortando los tiempos; en la búsqueda de la información y en los procesos operacionales. El sistema esta orientado la Web permitiendo acceder a esta información a través de un navegador con la url del sistema.



CAPÍTULO II

2. REVISIÓN BIBLIOGRÁFICA.

2.1. ANTECEDENTES.

El Centro Experimental Uyumbicho es una Unidad Operativa de la Facultad de Medicina Veterinaria y Zootecnia de la Universidad Central del Ecuador, el mismo que constituye un Centro Académico de Producción y Entrenamiento de las diferentes Unidades Didácticas y de Producción Animal de los estudiantes, ubicado en el Ex Colegio Nacional Carlos Zambrano de la parroquia Uyumbicho Cantón Mejía.

La principal prioridad del Sistema del Centro Experimental Uyumbicho es brindar una oportuna y rápida solución a las necesidades de información que tiene el Centro.

EL Sistema CEU se encargará en ahorrar tiempo y recursos a las personas que trabajan en el Centro con una oportuna toma de decisiones en los diferentes áreas.

Para la agilidad de esta información es importante realizar innovaciones y estar conscientes de las necesidades que presenta la sociedad y el mundo actual y del avance de la Ciencia y Tecnología, el mismo que nos permite ser competentes para mejorar nuestra calidad de vida, realización personal como profesionales y contribuir al desarrollo de nuestro país.

2.2. FUNDAMENTACIÓN TEÓRICA.

2.2.1. DEFINICION DE JAVA.

A continuación describiremos algunas definiciones del lenguaje Java el mismo que será aplicado en nuestro sistema.

Java es un lenguaje de programación orientado a objetos, desarrollado por Sun Microsystems a principios de los años 90.

El concepto orientado a objetos hace referencia a la herencia, encapsulamiento, polimorfismo entre ellos.



Java es multiplataforma es decir que se escribe una vez y se corre muchas veces a través de la maquina virtual de java (JVM) que interpreta el código bytecode generado por el compilador de java.

Las características más importantes son:

- Lenguaje orientado a objetos.
- Java es un lenguaje sencillo.
- Independiente de plataforma.
- Brinda un gran nivel de seguridad.
- Capacidad multihilo.
- Gran rendimiento.
- Creación de aplicaciones distribuidas.
- Su robustez o lo integrado que tiene el protocolo TCP/IP lo que lo hace un lenguaje ideal para Internet.

2.3. ARQUITECTURA JAVA ENTERPRISE EDITION 6 (JEE6).

La especificación de JEE6 define su arquitectura basándose en los conceptos de capas, containers, componentes, servicios y las características de cada uno de éstos. JEE6 proporciona librerías API de servicios web, modelos de componente, gestión y comunicación que lo convierten en el estándar para implementar aplicaciones SOA (arquitectura orientada a servicios) empresariales y aplicaciones web de nueva generación que permite tener un perfil web y varios perfiles.

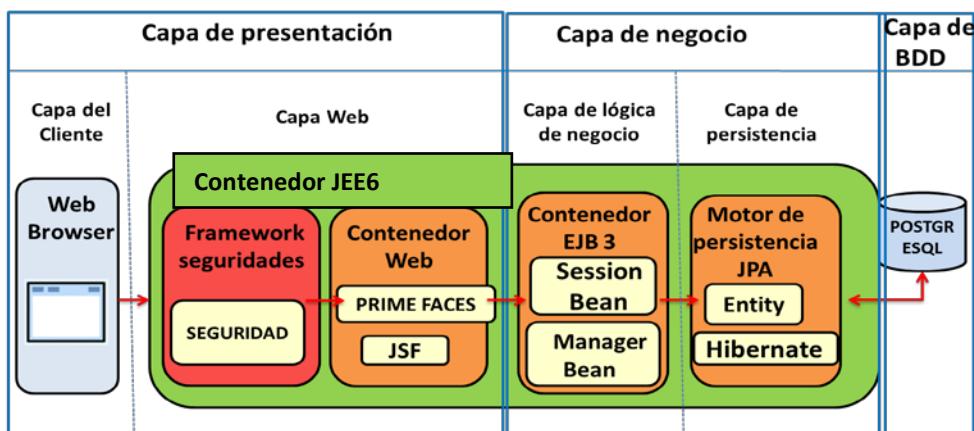


Grafico1 Arquitectura JEE6



La capa del cliente: Esta capa corresponde lo que se encuentra en el computador del cliente. Es la interfaz gráfica del sistema y se encarga de interactuar con el usuario a través del navegador web. JEE6 tiene soporte para diferentes tipos de clientes incluyendo HTML, applets Java y aplicaciones Java.

La capa web: Se encuentra en el servidor web (JBoss) y contiene la lógica de presentación que se utiliza para generar una respuesta al cliente. Recibe los datos del usuario desde la capa cliente y basado en éstos, genera una respuesta apropiada a la solicitud. JEE6 utiliza en esta capa las componentes Java Servlets, JavaServer Faces para crear los datos que se enviarán al cliente.

La capa de negocio: Se encuentra en el servidor de aplicaciones y contiene el núcleo de la lógica del negocio de la aplicación. Provee las interfaces necesarias para utilizar el servicio de componentes del negocio. Los componentes del negocio interactúan con la capa de datos y son típicamente implementadas como componentes EJB.

La capa de Datos: Esta capa es responsable del sistema de información de la empresa o Enterprise Information System (EIS) que incluye bases de datos, sistema de procesamiento de datos, sistemas *legados* y sistemas de planificación de recursos.

Las características de la plataforma JEE 6 son:

- **Portable:** La famosa frase de Sun ‘Escribe una Vez, usa en cualquier parte’ tiene un gran poder de convocatoria, se escribes la aplicación en una máquina Windows o Linux y cuando la hayas terminado puede utilizarse en cualquier plataforma para la que haya disponible una Máquina Virtual Java (JVM).

Especifica una sintaxis para los nombres JNDI (Java Naming and Directory Interface) que debe ser la misma en todos los servidores de aplicaciones.

EJB 3.0 especifica un contenedor embebido que es un API estándar para ejecutar EJBs dentro de un ambiente JSE (Java Estándar Edition), facilitando las pruebas.



- **Escalable:** Se puede añadir nuevos componentes JEE6 a una aplicación Web por ejemplo un aumento de usuarios, sin tener que escribir todo el código de nuevo.
- **Perfiles:** Java EE6 introduce el concepto de perfiles, distintas configuraciones específicas de la plataforma para distintos fines. El primero en ser introducido en esta versión es el Web Profile, un perfil más liviano que usa únicamente lo necesario para desarrollos web.
- **Productividad del desarrollo:** Se usan anotaciones en lugar de archivos de configuración en la capa Web. Servlets, JSF, Managed Beans, convertidores, validadores son ahora clases anotadas con archivos de configuración xml opcionales (el faces-config.xml es opcional).
Para crear EJBs locales, no es necesario definir la interface. Pueden ser desplegados directamente en un war (web archive) sin tener que empaquetarlos previamente en un jar (java archive).
- **Mayor diversidad de componentes:** Se agregan nuevas especificaciones como: DI 1.0, Managed Beans 1.0, Bean Validation 1.0, JAX – RS 1.1.
Se mejoran algunas existentes, JPA 2.0 se agregan colecciones de tipos de datos simples bloqueo pesimista, mejoras en JPQL, llamadas asíncronas Servicio de timer Enriquecido para calendarizar tareas Singleton Session Bean, JSF 2.0 Soporte para Ajax JAX-RS 1.1, creación de RESTFUL Web Services Anotados.

Tipos de aplicaciones empresariales que se puede desarrollar con Java EE 6 Web Profile como:

- Desarrollo de aplicaciones Web.
- Soporte Java EE 5.
- Debugging, testing, profiling.
- Struts y JavaServer Faces (JSF).
- Desarrollo de aplicaciones AJAX.



2.3.1. TECNOLOGÍA WEB JSF.

Esta tecnología se basa en el ciclo de vida de un JSF como interactúa el usuario con la aplicación.

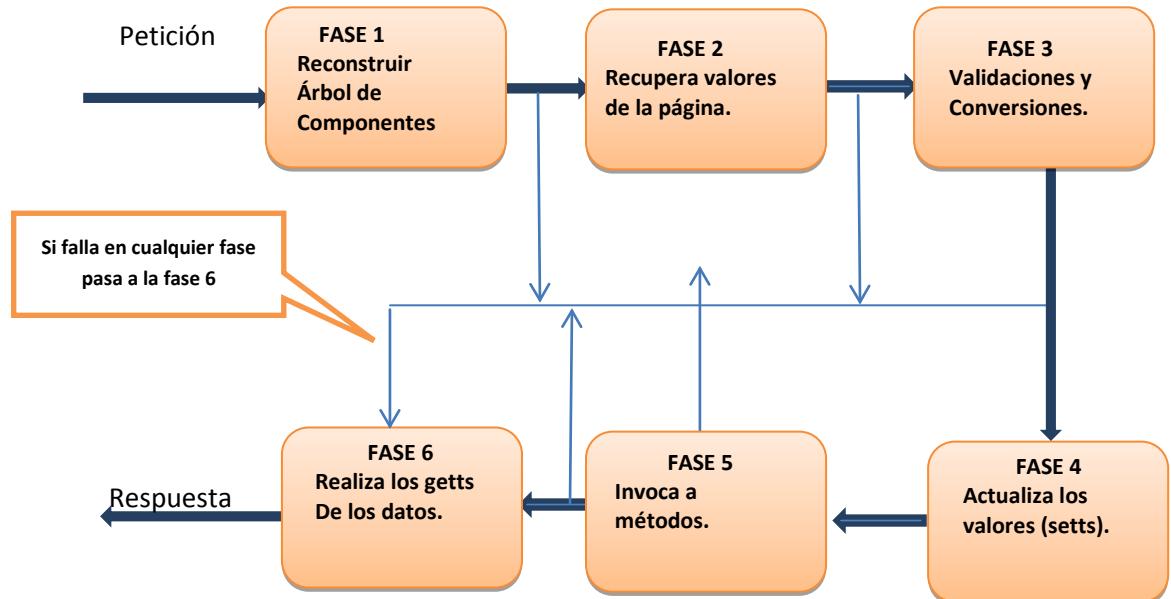


Gráfico 2 TECNOLOGÍA JSF

Siempre que se va a ejecutar por primera vez la página en el servidor se ejecuta la fase 1 y la fase 6.

Fase 1: Crea el árbol de componentes si es la primera vez en llamar la página sin no solo se restaura.

Fase2: Sobre el árbol de componentes pongo los valores que tengo en la página.

Fase3: En esta fase se realizan las validaciones y conversiones de los datos de la página.

Fase4: Se realiza el set (actualiza) los atributos, es decir los datos que se encuentra en la página van al manager vean, solo de los componentes de entrada (input) nunca de los componentes de salida.

Fase5: Se invocan los métodos de tipo action que está en el manager bean.



Fase 6: Se hacen getts (devuelve los datos) de los atributos, toma la información que se encuentra en el manager bean y lo pinta en la página, tanto de los componentes de entrada como de salida.

La tecnología JavaServer Faces es un framework de interfaz de componentes de usuarios del lado del servidor para las aplicaciones web basadas en la tecnología Java.

Los principales componentes de la tecnología JSF son:

- Un API para representar componentes de Interfaz de Usuario (UI) y gestionar su estado.
- Manejador de eventos para validar en el servidor y conversión de datos.
- Reglas de Navegación permite definir la navegación de páginas.
- Soporte de internacionalización y accesibilidad.

Para el desarrollo de aplicaciones de negocio se utiliza frecuentemente el patrón de diseño MVC Modelo Vista Controlador (Model View Controller) que además es sencillo de implementar en las aplicaciones web. En este patrón el modelo es modificable por las funciones de negocio. Estas funciones son solicitadas por el usuario mediante el uso de un conjunto de vistas de la aplicación que solicitan dichas funciones de negocio a través de un controlador, que es el módulo que recibe las peticiones de las vistas y las procesa. Se suele clasificar en dos tipos a las aplicaciones basadas en MVC:

- Tipo 1: Las vistas conocen la acción que se va a invocar en su petición, normalmente la función esta cableada dentro de la vista
- Tipo 2: El controlador introduce un conjunto de reglas que mapean a las peticiones con las funciones, controlando además el flujo de navegación por la aplicación.

La creación de aplicaciones basadas en el patrón MVC se ve facilitada por el uso de marcos de trabajo (frameworks). Un marco de trabajo es un conjunto de APIs y módulos normalmente acompañados de la documentación y guía de uso que definen la manera de implementar alguna de las capas de nuestra aplicación.



VENTAJAS DE JSF 2.0.

- Simplifica la utilización de componentes en las páginas web con la utilización de templates, facelets y la composición de componentes.
- Provee soporte predefinido para la tecnología Ajax ya que es parte de la especificación y el uso de anotaciones.
- La utilización del archivo faces-config.xml es opcional gracias a las anotaciones proporcionadas por el Framework.
- Tenemos un nuevo ámbito de las variables de entornos (manager bean) que es la de View, se puede realizar varias acciones mientras no cambie de página.
- Dado que Facelets es la configuración por default en lugar de JSP, no se necesita agregar jars ni configurar nada adicional en el faces-config.xml ni en el web.xml

2.3.2. PRIMEFACES.

Es la implementación que utiliza el estándar JSF 2.0 por las siguientes características:

- Soporte nativo de Ajax.
- Tiene componentes que mejora la parte visual.
- Las páginas son más livianas.

Algunos inconvenientes podrían ser:

- Al utilizar el soporte de Ajax tenemos que indicarlo explícitamente, por medio de atributos específicos de cada componente.
- No podemos utilizar el soporte de Ajax de JSF 2 (mediante < f:ajax>) con los componentes de PrimeFaces.

2.3.3. SERVIDOR DE APLICACIONES JBOSS.

Un servidor de aplicaciones es una plataforma de middleware para el desarrollo y despliegue de software basado en componentes.



2.3.3.1. JBOSS 7.

Es un servidor de aplicaciones que proporciona servicios que soportan la ejecución y disponibilidad de las aplicaciones, tareas relacionadas con el mantenimiento de la seguridad y del estado, acceso a datos y persistencia entre otros.

Características:

- Es rápido.
- Ligero.
- Núcleo modular.
- Despliegue en caliente y en paralelo.
- Administración elegante.
- Administración de dominios.
- Construido con componentes de primera clase.
- Tiene un único archivo de configuración standalone.xml para todo tipo de bases de datos, en el cual se coloca un dataSource para conectarse a la base de datos requerida.

La diferencia principal el servidor de aplicaciones siete, se puede iniciar en uno de dos modos de operación diferentes:

- 1.- Dominio administrado: Permite administrar múltiples instancias de servidores desde un mismo punto de control.
- 2.- Instancia de servidor independiente: Es un proceso autónomo, más de un servidor independiente se puede ejecutar y administrarse de forma independiente, varias instancias pueden juntarse para formar un clúster de alta disponibilidad.



Existen dos versiones de JBoss 7

- Web Profile (Java EE6 Certificado)
- Everthing (Java EE6 No Certificado): no está disponible aún.

La versión del servidor que utilizamos es Jboss web perfil está orientada a la ejecución de aplicaciones Web.

El beneficio del servidor es la consola web de administración. Esta consola es una aplicación GWT (Google Web Toolkit) lo cual la hacen mucho más rica en su funcionamiento. Dentro de esta podemos hacer varias actividades de administración, como son:

- Configuración de los datasource.
- Configuración de los logs.
- Configuración de los conectores web y virtual host.
- Realizar deployments con la posibilidad de, habilitar, deshabilitar, subir y eliminar aplicaciones.
- Configuración de los puertos.

La consola web no es la única forma de acceder a la interfaz nativa de administración. Tenemos estas cuatro opciones:

- La consola web que como vimos es limitada.
- Con un cliente de línea de comando, muy completo por cierto.
- Una API java que se puede acceder directamente con java remoting.
- Una API REST-like para enviar comando por http.

2.3.4. Java Persistence API.

Es un API (clases, interfaces, anotaciones) para la persistencia de objetos, es decir el esqueleto para la persistencia, que abstrae de las bases de datos y brinda un estándar para persistir los datos en java.



Mapea automáticamente nuestras clases en la base de datos de manera transparente, y utilizando un estándar, lo cual entre otras cosas nos permite poder migrar de cualquier motor de base de datos, y poder compartir código o trabajar en equipo sin ningún problema.

Un fichero muy importante que tenemos que crear a parte de las clases “Entity” es el fichero “persistence.xml”, en este fichero vamos a indicar precisamente que clases son Entity, sobre qué base de datos vamos a atacar, y cuál es la política de creación de esta base de datos. Tiene los datos de la conexión a la base de datos, la cual ya debe estar creada anteriormente.

En el caso de las tablas no es necesario, JPA automáticamente crea las tablas necesarias a partir de las tablas que poseamos.

Java Persistence API consta de tres áreas:

- El API de JPA.
- El lenguaje de query: Es el lenguaje de consultas similar al SQL pero se usan objetos (entidades) los joins entre entidades se hace automáticamente haciendo referencia a los objetos.
- El mapeo de los metadatos objeto/relacional.

Para usar jpa se necesita alguna librería que implemente el api de jpa, existen algunas librerías que lo implementan como: Hibernate, Toplink, OpenJpa, EclipseLink, etc.

2.3.5. Hibernate.

Es marco de trabajo para el desarrollo de aplicaciones empresariales en Java

Características principales:

- Es una implementación del estándar JPA en el archivo de persistence.xml le atamos el estándar a la implementación.
- En el mapeo Objeto/Relacional soporta el mapeo de relaciones complejas de objetos, claves simples y compuestas, mapeo de superclases y relaciones de herencia de clases, el uso de Lazy / Eager para carga



liviana y pesada de objetos relacionados, actualizaciones y eliminación de datos en cascada.

- Se tiene un Hibernate Dialect para cada motor de base de datos.
- Debe especificar el dialecto de la base de datos (Oracle, DB2, PostgreSQL, etc).
- La configuración de acceso a datos también puede hacerse mediante JNDI para nombrar un datasource.

2.3.6. Contenedor Enterprise Java Bean.

Provee funcionalidades comunes en forma de servicios que los componentes EJB pueden usar. La configuración se realiza con anotaciones, de forma que no se utiliza archivos XML. Las anotaciones convierten a un POJO (clase que depende del Framework) en un EJB. Permite la escalabilidad y el rendimiento del sistema, migrando a un conjunto de computadoras con sólo modificar la configuración

Existen tres tipos de EJBs:

- Session Beans: Lleva la lógica de negocio, hay dos grandes tipos de Stateless, Stateful, el primero no conserva el estado de ninguno de sus atributos de la invocación de un método a otro y el segundo conserva el estado a lo largo de toda una sesión. Los Session Beans Stateless se utiliza en el proyecto CEU, en la arquitectura JEE6 no se necesita tener explícitamente una interfaz porque utiliza lo que crea el contenedor por defecto, la interfaz local, estos servicios junto con las clases entidades realizan el ingreso, actualizaciones, eliminaciones entre otros a la base de datos.
- Message-Driven Beans (MDBs): Se usa para invocar métodos de forma asíncrona desde un cliente, la llamada no bloquea el código del cliente y el mismo puede seguir con su ejecución, sin tener que esperar indefinidamente por la respuesta del servidor. Los MDBs encapsulan el popular servicio de mensajería de Java, este tipo de EJB interactúan con la página xhtml.



- Entidades: Son los EJBs que manejan la Java Persistence API (JPA), también parte de la especificación de EJB 3.0. Las entidades son POJOs con cierta metadata que permite a la capa de persistencia mapear los atributos de la clase a las tablas de la base de datos y sus relaciones.

2.3.7. BASE DE DATOS.

2.3.7.1. POSTGRESQL.

Es un sistema de gestión de bases de datos objeto-relacional, ya que incluye características de la orientación a objetos, como puede ser la herencia, tipos de datos, funciones, restricciones, disparadores, reglas e integridad transaccional.

CARACTERÍSTICAS DE POSTGRESQL.

- **DBMS Objeto-Relacional.**

PostgreSQL aproxima los datos a un modelo objeto-relacional, y es capaz de manejar complejas rutinas y reglas. Ejemplos de su avanzada funcionalidad son consultas SQL declarativas, control de concurrencia multi-versión, soporte multi-usuario, transactions, optimización de consultas, herencia, y arrays.

- **Altamente Extensible.**

PostgreSQL soporta operadores, funciones métodos de acceso y tipos de datos definidos por el usuario.

- **Soporte SQL Comprensivo.**

PostgreSQL soporta la especificación SQL99 e incluye características avanzadas tales como las uniones (joins) SQL92.

- **Integridad Referencial.**

PostgreSQL soporta integridad referencial, la cual es utilizada para garantizar la validez de los datos de la base de datos.

- **API Flexible.**

La flexibilidad del API de PostgreSQL ha permitido a los vendedores proporcionar soporte al desarrollo fácilmente para el RDBMS PostgreSQL.



- **Lenguajes Procedurales.**

PostgreSQL tiene soporte para lenguajes procedurales internos, incluyendo un lenguaje nativo denominado PL/pgSQL. Este lenguaje es comparable al lenguaje procedural de Oracle, PL/SQL. Otra ventaja de PostgreSQL es su habilidad para usar Perl, Python, o TCL como lenguaje procedural embebido.

- **Cliente Servidor.**

PostgreSQL usa una arquitectura proceso-por-usuario cliente/servidor. Esta es similar al método del Apache 1.3.x para manejar procesos. Hay un proceso maestro que se ramifica para proporcionar conexiones adicionales para cada cliente que intente conectar a PostgreSQL.

2.3.8. JASPER REPORTS.

JasperReports es una poderosa y flexible solución de código abierto para la generación y gestión de informes. JasperReports es un módulo que dispone de un depósito de archivos que usa un sistema de carpetas, una aplicación web que muestra todos los informes que están en el depósito y un visor de dichos informes. La aplicación web permite subir todos los informes creados, e inmediatamente estos estás disponibles para todos los usuarios. Si por ejemplo, un informe dispone de algún parámetro, se puede crear un nuevo formulario que permita al usuario introducir nuevos parámetros y exportarlo al formato PDF.

La interfaz grafica para realizar reportes es ireport 4.5 que facilita la extracción de datos de la base y nos genera dos archivos .jasper y .jrxml quienes permite generar reportes desde la aplicación JEE6 utilizando una clase generadora.

2.4. MODELO DE UML.

El modelo de UML describe completamente aquellos aspectos del sistema que son relevantes al propósito del modelo y a un apropiado nivel de detalle. Un lenguaje de modelado consisten en:

Vista: Es una abstracción que consiste en un número de diagramas y todos esos diagramas juntos, muestran una representación completa del sistema. Las diferentes vistas de UML son:



- *Vista Lógica:* Muestra como se diseña la funcionalidad dentro del sistema.
- *Vista de Componentes:* Muestra la organización de los componentes de código.
- *Vista Concurrente:* Muestra la concurrencia en el sistema, es decir el direccionando los problemas con la comunicación y sincronización.
- *Vista de Distribución:* Muestra la distribución del sistema en la arquitectura física con computadoras y dispositivos.
- *Vista Use-Case:* Una vista que muestra la funcionalidad del sistema como la perciben los actores externos.

Diagramas: Son una representación gráfica de una colección de elementos del modelo que representan conceptos comunes orientados a objetos.

Símbolos o Elementos de modelo: Son los conceptos utilizados en los diagramas tales como clases, objetos y mensajes, y las relaciones entre estos conceptos incluyendo la asociación, dependencia y generalización. Un elemento es utilizado en varios diagramas diferentes, pero siempre tiene el mismo significado y simbología.

Reglas o Mecanismos generales: Proveen comentarios extras, información o semántica acerca del elemento de modelo; además proveen mecanismos y estos pueden ser: especificaciones, adornos, divisiones comunes, extensiones.

2.4.1. TIPOS DE DIAGRAMAS UML.

UML (Lenguaje Unificado de Modelado) Proporciona diagramas para representar modelos que dan una descripción completa de un sistema, desde las perspectivas estáticas y dinámicas así como de su modularización.

Existen nueve tipos de diagramas que se detallan a continuación.



2.4.1.1. Diagrama de Casos de Uso.

Es una especificación completa de todas las formas posibles de utilizar un sistema desde el punto vista del usuario.

También es considerado como una técnica para obtener requerimientos del sistema.

Un Diagrama de Casos de Uso contiene:

- **Actor:** Representa cualquier cosa que interactúe con el sistema.
- **Caso de Uso:** Es una interacción típica entre un usuario (Actor) y un sistema.
- **Relación:** Es el enlace que permite vincular al actor con el caso de uso.

2.4.1.2. Diagramas de Clases.

El diagrama de clases muestra las clases que hay en el sistema con sus relaciones estructurales y de herencia. Este tipo de diagrama es principal para el análisis y diseño.

El diagrama de clase se comprende de tres secciones:

- **La primera sección:** Contiene el nombre de la clase.
- **La segunda sección:** Muestra la estructura (atributos).
- **La tercera sección:** Muestra el comportamiento (operaciones o acciones).

La segunda y tercer sección pueden suprimirse si no es necesario que sean visibles en el diagrama.

2.4.1.3. Diagrama de Objetos.

Representa a los objetos, siendo un objeto una instancia de la clase que tiene valor específicos de los atributos y acciones.

Al igual que una clase, el símbolo de un objeto es un rectángulo. El nombre de la instancia específica se encuentra a la izquierda antes de los dos puntos y el nombre de la clase a la derecha.



2.4.1.4. Diagrama de Estados.

Este diagrama modela el comportamiento de una parte del sistema. Se elabora un diagrama de estados para cada clase.

El comportamiento es modelado en términos del estado en el cual se encuentra el objeto y las acciones que realiza en cada estado después de terminado evento.

2.4.1.5. Diagrama de Secuencias.

Un diagrama de secuencia muestra la interacción de un conjunto de objetos en una aplicación en una secuencia de tiempo. El diagrama muestra el siguiente tipo de información.

Un diagrama de secuencia contiene:

- **Objetos con sus “líneas de vida”:** Un objeto se representa como una línea vertical punteada con un rectángulo de encabezado (contiene el nombre del objeto y el de su clase, en un formato nombreObjeto: nombreClase).
- **Activación:** Muestra el periodo de tiempo en el cual el objeto se encuentra desarrollando alguna operación, bien sea por sí mismo o por medio de delegación a alguno de sus atributos. Se denota como un rectángulo delgado sobre la línea de vida del objeto.
- **Mensajes:** El envío de mensajes entre objetos se denota mediante una línea sólida dirigida, desde el objeto que emite el mensaje hacia el objeto que lo ejecuta.

2.4.1.6. Diagrama de Colaboración.

Un Diagrama de colaboración modela las interacciones entre objetos en el sistema. Ofrece una mejor visión de todos los efectos de un objeto dado durante un escenario.

Los objetos se conectan por medio de enlaces y a su vez el enlace muestra los mensajes enviados entre los objetos, el tipo de mensaje y la visibilidad de un objeto con respecto a los otros.



Un diagrama de colaboración contiene:

- Objetos.
- Enlace entre objetos.
- Mensajes intercambiados entre objetos.
- Flujo de datos entre objetos, si hay alguno.

2.4.1.7. Diagrama de Actividad.

Puede especificar el comportamiento de los objetos de una clase que tiene una gran cantidad de procesos paralelos.

También permiten describir Casos de Uso y de un Flujo de Trabajo y es considerado como un caso especial de los diagramas de estados, donde todos los estados son estados de acción y todas las transacciones son consecuencia de la finalización de la acción.

Un Diagrama de actividad contiene:

- **Actividad:** Es una acción a realizar.
- **La barra de sincronización:** Permite iniciar acciones una vez que se han realizado actividades concurrentes.
- **Decisión:** Es un punto en el que se pueden seguir alternativas distintas de acuerdo al resultado de la actividad anterior.
- **Condición de guarda:** Es el posible resultado de una acción que servirán como condición para la realización de otra.

2.4.1.8. Diagrama de Componentes.

Permite modelar la estructura del software y la dependencia entre componentes. Un componente es una unidad de código fuente, binario o ejecutable, que sirve como bloque constructor para la estructura física de un sistema.

2.4.1.9. Diagramas de Distribución.

Muestra la Arquitectura física de un sistema informático, también puede representar y mostrar los nodos, sus conexiones y el software que se encuentra en cada máquina.



Un nodo es un objeto físico en tiempo de ejecución que representa recursos de cómputo.

Una conexión indica comunicación, usualmente la relación directa entre hardware.

2.5. METODOLOGÍA.

2.5.1. MODELO EN ESPIRAL.

El modelo en espiral para la ingeniería de software ha sido desarrollado para cubrir las mejores características tanto del ciclo de vida clásico, como de la creación de prototipos, añadiendo al mismo tiempo un nuevo elemento: el análisis de riesgo.

En la siguiente figura se muestran las cuatro actividades principales que realiza el modelo en espiral:

- 1. Planificación:** Es la determinación de objetivos, alternativas y restricciones. Recolección de requisitos, planificación inicial del proyecto.
- 2. Análisis de riesgo:** Se indican los principales riesgos críticos que podría presentar el desarrollo del software y se han evitado con el análisis de alternativas e identificación / resolución de riesgos.
- 3. Ingeniería:** Muestra el desarrollo del producto del "siguiente nivel". Prototipo inicial del software.
- 4. Evaluación del cliente:** Da la valorización de los resultados de la ingeniería.

Durante la primera vuelta alrededor de la espiral se definen los objetivos, las alternativas y las restricciones, y se analizan e identifican los riesgos. Si el análisis de riesgo indica que hay una incertidumbre en los requisitos, se puede usar la creación de prototipos en el cuadrante de ingeniería para dar asistencia tanto al encargado de desarrollo como al cliente.

En el cuadrante de evaluación de cliente, el cliente evalúa el trabajo de ingeniería y sugiere modificaciones. Sobre la base de los comentarios del cliente se produce la siguiente fase de planificación y de análisis de riesgo. En cada bucle alrededor de la espiral, la culminación del análisis de riesgo resulta en una decisión de "seguir o no seguir".



Con cada iteración alrededor de la espiral (comenzando en el centro y siguiendo hacia el exterior), se construyen sucesivas versiones del software, cada vez más completa y, al final al propio sistema operacional.

2.5.2 PARADIGMA DEL MODELO EN ESPIRAL.

El paradigma del modelo en espiral para la ingeniería de software es el enfoque más realista para el desarrollo de software y de sistemas a gran escala. Utiliza un enfoque evolutivo para la ingeniería de software, permitiendo al desarrollador y al cliente entender y reaccionar a los riesgos en cada nivel evolutivo. Utiliza la creación de prototipos como un mecanismo de reducción de riesgo, pero, lo que es más importante permite a quien lo desarrolla aplicar el enfoque de creación de prototipos en cualquier etapa de la evolución de prototipos.

Mantiene un enfoque sistemático correspondiente a los pasos sugeridos por el ciclo de vida clásico pero dentro de un marco de trabajo interactivo que refleja de forma más realista el mundo real.

2.5.3 FASES DEL MODELO EN ESPIRAL.

El desarrollo en espiral está formado por ciclos divididos en cuatro fases que son:

1. Fase de análisis de requisitos: En esta fase se propone los requisitos que debe ir cumpliendo el software que se está desarrollando, dichos requisitos deben ser funcionales, estructurales y temporales.

En cuanto a las referencias y plazos temporales, no se define de manera estricta, por lo tanto se establece un plazo temporal global, pero no por partes o espirales de desarrollo.

2. Fase de diseño e implementación: Se identifican soluciones tecnológicas para cada una de las funciones del sistema, asignando recursos materiales para cada una de estas funciones.

También establece métodos de validación del diseño y se ajustan a las especificaciones del producto.

Esta fase se desarrolla y plantea usando las distintas alternativas arquitectónicas, de comportamiento y de implementación.



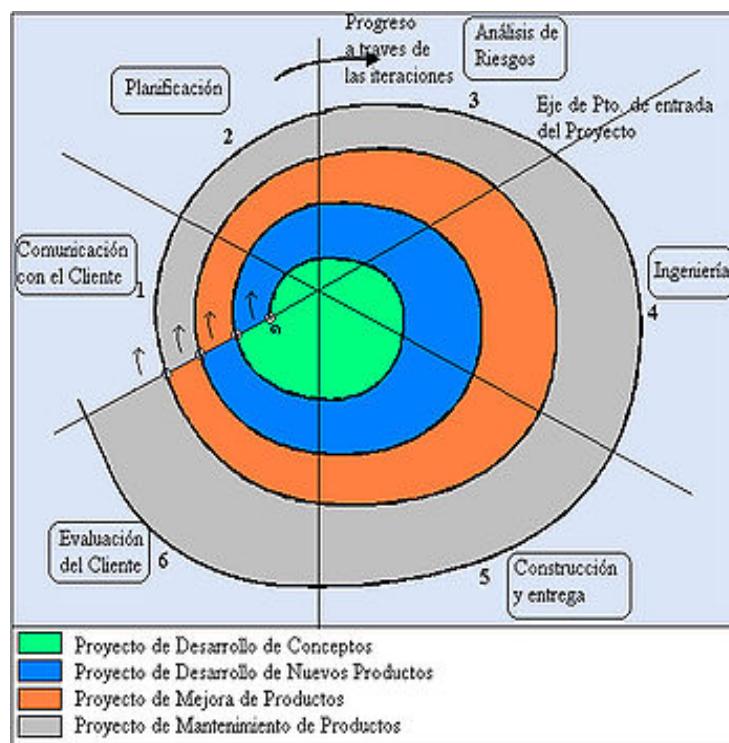
3. Fase de pruebas y planificación: Se realizan varias pruebas del software que resulte de la fase de diseño e implementación. El objetivo de esta fase es comprobar que efectivamente, se ha conseguido cumplir con los objetivos que se plantearon en el análisis de requisitos.

Si los resultados en esta fase han sido satisfactorios, la siguiente fase se planteará como un incremento de la funcionalidad del sistema, caso contrario se planteará como un incremento de la robustez y consistencia del mismo.

4. Fase de ciclo de desarrollo: El ciclo de desarrollo puede considerarse como una generalización del anterior para los casos en que no basta con una sola evaluación de un prototipo para asegurar la desaparición de incertidumbres y/o ignorancias.

El propio producto a lo largo de su desarrollo puede considerarse como una sucesión de prototipos que progresan hasta llegar a alcanzar el estado deseado.

En cada ciclo (espirales) las especificaciones del producto se van resolviendo paulatinamente.





CAPÍTULO III

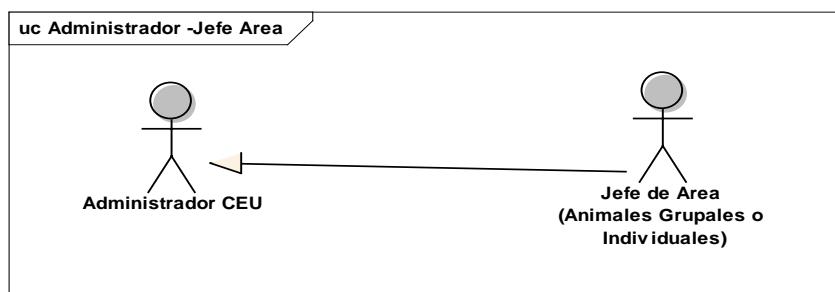
3.1. ACTORES QUE INTERVIENEN EN EL NEGOCIO

Los actores que intervienen en el Negocio son los siguientes:

- **Administrador del CEU:** Es la persona encargada de toda la parte Administrativa y el que toma decisiones en el Centro.
- **Administrador del Área:** Es un doctor veterinario encargado de una área específica que junto al empleado cuidan el estado de la misma.
- **Trabajadores del CEU:** Son personas encargadas de las áreas para la limpieza y cuidado de los animales
- **Pasantes:** Son estudiantes de los últimos años de la Facultad de Veterinaria de la Universidad Central del Ecuador que realizan sus prácticas preprofesionales y ayudan en el mantenimiento del Centro.

3.2. ACTORES QUE INTERVIENEN EN EL SISTEMA.

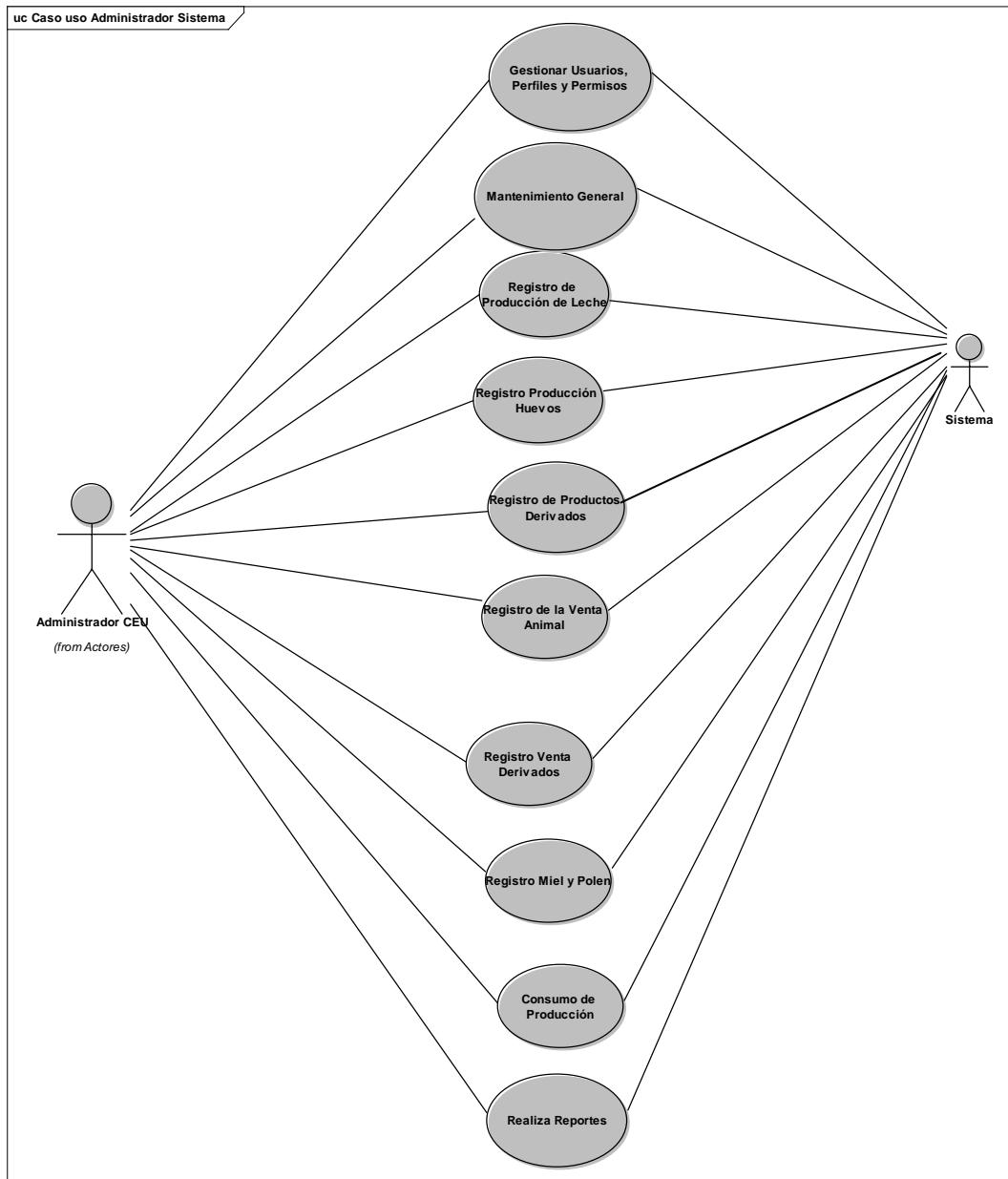
- **Administrador CEU:** Es la persona encargada de administrar las claves del Sistema: crea, modifica, actualiza, asigna roles de las claves de cada usuario, registra: ventas, producción, consumo, mantenimiento, genera reportes.
- **Administrador Área:** Es un doctor veterinario encargado de la parte técnica de cada Área y es quien ingresa información en el sistema de su respectiva Área.





3.3. DIAGRAMA GENERAL DE LOS CASOS DE USO DEL SISTEMA

CENTRO EXPERIMENTAL UYUMBICHO



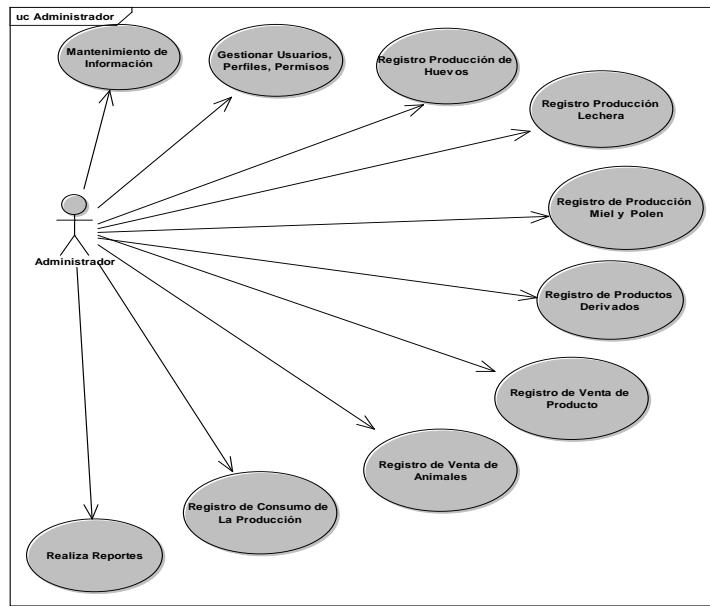


UNIVERSIDAD CENTRAL DEL ECUADOR
Sistema de Registro de Animales y sus Derivados del Centro Experimental Uyumbicho

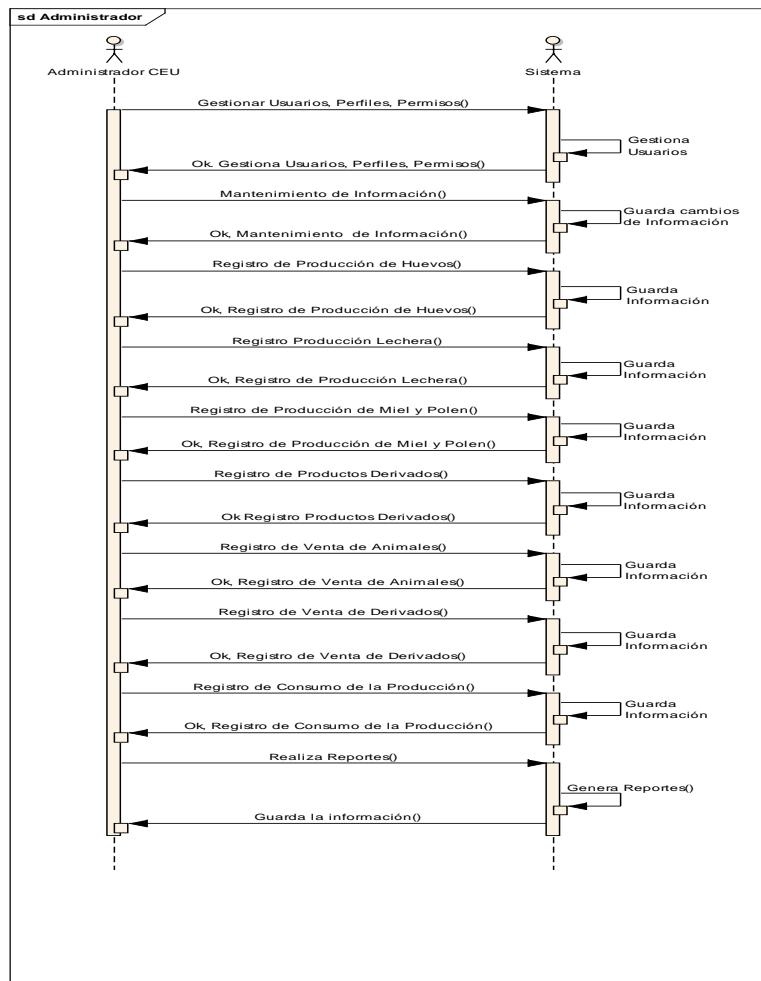




3.4. CASO DE USO DEL ADMINISTRADOR



3.4.1 DIAGRAMA DE SECUENCIA DEL ADMINISTRADOR





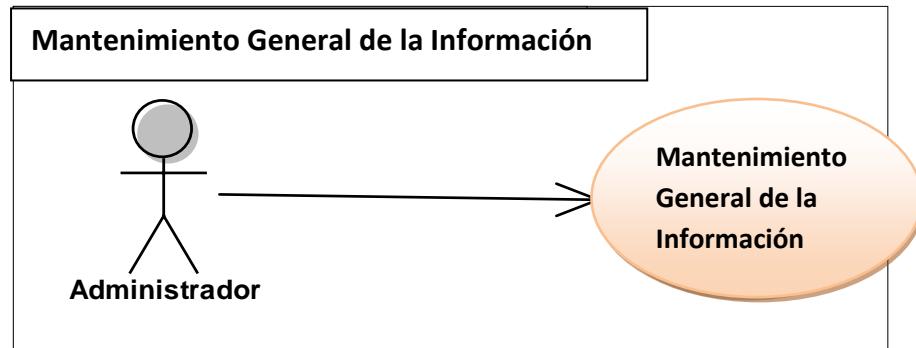
UNIVERSIDAD CENTRAL DEL ECUADOR
Sistema de Registro de Animales y sus Derivados del Centro Experimental Uyumbicho

Nombre:	EL ADMINISTRADOR
Sistema:	
Descripción:	Registrar en el sistema la información de carácter administrativo
Actores:	Administrador del CEU
Precondiciones:	<p>El encargado deberá haber ingresado todos los datos en el sistema:</p> <p>El sistema desplegará un formulario en el cual el usuario deberá ingresar los siguientes datos.</p>
Flujo Normal:	<ol style="list-style-type: none">1. Gestiona Usuarios, Permisos y Perfiles.2. Realiza el Mantenimiento de Usuarios.3. Registro de la Producción de Huevos4. Registro de la Producción de Leche.5. Registro de la Producción de Miel y Polen6. Registro de Productos Derivados7. Registro de Venta de Productos8. Registro de Venta de Animales.9. Registro de Consumo de la Producción10. Realiza Reportes.11. El sistema guarda la información.
Flujo Alternativo:	<ul style="list-style-type: none">▪ El Administrador ingresa datos en formulario del sistema.▪ El Administrador guarda los últimos datos ingresados una vez que haya terminado.
Pos condiciones:	El mensaje ha sido almacenado en el sistema.

