UNIVERSIDAD POLITÉCNICA SALESIANA
SEDE QUITO

CARRERA:

INGENIERÍA DE SISTEMAS

Trabajo de titulación previo a la obtención del título de:

Ingenieros de Sistemas

TEMA:

ANÁLISIS, DISEÑO, DESARROLLO E IMPLEMENTACIÓN DE UN SISTEMA WEB PARA EL PADRÓN DE USUARIOS DEL SISTEMA DE RIEGO COMUNITARIO EN EL CANTÓN CAYAMBE Y PEDRO MONCAYO

AUTORES:

MIGUEL ÁNGEL SILVA VILLACIS

NORMAN LEONEL CHUQUIRIMA ZAQUINAULA

TUTOR:

DANIEL GIOVANNY DÍAZ ORTIZ

Quito, agosto del 2018

CESIÓN DE DERECHOS DE AUTOR

Nosotros, Miguel Ángel Silva Villacis, con documento de identificación N° 1722726922 y Norman Leonel Chuquirima Zaquinaula, con documento de identificación N° 1720229234, manifestamos nuestra voluntad y cedemos a la Universidad Politécnica Salesiana la titularidad sobre los derechos patrimoniales en virtud de que somos autores del trabajo de titulación con el tema: "ANÁLISIS, DISEÑO, DESARROLLO E IMPLEMENTACIÓN DE UN SISTEMA WEB PARA EL PADRÓN DE USUARIOS DEL SISTEMA DE RIEGO COMUNITARIO EN EL CANTÓN CAYAMBE Y PEDRO MONCAYO", mismo que ha sido desarrollado para optar por el título de INGENIEROS DE SISTEMAS en la Universidad Politécnica Salesiana, quedando la Universidad facultada para ejercer plenamente los derechos cedidos anteriormente.

En aplicación a lo determinado en la Ley de Propiedad Intelectual, en nuestra condición de autores nos reservamos los derechos morales de la obra antes citada. En concordancia, suscribimos este documento en el momento que hacemos la entrega del trabajo final en formato impreso y digital a la Biblioteca de la Universidad Politécnica Salesiana.

MIGUEL ÁNGEL SILVA VILLACIS
C.I: 1722726922

NORMAN LEONEL CHUQUIRIMA ZAQUINAULA
C.I: 1720229234

Quito, agosto del 2018

DECLARATORIA DE COAUTORÍA DEL TUTOR

Yo declaro que bajo mi dirección y asesoría fue desarrollado el PROYECTO TÉCNICO, con el tema: "ANÁLISIS, DISEÑO, DESARROLLO E IMPLEMENTACIÓN DE UN SISTEMA WEB PARA EL PADRÓN DE USUARIOS DEL SISTEMA DE RIEGO COMUNITARIO EN EL CANTÓN CAYAMBE Y PEDRO MONCAYO", realizado por MIGUEL ÁNGEL SILVA VILLACIS y NORMAN LEONEL CHUQUIRIMA ZAQUINAULA, obteniendo un producto que cumple con todos los requisitos estipulados por la Universidad Politécnica Salesiana, para ser considerados como trabajo final de titulación.

Quito, agosto del 2018

DANIEL GIOVANNY DÍAZ ORTIZ
C.I: 1716975501

Dedicatoria

Este proyecto y todo el arduo trabajo va dedicado a toda mi familia, principalmente a mi padre Miguel Silva, a mi madre Martha Villacis que con todo su esfuerzo, dedicación, sacrificio, paciencia y amor han logrado convertirme en una gran persona capaz de cumplir mis metas y sueños. Por ser un pilar fundamental en mi vida y brindarme su apoyo incondicional.

También va dedicado a todas las personas que han aportado de una u otra manera a cumplir esta meta. No me queda más que agradecerles y compartir este logro con ustedes.

Miguel Ángel Silva Villacis

El proyecto fue realizado con un gran esfuerzo, así como los esfuerzos que realizaron mis padres con su apoyo y así poder terminar esta etapa de mi vida, por eso, son ellos a quienes les dedico este proyecto, ellos me han demostrado que se puede hacer todo lo que me proponga y que van a estar ahí cuando más lo necesite, mis hermanos fueron un gran pilar, les agradezco por todos sus consejos y apoyo.

En este proceso logré conocer a muchas personas que se convirtieron en un gran apoyo y que ahora están en mi vida por lo que también agradezco a Mike mi compañero de Tesis, ya que hemos pasado por todo el proceso tanto de universidad como de la Tesis , a Jhon quien se convirtió en alguien de la familia y que también contribuyó para que este proyecto culmine, en fin existen innumerables personas que contribuyeron a esto y que merecen ser nombradas pero saben que sin todos sus granitos de arena esto no podría culminarse.

Norman Leonel Chuquirima Zaquinaula

Agradecimiento

Extendemos un especial y sincero agradecimiento a nuestro tutor Daniel Díaz, que con su entrega, conocimiento y experiencia ha podido orientarnos durante todo el proceso para la construcción del proyecto técnico; Por su tiempo, valores y toda la ayuda brindada no solo en el ámbito educativo, sino también en el ámbito personal.

Además, agradecemos a todas las personas involucradas en nuestro desarrollo profesional, compañeros y docentes tanto de la Universidad Politécnica Salesiana "Sede Quito", como también a los miembros de los centros de investigación de la Universidad Politécnica Salesiana "Sede Cayambe".

Norman Leonel Chuquirima Zaquinaula

Miguel Ángel Silva Villacis

ÍNDICE

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

Resumen

En la actualidad en las comunidades del cantón Cayambe y Pedro Moncayo el registro de la información del padrón de usuarios no está sistematizado, por lo que la información se recolecta manualmente y esto hace que se dificulte el análisis, la recopilación y la gestión de dicha información que al final del día provoca una mayor carga de trabajo.

El presente proyecto se enfoca en el desarrollo de un sistema web "Padrón de Usuarios", que contribuirá a las comunidades del cantón Cayambe y Pedro Moncayo en el proceso de levantamiento de información y con esto proporcionar insumos a los usuarios para que puedan tomar decisiones que mejoren la gestión del agua.

La metodología XP está enfocada en las relaciones interpersonales y se preocupa por el aprendizaje de los desarrolladores, ayudó con las múltiples reuniones con el usuario, tener un esquema claro de que necesitan las comunidades.

Con el desarrollo de este proyecto se tiene una aplicación estable, simple y amigable para que el usuario campesino por medio de la capacitación de computación básica impartida la Universidad Politécnica Salesiana Sede Cayambe no presente dificultades al momento de utilizar el Padrón de Usuarios, con el sistema se puso en marcha el registro de información de los usuarios de las diferentes juntas de riego del cantón Cayambe y Pedro Moncayo que no cuentan con acceso al servicio de internet y se trata la información de manera automatizada.

Abstract

Currently, in the communities of the Cayambe and Pedro Moncayo cantons, the registration of information from the user census is not systematized, so the information is collected manually and this makes it difficult to analyze, collect and manage this information, which generates more work and consumes more resources at the end of the day.

The present project focuses on the development of a web system "Padrón de Usuarios", which will contribute to the communities of the Cayambe and Pedro Moncayo cantons in the process of gathering information and thus provide inputs to users so that they can make decisions that improve water management.

The User List system has a web environment for the communities that has the internet service and a local environment (desktop application) for those communities that do not, making it an accessible system.

With the development of this project, the objective is to obtain a supported application, simple and user-friendly application so that the rural user, through the basic computer training provided by the Salesian Polytechnic University, headquarters “Cayambe”, does not present difficulties when using the Users' Register.

INTRODUCCIÓN

La información es un punto crítico a la hora de tomar decisiones dentro de cualquier organización, en la actualidad existen varias tecnologías de la información que permiten el resguardo, manejo y tratamiento de esta. Un sistema web es un claro ejemplo de cómo las organizaciones manejan y presentan la información, permitiendo interactuar a diferentes usuarios en tiempo real, abaratando costos, disminuyendo tiempos y optimizando tareas.

Por lo tanto, este proyecto tiene como objetivo, facilitar el registro de datos de la familia, datos de unidad productiva (lote) y riego, sistema de producción agrícola, sistema de producción pecuaria de las comunidades del cantón Cayambe y Pedro Moncayo, con el fin de facilitar el análisis, recopilación y gestión de dicha información. Evitando así inconsistencia de datos, fichas repetidas, pérdida de información, información imprecisa e información duplicada que genera más trabajo a la hora del análisis consumiendo costos y recursos.

Por lo tanto, el sistema web facilita el tratamiento que se le da a la información para la toma de decisiones y mejora de la distribución del agua.

Antecedentes

Las comunidades del Cantón Cayambe y Pedro Moncayo tienen conformadas las juntas de agua de riego o los sistemas de agua de riego, las mismas que al solicitar financiamiento para la mejora de su sistema o para implementar un sistema de riego nuevo acuden a instituciones públicas, las mismas que solicitan ciertos estudios previos o información como: Planes productivos, planes de manejo ambiental, proyectos de estudios de factibilidad y padrón de usuarios.

Para la construcción de la información requerida por las instituciones públicas, dichas juntas acudieron a la Universidad Politécnica Salesiana sede Cayambe y al ser esta una instancia académica y no consultora, decide realizar el curso de educación continua de tecnificación y manejo de sistemas comunitarios y el de gestores de sistemas comunitarios por medio de vinculación con la sociedad a través de la carrera de Gestión para el Desarrollo Local.

Actualmente en el Cantón Cayambe y Pedro Moncayo el registro de la información del padrón de usuarios no está sistematizado, es decir se lo llena a mano y en algunos casos se registran en una hoja de Excel, por lo tanto, se dificulta el análisis, la recopilación y la gestión de dicha información. Además del gasto que se produce actualmente en contratación del personal, alimentación y movilización al levantar la información de forma manual.

Al levantar la información en fichas manualmente se presentan problemas de inconsistencia de datos, fichas repetidas, pérdida de información, información imprecisa e información duplicada que genera más trabajo a la hora del análisis consumiendo más recursos.

Con el proceso actual de gestión de la información no se puede tener una investigación concreta ni obtener resultados específicos para la toma de decisiones.

Tanto los usuarios de riego de Cayambe como Pedro Moncayo no poseen un control, método y horario adecuado para el uso responsable del agua.

Actualmente los usuarios de riego de Cayambe y Pedro Moncayo no llevan un correcto registro del sistema de producción agrícola y pecuario lo que impide tener un histórico que permita obtener estadísticas y proyecciones de dicha producción.

Problema

Las comunidades del Cantón Cayambe y Pedro Moncayo tienen conformadas las juntas de agua de riego o los sistemas de agua de riego, las mismas que al solicitar financiamiento para la mejora de su sistema o para implementar un sistema de riego nuevo acuden a instituciones públicas, las mismas que solicitan ciertos estudios previos o información como: Planes productivos, planes de manejo ambiental, proyectos de estudios de factibilidad y padrón de usuarios.

Para la construcción de la información requerida por las instituciones públicas, dichas juntas acudieron a la Universidad Politécnica Salesiana sede Cayambe y al ser esta una instancia académica y no consultora, decide realizar el curso de educación continua de tecnificación y manejo de sistemas comunitarios y el de gestores de sistemas comunitarios por medio de vinculación con la sociedad a través de la carrera de Gestión para el Desarrollo Local.

Actualmente en el Cantón Cayambe y Pedro Moncayo el registro de la información del padrón de usuarios no está sistematizado, es decir es llenado a mano y en algunos casos registrada en una hoja de Excel, por lo tanto, se dificulta el análisis, la recopilación y la gestión de dicha información. Además del gasto que se produce actualmente en contratación del personal, alimentación y movilización al levantar la información de forma manual.

Al levantar la información en fichas manualmente se presentan problemas de inconsistencia de datos, fichas repetidas, pérdida de información, información imprecisa e información duplicada que genera más trabajo a la hora del análisis consumiendo más recursos.

Con el proceso actual de gestión de la información no se puede tener una investigación concreta ni obtener resultados específicos para la toma de decisiones.

Tanto los usuarios de riego de Cayambe como Pedro Moncayo no poseen un control, método y horario adecuado para el uso responsable del agua.

Actualmente los usuarios de riego de Cayambe y Pedro Moncayo no llevan un correcto registro del sistema de producción agrícola y pecuario lo que impide tener un histórico que permita obtener estadísticas y proyecciones de dicha producción.

Justificación

La cuenca del río Pisque se encuentra ubicada al norte de la provincia de Pichincha, según el escurrimiento natural de sus aguas tiene cobertura en dos cantones; Cayambe (Parroquias de Olmedo, Ayora y Cangagua) y Pedro Moncayo (Tupigachi, Tabacundo, La Esperanza, Tocachi y Malchinguì).

Se han realizado trabajos de investigación enfocados a la gestión del agua y a la evaluación de la eficiencia de los sistemas de riego que existen dentro de la zona de cobertura hidrográfica de la cuenca, que en su mayoría demuestran que los sistemas comunitarios no llegan a la eficiencia en la gestión y manejo de los mismos, básicamente por la falta de conocimiento acerca de las herramientas, técnicas y metodologías aplicadas al riego que en suma contribuyen a la toma de decisiones adecuadas al momento de la planificación, organización, administración y manejo de los sistemas de riego.

Atendiendo a la necesidad de las comunidades del cantón Cayambe y Pedro Moncayo la Universidad Politécnica Salesiana ve la necesidad de poder construir un sistema de padrón de usuarios que ayude en el proceso de levantamiento de información que genere un mecanismo óptimo para el análisis dando insumos a los usuarios para que puedan tomar decisiones que mejoren la gestión del agua.

Al sistematizar el levantamiento de información de riego se podrá evitar la inconsistencia de datos, fichas repetidas, perdida de información, información imprecisa y duplicidad de datos, ahorrando recursos y tiempo, tanto a los usuarios de riego como a los miembros encargados de los Centros, Líneas de Investigación y Vinculación con la sociedad de la Universidad Politécnica Salesiana Sede Cayambe los cuales harán uso de la información para investigaciones en el tema de riego y producción.

El sistema padrón de usuarios optimiza el correcto uso del agua mediante el registro de métodos, tipos y turnos de riego, esto reducirá los costos del tiempo de riego.

El sistema de padrón de usuarios no solo recopila información de métodos, horarios, y tipos de riego si no también lleva información acerca del sistema de producción agrícola, sistema de producción pecuario y sistema de producción bovino ya que estos datos son indispensables para realizar una programación de riego que sirve para distribuir el agua de forma equitativa considerando el tipo de cultivo o superficie que posee cada usuario o productor.

Objetivos

Objetivo general

Analizar, diseñar, desarrollar e implementar un sistema web para el padrón de usuarios para sistemas de riego comunitarios del cantón Cayambe y Pedro Moncayo con el fin de recabar información para la toma de decisiones, mejorar la gestión del agua y automatizar el control y crecimiento de los sistemas de riego de dicho sector.

Objetivos específicos

Analizar los requerimientos del sistema para obtener qué tipo de información que se va a recolectar para una óptima construcción del padrón de usuarios de riego comunitarios

Diseñar un sistema de escritorio para poder levantar información detalla en cada comunidad de riego, en base a la metodología XP.

Diseñar un sistema web que ayudará en la toma de decisiones e investigación, en base a la metodología XP.

Desarrollar un sistema de escritorio con una interfaz amigable y de fácil uso en base a los diseños y artefactos proporcionados por la metodología.

Implementar un repositorio que almacene las fichas de información de riego de los usuarios para posteriormente hacer uso de la información con fines de investigación.

Realizar pruebas de funcionamiento con el fin de una implementación correcta del software.

Marco Metodológico

Metodologías Ágiles

El desarrollo de un proyecto de software es un proceso empírico, es decir, que dicho proyecto debe estar basado tanto en la observación de los hechos y este no puede exceder de los límites de la experiencia.

El empirismo es la teoría que afirma que el conocimiento surge principalmente de la experimentación sensorial. Un proceso empírico está caracterizado por el aprendizaje adquirido a partir de la experimentación y de las adaptaciones realizadas para mejorar el proceso, es decir: Haciendo y Aprendiendo

Adecuado en las situaciones en que la incertidumbre es alta: donde los cambios son frecuentes y el producto final es difícil de ser estimado.

“Las metodologías ágiles reconocen la naturaleza empírica del software y están preparadas para acoger los cambios frecuentes, ofrecen rapidez para realizar los cambios idóneos a partir del feedback de los usuarios y se presentan con metodologías leves, enfocadas al software funcional en vez del formalismo y de la documentación extensa”.(Fuentes, 2015).

El siguiente cuadro muestra una comparación de las características, ventajas y desventajas de las principales metodologías ágiles para el desarrollo de un proyecto de software el cual permite la selección del método óptimo para el desarrollo del sistema de padrón de usuarios de riego comunitario.