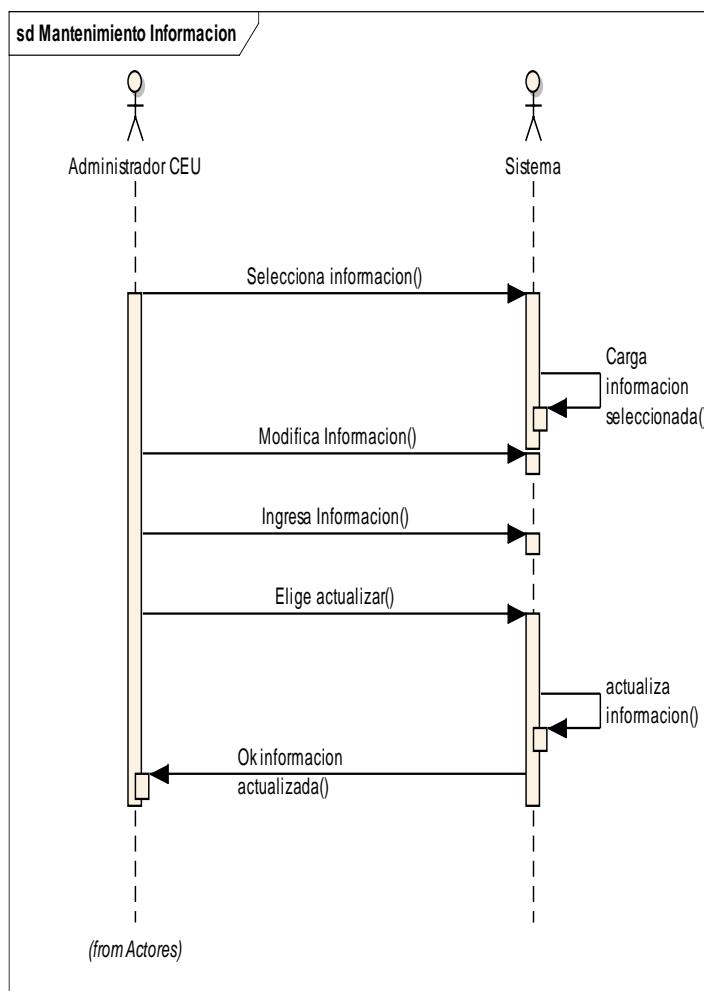
Tabla 1



3.4.2. CASOS DE USO DEL MANTENIMIENTO DE LA INFORMACIÓN DEL CENTRO EXPERIMENTAL UYUMBICHO



3.4.2.1 DIAGRAMA DE SECUENCIA DEL MANTENIMIENTO DE INFORMACIÓN DEL CENTRO EXPERIMENTAL UYUMBICHO





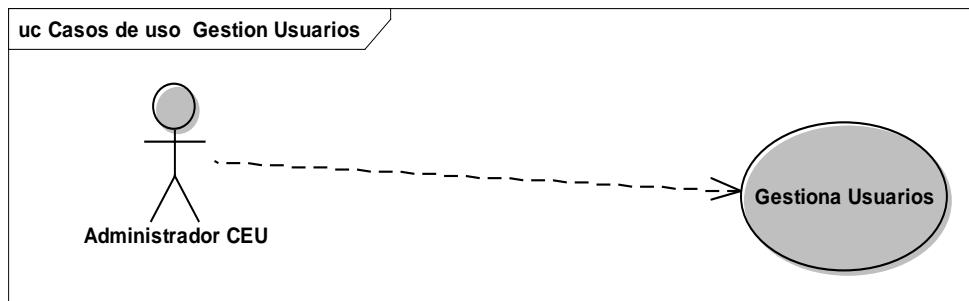
UNIVERSIDAD CENTRAL DEL ECUADOR
Sistema de Registro de Animales y sus Derivados del Centro Experimental Uyumbicho

Nombre:	MANTENIMIENTO DE INFORMACIÓN DEL CENTRO EXPERIMENTAL UYUMBICHO
Sistema:	
Descripción:	Registrar en el sistema la información de los usuarios
Actores	Administrador CEU
Precondiciones:	Para gestionar usuarios se debe tener clave de administrador previamente creada para tener todos los privilegios.
Flujo Normal:	El sistema permitirá: <ul style="list-style-type: none">• Seleccionar la información• Modificar Información.• Ingresar Información.• Elegir actualizar• El sistema guarda la información.
Flujo Alternativo:	Ninguno.
Pos condiciones:	El mensaje ha sido almacenado en el sistema.

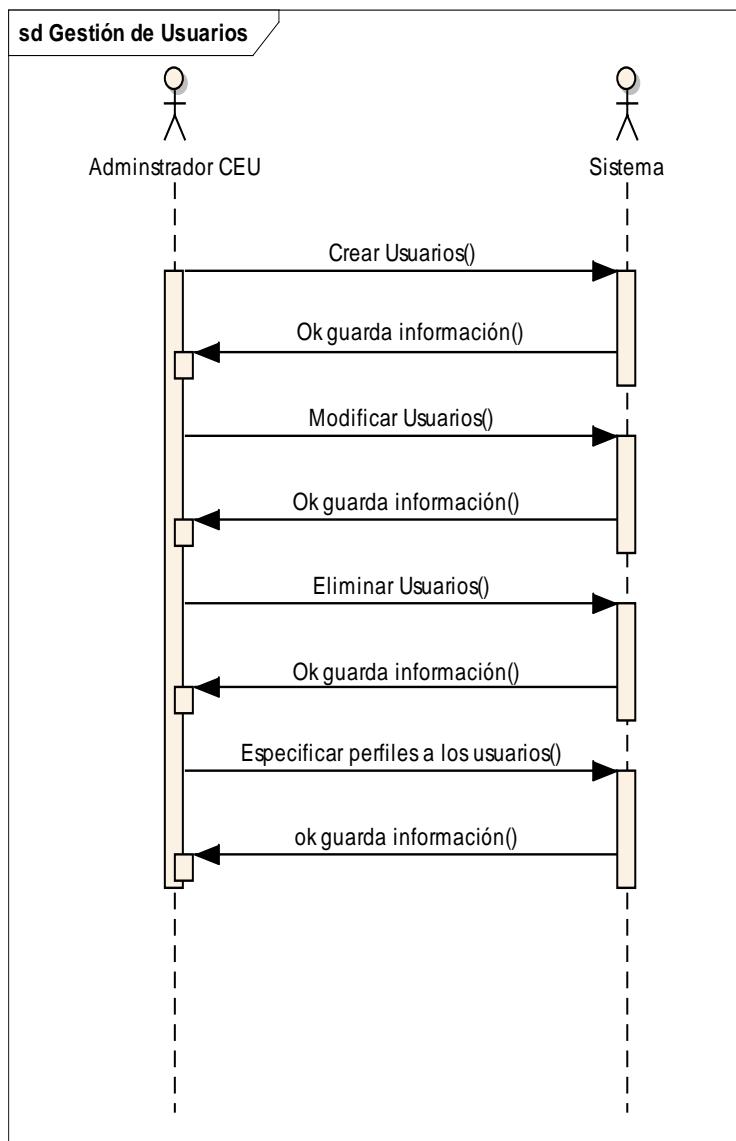
Tabla 2



3.4.3. CASO DE USO GESTIONAR USUARIOS, PERFILES, PERMISOS



3.4.3.1 DIAGRAMA DE SECUENCIA GESTIONAR USUARIOS



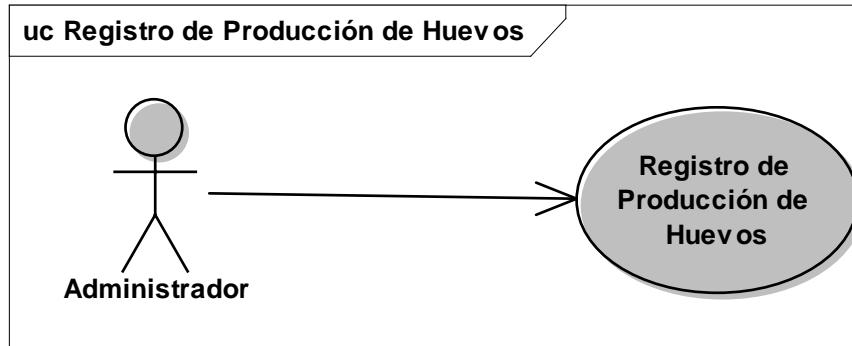


Nombre:	GESTIONAR USUARIOS, PERFILES, PERMISOS
Sistema:	
Descripción:	Registrar en el sistema la información de los usuarios
Actores	Administrador CEU
Precondiciones:	Para gestionar usuarios se debe tener clave de administrador previamente creada para tener todos los privilegios.
Flujo Normal:	<p>El sistema permitirá:</p> <ul style="list-style-type: none">• Crear usuarios: ingresar a nuevas personas que van a utilizar el sistema.• Modificar usuarios y claves: permitir cambiar o modificar nombres de los usuarios y sus claves en caso de olvido de estos.• Eliminar usuarios: cuando un usuario deja de utilizar el sistema se podrá eliminarlo.• Especificar perfiles a los usuarios: dar privilegios de acuerdo al cargo a las personas que utilicen el sistema. Será para los aspirantes únicamente para poder ingresar su hoja de vida y rendir pruebas; para Jefes de Área y de Personal poder construir pruebas, ver reportes, ver diagramas; para el Administrador todos los privilegios.
Flujo Alternativo:	Ninguno.
Pos condiciones:	El mensaje ha sido almacenado en el sistema.

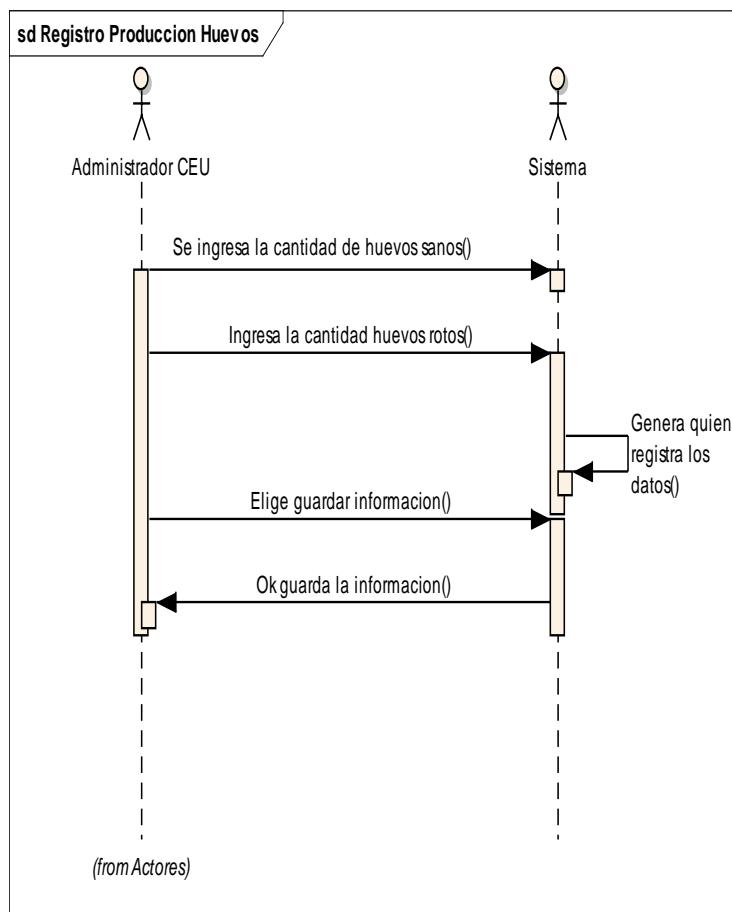
Tabla 3



3.4.4. CASO DE USO DEL REGISTRO DE PRODUCCIÓN DE HUEVOS



3.4.4.1. DIAGRAMA DE SECUENCIA DE LA PRODUCCIÓN DE HUEVOS





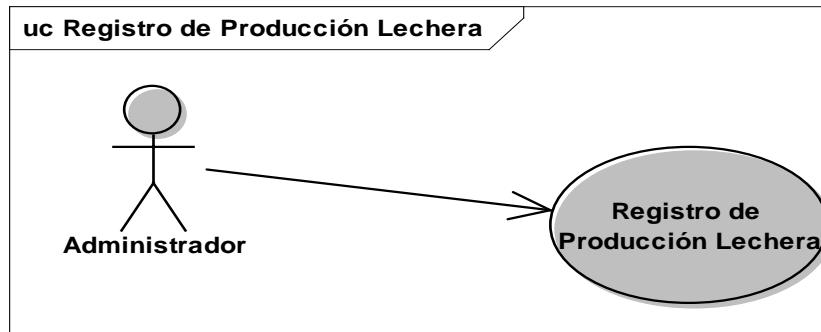
UNIVERSIDAD CENTRAL DEL ECUADOR
Sistema de Registro de Animales y sus Derivados del Centro Experimental Uyumbicho

Nombre:	REGISTRO DE PRODUCCIÓN DE HUEVOS
Sistema:	
Descripción:	Registrar en el sistema la información del registro de la producción de Huevos.
Actores:	Administrador
Precondiciones:	<p>El encargado deberá haber ingresado todos los datos en el sistema: El sistema desplegará un formulario en el cual el usuario deberá ingresar los siguientes datos.</p>
Flujo Normal:	<ol style="list-style-type: none">1. El usuario ingresa la fecha de registro de los huevos.2. El usuario ingresa la cantidad de huevos sanos.3. El usuario ingresa la cantidad de huevos rotos.4. El usuario selecciona guarda la información.5. El sistema guarda la información.
Flujo Alternativo:	<ul style="list-style-type: none">▪ El Administrador ingresa datos en formulario del sistema.▪ El Administrador guarda los últimos datos ingresados una vez que haya terminado.
Pos condiciones:	El mensaje ha sido almacenado en el sistema.

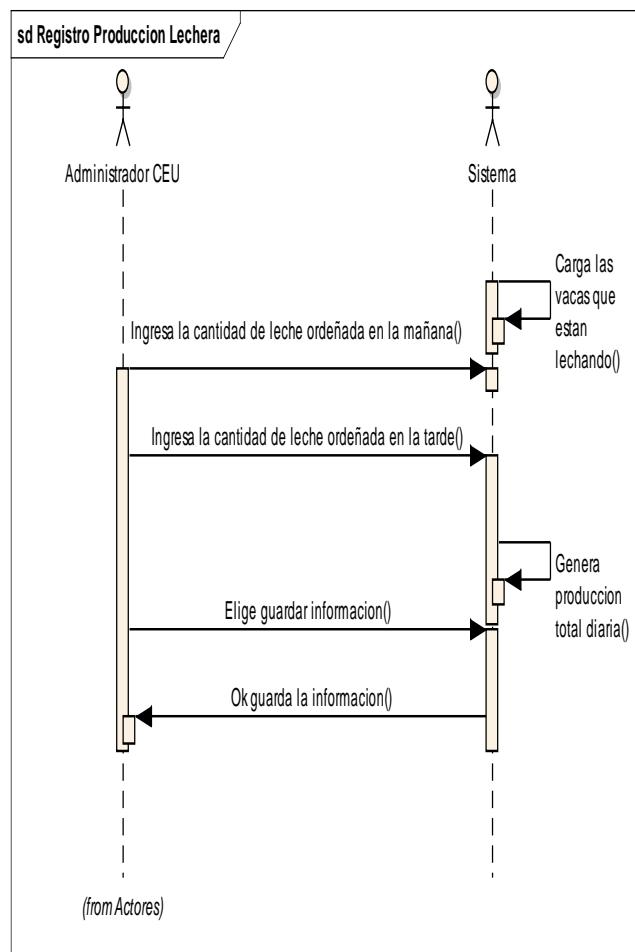
Tabla 4



3.4.5 CASO DE USO DEL REGISTRO DE PRODUCCIÓN LECHERA



3.4.5.1 DIAGRAMA DE SECUENCIA DEL REGISTRO DE LA PRODUCCIÓN LECHERA





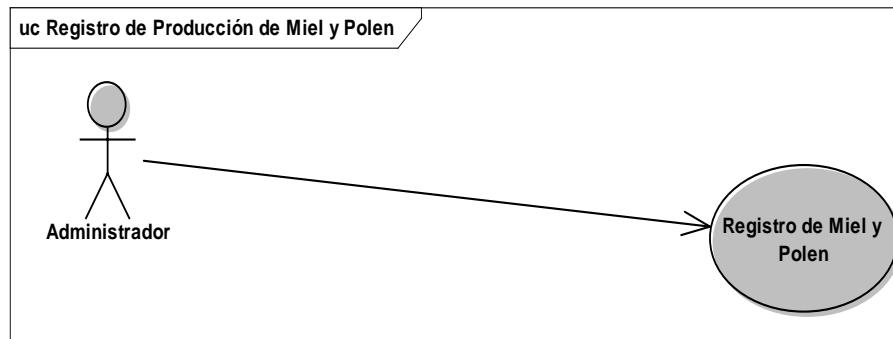
UNIVERSIDAD CENTRAL DEL ECUADOR
Sistema de Registro de Animales y sus Derivados del Centro Experimental Uyumbicho

Nombre:	REGISTRO DE PRODUCCIÓN LECHERA
Sistema:	
Descripción:	Registrar en el sistema la información de la producción lechera
Actores:	Administrador
Precondiciones:	<p>El encargado deberá haber ingresado todos los datos en el sistema:</p> <p>El sistema desplegará un formulario en el cual el usuario deberá ingresar los siguientes datos.</p>
Flujo Normal:	<ol style="list-style-type: none">1. El sistema muestra una lista con todas las vacas lechando.2.- El usuario ingresa la fecha de registro de la leche.3.- De cada vaca listada el usuario ingresa el número de litros de leche producida en la mañana.4.- De cada vaca listada el usuario ingresa el número de litros de leche producidos en la tarde.5.- El usuario selecciona guarda la información.6.- El sistema guarda la información.
Flujo Alternativo:	<ul style="list-style-type: none">▪ El Administrador ingresa datos en formulario del sistema.▪ El Administrador guarda los últimos datos ingresados una vez que haya terminado.
Pos condiciones:	El mensaje ha sido almacenado en el sistema.

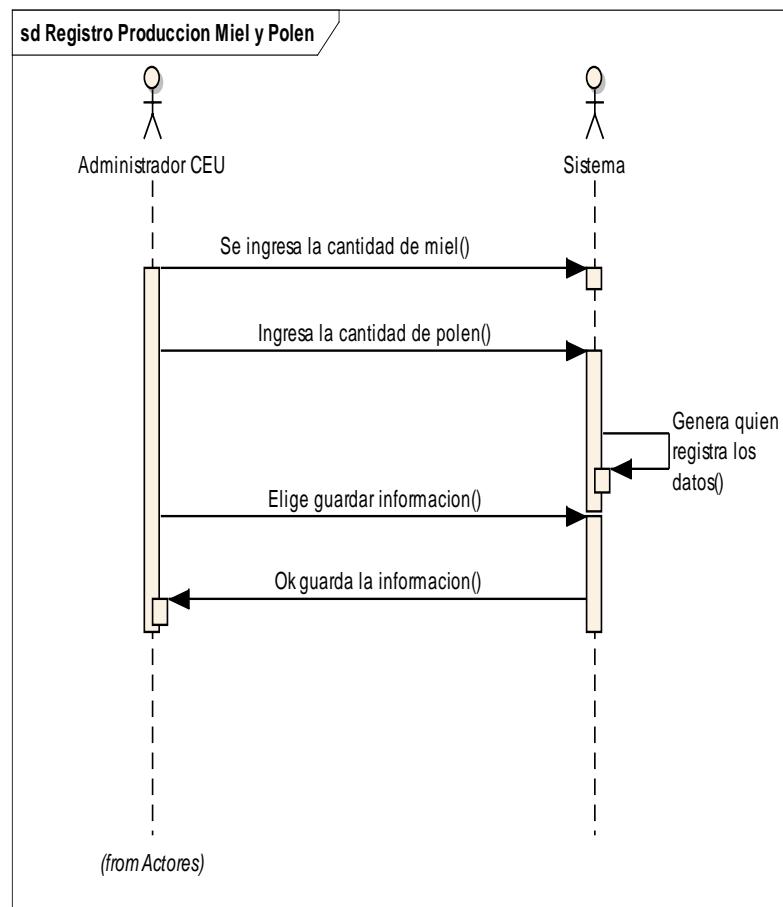
Tabla 5



3.4.6. CASO DE USO DEL REGISTRO PRODUCCIÓN DE MIEL Y POLEN.



3.4.6.1. DIAGRAMA DE SECUENCIA DE LA PRODUCCIÓN MIEL Y POLEN





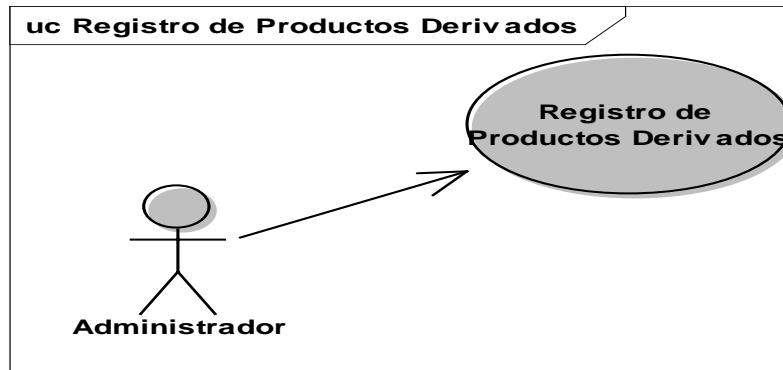
UNIVERSIDAD CENTRAL DEL ECUADOR
Sistema de Registro de Animales y sus Derivados del Centro Experimental Uyumbicho

Nombre:	REGISTRO DE LA PRODUCCIÓN DE MIEL Y POLEN
Sistema:	
Descripción:	Registrar en el sistema la información del registro de la producción de miel y polen.
Actores:	Administrador
Precondiciones:	<p>El encargado deberá haber ingresado todos los datos en el sistema: El sistema desplegará un formulario en el cual el usuario deberá ingresar los siguientes datos.</p>
Flujo Normal:	<ol style="list-style-type: none">1. El usuario ingresa la fecha.2. El usuario ingresa la cantidad de miel.3. El usuario ingresa la cantidad de Polen.4. El usuario selecciona guarda la información.5. El sistema guarda la información.
Flujo Alternativo:	<ul style="list-style-type: none">▪ El Administrador ingresa datos en formulario del sistema.▪ El Administrador guarda los últimos datos ingresados una vez que haya terminado.
Pos condiciones:	El mensaje ha sido almacenado en el sistema.

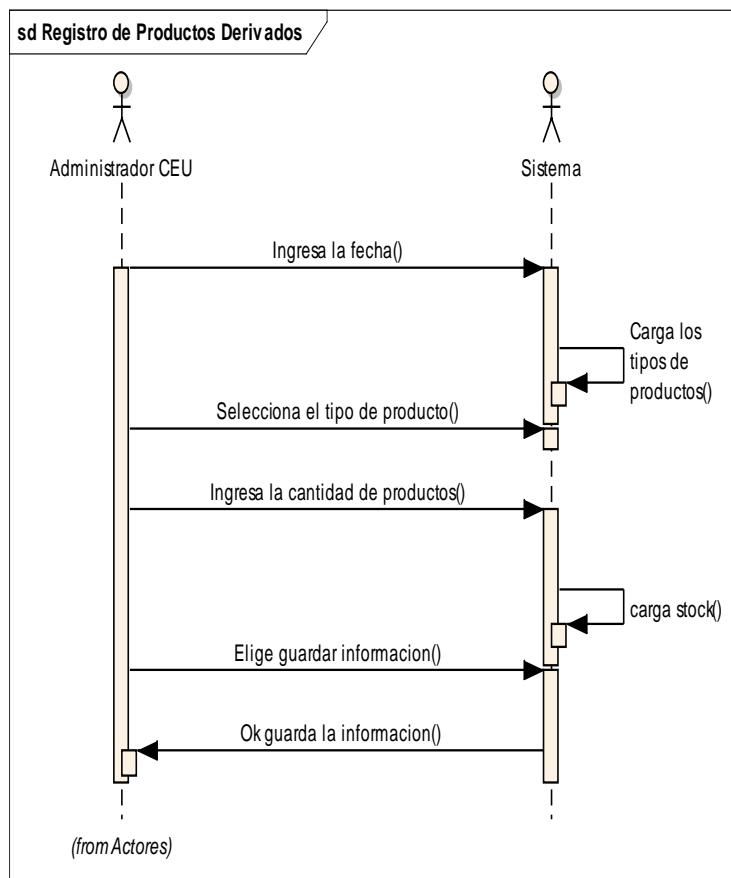
Tabla 6



3.4.7. CASO DE USO REGISTRO DE PRODUCTOS DERIVADOS



3.4.7.1. DIAGRAMA DE SECUENCIA DE PRODUCTOS DERIVADOS



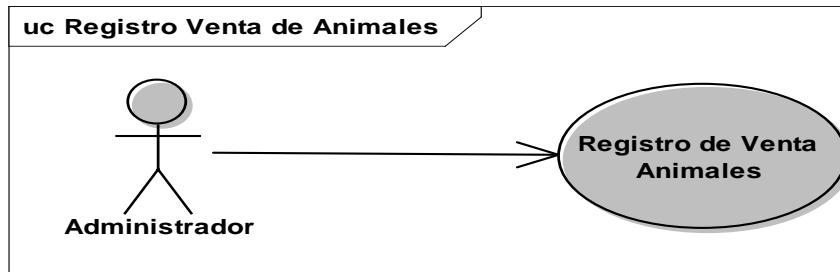


Nombre:	REGISTRO DE PRODUCTOS DERIVADOS
Sistema:	
Descripción:	Registrar en el sistema la información del registro de productos derivados.
Actores:	Administrador
Precondiciones:	<p>El encargado deberá haber ingresado todos los datos en el sistema:</p> <p>El sistema desplegará un formulario en el cual el usuario deberá ingresar los siguientes datos.</p>
Flujo Normal:	<ol style="list-style-type: none">1. El usuario ingresa la identificación del derivado.2. El usuario ingresa la fecha de registro del derivado.3. El usuario selecciona el tipo de producto.4. El usuario ingresa la cantidad.5. El usuario elige guardar información.6. El sistema guarda la información
Flujo Alternativo:	<ul style="list-style-type: none">▪ El Administrador ingresa datos en formulario del sistema.▪ El Administrador guarda los últimos datos ingresados una vez que haya terminado.
Pos condiciones:	El mensaje ha sido almacenado en el sistema.

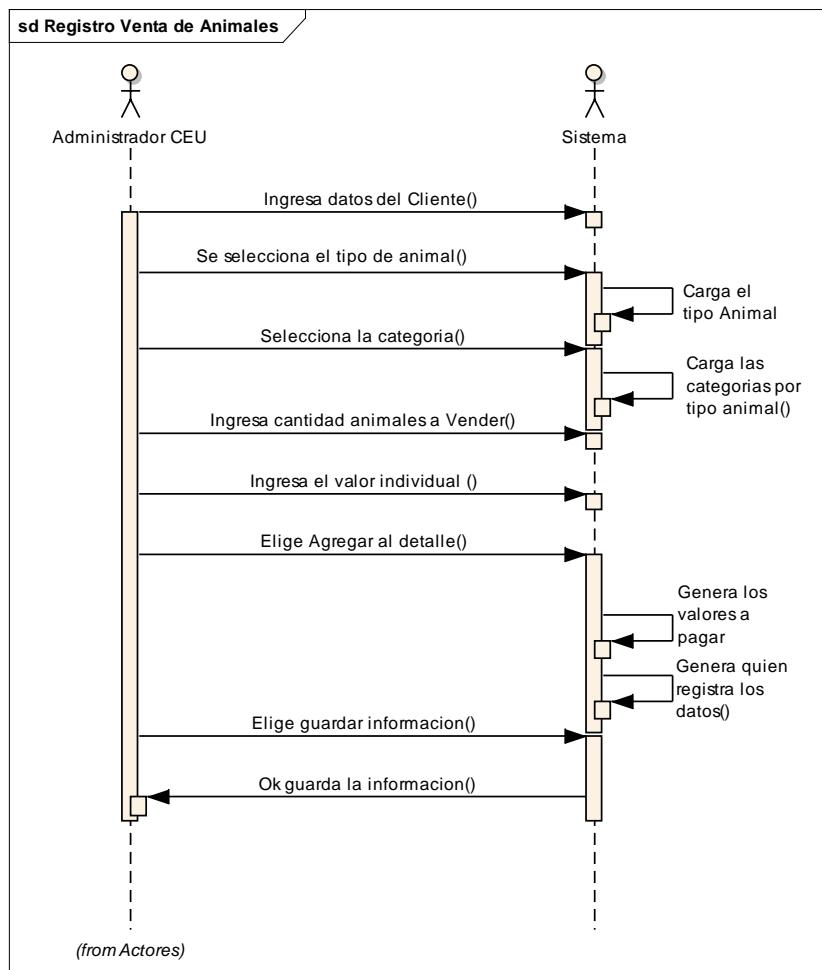
Tabla 7



3.4.8. CASO DE USO DE REGISTRO DE VENTA DE ANIMALES



3.4.8.1. DIAGRAMA DE SECUENCIA DEL REGISTRO DE VENTA DE ANIMALES





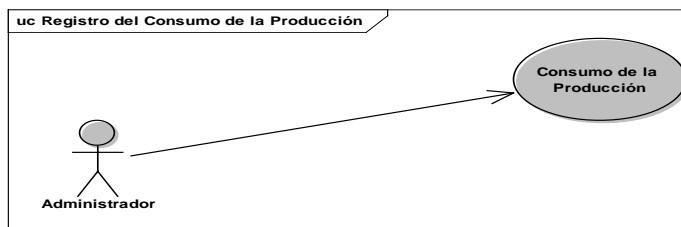
UNIVERSIDAD CENTRAL DEL ECUADOR
Sistema de Registro de Animales y sus Derivados del Centro Experimental Uyumbicho

Nombre:	REGISTRO DE LA VENTA DE ANIMALES
Sistema:	
Descripción:	Registrar en el sistema la información del registro de Venta de Animales
Actores:	Administrador
Precondiciones:	<p>El encargado deberá haber ingresado todos los datos en el sistema: El sistema desplegará un formulario en el cual el usuario deberá ingresar los siguientes datos.</p>
Flujo Normal:	<p>1. El usuario ingresa los datos del Cliente 2. El usuario selecciona el tipo de Animal (Animal Grupal o Individual)</p> <p>Si es Animal Grupal:</p> <ul style="list-style-type: none">- Se selecciona la categoría- Se ingresa la cantidad a vender- Se guarda la información <p>Si se Animal Individual:</p> <ul style="list-style-type: none">- Se selecciona la categoría,- Se selecciona el animal- Se ingresa el valor- Se guarda la información. <p>3. El sistema guarda la información</p>
Flujo Alternativo:	<ul style="list-style-type: none">▪ El Administrador ingresa datos en formulario del sistema.▪ El Administrador guarda los últimos datos ingresados una vez que haya terminado.
Pos condiciones:	El mensaje ha sido almacenado en el sistema.

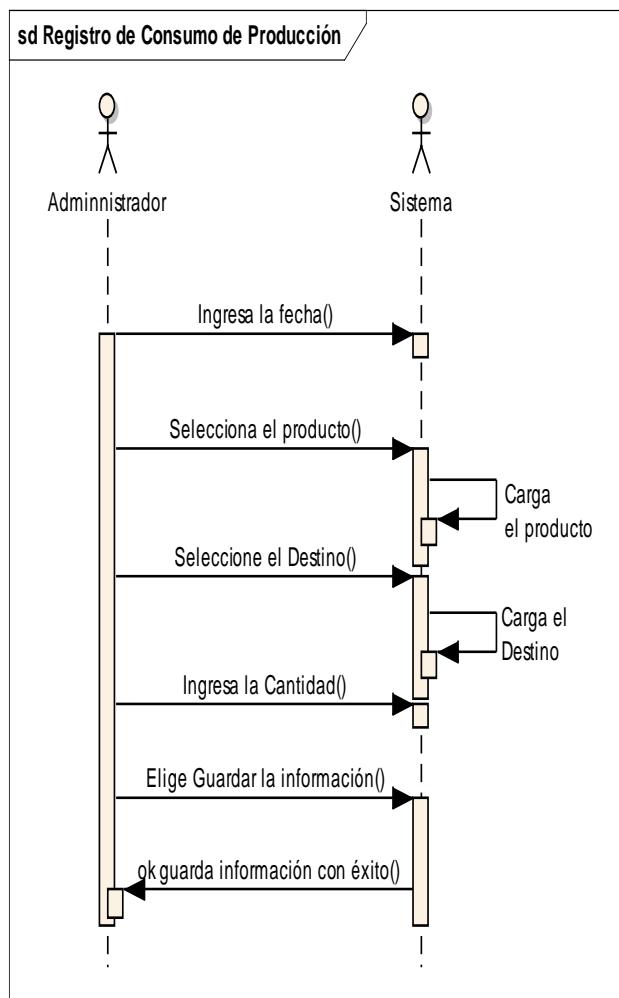
Tabla 8



3.4.9. CASO DE USO DE REGISTRO DEL CONSUMO DE LA PRODUCCIÓN



3.4.9.1. DIAGRAMA DE SECUENCIA DEL CONSUMO DE LA PRODUCCIÓN





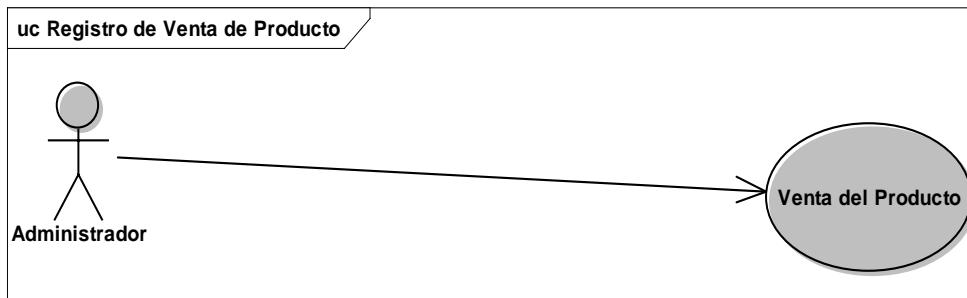
UNIVERSIDAD CENTRAL DEL ECUADOR
Sistema de Registro de Animales y sus Derivados del Centro Experimental Uyumbicho

Nombre:	REGISTRO DEL CONSUMO DE LA PRODUCCIÓN
Sistema:	
Descripción: Registrar en el sistema la información del registro de Consumo de la Producción.	
Actores Administrador CEU	
Precondiciones: El encargado deberá haber ingresado todos los datos en el sistema: El sistema desplegará un formulario en el cual el usuario deberá ingresar los siguientes datos.	
Flujo Normal: 1. El usuario ingresa la fecha 2. El usuario selecciona el producto. 3. El usuario selecciona el destino 4- El usuario ingresa la cantidad. 5. El sistema guarda la información	
Flujo Alternativo: Ninguno.	
Pos condiciones: El mensaje ha sido almacenado en el sistema.	

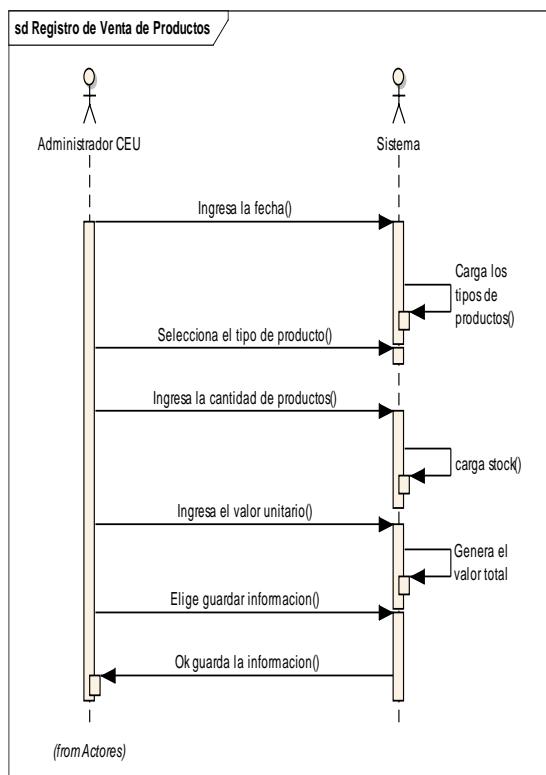
Tabla 9



3.4.10. CASO DE USO DEL REGISTRO DE LA VENTA DE PRODUCTOS



3.4.10.1. DIAGRAMA DE SECUENCIA DEL REGISTRO DE LA VENTA DE PRODUCTOS





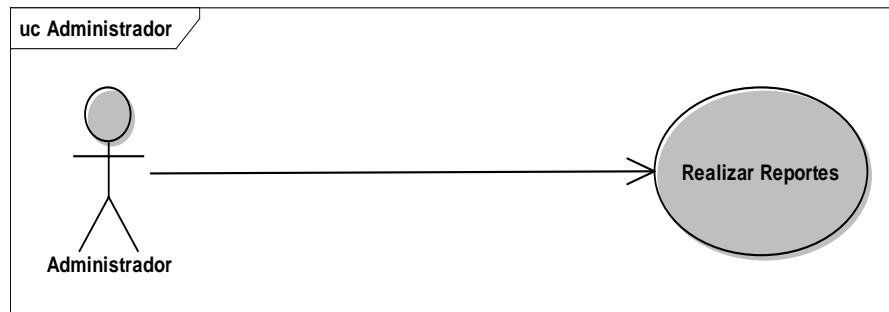
UNIVERSIDAD CENTRAL DEL ECUADOR
Sistema de Registro de Animales y sus Derivados del Centro Experimental Uyumbicho

Nombre:	REGISTRO DE LA VENTA DE PRODUCTOS
Sistema:	
Descripción:	Registrar en el sistema la información del registro de realizar reportes
Actores	Administrador CEU
Precondiciones:	<p>El encargado deberá haber ingresado todos los datos en el sistema: El sistema desplegará un formulario en el cual el usuario deberá ingresar los siguientes datos.</p>
Flujo Normal:	<ol style="list-style-type: none">1. El usuario ingresa la fecha de venta2. El usuario selecciona el cliente3. El usuario ingresa nombre del producto4. El usuario ingresa la cantidad del producto5. El usuario ingresa el valor unitario6. El sistema genera el valor total7. El usuario selecciona guarda la información8. El sistema guarda la información
Flujo Alternativo:	Ninguno.
Pos condiciones:	El mensaje ha sido almacenado en el sistema.

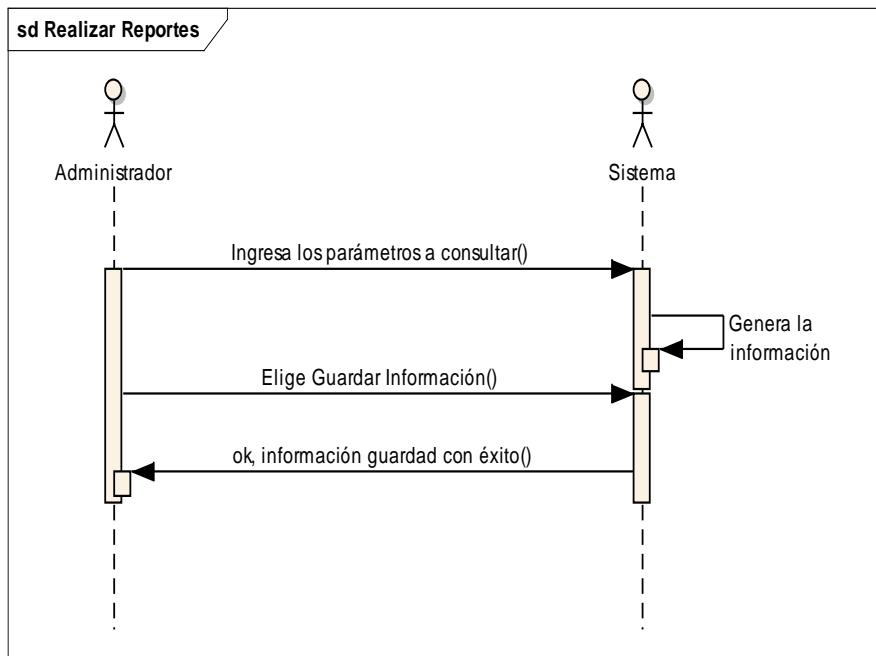
Tabla 10



3.4.11. CASO DE USO REALIZAR REPORTES



3.4.11.1. DIAGRAMA DE SENCUENCIA REALIZAR REPORTES



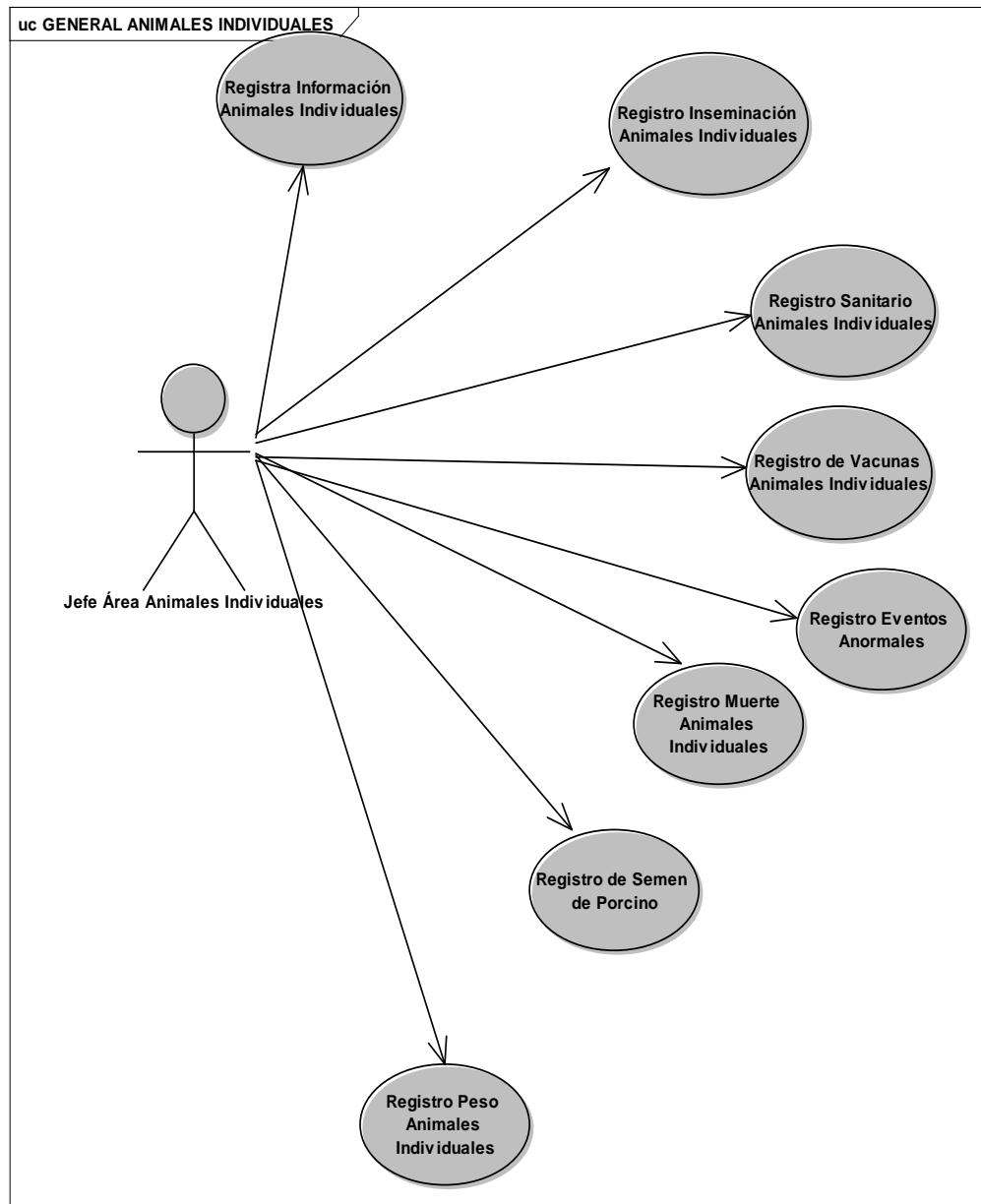


Nombre:	REALIZAR REPORTES
Sistema:	
Descripción:	Registrar en el sistema la información del registro de realizar reportes
Actores	Administrador CEU
Precondiciones:	<p>El encargado deberá haber ingresado todos los datos en el sistema: El sistema desplegará un formulario en el cual el usuario deberá ingresar los siguientes datos.</p>
Flujo Normal:	<ol style="list-style-type: none">1. El usuario ingresa los parámetros a consultar2. El sistema guarda la información
Flujo Alternativo:	Ninguno.
Pos condiciones:	El mensaje ha sido almacenado en el sistema.