Tabla 1 Comparación de metodologías ágiles

Metodologías			
Características	**XP**	**SCRUM**	**RUP**
Metodología basada en simplicidad y agilidad	X	X	
Prioridad en el control de calidad en los aspectos de producción			X
Mayor importancia en la adaptabilidad	X	X	
Mayor importancia en la verificación de calidad del software			X
Enfoque de desarrollo iterativo	X	X	X
Enfoque de desarrollo incremental	X	X	X

Características	**XP**	**SCRUM**	**RUP**
Propone un ciclo de vida del software dinámico	X		
Responde a los cambios de requisitos del cliente en cualquier punto del proyecto	X		
Responde a los cambios de requisitos del cliente hasta terminar la iteración		X	
Se valora la colaboración del cliente sobre negociaciones contractuales	X		
Pruebas unitarias automatizadas y continuas	X		
Tareas de desarrollo en pares	X		
Tareas de desarrollo individuales		X	
Tareas de desarrollo por equipos			X
Integraciones permanentes	X		
Corrección de errores y entregas frecuentes	X		
Reestructuración de código para aumentar mantenibilidad y legibilidad	X		
Extensión y corrección de código compartida	X		
Solapamiento de las fases de desarrollo del software		X	
Apropiado para proyectos de requisitos inestables	X	X	
Enfatiza el trabajo en equipo	X	X	X
Metodología basada en la creación del producto	X		
Metodología basada en la administración del proyecto		X	
Centrado en la arquitectura			X
Complejidad de desarrollo según el tamaño del proyecto			X
Utiliza equipos auto dirigidos y auto organizados		X	
Ideales en ambientes con herramientas basadas en Programación web	X	X	X

Características	XP	SCRUM	RUP
Se valora a los individuos y las interacciones sobre los procesos y las herramientas	X		
Se valora a las aplicaciones funcionando correctamente sobre la documentación extensa	X		
Se valora la respuesta al cambio sobre el seguimiento de un plan estricto	X		
Planificación basada en "Historias de usuarios"	X	X	
Planificación basada en "Casos de uso"			X
Metodología basada en tareas y responsabilidades			X
Alcance del proyecto definido antes de comenzar			X
Alcance definido según los objetivos y revaluado en cada iteración		X	
Diversas reuniones con los clientes e individuos involucrados	X		X
Simplicidad del Plan	X	X	
Reuniones de seguimiento entre el equipo de trabajo	X	X	
Simplicidad de Diseño	X		
Programación dirigida por las pruebas	X		
Propiedad colectiva del código	X		
Ritmo sostenido de trabajo	X	X	
Prioriza el orden de requerimientos del producto para su desarrollo		X	
Administra los cambios de acuerdo con evaluaciones			X

Nota: Esta tabla contiene una comparación de las principales metodologías ágiles para el desarrollo de software

La metodología que se usa es "Programación extrema o XP" que tiene como características la metodología ágil enfocada en las relaciones interpersonales para el éxito del desarrollo del software, además que se preocupa por el aprendizaje de los

desarrolladores y proporciona un buen clima de trabajo, así como la realimentación continua entre el cliente y el equipo de desarrollo, esta metodología es especialmente adecuada para proyectos imprecisos, muy cambiantes y donde existe un alto riesgo técnico.

1. CAPÍTULO I

1.1. Marco Institucional

La Universidad Politécnica Salesiana a través de los Centros y Líneas de Investigación y Vinculación con la sociedad llevan a cabo proyectos tanto de Investigación y Vinculación enfocados a las zonas más vulnerables. La Universidad cuenta con un Centro de Educación Continua en el cantón Cayambe, donde se ejecutó el proyecto "DETERMINACIÓN DE LA EFICIENCIA DEL USO DEL AGUA EN CULTIVO DE PASTOS CON SISTEMAS COMUNITARIOS DEL MARGEN IZQUIERDO DE LA CUENCA DEL RÍO PISQUE", con el apoyo de la Línea de Investigación del Agua y el Suelo, los Laboratorios de Sistemas de Información Geográfica, el Centro de Investigación de la Leche – CILEC de la UPS en convenio con la Dirección de Riego y Drenaje del Gobierno Autónomo Descentralizado de la Provincia de Pichincha – GADPP.

Tabla 2 Juntas de Riego

Número	Descripción
1	Asociación San Francisco de Monjas
2	Asociación Canal Ishigto
3	Comunidad La Chimba
4	Comunidad Cariacu
5	Comunidad de Moras
6	Directorio Canal Maldonado
7	Directorio de Agua de Riego Espiga de Oro
8	Directorio de Aguas de la Acequia Guanguilqui
9	Directorio de la Acequia San Antonio
10	Directorio de la Comunidad de Pisambilla
11	Directorio de Riego Santo Domingo 1
12	Junta de Agua de Riego Santo Domingo 2

Número	Descripción
13	Directorio de Riego Santo Domingo de Guzmán
14	Directorio de Riego Miraflores
15	Junta de Riego Río Blanco
16	Junta Administradora de Agua de Riego El Pisque
17	Junta Cubero-Izacata
18	Junta de Agua de Riego Borja Toma
19	Junta de Agua de Riego Carbun Hurco
20	Junta de Riego del Canal Razoschupa
21	Junta de Riego Perugachi
22	Junta de Riego y Abrevadero Paquiestancia
23	Junta San Francisco de Wachu - Wachu
24	Pre-comité de Desarrollo Totorauco

Nota: Esta tabla contiene las Juntas de Riego que participaron en el proyecto Sumando un total de 24 Sistemas de agua de riego, que abarcan varias comunidades de las parroquias del cantón Cayambe y Pedro Moncayo

La formulación del Padrón de Usuarios como proceso son las acciones que las comunidades aprobaron conjuntamente con la autoridad correspondiente para inscribir a las usuarias y los derechos y obligaciones que tiene con el agua.

- **La elaboración,** se trata de seguir la norma o ley que está en el país con respecto a Ley de aguas que existe y las comunidades deben prepararse, estudiarla y aplicarla.
- **La adecuación,** es la adaptación de padrones que pueden existir en las juntas aplicándolo el nuevo padrón.
- **La actualización,** se realiza conjuntamente con los encargados del padrón que ya conocen el funcionamiento de este y que saben que parámetros son los que se están actualizando.

- **La automatización**, como su nombre lo indica se desarrollan actividades automatizadas que hacen del padrón una solución rápida y confiable.(Gobierno de Pichincha & Universidad Politécnica Salesiana, 2015).

La primera etapa para formular el Padrón de Usuarios es la elaboración de la estructura para el levantamiento de información, la metodología para desarrollar esta etapa se resume en los siguientes pasos:

Tabla 3 Definición de pasos para definir la primera etapa del formulario

Número	Pasos
1	Determinación del alcance
2	Validación de la ficha
3	Elaboración de la ficha base y manual
4	Recopilación de información geoespacial
5	Socialización de la ficha con los usuarios
6	Planificación de trabajo de campo
7	Levantamiento de información en campo
8	Elaboración de base de datos del Padrón
9	Padrón de usuarios
10	Obtención de información de acuerdo a los objetivos planteados

Nota: Esta tabla contiene los pasos para definir la primera etapa del Padrón de Usuarios.

Uno de los resultados del proyecto "DETERMINACIÓN DE LA EFICIENCIA DEL USO DEL AGUA EN CULTIVO DE PASTOS CON SISTEMAS COMUNITARIOS DEL MARGEN IZQUIERDO DE LA CUENCA DEL RÍO PISQUE", fue la elaboración del Padrón de usuarios de las Juntas de riego, que de acuerdo a la Secretaria Nacional de Planificación (SENPLADES) y el Consejo Nacional de Competencias (CNC), los Gobiernos Provinciales, acorde a su facultad de gestión de los sistemas, tiene como competencia: la estandarización, consolidación y registro de

los Padrones de usuarios y de catastro de predios servidos; y las Organizaciones de usuarios: actualizar y levantar el Padrón de usuarios y catastro de predios servidos.

Las instituciones responsables de la gestión del agua de riego, no han emitido oficialmente en el país una referencia metodológica para la elaboración de Padrones de usuarios de riego, entendiéndose que cada una de ellas utiliza formatos y definiciones diversas, por lo que en el proyecto "DETERMINACIÓN DE LA EFICIENCIA DEL USO DEL AGUA EN CULTIVO DE PASTOS CON SISTEMAS COMUNITARIOS DEL MARGEN IZQUIERDO DE LA CUENCA DEL RÍO PISQUE" se plantea la metodología y herramientas para que el Gobierno Autónomo Descentralizado de la Provincia de Pichincha (GADPP), así como la SENPLADES, estandarice, consolide y avale la información de Padrones de usuarios de la cuenca que tiene a su cargo para la gestión del agua de riego.

1.2. Marco Teórico

A continuación se describe los fundamentos teóricos necesarios para el proyecto.

1.2.1. Metodología XP

"Es una metodología de desarrollo de software que tiene como principales pilares la comunicación, la simplicidad, la retroalimentación o reutilización del código desarrollado".(Ramírez, 2011)

"Es una metodología ágil centrada en potenciar las relaciones interpersonales como clave para el éxito en desarrollo de software, promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los desarrolladores, y propiciando un buen clima de trabajo. XP se basa en realimentación continua entre el cliente y el equipo de desarrollo, comunicación fluida entre todos los participantes, simplicidad en las soluciones implementadas y coraje para enfrentar los cambios. XP se define como

especialmente adecuada para proyectos con requisitos imprecisos y muy cambiantes, y donde existe un alto riesgo técnico".(Beck & Andres, 2004)

"Los principios y prácticas son de sentido común pero llevadas al extremo, de ahí proviene su nombre. Kent Beck, el padre de XP, describe la filosofía de XP sin cubrir los detalles técnicos y de implantación de las prácticas. Posteriormente, otras publicaciones de experiencias se han encargado de dicha tarea".(Beck & Andres, 2004).

Ventajas

- La simplicidad de esta metodología consiste en desarrollar el sistema que realmente se necesita, esto quiere decir que se trabaja con los requerimientos del usuario.
- El FeedBack es un punto fuerte en esta metodología ya que, al contar con pequeñas entregas o pruebas frecuentes, proporciona un flujo de retroalimentación con la que se detecta posibles fallos antes de implementar el sistema.
- La comunicación es primordial, tanto el cliente como los desarrolladores forman un equipo y en el transcurso del desarrollo del software el cliente puede ayudar a resolver problemas.(Beck & Andrés, 2007).

Desventajas

- Una de las desventajas es que resulta muy complicado planear el proyecto y establecer el costo y la duración de este.
- Es difícil aplicar esta metodología a proyectos de gran magnitud, sin embargo, el proyecto puede dividirse en pequeños módulos para ser tratado.
- Se dificulta la medición del proyecto y de sus avances.

1.2.2. Artefactos UML que utiliza la metodología

Los artefactos son utilizados por utilizan los artefactos para realizar sus actividades y de la misma manera producen artefactos u objetos al cumplir con dichas tareas por lo tanto un artefacto es el resultado de un producto de trabajo en un proceso. Así mismo los artefactos son responsabilidad de cada persona que lo realice por lo tanto cada persona tiene responsabilidades específicas.(«Artefactos y UML en el Proceso Unificado», 2012)

Los artefactos de la metodología XP son:

- Historia de Usuarios.
- Casos de Uso.
- Diagramas de Secuencia.

1.2.2.1. Historias de Usuarios

Las historias de usuarios son los pasos a seguir para cumplir un proceso, se las expresa de tal manera que un usuario final las pueda entender, además da a conocer los pasos alternativos que realiza el sistema una vez que proceso comienza.

Las Historias de Usuario tienen varios aspectos importantes:

- **Tarjeta:** Identifica y detalla la historia de usuario.
- **Conversación:** Se reúne el programador con el cliente y hablan sobre los procesos que el cliente necesita.
- **Pruebas de Aceptación:** Una vez realizada la historia de usuario se realizan pruebas para ver si los pasos de la historia se cumplen.
- **Tareas de ingeniería:** Distribuye las actividades de cada uno de los desarrolladores y encargados del sistema, por lo que controla las

responsabilidades de cada uno haciendo que el trabajo se distribuya equitativamente.

- **Tarjetas CRC:** Contiene una información global de todo lo que implica el desarrollo, como: los responsables de cada actividad y toda la información de la historia de usuario.

Tabla 4 Pasos a seguir para llenar la tarjeta

Número	**Pasos**
1	Encontrar Clases
2	Encontrar responsabilidades
3	Definir colaboradores
4	Disponer las tarjetas

Nota: Esta tabla contiene los pasos llenar las tarjetas CRC.(Beck & Andrés, 2007)

Las siguientes historias de usuarios describen las principales funcionalidades del proyecto.

- **Registro de Datos**

Tabla 5 HU Registro de Datos

	Historias de Usuario	
	CU1	**Registro**
	Actores	Juntas, Público, Regantes
Secuencia	**Flujo Normal**	**Flujo Alternativo**
1	El usuario ingresa a Registro	
2	El sistema proporciona el formulario de Registro	
3	El usuario completa el formulario y da clic en registrarse	
4	El sistema guarda en la BBDD	El sistema encuentra falta de datos en el formulario de registro y lanza una alerta
5	El usuario puede empezar a hacer uso del Padrón	

Nota: Esta tabla describe la Historia de Usuario de Registro de Datos

El flujo corresponde a los actores Juntas, público y regantes, los cuales tienen acceso a la creación de un registro que les permita utilizar en lo posterior el software.

- **Carga de Datos**

Tabla 6 HU Carga de Datos

	Historias de Usuario	
	CU	**Carga de datos**
	Actores	Administrador
Secuencia	**Flujo Normal**	**Flujo Alternativo**
1	El usuario ingresa a carga de datos	
2	El sistema proporciona dos opciones Importar Datos y Exportar Datos	
3	Si el usuario selecciona Importar datos	
4	El sistema proporciona Buscar Datos	
5	El usuario selecciona Buscar datos	
6	El sistema proporciona dos opciones Aceptar y Cancelar	
7	Si el usuario selecciona Aceptar	
8	El sistema importa los datos	El sistema da un error de importación de datos y muestra un mensaje de alerta
9	Si el usuario selecciona Cancelar	
10	El sistema se mantiene en Buscar datos	
11	Si el usuario selecciona Exportar Datos	
12	El sistema proporciona Buscar donde exportar	
13	El usuario selecciona Buscar donde exportar	
14	El sistema proporciona dos opciones Aceptar y Cancelar	

Secuencia	Flujo Normal	Flujo Alternativo
15	Si el usuario selecciona Aceptar	
16	El Sistema Exporta los datos	
17	Si el usuario selecciona Cancelar	
18	El sistema permanece en Buscar donde exportar	

Nota: Esta tabla describe la Historia de Usuario de Carga de Datos

El flujo muestra como el actor administrador puede realizar la carga de datos en caso de ser necesario y cuáles son los pasos que el sistema genera.

- **Reporte**

Tabla 7 HU Reporte

	Historias de Usuario	
	CU	**Consulta**
	Actores	Público, Administrador, Regante, Juntas
Secuencia	**Flujo Normal**	**Flujo Alternativo**
1	El usuario ingresa a reporte	
2	El sistema proporciona la opción de reporte en función de tipo de usuario: Reporte_Administrador, Consulta_regantes, Regantes_juntas, Consulta_juntas, Consulta_público	
3	El usuario selecciona el reporte correspondiente	
4	El sistema carga los datos desde la BBDD y proporciona varias opciones: Exportar y Atrás	El sistema no encuentra los datos y lanza un mensaje de alerta
5	Si el usuario selecciona Exportar	
6	El sistema muestra las opciones de exportar donde se elige el lugar y el formato a exportar.	
7	Si el usuario selecciona Atrás	
8	El sistema vuelve a "seleccione el tipo de reporte"	
9	Si el sistema selecciona Aceptar	
10	El sistema genera el Reporte	

Nota: Esta tabla describe la Historia de Usuario de Reporte

Este flujo lo utiliza todo tipo de actor y se refiere a la creación del de reportes que ayudarán a que la información sea más legible.

- **Mantenimiento**

Tabla 8 HU Mantenimiento

	Historias de Usuario	
	CU1	**Mantenimiento del sistema**
	Actores	Administrador
Secuencia	**Flujo Normal**	**Flujo Alternativo**
1	El usuario accede al mantenimiento del sistema	
2	El sistema proporciona dos opciones: Gestionar Categoría y BackUp de BBDD	
3	Si el usuario elige Gestionar categoría	
4	El sistema proporciona dos opciones: Ingresar Nuevo Registro y Actualizar Registro	
5	Si el usuario elige Ingresar Nuevo Registro	
6	El sistema proporciona dos opciones Guardar o Cancelar	
7	Si el usuario selecciona Guardar	
8	El sistema Guarda el Nuevo Registro	El sistema no puede guardar el nuevo registro por datos inconsistentes y emite un mensaje de error
9	Si el usuario elige Actualizar Registro	
10	El sistema carga los datos	El sistema no encuentra datos a cargar y genera un mensaje alerta
11	El usuario actualiza los registros	
12	El sistema proporciona dos opciones Guardar o Cancelar	

Secuencia	Flujo Normal	Flujo Alternativo
13	Si el usuario selecciona Guardar	
14	El sistema Guarda y Actualiza el Registro	
15	Si el usuario selecciona BackUp BBDD	
16	El sistema presenta dos opciones Generar o Cancelar	
17	Si el usuario selecciona Generar	
18	El sistema Genera el BackUp	

Nota: Esta tabla describe la Historia de Usuario de Mantenimiento

El flujo presenta los pasos obligatorios a seguir cuando un actor administrador realiza un backup de los datos.

1.2.3. Casos de Uso

Los casos de uso sirven para presentar de manera básica y simple el funcionamiento de un sistema desde la perspectiva del usuario final de tal manera que sea comprensibles, por lo que se puede utilizar como manuales de usuarios y también para la documentación del desarrollo de un sistema.(Fernández Alarcón, 2006, p.132)

Las siguientes ilustraciones representan el funcionamiento del sistema.

- **Caso de uso #1: Registro de Datos**

Este caso de uso muestra los pasos que sigue el sistema después de que el usuario ingresa los datos en el Formulario de Registro y si está correcto guarda en la Base de Datos.

CU Registro de Datos

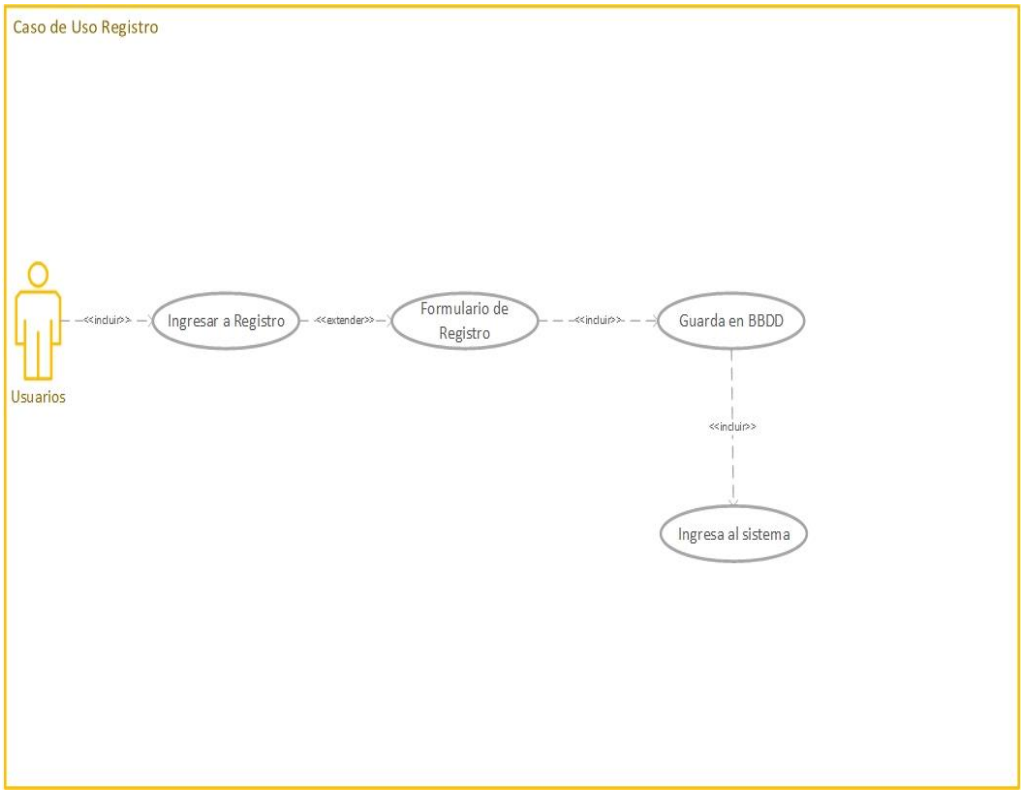


Figura 1 CU Registro de Datos

Elaborado por: Norman Chuquirima, Miguel Silva. (2018)

- **Carga de Datos**

Este caso de uso muestra los permisos que tiene el usuario Administrador y cuáles son las acciones que puede realizar, en este caso podrá Importar y Exportar Datos, con lo que el sistema negará o aprobará las acciones.

CU Carga de Datos

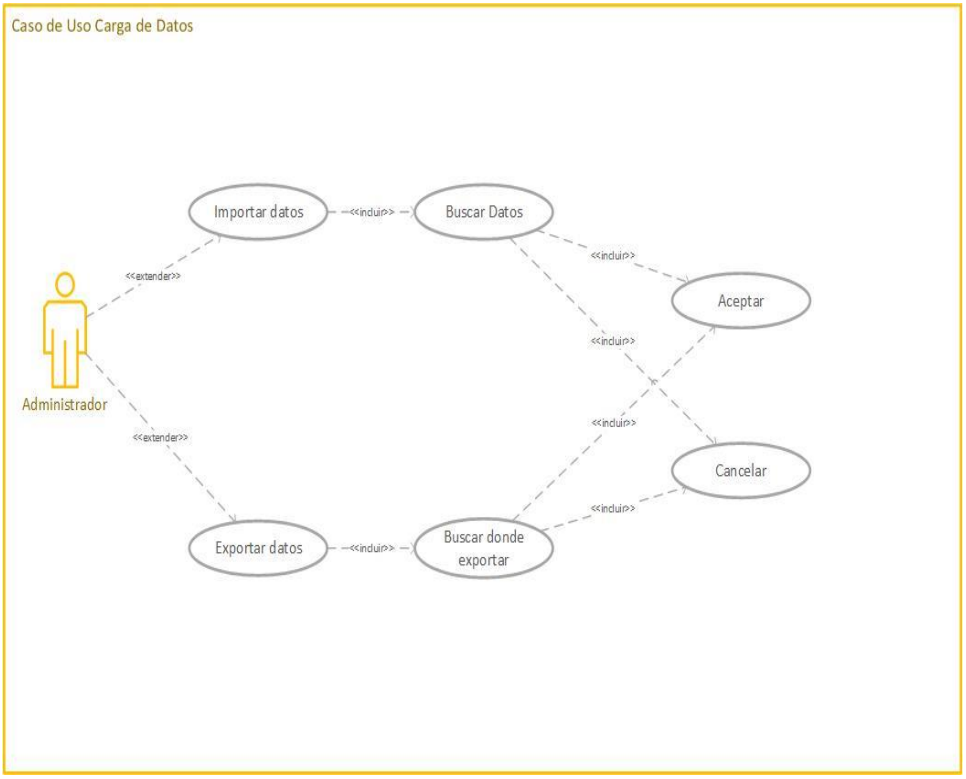


Figura 2 CU Carga de Datos

Elaborado por: Norman Chuquirima, Miguel Silva. (2018)

- **Reporte**

El usuario ingresa al módulo de reportes y selecciona el tipo de reporte, el sistema realiza un pedido a la Base de Datos con los requerimientos que dependen del tipo de reporte.

CU Reporte

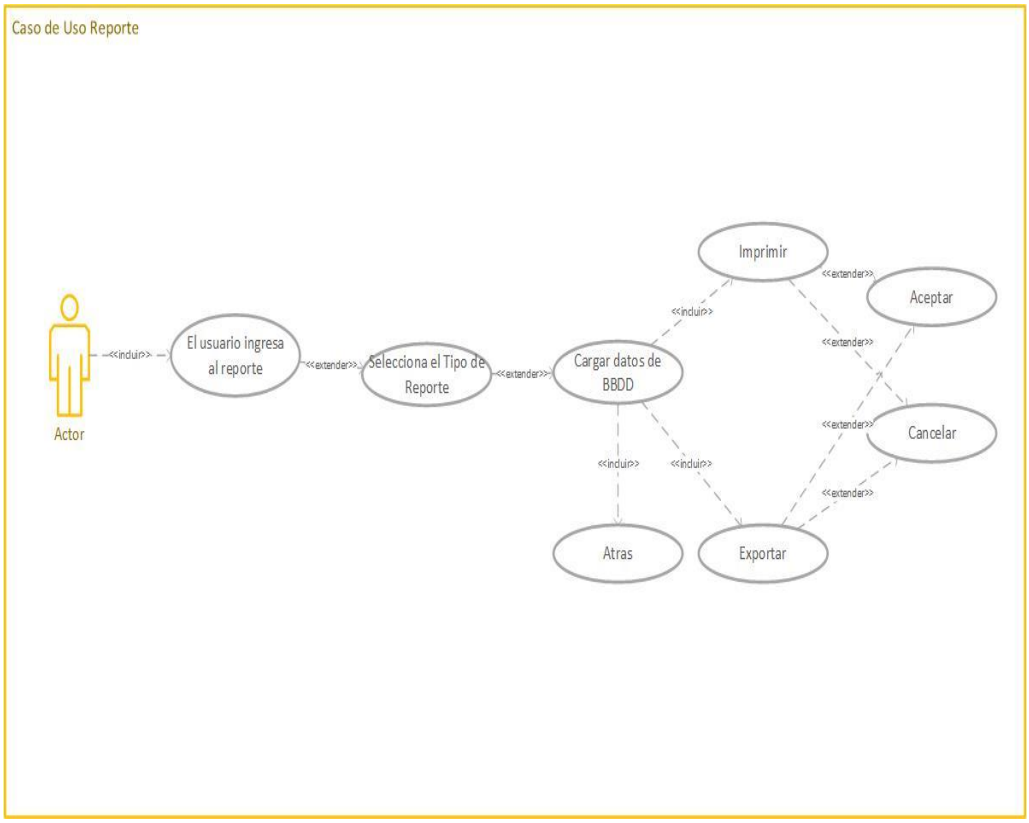


Figura 3 CU Reporte

Elaborado por: Norman Chuquirima, Miguel Silva. (2018)

- **Mantenimiento**

El usuario administrador ingresa a este módulo para realizar diferentes opciones de mantenimiento como generar un BackUp de la Base de Datos.

CU Mantenimiento

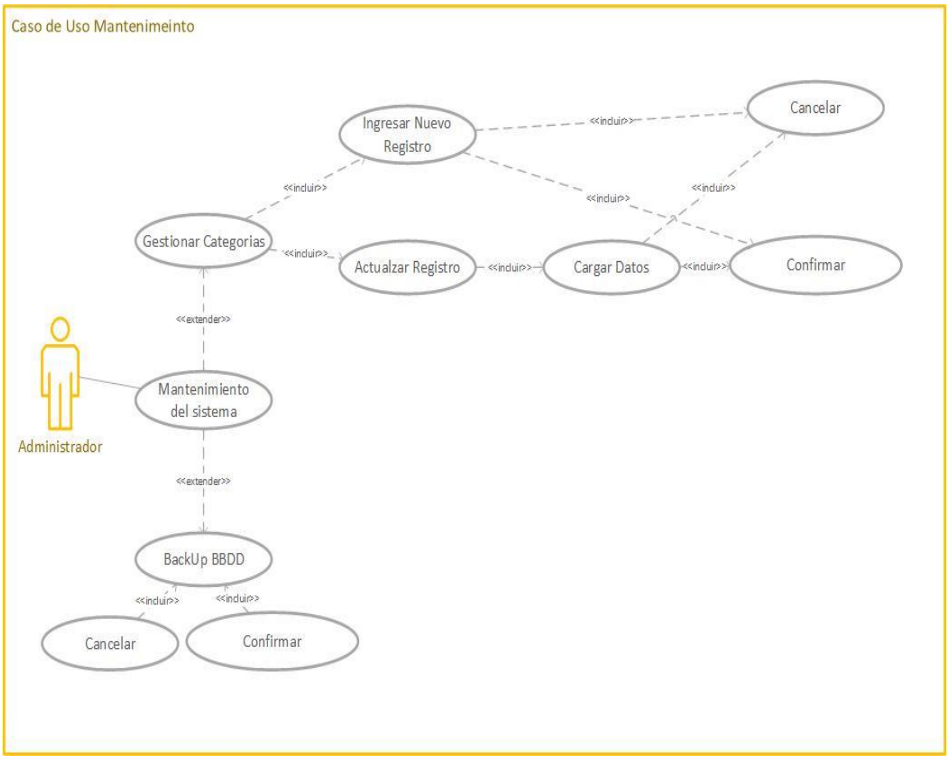


Figura 4 CU Mantenimiento

Elaborado por: Norman Chuquirima, Miguel Silva. (2018)

1.2.4. Diagrama de Secuencias

Los diagramas de secuencia corresponden con mucho más detalle a escenarios y sub-escenarios de proyecto, se los puede perfeccionar un poco más en la etapa de desarrollo, a continuación se muestran los más representativos del proyecto.(«Qué es un diagrama de secuencia», 2017)

- **Diagrama de secuencia - Registro**

Diagrama de secuencia Registro

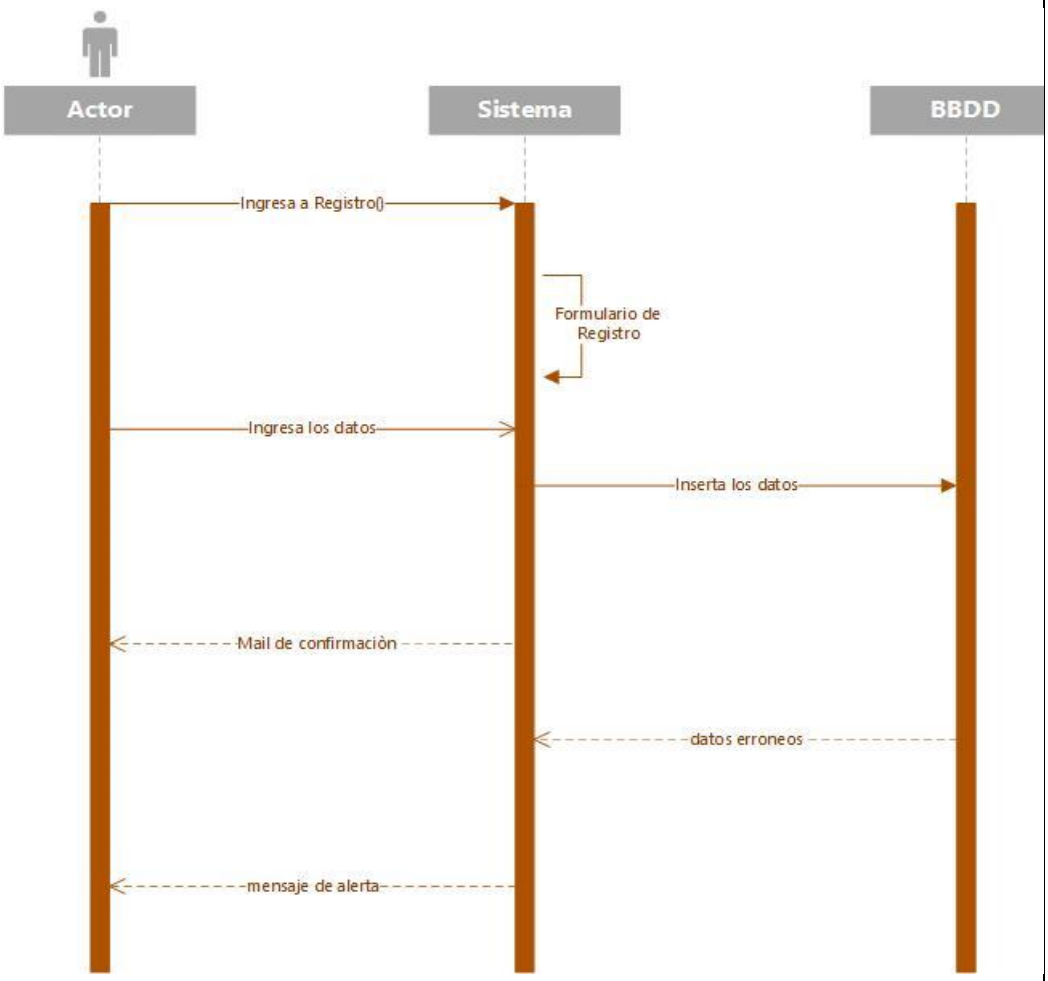


Figura 5 Diagrama de Secuencia - Registro

Elaborado por: Norman Chuquirima, Miguel Silva. (2018)

El diagrama de registro realiza el camino que realiza el sistema cuando un usuario se registra por primera vez.