Tabla 11

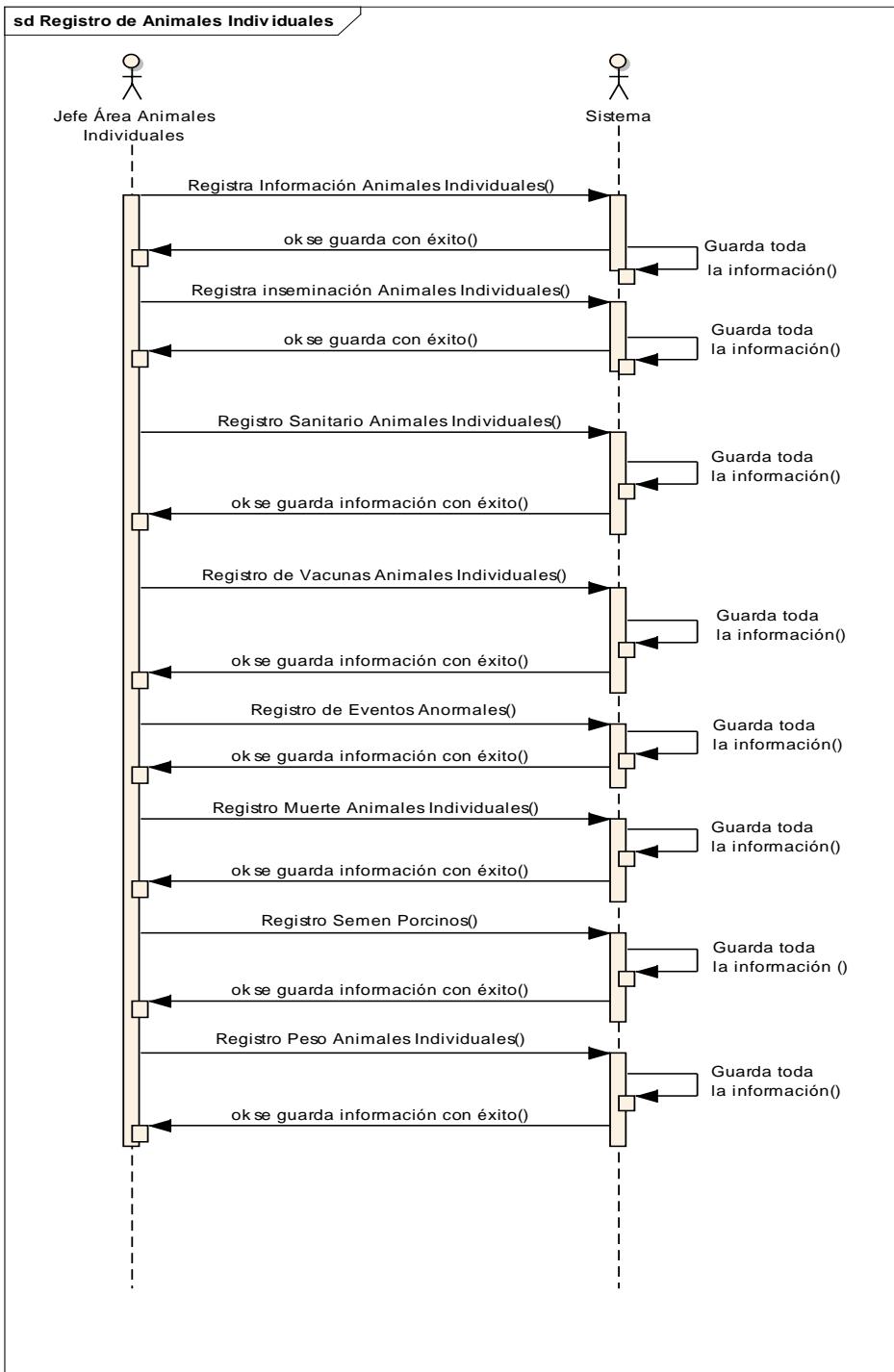


3.5. CASO DE USO EN GENERAL ANIMALES INDIVIDUALES





3.5.1. DIAGRAMA DE SECUENCIA GENERAL ANIMALES INDIVIDUALES





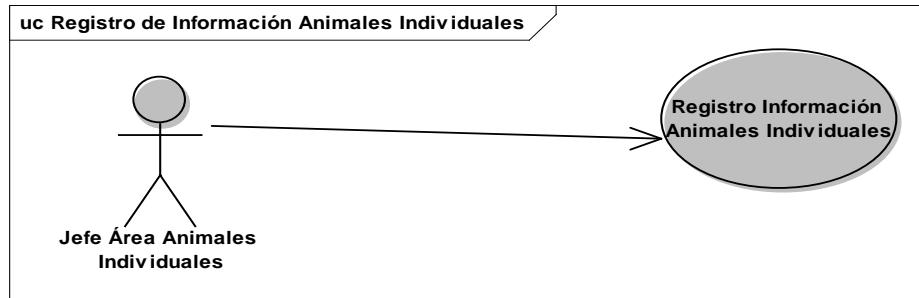
UNIVERSIDAD CENTRAL DEL ECUADOR
Sistema de Registro de Animales y sus Derivados del Centro Experimental Uyumbicho

Nombre:	REGISTRO GENERAL DE ANIMALES INDIVIDUALES
Sistema:	
Descripción:	Registrar en el sistema toda la información general de Animales Individuales.
Actores:	Jefe de Área de Animales Individuales
Precondiciones:	El encargado deberá haber ingresado todos los datos en el sistema: El sistema desplegará un formulario en el cual el usuario deberá ingresar los siguientes datos.
Flujo Normal:	<ol style="list-style-type: none">1. Se registra información de Animales Individuales2 Se registra inseminación de Animales Individuales3. Se registra Control Sanitario de Animales Individuales4. Se registra Vacunas Animales Individuales5. Se registra Eventos Anormales Animales Individuales6. Se registra Muerte de Animales Individuales7. Se registra Control Sanitario de Animales Individuales8. Se registra la Venta del Animales Individuales9. Se registra el Peso del Animales Individuales10 .El sistema guarda información.
Flujo Alternativo:	El Jefe de área ingresa datos en formulario del sistema. El Jefe de área guarda los últimos datos ingresados una vez que haya terminado.
Pos condiciones:	El mensaje ha sido almacenado en el sistema.

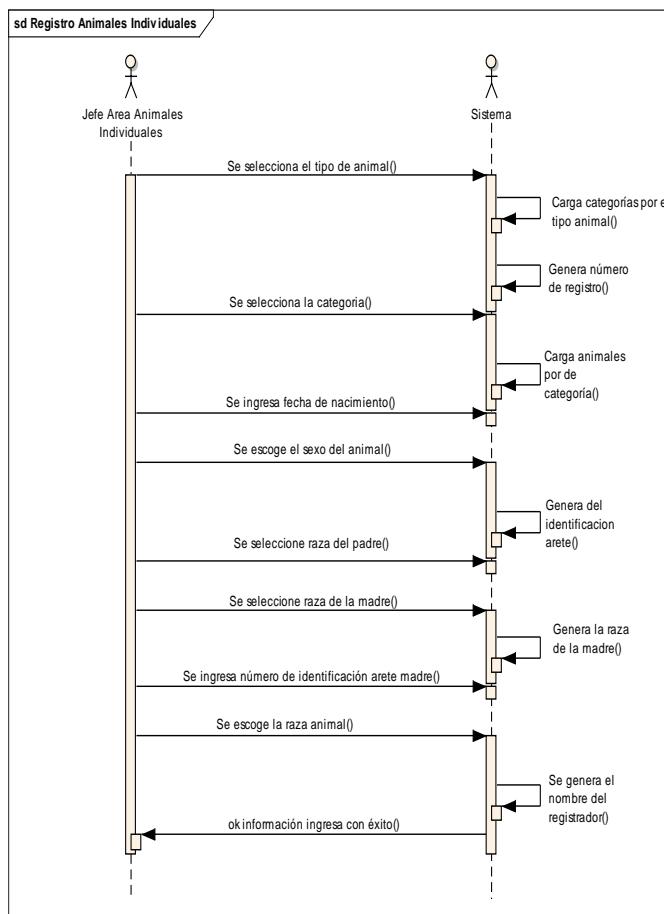
Tabla 12



3.5.2. CASO DE USO DE REGISTRO DE INFORMACIÓN DE ANIMALES INDIVIDUALES



3.5.2.1. DIAGRAMA DE SECUENCIA DEL REGISTRO DE INFORMACIÓN ANIMALES INDIVIDUALES





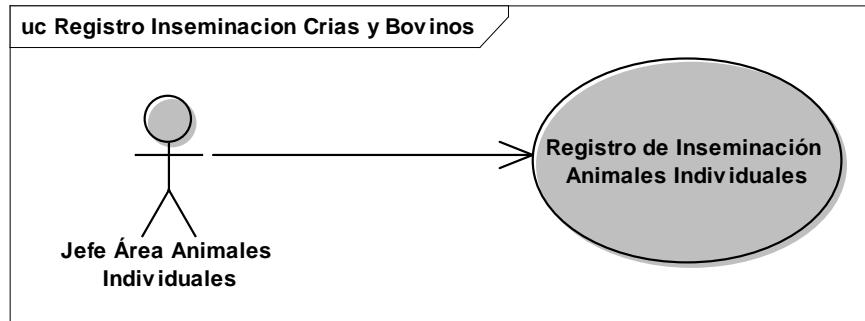
UNIVERSIDAD CENTRAL DEL ECUADOR
Sistema de Registro de Animales y sus Derivados del Centro Experimental Uyumbicho

Nombre:	REGISTRO DE INFORMACIÓN ANIMALES INDIVIDUALES
Sistema:	
Descripción:	Registrar en el sistema la información de Animales Individuales
Actores:	Jefe de Área de Animales Individuales
Precondiciones:	El encargado deberá haber ingresado todos los datos en el sistema: El sistema desplegará un formulario en el cual el usuario deberá ingresar los siguientes datos.
Flujo Normal:	<ol style="list-style-type: none">1. El usuario ingresa la fecha de nacimiento2. El usuario selecciona la el sexo del animal(Macho o hembra)3 .El sistema genera el número de arete4. El usuario elige la raza del padre.5. El usuario elige la raza de la madre6- El usuario ingresa el código de la madre.7- El usuario elige la raza8. El usuario elige la categoría de la bobino9.El sistema genera el número de registro10. El usuario selecciona guardar información11. El sistema guarda información
Flujo Alternativo:	El Jefe de área ingresa datos en formulario del sistema. El Jefe de área guarda los últimos datos ingresados una vez que haya terminado.
Pos condiciones:	El mensaje ha sido almacenado en el sistema.

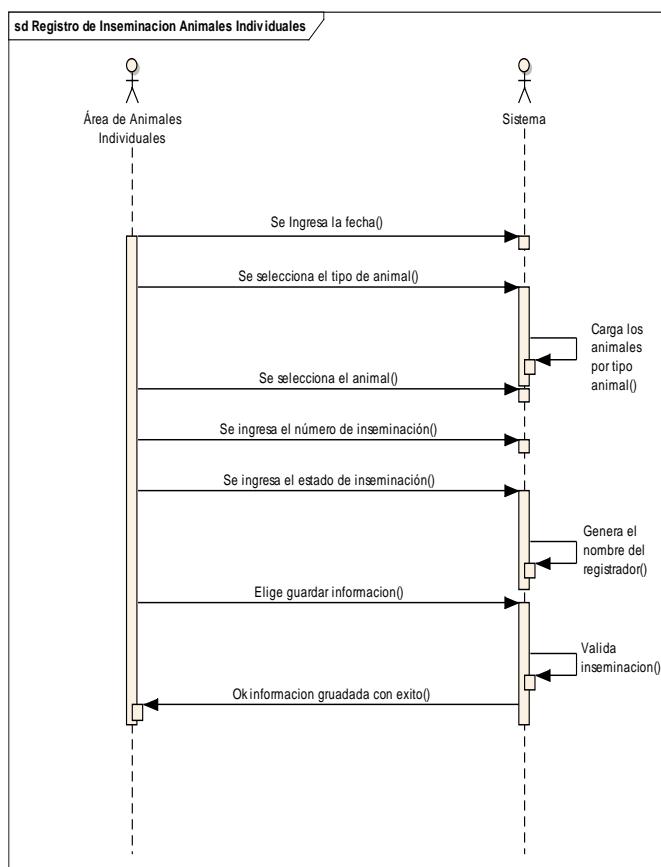
Tabla 13



3.5.3. CASO DE USO DEL REGISTRO DE INSEMINACIÓN DE ANIMALES INDIVIDUALES



3.5.3.1. DIAGRAMA DE SECUENCIA DEL REGISTRO INSEMINACIÓN ANIMALES INDIVIDUALES





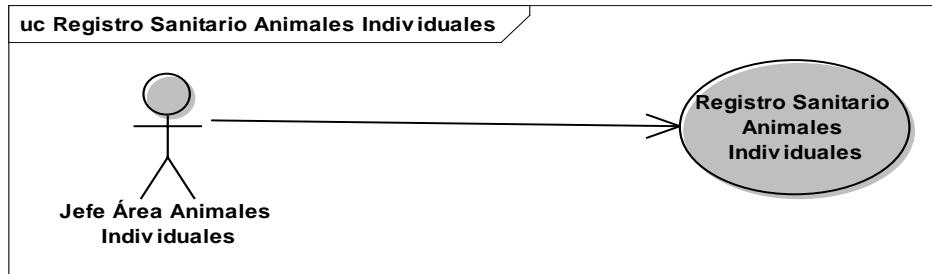
Nombre:	REGISTRO INSEMINACIÓN ANIMALES INDIVIDUALES
Sistema:	
Descripción:	Registrar en el sistema la información de inseminación de Animales Individuales.
Actores:	Jefe de Área Animales Individuales
Precondiciones:	<p>El encargado deberá haber ingresado todos los datos en el sistema:</p> <p>El sistema desplegará un formulario en el cual el usuario deberá ingresar los siguientes datos.</p>
Flujo Normal:	<ol style="list-style-type: none">1. El usuario ingresa la fecha2. El usuario selecciona el tipo de animal3. El sistema devuelve tipo de animal seleccionada4. El usuario ingresa el número de inseminación5. El usuario ingresa el estado de la inseminación6. El sistema guarda información.
Flujo Alternativo:	<ul style="list-style-type: none">▪ El Jefe de área ingresa datos en formulario del sistema.▪ El Jefe de área guarda los últimos datos ingresados una vez que haya terminado.
Pos condiciones:	El mensaje ha sido almacenado en el sistema.

Tabla 14

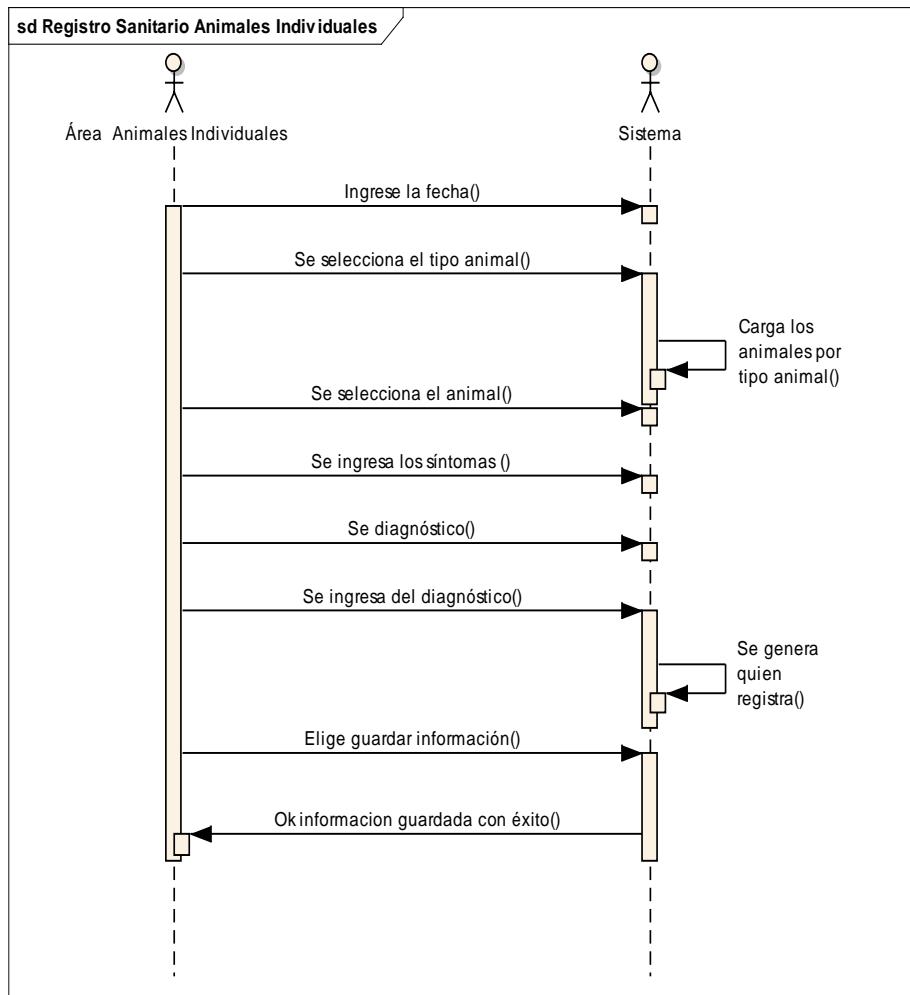


3.5.4. CASO DE USO DEL REGISTRO SANITARIO DE ANIMALES

INDIVIDUALES



3.5.4.1 DIAGRAMA DE SECUENCIA DEL REGISTRO SANITARIO DE ANIMALES INDIVIDUALES





UNIVERSIDAD CENTRAL DEL ECUADOR
Sistema de Registro de Animales y sus Derivados del Centro Experimental Uyumbicho

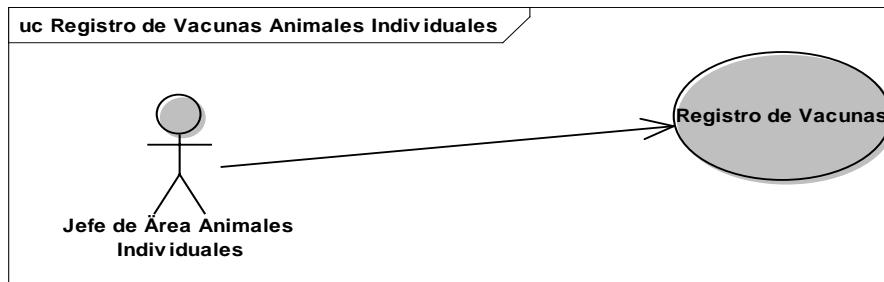
Nombre:	REGISTRO SANITARIO DE ANIMALES INDIVIALES
Sistema:	
Descripción:	Registrar en el sistema la información del registro sanitario Animales Individuales
Actores:	Jefe de Área de Animales Individuales
Precondiciones:	<p>El encargado deberá haber ingresado todos los datos en el sistema:</p> <p>El sistema desplegará un formulario en el cual el usuario deberá ingresar los siguientes datos.</p>
Flujo Normal:	<ol style="list-style-type: none">1. El usuario ingresa la fecha2. El usuario selecciona el tipo de animal3. El sistema devuelve tipo de animal seleccionado4. El usuario ingresa el número de identificación del animal y elige buscar5. El sistema busca y devuelve el animal correspondiente a la identificación6. El usuario ingresa la fecha del chequeo7. El usuario ingresa los síntomas al chequeo8. El usuario ingresa el diagnóstico veterinario9. El usuario ingresa el tratamiento10. El usuario selecciona guarda la información11. El sistema guarda la información
Flujo Alternativo:	<ul style="list-style-type: none">▪ El Jefe de área ingresa datos en formulario del sistema.▪ El Jefe de área guarda los últimos datos ingresados una vez que haya terminado.
Pos condiciones:	El mensaje ha sido almacenado en el sistema.

Tabla 15



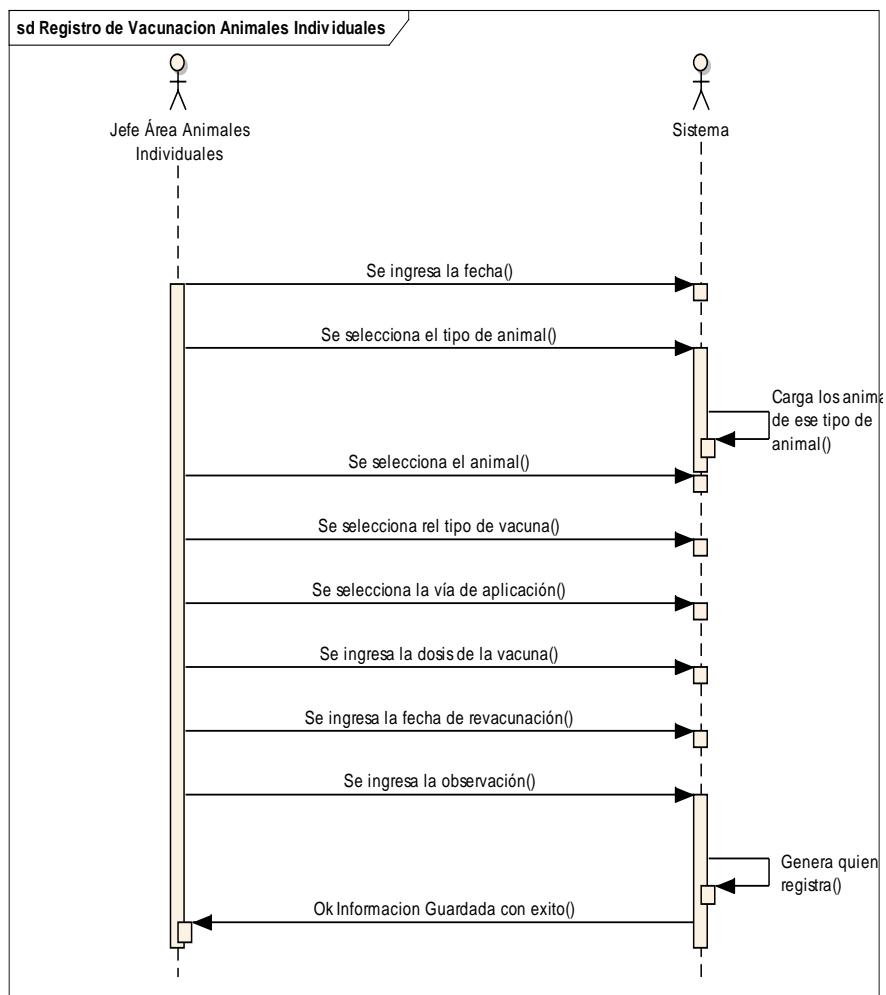
3.5.5. CASO DE USO DEL REGISTRO DE VACUNAS ANIMALES

INDIVIDUALES



3.5.5.1. DIAGRAMA DE SECUENCIA DEL REGISTRO DE VACUNAS

ANIMALES INDIVIDUALES





UNIVERSIDAD CENTRAL DEL ECUADOR
Sistema de Registro de Animales y sus Derivados del Centro Experimental Uyumbicho

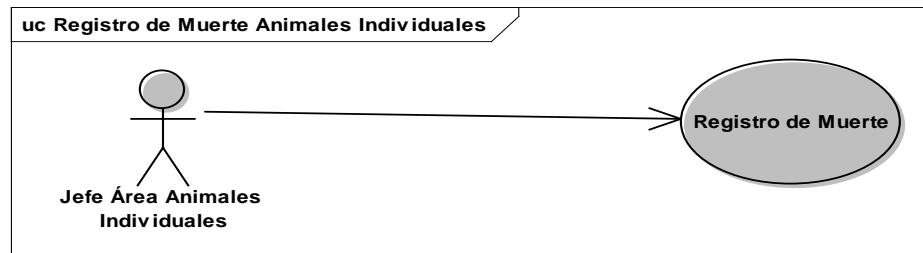
Nombre:	REGISTRO DE VACUNAS ANIMALES INDIVIDUALES
Sistema:	
Descripción:	Registrar en el sistema la información del registro de vacunas Animales Individuales.
Actores:	Jefe de Área Animales Individuales
Precondiciones:	<p>El encargado deberá haber ingresado todos los datos en el sistema:</p> <p>El sistema desplegará un formulario en el cual el usuario deberá ingresar los siguientes datos.</p>
Flujo Normal:	<ol style="list-style-type: none">1. El usuario ingresa el número de arete y elige buscar2.- El sistema busca y devuelve el bovino correspondiente al número de arete3.- El usuario ingresa la fecha de colocación de la vacuna4.- El usuario ingresa el nombre de la vacuna5.- El usuario ingresa la fecha de revacunación de la vacuna6.- El usuario selecciona y guarda la información7.-El sistema guarda la información
Flujo Alternativo:	<ul style="list-style-type: none">▪ El Jefe de área ingresa datos en formulario del sistema.▪ El Jefe de área guarda los últimos datos ingresados una vez que haya terminado.
Pos condiciones:	El mensaje ha sido almacenado en el sistema.

Tabla 16



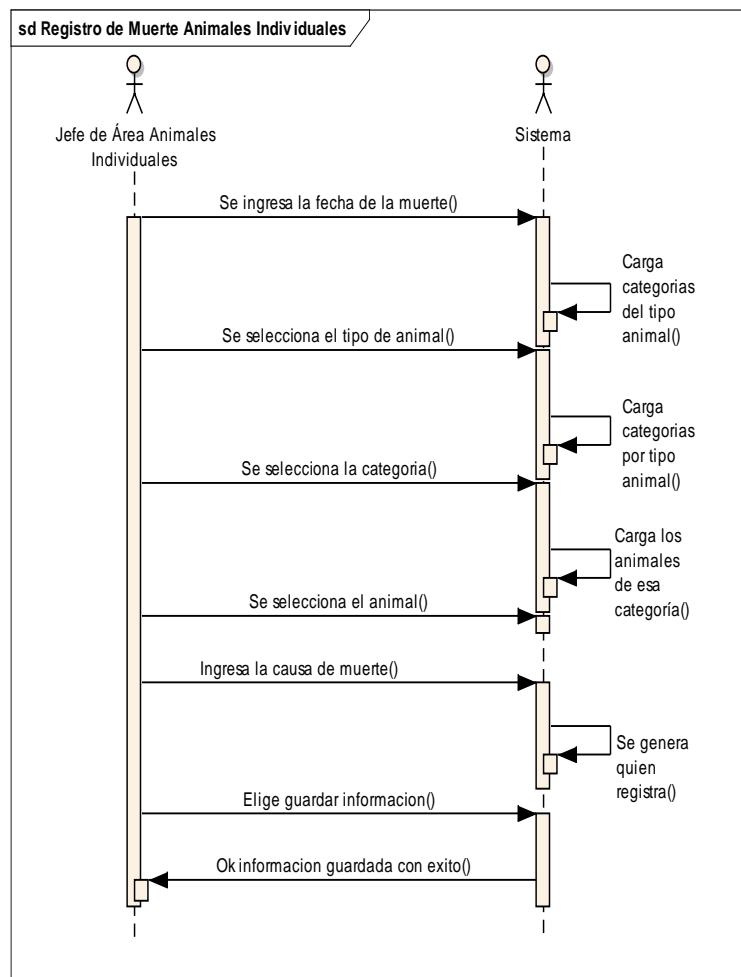
3.5.6. CASO DE USO DEL REGISTRO DE MUERTE ANIMALES

INDIVIDUALES



3.5.6.1. DIAGRAMA DE SECUENCIA REGISTRO DE MUERTE

ANIMALES INDIVIDUALES





UNIVERSIDAD CENTRAL DEL ECUADOR
Sistema de Registro de Animales y sus Derivados del Centro Experimental Uyumbicho

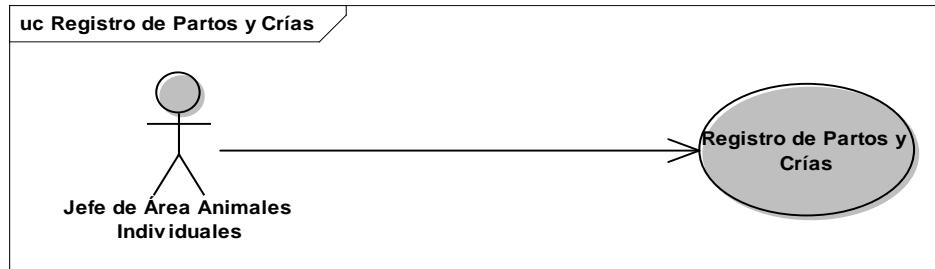
Nombre:	REGISTRO DE MUERTE ANIMALES INDIVIDUALES
Sistema:	
Descripción:	Registrar en el sistema la información del registro de la muerte de Bovinos y Porcinos.
Actores:	Jefe de Área de Animales Individuales
Precondiciones:	El encargado deberá haber ingresado todos los datos en el sistema. El sistema desplegará un formulario en el cual el usuario deberá ingresar los siguientes datos.
Flujo Normal:	<ol style="list-style-type: none">1. El usuario selecciona la clase de animal.2. El sistema devuelve la clase de animal seleccionada.3. El usuario ingresa el número de identificado del animal y elige buscar.4. El sistema busca y devuelve el animal correspondiente a la identificación.5. El usuario ingresa la fecha de la muerte.6. El usuario ingresa la causa de la muerte.7. El usuario selecciona guarda la información.8. El sistema guarda la información.
Flujo Alternativo:	<ul style="list-style-type: none">• El Jefe de área ingresa datos en formulario del sistema.• El Jefe de área guarda los últimos datos ingresados una vez que haya terminado.
Pos condiciones:	El mensaje ha sido almacenado en el sistema.

Tabla 17

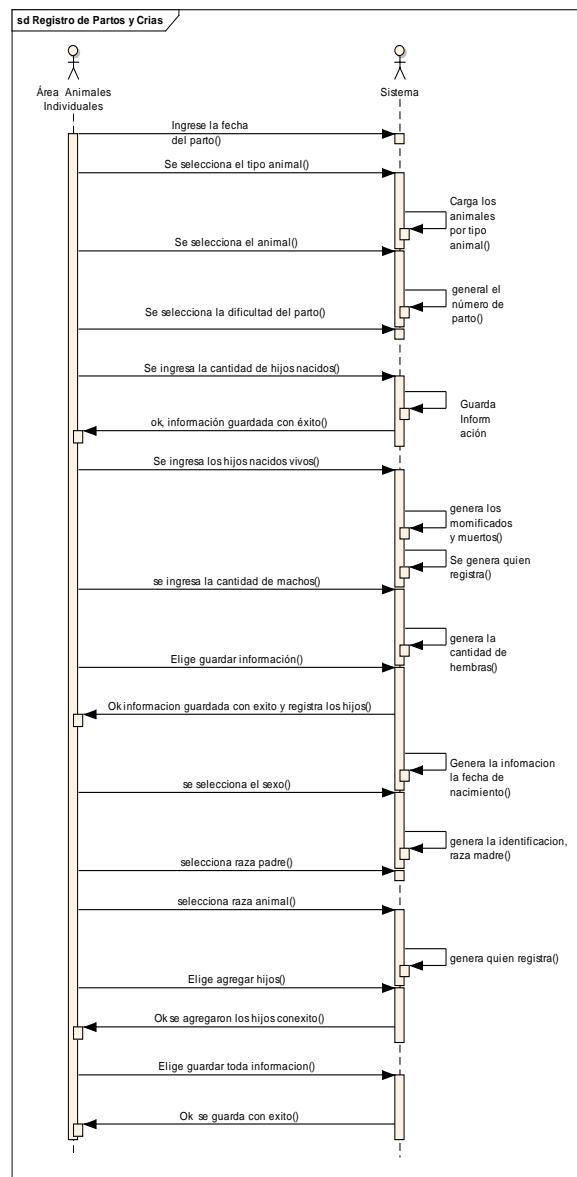


3.5.7. CASO DE USO DE REGISTRO DE PARTOS Y CRÍAS

ANIMALES INDIVIDUALES



3.5.7.1. DIAGRAMA DE SECUENCIA DEL REGISTRO DE PARTOS Y CRÍAS ANIMALES INDIVIAUALES





UNIVERSIDAD CENTRAL DEL ECUADOR
Sistema de Registro de Animales y sus Derivados del Centro Experimental Uyumbicho

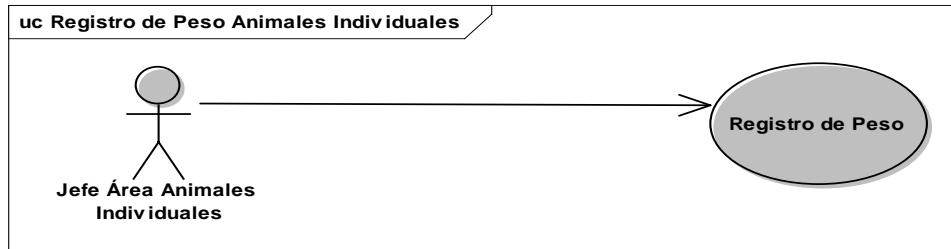
Nombre:	REGISTRO DE PARTOS Y CRIAS DE ANIMALES INDIVIDUALES
Sistema:	
Descripción:	Registrar en el sistema la información del registro de partos y crías de Animales Individuales
Actores:	Jefe de Área de Animales Individuales.
Precondiciones:	El encargado deberá haber ingresado todos los datos en el sistema: El sistema desplegará un formulario en el cual el usuario deberá ingresar los siguientes datos.
Flujo Normal:	<ol style="list-style-type: none">1. El usuario ingresa la fecha de parto.2. El usuario selecciona el tipo de animal.3. El usuario selecciona al animal.4. El usuario ingresa la dificultad del parto.5. El usuario ingresa hijos nacidos vivos, momificados6. El sistema genera los muertos7. El usuario ingresa el número de machos.8. El sistema genera las hembras.9. El sistema guarda la información.10. El sistema genera la información de la fecha de nacimiento del hijo.11. El usuario selecciona el sexo.12. El sistema genera datos del hijo.13. El sistema guarda la información.
Flujo Alternativo:	<ul style="list-style-type: none">• El Jefe de área ingresa datos en formulario del sistema.• El Jefe de área guarda los últimos datos ingresados una vez que haya terminado.
Pos condiciones:	El mensaje ha sido almacenado en el sistema.

Tabla 18



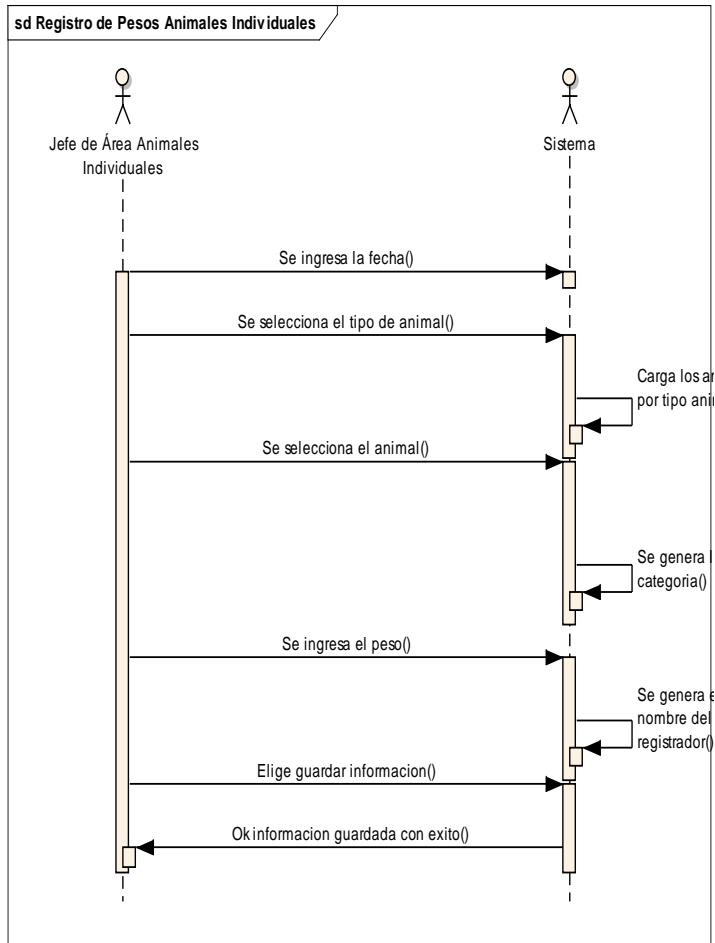
3.5.8. CASO DE USO DE REGISTRO DE PESOS ANIMALES

INDIVIDUALES



3.5.8.1. DIAGRAMA DE SECUENCIA DE REGISTRO DE PESOS

ANIMALES INDIVIDUALES





UNIVERSIDAD CENTRAL DEL ECUADOR
Sistema de Registro de Animales y sus Derivados del Centro Experimental Uyumbicho

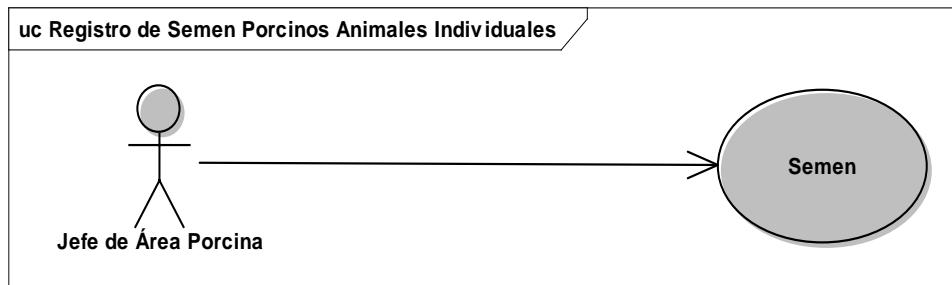
Nombre:	REGISTRO DE PESOS ANIMALES INDIVIDUALES
Sistema:	
Descripción:	Registrar en el sistema la información del registro de pesos Animales Individuales.
Actores:	Jefe de Área de Animales Individuales.
Precondiciones:	<p>El encargado deberá haber ingresado todos los datos en el sistema:</p> <p>El sistema desplegará un formulario en el cual el usuario deberá ingresar los siguientes datos.</p>
Flujo Normal:	<ol style="list-style-type: none">1. El usuario ingresa el número de arete y elige buscar.2. El sistema busca y devuelve el bovino correspondiente al número de arete.3. El usuario ingresa la fecha del registro de peso.4. El usuario ingresa el peso del animal individual.5. El usuario selecciona guarda la información.6. El sistema guarda la información
Flujo Alternativo:	<ul style="list-style-type: none">▪ El Jefe de área ingresa datos en formulario del sistema.▪ El Jefe de área guarda los últimos datos ingresados una vez que haya terminado.
Pos condiciones:	El mensaje ha sido almacenado en el sistema.

Tabla 19



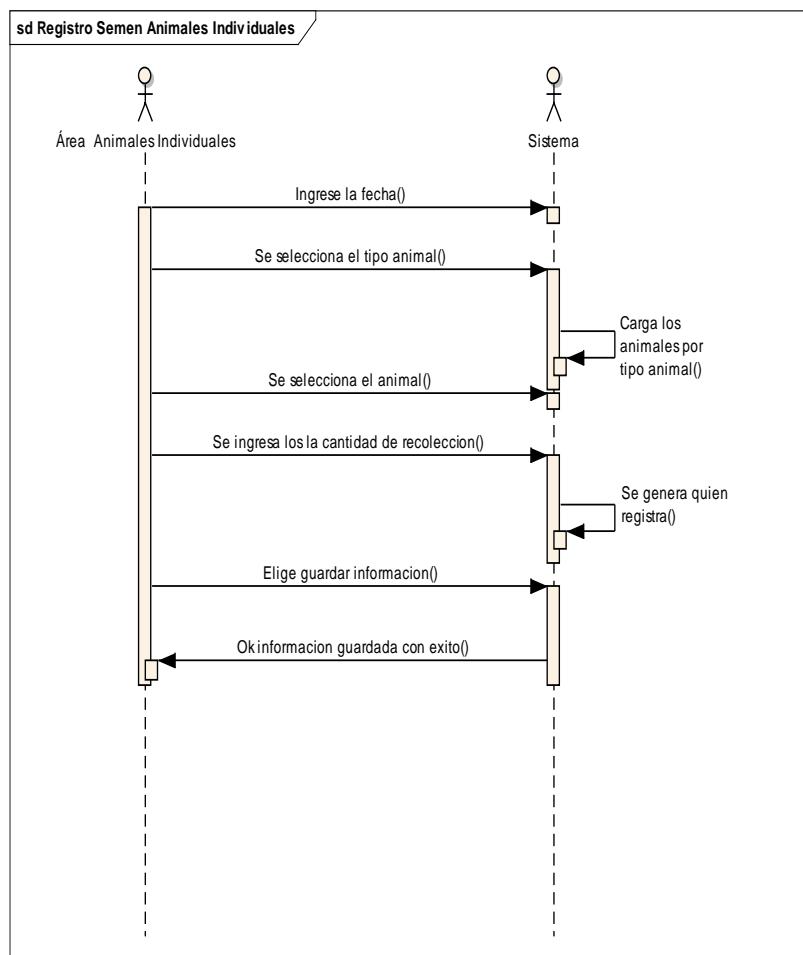
3.5.9. CASO DE USO DE REGISTRO DE SEMEN PORCINOS

ANIMALES INDIVIDUALES



3.5.9.1. DIAGRAMA DE SECUENCIA DE REGISTRO DE SEMEN

PORCINOS ANIMALES INDIVIDUALES





UNIVERSIDAD CENTRAL DEL ECUADOR
Sistema de Registro de Animales y sus Derivados del Centro Experimental Uyumbicho

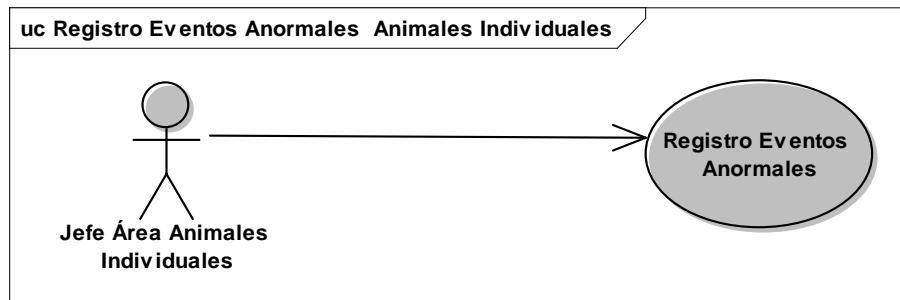
Nombre:	REGISTRO DE SEMEN PORCINOS ANIMALES INDIVIDUALES
Sistema:	
Descripción:	Registrar en el sistema la información del registro del semen de Porcinos (Animales Individuales).
Actores:	Jefe de Área Porcina.
Precondiciones:	<p>El encargado deberá haber ingresado todos los datos en el sistema:</p> <p>El sistema desplegará un formulario en el cual el usuario deberá ingresar los siguientes datos.</p>
Flujo Normal:	<ol style="list-style-type: none">1. El usuario ingresa la identificación del cerdoy elige buscar.2. El sistema busca y devuelve el cerdo correspondiente a la identificación.3. El usuario ingresa la fecha de recolección.4. El usuario ingresa la cantidad de eyaculado.5. El usuario selecciona guarda la información.6. El sistema guarda la información.
Flujo Alternativo:	<ul style="list-style-type: none">▪ El Jefe de área ingresa datos en formulario del sistema.▪ El Jefe de área guarda los últimos datos ingresados una vez que haya terminado.
Pos condiciones:	El mensaje ha sido almacenado en el sistema.