- **Diagrama de secuencia – Carga de Datos**

Diagrama de Secuencia Carga de Datos

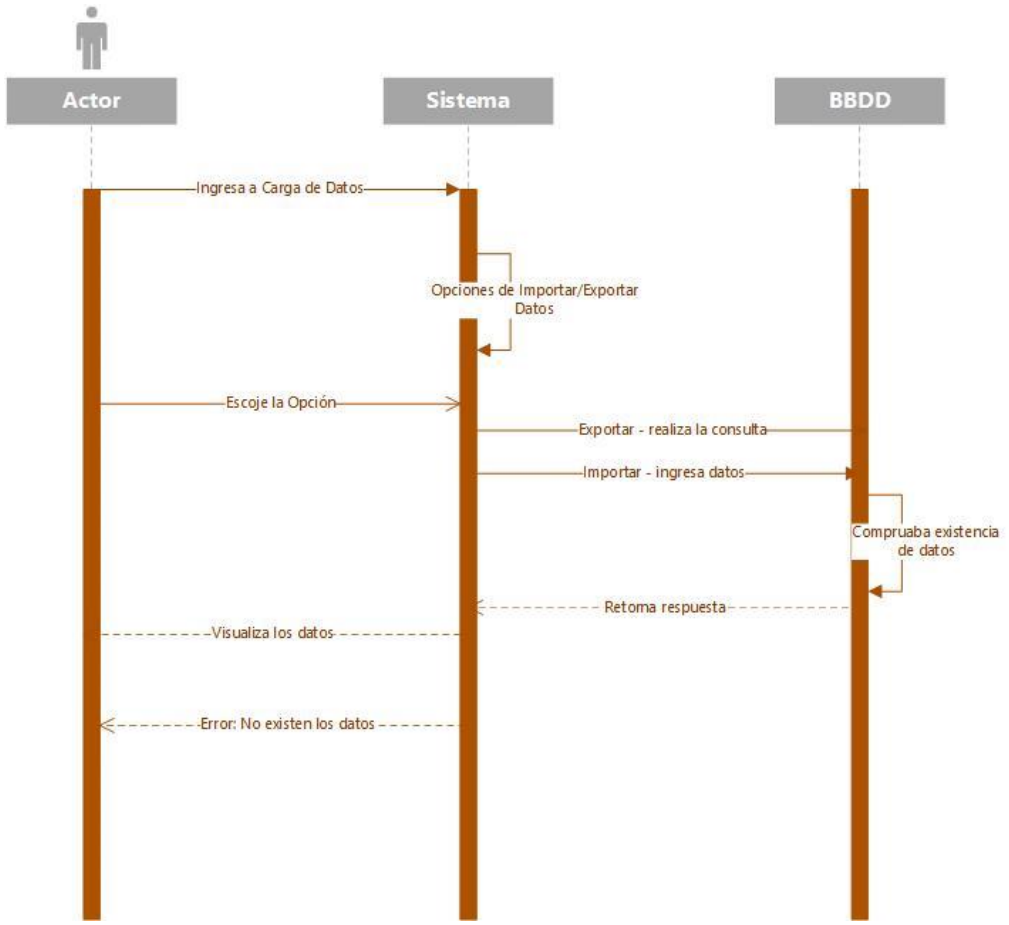


Figura 6 Diagrama de Secuencia – Carga de Datos

Elaborado por: Norman Chuquirima, Miguel Silva. (2018)

El diagrama de carga describe los pasos internos que el sistema web debe realizar para poder realizar la carga de datos.

- **Diagrama de secuencia - Reporte**

Diagrama de Secuencia Reporte

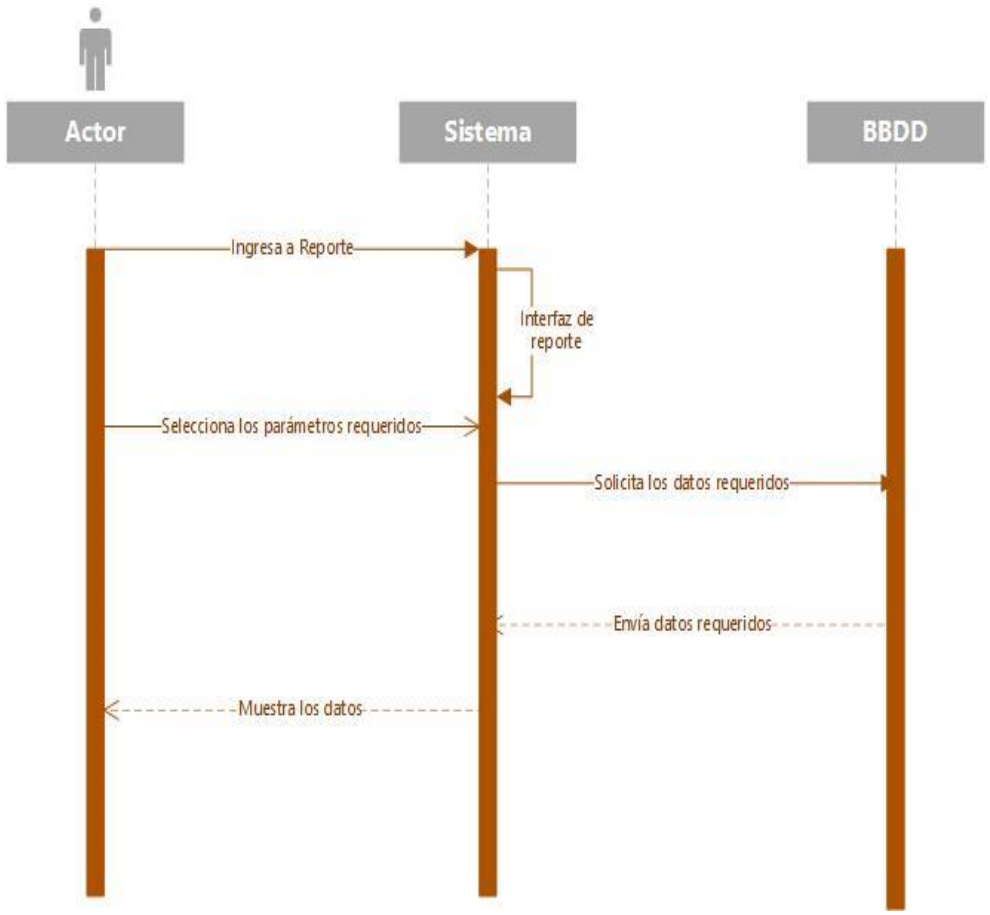


Figura 7 Diagrama de Secuencia - Reporte

Elaborado por: Norman Chuquirima, Miguel Silva. (2018)

El Diagrama de Reporte muestra los pasos que el sistema web realiza para poder realizar un reporte.

- **Diagrama de secuencia – Mantenimiento**

Diagrama de Secuencia Mantenimiento

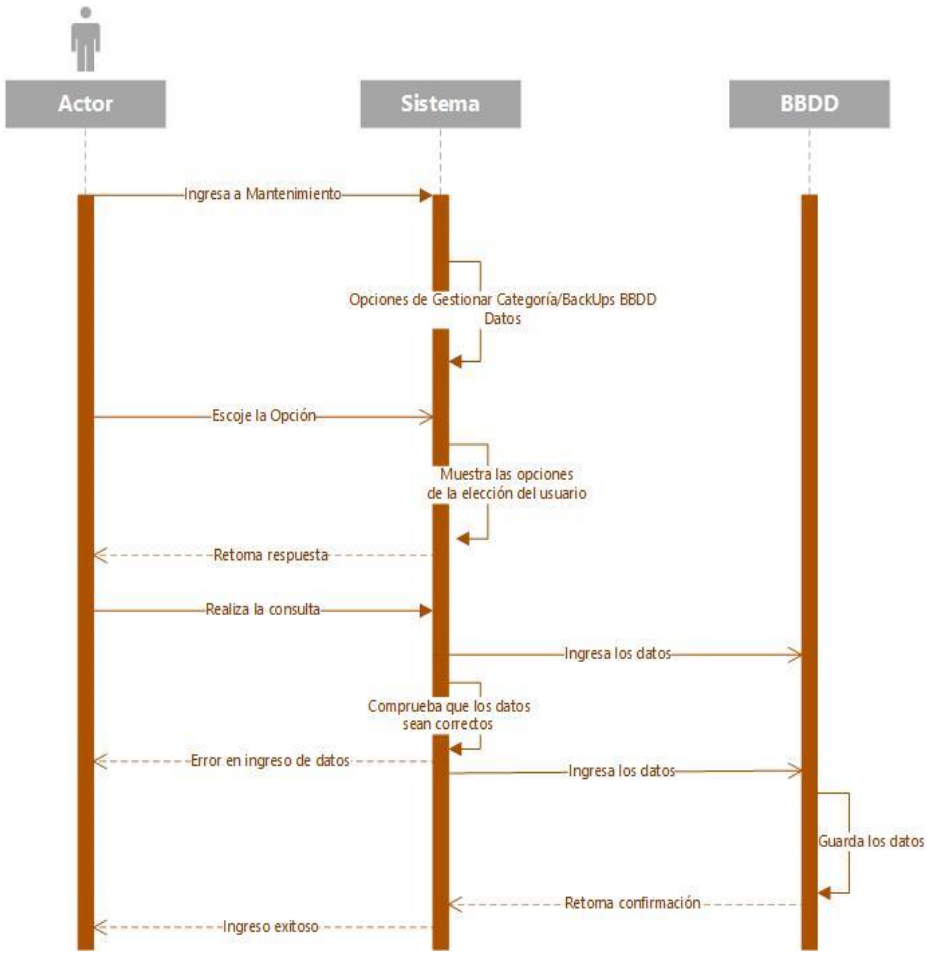


Figura 8 Diagrama de Secuencia - Mantenimiento

Elaborado por: Norman Chuquirima, Miguel Silva. (2018)

El Diagrama de mantenimiento muestra los pasos que el sistema web sigue para poder realizar un mantenimiento del sistema.

1.3. Herramientas de construcción

A continuación se muestran herramientas que ayudaron a construir el sistema.

1.3.1. PowerDesigner.

"Es una herramienta para análisis, diseño inteligente y construcción sólida de una base de datos, tiene un desarrollo orientado a modelos de datos a nivel físico y conceptual, que da a los desarrolladores del modelo Cliente/Servidor la más firme base para aplicaciones de alto rendimiento".(Pressman, Ince, Ojeda Martín, & Joyanes Aguilar, 2004)

La herramienta ayuda a un correcto diseño tanto del modelo físico como lógico de la base de datos, proporcionando las herramientas necesarias para ello, además de contar con diversas técnicas de modelado que haciendo un uso correcto de la misma permite realizar un diseño formal de la base de datos que es lo que se realizó en el proyecto.

El uso de ilustraciones es otra ventaja que tiene PowerDesigner, una vez dominada la herramienta, el desarrollador puede personalizar el menú de opciones y herramientas, colocado atajos que hacen más rápido el trabajo.

"El objetivo de PowerDesigner con el modelamiento de datos, UML y de negocio, se ha puesto a prueba en el desarrollo, por lo que es una correcta y segura elección para el modelamiento de la base de datos del proyecto".(Pressman et al., 2004)

1.3.2. Netbeans.

Es una herramienta que permite desarrollar programas que soporta diferentes lenguajes de programación, entre estos JAVA; Esta herramienta permite compilar, depurar y ejecutar los sistemas, además de ofrecer varios módulos con diferentes funcionalidades que hacen de este IDE una de las herramientas más poderosas y utilizada para el desarrollo de software.(«Bienvenido a NetBeans y www.netbeans.org, Portal del IDE Java de Código Abierto.», s. f.).

1.3.3. PostgresSQL.

“Es un sistema de gestión de base de datos relacional orientada a objetos y libre, publicado bajo la licencia BSD. Tal y como funcionan otros proyectos de código abierto el desarrollo de PostgreSQL no es manejada por una sola empresa, sino que es dirigida por una comunidad de desarrolladores y organizaciones comerciales”.(«PostgreSQL - EcuRed», s. f.)

PostgreSQL soporta funciones que retornan "filas", donde la salida puede tratarse como un conjunto de valores que pueden ser tratados igual a una fila retornada por una consulta y gracias a esto permite tener herencia de tablas e integridad transaccional.

“Una de las características principales de esta herramienta es que maneja un sistema denominado MVCC (Acceso Concurrente Multiversión),esto permite que mientras un usuario escriba en una tabla, otros accedan a la misma tabla sin tener ningún inconveniente ni bloqueos, esto hace que cada usuario tenga una visión consistente de lo último que se hizo en dicha tabla, lo que ayuda al proyecto, ya que al ser un sistema web, varios usuarios pueden ingresar a la misma vez y modificar los datos o actualizarlos sin ningún inconveniente”.(«PostgreSQL - EcuRed», s. f.).

1.3.4. Java Server Faces JSF.

Está diseñada para poder simplificar el desarrollo web en JAVA, siendo esta una tecnología potente y flexible. Es respaldado por todo el mundo de JAVA por esto se la considera un estándar en la industria, y se han generado entornos de desarrollo de alta productividad y una gran variedad de componentes de terceros.(«JSF - EcuRed», s. f.)

1.3.5. JAVA.

“Es un lenguaje de programación orientado a objetos, en mayo de 1995 fue presentado oficialmente por la empresa desarrolladora Sun Microsystems en la conferencia

SunWorld, este lenguaje de programación nació por la necesidad de tener alguna tecnología que permita el desarrollo de programas capaces de ejecutarse en entornos distribuidos y heterogéneos, esto quiere decir, programas que fueran capaces de ejecutarse en muchas plataformas, tanto de Hardware como de Software.”(P. Pablo Garrido Abenza, s. f.)

Tabla 9 Características de JAVA

1	Sencillo
2	Multiplataforma
3	Alto Rendimiento
4	Robusto
5	Orientado a Objetos
6	Distribuido
7	Concurrente

Nota: Esta tabla contiene algunas características de JAVA(P. Pablo Garrido Abenza, s. f.)

1.3.6. Glassfish

Es un servidor de aplicaciones de código libre, compatible con la implementación de las aplicaciones Java EE desarrollado por Sun Microsystems.(Urbano López, 2015), Este servidor soporta las últimas versiones de las tecnologías JSP,JSF, Servlets entre otras.

1.3.7. Framework

Es una infraestructura de código que facilita la elaboración de aplicaciones, que ayuda y ahorra mucho esfuerzo a los desarrolladores. En Java hay una gran cantidad de frameworks web a la disposición: JSF, Struts, SpringMVC, VRaptor, Wicket, entre otros.(MNDEZ GONZLEZ, 2016)**.**

1.3.8. Enterprise JavaBeans

Es una plataforma para construir aplicaciones de negocios portables, reutilizables y escalables usando lenguaje de programación JAVA.(Panda, Rahman, & Lane, 2007)

1.3.9. Primefaces

"Es una librería de componentes visuales de código abierto para el conjunto Java Server Faces 2.0 desarrollada y mantenida por Prime Technology. Su objetivo principal es ofrecer un conjunto de componentes para facilitar la creación y diseño de aplicaciones web. Los componentes de PrimeFaces cuentan con soporte nativo de Ajax, pero no se encuentra implícito, de tal manera que se tiene que especificar que componentes se deben actualizar al realizar una petición proporcionando así mayor control sobre los eventos.

Cuenta también con un módulo adicional TouchFaces para el desarrollo de aplicaciones web para dispositivos móviles con navegadores basados en WebKit". (Fernando Pech-May, Mario A. Gomez-Rodriguez, Luis A. de la Cruz-Diaz, & Salvador U. Lara-Jeronimo, s. f.)

Tabla 10 Algunas características de Primefaces

PRINCIPALES CARACTERÍSTICAS DE PRIMEFACES	
1	Soporte nativo de Ajax, incluyendo Push/Coment
2	Kit para crear aplicaciones web móviles
3	Es compatible con otras librerías de componentes como Jboss RichFaces
4	Uso de JavaScript no intrusivo
5	Es un proyecto open source, activo y estable

Nota: Esta tabla contiene algunas características de PrimeFaces.

1.3.10. Balsamiq

Es una herramienta de enmarcado que ayuda a trabajar más rápido y con más inteligencia mostrando al usuario final un boceto de cómo quedaría el sistema con lo que reproduce la experiencia de diseño, pero de una manera más simple.(«Balsamiq. Rápido, efectivo y divertido software de wire framing. | Balsamiq», s. f.)