Tabla 20



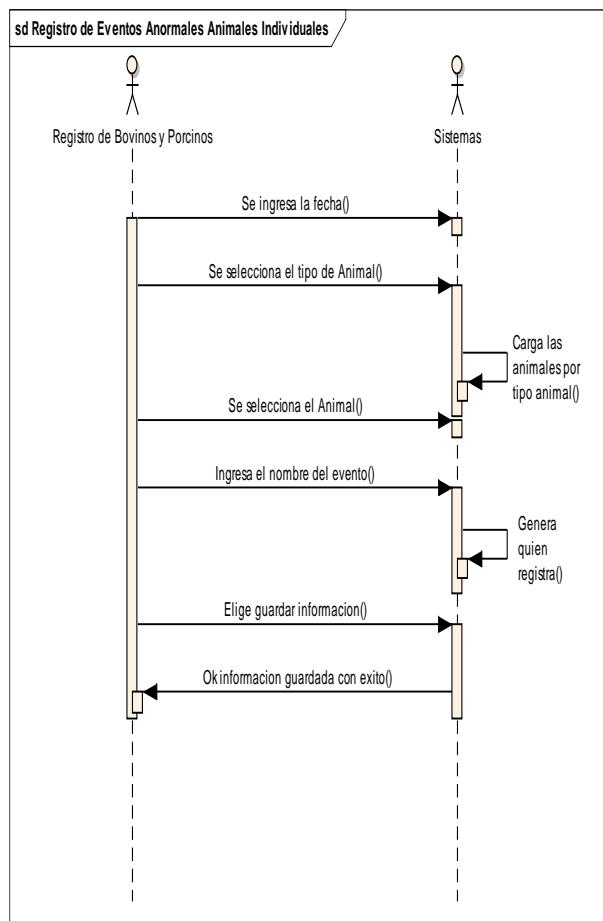
3.5. 10. CASO DE USO EVENTOS ANORMALES ANIMALES

INDIVIDUALES



3.5.10.1. DIAGRAMA DE FLUJO DE EVENTOS ANORMALES

ANIMALES INDIVIDUALES





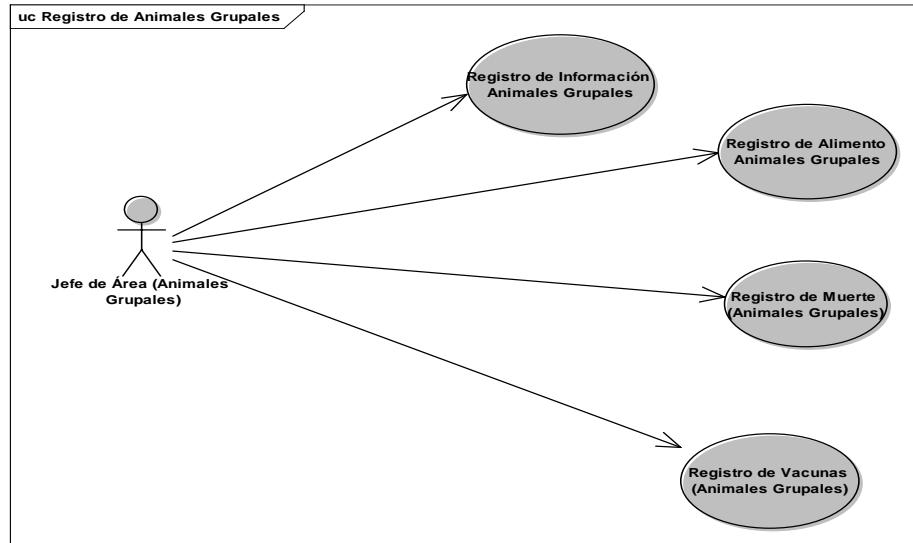
Nombre:	REGISTRO DE EVENTOS ANORMALES INDIVIDUALES ANIMALES
Sistema:	
Descripción:	Registrar en el sistema la información del registro de Eventos Anormales Animales Individuales.
Actores:	Jefe de Área de Animales Individuales.
Precondiciones:	<p>El encargado deberá haber ingresado todos los datos en el sistema:</p> <p>El sistema desplegará un formulario en el cual el usuario deberá ingresar los siguientes datos.</p>
Flujo Normal:	<ol style="list-style-type: none">1. El usuario ingresa la fecha.2. El usuario selecciona el tipo animal.3. El usuario selecciona el animal.4. El usuario ingresa el nombre del evento.5. El sistema guarda la información.
Flujo Alternativo:	<ul style="list-style-type: none">▪ El Jefe de área ingresa datos en formulario del sistema.▪ El Jefe de área guarda los últimos datos ingresados una vez que haya terminado.
Pos condiciones:	El mensaje ha sido almacenado en el sistema.

Tabla 21

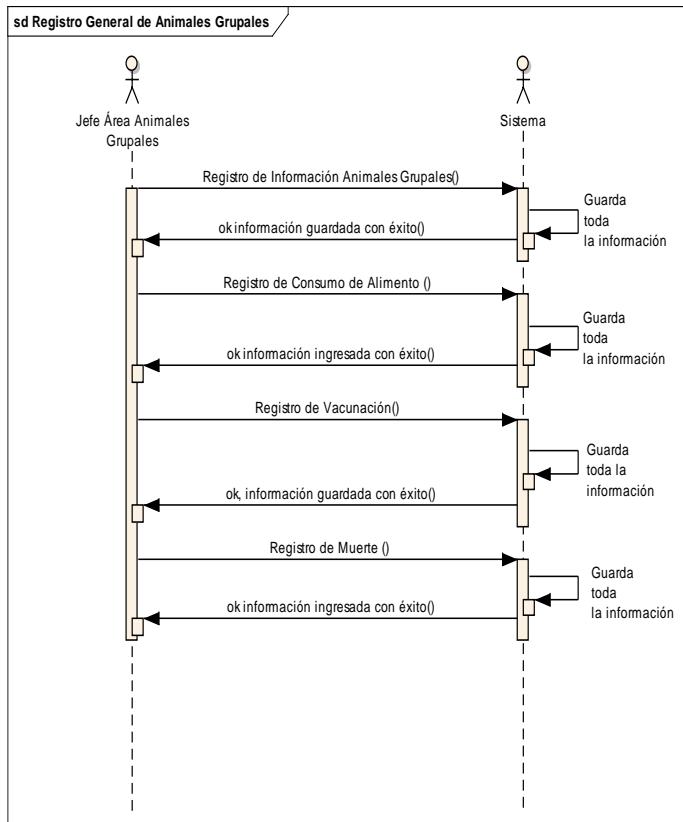


3.6. CASO DE USO DEL REGISTRO GENERAL ANIMALES

GRUPALES



3.6.1. DIAGRAMA DE SECUENCIA DEL REGISTRO GENERAL DE ANIMALES GRUPALES





UNIVERSIDAD CENTRAL DEL ECUADOR
Sistema de Registro de Animales y sus Derivados del Centro Experimental Uyumbicho

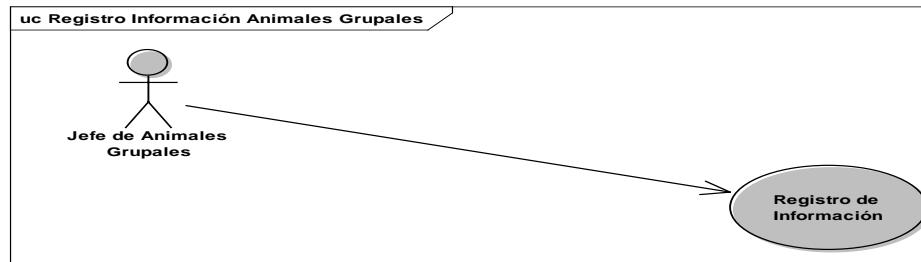
Nombre:	REGISTRO GENERAL ANIMALES GRUPALES
Sistema:	
Descripción:	Registrar en el sistema la información del registro General Animales Grupales.
Actores:	Jefe de Área Animales Grupales.
Precondiciones:	<p>El encargado deberá haber ingresado todos los datos en el sistema:</p> <p>El sistema desplegará un formulario en el cual el usuario deberá ingresar los siguientes datos.</p>
Flujo Normal:	<ol style="list-style-type: none">1. Se ingresa la información de Animales Grupales.2. Se ingresa el registro de alimento de Animales Grupales.3. Se ingresa el registro de vacunación de Animales Grupales.4. Se ingresa el registro de muerte de Animales Grupales.5. El sistema guarda la información.
Flujo Alternativo:	<ul style="list-style-type: none">▪ El Administrador ingresa datos en formulario del sistema.▪ El Administrador guarda los últimos datos ingresados una vez que haya terminado.
Pos condiciones:	El mensaje ha sido almacenado en el sistema.

Tabla 22



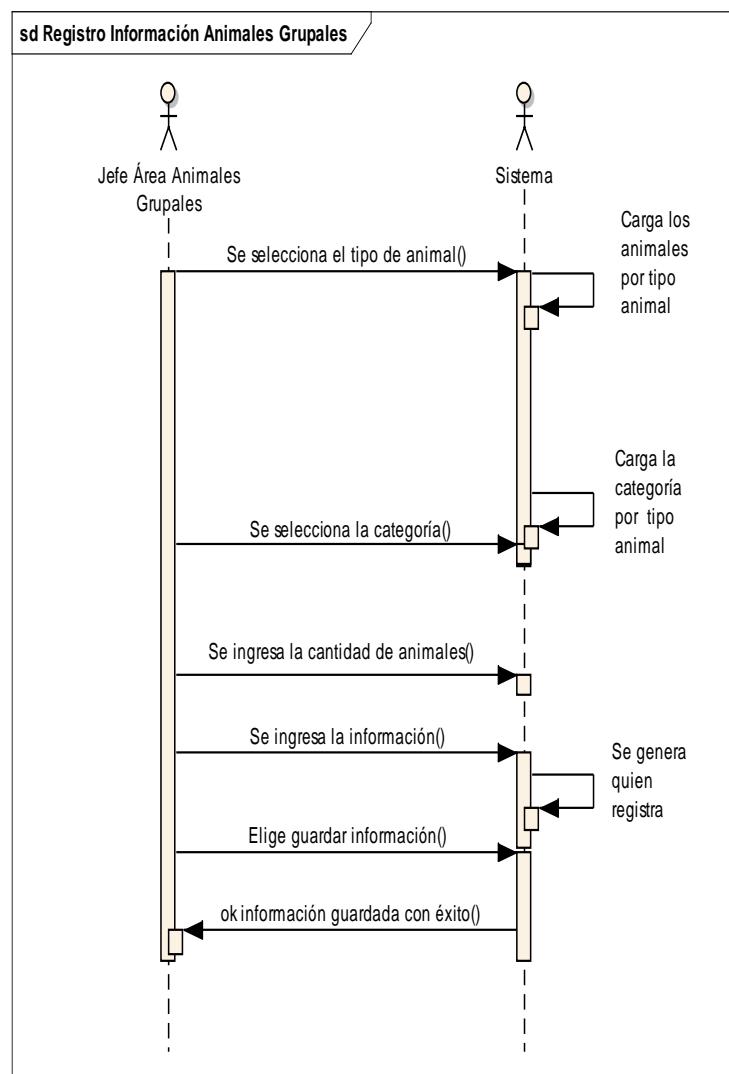
3.6.2. CASO DE USO DEL REGISTRO DE INFORMACIÓN

ANIMALES GRUPALES



3.6.2.1. DIAGRAMA DE SECUENCIA REGISTRO DE INFORMACIÓN

ANIMALES GRUPALES





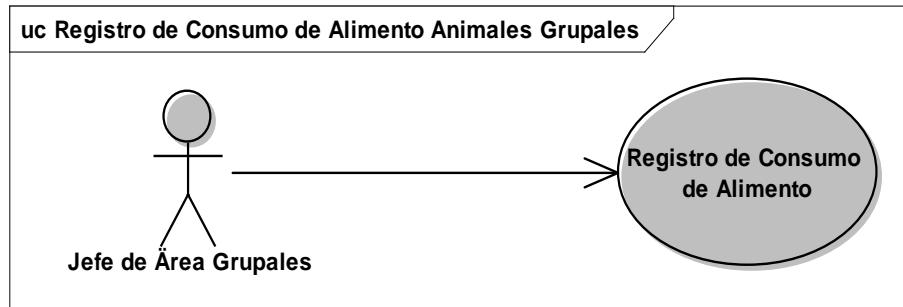
Nombre:	REGISTRO DE INFORMACIÓN ANIMALES GRUPALES
Sistema:	
Descripción:	Registrar en el sistema la información del registro de información Animales Grupales.
Actores:	Jefe de Área Animales Grupales.
Precondiciones:	<p>El encargado deberá haber ingresado todos los datos en el sistema:</p> <p>El sistema desplegará un formulario en el cual el usuario deberá ingresar los siguientes datos.</p>
Flujo Normal:	<ol style="list-style-type: none">1. El usuario selecciona la clase de animal.2. El sistema devuelve la clase seleccionada.3. El usuario selecciona la identificación de la categoría de animal y elige buscar.4. El sistema busca y devuelve la categoría del animal correspondiente a la identificación.5. El usuario Ingrera la fecha del registro.6. El usuario ingresa la cantidad de animales.7. El usuario selecciona guarda la información.8. El sistema guarda la información.
Flujo Alternativo:	<ul style="list-style-type: none">▪ El Jefe de área ingresa datos en formulario del sistema.▪ El Jefe de área guarda los últimos datos ingresados una vez que haya terminado.
Pos condiciones:	El mensaje ha sido almacenado en el sistema.

Tabla 23

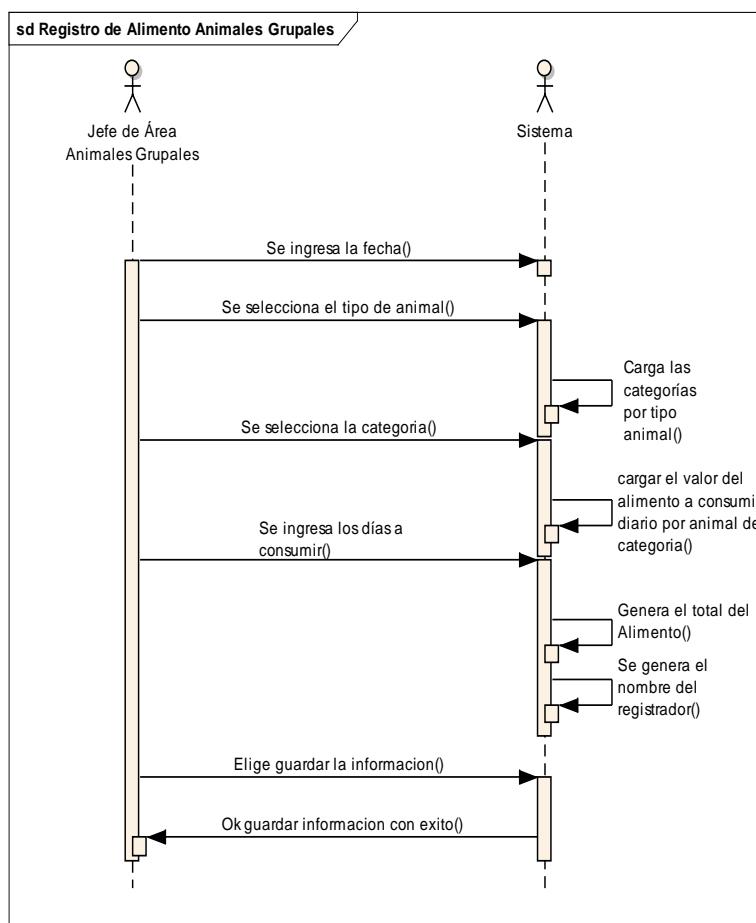


3.6.3. CASO DE USO DEL REGISTRO DE CONSUMO DE ALIMENTOS

ANIMALES GRUPALES



3.6.3.1. DIAGRAMA DE SECUENCIA DEL REGISTRO DE CONSUMO DE ALIMENTOS ANIMALES GRUPALES





UNIVERSIDAD CENTRAL DEL ECUADOR
Sistema de Registro de Animales y sus Derivados del Centro Experimental Uyumbicho

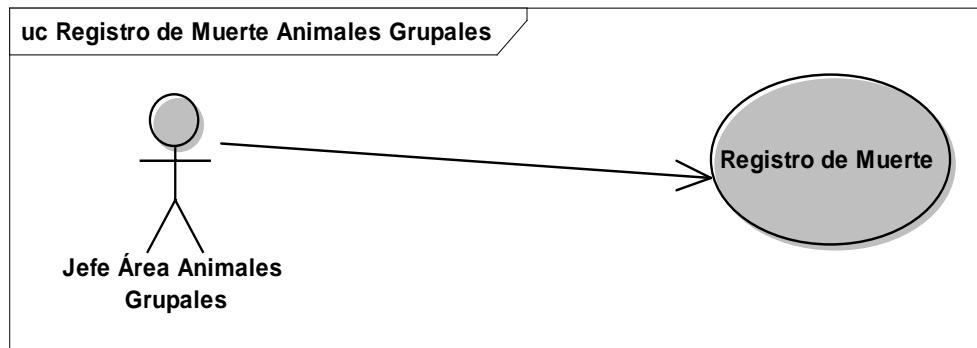
Nombre:	REGISTRO DE CONSUMO DE ALIMENTOS ANIMALES GRUPALES
Sistema:	
Descripción:	Registrar en el sistema la información del registro del consumo de alimento Animales Grupales.
Actores:	Jefe de Área Animales Grupales.
Precondiciones:	El encargado deberá haber ingresado todos los datos en el sistema: El sistema desplegará un formulario en el cual el usuario deberá ingresar los siguientes datos.
Flujo Normal:	<ol style="list-style-type: none">1. El usuario selecciona la clase de animal.2. El sistema devuelve la clase seleccionada.3. El usuario ingresa la identificación de la categoría del animal y elige buscar.4. El sistema muestra la categoría correspondiente a la identificación.5. El usuario ingresa fecha.6. El sistema devuelve la cantidad de alimento del animal.7. El usuario ingresa los días.8. El sistema calcula total de alimento.9. El usuario selecciona guardar.10. El sistema guarda la información.
Flujo Alternativo:	<ul style="list-style-type: none">▪ El Jefe de área ingresa datos en formulario del sistema.▪ El Jefe de área guarda los últimos datos ingresados una vez que haya terminado.
Pos condiciones:	El mensaje ha sido almacenado en el sistema.

Tabla 24



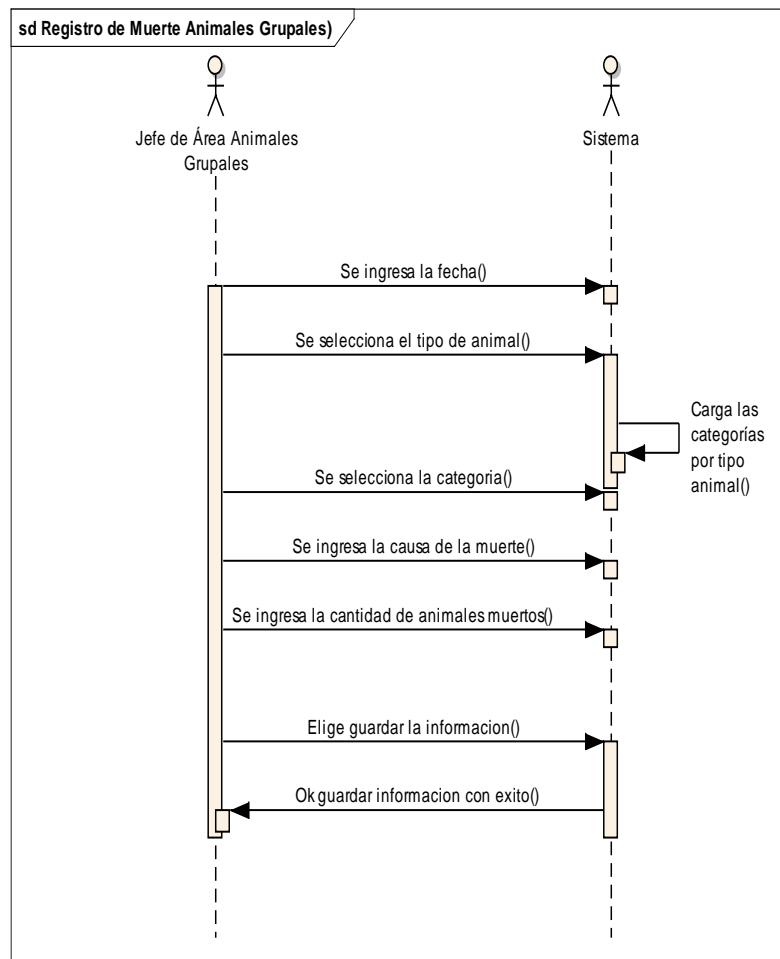
3.6.4. CASO DE USO REGISTRO DE MUERTE DE ANIMALES

GRUPALES



3.6.4.1 DIAGRAMA DE SECUENCIA DEL REGISTRO DE MUERTE

ANIMALES GRUPALES





UNIVERSIDAD CENTRAL DEL ECUADOR
Sistema de Registro de Animales y sus Derivados del Centro Experimental Uyumbicho

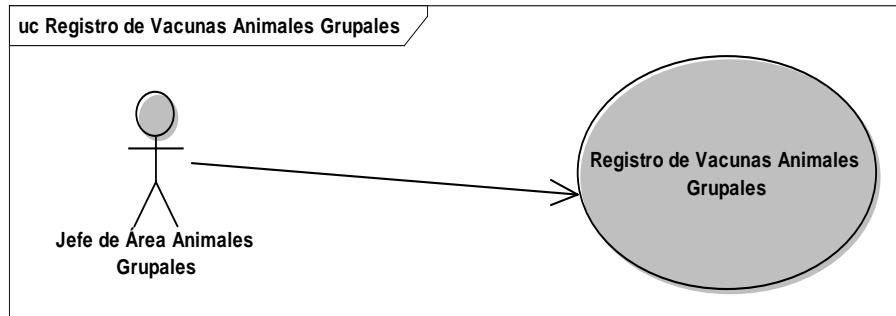
Nombre:	REGISTRO MUERTE ANIMALES GRUPALES
Sistema:	
Descripción:	Registrar en el sistema la información del registro de muerte Animales Grupales
Actores:	Jefe de Área Animales Grupales
Precondiciones:	<p>El encargado deberá haber ingresado todos los datos en el sistema:</p> <p>El sistema desplegará un formulario en el cual el usuario deberá ingresar los siguientes datos.</p>
Flujo Normal:	<ol style="list-style-type: none">1. El usuario selecciona la clase de animal.2. El sistema devuelve la clase seleccionada.3. El usuario ingresa la identificación de la categoría del animal y elige buscar.4. El sistema busca y devuelve la categoría correspondiente a la identificación.5. El usuario ingresa la fecha del registro de muerte.6. El usuario ingresa la cantidad de animales muertos.7. El usuario selecciona guarda la información.8. El sistema guarda la información.
Flujo Alternativo:	<ul style="list-style-type: none">▪ El Jefe de área ingresa datos en formulario del sistema.▪ El Jefe de área guarda los últimos datos ingresados una vez que haya terminado.
Pos condiciones:	El mensaje ha sido almacenado en el sistema.

Tabla 25



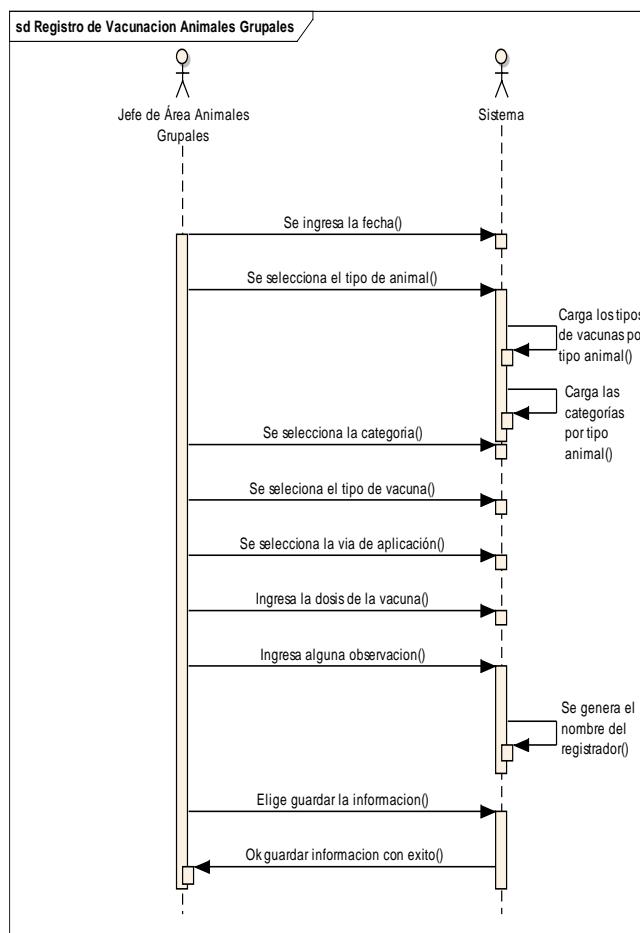
3.6.5. CASO DE USO REGISTRO DE VACUNAS ANIMALES

GRUPALES



3.6.5.1. DIAGRAMA DE SECUENCIA DEL REGISTRO DE VACUNAS

ANIMALES GRUPALES





Nombre:	REGISTRO DE VACUNAS ANIMALES GRUPALES
Sistema:	
Descripción: Registrar en el sistema la información del registro de vacunas Animales Grupales.	
Actores: Jefe de Área Animales Grupales.	
Precondiciones: El encargado deberá haber ingresado todos los datos en el sistema: El sistema desplegará un formulario en el cual el usuario deberá ingresar los siguientes datos.	
Flujo Normal: 1. El usuario selecciona la clase de animal. 2. El sistema devuelve la clase seleccionada. 3. El usuario ingresa la identificación del animal o categoría y elige buscar. 4. El sistema busca y devuelve el animal o categoría correspondiente a la identificación. 5. El usuario ingresa la fecha de colocación de la vacuna. 6. El usuario ingresa el nombre de la vacuna. 7. El usuario selecciona la vía de aplicación de la vacuna. 8. El usuario ingresa la dosis de la vacuna. 9. El usuario ingresa la fecha de revacunación. 10. El usuario selecciona guarda la información. 11. El sistema guarda la información.	
Flujo Alternativo: ▪ El Jefe de área ingresa datos en formulario del sistema. ▪ El Jefe de área guarda los últimos datos ingresados una vez que haya terminado.	
Pos condiciones: El mensaje ha sido almacenado en el sistema.	

Tabla 26



CAPÍTULO IV

4. ESPECIFICACIONES FUNCIONALES

4.1 MODELO DE LA SOLUCIÓN DE LA SOLUCIÓN DE LA PROPUESTA (Arquitectura del ProyectoCEU¹, Sistema de Registro de animales y sus Derivados del Centro Experimental Uyumbicho).

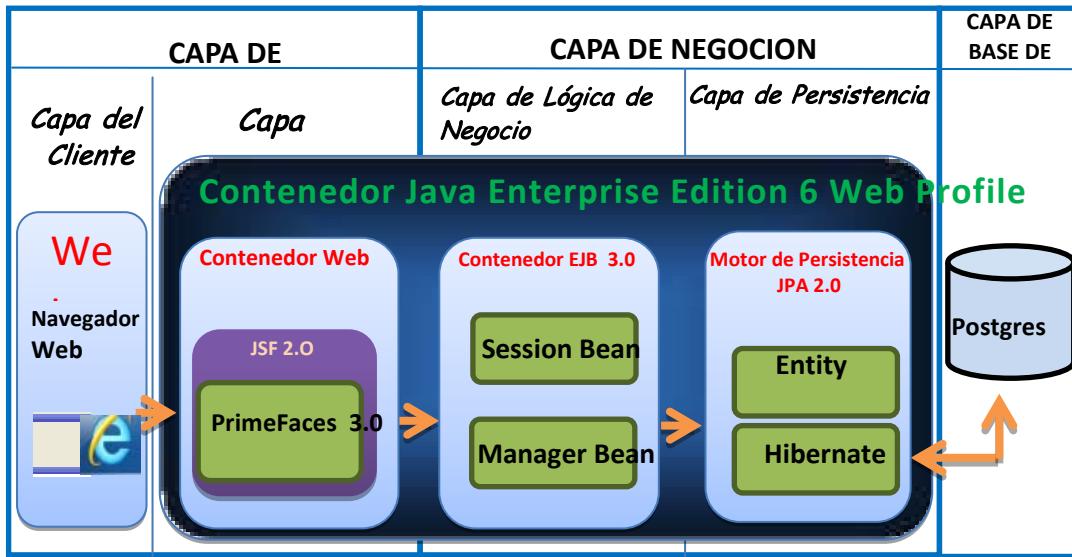
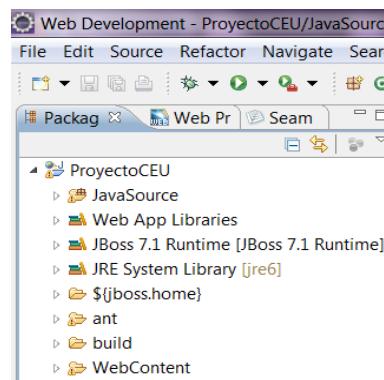


Grafico 3 Arquitectura JEE6 del ProyectoCEU

Visualización del Proyecto CEU.



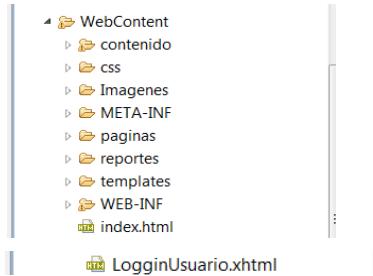
Figura_1. Visualización del proyectoCEU

¹ProyectoCEU: Nombre del proyecto del Sistema.



4.1.1. Capa del Cliente: Se ingresa a cualquier navegador web, en la barra de navegación se coloca la URL del proyecto.

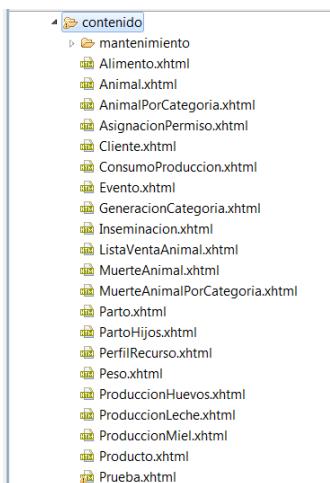
4.1.2. Capa Web: En el proyecto las páginas JSF se encuentran en la carpeta WebContent distribuido como de la siguiente manera:



En esta raíz se encuentra la página index.html que redirecciona a la página LogginUsuario.xhtml.

- Contenido.
- CSS e Imágenes.
- Páginas.
- Reportes.
- Template.

4.1.2.1. Contenido: Se encuentran las páginas del sistema tanto las de mantenimiento de la información, como las de funcionamiento del Centro Experimental Uyumbicho.

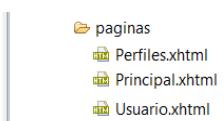




4.1.2.2. CSS e Imágenes: Estas carpetas se utilizan para dar estilo y almacenar imágenes que serán llamadas desde las páginas.



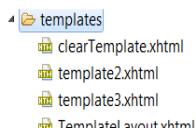
4.1.2.3. Páginas: En estas se encuentran la parte de administración de usuarios.



4.1.2.4. Reportes: Se ubican las páginas de reportes que permite ingresar los parámetros para la consulta.



4.1.2.5. Template: Se encuentran las plantillas que son páginas que se utilizan como base para las demás páginas del sistema.



4.1.3. Capa de Lógica de Negocio:

En esta capa se encuentran los EJB o Enterprise Java Beans son componentes JEE que se ejecutan dentro de un container EJB, un entorno de ejecución dentro de un Application Server. El contenedor de EJB provee servicios al usuario, los cuales expone de manera transparente, como: Transacciones, Seguridad, etc.

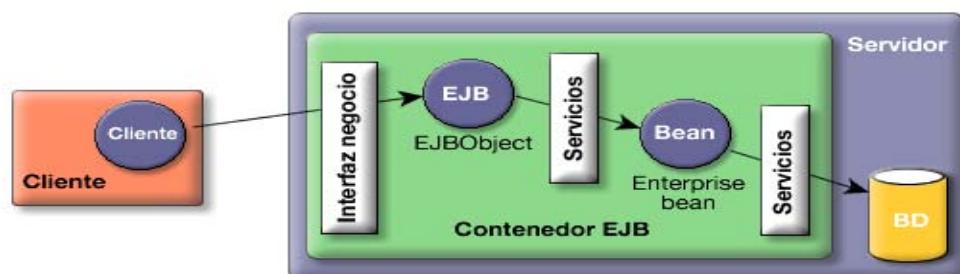
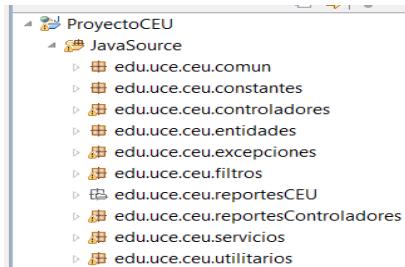


Gráfico 4: ENTERPRISE JAVA BEANS



En el proyectoCEU dentro de la carpeta JavaSource se encuentran todas las clases java utilizadas.

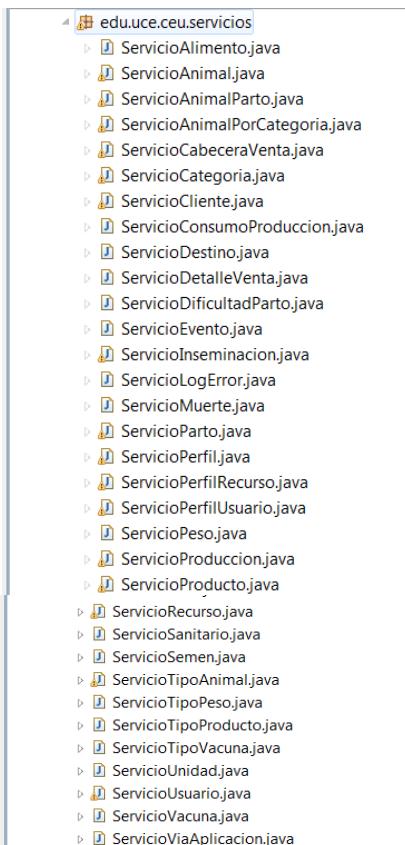


En esta capa tiene dos partes:

- Session Beans.
- Manager Beans.

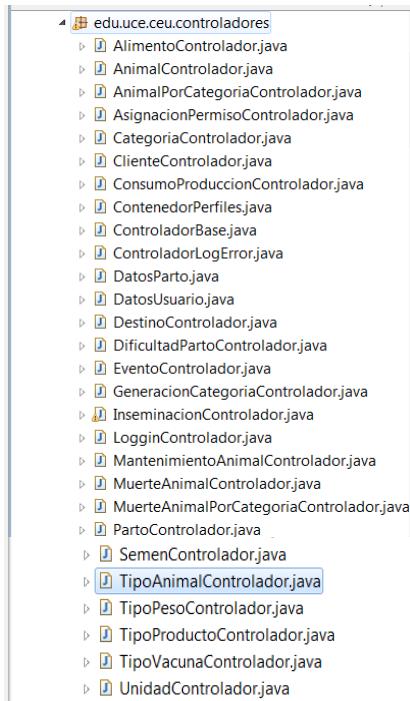
4.1.3.1. Session Beans: Son las clases que tienen los EJBs la lógica del negocio.

Estas clases se encuentran en el paquete edu.uce.ceu.servicios.



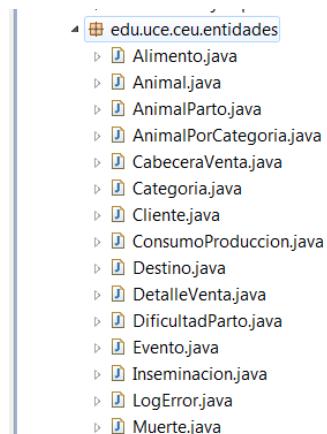


4.1.3.2. Manager Beans: Estas clases que permiten interactuar con las páginas xhtml se encuentran en el paquete edu.uce.ceu.controladores.



4.1.4. Capa de Persistencia:

En esta capa se encuentran las clases java que permiten interactuar con la base de datos, estas clases son las mapeadas de la base de datos, se encuentran en el paquete edu.uce.ceu.entidades.





```
>  Parto.java  
>  Perfil.java  
>  PerfilRecurso.java  
>  PerfilUsuario.java  
>  Peso.java  
>  Produccion.java  
>  Producto.java  
>  Raza.java  
>  Recurso.java  
>  Sanitario.java  
>  Semen.java  
>  TipoAnimal.java  
>  TipoPeso.java  
>  TipoProducto.java  
>  TipoVacuna.java  
>  Unidad.java  
>  Usuario.java  
>  Vacuna.java  
>  ViaAplicacion.java
```

4.1.5. Capa de Base de Datos: Esta capa corresponde a la base de datos PostgeSQL 8.4 a continuación tenemos los diagramas correspondientes.



4.2. Modelo Conceptual de la base de datos del Proyecto CEU (Sistema de Registro de animales y sus Derivados del Centro Experimental Uyumbicho).

MODELO CONCEPTUAL



4.3. Modelo Físico de la base de datos del Proyecto CEU (Sistema de registro de animales y sus Derivados del Centro Experimental Uyumbicho).

MODELO FÍSCO



**4.4 Modelo de Objetos y Clases de la base de datos del Proyecto CEU
(Sistema de registro de animales y sus Derivados del Centro Experimental
Uyumbicho).**

MODELO DE OBJETOS Y CLASES



CAPÍTULO V

CONCLUSIONES Y RECOMENDACIONES

5.1. CONCLUSIONES

Al haber culminado nuestro trabajo de Graduación previo a la obtención del título de ingeniero Informático “SISTEMA DE REGISTRO DE ANIMALES Y SUS DERIVADOS DEL CENTRO EXPERIMENTAL UYUMBICHO DE LA FACULTAD DE VETERINARIA DE LA UNIVERSIDAD CENTRAL DEL ECUADOR”, se ha logrado establecer las siguientes conclusiones:

El análisis de la ingeniería de software permite tener una visión real de la lógica de los procesos del Centro Experimental Uyumbicho, permitiendo la construcción del software que satisfaga todos los requerimientos reales del Centro.

Al aplicar la metodología espiral en el desarrollo del software que fue utilizado en nuestro sistema, ayuda a determinar los posibles riesgos en su construcción, ya que a pesar de tener diagramado los procesos se presenta escenarios que no fueron considerados en la fase del análisis, permitiendo minimizar el impacto de los riesgos.

El sistema permite llevar un registro de los animales y sus derivados vía Web del Centro Experimental de Uyumbicho de la Facultad de Veterinaria de la Universidad Central del Ecuador.

Con la utilización del sistema dará, seguimiento a los animales desde que nacen hasta que sea faenado o vendido, con indicadores en cada etapa de vida.

Con el software tendrá un control de la producción actual, de animales que se encuentran en el CEU para su comercialización.

El sistema genera reportes, consultas e información que ayuda a la toma de decisiones en el Centro Experimental Uyumbicho



5.2 RECOMENDACIONES

En los posteriores desarrollos de tesis se debe realizar un levantamiento de todos los procesos y una óptima Ingeniería de Software para que cuando se construya el Sistema no se tenga problemas en su funcionamiento y el usuario final tenga los mejores beneficios.

A la Facultad de Veterinaria de la Universidad Central del Ecuador se sugiere implementar el Sistema CEU para la agilitación de los procesos y así tener la mejor utilidad.

Es fundamental que la Facultad de Medicina Veterinaria de la Universidad Central del Ecuador suministre al Centro Experimental Uyumbicho, una actualización de los equipos informáticos y nuevas infraestructura para su mejor funcionamiento.

Es necesario fomentar el uso del sistema, para de esta manera poder gozar de los beneficios que este nos presta, e ir mejorando día a día en base a las sugerencias de los usuarios.



BIBLIOGRAFÍA

MATERIALES DE REFERENCIA

1. DEITEL Harvey M, DEITEL Paúl J. Como programar en Java. 5ta Edición. Pearson 2004.840
2. STEVEN W. Tips Para Programar Java. 1era Ed Mexico.McGraw-Hill.1997.1001

ENLACES WEB

- www.metodologia-unmsm.com/clases/8/index.htm
- [www.unico.edu.sv/investiga/proyinvestigacion.htm.](http://www.unico.edu.sv/investiga/proyinvestigacion.htm)
- <http://ccia.ei.uvigo.es/docencia/SCS/1011/transparencias/Tema5-3.JSF.pdf>
- http://en.wikipedia.org/wiki/JBoss_application_server
- <http://sistemas.uniandes.edu.co/~isis3702/dokuwiki/lib/exe/fetch.php?media=principal:isis3702-ejb3.pdf>
- www.slideshare.net/kika19/sistemas-manejadores-de-base-de-datos-5071421
- www.apmarin.com/download/480_dpt1.pdf
- mistock.lcompras.biz/tallersoftware/1249-ireport
- www.softpedia.com/es/programa-R-for-Windows-136112.html



ANEXOS



ANEXO 1.

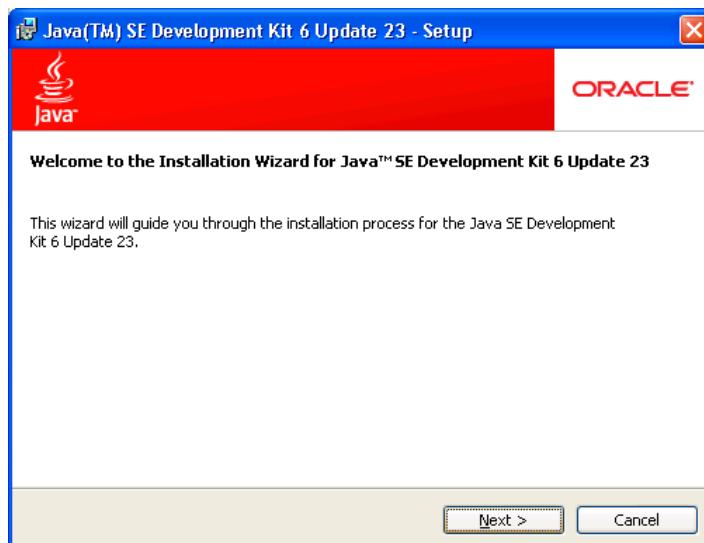
INSTALACIÓN PROYECTOCEU

PRE- REQUISITOS.

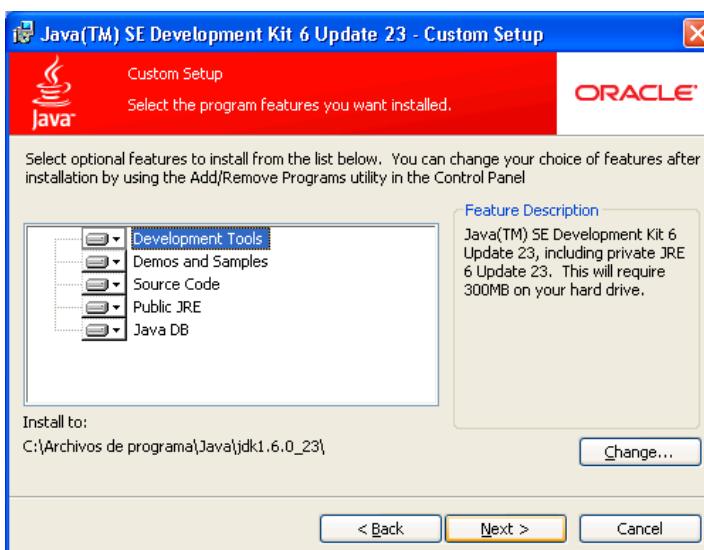
1. INSTALACION JDK jdk-6u23-windows-i586.

Ejecutar el instalador dando doble click en el icono.

Aparece la siguiente pantalla y en el resto de pantallas, se selecciona **Next** para la instalación.

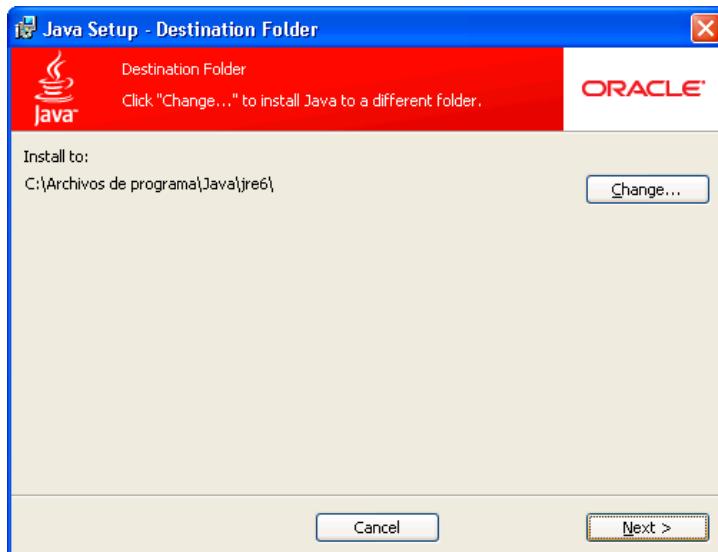


Se despliega los componentes que se van ha instalar.

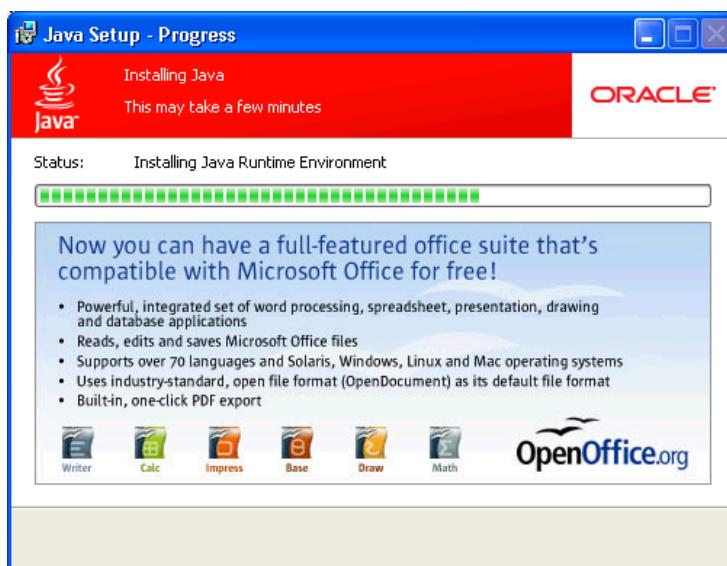




En la siguiente ventana aparece el lugar donde será instalado.



Aparecerá la siguiente ventana que muestra el progreso de la instalación.



Finalmente se mostrará la ventana que indica que el **jdk-6u23-windows-i586** a sido instalado correctamente y por último se da un clic en **Finish**.

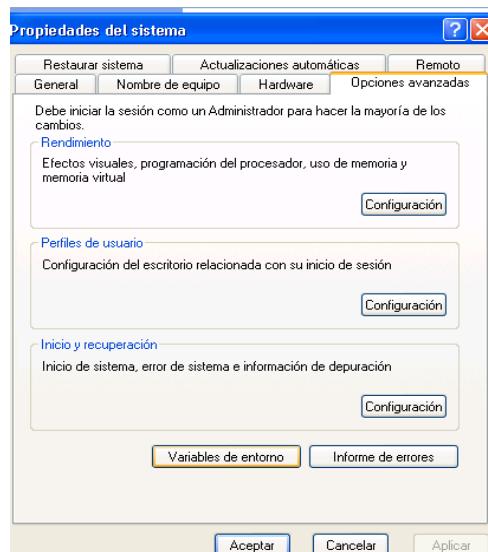


1. CREACIÓN DE LA VARIABLE JAVA_HOME

Primero se da clic derecho en MI PC, Propiedades.

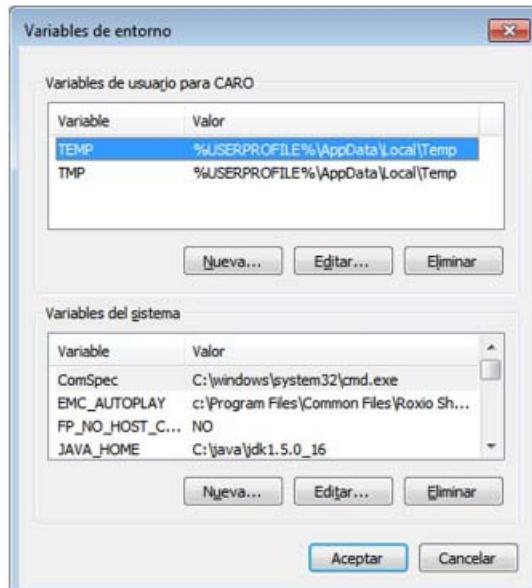


Se selecciona **Opciones Avanzadas**, como se muestra en el siguiente esquema:

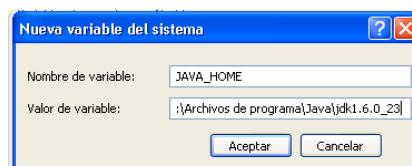




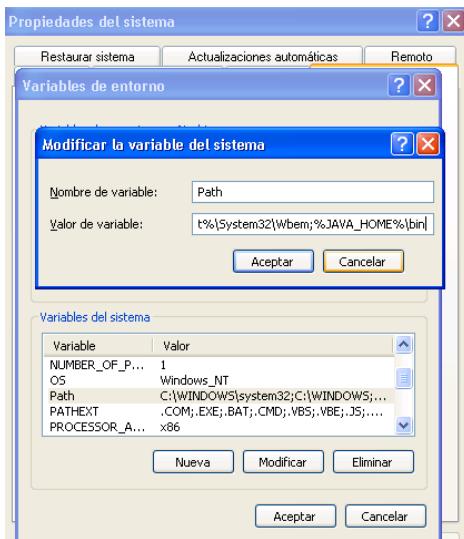
A continuación se da un clic en Variables de Entorno, con lo cual se despliega la siguiente ventana en la que se da un clic en Nueva.



En la siguiente ventana que se despliega se llenan los campos de edición; en **Nombre de variable** se escribirá JAVA_HOME, en **Valor De La Variable** ellugar donde se instaló previamente el jdk, C:\Archivos de programa\Java\jdk1.6.0_23; finalmente seda un clic en Aceptar. Y ya está creada la variable.



La variable JAVA_HOME, se adiciona a la variable Path del sistema, se coloca al final del valor de ésta, a continuación el valor de la variable creada %JAVA_HOME%/bin.

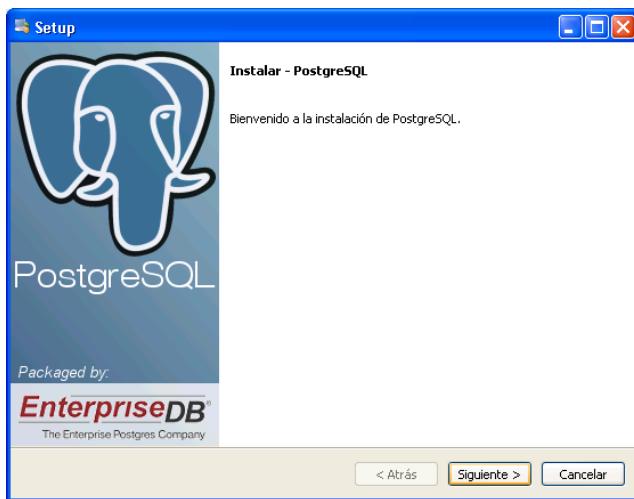


3. INSTALACIÓN DE LA BASE DE DATOS POSTGRESQL 8.4.4.1.

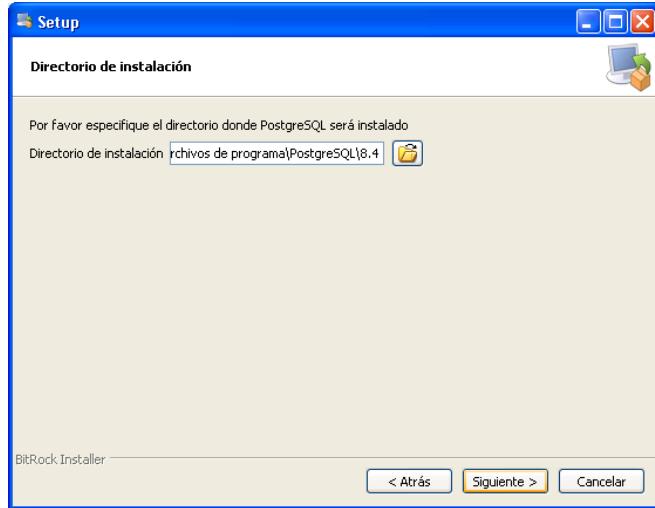
Ejecutar la instalación dando doble click en el icono y seleccionar el botón siguiente.



La primera es una ventana de bienvenida al instalador, se da un clic en siguiente.



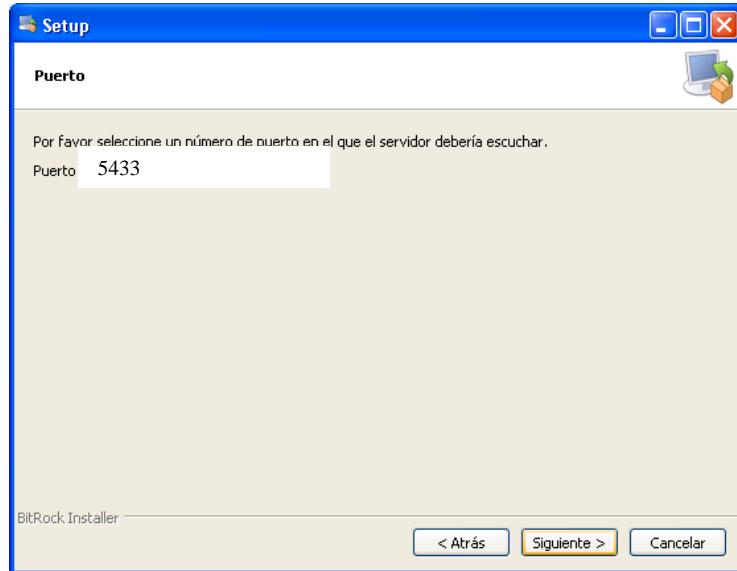
Aquí será donde se va ha instalar PostgreSQL.



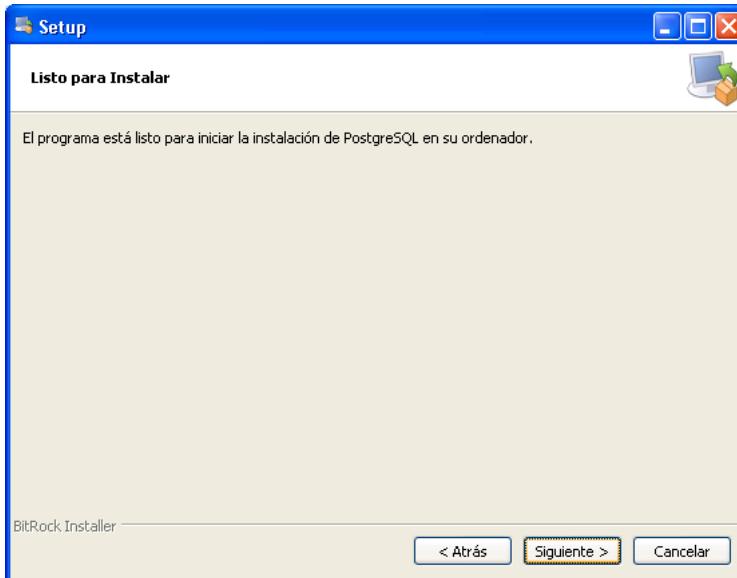
Luego el instalador pedirá una contraseña clave para el super-usuario (**postgres**).



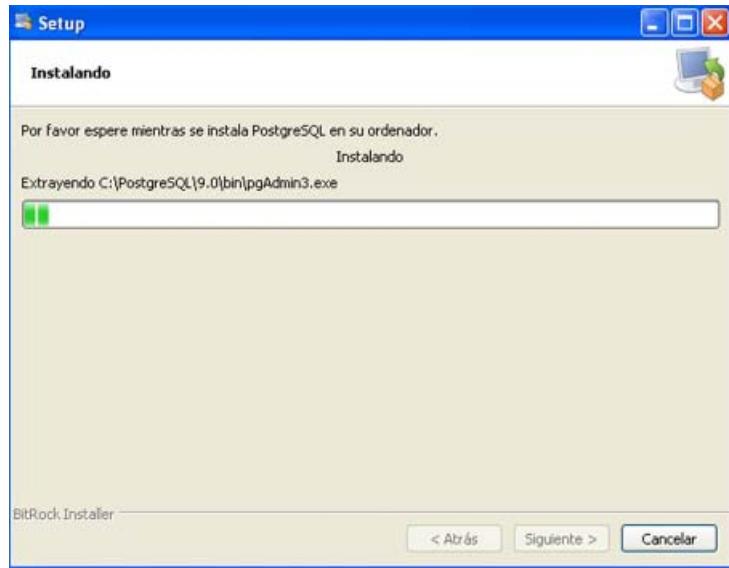
Luego pide el número de puerto, se ingresa el puerto en que el servidor podrá ejecutarse.



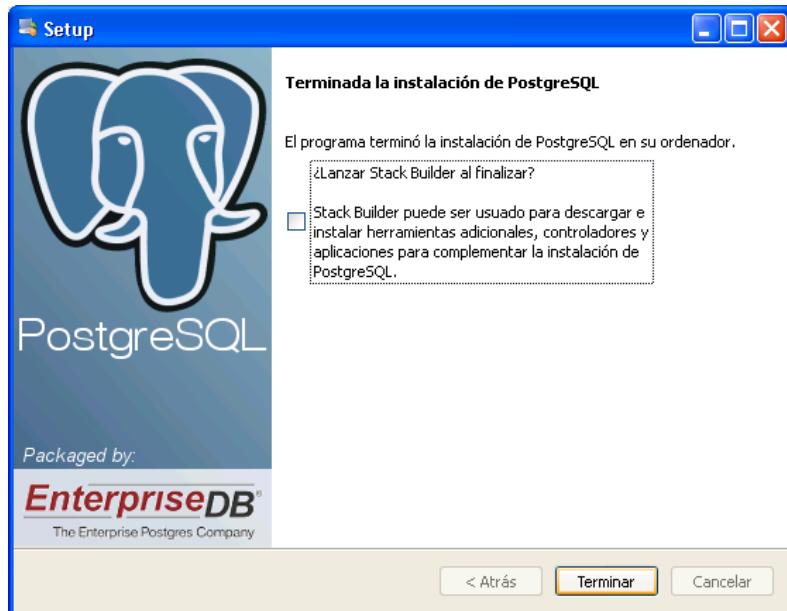
Aparecerá una ventana que informa que PostgreSQL está listo para ser instalado, dar clic en siguiente.



La siguiente ventana muestra el progreso de la instalación.

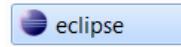


Finalmente se da un clic en Terminar y ya esta instalada la herramienta.



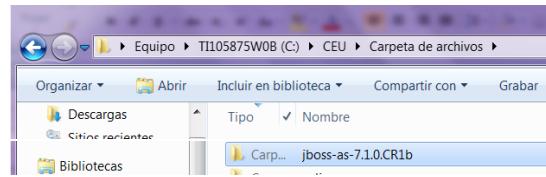
4. INSTALACION DEL ECLIPSE VERSION INDIGO

Se descarga del internet y se descomprime en cualquier lugar que deseé como por ejemplo C:\CEU\eclipse.



5. INSTALACIÓN DEL SERVIDOR WEB JBOSS 7.

Se descarga del internet y se descomprime en cualquier lugar que deseé como por ejemplo C:\CEU\.

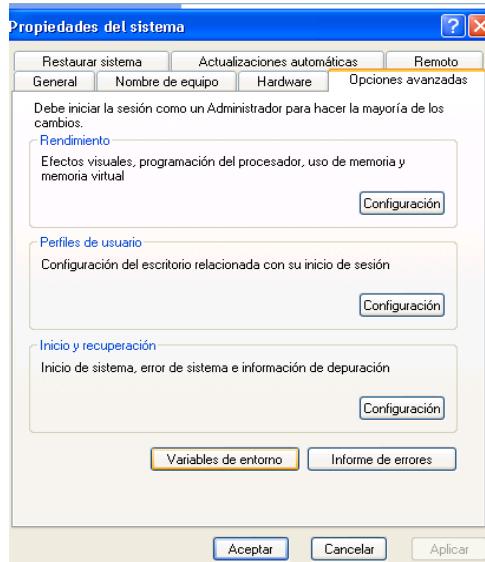


6. CREACIÓN DE LA VARIABLE JBOSS_HOME

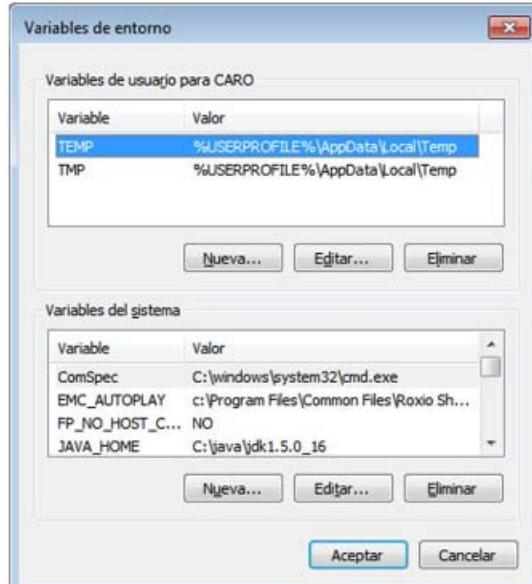
Primero se da clic derecho en MI PC, Propiedades.



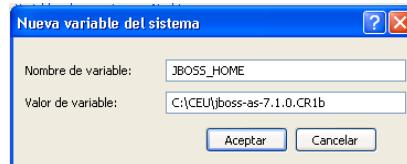
Se selecciona **Opciones Avanzadas**, como se muestra en el siguiente esquema:



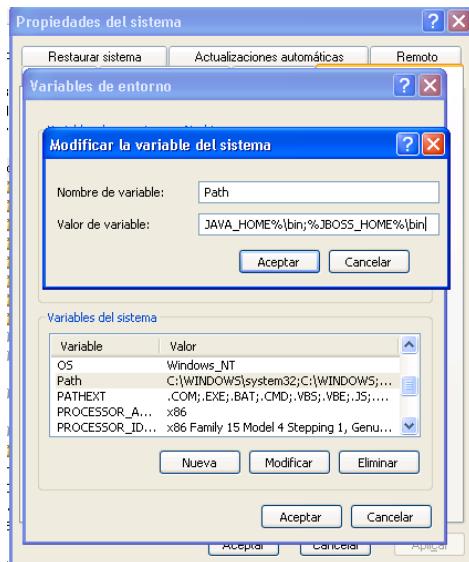
A continuación se da un clic en Variables de Entorno, con lo cual se despliega la siguiente ventana en la que se da un clic en Nueva.



En la siguiente ventana que se despliega se llenan los campos de edición; en **Nombre de variable** se escribirá JBOSS_HOME, en **Valor De La Variable** el lugar donde se copio, el servidor de aplicaciones JBoss, C:\CEU\jboss-as-7.1.0.CR1b; finalmente se da un clic en Aceptar. Y ya está creada la variable.



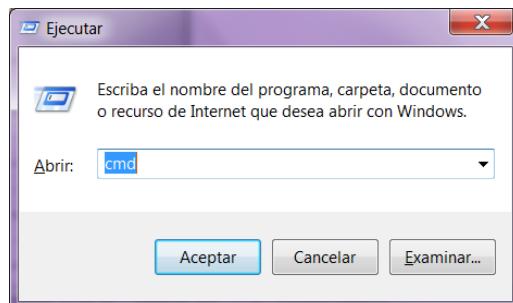
La variable JBOSS_HOME, se adiciona a la variable Path del sistema, se coloca al final del valor de ésta, a continuación el valor de la variable creada %JBOSS_HOME%/bin.



Pegar el jar de conexión a la base de datos postgresql de acuerdo a la versión instalada (ejemplo:postgresql-8.4-702.jdbc4.jar) en el servidor web en la dirección <http://.....\jboss-as-7.1.0.CR1b\standalone\deployments>

7. LEVANTAR EL SERVIDOR WEB JBOSS 7.

Se ingresa a la consola de comandos,





Se ejecuta el comando standalone.bat para levantar el servicio desde consola.

```
C:\Windows\system32\cmd.exe - standalone.bat
Microsoft Windows [Versión 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. Reservados todos los derechos.

C:\Users\Narcisa>standalone.bat
Calling "C:\CEU\jboss-as-7.1.0.CR1b\bin\standalone.conf.bat"
=====
JBoss Bootstrap Environment
JBOSS_HOME: C:\CEU\jboss-as-7.1.0.CR1b
JAVA: C:\Program Files (x86)\Java\jre6\bin\java
JAVA_OPTS: -Dprogram.name=standalone.bat -Xms64M -Xmx512M -XX:MaxPermSize=256M
-Dsun.rmi.dgc.client.gcInterval=3600000 -Dsun.rmi.dgc.server.gcInterval=3600000
-Dorg.jboss.resolver.warning=true -Djboss.modules.system.pkgs=org.jboss.byteman
=====

C:\Windows\system32\cmd.exe - standalone.bat
SC service thread 1-1> HHH000397: Using ASTQueryTranslatorFactory
18:49:42,500 INFO [org.hibernate.validator.util.Version] <MSC service thread 1-1> Hibernate Validator 4.2.0.Final
18:49:43,218 INFO [javax.enterprise.resource.webcontainer.jsf.config] <MSC service thread 1-1> Inicializando Mojarra 2.1.5 <SNAPSHOT 20111202> para el contexto '/ProyectoCEU'
18:49:45,890 INFO [org.primefaces.webapp.PostConstructApplicationEventListener] <MSC service thread 1-1> Running on PrimeFaces 3.0
18:49:45,890 INFO [org.primefaces.webapp.PostConstructApplicationEventListener] <MSC service thread 1-1> Running on PrimeFaces 3.0
18:49:45,921 INFO [org.jboss.web] <MSC service thread 1-1> registering web context: '/ProyectoCEU'
18:49:45,921 INFO [org.jboss.msc.service] <MSC service thread 1-1> JBoss AS 7.1.0.CR1b "Flux Capacitor" started in 1732ms! Started 942 of 1018 services (72 services are passive or on demand)
18:49:45,968 INFO [org.jboss.as.server] <DeploymentScanner-threads - 2> JBAS018559: Deployed "ProyectoCEU.war"
18:49:45,968 INFO [org.jboss.as.server] <DeploymentScanner-threads - 2> JBAS018559: Deployed "postgresql-8.4-702.jdbc4.jar"
18:49:45,968 INFO [org.jboss.as.controller] <DeploymentScanner-threads - 2> JBAS014774: Service status report
JBAS014776: Newly connected services:
    service jboss.jdbc-driver.postgresql-8_4-702.jdbc4.jar (no longer required)
>
```

8. CONFIGURACIÓN CONEXIÓN A LA BASES DE DATOS.

Acceso a la consola administrativa.

Ingresar en el browser:

<http://localhost:8080/>

Elegir la opción.

Administration Console

La primera vez que ingresemos, aparece una pantalla indicando que no tenemos ningún usuario agregado para ingresar a la consola.



Your JBoss Application Server 7 is running.
However you have not yet added any users to be able to access the admin console.
To add a new user execute the add-user.bat script within the bin folder of your AS 7 installation and enter the requested information.
By default the realm name used by AS 7 is "ManagementRealm" this is already selected by default.

```
on C:\Windows\system32\cmd.exe
Enter details of new user to add.
Realm <ManagementRealm> :
Username : myNewUser
Password :
Re-enter Password :
About to add user 'myNewUser' for realm 'ManagementRealm'
Is this correct yes/no? yes
Added user 'myNewUser' to file 'C:\Work\jboss-as\build\target\jboss-as-7.1.0.Alpha2-SNAPSHOT\standalone\configuration\mgmt-users.properties'
Added user 'myNewUser' to file 'C:\Work\jboss-as\build\target\jboss-as-7.1.0.Alpha2-SNAPSHOT\domain\configuration\mgmt-users.properties'
Press any key to continue . . .
```

After you have added the user follow this link to [Try Again](#).

En la línea de comando Ejecutar el comando add-user en el directorio \$JBOSS_HOME/bin.

```
C:\Windows\system32\cmd.exe
C:\JEE6\jboss-as-7.1.0.CR1b\bin>add-user

Enter the details of the new user to add.
Realm (ManagementRealm) :
Username : smosquera
Password :
Re-enter Password :
About to add user 'smosquera' for realm 'ManagementRealm'
Is this correct yes/no? yes
Added user 'smosquera' to file 'C:\JEE6\jboss-as-7.1.0.CR1b\standalone\configuration\mgmt-users.properties'
Added user 'smosquera' to file 'C:\JEE6\jboss-as-7.1.0.CR1b\domain\configuration\mgmt-users.properties'
Presione una tecla para continuar . . .

C:\JEE6\jboss-as-7.1.0.CR1b\bin>
```

Realm: dejar en blanco.

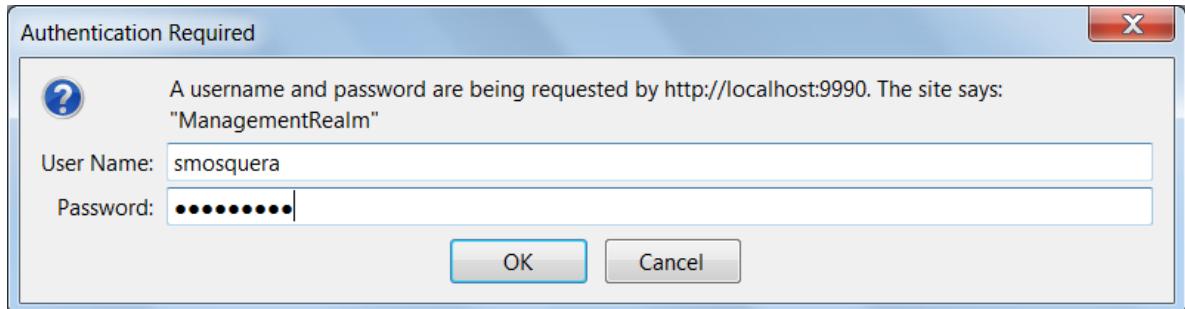
Username: ingresar cualquier nombre.

Password: ingresar un password.

Re-enter password: confirmar el password.



Regresar al browser y presionar la opción Try Again.



Ingresar el usuario y password creados.

CREACION DE UN DATASOURCE

1. Ingresar a la consola administrativa.
2. Elegir Profile.



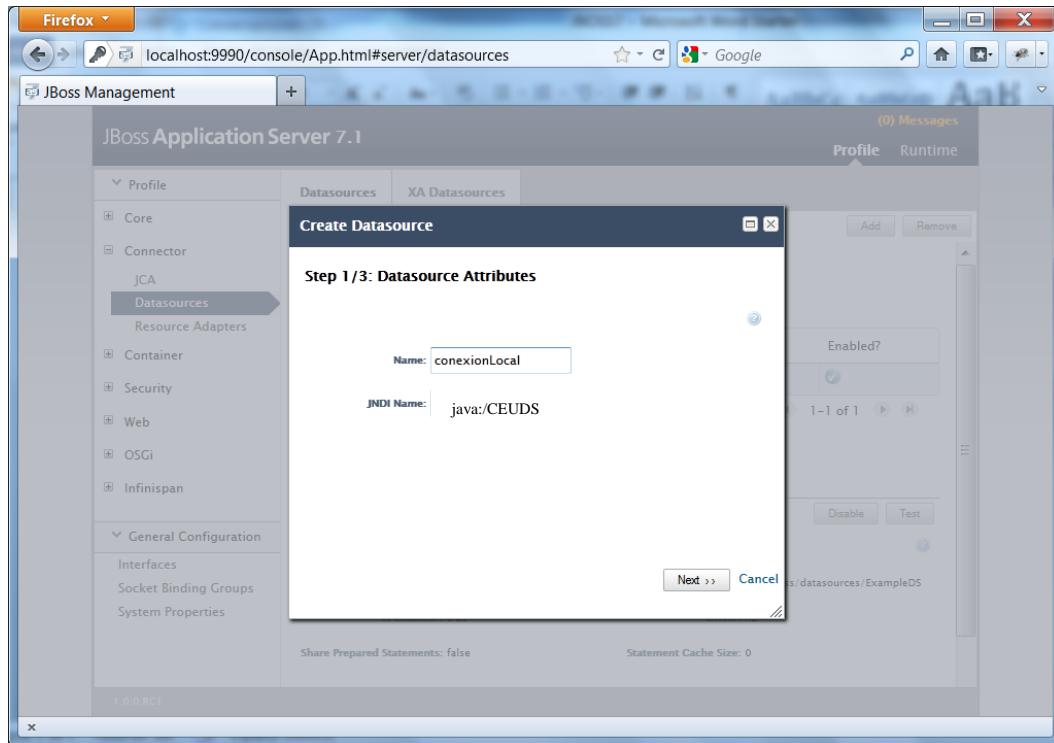
3. Elegir DataSource y el botón Add.



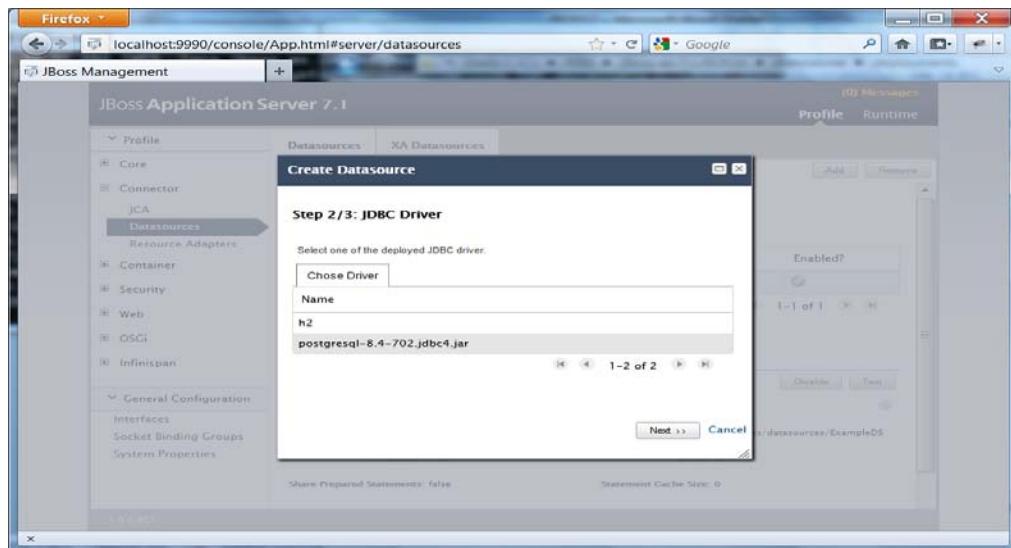
4. Ingresar los datos.

Name: cualquier nombre.

JNDI Name: El nombre con el cual se registra el recurso en el servidor, este nombre es el que debe colocarse en el archivo persistence.xml para que la aplicación use el datasource. java:/CEUDS.



5. Elegir el datasource de postgresql.



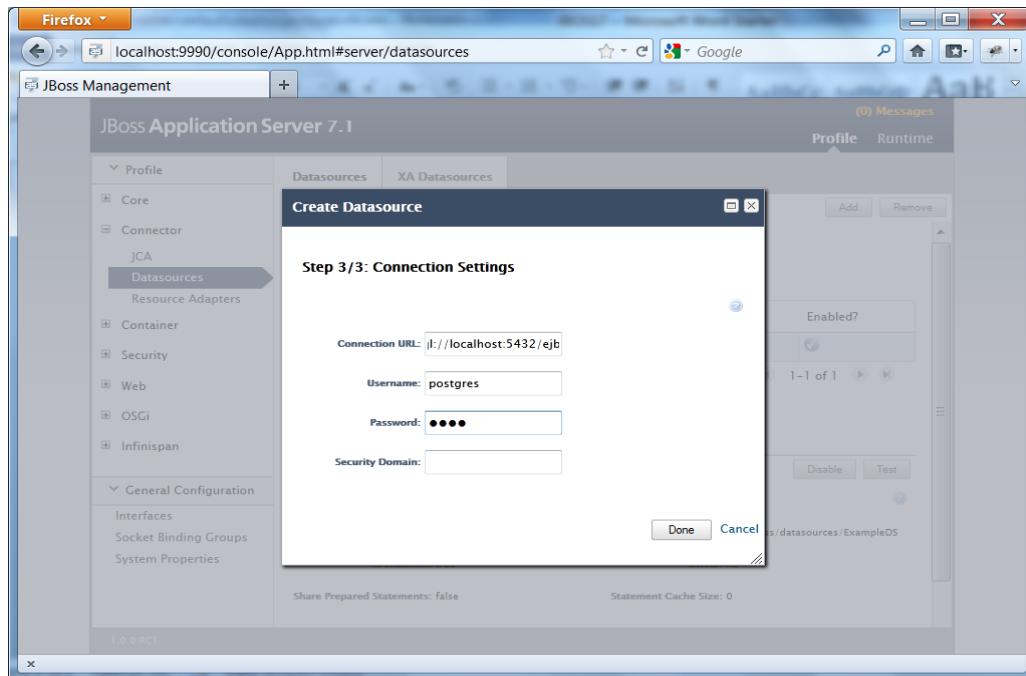
6. Colocar los datos de la conexión.

URL: jdbc:postgresql://localhost:5433/base_ceu

Username: usuario para conectarse a la base de datos (ejemplo:postgres).



Password: password del usuario que se conecta a la base de datos Security
Domain: Dejar en blanco.



Elegir Enable.





Elegir Test.

The screenshot shows the JBoss Application Server 7.1 management console. On the left, there's a navigation tree with sections like Profile, Core, Connector, JCA, Container, Security, Web, and OSGi. Under JCA, 'DataSources' is selected. The main panel shows two data sources: 'ExampleDS' and 'conexionLocal'. Below the table is a 'Datasource' tab with tabs for Attributes, Connection, Security, Properties, Pool, and Validation. The 'Connection' tab is active. In the bottom right corner of the table area, there is a 'Test' button, which is highlighted with a red box.

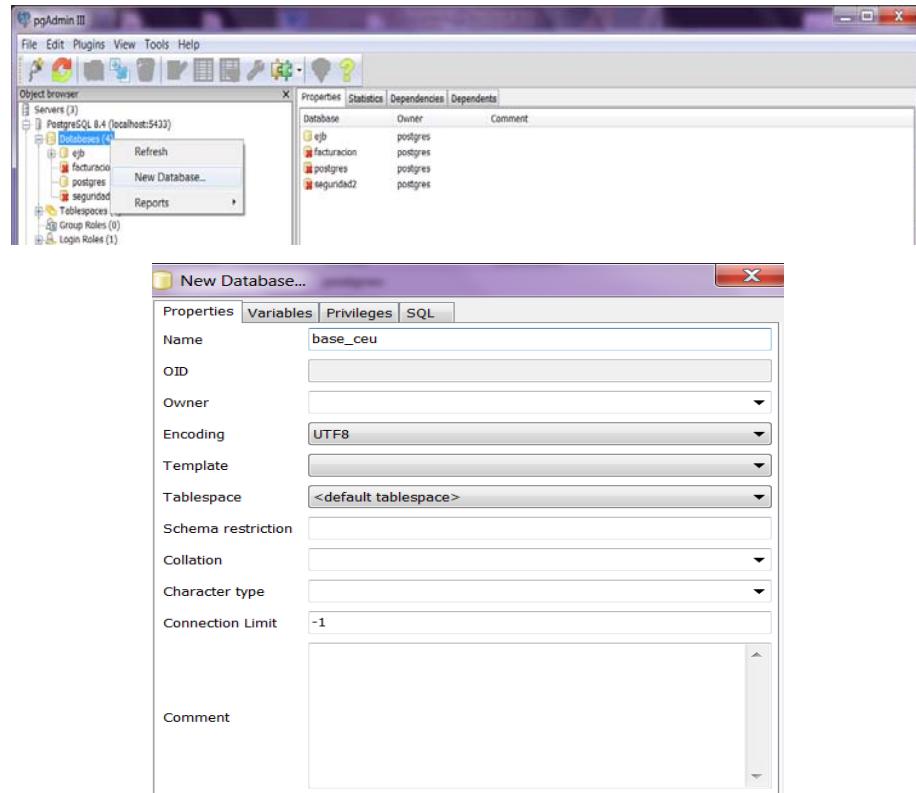
Si la conexión es exitosa, se mostrará el mensaje.

This screenshot shows a modal dialog box titled 'Datasource Connection' with the message: 'Successfully created JDBC connection. Successfully connected to database ExampleDS.' There is an 'OK' button at the bottom. The background shows the JBoss Application Server 7.1 management console with the same DataSources configuration as the previous screenshot.

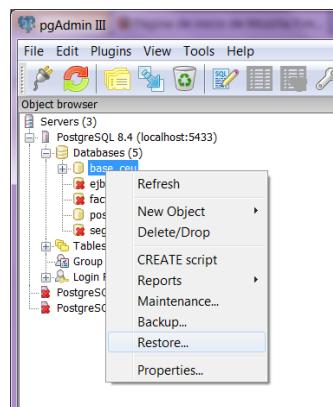
REQUISITOS

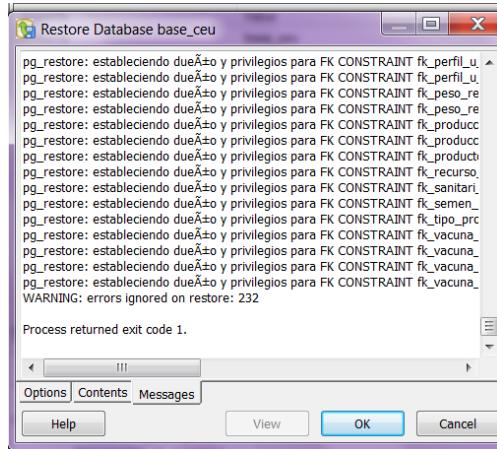
1. Asegúrese que se encuentre levantado el servidor web Jboss.
2. Copiar el archivo .war en la carpeta de deployments del servidor web.
<\\\\....\\jboss-as-7.1.0.CR1b\\standalone\\deployments>
(ejemploC:\\CEU\\jboss-as-7.1.0.CR1b\\standalone\\deployments).
3. Restaurar el base_ceu.backup de la base:

Se ingresa a la base de datos postgresql, se crea una nueva base de datos.



En la nueva base se restaura la base.





4. Verifica ejecución del proyectoCEU, ingresando a cualquier navegador web la URL del proyecto: localhost:8080/ProyectoCEU y observar la pagina de Loggin.





MANUAL DE USUARIO.

Para poder ingresar al sistema debemos abrir cualquier navegador como



. Ingresamos la dirección web del proyecto:
localhost:8080/ProyectoCEU



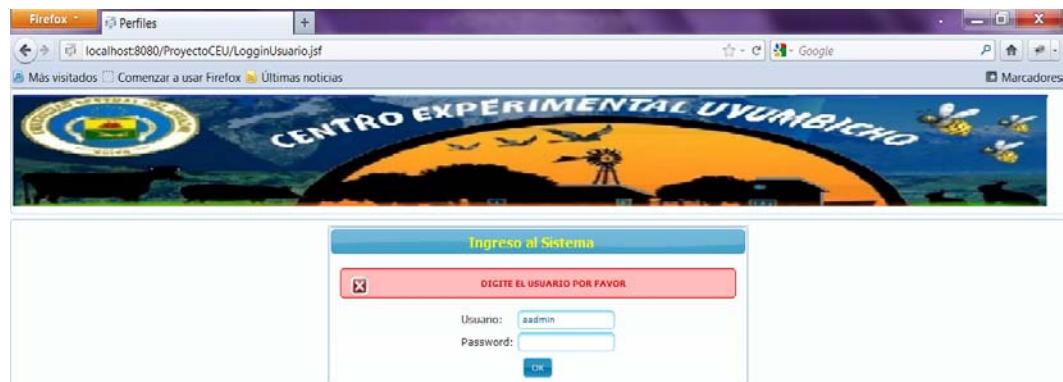
INGRESO AL SISTEMA.

Al ingresar a la dirección URL se nos despliega el sistema.

Para poder ingresar al sistema se loguea con el Usuario y Password que se nos fue otorgado.



Si ingresamos los datos incorrectos nos presenta un mensaje de error de “Usuario o contraseña inválidos”



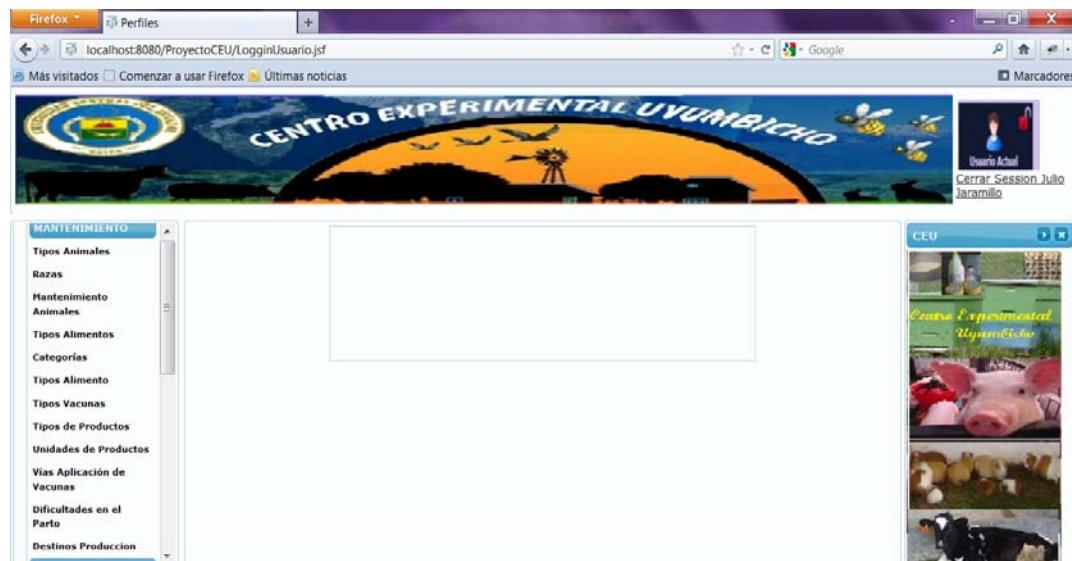


El sistema tiene una pantalla inicial en la cual el usuario puede ingresar según su perfil entre los cuales se tiene:

- Administrador del Centro Experimental Uyumbicho.
- Jefe del Área de Bovinos y Porcinos (Animales Individuales).
- Jefe del Área de Avícola y Cuyes (Animales Grupales).

ADMINISTRADOR DEL CENTRO EXPERIMENTAL UYUMBICHO.

Loguearse como administrador nos despliega en el lado izquierdo de la pantalla, el menú que tiene acceso este perfil.



En la parte superior derecha nos indica el usuario con el que se a logueado para ingresar al sistema



Al loguearse con este perfil tiene acceso al siguiente menú:

- Mantenimiento.
- Producción.
- Facturación.
- Consumo.
- Reportes.
- Administración.



Mantenimiento: Dentro de este menú se tiene las pantallas que permite guardar, actualizar, seleccionar, eliminar los diferentes datos y son:



Tipos Animales: En la parte superior se ingresa los datos de un nuevo tipo de animal se escoge la fecha, el nombre del tipo de animal y se selecciona la condición de animales individuales o grupales, en la tabla se muestra los datos ingresados, el botón Seleccionar muestra los datos de esa fila en la parte superior, podemos modificar la información al dar click en el botón actualizar se guarda ésta y el botón eliminar borra la información de esa fila.

Categorías: En la parte superior ingresa la fecha, el nombre de la categoría, si son animales individuales se ingresa el intervalo de tiempo, en la tabla se muestra



los datos ingresados, el botón Seleccionar muestra los datos de esa fila en la parte superior y podemos modificar la información al dar click en el botón actualizar se guarda ésta y el botón eliminar borra la información de esa fila.

The screenshot shows a Firefox browser window with the title 'Registro de Categorías'. The URL in the address bar is 'localhost:8080/ProyectoCEU/contenido/mantenimiento/Categoría.jsf'. The page header features the logo of the 'CENTRO EXPERIMENTAL UYUMBICHO' and a user session icon for 'Gustavo Alvaréz'.

ADMINISTRACION

- Perfiles
- Usuarios
- Asignación de Permisos
- Mantenimiento de Permisos
- MANTENIMIENTO**
- Hanternimiento Animales
- Destinos Producción
- Dificultades en el Parto
- Vías Aplicación de Vacunas
- Unidades de Productos
- Tipos de Productos

REGISTRO DE CATEGORÍAS

Fecha: 01/03/2012
Seleccione el Tipo del Animal: Bovinos
Nombre de la Categoría: Baquillas Medianas
Tiempo Inicial Categoría: 7
Tiempo Final Categoría: 12
En Meses Días
Registrador por: Gustavo Alvaréz Jacome
Guarda Actualizar

TABLA DE CATEGORIAS

FECHA	Tipo Animal	Nombre Categorías	ACCIONES
2012-03-01	Bovinos	Tereras	<input type="button" value="Editar"/> <input type="button" value="Eliminar"/>
2012-03-01	Bovinos	Baquillas Medianas	<input type="button" value="Editar"/> <input type="button" value="Eliminar"/>

CEU

Centro Experimental Uyumbicho

Pictures of pigs, chickens, and a cow are displayed on the right side of the page.

Tipos Alimentos: En la parte superior se ingresa el nombre del tipo Alimento, en la tabla se muestra los datos ingresados, el botón Seleccionar muestra los datos de esa fila en la parte superior y podemos modificar la información al dar click en el botón actualizar se guarda ésta y el botón eliminar borra la información de esa fila.



UNIVERSIDAD CENTRAL DEL ECUADOR

Sistema de Registro de Animales y sus Derivados del Centro Experimental Uyumbicho

Firefox - Registro de Tipos de Alimentos +
localhost:8080/ProyectoCEU/contenido/mantenimiento/TipoAlimento.jsf
Más visitados Comenzar a usar Firefox Últimas noticias Marcadores

CENTRO EXPERIMENTAL UYUMBICHO

Los Datos se Guardaron Correctamente

User Actual Cerrar Session Julio Jaramillo

MANTENIMIENTO

- Tipos Animales
- Razas
- Hanentamiento Animales
- Tipos Alimentos
- Categorías
- Tipos Alimento
- Tipos Vacunas
- Tipos de Productos
- Unidades de Productos
- Vias Aplicación de Vacunas
- Dificultades en el Parto

REGISTRO TIPOS ALIMENTOS

Nombre:

Guardar Actualizar

TABLA DE TIPOS DE ALIMENTOS

IDENTIFICADOR	NOMBRE	Acciones
3	balanceado	<input type="button" value="Seleccionar"/> <input type="button" value="Eliminar"/>
4	balanceado2	<input type="button" value="Seleccionar"/> <input type="button" value="Eliminar"/>

CEU

Centro Experimental Uyumbicho

Tipos Vacunas: En la parte superior se ingresa los datos de la nueva tipo de vacuna como: fecha, se selecciona el tipo de animal, el nombre del tipo de vacuna y en la tabla se muestra los datos ingresados, el botón Seleccionar muestra los datos de esa fila en la parte superior y podemos modificar la información al dar click en el botón actualizar se guarda ésta y el botón eliminar borra la información de esa fila.