1.3.10.1. Mockups

Es una herramienta visual de diseño, que se utiliza para presentar prototipos de un sistema, que ayuda al equipo desarrollador a obtener ideas claras de cómo se verá el sitio web.

El diseño del prototipo permite mostrar al cliente de forma visual los elementos como: imágenes, texto, íconos además de su distribución, dando una idea clara del sitio.

Al ser una representación visual, la funcionalidad del sitio viene expresada en gráficos sin ninguna interacción entre sí, por ejemplo, cuando se quiere representar un carrusel, solo se coloca un ícono de imagen y flechas a los lados .(«Diferencias esenciales entre un wireframe, mockup y prototipo», 2016)

- **Pantalla de inicio (Home)**

Iniciar sesión del Padrón Web

Figura 9 Mockups Home Padrón Web

Elaborado por: Norman Chuquirima, Miguel Silva. (2018)

Interfaz amigable con el usuario con componentes gráficos de tamaño mediano; logo de la Universidad Politécnica Salesiana Sede Cayambe, título principal "PADRON DE USUARIOS PARA SISTEMAS DE RIEGO COMUNITARIO"; Carrusel de imágenes administrable; información principal acerca del padrón; Información de contacto; panel de logeo.

- **Tipos de usuarios (Roles de usuario)**

La aplicación web va a contar con 4 tipos de usuario:

- **Regantes**

Perfil Regante

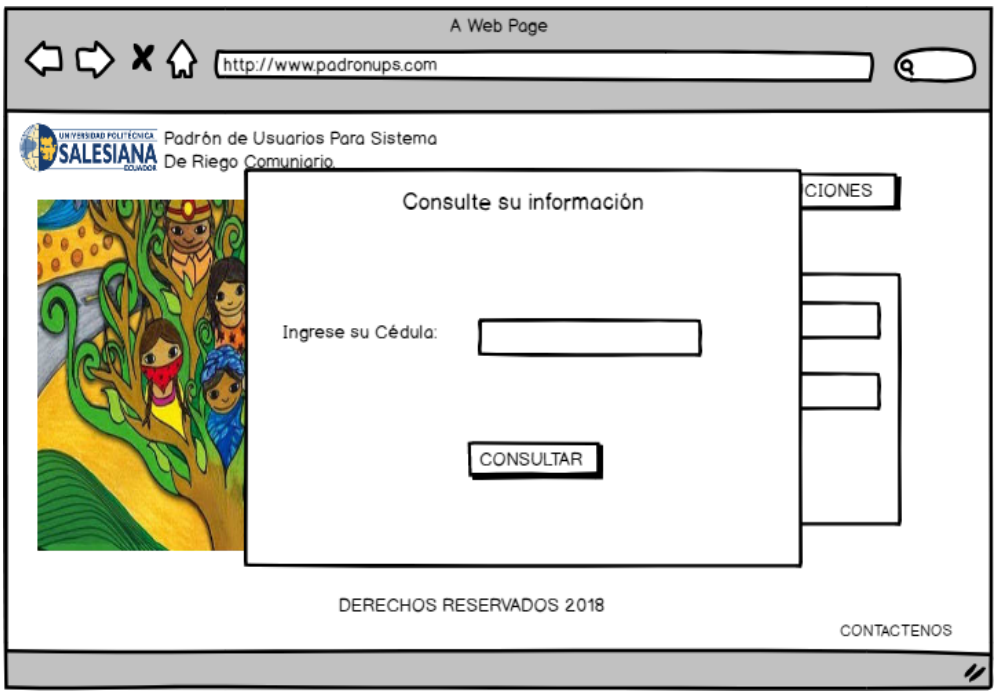


Figura 10 Mockups Perfil Regante

Elaborado por: Norman Chuquirima, Miguel Silva. (2018)

Este mockup muestra la interfaz de los usuarios regantes van a consultar su ficha de registro, van a poder imprimirla mediante su número de cédula.

- **Presidente de Junta**

Perfil Presidente de Junta

Figura 11 Mockups Presidente de Junta

Elaborado por: Norman Chuquirima, Miguel Silva. (2018)

Estos usuarios van a poder ingresar las fichas, realizar consultas, imprimir datos, eliminar y actualizar ciertos datos permitidos.

- **Administrador**

Perfil Adminstrador

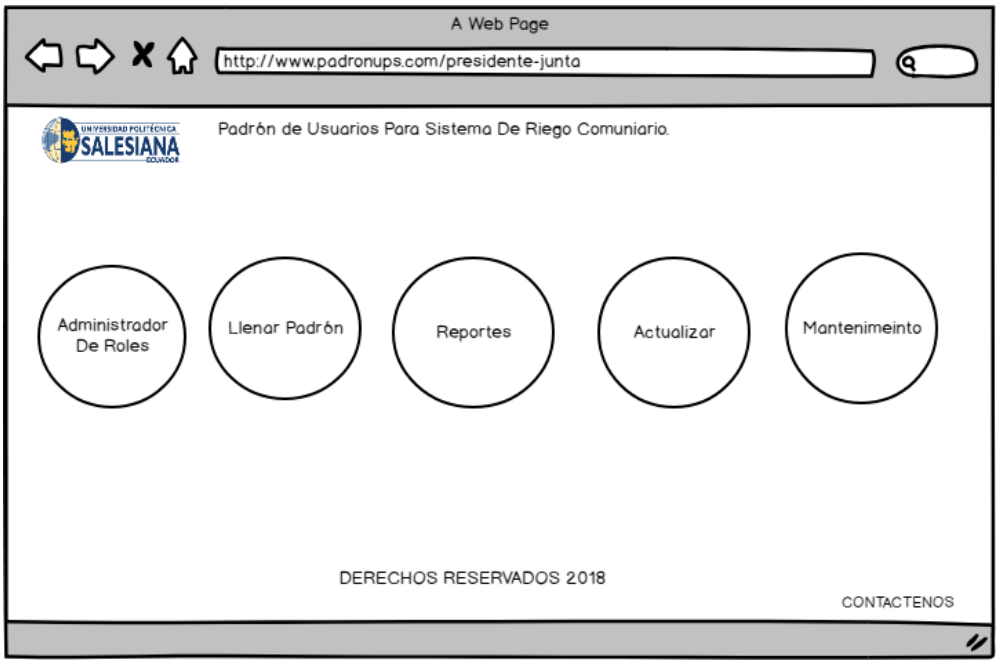


Figura 12 Mockups Administrador

Elaborado por: Norman Chuquirima, Miguel Silva. (2018)

Este usuario va a tener todos los permisos de tal manera que podrá, actualizar, consultar, eliminar, imprimir datos.

- **Público**

Perfil Público

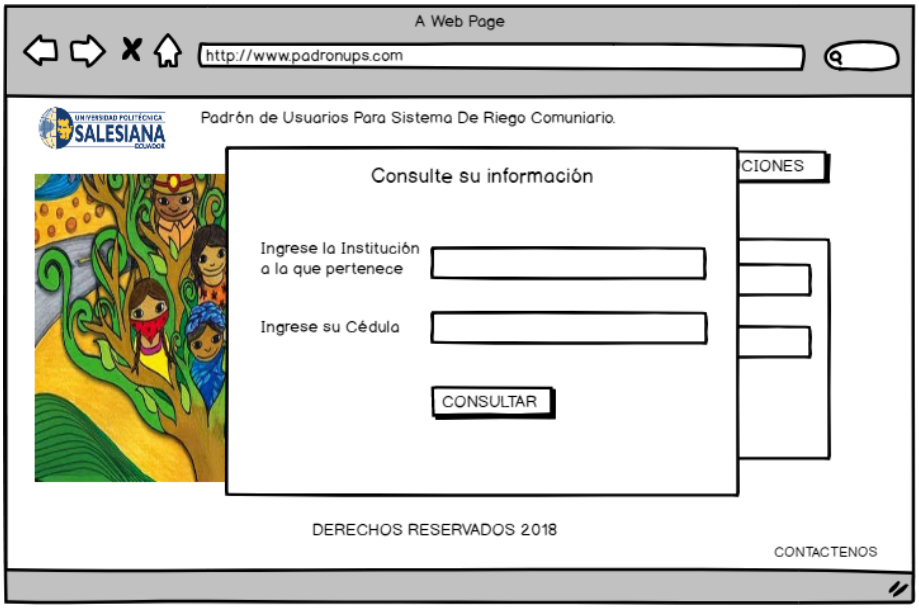


Figura 13 Mockup Perfil Público

Elaborado por: Norman Chuquirima, Miguel Silva. (2018)

Este usuario va a tener la opción para buscar, descargar reportes en pdf, he imprimir.

- **Mantenimiento**

Permitirá generar un Backup para tener un histórico de los datos obtenidos en un tiempo determinado para que la información sea más confiable y no se pierdan datos.

1.4. Modelo Vista Controlador

Es un modelo de diseño que sirve para el desarrollo de interfaces gráficas de usuarios, para esto el Modelo Vista Controlador divide a las aplicaciones en tres:

1.4.1. Modelo

Contiene las clases creadas de acuerdo con cada entidad de la base de datos, la cual permite la interacción al gestor de base de datos mediante el modelo DAO y la interacción con la Vista y el controlador mediante el modelo VO.

1.4.2. Vista

Hace referencia a los componentes gráficos que son la salida de información hacia el usuario de la aplicación. En el caso del sistema los componentes se manejan con JSP.

1.4.3. Controlador

Son controles que se encuentran disponibles al usuario mediante la interfaz gráfica. Estos controles los usa el usuario a través del evento como hacer clic a un botón, presionar una tecla, mover el mouse, entre otros. El controlador es quien altera el modelo en el momento de que el usuario realiza el evento.

Entonces, el Modelo Vista Controlador es una división de objetos de dominio que modelan la abstracción del mundo real que la aplicación representa, con objetos de presentación que proporcionan una interfaz gráfica, por lo cual se le presenta la información al usuario.(Flórez Fernández, 2012).

1.5. Patrón de modelo DAO

Los componentes de acceso a datos (DAO) encapsulan la tecnología empleada para acceder a la capa de datos, separando completamente la lógica de negocio de la lógica de acceso a datos. De esta forma se garantiza la portabilidad del sistema en los diferentes motores de base de datos (Oracle, SqlServer, etc). (Jack 2.0, 2014)

1.6. Patrón de modelo VO

El patrón de modelo VO contiene los métodos Get y Set que permiten la separación de las capas presentación, acceso del resto de capas (lógica del negocio). Permitiendo así un sistema fácil de mantener y evolucionar.(Jack 2.0, 2014)

1.7. HTML5

Es una evolución de Html. Retoma los grandes principios, los modifica para mejorar ciertos aspectos. Ha sido diseñado para asegurar una compatibilidad hacia atrás con todo lo que he desarrollado hasta el presente, en términos de publicación sobre la web. A pesar de que el código es menos formal que Xhtml5, la exigencia de un código propio, respetuoso con las reglas elementales del etiquetado, continua existiendo.(Van Lancker, 2012)

1.8. Javascript

Con este lenguaje de programación se realizó la creación del sistema web que interactúa con el usuario, generalmente las páginas web no se consideran dinámicas, por lo que este lenguaje ayuda a que el usuario encuentre una web con cierto dinamismo y que por lo tanto tenga un mejor resultado al momento de su presentación, al ser este proyecto enfocado a un público específico, este lenguaje ayuda a mejorar la interacción con el usuario final.(Mohedano, Saiz, & Salazar Román, 2012)

2. CAPÍTULO II

2.1. Análisis y Diseño

2.1.1. Requerimientos funcionales del sistema

"Los requerimientos funcionales describen lo que el sistema debe hacer. Estos requerimientos dependen del tipo de software que se desarrolle, de los usuarios del sistema y del enfoque general tomado por la organización al redactar requerimientos. Cuando se expresan como requerimientos del usuario, habitualmente se describen de una forma bastante abstracta. Sin embargo, los requerimientos funcionales del sistema describen con detalle la función de éste, sus entradas y salidas, excepciones, entre otros".(Sommerville & Alfonso Galipienso, 2005)

2.2. Registro de información

2.2.1. Registro de datos de la familia

Se registran los datos personales de los usuarios pertenecientes a las juntas de riego de cada comunidad, datos del conyugue y datos de los hijos.

Los datos a registrar del usuario de riego de la junta son: Fecha de ingreso de datos; Provincia; Cantón; Parroquia; Comunidad; Nombres; Apellidos; Numero de cedula; Edad; La actividad principal a la que se dedica; Número de teléfono (Convencional o Celular); Sexo (Masculino o Femenino); Nivel de instrucción (Primaria, Secundaria, Superior, alfabetizado o ninguno); Estado civil (Soltero, Casado, Unión libre u otro); jefe de hogar (Si o No).

Los datos a registrar del conyugue son: Nombres; Apellidos; edad; Sexo (Masculino o Femenino); La actividad principal a la que se dedica; Nivel de instrucción (Primaria, Secundaria, Superior, alfabetizado o ninguno);

Los datos a registrar de los hijos son: Nombres; Apellidos; Dependiente (Si o No); Edad; Sexo (Masculino o Femenino); La actividad principal a la que se dedica; Nivel de instrucción (Primaria, Secundaria, Superior, alfabetizado o ninguno);

2.2.2. Registro de la unidad productiva (Lote) y riego

Se registran los datos generales del lote, la superficie, el método de riego, el turno de riego, el valor de tarifa, la tenencia de la tierra y el estado de dicha tierra.

Los datos a registrar de la organización de riego son: Nombre de la organización de riego

Los datos a registrar del Lote son: Código de Lote; Número de lote; Sector; Nombre de la Acequia; Área total del lote (Hectáreas y metros cuadrados); Área con riego (Hectáreas y metros cuadrados); Método de riego (porcentaje de inundación, aspersión o goteo); en caso de método de riego aspersión el tipo de aspersor(1/2, 3/4, 1, 1 ¼, 2) y el número de aspersores de cada lote; Turno de riego (frecuencia, número de días y horas de turno); Valor de la tarifa de riego (En dólares, por turno, fijo mensual, fijo anual o por hectárea); tenencia de la tierra(Nombre del propietario, estado de la tierra).

2.2.3. Registro de sistema de producción Agrícola por lote

Se registran los cultivos por cada lote y la superficie que cada uno de estos ocupan **l**os datos a registrar de la producción agrícola son: Nombre del cultivo; Lote al que pertenece cada cultivo; Superficie de cultivo en metros cuadrados

2.2.4. Registro del sistema de producción pecuario

Se registra el número de las especies de producción pecuario e inventario de ganado. Los datos a registrar de producción pecuario son: Animales menores; Número de cada animal menor; Ganado; Número de ganado; producción diaria de leche(Litros/hato)

2.2.5. Registro general de ficha

Se registra el nombre del encuestador para el control al momento del ingreso de las fichas.

2.3. Operaciones

- Almacenamiento de todos los registros en la base de datos postgreSQL.
- La fecha actual del registro de información del propietario tomada del sistema (Edición deshabilitada).
- Validación del registro provincia (Lista despegable de provincias, letras mayúsculas).
- Validación del registro cantón (Lista desplegable de cantones, filtra cantón por provincia seleccionada, letras mayúsculas).
- Validación del registro parroquia (Lista despegable de parroquias, filtro de parroquias por cantón seleccionado, letras mayúsculas).
- Validación del registro comunidad (Lista despegable de las comunidades participantes, filtro de comunidades por parroquia seleccionada, letras mayúsculas).
- Validación de registro nombres de usuario de riego de la junta (cantidad de caracteres, letras mayúsculas, registro obligatorio).
- Validación de registro apellidos de usuario de riego de la junta (cantidad de caracteres, letras mayúsculas, registro obligatorio).
- Validación de registro número de cédula (cantidad de caracteres, validez de cédula ecuatoriana, registro obligatorio).
- Validación de registro edad (registro obligatorio, validar solo ingreso de números positivos, rango de 0 a 100).
- Validación de registro actividad principal (registro obligatorio, letras mayúsculas).

- Validación de registro número de teléfono casa (registro opcional, ingreso solo de números).
- Validación de registro número de teléfono celular (registro opcional, ingreso solo de números).
- Validación de registro sexo (registro obligatorio, se selecciona una sola opción).
- Validación de registro nivel de instrucción (Selecciona una sola opción, registro obligatorio).
- Validación de registro estado civil (registro obligatorio, escoger solo una de las opciones).
- Validación de registro jefe de hogar (registro obligatorio, seleccionar una sola opción).
- Validación de registro dependiente (registro obligatorio, escoger una sola opción).
- Validación de registro código de Lote (el código lo ingresa el administrador del sistema, registro obligatorio).
- Validación de registro número de lote (registro se crea automático por el sistema).
- Validación de registro sector (registro obligatorio, validación de caracteres especiales).
- Validación de registro acequia (registro obligatorio, validación de caracteres especiales).
- Validación de registro área total (registro obligatorio, ingreso puede ser en metros cuadrados o en hectáreas).