Firefox - Registro de Tipos de Pesos +
localhost:8080/ProyectoCEU/contenido/mantenimiento/TipoVacuna.jsf
Más visitados Comenzar a usar Firefox Últimas noticias Marcadores

CENTRO EXPERIMENTAL UYUMBICHO

User Actual Cerrar Session Gustavo Alvarez

ADMINISTRACION

- Perfiles
- Usuarios
- Asignación de Permisos
- Mantenimiento de Permisos

MANTENIMIENTO

- Mantenimiento Animales
- Destinos Producción
- Dificultades en el Parto
- Vias Aplicación de Vacunas
- Unidades de Productos
- Tipos de Productos

REGISTRO TIPOS DE VACUNAS

Fecha: 04/03/2012
Seleccione el Tipo Animal: Bovinos
Nombre: Abtosa

Guardar Actualizar

TABLA DE TIPOS DE VACUNAS

FECHA	TIPO ANIMAL	NOMBRE	Acciones
2012-03-01	Bovinos	Triple	<input type="button" value="Seleccionar"/> <input type="button" value="Eliminar"/>
2012-03-01	Bovinos	Leptospirosis	<input type="button" value="Seleccionar"/> <input type="button" value="Eliminar"/>
2012-03-01	Bovinos	Brucelosis	<input type="button" value="Seleccionar"/> <input type="button" value="Eliminar"/>
2012-03-01	Bovinos	Abtosa	<input type="button" value="Seleccionar"/> <input type="button" value="Eliminar"/>
2012-03-01	Bovinos	Vircas	<input type="button" value="Seleccionar"/> <input type="button" value="Eliminar"/>

CEU

Centro Experimental Uyumbicho



Tipo de Productos: En la parte superior se ingresa los datos de un nuevo tipo de producto como: fecha, el nombre del tipo de producto y se selecciona la unidad de medida en el combo de no estar dar clic en el botón Nueva Unidad e ingresar.



En la tabla se muestra los datos ingresados, el botón Seleccionar muestra los datos de esa fila en la parte superior y podemos modificar la información al dar click en el botón actualizar se guarda ésta y el botón eliminar borra la información de esa fila.

Unidades de Productos: En la parte superior se escoge la fecha, se ingresa el nombre de la unidad, en la tabla se muestra los datos ingresados, el botón Seleccionar muestra los datos de esa fila en la parte superior y podemos modificar la información al dar click en el botón actualizar se guarda ésta y el botón eliminar borra la información de esa fila.



UNIVERSIDAD CENTRAL DEL ECUADOR

Sistema de Registro de Animales y sus Derivados del Centro Experimental Uyumbicho

REGISTRO DE UNIDADES

FECHA	IDENTIFICADOR	NOMBRE	ACCIONES
2012-03-05	2	1 Litro	<input type="button" value="Seleccionar"/> <input type="button" value="Eliminar"/>
2012-03-06	13	1 libra	<input type="button" value="Seleccionar"/> <input type="button" value="Eliminar"/>
2012-03-06	14	1 Libra	<input type="button" value="Seleccionar"/> <input type="button" value="Eliminar"/>

Vías de aplicación de Vacunas: En la parte superior se escoge la fecha, se ingresa el nombre de las vías de aplicación de la vacuna, en la tabla se muestra los datos ingresados, el botón Seleccionar muestra los datos de esa fila en la parte superior y podemos modificar la información al dar click en el botón actualizar se guarda ésta y el botón eliminar borra la información de esa fila.

REGISTRO DE LAS VÍAS DE APLICACIÓN DE VACUNAS

IDENTIFICADOR	NOMBRE	ACCIONES
3	Intramuscular	<input type="button" value="Seleccionar"/> <input type="button" value="Eliminar"/>
4	Venosa	<input type="button" value="Seleccionar"/> <input type="button" value="Eliminar"/>
5	Sucutanea	<input type="button" value="Seleccionar"/> <input type="button" value="Eliminar"/>



Dificultades en el Parto: Se escoge la fecha, se ingresa el nombre de la dificultad del parto, en la tabla se muestra los datos ingresados, el botón Seleccionar muestra los datos de esa fila en la parte superior y podemos modificar la información al dar click en el botón actualizar se guarda ésta y el botón eliminar borra la información de esa fila.

The screenshot shows a web application window titled "Registro de las Dificultades al P...". The URL is "localhost:8080/ProyectoCEU/contenido/mantenimiento/DificultadParto.jsf". The page has a header with the university logo and the text "CENTRO EXPERIMENTAL UYUMBICHO". A message at the top right says "Los Datos fueron Modificado exitosamente". On the left is a sidebar with navigation links like "MANTENIMIENTO", "Tipos Animales", "Razas", etc. The main content area has a form for "REGISTRO TIPOS DE DIFICULTADES AL PARTO" with fields for "Fecha", "Nombre", and "Registrador por". Below it is a table titled "TABLA DE DIFICULTADES AL PARTO" with two rows of data. The table columns are "IDENTIFICADOR", "NOMBRE", and "ACCIONES". The first row has "1" and "Sin Dificultad" in the first two columns, and "Seleccionar" and "Eliminar" buttons in the last column. The second row has "2" and "Normal" in the first two columns, and "Seleccionar" and "Eliminar" buttons in the last column. To the right of the table is a sidebar with images of farm animals and the text "Centro Experimental Uyumbicho".

Destino de la Producción: Se ingresa el nombre de las aplicación de la vacuna, en la tabla se muestra los datos ingresados, el botón Seleccionar muestra los datos de esa fila en la parte superior y podemos modificar la información al dar click en el botón actualizar se guarda ésta y el botón eliminar borra la información de esa fila.



UNIVERSIDAD CENTRAL DEL ECUADOR

Sistema de Registro de Animales y sus Derivados del Centro Experimental Uyumbicho

MANTENIMIENTO

- Tipos Animales
- Razas
- Mantenimiento Animales
- Tipos Alimentos
- Categorías
- Tipos Alimento
- Tipos Vacunas
- Tipos de Productos
- Unidades de Productos
- Vías Aplicación de Vacunas
- Dificultades en el Parto

REGISTRO DE DESTINOS DE LA PRODUCCIÓN

Nombre :

Registrador por: Julio Jaramillo Sanchez

Guardar Actualizar

TABLA DE DESTINO PRODUCCIÓN

Identificador	Fecha	NOMBRE	Acciones
1	2012-03-06	Tereros	<input type="button" value="Seleccionar"/> <input type="button" value="Eliminar"/>

CEU

Razas: Se ingresa los datos de la nueva raza como la fecha, el nombre de la raza y se selecciona el tipo animal a la cual pertenece esa raza, en la tabla se muestra los datos ingresados, el botón seleccionar muestra los datos de esa fila en la parte superior y podemos modificar la información al dar click en el botón actualizar se guarda ésta y el botón eliminar borra la información de esa fila.

MANTENIMIENTO

- Tipos Animales
- Razas
- Mantenimiento Animales
- Tipos Alimentos
- Categorías
- Tipos Alimento
- Tipos Vacunas
- Tipos de Productos
- Unidades de Productos
- Vías Aplicación de Vacunas
- Dificultades en el Parto

REGISTRO RAZAS

Fecha:

Seleccione el Tipo Animal :

Nombre :

Registrador por: Julio Jaramillo Sanchez

Guarda Actualizar

TABLA DE RAZAS

FECHA	NOMBRE	Tipo Animal	Acciones
2012-03-01	Jersy	Bovinos	<input type="button" value="Seleccionar"/> <input type="button" value="Eliminar"/>
2012-03-01	Holstein Friesian	Bovinos	<input type="button" value="Seleccionar"/> <input type="button" value="Eliminar"/>
2012-03-01	Landrace por Yock	Porcinos	<input type="button" value="Seleccionar"/> <input type="button" value="Eliminar"/>
2012-03-01	Durock	Porcinos	<input type="button" value="Seleccionar"/> <input type="button" value="Eliminar"/>

CEU

MantenimientoAnimales: En la tabla muestra los datos de los animales registrados, el botón seleccionar muestra los datos de esa fila en la parte superior



y podemos modificar la información al dar click en el botón actualizar se guarda ésta información y el botón eliminar borra la información de esa fila.

The screenshot shows a Firefox browser window with the URL <localhost:8080/ProyectoCEU/contenido/mantenimiento/MantenimientoAnimal.jsf>. The title bar says "Mantenimiento de los Animal". The page header features the logo of the Universidad Central del Ecuador and the text "CENTRO EXPERIMENTAL UYUMBICHO". On the right, there's a user session icon for "Gustavo Gutiérrez". A sidebar on the left lists various animal-related categories. The main content area has a form for entering animal details like type, category, date of birth, sex, and parents, followed by a table titled "TABLA DE ANIMALES" showing four entries:

FECHA	TIPO ANIMAL	Número Identificación Animal	ACCIONES
2012-03-01	Bovinos	12H1- 03- 01	[Actualizar] [Eliminar]
2011-10-03	Avícola	11H1- 10- 03	[Actualizar] [Eliminar]
2012-03-01	Bovinos	12H2- 03- 01	[Actualizar] [Eliminar]

To the right of the table, there's a sidebar titled "CEU" showing small images of various farm animals.

Producción: En este menú se registra la información de la producción del Centro Experimental Uyumbicho y solo tiene la opción de guardar los datos, se tiene las siguientes páginas.

The screenshot shows a menu titled "PRODUCCION" with options: Lechera, Huevos, Miel y Polen, and Productos.

Lechera: En la tabla cargan las vacas que están lechando y se ingresa la cantidad de producción de leche de la mañana y de la tarde, se visualiza la cantidad total del grupo de vacas de producción de leche en la mañana y en la tarde además la producción diaria de guarda la información.



UNIVERSIDAD CENTRAL DEL ECUADOR

Sistema de Registro de Animales y sus Derivados del Centro Experimental Uyumbicho

IDENTIFICADOR	Cantidad Leche en Mañana	Cantidad Leche en Tarde
1H1- 10- 03	48	39
1H2- 03- 01	34	28

Huevos: Se ingresa los datos de la producción de huevos del Centro Experimental Uyumbicho como: la cantidad de huevos sanos y cantidad de huevos rotos, se guarda la información.

Los Datos se Guardaron Correctamente

Miel y Polen: Se ingresa los datos de la producción de miel del Centro Experimental Uyumbicho como: la cantidad de Miel y cantidad de Polen, se guarda la información.



UNIVERSIDAD CENTRAL DEL ECUADOR

Sistema de Registro de Animales y sus Derivados del Centro Experimental Uyumbicho

Productos: Se ingresa los datos de los productos realizados en el Centro Experimental Uyumbicho como: la fecha, se selecciona el tipo de producto, se ingresa la cantidad de productos realizados, se guarda la información adicionalmente se muestra el stock de cada tipo de producto.

Facturación: En este menú se registra las ventas del Centro Experimental Uyumbicho tanto de animales como de productos, los Clientes y solo tiene la opción de guardar los datos y tenemos las siguientes páginas.

Clientes: En la parte superior se ingresa la información nueva de los clientes como: el número de cedula, nombre, apellido, dirección, teléfono y se selecciona guardar, en la tabla se muestra los datos ingresados, el botón Seleccionar muestra



UNIVERSIDAD CENTRAL DEL ECUADOR

Sistema de Registro de Animales y sus Derivados del Centro Experimental Uyumbicho

los datos de esa fila en la parte superior y podemos modificar la información al dar click en el botón actualizar se guarda ésta y el botón eliminar borra la información de esa fila.

DATOS DEL CLIENTE

Cédula:	011234567890
Nombre :	Maria
Apellido :	Caizapanta
Dirección :	Latacunga
Teléfono :	022716082

ACCIONES

FECHA	Cédula	Nombre	Apellido	ACCIONES
2012-03-06	1715563944	Narcisa	Chisaguano	<input type="button" value="Seleccionar"/> <input type="button" value="Eliminar"/>
2012-03-06	512367890	Maria	Caizapanta	<input type="button" value="Seleccionar"/> <input type="button" value="Eliminar"/>
2012-03-06	1717592876	Jorge	Caiza	<input type="button" value="Seleccionar"/> <input type="button" value="Eliminar"/>

REGISTRO DE CLIENTES

Guardar Actualizar

TABLA DE CLIENTES

FECHA	Cédula	Nombre	Apellido	ACCIONES
2012-03-06	1715563944	Narcisa	Chisaguano	<input type="button" value="Seleccionar"/> <input type="button" value="Eliminar"/>
2012-03-06	512367890	Maria	Caizapanta	<input type="button" value="Seleccionar"/> <input type="button" value="Eliminar"/>
2012-03-06	1717592876	Jorge	Caiza	<input type="button" value="Seleccionar"/> <input type="button" value="Eliminar"/>

CEU

Venta de Animales: Esta página permite facturar la venta de todo tipo de animales y registrar los clientes si son nuevos al presionar en el botón Nuevo Cliente, se ingresa la información y se guarda.

DATOS DEL ANIMAL

Fecha: Tue Mar 06 15:50:56 COT 2012

Nuevo Cliente:

Seleccione el Cliente:

Cédula:	Nombre:	Apellido:	Dirección:	Teléfono:
1716543890	Brittany	Caiza	Gumani	098765135

CLIENTE

Cédula: 1716543890
Nombre: Brittany
Apellido: Caiza
Dirección: Gumani
Teléfono: 098765135
Aceptar Cancelar

REGISTRO DE VENTA DE ANIMAL

Seleccione el tipo Animal: Seleccione la Cate

Cantidad: 0 Valor: 0.0

ID	Tipo Animal	ID Animal
No records found.		

SubTotal:
IVA:
Total:

Guardar Listar Facturas

Gracias por su Compra que tenga UN BUEN DÍA Fue Atendido por:Julio Jaramillo Sanchez

CEU



UNIVERSIDAD CENTRAL DEL ECUADOR

Sistema de Registro de Animales y sus Derivados del Centro Experimental Uyumbicho

Si el Cliente ya está registrado, se selecciona el cliente del combo cliente y carga la información, se selecciona del combo tipo de animal a vender, si son animales individuales, se selecciona la categoría, el animal y se ingresar el valor, se agrega los detalles a la factura y se guarda.

The screenshot shows a Firefox browser window with the title 'Registro de Venta de Animales'. The URL is 'localhost:8080/ProyectoCEU/contenido/VentaAnimal.jsf'. The main content area displays a form for a single animal sale. The form includes fields for Date (Tue Mar 27, 2012), Client (Nuevo Cliente, Cesar Chisaguano), and Animal Type (Bovinos). The 'DETALLES VENTA' table shows one row: Bovinos at 500.0. SubTotal: 500.0, IVA: 60.0, Total: 560.0. A message at the bottom says 'Gracias por su Compra que tenga UN BUEN DÍA Fue Atendido por: Gustavo Alvaréz Jacome'. On the left sidebar, under 'ADMINISTRACION', 'Perfiles' is selected. On the right sidebar, there's a 'CEU' section with images of animals.

Si los animales a vender son grupales, se selecciona la categoría, se ingresa la cantidad a vender y el valor individual proceder a agregar al detalle factura y se guarda.

The screenshot shows the same 'Registro de Venta de Animales' page but for a group sale. The 'DETALLES VENTA' table shows two rows: Bovinos at 400.0 and Avicola at 2.0, resulting in a total of 452.40. The rest of the interface is identical to the previous screenshot, including the sidebar menus and the 'CEU' section on the right.



UNIVERSIDAD CENTRAL DEL ECUADOR

Sistema de Registro de Animales y sus Derivados del Centro Experimental Uyumbicho

Se tiene la opción de listar las Facturas.

Id	Fecha	Cliente	Total Venta	Detalles			
				Id Det	Descripción	Cantidad	Valor Unitario
1	2012-03-06	Brittany	452.48	2	Animales: Bovinos Categorías: Pollos	2	2.0
				1	Animals: Bovinos Categorías: Baconas Vientre Identificación: 11H1-10-03	1	400.0
2	2012-03-06	Maria	459.2	4	Animales: Bovinos Categorías: Pollos	5	2.0
				3	Animals: Bovinos Categorías: Bequillas Medianas Identificación: 10H5-03-01	1	400.0

Venta de Productos: Esta página permite registrar la venta de productos elaborados en el CEU y registrar los clientes.

Id	Descripción	Cantidad	Valor Unitario	Valor SubTotal
SubTotal:	6.0			
IVA:	0.0720000000000001			
Total:	6.072			

Consumo: En este menú se registra la información del consumo de la producción del CEU y solo tiene la opción de guardar los datos, se tiene la siguiente página.



CONSUMO
Consumo de
Producción

Consumo de Producción: En esta página se registra el consumo de la producción sea de los productos elaborados en centro, como la leche, huevos, miel, etc.

Se ingresa la información como la fecha, se selecciona el tipo de producto y el destino del consumo, ingresamos la cantidad y se guarda la información.

Reportes:

REPORTES
Ficha animal
Animales Por Nacimiento
Partos Animales
Producción

Reporte de la Ficha del animal: Se saca la información referente al animal ingresado su número de identificación y lo exportamos a pdf para imprimir.



UNIVERSIDAD CENTRAL DEL ECUADOR
Sistema de Registro de Animales y sus Derivados del Centro Experimental Uyumbicho

Ficha-2 X

Tipo Animal	Animal	Categoría	Fecha Nacimiento	Raza	Edad	Sexo Animal	Identificación Madre	Raza Padre
Bovinos	08H1- 02- 06	Vacas	6/02/08 12:	Holstein	4 años 1 meses 11dias	Hembra	02H13	Nicopol

Animales por Nacimiento: Reporte de Animales que nacieron en una fecha seleccionada, se ingresa la fecha a consultar y se selecciona exportar a pdf para imprimir.

Firefox Perfiles

localhost:8080/ProyectoCEU/reportes/AnimalesReporte.jsf

Más visitados Comenzar a usar Firefox Últimas noticias

Centro Experimental Uyumbicho

Administración Perfiles Usuarios Asignación de Permisos

CEU

Fecha de nacimiento: 1/03/12 Exportar a PDF

Usuario Actual Gustavo Alvarez

Cerrar Sesión

Página: 1 de 1

LISTADOS DE ANIMALES

Fecha de Nacimiento: Thu Mar 01 00:00:00 COT 2012

ANIMAL	TIPO ANIMAL	ESTADO
12M1- 03- 01	1	Bovinos
12H4- 03- 01	1	Bovinos
12M2- 03- 01	1	Bovinos
12H5- 03- 01	1	Bovinos

Cantidad de Animales: 4



UNIVERSIDAD CENTRAL DEL ECUADOR

Sistema de Registro de Animales y sus Derivados del Centro Experimental Uyumbicho

Producción: Dentro de esta opción se tiene la pantalla que permite ver los reportes de la producción del Centro Experimental Uyumbicho.

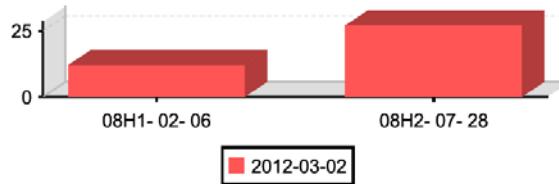
Lechera: Reporte de producción de leche Diaria ingresamos la fecha a calcular y nos presenta en una tabla y se puede exportar a pdf el reporte.

Número Arete/Identificación	Ordeño en la Mañana	Ordeño en la Tarde	producción Diaria Individual
08H1- 02- 06	10.0	12.0	22.0
08H2- 07- 28	15.0	15.0	30.0

Cantidad animales:2 Total Mañana:25 Total Tarde:27 52



Página:1



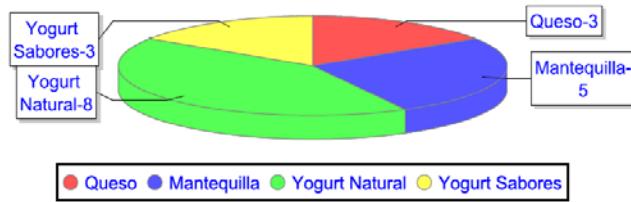
Reportes de Productos: Se selecciona la fecha inicial y final de la consulta, se genera el reporte.

Screenshot of a web browser displaying a report titled "Informe de Elaboración de Prod...". The URL is localhost:8080/ProyectoCEU/reportes/ProduccionReporte.jsf. The page features the logo of the Centro Experimental Uyumbicho and a user session icon for "User Actual Gustavo Alvarez". On the left, a sidebar menu includes "ADMINISTRACIÓN" (Perfiles, Usuarios, Asignación de Permisos, Mantenimiento de Permisos) and "MANTENIMIENTO" (Mantenimiento). The main content area shows a table of production data:

Fecha Elaboración	Cantidad	Tipo de Producto
2012-03-02	5	Mantequilla
2012-03-03	3	Queso
2012-03-01	3	Yogurt Sabores
2012-03-01	8	Yogurt Natural

Screenshot of an Adobe Reader window titled "Productos-4.pdf - Adobe Reader". The window displays the "REGISTRO DE ELABORACIÓN DE PRODUCTOS" report. It includes a header image of various dairy products and a table of production data:

FECHA	CANTIDAD	TIPO PRODUCTO
3/03/12 12:00 AM	3	Queso
2/03/12 12:00 AM	5	Mantequilla
1/03/12 12:00 AM	8	Yogurt Natural
1/03/12 12:00 AM	3	Yogurt Sabores

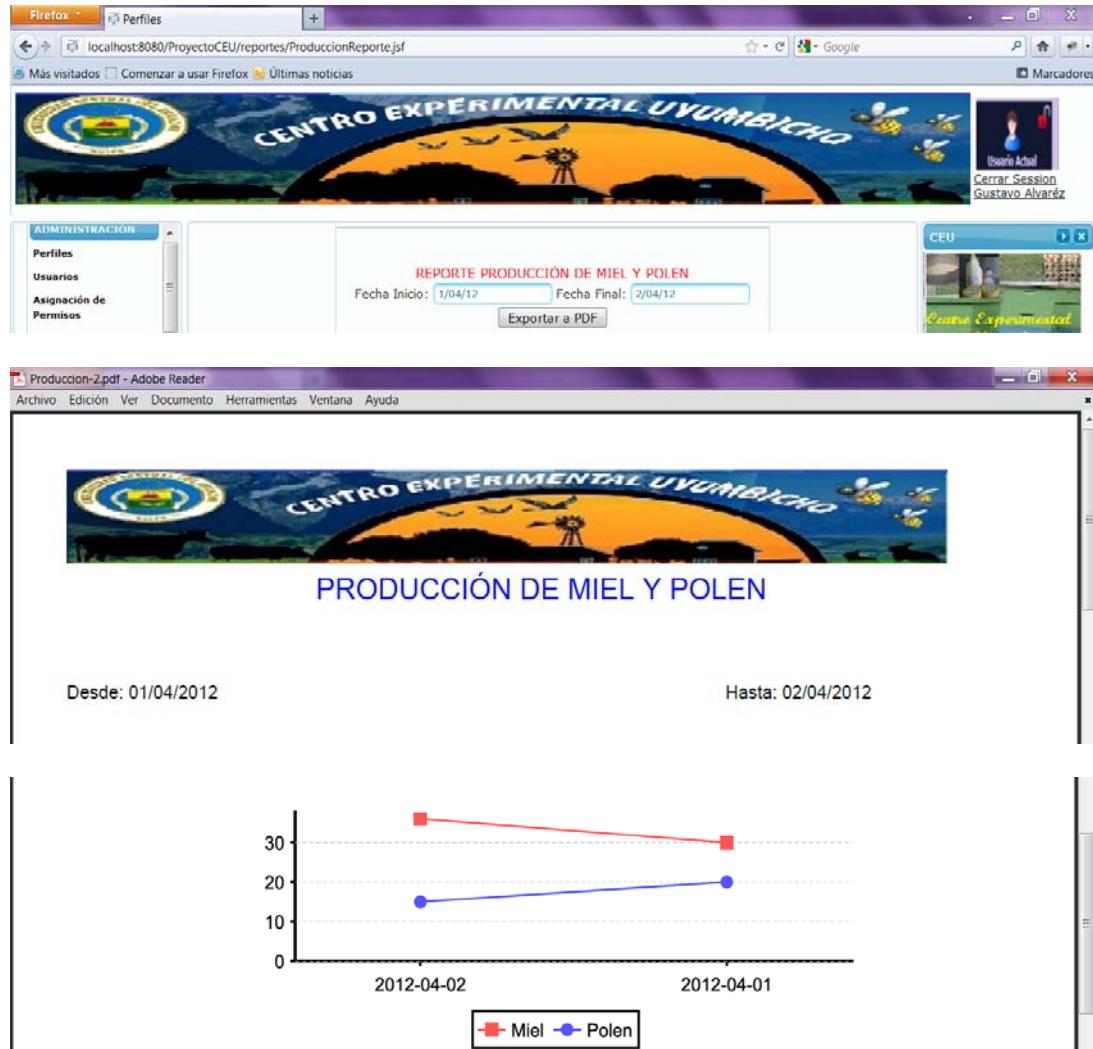


Reportes de Huevos: Se selecciona la fecha inicial y final de la consulta, se genera el reporte y se despliega un grafico de curva de producción.





Reportes de Miel y Polen: Se selecciona la fecha inicial y final de la consulta, se genera el reporte.



4.3.5.5 Reporte de Consumo de productos: En esta consulta se despliega el consumo de los productos elaborados en el Centro Experimental Uyumbicho.



UNIVERSIDAD CENTRAL DEL ECUADOR
Sistema de Registro de Animales y sus Derivados del Centro Experimental Uyumbicho



LISTADO DE CONSUMO DE PRODUCTOS

FECHA CONSUMO	NOMBRE DE TIPO DE PRODUCTO	CANTIDAD	DESTINO
18/03/12 12:00 AM	Mantequilla	2.0	Ingeniero
18/03/12 12:00 AM	Queso	3.0	Estudiantes
18/03/12 12:00 AM	Yogurt Sabores	1.0	Estudiantes
18/03/12 12:00 AM	Mantequilla	5.0	Estudiantes
18/03/12 12:00 AM	Yogurt Sabores	6.0	Industrializacion

Página: 1

Reporte de Consumo de alimento de los animales grupales: En este informe se despliega el consumo de los alimentos por parte de los animales grupales sean avícolas o cuyes.



INFORME DE ALIMENTO

FECHA ALIMENTO	TIPO ANIMAL	CATEGORIA	ALIMENTO INDIVIDUAL	DIAS	CANTIDAD
1/03/12 12:00 AM	Avicola	Pollos	2.5	7	700.0
2/03/12 12:00 AM	Cuyes	Crias Lactantes	2.0	56	5040.0
3/03/12 12:00 AM	Avicola	Gallinas	3.0	4	528.0
4/03/12 12:00 AM	Avicola	Gallinas Descarte	3.5	5	402.5
6/03/12 12:00 AM	Cuyes	Machos Engorde	3.0	4	320.0
13/03/12 12:00 AM	Cuyes	Padres	5.0	5	300.0

Reporte de las muertes de los animales:



REPORTE DE MUERTE ANIMAL

numero_arete

fecha_muerte

causa_muerte

12H3- 01- 03

17/03/12 0:00

lobado

11H1- 05- 01

2/03/12 0:00

tuberculosis

Administración: En este menú se tiene la administración de usuarios, perfiles, permisos y se tiene las siguientes páginas.



Usuarios: Esta página tiene la tabla de usuarios registrados con sus perfiles y las opciones para editar, eliminar y crear un nuevo.



UNIVERSIDAD CENTRAL DEL ECUADOR

Sistema de Registro de Animales y sus Derivados del Centro Experimental Uyumbicho

Al seleccionar Nuevo, se ingresa la información del usuario y se selecciona los perfiles que va a tener.

Perfiles: Esta página se tiene la lista de perfiles con opciones para editar, eliminar y crear un nuevo.



UNIVERSIDAD CENTRAL DEL ECUADOR

Sistema de Registro de Animales y sus Derivados del Centro Experimental Uyumbicho

Nro	Descripción	Acciones
1	Administrador del Sistema	<input checked="" type="checkbox"/> <input type="checkbox"/>
5	Jefe Área Porcina	<input checked="" type="checkbox"/> <input type="checkbox"/>
6	Jefe Área Avícola	<input checked="" type="checkbox"/> <input type="checkbox"/>
2	Pasante Bovinos	<input checked="" type="checkbox"/> <input type="checkbox"/>
7	Jefe Área Cuyes	<input checked="" type="checkbox"/> <input type="checkbox"/>

Al seleccionar Nuevo, se ingresa el nombre del perfil y se selecciona si es de escritura es decir al loguearse con el usuario con el perfil de escritura puede editar, guardar, actualizar y eliminar, pero cuando no se selecciona escritura solo se puede editar, guardar y actualizar.

Nro	Descripción	Acciones
1	Administrador del sistema	<input checked="" type="checkbox"/> <input type="checkbox"/>
5	Jefe Área Porcina	<input checked="" type="checkbox"/> <input type="checkbox"/>
6	Jefe Área Avícola	<input checked="" type="checkbox"/> <input type="checkbox"/>
2	Pasante Bovinos	<input checked="" type="checkbox"/> <input type="checkbox"/>
7	Jefe Área Cuyes	<input checked="" type="checkbox"/> <input type="checkbox"/>

Asignación de Permisos: En esta página se puede asignar permisos a los perfiles, se puede crear nuevos permisos y se guarda la información.



Al seleccionar nuevo permiso ingresamos el nombre del permiso y la dirección URL de su ubicación.

Mantenimiento de Permisos: En la parte superior se ingresa los datos de un nuevo permiso como nombre del permiso y la dirección URL de su ubicación, en la tabla se muestra los datos ingresados, el botón Seleccionar muestra los datos de esa fila en la parte superior y se puede modificar la información al dar click en el botón actualizar se guarda ésta y el botón eliminar borra la información de esa fila.



UNIVERSIDAD CENTRAL DEL ECUADOR

Sistema de Registro de Animales y sus Derivados del Centro Experimental Uyumbicho

Permisos	ACCIONES
Tipos Animales	<input type="button" value="Seleccionar"/> <input type="button" value="Eliminar"/>
Razas	<input type="button" value="Seleccionar"/> <input type="button" value="Eliminar"/>
Mantenimiento Animales	<input type="button" value="Seleccionar"/> <input type="button" value="Eliminar"/>
Tipos Alimentos	<input type="button" value="Seleccionar"/> <input type="button" value="Eliminar"/>

JEFE DEL ÁREA DE BOVINOS Y PORCINOS (Animales Individuales).

Al loguearse como jefe del área de bovino y porcinos nos despliega en el lado izquierdo el menú que tiene acceso este perfil.

Permisos	ACCIONES
Tipos Animales	<input type="button" value="Seleccionar"/> <input type="button" value="Eliminar"/>
Razas	<input type="button" value="Seleccionar"/> <input type="button" value="Eliminar"/>
Mantenimiento Animales	<input type="button" value="Seleccionar"/> <input type="button" value="Eliminar"/>
Tipos Alimentos	<input type="button" value="Seleccionar"/> <input type="button" value="Eliminar"/>

En la parte superior derecha nos indica el usuario con el que se a logeado para

ingresar al sistema



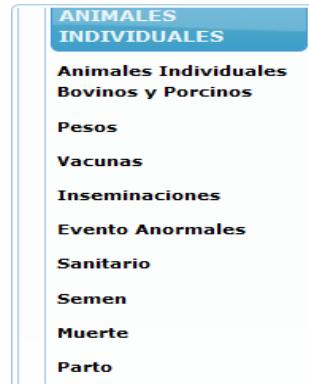
Cerrar Sesión
Narcisa Chisaguano

Al loguearse con este perfil tiene acceso al siguiente menú:

- Animales Individuales



Animales Individuales: Dentro de este menú se tiene las pantallas para registrar la información referente a estos animales que tienen identificación y son.



Animales Individuales Bovinos y Porcinos: En esta página se ingresa información de los animales como: el tipo de animal, la categoría, raza del padre, raza de la madre, raza del animal, se ingresa la fecha de nacimiento, la identificación de la madre y se guarda la información.

Pesos: En esta página se registra el peso del animal en sus diferentes categorías de vida, ingresando la siguiente información: la fecha del registro, se selecciona el tipo del animal, al animal, se ingresa el peso y se guarda.



UNIVERSIDAD CENTRAL DEL ECUADOR

Sistema de Registro de Animales y sus Derivados del Centro Experimental Uyumbicho

Vacunas: En esta página se registra la vacuna del animal, ingresando la siguiente información: fecha del registro, se selecciona el tipo del animal, al animal, el tipo de vacuna, la vía de aplicación, se ingresa la fecha de revacunación, una observación y se guarda.

Inseminaciones: Se registra las inseminaciones del animal, ingresando la siguiente información: fecha del registro, se selecciona el tipo del animal, al animal, el estado de inseminación que al registro es pendiente y se guarda.



UNIVERSIDAD CENTRAL DEL ECUADOR

Sistema de Registro de Animales y sus Derivados del Centro Experimental Uyumbicho

REGISTRO DE INSEMINACIONES

Fecha:
Seleccione el Tipo de Animal: Bovinos |
Seleccione el Animal: 12H2- 03- 01 |
Número de Inseminación:
Estado de Inseminación: Pendiente Positivo Negativo
Registrador por: Narcisa Chisaguano Caizapanta

TABLA DE INSEMINACIONES

FECHA	Tipo Animal	ANIMAL	NUMERO DE INSE	ESTADO	FECHA INSE-EFECTIVA	ACCIONES
2012-03-01	Bovinos	12H2- 03- 01	1	Pendiente	2012-03-01	<input type="button" value="Seleccionar"/>

Luego de la verificación se actualiza el estado y si es positivo se genera la fecha efectiva de la inseminación.

REGISTRO DE INSEMINACIONES

Fecha:
Seleccione el Tipo de Animal: Bovinos |
Seleccione el Animal: 12H2- 03- 01 |
Número de Inseminación:
Estado de Inseminación: Pendiente Positivo Negativo
Registrador por: Narcisa Chisaguano Caizapanta

TABLA DE INSEMINACIONES

FECHA	Tipo Animal	ANIMAL	NUMERO DE INSE	ESTADO	FECHA INSE-EFECTIVA	ACCIONES
2012-03-01	Bovinos	12H2- 03- 01	1	Positivo	2012-03-01	<input type="button" value="Seleccionar"/>

Eventos Anormales: En esta página se registra los eventos anormales del animal, ingresando la siguiente información: la fecha del registro, se selecciona el tipo del animal, al animal, se ingresa el nombre del evento y se guarda.



UNIVERSIDAD CENTRAL DEL ECUADOR

Sistema de Registro de Animales y sus Derivados del Centro Experimental Uyumbicho

REGISTRO DE EVENTOS DE LOS ANIMALES

FECHA EVENTO	TIPO ANIMAL	ANIMAL	NOMBRE	AÇIONES
2012-03-01	Bovinos	12H3- 03- 01	Aborto	[Seleccionar] [Eliminar]
2012-03-01	Bovinos	12H3- 03- 01	Aborto	[Seleccionar] [Eliminar]

Sanitario: En esta página se registra los controles veterinarios del animal en sus diferentes etapas de vida, ingresando la siguiente información la fecha del registro, se selecciona el tipo del animal, al animal, se ingresan los síntomas, diagnósticos, tratamiento y se guarda.

REGISTRO DE CONTROLES SANITARIOS

FECHA	TIPO ANIMAL	ANIMAL	SINTOMAS	DIAGNOSTICO	TRATAMIENTO	AÇIONES
2012-03-01	Bovinos	12H2- 03- 01	Fiebre Alta	Resfrio	Antibioticos	[Seleccionar]

Semen: En esta página se registra la recolección de cantidad de semen ingresando la siguiente información: la fecha del registro, se selecciona el tipo del animal, al animal, se ingresa la cantidad de recolección y se guarda.



UNIVERSIDAD CENTRAL DEL ECUADOR

Sistema de Registro de Animales y sus Derivados del Centro Experimental Uyumbicho

Muerte: En esta página se registra la muerte del animal, ingresando la siguiente información: la fecha del registro, se selecciona el tipo del animal, la categoría, al animal, se ingresa la causa de la muerte y se guarda.

Parto: En esta página se registra el parto del animal, ingresando la siguiente información: la fecha del parto, se selecciona el tipo del animal, al animal, la dificultad al parto se ingresa los hijos nacidos, vivos, momificados y se generan los muertos, el número de machos y se genera las hembras, se guarda.



UNIVERSIDAD CENTRAL DEL ECUADOR

Sistema de Registro de Animales y sus Derivados del Centro Experimental Uyumbicho

REGISTRO DE PARTOS

Fecha: 01/03/2012
Seleccione el Tipo de Animal: Porcinos
Seleccione el Animal: 1433
Seleccione la Dificultad: Normal
Número de Hijos Nacidos: 2
Número de Hijos VIVOS: 2
Número de Hijos Momificados: 0 Número de Machos: 1
Número de Hijos Muertos: 0 Número de Hembras: 1
Número de Parto: 1
Registrador por: Narcisa Chisaguano Caizapanta

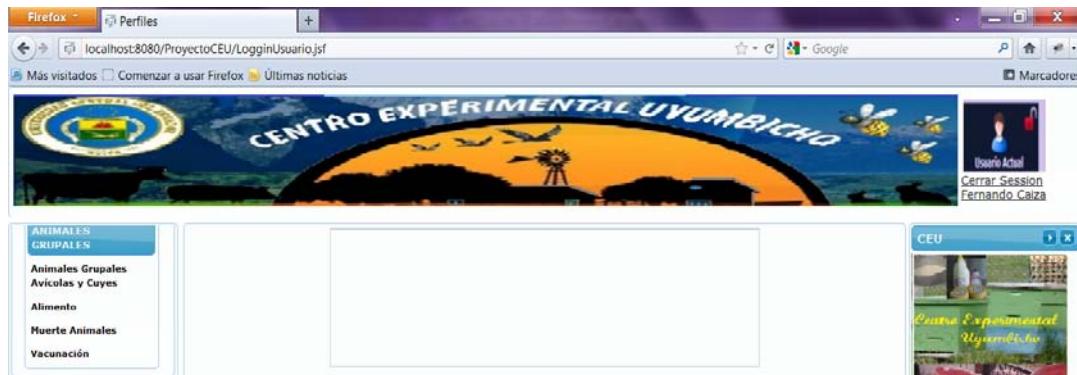
A continuación se registran los hijos dando click en el botón agregar y se guarda la lista de hijos.

REGISTRO DE ANIMALES DE LOS HIJOS

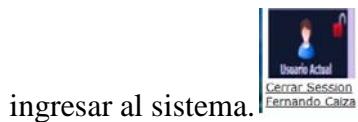
Fecha de Nacimiento: Thu Mar 01 00:00:00 COT 2012
Escoja el sexo: Macho
Tipo de Animal de la Madre: Porcinos
Identificador de la Madre: 1433
Número de Parto: 1
Número de Identificación/Arte: _____
Raza de la Madre: Durock
Seleccione Raza del Padre: Durock
Raza: Durock
Registrador por: Narcisa Chisaguano Caizapanta

JEFE DEL ÁREA AVÍCOLA Y CUYES (Animales Grupales).

Al loguearse como jefe del área de avícola y cuyes nos despliega en el lado izquierdo el menú que tiene acceso este perfil.



En la parte superior derecha nos indica el usuario con el que se a logeado para



ingresar al sistema.

Al loguearse con este perfil tiene acceso al siguiente menú:

- Animales Grupales.

Animales Grupales: Dentro de este menú se tiene las pantallas para registrar la información referente, a estos animales grupales que no tienen identificación individual se manejan por categorías.



Animales Grupales Avícolas y Cuyes: En esta página se registra la cantidad de animales que se tiene en cada categoría, se ingresa los datos como: la fecha de registro, se selecciona el tipo de animal, la categoría y se ingresa la cantidad de animales, observación y se guarda.



UNIVERSIDAD CENTRAL DEL ECUADOR

Sistema de Registro de Animales y sus Derivados del Centro Experimental Uyumbicho

Alimentación: En esta página se registran la cantidad de alimento que consumen los animales que se tiene en cada categoría, se ingresa los datos como la fecha de registro, se selecciona el tipo de animal, la categoría y se ingresa la cantidad de animales, observación y se guarda.

Muerte: En esta página se registran la cantidad de animales que mueren en cada categoría, se ingresa los datos como: la fecha de registro, se selecciona el tipo de animal, la categoría y se ingresa la cantidad de animales muertos, causa y se guarda.



UNIVERSIDAD CENTRAL DEL ECUADOR

Sistema de Registro de Animales y sus Derivados del Centro Experimental Uyumbicho

Vacunación: En esta página se registran las vacunas que se les colocan a los animales en cada categoría, se ingresa los datos como: la fecha de registro, se selecciona el tipo de animal, la categoría, tipo vacuna, vía de aplicación, se ingresa la dosis de la vacuna y se guarda.



ANEXO 3

MANUAL TÉCNICO

Explicar el código de programación de tesis de grado.

OBJETIVOS

Detallar cada una de las clases empleadas bajo el uso de la herramienta libre, ECLIPSE, JBOSS 7.1, POSTGRESQL 8.4.8, JDK1.6.u3 I.586,IREPORT 4.5.0

HERRAMIENTAS UTILIZADAS

ECLIPSE

Herramienta entorno de programación de java.

JBOSS 7.1

Servidor de Aplicaciones.

POSTGRES

Base de datos.

JDK 1.6.u3

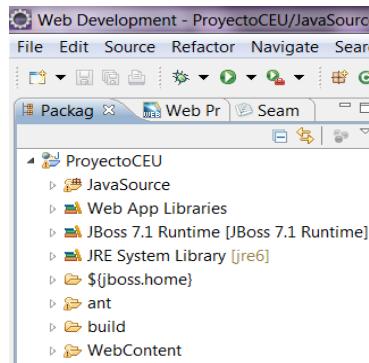
Máquina Virtual.

IREPORT 4.5.0

Creación de Reportes.



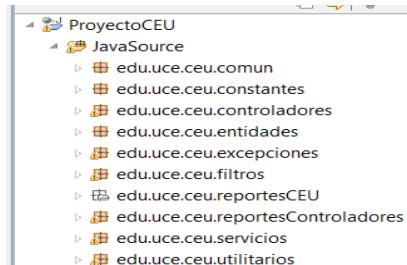
CODIGO FUENTE



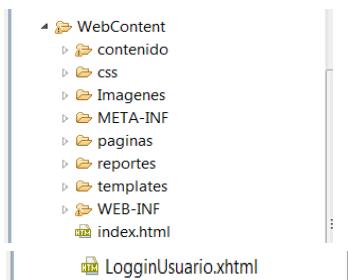
Figura_1. Visualización del proyecto

Dentro del proyecto se tiene las siguientes partes:

Primera el código Java en la carpeta JavaSource.



Segunda la parte páginas web en la carpeta webContent.



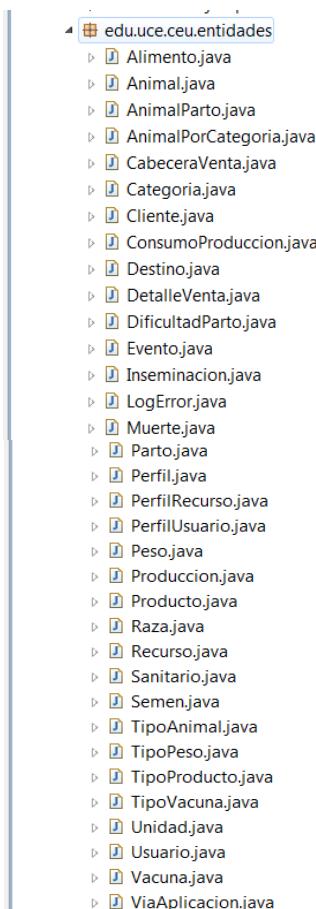


Dentro de la primera parte se tiene:

JPA

Java Persistence API, más conocida por su sigla JPA, es la API de persistencia desarrollada para la plataforma Java EE e incluida en el estándar EJB3. Esta API busca unificar la manera en que funcionan las utilidades que proveen un mapeo objeto-relacional. El objetivo que persigue el diseño de esta API es no perder las ventajas de la orientación a objetos al interactuar con una base de datos, y permitir usar objetos regulares (conocidos como POJOs).

Edu.uce.ceu.entidades



Figura_2. Creación de los Objetos de la BDD en el JPA.



Todas las clases que están en proyecto en el paquete que son clases de persistencia se manejan de la misma forma con anotaciones @ y se describirá en la siguiente clase como ejemplo:

TipoAnimal.java

Sección que permite importar las **librerías** que se necesitan para la persistencia

```
package edu.uce.ceu.entidades;  
  
import java.io.Serializable;  
  
import javax.persistence.*;  
  
import java.util.Date;  
  
import java.util.List;
```

Las anotaciones: **@Entity** nos permite identificar que es una entidad, **@Table** permite refenciar a la tabla en la Base de Datos

```
/**  
 * The persistent class for the tipo_animal database table.  
 *  
 */  
  
@Entity  
  
@Table(name="tipo_animal")  
  
Public class TipoAnimal implements Serializable {  
  
    Private static final long serialVersionUID = 1L;
```

Las anotaciones: **@Id** para identificar el identificador de la clase, **@GeneratedValue** para identificar la estrategia de autogeneración de la clave primaria, **@Column** hace referencia a la columna de la Base de Datos, **@Temporal** se utiliza para campos de tipo fecha y permite especificar el tipo de temporal.

```
@Id  
  
@GeneratedValue(strategy=GenerationType.IDENTITY)  
  
@Column(name="id_tipo_animal")
```



```
private Integer idTipoAnimal;

@Temporal( TemporalType.DATE)
@Column(name="fecha_tipo_animal")

private Date fechaTipoAnimal;

@Column(name="nombre_tipo_animal")
private String nombreTipoAnimal;
```

Las anotaciones: **@OneToMany**, **@ManyToOne** las que permiten mapear las relaciones de la base de la datos.

```
//bi-directional many-to-one association to Animal
@OneToMany(mappedBy="tipoAnimal")
private List<Animal>animals;

//bi-directional many-to-one association to Categoría
@ManyToOne(mappedBy="tipoAnimal")
private List<Categoría>categorías;
```

A continuación se tiene el constructor y sus respectivos Gestos and Sets características de los POJOS y sobrescribimos el método `toString` de la clase padre para visualizar las entidades.

```
public TipoAnimal() {
}

public Integer getIdTipoAnimal() {
    return this.idTipoAnimal;
}

public void setIdTipoAnimal(Integer idTipoAnimal) {
    this.idTipoAnimal = idTipoAnimal;
}
```



```
public Date getFechaTipoAnimal() {  
    return this.fechaTipoAnimal;  
}  
  
public void setFechaTipoAnimal(Date fechaTipoAnimal) {  
    this.fechaTipoAnimal = fechaTipoAnimal;  
}  
  
public String getNombreTipoAnimal() {  
    return this.nombreTipoAnimal;  
}  
  
public void setNombreTipoAnimal(String nombreTipoAnimal) {  
    this.nombreTipoAnimal = nombreTipoAnimal;  
}  
  
public List<Animal> getAnimals() {  
    return this.animals;  
}  
  
public void setAnimals(List<Animal> animals) {  
    this.animals = animals;  
}  
  
public List<Categoria> getCategorias() {  
    return this.categorias;  
}  
  
public void setCategorias(List<Categoria> categorias) {  
    this.categorias = categorias;  
}  
  
public String toString(){  
    return "Tipo Animal>> id:"+idTipoAnimal+" nombre:"+nombreTipoAnimal;
```

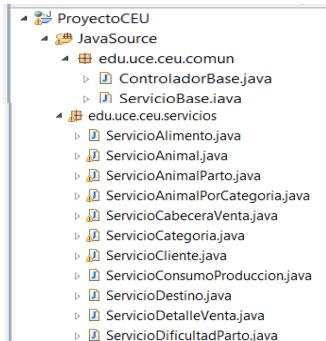


```
}
```

```
}
```

EJB (Enterprise Java Beans).

Los EJB o Enterprise Java Beans son componentes JEE que se ejecutan dentro de un container EJB, un entorno de ejecución dentro de un Application Server. El contenedor de EJB provee servicios al usuario, los cuales expone de manera transparente, como: Transacciones, Seguridad, etc. Los Enterprise Java Beans son componentes del lado del servidor que encapsulan la lógica del negocio de una aplicación.



Los Enterprise Java Beans encapsulan operaciones accesibles de modo remoto desde los clientes finales o desde los componentes de la capa de presentación WEB.

- Cada método de un EJB se ejecuta en su propio hilo.
- La ejecución de cada método de un EJB conforma una transacción (se ejecuta completamente con éxito o las posibles modificaciones realizadas son anuladas).

Tipos:

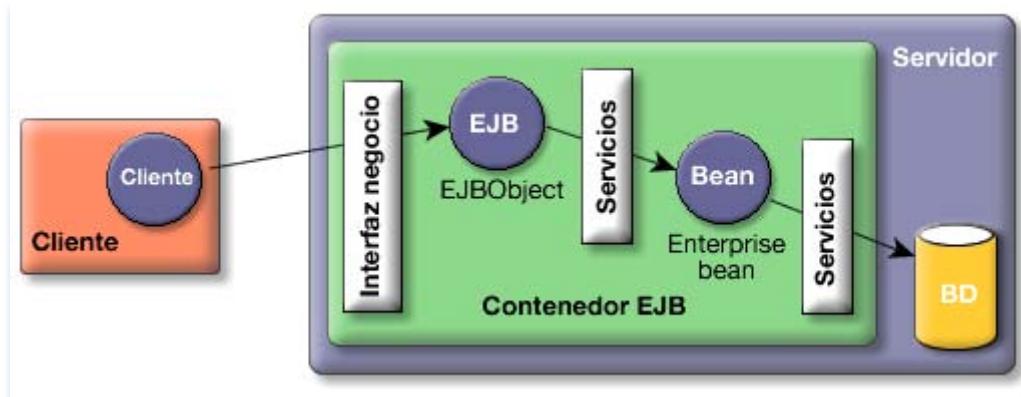
- Message driven EJBs.
- Implementan operaciones asíncronas.
- Suscripción, escucha y notificación de eventos.



- Session EJBs.

Implementan operaciones síncronas (llamada-respuesta).

- Stateful: Mantienen el estado conversacional con cliente atienden a un cliente específico y sus atributos se mantienen entre llamadas.
- Stateless: No mantienen estado (son independientes de los clientes).
- Singleton (en Java EE 6): Aseguran una instancia única compartida por todos los clientes.



edu.uce.ceu.comun: En este paquete está las clases comunes que tiene los métodos básicos para interactuar con la base de datos. De esta clase deben heredar los servicios que manipulan una entidad específica y es el padre de resto de servicios.

◀ edu.uce.ceu.comun
▷ ControladorBase.java
▷ ServicioBase.java



Proporciona métodos para manejar la persistencia de un Bean de Entidad, permite añadir, eliminar, actualizar y consultar así como manejar su ciclo de vida. Sus métodos más importantes son:

- **Insertar (T entidad):** Guarda la entidad en la base de datos .
- **Actualizar (T entidad):** Actualiza las modificaciones en la entidad en la base de datos.
- **Eliminar (T entidad):** Elimina la entidad en la base de datos.
- **BuscarPorId (Integer id):** Busca la entidad a través de su clave primaria.
- **Flush():** Sincroniza las entidades con el contenido de la base de datos.
- **List<T>buscarTodos():** Recupera todos los datos de la entidad, de la base de datos.

```
public abstract class ServicioBase<T> {

    @PersistenceContext
    protected EntityManager em;

    protected static Logger LOG;

    private Class<T> tipoEntidad;
    private Class<?> tipoServicio;

    /**
     * Constructor por defecto,
     *
     * @param tipoEntidad
     *         clase entidadsobre la cual se realizaran los operaciones
     *         sobre la base de datos
     * @param tipoServicio
     */
    public ServicioBase(Class<T> tipoEntidad, Class<?> tipoServicio) {
        this.tipoEntidad = tipoEntidad;
```



```
this.tipoServicio = tipoServicio;  
}  
  
/**  
 * Guardaunobjetoenla base dedatos  
 *  
 * @param entidad  
 */  
  
publicvoid insertar(T entidad) throws SeguridadesExpcion {  
    LOG.debug("Insertando Entidad>>" + entidad);  
    try {  
        em.persist(entidad);  
        em.flush();  
    } catch (Exception e) {  
        thrownew SeguridadesExpcion(Mensajes.ERROR_GUARDAR);  
    }  
}  
/**  
 * Actualizaunobjetoexistente, si no existecreaunonuevo  
 *  
 * @param entidad  
 */  
  
publicvoid actualizar(T entidad) throws SeguridadesExpcion {  
    try {  
        em.merge(entidad);  
        em.flush();  
    } catch (Exception e) {  
        thrownew SeguridadesExpcion(Mensajes.ERROR_EDITAR);  
    }  
}
```



```
/**  
 * Elimina un objeto de la base de datos  
 *  
 * @param entidad  
 */  
  
public void eliminar(T entidad) throws SeguridadesExpcion {  
  
    try {  
  
        em.remove(em.merge(entidad));  
  
        em.flush();  
  
    } catch (Exception e) {  
  
        thrownew SeguridadesExpcion(Mensajes.ERROR_ELIMINAR);  
  
    }  
  
}  
  
  
/**  
 * Busca un objeto dado su llave primaria  
 *  
 * @param id  
 * @return  
 */  
  
public T buscarPorId(Integer id) throws SeguridadesExpcion {  
  
    try {  
  
        LOG.debug("Buscando Entidad con id>>" + id);  
  
        returnem.find(tipoEntidad, id);  
  
    } catch (Exception e) {  
  
        thrownew SeguridadesExpcion(Mensajes.ERROR_RECUPERAR);  
  
    }  
  
}  
  
  
/**  
 * Obtiene todos los registros de la tabla  
 *  
 * @return  
 */
```



```
@SuppressWarnings({ "unchecked", "rawtypes" })  
  
public List<T> buscarTodos() throws SeguridadesExpcion {  
  
    try {  
  
        CriteriaQuery cq = em.getCriteriaBuilder().createQuery();  
  
        cq.select(cq.from(tipoEntidad));  
  
        returnem.createQuery(cq).getResultList();  
  
    } catch (Exception e) {  
  
        thrownew SeguridadesExpcion(Mensajes.ERROR_RECUPERAR);  
  
    }  
  
}  
  
/**  
 * Crea un logger  
 */  
  
@PostConstruct  
  
public void setLogger() {  
  
    if (LOG == null) {  
  
        LOG = Logger.getLogger(tipoServicio);  
  
    }  
  
}
```

edu.uce.ceu.servicios

Este paquete se encuentra las clases hijas que implementas del Servicio Base y además tenemos las reglas de negocio que son diferentes para cada entidad.



```
edu.uce.ceu.servicios
├── ServicioAlimento.java
├── ServicioAnimal.java
├── ServicioAnimalParto.java
├── ServicioAnimalPorCategoria.java
├── ServicioCabeCeraVenta.java
├── ServicioCategoria.java
├── ServicioCliente.java
├── ServicioConsumoProduccion.java
├── ServicioDestino.java
├── ServicioDetalleVenta.java
├── ServicioDificultadParto.java
├── ServicioEvento.java
├── ServicioInseminacion.java
├── ServicioLogError.java
├── ServicioMuerte.java
├── ServicioParto.java
├── ServicioPerfil.java
├── ServicioPerfilRecurso.java
├── ServicioPerfilUsuario.java
├── ServicioPeso.java
├── ServicioProduccion.java
├── ServicioProducto.java
├── ServicioRecurso.java
├── ServicioSanitario.java
├── ServicioSemen.java
├── ServicioTipoAnimal.java
├── ServicioTipoPeso.java
├── ServicioTipoProducto.java
├── ServicioTipoVacuna.java
├── ServicioUnidad.java
├── ServicioUsuario.java
└── ServicioVacuna.java
└── ServicioViaAplicacion.java
```

ServicioTipoAnimal.java

```
package edu.uce.ceu.servicios;

import java.util.List;

import javax.ejb.Stateless;
import javax.persistence.Query;

import edu.uce.ceu.comun.ServicioBase;
import edu.uce.ceu.entidades.TipoAnimal;

@Stateless
public class ServicioTipoAnimal extends ServicioBase<TipoAnimal> {

    public ServicioTipoAnimal() {
```



```
super(TipoAnimal.class, ServicioTipoAnimal.class);

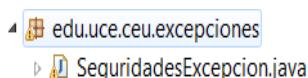
}

@SuppressWarnings("unchecked")
public List<TipoAnimal> busquedaTipoAnimalIndividuales() {
    Query consultaPorTipoAnimal = em
        .createQuery("SELECT tipoAnimal FROM TipoAnimal tipoAnimal
where tipoAnimal.estadoTipoAnimal ='Individuales'");
    return consultaPorTipoAnimal.getResultList();
}

@SuppressWarnings("unchecked")
public List<TipoAnimal> busquedaTipoAnimalGrupales() {
    Query consultaPorTipoAnimal = em
        .createQuery("SELECT tipoAnimal FROM TipoAnimal tipoAnimal
where tipoAnimal.estadoTipoAnimal ='Grupales'");
    return consultaPorTipoAnimal.getResultList();
}
}
```

Edu.uce.ceu.excepciones

Este paquete tiene el manejo de excepciones propias del proyecto que hereda de la clase padre Exception.



```
package edu.uce.ceu.excepciones;

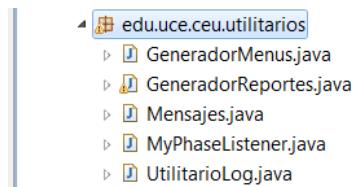
public class SeguridadesExpcion extends Exception {
```



```
public SeguridadesExcepcion(String mensaje) {  
    super(mensaje);  
}  
}
```

Edu.uce.ceu.utilitarios

En este paquete tiene las clases para optimizar el funcionamiento del proyecto.



En la clase UtilitarioLog.java tenemos el tipo de mensaje para capturar las excepciones.

```
package edu.uce.ceu.utilitarios;  
  
import javax.faces.application.FacesMessage;  
  
import javax.faces.context.FacesContext;  
  
  
public class UtilitarioLog {  
  
    public static void mostrarMensajeError(String mensajeError) {  
        FacesContext.getCurrentInstance()  
            .addMessage(  
                null,  
                new  
                    FacesMessage(FacesMessage.SEVERITY_ERROR,  
                        mensajeError, ""));  
    }  
  
    public static void mostrarMensajeInformacion(String mensajeInformacion) {  
        FacesContext.getCurrentInstance().addMessage(  
            null,
```



```
        null,  
  
        new FacesMessage(FacesMessage.SEVERITY_INFO,  
  
                         mensajeInformacion, ""));  
  
    }  
  
}
```

En la clase Mensajes.java está el mensaje a enviar al usuario.

```
package edu.uce.ceu.utilitarios;  
  
public class Mensajes {  
  
    public static final String ERROR_EDITAR = "NO SE PUEDE CARGAR LOS DATOS PARA EDITAR";  
  
    public static final String GUARDAR = "Los Datos se Guardaron Correctamente";  
  
    public static final String GUARDAR_PARTO = "El Parto se Guardó Corretamente Registre sus Hijos";  
  
    public static final String MODIFICAR = "Los Datos fueron Modificado exitosamente";  
  
    public static final String ELIMINAR = "Se eliminaron los datos exitosamente";  
  
    public static final String AGREGAR = "Se agrego los datos exitosamente";  
  
    public static final String VERIFICAR_STOCK = "No se puede vender mas productos de los existentes";  
  
    public static final String ERROR_GUARDAR = "NO SE PUDO GUARDAS LOS DATOS";  
  
    public static final String ERROR_ELIMINAR = "NO SE PUEDE ELIMINAR";  
  
    public static final String ERROR_RECUPERAR = "NO SE ENCUENTRA LO SOLICITADO";  
  
    public static final String ERROR_GUARDAR_IDENTIFICACION = "ESA IDENTIFICACION YA FUE ASIGNADA";  
  
    public static final String ERROR_GUARDAR_TIPO_ANIMALES_DUPLICADOS = "ESA TIPO DE ANIMAL YA EXISTE";  
  
    public static final String ERROR = "HA OCURRIDO UN ERROR INESPERADO";  
  
    public static final String ERROR_DIGITAR = "DIGITE EL USUARIO POR FAVOR";  
  
    public static final String ERROR_AUTENTICAR = "CLAVE INCORRECTA";  
  
    public static final String ERROR_BUSCAR = "NO EXISTE DATOS";  
  
    public static final String ALVERTENCIA = "NO EXISTE DATOS";  
  
    public static final String ERROR_CONVERSION = "No se puede convertir los datos";  
  
    public static final String ERROR_AGGREGACION = "No se agregaron los datos";  
  
}
```



La clase MyPhaseListener.java permite ver el ciclo de vida de los jsf.

```
package edu.uce.ceu.utilitarios;

import javax.faces.event.*;

public class MyPhaseListener implements PhaseListener

{

    private static final long serialVersionUID = 1L;

    public MyPhaseListener()

    {

    }

    public void afterPhase(PhaseEvent event)

    {

        System.out.println("new      StringBuilder("           Despues     de     la     fase-->
").append(event.getPhaseId().toString()).append("--Vista
").append(event.getFacesContext().getViewRoot().getViewId().toString());

        if(event.getPhaseId() == PhaseId.RENDER_RESPONSE)

        {

            System.out.println("*****Peticion Procesada!*****");

            System.out.println("<<<<<<<<");

            System.out.println("");

        }

    }

    public void beforePhase(PhaseEvent event)

    {

        if(event.getPhaseId() == PhaseId.RESTORE_VIEW)

        {

            System.out.println("");

            System.out.println(" >>>>>>>>>");

            System.out.println("****  Procesando una nueva Peticion  ****");

        }

    }

}
```



```
}

System.out.println((new StringBuilder("          Antes      de      la      fase-->
").append(event.getPhaseId().toString()).toString()));

}

public PhaseId getPhaseId()

{

return PhaseId.ANY_PHASE;

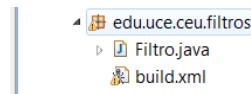
}

}
```

Para el menú dinámico de usuarios, perfiles y permiso se tiene:

Edu.uce.edu.filtros

En este paquete esta la clase Filtro.java que nos permite filtrar las páginas cuando el usuario tiene acceso al sistema.



Filtro.java

```
packageedu.uce.ceu.filtros;

import java.io.IOException;

import java.util.List;

import javax.servlet.Filter;

import javax.servlet.FilterChain;

import javax.servlet.FilterConfig;

import javax.servlet.ServletException;

import javax.servlet.ServletRequest;

import javax.servlet.ServletResponse;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;
```



```
import javax.servlet.http.HttpSession;

import edu.uce.ceu.controladores.DatosUsuario;

public class Filtro implements Filter {

    @Override
    public void destroy() {
        // TODO Auto-generated method stub
    }

    @Override
    public void doFilter(ServletRequest request, ServletResponse response,
                         FilterChain chain) throws IOException, ServletException {
        HttpServletRequest req = (HttpServletRequest) request;
        HttpServletResponse resp = (HttpServletResponse) response;

        // Para acceder a la sesión del usuario
        HttpSession ses = req.getSession(true);

        // Para saber que ingresó al filtro
        System.out.println("paso por el filtro");
        // Valor de la atributo usuario.... Este valor se colocará en sesión
        // cuando el usuario se loguea exitosamente
        System.out.println("Pagina solicitada: " + req.getRequestURI());
        DatosUsuario du = (DatosUsuario) ses.getAttribute("datosUsuario");
        boolean exitoso = false;
        // Si el usuario ya está logueado
        if (ses.getAttribute("datosUsuario") != null) {
            List<String> pags = du.getPaginas();
            if (pags != null) {
                for (String pag : pags) {
                    if (pag.equals(req.getRequestURI())) {

```



```
System.out.println("pase nomas ya esta
logueado");

exitoso = true;
chain.doFilter(request, response);

}

}

}

if (!exitoso) {

System.out

.println("Pagina de error, logueado pero
sin permiso para la página");

resp.sendRedirect(req.getContextPath()
+ "/RecursoNoPermitido.jsf");

}

}

else {

// Si el usuario no estálogueado, redirecciona la petición a la
// página index.jsp

System.out.println("De regreso al index, no esta logueado");

resp.sendRedirect(req.getContextPath() + "/index.html");

}

}

@Override
public void init(FilterConfig arg0) throws ServletException {
// TODO Auto-generated method stub
}
}
```



DatosUsuario.java es una clase manager bean por ello las anotaciones **@ManagedBean** que indica que es una clase manejadora, **@SessionScoped** que además el ámbito de la variable es de sesión ésta permite guardar y navegar al usuario desde otras páginas.

```
package edu.uce.ceu.controladores;

import java.util.List;

import javax.ejb.EJB;
import javax.faces.bean.ManagedBean;
import javax.faces.bean.SessionScoped;
import org.primefaces.model.MenuModel;

import edu.uce.ceu.entidades.Perfil;
import edu.uce.ceu.entidades.Usuario;
import edu.uce.ceu.servicios.ServicioPerfil;

@ManagedBean
@SessionScoped
public class DatosUsuario {

    private List<String> paginas;
    private List<Perfil> perfiles;
    private MenuModel menu;
    private Usuario usuario;
    private String nombresLoggin;

    private Perfil perfil;

    @EJB
    private ServicioPerfil servPerfil;

    public Perfil getPerfil() {
        return perfil;
    }
}
```



```
public void setPerfil(Perfil perfil) {  
    this.perfil = perfil;  
}  
  
public List<String> getPaginas() {  
    return paginas;  
}  
  
public void setPaginas(List<String> paginas) {  
    this.paginas = paginas;  
}  
  
public List<Perfil> getPerfiles() {  
    return perfiles;  
}  
  
public void setPerfiles(List<Perfil> perfiles) {  
    this.perfiles = perfiles;  
}  
  
public MenuModel getMenu() {  
    return menu;  
}  
  
public void setMenu(MenuModel menu) {  
    this.menu = menu;  
}  
  
public Usuario getUsuario() {  
    return usuario;  
}  
  
public void setUsuario(Usuario usuario) {
```



```
this.usuario = usuario;
//Recupera el perfil del usuario
perfil=servPerfil.buscarPerfilDeUsuario(usuario.getCodigoUsuario());  
  
}  
  
public String getNombresLoggin() {
    nombresLoggin=      "Cerrar      Session      "+usuario.getNombres()      +""
    "+usuario.getApellidoPaterno();
    returnnombresLoggin;
}  
  
public void setNombresLoggin(String nombresLoggin) {
    this.nombresLoggin = nombresLoggin;
}  
  
}  

```

ContenedorPerfiles.java

En esta clase recupera todos los perfiles de la base de datos y los guarda, esta se crea una sola vez porque su ámbito es de aplicación.

Las anotaciones: **@ManagedBean** indica que es una clase manager bean, **@ApplicationScoped** el ámbito de la variable es de aplicación, es decir que se crea solo cuando corre la aplicación.

```
package edu.uce.ceu.controladores;

import java.util.HashMap;
import java.util.List;
import java.util.Map;
import javax.annotation.PostConstruct;
import javax.ejb.EJB;
import javax.faces.bean.ApplicationScoped;
import javax.faces.bean.ManagedBean;
```



```
import edu.uce.ceu.comun.ControladorBase;
import edu.uce.ceu.entidades.Perfil;
import edu.uce.ceu.entidades.Recurso;
import edu.uce.ceu.excepciones.SeguridadesExpcion;
import edu.uce.ceu.servicios.ServicioPerfil;
import edu.uce.ceu.servicios.ServicioPerfilRecurso;
import edu.uce.ceu.utilitarios.Mensajes;
import edu.uce.ceu.utilitarios.UtilitarioLog;

@ManagedBean
@ApplicationScoped
public class ContenedorPerfiles {

    @EJB
    private ServicioPerfilRecurso servPerfilRecurso;

    @EJB
    private ServicioPerfil servPerfiles;

    // laclavees el id delperfil
    private Map<Integer, List<Recurso>> recursosPadre;

    public Map<Integer, List<Recurso>> getRecursosPadre() {
        return recursosPadre;
    }

    // laclavees el id delperfil
    private Map<Integer, List<Recurso>> recursosNivel1;

    private ControladorBase controladorBase = new ControladorBase();

    @PostConstruct
    public void init() {
        //cargarRecursos();
    }
}
```



```
public void cargarRecursos() {  
    try {  
        List<Perfil> todosPerfiles;  
  
        todosPerfiles = servPerfiles.buscarTodos();  
  
        List<Recurso> recursosPadresUnPerfil;  
        List<Recurso> recursosNivel1UnPerfil;  
        recursosPadre = new HashMap<Integer, List<Recurso>>();  
        recursosNivel1 = new HashMap<Integer, List<Recurso>>();  
  
        // Por cada perfil recuperar los padres  
        for (Perfil perfil : todosPerfiles) {  
            recursosPadresUnPerfil = servPerfilRecurso  
                .buscarMenusPadre(perfil.getCodigoPerfil());  
            recursosPadre.put(perfil.getCodigoPerfil(),  
                recursosPadresUnPerfil);  
        }  
  
        System.out.println("recursosPadres: " + recursosPadre.size());  
        // Por cada perfil recuperar los padres  
        for (Perfil perfil : todosPerfiles) {  
            recursosNivel1UnPerfil = servPerfilRecurso  
                .buscarSubmenus(perfil.getCodigoPerfil());  
            recursosNivel1.put(perfil.getCodigoPerfil(),  
                recursosNivel1UnPerfil);  
        }  
  
        System.out.println("recursosNivel1: " + recursosNivel1.size());  
        // Por cada perfil recuperar los hijos  
    } catch (SeguridadesExpcion e) {  
        UtilitarioLog.mostrarMensajeError(Mensajes.ERROR_GUARDAR);  
    }  
}
```



```
controladorBase.guardarError(Mensajes.ERROR_GUARDAR);

} catch (Exception e) {

    UtilitarioLog.mostrarMensajeError(Mensajes.ERROR_RECUPERAR);

    controladorBase.guardarError(Mensajes.ERROR_RECUPERAR);

}

}

public List<Recurso> recuperarRecursosNivel1(int codigoPerfil) {

    return recursosNivel1.get(codigoPerfil);

}

public List<Recurso> recuperarRecursosPadre(int codigoPerfil) {

    return recursosPadre.get(codigoPerfil);

}

}
```

GeneradorMenus.java esta clase se genera el menú dinámico en función al perfil de usuario.

```
package edu.uce.ceu.utilitarios;

import java.util.List;

import org.primefaces.component.menuitem.MenuItem;

import org.primefaces.component.submenu.Submenu;

import org.primefaces.model.DefaultMenuModel;

import org.primefaces.model.MenuModel;

import edu.uce.ceu.constantantes.Constantes;

import edu.uce.ceu.entidades.Recurso;

public class GeneradorMenus {

    public static MenuModel generarMenu(List<Recurso> recursosPadre,
                                         List<Recurso> recursosHijos) {
        // Por cada recurso padre, un submenu
        MenuModel model = new DefaultMenuModel();
```



```
for (Recurso recurso : recursosPadre) {  
    Submenu sm = newSubmenu();  
    sm.setLabel(recurso.getDescripcion());  
    //sm.setId(recurso.getCodigoRecurso()+"");  
    model.addSubmenu(sm);  
}  
  
//Por cada recurso hijo, secrea un menuItem, este se coloca en el submenu correcto  
  
for(Recurso recurso:recursosHijos){  
    MenuItem menuItem = newMenuItem();  
    menuItem.setValue(recurso.getDescripcion());  
    //Retirar el contextPath, se usa la constante  
CONTEXT_PATH en la cual se tiene el nombre del  
//proyecto.  
  
menuItem.setUrl(recurso.getPagina().substring(Constantes.CONTEXT_PATH.length()));  
menuItem.setStyle("color: #000000;font-weight:bold;font-size:10px;font-family:Verdana;");  
Submenu sm=buscarPadre(recurso.getRecursoPadre().getDescripcion(),  
model);  
if(sm!=null){  
    sm.getChildren().add(menuItem);  
}  
else{  
    System.out.println("No se encontro el parent: "+recurso.getRecursoPadre().getCodigoRecurso());  
}  
return model;  
}  
  
private static Submenu buscarPadre(String labelPadre,MenuModel model){  
List<Submenu> submenus=model.getSubmenus();  
for( Submenu submenu:submenus){  
if(submenu.getLabel().equals(labelPadre)){  
return submenu;  
}  
}
```



```
    }

}

returnnull;

}

}
```

MenuUsuarioControlador.java esta clase permite pintar en la página los menús.

```
package edu.uce.ceu.menu;

import javax.annotation.PostConstruct;

@ManagedBean
@SessionScoped

publicclass MenuUsuarioControlador {

    @ManagedProperty(value = "#{generadorMenuControlador}")

    private GeneradorMenuControlador generadorMenu;

    private MenuModel menuUsuario;

    public MenuUsuarioControlador(){

    }

    @PostConstruct
    publicvoid init(){

        menuUsuario=generadorMenu.recuperarMenu(1);

    }

    public MenuModel getMenuUsuario() {

        returnmenuUsuario;

    }

    publicvoid setMenuUsuario(MenuModel menuUsuario) {

        this.menuUsuario = menuUsuario;

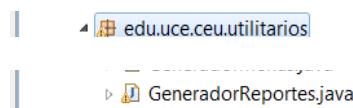
    }

    public GeneradorMenuControlador getGeneradorMenu() {
```



```
returngeneradorMenu;  
}  
  
publicvoid setGeneradorMenu(GeneradorMenuControlador generadorMenu) {  
    this.generadorMenu = generadorMenu;  
}  
}
```

Edu.uce.ceu.utilitarios



GeneradorReportes.java permite generar reportes previamente establecidos con JasperReport y presentarlos en pdf.

Esta clase utiliza un archivo recursos.properties, que se encuentra dentro del directorio Reportes, al ejecutarse la aplicación, este directorio Reportes deberá colocarse en la carpeta \$JBOSS_HOME/modules (en Jboss7). En el archivo recursos.properties se puede definir el path en el cual se van a ubicar los archivos .jasper y .jrxml

```
package edu.uce.ceu.utilitarios;  
  
import java.io.FileInputStream;  
  
import java.io.FileNotFoundException;  
  
import java.io.IOException;  
  
import java.sql.Connection;  
  
import java.sql.SQLException;  
  
import java.util.HashMap;  
  
import java.util.Map;  
  
import java.util.Properties;  
  
import javax.annotation.PostConstruct;  
  
import javax.annotation.Resource;
```



```
import javax.faces.bean.ManagedBean;  
  
import javax.faces.bean.SessionScoped;  
  
import javax.faces.context.FacesContext;  
  
import javax.servlet.ServletOutputStream;  
  
import javax.servlet.http.HttpServletResponse;  
  
import javax.sql.DataSource;  
  
import net.sf.jasperreports.engine.JRException;  
  
import net.sf.jasperreports.engine.JREporterParameter;  
  
import net.sf.jasperreports.engine.JasperExportManager;  
  
import net.sf.jasperreports.engine.JasperFillManager;  
  
import net.sf.jasperreports.engine.JasperPrint;  
  
import net.sf.jasperreports.engine.export.oasis.JR0dtExporter;  
  
import net.sf.jasperreports.engine.export.ooxml.JRDocxExporter;  
  
import net.sf.jasperreports.engine.export.ooxml.JRDocxExporterParameter;  
  
import net.sf.jasperreports.engine.export.ooxml.JRPptxExporter;  
  
import net.sf.jasperreports.engine.export.ooxml.JRXlsxExporter;
```

```
@ManagedBean  
  
@SessionScoped  
  
public class GeneradorReportes {  
  
    /**  
     * Clase Jasper para generar el reporte  
     */  
  
    private JasperPrint jasperPrint;  
  
    /**  
     * Mapa de parámetros que se van a pasar al reporte  
     */  
  
    private Map<String, Object> parametrosReporte = new HashMap();  
  
    /**  
     * NOMBRE del archivo .jasper que genera el reporte  
     */  
  
    private String nombreJasper;  
  
    /**  
     * Path en el cuál se van a guardar los archivos .jasper y .jrxml  
     */
```



```
* reporte. Este path lo lee delarchivo reportes.properties
*/
private String path;
/***
 * Ruta del archivo reportes.properties. Es un path relativo a la carpeta
 * bin del Jboss
*/
private final String ARCHIVO_CONFIGURACION =
"C:\\\\Users\\\\Narcisa\\\\Reportes\\\\reportes.properties";

/**
 * Nombre con el cual se genera el archivo de reporte en los diferentes
 * formatos
*/
private String nombreReporte;
/***
 * DataSource que se lo obtiene del servidor de aplicaciones
*/
@Resource(name = "java:/CEUDS")
private DataSource dataSource;

/**
 * Constructor. Carga el archivo de propiedades resource.properties y lee la
 * propiedad path
*/
public GeneradorReportes() {

    Properties props = newProperties();
    try {
        props.load(new FileInputStream(ARCHIVO_CONFIGURACION));
        path = props.getProperty("path");
    } catch (FileNotFoundException e) {
```



```
// TODO Auto-generated catch block
e.printStackTrace();
} catch (IOException e) {
// TODO Auto-generated catch block
e.printStackTrace();
}

}

/**
 * Obtiene la conexión a la base y genera el reporte
 *
 * @throws Exception
 *           si no pudo conectarse a la base de datos o no pudo generar el
 *           reporte
 */
public void generarReporte() throws Exception {
    // JRBeanCollectionDataSource beanCollectionDataSource=new
    // JRBeanCollectionDataSource(listaEmpleados);
    Connection conn = null;
    try {
        conn = dataSource.getConnection();
        // Class.forName("org.postgresql.Driver");
        //
        // conn = DriverManager
        // .getConnection("jdbc:postgresql://localhost:5432/test",
        // "postgres", "root");
        conn.setAutoCommit(false);
        //
path="C:\\CEU\\workspaceCEU\\ProyectoJSFCEU\\JavaSource\\edu\\uce\\ceu\\reportesCE
U";
System.out.println("****path****"+path);
jasperPrint = JasperFillManager.fillReport(path + "\\"

```



```
+ nombreJasper, parametrosReporte, conn);  
}  
} catch (SQLException e) {  
  
    System.out.println("Error de conexión: " + e.getMessage());  
  
    e.printStackTrace();  
  
    thrownew Exception("Error al conectarse a la Base de Datos");  
  
}  
  
}  
  
}  
  
/**  
  
 * Invoca a la generación del reporte y genera el Stream con la información  
 * del reporte que será devuelto a la página  
 *  
 * @param extension  
 *          extensión del archivo a generarse (pdf, docx, xlsx. odt. pptx)  
 * @return el Stream con la información del reporte que será devuelto a la  
 *         página  
 * @throws Exception  
 *          si no logró generar el reporte  
 */  
  
  
public ServletOutputStream generarResponse(String extension)  
  
    throws Exception {  
  
    generarReporte();  
  
    HttpServletResponse httpServletResponse = (HttpServletResponse)  
FacesContext  
  
        .getCurrentInstance().getExternalContext().getResponse();  
  
    httpServletResponse.addHeader("Content-disposition",  
        "attachment; filename=" + nombreReporte + "."+ extension);  
  
    ServletOutputStream servletOutputStream = null;  
  
    try {  

```



```
        servletOutputStream = httpServletResponse.getOutputStream();

    } catch (IOException e) {
        e.printStackTrace();
    }

    return servletOutputStream;
}

/**
 * Retorna el reporte en formato PDF con el Stream que obtiene de
 * generarResponse
 *
 * @throws Exception
 *           si no logra generar el reporte
 */

public void generarPDF() throws Exception {
    ServletOutputStream servletOutputStream = generarResponse("pdf");
    JasperExportManager.exportReportToPdfStream(jasperPrint,
        servletOutputStream);
}

// getters y setters

public String getNombreJasper() {
    return nombreJasper;
}

public void setNombreJasper(String nombreJasper) {
    this.nombreJasper = nombreJasper;
}

public String getPath() {
    return path;
}

public void setPath(String path) {
```



```
this.path = path;  
}  
  
public Map getParametrosReporte() {  
    returnparametrosReporte;  
}  
  
public void setParametrosReporte(Map parametrosReporte) {  
    this.parametrosReporte = parametrosReporte;  
}  
  
public String getNombreReporte() {  
    returnnombreReporte;  
}  
  
public void setNombreReporte(String nombreReporte) {  
    this.nombreReporte = nombreReporte;  
}  
}
```

Edu.uce.ceu.reportesControladores

En este paquete están las clases que permite mandar los parámetros del reporte y exportarlos a .pdf en el cual ponemos el nombre del reporte realizado en JasperReport.





FichaAnimalReporteControlador.java

```
package edu.ucec.ceu.reportesControladores;

import java.io.File;
import java.util.Date;
import java.util.HashMap;
import java.util.Map;
import javax.faces.application.FacesMessage;
import javax.faces.bean.ManagedBean;
import javax.faces.bean.ManagedProperty;
import javax.faces.bean.ViewScoped;
import javax.faces.context.FacesContext;
import javax.faces.event.ActionEvent;

import edu.uce.ceu.utilitarios.GeneradorReportes;

@ManagedBean
@ViewScoped
public class FichaAnimalReporteControlador {

    @ManagedProperty("#{generadorReportes}")
    private GeneradorReportes generadorJasper;

    private String numAC;

    private final String NOMBRE_Reporte_JASPER = "report_ficha_animal.jasper";
    private final String NOMBRE_Reporte = "Ficha Animal";

    public String getNumAC() {
        return numAC;
    }

    public void setNumAC(String numAC) {
```



```
this.numAC = numAC;
}

public GeneradorReportes getGeneradorJasper() {
    return generadorJasper;
}

public void setGeneradorJasper(GeneradorReportes generadorJasper) {
    this.generadorJasper = generadorJasper;
}

public String getNOMBRE_REPORTE_JASPER() {
    return NOMBRE_REPORTE_JASPER;
}

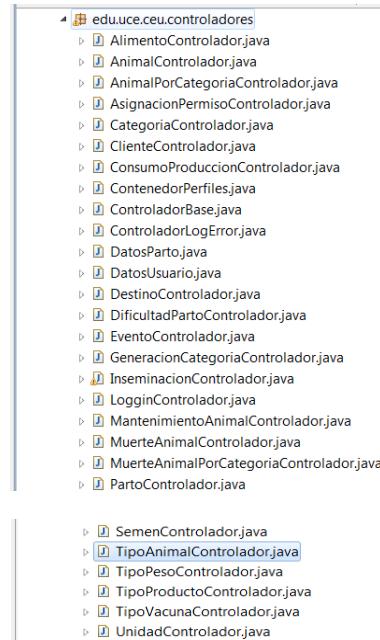
public String getNOMBRE_REPORTE() {
    return NOMBRE_REPORTE;
}

public void exportarPdf(ActionEvent e) {
    try {
        //para saber donde se crean los archivos por defecto
        File a=newFile("archivo1");
        System.out.println("PATH ....."+a.getAbsolutePath());
        Map parametros = newHashMap();
        generadorJasper.setNombreJasper(NOMBRE_REPORTE_JASPER);
        generadorJasper.setNombreReporte(NOMBRE_REPORTE);
        parametros.put("numArete", numAC);
        generadorJasper.setParametrosReporte(parametros);
        generadorJasper.generarPDF();
    } catch (Exception ex) {
        FacesContext.getCurrentInstance().addMessage(
            null,
            new FacesMessage(FacesMessage.SEVERITY_ERROR, ex
                .getMessage(), ""));
    }
}
```



Edu.uce.ceu.controladores

En este paquete tenemos las clases java que actúan como clases manejadoras que interactúan directamente con la página xhtml.



TipoAnimalsControlador.java

```
package edu.uce.ceu.controladores;

import java.util.List;

import javax.annotation.PostConstruct;

import javax.ejb.EJB;

import javax.faces.bean.ManagedBean;

import javax.faces.bean.ViewScoped;

import edu.uce.ceu.entidades.TipoAnimal;

import edu.uce.ceu.excepciones.SeguridadesExpcion;

import edu.uce.ceu.servicios.ServicioTipoAnimal;

import edu.uce.ceu.utilitarios.Mensajes;

import edu.uce.ceu.utilitarios.UtilitarioLog;
```

@ManagedBean



```
@ViewScoped

public class TipoAnimalControlador {

    //atributos

    private TipoAnimal tipoAnimal;
    private List<TipoAnimal> tiposAnimales;

    //servicios

    @EJB
    private ServicioTipoAnimal servicioTipoAnimal;

    //constructores

    public TipoAnimalControlador(){
        tipoAnimal= new TipoAnimal();
    }

    //PostConstruct
    @PostConstruct
    public void cargarTabla(){
        try {
            tiposAnimales= servicioTipoAnimal.buscarTodos();
        } catch (SeguridadesExpcion e) {
            UtilitarioLog.mostrarMensajeInformacion(Mensajes.ERROR_BUSCAR);
            e.printStackTrace();
        }
    }

    //Gets and Sets

    public TipoAnimal getTipoAnimal() {
        return tipoAnimal;
    }

    public void setTipoAnimal(TipoAnimal tipoAnimal) {
        this.tipoAnimal = tipoAnimal;
    }
}
```



```
public List<TipoAnimal> getTiposAnimales() {  
    return tiposAnimales;  
}  
  
public void setTiposAnimales(List<TipoAnimal> tiposAnimales) {  
    this.tiposAnimales = tiposAnimales;  
}  
  
//metodos  
  
public String guardarTipoAnimal(){  
    try {  
        servicioTipoAnimal.insertar(tipoAnimal);  
        tiposAnimales= servicioTipoAnimal.buscarTodos();  
        UtilitarioLog.mostrarMensajeInformacion(Mensajes.GUARDAR);  
    } catch (SeguridadesExpcion e) {  
        // TODO Auto-generated catch block  
        UtilitarioLog.mostrarMensajeError(Mensajes.ERROR_GUARDAR);  
  
        UtilitarioLog.mostrarMensajeError(Mensajes.ERROR_GUARDAR_TIPO_ANIMALES_DUPLICADOS);  
    };  
    e.printStackTrace();  
}  
  
    tipoAnimal= new TipoAnimal();  
  
    return"";  
}  
  
public String eliminarTipoAnimal(){  
    try {  
        servicioTipoAnimal.eliminar(tipoAnimal);  
        tiposAnimales= servicioTipoAnimal.buscarTodos();  
        UtilitarioLog.mostrarMensajeInformacion(Mensajes.ELIMINAR);  
    } catch (SeguridadesExpcion e) {  
        UtilitarioLog.mostrarMensajeError(Mensajes.ERROR_ELIMINAR);  
        e.printStackTrace();  
    }  
}
```



```
tipoAnimal= new TipoAnimal();

return "";
}

public String actualizarTipoAnimal(){

try {

    servicioTipoAnimal.actualizar(tipoAnimal);

    tiposAnimales= servicioTipoAnimal.buscarTodos();

    UtilitarioLog.mostrarMensajeInformacion(Mensajes.MODIFICAR);

} catch (SeguridadesExcepcion e) {

    UtilitarioLog.mostrarMensajeError(Mensajes.ERROR_EDITAR);

    e.printStackTrace();

}

tipoAnimal= new TipoAnimal();

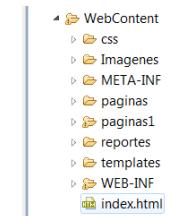
return "";
}

}
```

Dentro de la Segunda parte tenemos:

Webcontent

En esta carpeta encontramos la interfaz grafica. Con la que el usuario podrá interactuar con el sistema.





Index.html En esta página que encuentra dentro del WebContent, es la principal. Al ejecutar el sistema es la primera que se activa y nos redirecciona a la página LogginUsuario.jsf

```
<html>
<head>
<metahttp-equiv="Refresh"content="0; URL=LogginUsuario.jsf"/>
</head>
</html>
```

LogginUsuario.xhtml

```
<!DOCTYPE html PUBLIC "-//W3C//DTD Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:h="http://java.sun.com/jsf/html"
      xmlns:f="http://java.sun.com/jsf/core"
      xmlns:ui="http://java.sun.com/jsf/facelets"
      xmlns:p="http://primefaces.org/ui">

<ui:composition template="/templates/template2.xhtml">
    <ui:define name="tituloPagina">Perfiles</ui:define>
    <ui:define name="contenido">
        <h:form id="formIngresoDatos">
            <p:panel id="pnLListaUsuarios">
                <f:facet name="header">
                    <h:outputText value="Ingreso al Sistema" styleClass="panelTitulo"/>
                </f:facet>
                <p:messages/>
                <h:panelGrid columns="2">
                    <h:outputText value="Usuario:" styleClass="texto"/>

```



```
<p:inputText value="#{LogginControlador.usuario.identificacion}" styleClass="texto"/>

<h:outputText value="Password:<br/>
" styleClass="texto"/>

<p:password value="#{LogginControlador.usuario.clave}" feedback="false"/>

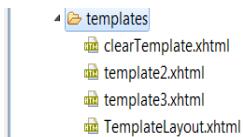
<h:inputHidden value="#{generacionCategoriaControlador.a}" />
</h:panelGrid>

<p:commandButton value="OK" action="#{LogginControlador.validarUsuario}" ajax="false"/>

</p:panel>

</h:form>
</ui:define>
</ui:composition>
</html>
```

En la carpeta template, encontramos las plantillas que utilizamos para el sistema.



ClearTemplate.xhtml

Esta plantilla es la más utilizada dentro del sistema .



```
<!DOCTYPE html PUBLIC "-//W3C//DTD Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:ui="http://java.sun.com/jsf/facelets"
      xmlns:h="http://java.sun.com/jsf/html"
      xmlns:f="http://java.sun.com/jsf/core"
      xmlns:p="http://primefaces.prime.com.tr/ui">

    <h:head>
        <title><ui:insert name="tituloPagina">Coloque el titulo de la pagina</ui:insert>
        </title>
        <link rel="stylesheet" type="text/css"
              href="#{request.contextPath}/css/base.css"/>
    </h:head>
    <h:body>
        <link rel="stylesheet" type="text/css"
              href="#{request.contextPath}/css/estilo.css"/>
        <p:layout fullPage="true">
            <p:layoutUnit position="north" size="150" header="" gutter="0" style="background=blue;">
                <h:form>
                    <h:panelGrid columns="2">
                        <h:graphicImage value="/Imagenes/Logo3.jpg"/>
                        <h:panelGroup>
                            <h:panelGrid columns="1">
                                <h:graphicImage value="/Imagenes/cerrarSession.jpg"/>
                            </h:panelGrid>
                        </h:panelGroup>
                    </h:panelGrid>
                </h:form>
            </p:layoutUnit>
            <p:commandLink value="#{datosUsuario.nombresLogin}"
                           action="/LogginUsuario.jsf" ajax="false"/>
        </p:layout>
    </h:body>

```



```
</h:panelGrid>

</h:form>

</p:layoutUnit>

<!--

<p:layoutUnit position="south" size="100" header="Bottom"
    resizable="true" closable="true" collapsible="true">
    <h:outputText value="Contenido Inferior" />
</p:layoutUnit>

-->

<p:layoutUnit position="west" size="200" resizable="true" closable="true" collapsible="true">
    <tr style="margin-top:-24px;color:yellow;font-size:15px;font-family:Verdana;background:red;">
        <h:form>
            <p:panel>

<p:menu model="#{datosUsuario.menu}"/>
            </p:panel>
        </h:form>
    </tr>
</p:layoutUnit>

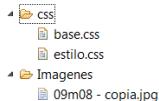
<p:layoutUnit position="east" size="200" header="CEU" resizable="true"
    closable="true" collapsible="true" effect="drop">
    <h:outputText value="Espacio para publicidad"/>
</p:layoutUnit>

<p:layoutUnit position="center">
    <tr>
        <td align="center" width="100%" valign="middle" class="formulario"
            style="font-family: Verdana, Arial, Helvetica, sans-serif; font-size: 20px;">
```



```
<ui:insertname="contenido">  
    <h:outputTextvalue="Inserte el CONTENIDO  
PRINCIPAL"/>  
    </ui:insert>  
</td>  
</tr>  
</p:layoutUnit>  
  
</p:layout>  
<ui:insertname="dialogos">  
  
</ui:insert>  
</h:body>  
</html>
```

La carpeta css tiene los estilos utilizados en la paginas xhtml del sistema y la carpeta Imágenes tiene las imágenes utilizadas en el sistema.