- Validación de registro área con riego (registro obligatorio, ingreso puede ser en metros cuadrados o en hectáreas, no debe sobre pasar las medidas del área total).
- Validación de registro método de riego (registro obligatorio, validar que el método de riego sea 100%)
- Validación de registro tipo de aspersor (registro obligatorio, validar que tipo de aspersor utiliza).
- Validación de registro número de aspersores que deja por lote (registro obligatorio cuando se seleccione método de riego aspersor).
- Validación de registro turno de riego, frecuencia (registro obligatorio, solo una opción).
- Validación de registro número de días (registro obligatorio, validar si según el tipo de frecuencia que escoja).
- Validación de registro horas/turno (registro obligatorio, validar que las horas vayan acorde con la frecuencia).
- Validación de registro valor tarifa (registro obligatorio, puede escoger más de un valor de tarifa).
- Validación de registro tenencia de la tierra (registro obligatorio, ingresa el número de cédula correspondiente).
- Validación de registro estado de la tierra (registro obligatorio, escoger solo un ítem).
- Validación de registro cultivos (registro obligatorio, lista desplegable de cultivos).
- Validación de registro producción pecuaria (registro obligatorio, producción pecuaria, validación solo números).

- Validación de registro inventario de ganado (registro obligatorio, validación solo números).
- Validación de registro producción diaria de leche (registro obligatorio validación de números decimales),
- Validación de registro nombre del encentador (requisito obligatorio).

2.4. Interfaz gráfica de usuario:

2.4.1. Reportes

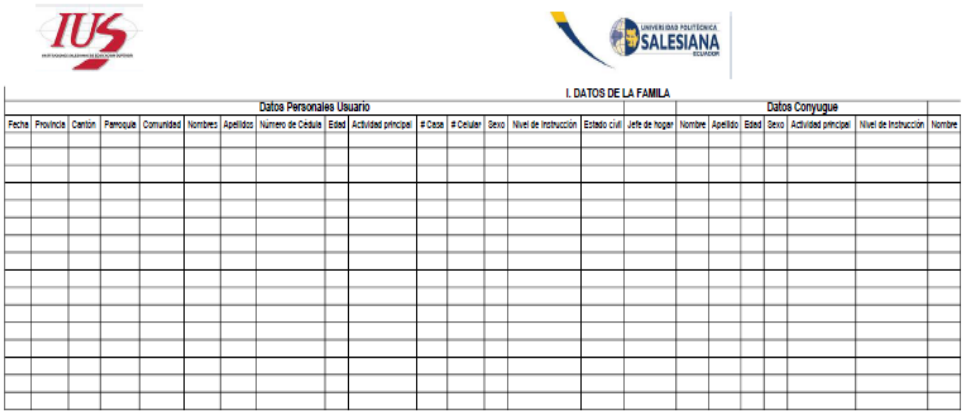
La extensión permite generar archivos que facilitarán la integración entre la aplicación de escritorio y el sistema web los mismos que mantendrán extensión en formato XML y XLS para tareas internas del software. La carga permitirá al sistema web actualizar los datos.

Los reportes que el sistema permitirá obtener son los siguientes:

- **Reporte Administrador**

Este reporte permite al usuario administrador obtener los datos de todas las fichas ingresadas en el sistema.

Plantilla de Reporte Regante



I. DATOS DE LA FAMILA

Datos Personales Usuario																	Datos Conyugue						
Fecha	Provincia	Cantón	Parroquia	Comunidad	Nombres	Apellidos	Número de Cédula	Edad	Actividad principal	# Casa	# Celular	Sexo	Nivel de instrucción	Estado civil	Jefe de hogar	Nombre	Apellido	Edad	Sexo	Actividad principal	Nivel de instrucción	Nombre

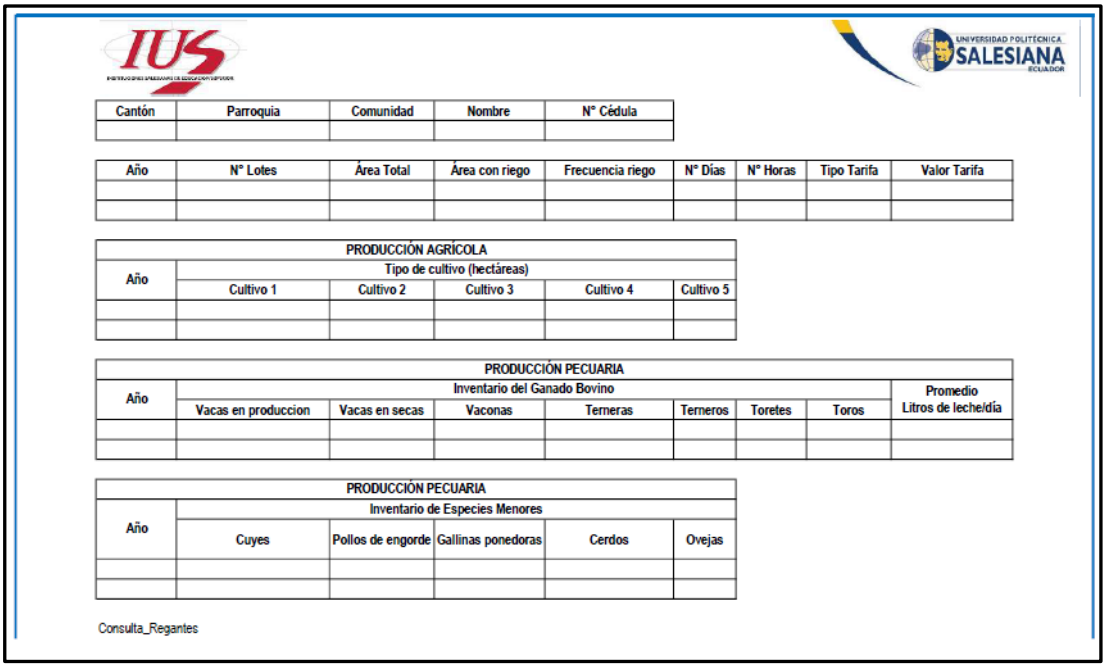
Figura 14 Perfil Reporte Regante

Elaborado por: Norman Chuquirima, Miguel Silva. (2018)

- **Reporte Regante**

Este reporte permite que el usurario regante pueda obtener algunos datos que les sean útiles para la comunidad como puede ser, el número de lotes por Cantón, Parroquia, Comunidad entre otros.

Plantilla de Reporte Regante



Cantón	Parroquia	Comunidad	Nombre	N° Cédula

Año	N° Lotes	Área Total	Área con riego	Frecuencia riego	N° Días	N° Horas	Tipo Tarifa	Valor Tarifa

PRODUCCIÓN AGRÍCOLA					
Año	Tipo de cultivo (hectáreas)				
	Cultivo 1	Cultivo 2	Cultivo 3	Cultivo 4	Cultivo 5

PRODUCCIÓN PECUARIA							
Año	Inventario del Ganado Bovino						Promedio Litros de leche/día
	Vacas en produccion	Vacas en secas	Vaconas	Terneras	Terneros	Toretes	Toros

PRODUCCIÓN PECUARIA					
Año	Inventario de Especies Menores				
	Cuyes	Pollos de engorde	Gallinas ponedoras	Cerdos	Ovejas

Consulta_Regantes

Figura 15 Perfil Reporte Regante

Elaborado por: Norman Chuquirima, Miguel Silva. (2018)

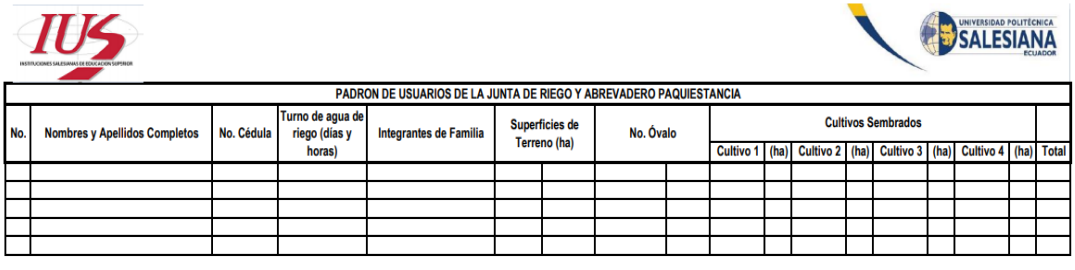
- **Reporte Juntas**

El reporte de juntas conta de dos partes que se detallan a continuación:

- **Reporte Juntas I**

El primer reporte de juntas muestra un detalle total de todos los datos de cada lote en cierta junta.

Plantilla de Reporte Juntas I



PADRON DE USUARIOS DE LA JUNTA DE RIEGO Y ABREVADERO PAQUIESTANCIA																		
No.	Nombres y Apellidos Completos	No. Cédula	Turno de agua de riego (días y horas)	Integrantes de Familia	Superficies de Terreno (ha)		No. Óvalo		Cultivos Sembrados									
									Cultivo 1	(ha)	Cultivo 2	(ha)	Cultivo 3	(ha)	Cultivo 4	(ha)	Total	

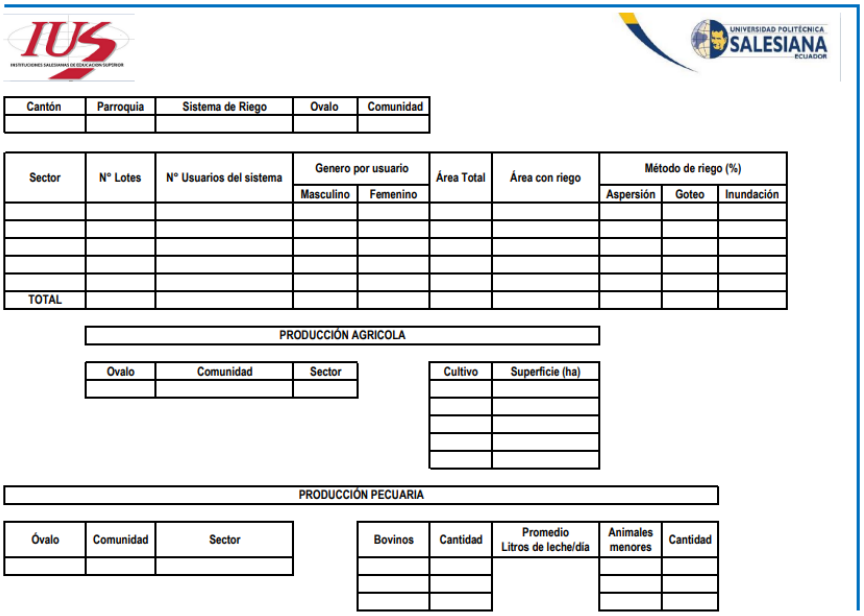
Figura 16 Primer Reporte de Juntas

Elaborado por: Norman Chuquirima, Miguel Silva. (2018)

- **Reporte Juntas II**

El segundo reporte de juntas muestra de manera más detallada datos como el número de lotes por Sector de cada Comunidad.

Plantilla de Reporte Juntas II



Cantón	Parroquia	Sistema de Riego	Ovalo	Comunidad

Sector	N° Lotes	N° Usuarios del sistema	Genero por usuario		Área Total	Área con riego	Método de riego (%)		
			Masculino	Femenino			Aspersión	Goteo	Inundación
TOTAL									

PRODUCCIÓN AGRICOLA

Ovalo	Comunidad	Sector

Cultivo	Superficie (ha)

PRODUCCIÓN PECUARIA

Óvalo	Comunidad	Sector

Bovinos	Cantidad	Promedio Litros de leche/día	Animales menores	Cantidad

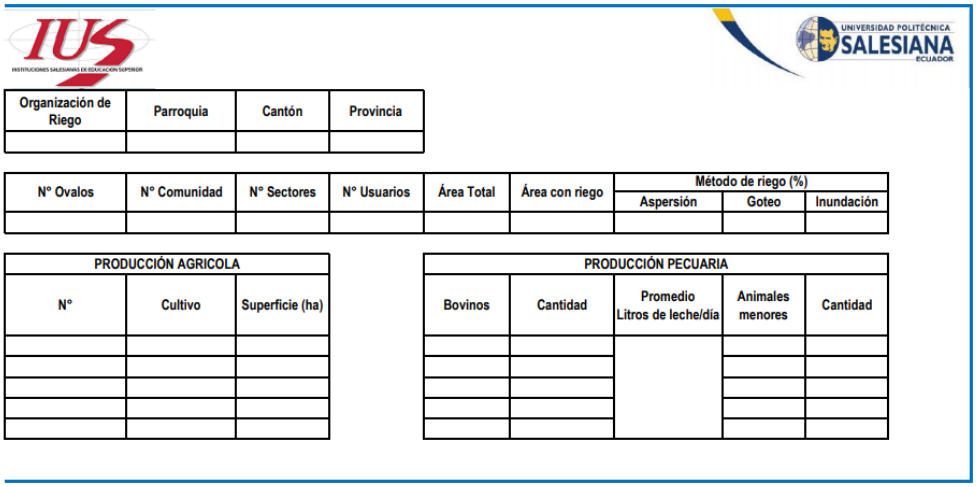
Figura 17 Segundo Reporte de Juntas

Elaborado por: Norman Chuquirima, Miguel Silva. (2018)

- **Reporte Público**

Este reporte será como su nombre lo indica público y servirá para obtener ciertos datos que se utilizarán a conveniencia de las instituciones o personas que hagan uso de este reporte.

Plantilla de Reporte Público



Organización de Riego	Parroquia	Cantón	Provincia

N° Ovalos	N° Comunidad	N° Sectores	N° Usuarios	Área Total	Área con riego	Método de riego (%)		
						Aspersión	Goteo	Inundación

PRODUCCIÓN AGRICOLA		
N°	Cultivo	Superficie (ha)

PRODUCCIÓN PECUARIA				
Bovinos	Cantidad	Promedio Litros de leche/día	Animales menores	Cantidad

Figura 18 Reporte Público

Elaborado por: Norman Chuquirima, Miguel Silva. (2018)

2.5. Requerimientos no funcionales:

"Como su nombre lo indica, son aquellos requerimientos que no se refieren directamente a las funciones específicas que proporciona el sistema, sino a las propiedades emergentes de éste como la fiabilidad, el tiempo de respuesta y la capacidad de almacenamiento. De forma alternativa, definen las restricciones del sistema como la capacidad de los dispositivos de entrada/salida y las representaciones de datos que se utilizan en las interfaces del sistema".(Sommerville & Alfonso Galipienso, 2005).

En la siguiente tabla se muestran los requisitos no funcionales del sistema:

Tabla 11 Requerimientos no funcionales

REQUERIMIENTOS NO FUNCIONALES	
1	El sistema web va a ser funcional 24/7
2	El sistema debe contar con colores claros
3	La interfaz debe ser clara y entendible
4	Se va a utilizar JAVA con lenguaje de programación, JSF con framework y HTML como lenguaje de modelado
5	Se va a visualizar correctamente en cualquier navegador
6	La Interfaz gráfica va a ser de fácil lectura

Nota: Esta tabla contiene los requerimientos no funcionales

3. CAPÍTULO III

3.1. Modelo Vista Controlador (MVC)

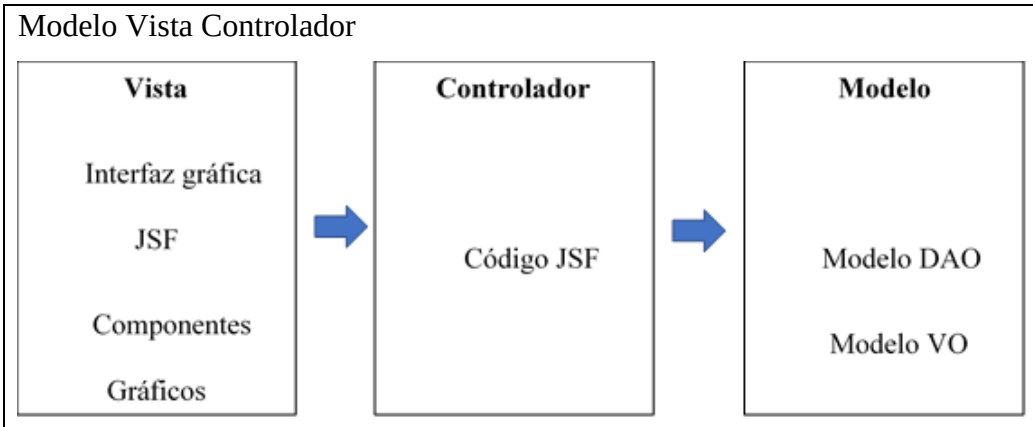


Figura 19 Modelo Vista Controlador

Elaborado por: Norman Chuquirima, Miguel Silva. (2018)

- La vista contiene los componentes gráficos con los que el usuario va a interactuar.
- EL controlador contiene todos los métodos que otorgan la funcionalidad al sistema, también hace de pasarle entre la vista y el modelo.
- El modelo permite la interacción con la BBDD utilizando las operaciones CRUD, además contiene los métodos getter y setter para la manipulación de datos.

3.2. Diagrama de Clases

Describe el conjunto (o subconjunto) de clases e interfaces de un sistema así como las asociaciones que los unen.(Debrauwer & Van der Heyde, 2016)

El diagrama de clases muestra todos los objetos creados a nivel de arquitectura MVC, las clases de la capa Vista contienen todos los componentes de la GUI con los que el usuario final va a interactuar, mientras que las clases del modelo controlador contiene todos los métodos con la lógica del sistema, es decir todos los cálculos y métodos que permiten la interacción de la interfaz gráfica de usuario con el modelo,

validaciones en general solicitadas como requerimientos de usuario; Las clases del modelo contienen todos los métodos que permiten la conexión, obtención y tratamiento de los datos del gestor PostgreSQL, estas clases obtienen mediante una consulta SQL los datos deseados CRUD.

Diagrama de Clases

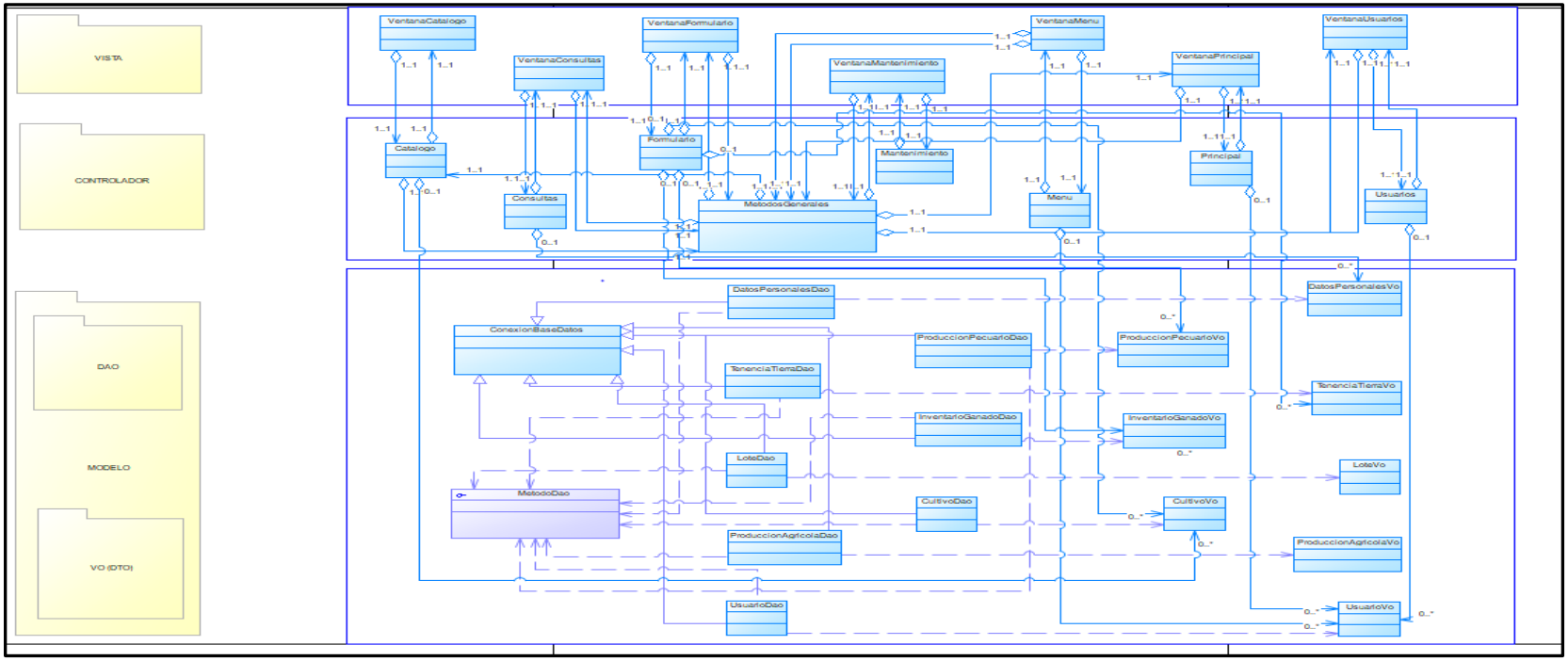


Figura 20 Diagrama de Clases

Elaborado por: Norman Chuquirima, Miguel Silva. (2018)

Diagrama de clases parcial de la capa controlador

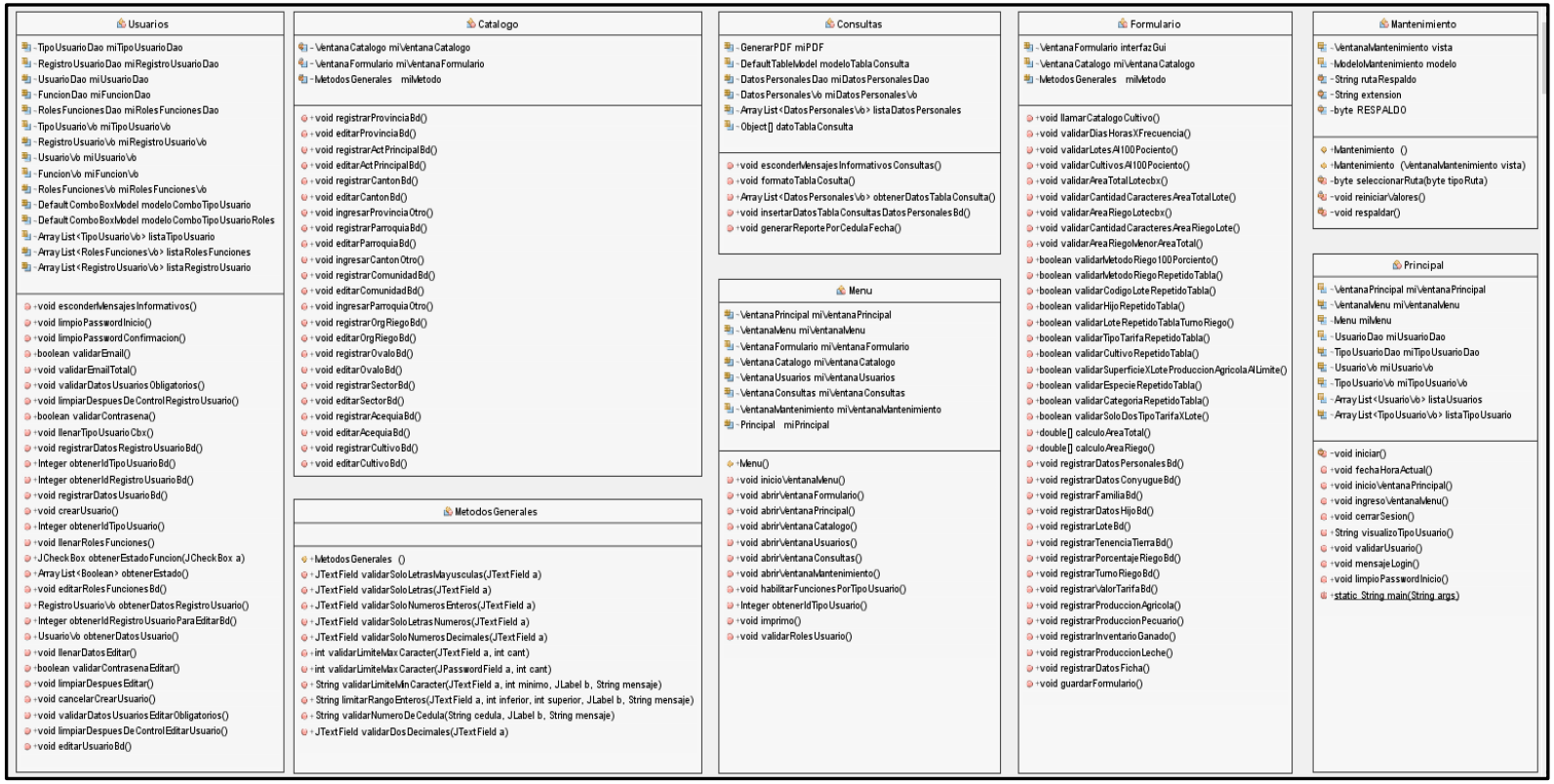


Figura 21 Diagrama de clases parcial de la capa controlador

Elaborado por: Norman Chuquirima, Miguel Silva. (2018)

Diagrama de clases parcial de la capa modelo DAO

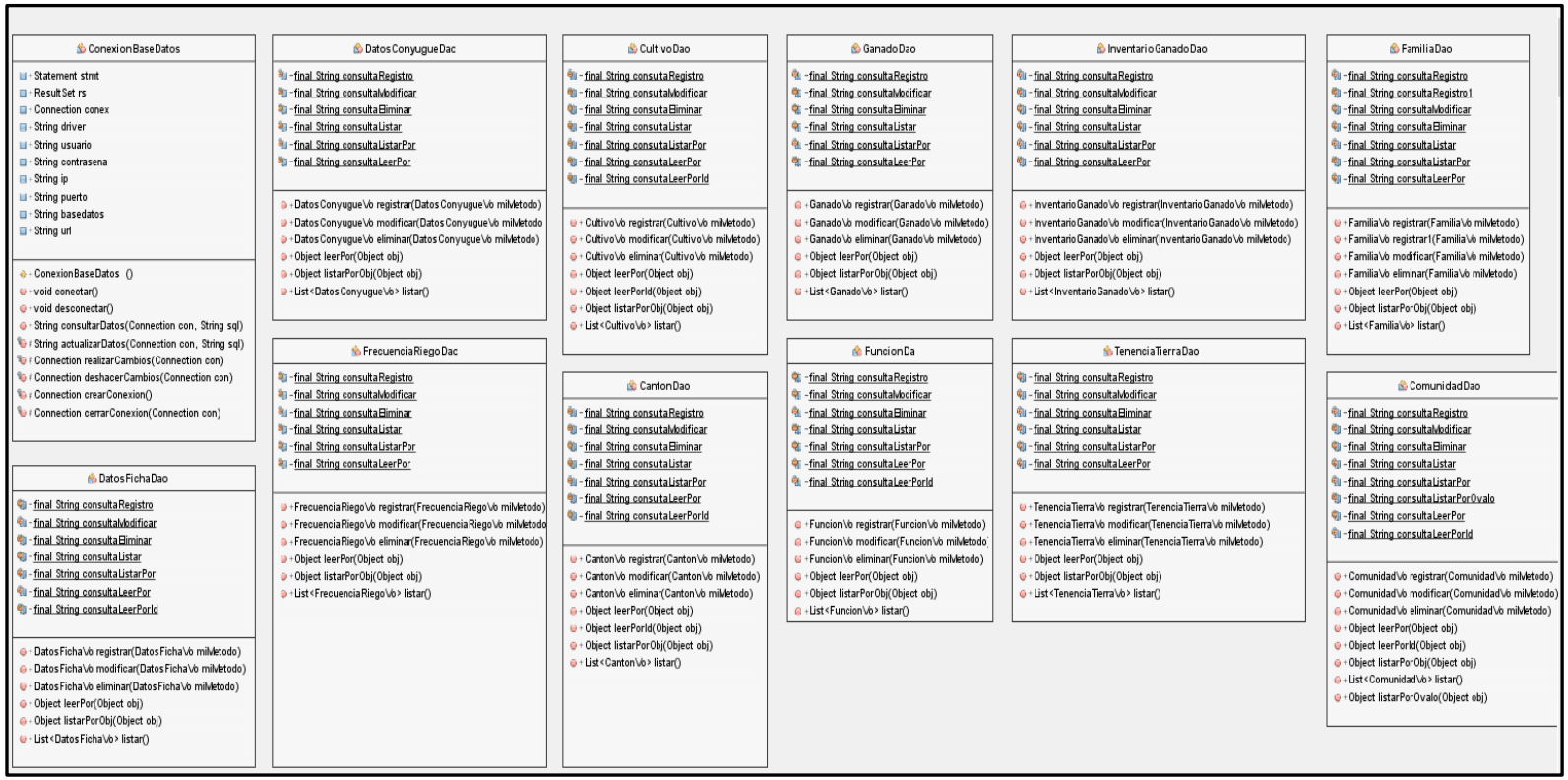


Figura 22 Diagrama de clases parcial de la capa modelo DAO

Elaborado por: Norman Chuquirima, Miguel Silva. (2018)

Diagrama de clases parcial de la capa modelo DAO parte 2

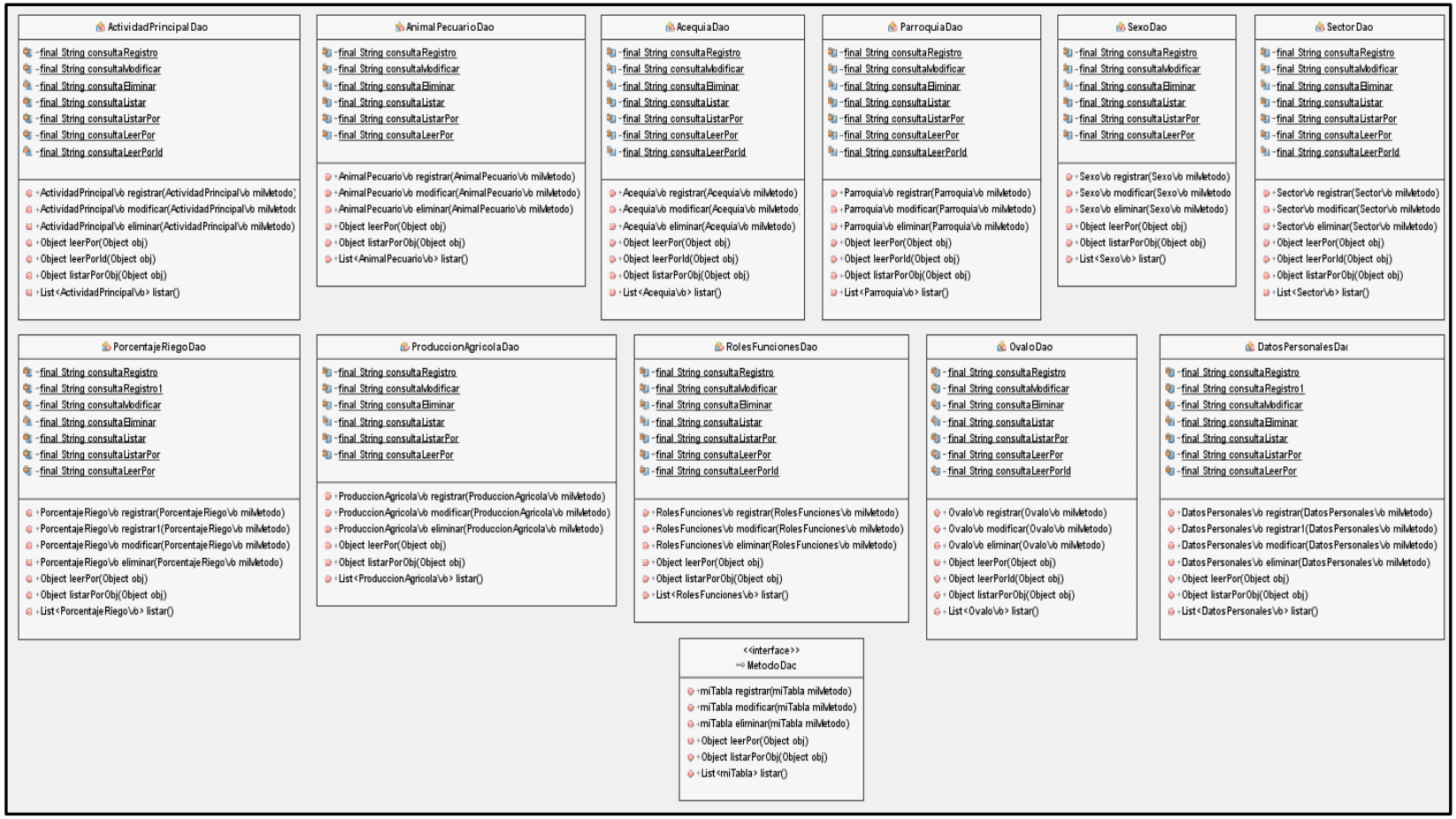


Figura 23 Diagrama de clases parcial de la capa modelo DAO parte 2

Elaborado por: Norman Chuquirima, Miguel Silva. (2018)