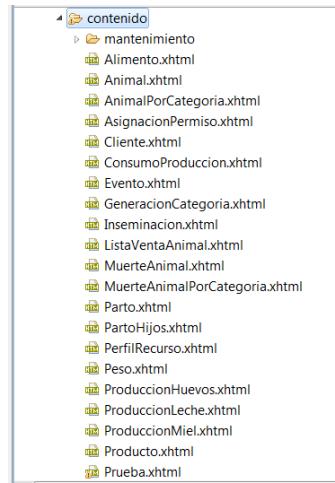
```
.texto{  
    font-family:Verdana,Arial,Helvetica,sans-serif;  
    font-size:12px;  
}  
  
.textoPequeno{  
    font-family:Verdana,Arial,Helvetica,sans-serif;  
    font-size:10px;  
}  
  
.textoSombreado{
```



```
font-family:Verdana,Arial,Helvetica,sans-serif;  
font-size:14px;  
color:#000000;font-weight:bold;  
  
}  
  
.LetrasMenuLink{  
    text-align:right;  
    font-size:15px;  
    color:#FFFFFF;  
    font-weight:bold;  
    font-family:Verdana;  
    padding-top:0;  
    padding:0px;  
    padding-bottom:0px;  
    border:0px;  
    text-decoration:none  
    background:pink;  
}  
  
.panelTitulo{  
    margin-top:-24px;  
    color:yellow;  
    font-size:15px;  
    font-family:Verdana;  
}  
  
.LetrasMenuLinkCRUD{  
    margin-left:30px;  
    margin-right:30px;  
    width:100%;  
    text-align:left;  
    font-size:12px;  
    font-family:Verdana;  
    height:15px;  
}
```



Dentro de la carpeta contenido se tiene las páginas de Ingreso de información del Centro Experimental Uyumbicho dependiendo de su funcionalidad.



A su vez dentro de esta carpeta se tiene la de mantenimiento que consta de las páginas que rara vez actualizan su información.

Categoría.xhtml

```
<!DOCTYPE html PUBLIC "-//W3C//DTD  
Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">  
  
<html xmlns="http://www.w3.org/1999/xhtml"  
      xmlns:h="http://java.sun.com/jsf/html"  
      xmlns:f="http://java.sun.com/jsf/core"  
      xmlns:ui="http://java.sun.com/jsf/facelets"  
      xmlns:p="http://primefaces.prime.com.tr/ui">  
  
<ui:composition template="/templates/clearTemplate.xhtml">  
    <ui:define name="tituloPagina">Registro de Categorías</ui:define>  
    <ui:define name="contenido">  
        <h:form id="formCategoria">  
            <p:panel>  
                <f:facet name="header">  
                    <h:outputText value="REGISTRO DE  
CATEGORÍAS" styleClass="panelTitulo"/>  
                </f:facet>  
                <p:outputPanel id="panelDatos">  
                    <h:panelGrid columns="2">
```



```
<h:outputTextvalue="Fecha:"styleClass="texto"/>
<p:calendarstyleClass="texto"
value="#{categoriaControlador.categoría.fechaCategoria}"
pattern="dd/MM/yyyy"/>

<h:outputTextvalue="Seleccione el Tipo
del Animal:"styleClass="texto"/>
<p:selectOneMenustyleClass="texto"
value="#{categoriaControlador.tipoAnimalSeleccionado}">
<f:selectItemsvalue="#{categoriaControlador.itemsTiposAnimales}"/>
</p:selectOneMenu>
<h:outputTextvalue="Nombre de La
Categoría:"styleClass="texto"/>
<p:inputTextstyleClass="texto"
value="#{categoriaControlador.categoría.nombreCategoria}"required="true"requiredMe
ssage="Nombre Obligatorio"/>
<h:outputTextvalue="Registrador
por:"styleClass="texto"/>

<h:outputTextvalue="#{datosUsuario.usuario}"styleClass="texto"/>
</h:panelGrid>
</p:outputPanel>
</p:panel>
<p:commandButtonvalue="Guardar"image="ui-icon ui-icon-disk"
update="tablaCategoria,panelDatos"oncomplete="confirmacion.show()"

action="#{categoriaControlador.guardarCategoria}"ajax="false"/>
<p:commandButtonvalue="Actualizar"icon="ui-icon ui-icon-refresh"
action="#{categoriaControlador.actualizarCategoria}"

oncomplete="confirmacion.show()"update="panelDatos,tablaCategoria"
ajax="false"/>
```



```
<p:dataTable id="tablaCategoria"  
  
value="#{categoriaControlador.categorias}" var="CategoriaItem"  
paginator="true" rows="4">  
  
<f:facet name="header">  
  
    <h:outputText value="TABLA DE  
CATEGORIAS" styleClass="panelTitulo"/>  
  
</f:facet>  
  
<p:column>  
  
    <f:facet name="header">  
  
        <h:outputText value="FECHA"/>  
  
</f:facet>  
  
<h:outputText value="#{CategoriaItem.fechaCategoria}" styleClass="texto"/>  
  
</p:column>  
  
<p:column filterBy="#{CategoriaItem.tipoAnimal.nombreTipoAnimal}">  
  
    <f:facet name="header">  
  
        <h:outputText value="Tipo Animal"/>  
  
</f:facet>  
  
<h:outputText value="#{CategoriaItem.tipoAnimal.nombreTipoAnimal}" styleClass="texto"/>  
/>  
  
</p:column>  
  
<p:column filterBy="#{CategoriaItem.nombreCategoria}">  
  
    <f:facet name="header">  
  
        <h:outputText value="Nombre Categorías"/>  
  
</f:facet>  
  
<h:outputText value="#{CategoriaItem.nombreCategoria}" styleClass="texto"/>  
  
</p:column>  
  
<p:column>  
  
    <f:facet name="header">  
  
        <h:outputText value="ACCIONES"/>  
  
</f:facet>
```



```
<p:commandButton value="Editar" icon="ui-icon ui-icon-pencil" />

update="formCategoria:panelDatos" process="formCategoria:tablaCategoria">
    <f:setPropertyActionListener
        target="#{categoriaControlador.categoría}"
        value="#{CategoriaItem}"/>
    <f:setPropertyActionListener
        value="#{CategoriaItem.tipoAnimal.idTipoAnimal}" />
    <target="#{categoriaControlador.tipoAnimalSeleccionado}"/>
    </p:commandButton>
</p:column>
<p:column>
    <p:commandButton value="Eliminar" icon="ui-icon ui-icon-close" />
    <oncomplete="confirmacion.show()" />
    <f:setPropertyActionListener value="#{CategoriaItem}" />
    <target="#{categoriaControlador.categoría}"/>
    </p:commandButton>
</p:column>
</p:dataTable>
</h:form>

<h:form id="fromGrowl">
    <p:growl id="growlMensajes">
        </p:growl>
    </h:form>
</ui:define>
<ui:define name="dialogos">
    <h:form id="formConfirmarCategoria">
```



```
<p:confirmDialogmessage="Está seguro que desea Eliminar?"styleClass="texto"

width="210"showEffect="explode"hideEffect="explode"
severity="alert"widgetVar="confirmacion">

<p:panel>

<h:panelGridcolumns="2">

<p:commandButtonvalue="Aceptar"

actionListener="#{categoriaControlador.eliminarCategoria}"

oncomplete="confirmacion.hide()"ajax="false"

action="/paginas1/mantenimiento/Categoría"

update="formCategoria:tablaCategoria,formCategoria:panelDatos"/>

<p:commandButtonvalue="Cancelar"oncomplete="confirmacion.hide()"/>

</h:panelGrid>

</p:panel>

</p:confirmDialog>

<p:confirmDialogmessage=""></p:confirmDialog>

</h:form>

</ui:define>

</ui:composition>

</html>
```

Dentro de la carpeta reportes se tiene las páginas que nos permite visualizar los reportes y exportar al formato .pdf





FichaAnimalReporte.xhtml

```
<!DOCTYPE html PUBLIC "-//W3C//DTD  
XHTML  
Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">  
  
<html xmlns="http://www.w3.org/1999/xhtml"  
  
      xmlns:h="http://java.sun.com/jsf/html"  
      xmlns:f="http://java.sun.com/jsf/core"  
      xmlns:ui="http://java.sun.com/jsf/facelets"  
      xmlns:p="http://primefaces.org/ui">  
  
  
<ui:composition template="/templates/template3.xhtml">  
  
    <ui:define name="tituloPagina">Perfiles</ui:define>  
  
    <ui:define name="contenido">  
  
        <h:form>  
  
            <h:outputText value="Ingrese Número Arete/Identificación:"/>  
  
            <p:inputText value="#{fichaAnimalReporteControlador.numAC}"/>  
            <h:commandButton value="Exportar a PDF"  
  
            actionListener="#{fichaAnimalReporteControlador.exportarPdf}"/>  
        </h:form>  
    </ui:define>  
  
</ui:composition>  
</html>
```

Dentro de la carpeta META-INF



Persistence.xml se maneja la unidad persistencia con Hibernate.

```
<?xml version="1.0" encoding="UTF-8"?>  
<persistenceversion="1.0"
```



```
xmlns="http://java.sun.com/xml/ns/persistence" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://java.sun.com/xml/ns/persistence
http://java.sun.com/xml/ns/persistence/persistence_1_0.xsd">

<persistence-unit name="cmd">
    <jta-data-source>java:/CEUDS</jta-data-source>
    <class>edu.uce.ceu.entidades.Alimento</class>
    <class>edu.uce.ceu.entidades.Animal</class>
    <class>edu.uce.ceu.entidades.AnimalParto</class>
    <class>edu.uce.ceu.entidades.AnimalPorCategoria</class>
    <class>edu.uce.ceu.entidades.CabeceraVenta</class>
    <class>edu.uce.ceu.entidades.Categoría</class>
    <class>edu.uce.ceu.entidades.Cliente</class>
    <class>edu.uce.ceu.entidades.ConsumoProducción</class>
    <class>edu.uce.ceu.entidades.Destino</class>
    <class>edu.uce.ceu.entidades.DetalleVenta</class>
    <class>edu.uce.ceu.entidades.DificultadParto</class>
    <class>edu.uce.ceu.entidades.Evento</class>
    <class>edu.uce.ceu.entidades.Inseminación</class>
    <class>edu.uce.ceu.entidades.LogError</class>
    <class>edu.uce.ceu.entidades.Muerte</class>
    <class>edu.uce.ceu.entidades.Parto</class>
    <class>edu.uce.ceu.entidades.Perfil</class>
    <class>edu.uce.ceu.entidades.PerfilRecurso</class>
    <class>edu.uce.ceu.entidades.PerfilUsuario</class>
    <class>edu.uce.ceu.entidades.Peso</class>
    <class>edu.uce.ceu.entidades.Producción</class>
    <class>edu.uce.ceu.entidades.Producto</class>
    <class>edu.uce.ceu.entidades.Raza</class>
    <class>edu.uce.ceu.entidades.Recurso</class>
    <class>edu.uce.ceu.entidades.Sanitario</class>
    <class>edu.uce.ceu.entidades.Semen</class>
    <class>edu.uce.ceu.entidades.TipoAnimal</class>
    <class>edu.uce.ceu.entidades.TipoPeso</class>
```



```
<class>edu.uce.ceu.entidades.TipoProducto</class>
<class>edu.uce.ceu.entidades.TipoVacuna</class>
<class>edu.uce.ceu.entidades.Unidad</class>
<class>edu.uce.ceu.entidades.Usuario</class>
<class>edu.uce.ceu.entidades.Vacuna</class>
<class>edu.uce.ceu.entidades.ViaAplicacion</class>

<properties>
    <property name="hibernate.dialect" value="org.hibernate.dialect.PostgreSQLDialect"/>
    <property name="hibernate.show_sql" value="true"/>
    <property name="hibernate.format_sql" value="true"/>
</properties>
</persistence-unit>

</persistence>
```

El archivo que permite la conexión a la base de datos es standalone que está en el servidor web JBoss 7.1 que es único y se añade un nuevo dataSource.

```
<datasource jndi-name="java:/CEUDS" pool-name="base_ceu" enabled="true" jta="false" use-ccm="false">
    <connection-url>
        jdbc:postgresql://localhost:5433/base_ceu
    </connection-url>
    <driver-class>
        org.postgresql.Driver
    </driver-class>
    <driver>
        postgresql-8.4-702.jdbc4.jar
    </driver>
    <security>
        <user-name>
```



```
postgres

</user-name>

<password>
estefania
</password>

</security>

<validation>

<validate-on-match>
false
</validate-on-match>

<background-validation>
false
</background-validation>

<background-validation-millis>
0
</background-validation-millis>

</validation>

<statement>

<prepared-statement-cache-size>
0
</prepared-statement-cache-size>

<share-prepared-statements>
false
</share-prepared-statements>

</statement>

</datasource>
```



ANEXO 4

DICCIONARIO DE DATOS

- **TABLA ANIMAL:** Se registra toda la información que tiene el animal

ANIMAL			
id_animal	<pi>	Serial	<M>
fecha_nacimiento		Date	
raza_padre		Variable characters (50)	
codigo_madre		Variable characters (30)	
raza_madre		Variable characters (50)	
raza_animal		Variable characters (50)	
sexo_animal		Variable characters (50)	
numero_registro		Integer	
foto_animal		Image	
numero_arete		Variable characters (50)	
nombre_registra_animal		Variable characters (50)	
estado_animal		Variable characters (50)	
edad_animal		Variable characters (50)	
id_animal <pi>			

ATRIBUTO	DESCRIPCIÓN	TIPO	LONGITUD	ME	MFIL	OTRA VALIDACIÓN
Id_animal	Es el código que identifica a la tabla animal	Serial				
Fecha_nacimiento	Es la fecha de nacimiento que se ingresa de cada animal	Date				
Raza_padre	Es el nombre de la raza del padre del animal	Variable characters	50			
Código_madre	Es el código que tiene la madre del animal	Variable characters	50			
raza_madre	En este campo se ingresa la raza de la madre	Variable characters	50			
Raza_animal	En este campo se ingresa la raza del animal	Variable characters	50			
Sexo_animal	En este campo ingresamos el sexo del animal	Variable characters	50			
Número_registro	En este campo ingresaos el numero de registro del animal	Integer				
Foto_animal	En este campo se ingresa la foto del animal	Image				
Numero_arete	Ingresamos el número de arete que tiene el animal	Variable characters	50			
Nombre_registra_animal	En este campo se ingresa el nombre que registra el animal	Variable characters	50			
Estado_animal	En este campo se ingresa el estado del animal	Variable characters	50			
Edad_animal	En este campo se registra la edad del animal	Variable characters	50			



- **TABLA PARTO:** Se ingresa toda la información de la madre bovina, o porcina al momento del parto.

PARTO		
id_parto	<pi>	Serial
fecha_parto		Date
num_hijos_nacidos		Integer
num_hijos_vivos		Integer
num_hijos_momificados		Integer
num_hijos_muertos		Integer
num_machos		Integer
num_hembras		Integer
nombre_registra_parto		Variable characters (50)
id_parto	<pi>	

ATRIBUTO	DESCRIPCIÓN	TIPO	LONGITUD	ME	MFILE	OTRA VALIDACIÓ N
Id_parto	Es el código que identifica a la tabla parto	Serial		X		
Fecha_parto	Es la fecha del parto de la madre	Date		X		
num_hijos_nacidos	En este campo se ingresa el número de hijos nacidos	Integer		X		
Num_hijos_momificados	En este campo ingresamos el número de hijos momificados en el vientre	Integer		X		
Num_hijos_muertos	En este campo se ingresa el número de hijos muertos al nacer	Integer		X		
Num_machos	En este campo se ingresa el número de hijos machos	Integer		X		
Num_hembras	En este campo se ingresa el número de hijos hembras	Integer		x		
Nombre_registra_parto	En este campo se ingresa el nombre que ingresa cada parto	Variable characters	50	X		

- **TABLA ANIMAL_PARTO:** Nace de la relación de muchos a muchos de la tabla animal y la tabla parto



ANIMAL_PARTO	
id_parto_animal <pi>	Serial <M>
num_parto	Integer
especificacion	Variable characters (50)
id_parto_animal <pi>	

ATRIBUTO	DESCRIPCIÓN	TIPO	LONGITUD	ME	MFIL	OTRA VALIDACIÓN
Id_parto_animal	Es el código que identifica a la tabla parto_animal	Serial		X		
Num_parto	En este campo se ingresa el número de parto que ha tenido el animal	Integer		X		
Especificación	Se ingresa las especificaciones del animal	Variable characters	50	X		

- **TABLA DIFICULTAD PARTO:** Se registra todas las dificultades de la madre del animal que en el momento de parir se presenta.

DIFICULTAD_PARTO	
id_dificultad_parto <pi>	Serial <M>
fecha_dificultad_parto	Date
nombre_dificultad_parto	Variable characters (50)
nombre_registra_dp	Variable characters (50)
id_dificultad_parto <pi>	

ATRIBUTO	DESCRIPCIÓN	TIPO	LONGITUD	ME	MFIL	OTRA VALIDACIÓN
Id_dificultad_parto	Es el código que identifica a la tabla id_dificultad_parto	Serial		X		
fecha_dificultad_parto	En este campo se ingresa la fecha que se presenta la dificultad	Integer		X		
Nombre_dificultad_parto	Se ingresa el nombre de la dificultad	Variable characters	50	X		
Nombre_registra_dificultad_parto	Se registra el nombre de la dificultad	Variable characters	50			

- **TABLA TIPO_ANIMAL:** Se registra los datos que tiene tipo animal



UNIVERSIDAD CENTRAL DEL ECUADOR
Sistema de Registro de Animales y sus Derivados del Centro Experimental Uyumbicho

TIPO_ANIMAL	
id_tipo_animal	<pi> Serial
nombre_tipo_animal	Variable charac
fecha_tipo_animal	Date
estado_tipo_animal	Variable charac

ATRIBUTO	DESCRIPCIÓN	TIPO	LONGITUD	ME	MFIL	OTRA VALIDACIÓN
Id_tipo_animal	Es el código que identifica a la tabla tipo_animal	Serial		X		
Nombre_tipo_animal	Se ingresa el nombre tipo de animal	Variable characters	50			
Fecha_tipo_animal	En este campo se ingresa el número de parto que ha tenido el animal	Date		X		
Estado_tipo_animal	Se ingresa el estado que tiene ese momento el tipo de animal	Variable characters	50	X		

- **RAZA:** En esta tabla se ingresa la información de la raza del animal y se relaciona de 1 : M con la tabla ANIMAL.

RAZA		
id_raza	<pi> Serial	<M>
fecha_raza	Date	
nombre_raza	Variable characters (50)	
nombre_registra_raza	Variable characters (50)	
id_raza	<pi>	

ATRIBUTO	DESCRIPCIÓN	TIPO	LONGITUD	ME	MFIL	OTRA VALIDACIÓN
Id_raza	Es el código que identifica a la tabla vacunas	Serial		X		
Fecha_raza	En este campo se registra la fecha que se ingresó la raza del animal	Date		X		
Nombre_raza	En este campo se ingresa el nombre de la raza del animal	Variable characters	50	x		
Nombre_regisra_raza	En este campo se ingresa el nombre_registra_raza	Variable characters	50	x		

- **TABLA CATEGORÍA:** Se registra la información que tiene categoría



UNIVERSIDAD CENTRAL DEL ECUADOR
Sistema de Registro de Animales y sus Derivados del Centro Experimental Uyumbicho

CATEGORIA	
id_categoria	<pi> Serial
fecha_categoria	Date
nombre_categoria	Variable characters (5)
nombre_registra_categoria	Variable characters (5)
inicio_categoria	Integer
fin_categoria	Integer
unidad_tiempo	Variable characters (5)

ATRIBUTO	DESCRIPCIÓN	TIPO	LONGITUD	ME	MFIL	OTRA VALIDACIÓN
Id_categoria	Es el código que identifica a la tabla tipo_animal	Serial		X		
Fecha_categoria	Se ingresa la fecha de la categoría	Variable characters	50			
Nombre_categoria	En este campo se ingresa el nombre de la categoría	Date		X		
Nombre_registra_categoria	Se ingresa el nombre que registra la categoría	Variable characters	50	X		
Inicio_categoria	Se registra el inicio de la categoría	Integer		x		
Fin_categoria	Se ingresa el fin de la categoría	Integer		x		
Unidad_tiempo	Se ingresa la unidad de tiempo la categoría	Variable characters	50	x		

- **TABLA ANIMAL POR CATEGORÍA:** Se relaciona de M : 1 con la tabla categoría y se ingresa la información que se tiene de los animales por cada una de sus categorías

ANIMAL_POR_CATEGORIA		
id_animal_categoria	<pi> Serial	<\n
fecha_animal_categoria	Date	
cantidad_animal_categoria	Integer	
nombre_registra_ac	Variable characters (50)	
observacion_ac	Variable characters (60)	
id_historial_categoria	<pi>	

ATRIBUTO	DESCRIPCIÓN	TIPO	LONGITUD	ME	MFIL	OTRA VALIDACIÓN
Id_animal_categoria	Es el código se identifica a la tabla	Serial		X		



	animal_por_categoría					
Fecha_animal_categoría	Se ingresa la fecha_animal_categoría	Date		X		
Nombre_registro_animal_categoría	En este campo se ingresa el nombre del animal por categoría	Variable characters	50	X		
Observación	Se ingresa alguna obsevación que tenga	Variable characters	50	X		

- **TABLA ALIMENTO:** Seingresa la información del alimento que consume el animal.

ALIMENTO		
id_alimento	<pi>	Serial
fecha_alimento		Date
descripcion		Variable characters (100)
cantidad_alimento		Float (4)
numero_dias		Integer
valor_alimento		Float (4)
nombre_registro_alimento		Variable characters (50)
id_alimento	<pi>	

ATRIBUTO	DESCRIPCIÓN	TIPO	LONGITUD	ME	MFIL	OTRA VALIDACIÓN
Id_alimento	Es el código que identifica a la tabla vacunas	Serial		X		
Fecha_alimento	En este campo se registra la fecha que se proporciona el alimento	Date		X		
descripción	En este campo se da alguna descripción sobre el alimento	Variable characters	50	X		
Cantidad_alimento	En este campo se ingresa la cantidad de alimento	Float	4	X		
Número_días	En este campo se ingresa el número de días que se da el alimento	Integer		X		
Valor_alimento	En este campo se ingresa el valor del alimento	Float	4	X		
Nombre_registro_alimento	En este campo se ingresa del encargado que registra el alimento	Variable characters	50	X		



- **TABLA TIPO DE ALIMENTO:** Se relaciona de 1:M con la tabla ALIMENTO y aquí se regista los tipos de alimentos que se le dan a los animales en el centro.

TIPO_ALIMENTO		
id_tipo_alimento	<pi>	Serial
nombre_tipo_alimento		Variable characters (50)
id_tipo_alimento	<pi>	

ATRIBUTO	DESCRIPCIÓN	TIPO	LONGITUD	ME	MFIL	OTRA VALIDACIÓN
Id_tipo_alimento	Es el código que identifica a la tabla tipo_alimento	Serial		X		
nombre_tipo_alimento	En este campo se registra el nombre del tipo de alimento	Variable characters	50	X		

- **TABLA VACUNA:** Se ingresa la información de las vacunas que se realizan a los animales en el centro.

VACUNA		
id_vacunas	<pi>	Serial
fecha_vacuna		Date
dosis_vacuna		Float (4)
fecha_revacunacion		Date
observaciones		Variable
nombre_registro_vacuna		Variable
id_vacunas	<pi>	

ATRIBUTO	DESCRIPCIÓN	TIPO	LONGITUD	ME	MFIL	OTRA VALIDACIÓN
Id_vacunas	Código que identifica a la tabla de vacuna	Serial				
Fecha_vacuna	En este campo se ingresa la fecha de la vacuna	Date				
Dosis_vacuna	En este campo se ingresa la dosis de la vacuna	Variable characters	50			
Fecha_revacunación	Se ingresa la fecha de revacunación	Date				
Observaciones	En este campo se ingresa la observación	Variable characters	50			
Nombre_registro_vacuna	En este campo se ingresa el nombre que registra la vacuna	Variable characters	50			

- **TABLA TIPO DE VACUNA:** Se ingresa los datos de los tipos de vacunas que existen para los animales se relaciona de 1: M con la tabla VACUNA



TIPO_VACUNA		
id_tipo_vacuna	<pi>	Serial <M>
nombre_vacuna		Variable characters (50)
fecha_tipo_vacuna		Date
id_tipo_vacuna	<pi>	

ATRIBUTO	DESCRIPCIÓN	TIPO	LONGITUD	ME	MFIL	OTRA VALIDACIÓN
Id_tipo_vacuna	Es el código que identifica a la tabla tipo_vacuna	Serial		X		
nombre_vacuna	En este campo se registra el nombre de la vacuna	Variable characters	50	X		
Fecha_tipo_vacuna	En este campo se registra la fecha del tipo de vacuna	date		X		

- **TABLA VIA DE APLICACIÓN:** En esta tabla se ingresa la información de la vía que se pone las vacunas se relaciona de 1:M con la tabla VACUNA

VIA_APPLICACION		
id_via_aplicacion	<pi>	Serial <M>
nombre_via_aplicacion		Variable characters (50)
id_via_aplicacion	<pi>	

ATRIBUTO	DESCRIPCIÓN	TIPO	LONGITUD	ME	MFIL	OTRA VALIDACIÓN
Id_vía_aplicación	Es el código que identifica a la tabla vía_aplicación	Serial		X		
Nombre_vía_aplicación	En este campo se ingresa el nombre de la vía de aplicación	Variable characters	50	X		

- **TABLA PRODUCCIÓN:** Se ingresa los datos de la producción de leche, huevos diarios y miel de abeja según la temporada.



PRODUCCION		
id_produccion	<pi>	Serial
fecha_produccion		Date
cantidad_leche_mañana	Float (4)	
cantidad_leche_tarde	Float (4)	
cantidad_huevos_rotos	Integer	
cantidad_huevos_rotos_viaje	Integer	
cantidad_huevos_sanos	Integer	
cantidad_miel	Float (4)	
cantidad_polen	Float (4)	
nombre_registra_produccion	Variable	
total_produccion	Float (4)	

ATRIBUTO	DESCRIPCIÓN	TIPO	LONGITUD	ME	MFIL	OTRA VALIDACIÓN
Id_producción	En este campo se ingresa el código de la producción	Serial		x		
Fecha_producción	En este campo se registra la fecha de la producción	Date		x		
Cantidad_leche_mañana	En este campo se registra la cantidad de leche mañana	Float	4	x		
Cantidad_de_leche_tarde	En este campo se registra la cantidad de leche tarde	Float	4	x		
Cantidad_huevos_rotos	En este campo se registrará la cantidad de huevos rotos	Integer		x		
Cantidad_huevos_rotos_viaje	En este campo se registra la cantidad de huevos rotos por viaje	Integer		x		
Cantidad_huevos_sanos	En este campo se registra la cantidad de huevos sanos	Integer		x		
Cantidad_miel	En este campo se registra la cantidad de miel	Integer		x		
Cantidad_polen	En este campo se registra la cantidad de polen	Integer		x		
Nombre_registra_producción	En este campo se registra la producción	Variable characters		x		
Total_producción	En este campo se registra el total de la producción	float	4	x		

- **TABLA MUERTE:** Se registra la información de la muerte de los animales que se dan en el Centro.



UNIVERSIDAD CENTRAL DEL ECUADOR
Sistema de Registro de Animales y sus Derivados del Centro Experimental Uyumbicho

MUERTE	
id_muerte	<pi> Serial
fecha_muerte	Date
causa_muerte	Variable characters (50)
cantidad_muerte	Integer
nombre_registra_muerte	Variable characters (50)
id_muerte	<pi>

ATRIBUTO	DESCRIPCIÓN	TIPO	LONGITUD	ME	MFIL	OTRA VALIDACIÓN
Id_muerte	Es el código que identifica a la tabla Id_muerte	Serial		X		
Fecha_muerte	En este campo se registra la fecha de la muerte del animal	Date		X		
Causa_muerte	En este campo se ingresa la causa de la muerte del animal	Variable characters	50	X		
Cantidad_muerte	En este campo se ingresa la cantidad de muerte	Integer				
Nombre_registra_muerte	Se ingresa el nombre que registra la muerte	Variable characters	50	X		

- **TABLA CLIENTE:** Se ingresa los datos de los clientes que compran en el CEU.

CLIENTE	
id_cliente	<pi> Serial <M>
fecha_cliente	Date
cedula_cliente	Variable characters (50)
nombre_cliente	Variable characters (50)
apellido_cliente	Variable characters (50)
direccion_cliente	Variable characters (50)
telefono_cliente	Variable characters (50)
id_cliente	<pi>

ATRIBUTO	DESCRIPCIÓN	TIPO	LONGITUD	ME	MFIL	OTRA VALIDACIÓN
Id_cliente	Es el código que identifica a la tabla id_cliente	Serial		X		
Cédula_cliente	En este campo se ingresa la cédula del cliente que va ha comprar el producto	Variable characters		X		
Nombre_cliente	Ingresar el nombre del cliente			X		



Apellido_cliente	Se ingresa el apellido del cliente	Variable characters	50	X		
Dirección_cliente	Se ingresa la dirección del cliente	Variable characters	50	X		
Teléfono_cliente	Se ingresa el teléfono del cliente	Variable characters	50	X		

- **TABLA CABECERA VENTA:** Se registra los datos que deben estar en la factura de la venta del producto y se relaciona de M:1 con la tabla CLIENTE

CABECERA_VENTA		
id_cab_venta	<pi>	Serial <M>
fecha_cab_venta		Date
descripción_general		Variable characters (100)
total_venta		Float (4)
nombre_registra_factura		Variable characters (50)
id_cab_venta	<pi>	

ATRIBUTO	DESCRIPCIÓN	TIPO	LONGITUD	ME	MFIL	OTRA VALIDACIÓN
Id_cab_venta	Es el código que identifica a la tabla cab_venta	Serial		X		
Fecha_cab_venta	En este campo se ingresa la fecha que se realizó la factura	Date		X		
Descripción_general	En este campo se ingresa una descripción general de la venta	Variable characters	50		X	
Total_venta	Se Ingrasa el total de venta	float	4	X		
Nombre_registra_factura	Se ingresa el nombre que se registra la factura	Variable characters	50	x		

- **TABLA DETALLE VENTA:** Se registra los datos que se realizan en la factura.

DETALLE_VENTA		
id_det_venta	<pi>	Serial <M>
cantidad_venta		Integer
descripción_venta		Variable characters (100)
valor_unitario_venta		Float (4)
id_det_venta	<pi>	



ATRIBUTO	DESCRIPCIÓN	TIPO	LONGITUD	ME	MFIL	OTRA VALIDACIÓN
Id_cab_venta	Es el código que identifica a la tabla cab_venta	Serial		X		
Fecha_cab_venta	En este campo se ingresa la fecha que se realizó la factura	Date		X		
Descripción_general	En este campo se ingresa una descripción general de la venta	Variable characters	50	X		
Total_venta	Se ingresa el total de la venta	float	4	X		

- **TABLA PRODUCTO:** Se ingresa los datos del producto elaborado

PRODUCTO	
id_producto	<pi> Serial
fecha_producto	Date
cantidad_producto	Integer
nombre_registra_producto	Variable char
total_producto	Integer

ATRIBUTO	DESCRIPCIÓN	TIPO	LONGITUD	ME	MFIL	OTRA VALIDACIÓN
Id_producto	Es el código que identifica a la tabla producto	Serial		X		
Fecha_producto	En este campo se ingresa la fecha que se elabora el producto	Date		X		
Cantidad_producto	Se ingresa la cantidad del producto	Integer		X		
Nombre_registra_producto	En este campo se ingresa la cantidad del producto	Variable characters	50	X		

- **TABLA TIPO PRODUCTO:** Se ingresa la información de los diferentes tipos de productos que tiene el CEU.



TIPO_PRODUCTO		
id_tipo_producto	<pi>	Serial
fecha_tipo_producto		Date
nombre_tipo_producto		Variable characters (30)
id_tipo_producto	<pi>	

ATRIBUTO	DESCRIPCIÓN	TIPO	LONGITUD	ME	MFIL	OTRA VALIDACIÓN
Id_tipo_producto	Es el código que identifica tabla tipo_producto	Serial		X		
Fecha_tipo_producto	En este campo se ingresa la fecha del tipo del producto	Date		X		
Nombre_tipo_producto	En este campo se ingresa el nombre del tipo del producto	Variable characters	50	X		

- **TABLA UNIDAD:** Se ingresa el valor de la unidad del producto pudiendo estar en kilo, medio kilo, galón, etc. Tiene una relación de 1:M con la tabla TIPO PRODUCTO

UNIDAD		
id_unidad	<pi>	Serial
fecha_unidad		Date
nombre_unidad		Variable characters (20)
id_unidad	<pi>	

ATRIBUTO	DESCRIPCIÓN	TIPO	LONGITUD	ME	MFIL	OTRA VALIDACIÓN
Id_unidad	Es el código que identifica tabla unidad	Serial		X		
Fecha_unidad	En este campo se ingresa la fecha de la unidad del producto	Date		X		
Nombre_unidad	En este campo se ingresa el nombre de la unidad	Variable characters	50	x		

- **TABLA CONSUMO PRODUCCIÓN:** Se ingresa el consumo de la producción que se tiene en el CEU.



CONSUMO_PRODUCCION		
id_consumo_produccion	<pi>	Serial <M>
fecha_consumo_produccion		Date
cantidad_materia_prima		Float
nombre_materia_prima		Variable characters (50)
utilizacion_materia_prima		Variable characters (50)
id_consumo_produccion	<pi>	

ATRIBUTO	DESCRIPCIÓN	TIPO	LONGITUD	ME	MFIL	OTRA VALIDACIÓN
Id_consumo	Es el código que identifica a la tabla Id_consumo	Serial		X		
Fecha_consumo_producción	En este campo se ingresa la fecha que se hizo el consumo de la producción	Date		X		
Cantidad_materia_prima	Se ingresa la cantidad de la materia prima que existe	float	4	X		
Nombre_materia_prima	Se ingresa el nombre de la materia prima	Variable characters	50	X		
Utilizacionmateria_prima	Se ingresa la utilización de la materia prima	Variable characters	50	x		

- **TABLA DESTINO:** Se ingresa a que esta destinado los productos que tiene el CEU y se relaciona de 1:M con la tabla CONSUMO PRODUCCIÓN.

DESTINO		
id_destino	<pi>	Serial <M>
nombre_destino		Variable characters (50)
nombre_registra_destino		Variable characters (50)
fecha_destino		Date
id_destino	<pi>	

ATRIBUTO	DESCRIPCIÓN	TIPO	LONGITUD	ME	MFIL	OTRA VALIDACIÓN
Id_destino	Es el código que identifica a la tabla Id_destino	Serial		X		
Nombre_destino	En este campo se ingresa el	Date		X		



	nombre del destino					
Nombre_registro_destino	Se ingresa el nombre que registra el destino	float	4	X		

- **TABLA INSEMINACIÓN:** Se registra los datos de la inseminación que se realiza a los animales en el CEU.

INSEMINACION		
id_inseminacion	<pi>	Serial
num_inseminacion	Integer	<M>
fecha_inseminacion	Date	
fecha_inseminacion_efectiva	Date	
nombre_registro_inseminacion	Variable characters (50)	
estado_inseminacion	Variáble characters (50)	
id_inseminacion	<pi>	

ATRIBUTO	DESCRIPCIÓN	TIPO	LONGITUD	ME	MFIL	OTRA VALIDACIÓN
Id_inseminación	Es el código que identifica la inseminación	Serial		X		
Num_inseminación	En este campo se registra el número de inseminación	Integer		X		
Fecha_inseminacion	En este campo ingresa la fecha de inseminación	Date		x		
Fecha_inseminación_efectiva	En este campo se ingresa la fecha de la inseminación efectiva	Date		x		
Nombre_registro_inseminaci ón	En este campo ingresa el nombre de quien registra la inseminación	Variable characters	50	x		
Estado_inseminacion	En este campo ingresa el estado de la inseminación	Variable characters	50	x		



- **TABLA PESO:** Se registra los datos del peso del animal

PESO	
id_peso	<pi> Serial
fecha_peso	Date
peso	Float (4)
nombre_registra_peso	Variable characters
descripción_peso	Variable characters
id_peso	<pi>

ATRIBUTO	DESCRIPCIÓN	TIPO	LONGITUD	ME	MFIL	OTRA VALIDACIÓN
Id_peso	Es el código que identifica el id_peso	Serial		X		
Fecha_peso	En este campo se registra la fecha del peso	Date		X		
peso	En esta opción se registra el peso del animal	float	4	x		
Nombre_registra_peso	En este campo se ingresa el nombre que registra el peso	Variable characters		x		
Descripción_peso	En este campo se registra la descripción del animal	Variable characters		x		

- **TABLA EVENTO:** Se registra el evento que tiene el animal

EVENTO		
id_evento	<pi> Serial	<M>
fecha_evento	Date	
nombre_evento	Variable characters (50)	
nombre_registra_evento	Variable characters (50)	
id_evento	<pi>	

ATRIBUTO	DESCRIPCIÓN	TIPO	LONGITUD	ME	MFIL	OTRA VALIDACIÓN
Id_evento	Es el código que identifica el id_evento	Serial		X		
Fecha_evento	En este campo se registra la fecha del evento	Date		X		
Nombre_evento	En esta opción se registra el nombre del evento	Variable characters		x		
Nombre_registra_evento	En este campo se registra el nombre del evento	Variable characters		x		



- **TABLA SEMEN:** Se registra la recolección que se realiza al animal del semen

SEmen		
id_semen	<pi>	Serial <M>
fecha_recolección		Date
cantidad_recolección		Float (4)
nombre_registro_semen		Variable characters (50)
id_semen	<pi>	

ATRIBUTO	DESCRIPCIÓN	TIPO	LONGITUD	ME	MFIL	OTRA VALIDACIÓN
Id_semen	Es el código que identifica el id_semen	Serial		X		
Fecha_recolección	En este campo se registra fecha de recolección	Date		X		
Cantidad_recolección	En esta opción se registra la cantidad de recolección	float	4	x		
Nombre_registro_semen	En este campo se registra el semen	Variable characters		x		

- **TABLA SANITARIO:** Se ingresa el control sanitario que se realiza al animal.

SANITARIO		
id_sanitario	<pi>	Serial <M>
fecha_chequeo		Date
síntomas		Variable characters (50)
diagnóstico		Variable characters (50)
tratamiento		Variable characters (50)
nombre_registro_sanitario		Variable characters (50)
id_sanitario	<pi>	

ATRIBUTO	DESCRIPCIÓN	TIPO	LONGITUD	ME	MFIL	OTRA VALIDACIÓN
Id_sanitario	Es el código que identifica el id_evento	Serial		X		
Fecha_chequeo	En este campo se registra fecha del evento	Date		X		
síntomas	En esta opción se registra el nombre del evento	Variable character s		x		
diagnóstico	En este campo se registra el nombre del evento	Variable character s		x		



tratamiento	En esta opción se registra el tratamiento	Variable character s		x		
Nombre_registro_sanitario	En este campo se ingresa el nombre de quien registra el control sanitario	Variable character s		x		

- **TABLA USUARIO:** Se ingresa los datos para la identificación del usuario

USUARIO		
codigo_usuario	<pi>	Serial
identificacion		Variable characters (50)
nombres		Variable characters (50)
apellido_materno		Variable characters (50)
apellido_paterno		Variable characters (50)
clave		Variable characters (50)
email		Variable characters (50)
estado		Integer
codigo_usuario	<pi>	<M>

ATRIBUTO	DESCRIPCIÓN	TIPO	LONGITUD	ME	MFIL	OTRA VALIDACIÓN
Código_usuario	Es el código que identifica al usuario	Serial		X		
identificación	En este campo se registra la identificación del usuario	Date		X		
nombres	En esta opción se registra los nombres de los usuarios	Variable characters		x		
Apellido materno	En este campo se registra el apellido materno	Variable characters		x		
Apellido paterno	En esta opción se registra el apellido paterno	Variable characters		x		
clave	En este campo se ingresa la clave	Variable characters		x		
Email	En este campo se ingresa el email del usuario	Variable characters		x		
estado	En este campo se realiza el estado	Variable characters		x		



- **TABLA DE PERFIL:** Se registra los datos que se tienen que dar al perfil

PERFIL		
codigo_perfil	<pi>	Serial <M>
descripcion		Variable characters (100)
estado		Integer
lectura_escritura		Boolean
codigo_perfil	<pi>	

ATRIBUTO	DESCRIPCIÓN	TIPO	LONGITUD	ME	MFILE	OTRA VALIDACIÓN
Código_perfil	Es el código que identifica el perfil del usuario	Serial		X		
descripción	En este campo se registra la descripción	Variable characters		X		
estado	En esta opción se registra el estado	Integer		x		
Lectura_escritura	En este campo se registra el nombre del evento	Boolean		x		

- **TABLA PERFIL DE USUARIO:** Esta tabla nace del rompimiento de M:M de la tabla PERFIL Y USUARIO.

PERFIL_USUARIO
codigo_perfil_usuario <pi> Serial <M>
codigo_perfil_usuario <pi>

ATRIBUTO	DESCRIPCIÓN	TIPO	LONGITUD	ME	MFILE	OTRA VALIDACIÓN
Código_perfil_usuario	Es el código que identifica a la tabla perfil_usuario	Serial		X		



ANEXO5:

GLOSARIO DE TÉRMINOS:

API: Es una Interfaz de Programación de Aplicaciones (API: por sus siglas en inglés) provista por los creadores del lenguaje Java, y que da a los programadores los medios para desarrollar aplicaciones Java

CEU: Centro Experimental Uyumbicho.

JAVA: Java es un lenguaje de programación por objetos creado por Sun Microsystems, Inc. que permite crear programas que funcionan en cualquier tipo de ordenador y sistema operativo.

Bases De Datos: Es una colección de datos distribuidos de una manera tabular especial de acuerdo a la forma de utilización de los datos mismos.

Programación Orientada Objetos: (OOP) se basa en la idea de un mundo lleno de objetos y que la resolución del problema se realiza en términos de objetos.

Eclipse: Eclipse es un IDE (Integrated Development Environment, entorno integrado de desarrollo) para Java muy potente. Es libre y fue creado originalmente por IBM. Se está convirtiendo en el estándar de facto de los entornos de desarrollo para Java.

JPA: Java Persistence API, más conocida por su sigla JPA, es la API de persistencia desarrollada para la plataforma Java EE e incluida en el estándar EJB3. Esta API busca unificar la manera en que funcionan las utilidades que proveen un mapeo objeto-relacional.

JSP: JavaServer Pages (JSP) es una tecnología Java que permite generar contenido dinámico para web, en forma de documentos HTML.

EJB: Los EJB o Enterprise Java Beans son componentes JEE que se ejecutan dentro de un container EJB, un entorno de ejecución dentro de un Application Server.

Interfaz: en Java es una colección de métodos abstractos y propiedades.



Beans: Se usan para encapsular varios objetos en un único objeto (la vaina o Bean en inglés), para hacer uso de un solo objeto en lugar de varios más simples.

Métodos: Los métodos son las acciones funciones o procedimientos que realiza nuestro programa; los métodos son subrutinas que manipulan los datos definidos por una clase.

Eventos: Permiten comunicarnos con otros JavaBeans

Página Web: Información publicada en Internet, a la cual puede acceder mediante una dirección web.

PostgreSQL: Es un sistema de gestión de base de datos relacional orientada a objetos y libre, publicado bajo la licencia BSD.

Localhost: Publicación de páginas en el servidor propio del equipo.



ANEXO 6

FOTOS TOMADAS EN LA INVESTIGACIÓN.

En esta imagen muestra la entrada de la carretera al **CENTRO EXPERIMENTAL UYUMBICHO DE LA FACULTAD DE VETERINARIA Y ZOOTECNIA**



En esta imagen se muestra la entrada al **CENTRO EXPERIMENTAL UYUMBICHO DE LA FACULTAD DE VETERINARIA Y ZOOTECNIA**



Aquí encontramos las oficinas del Centro Experimental Uyumbicho.





ÁREA PORCINA

Está imagen muestra la distribución de los cerdos en etapa adulta.



Está imagen muestra la distribución de los cerdos en etapa de crecimiento.



La imagen muestra la alimentación que reciben los cerdos en la etapa de crecimiento





ÁREA CUYES

La casa que se ve en la imagen indica el área de cuyes.



Se muestra los cuyes al momento de comer



La imagen muestra la distribución de los cuyes por tamaño, raza, edad.





ÁREA BOVINA

Se muestra el establo donde permanecen en la noche las vacas y toros.



La imagen muestra una vaca con su número de referencia que es el arete que se encuentra en la oreja.



La imagen muestra algunas de las vacas que tiene el Centro.





ÁREA AVÍCOLA

La imagen muestra la alimentación de las gallinas.



ÁREA DE APICULTURA

La imagen muestra una de las colmenas que tiene el Centro para su posterior recolección de miel



El Centro Experimental Uyumbicho produce productos como:

- LECHE
- QUESOS
- YOGURT NATURAL Y SABORES



Producción de Leche



Productos Elaborados