Diagrama de clases parcial de la capa modelo VO

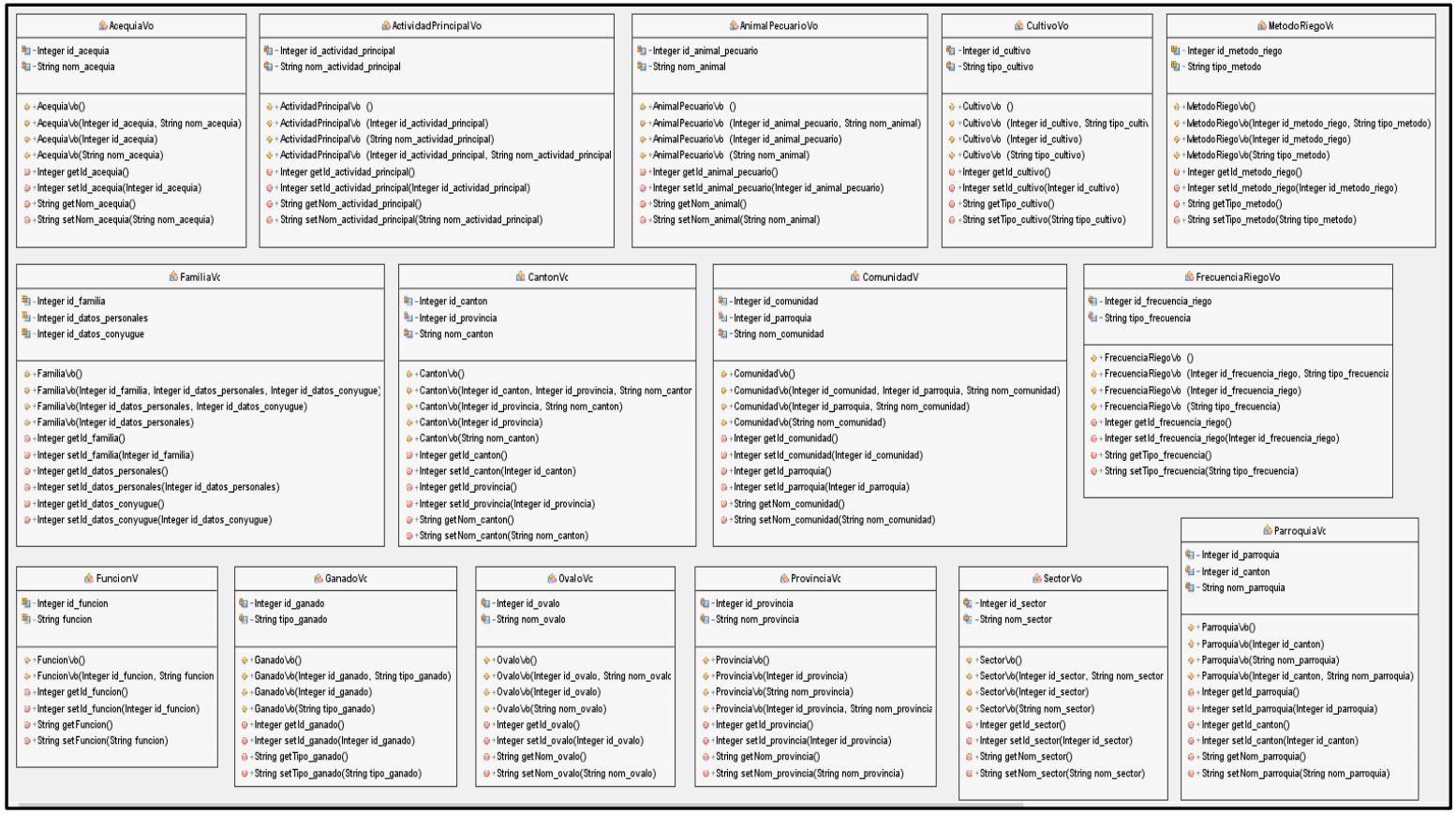


Figura 24 Diagrama de clases parcial de la capa modelo VO

Elaborado por: Norman Chuquirima, Miguel Silva. (2018)

3.3. Diagrama de Despliegue

Describe la arquitectura material del sistema.(Debrauwer & Van der Heyde, 2016)

Diagrama de despliegue

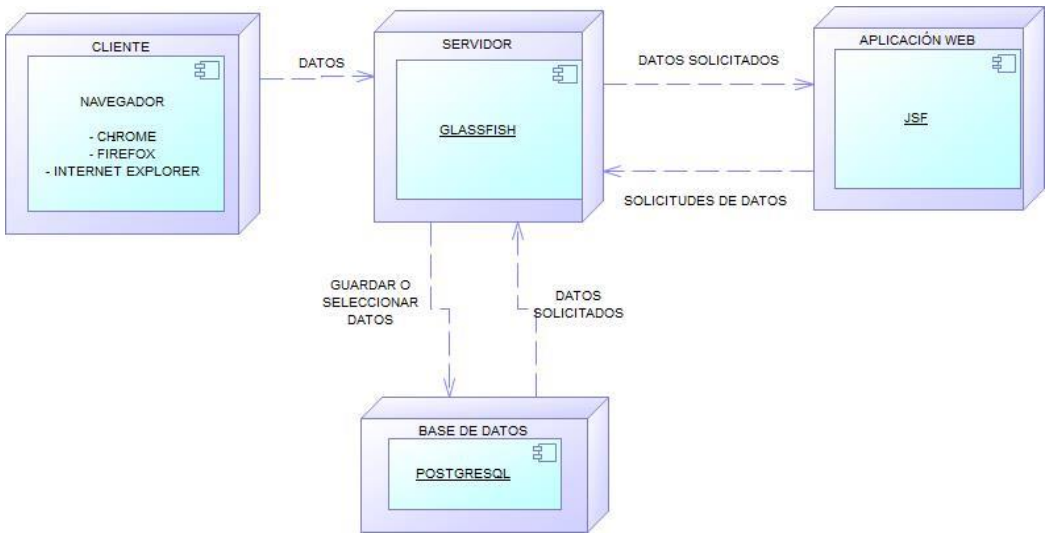


Figura 25 Diagrama de despliegue

Elaborado por: Norman Chuquirima, Miguel Silva. (2018)

- El cliente utiliza los diferentes navegadores para poder conectarse hacia la interfaz gráfica del sistema web.
- Una vez se conecten los usuarios con la ayuda de los navegadores se conectarán al servidor el cual hace los pedidos correspondientes a la BBDD y a los diferentes componentes de la aplicación web.

3.4. Modelo Conceptual de BBDD

Modelo Conceptual de la BBDD

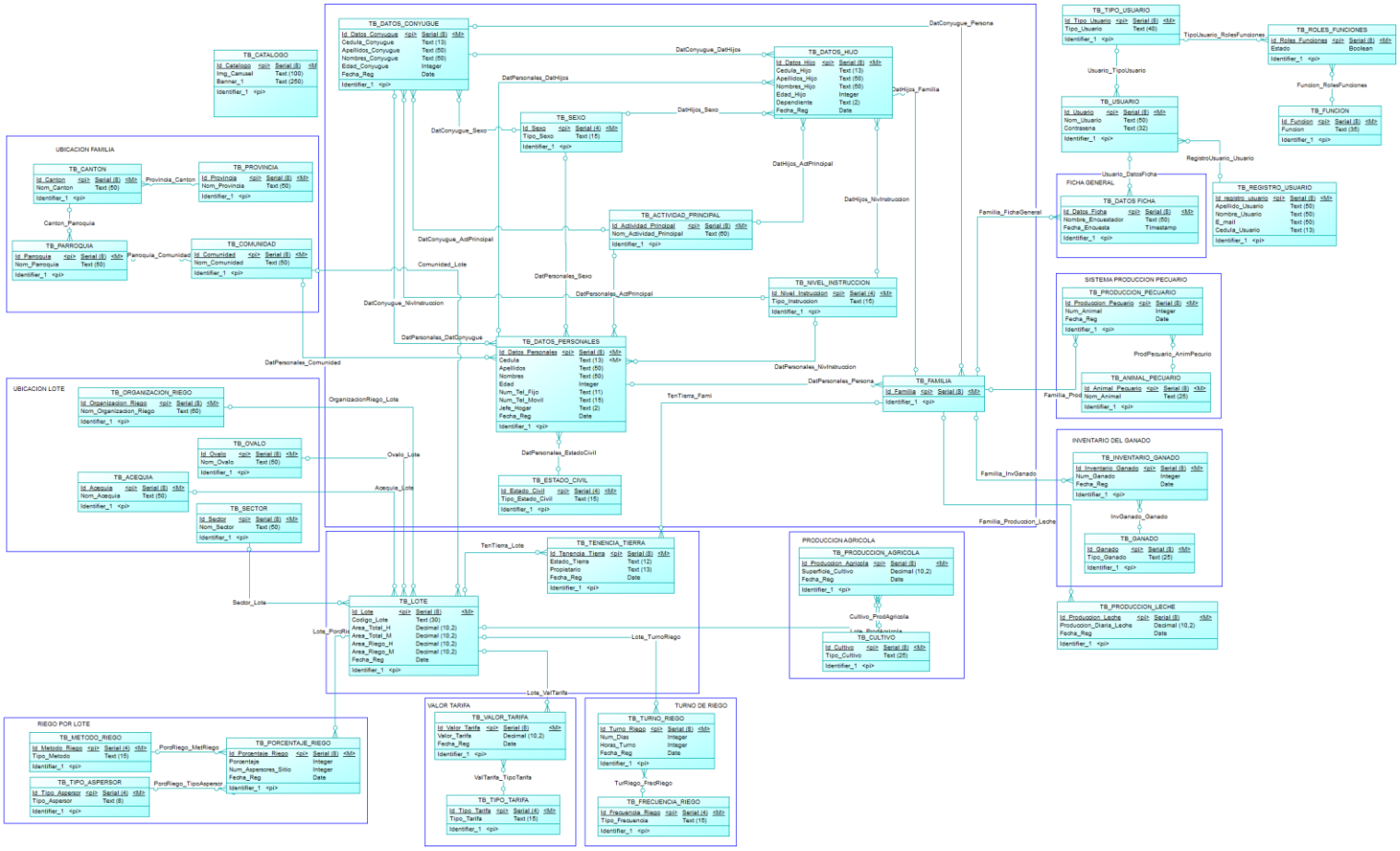


Figura 26 Modelo Conceptual de la BBDD

Elaborado por: Norman Chuquirima, Miguel Silva. (2018)

3.5. Diagrama de modelo físico de la BBDD

Modelo Físico de la Base de Datos

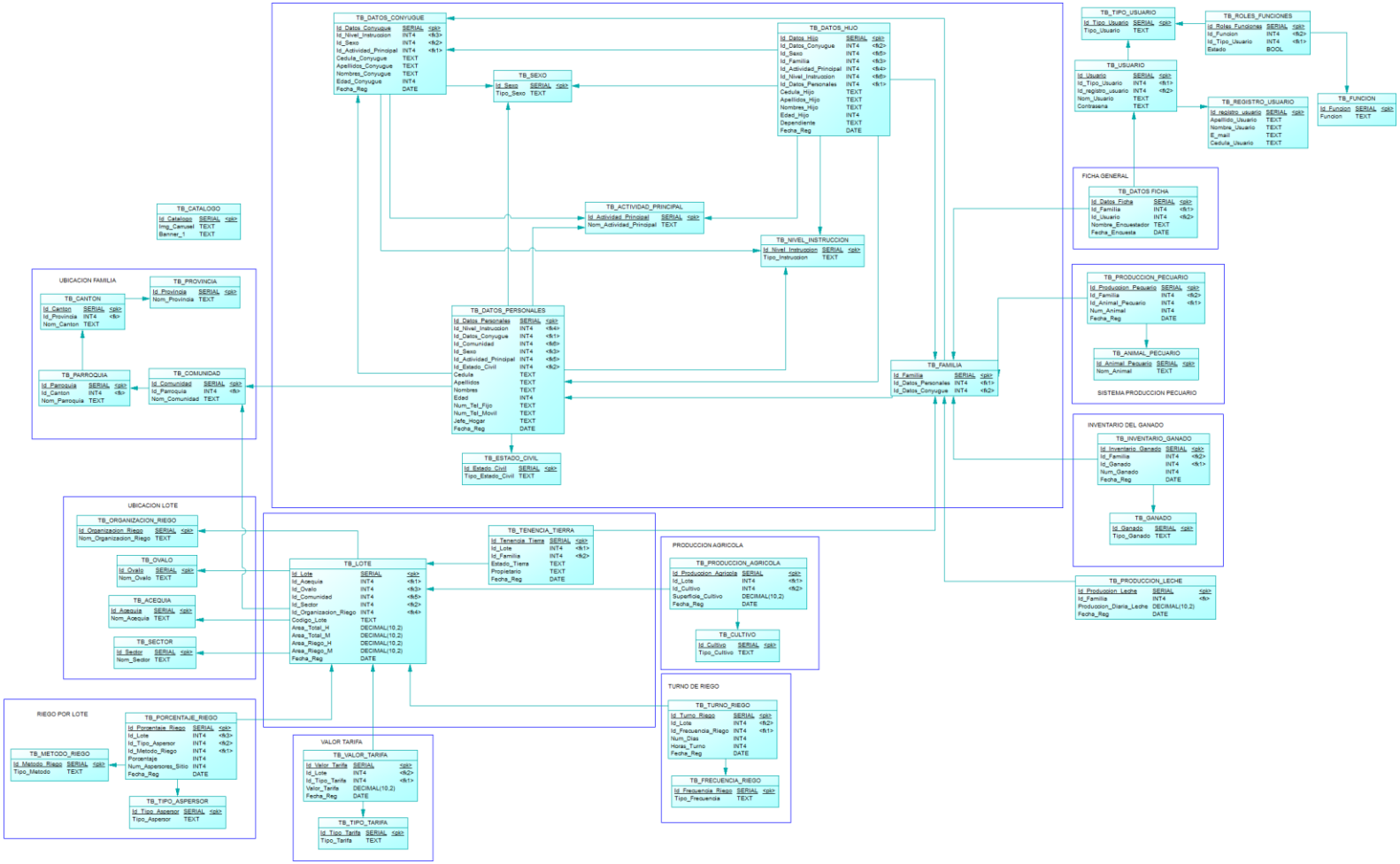


Figura 27 Modelo Físico de la Base de Datos

Elaborado por: Norman Chuquirima, Miguel Silva. (2018)

3.6. Diccionario de datos

El diccionario de datos presenta información de las diferentes entidades con sus respectivos campos, descripción, y tipo de dato. Cada entidad tiene un identificador autoincrementadle y relaciones mediante claves foráneas, además cada entidad está normalizada para evitar la redundancia de los datos para así mejorar la gestión, el diseño, y agilizar el tratamiento de los diferentes pedidos.

Figura 28 Contiene el diccionario de datos

TABLA	CAMPO	DESCRIPCIÓN	TIPO
Tb_acequia	id_acequia	Clave primaria, identificador único, autoincrementable.	serial (8)
	nom_acequia	Este campo almacena los nombres de los diferentes canales de riego.	text (50)
Tb_actividad_principal	id_actividad_principal	Clave primaria, identificador único, autoincrementable	serial (8)
	nom_actividad_principal	Nombres de actividades económicas a la que se dedica el usuario de riego, su conyugue y sus hijos.	text (60)
Tb_animal_pecuario	id_animal_pecuario	Clave primaria, identificador único, autoincrementable.	serial (8)
	nom_animal	Nombres de los animales pecuarios	text (25)
Tb_canton	id_canton	Clave primaria, identificador único, autoincrementable.	serial (8)

TABLA	CAMPO	DESCRIPCIÓN	TIPO
	id_provincia	Clave foránea de la tabla provincia.	integer
	nom_canton	Nombres de los cantones del Ecuador	text (50)
Tb_catalogo	id_catalogo	Clave primaria, identificador único, autoincrementable	serial (8)
	img_carrusel	Este campo contiene el path donde se encuentra ubicada cada imagen del carrusel	text (100)
	banner_1	Este campo contiene el path donde se encuentra ubicado cada imagen tipo banner	text (250)
Tb_comunidad	id_comunidad	Clave primaria, identificador único, autoincrementable	serial (8)
	id_parroquia	Clave foránea de la tabla parroquia.	integer
	nom_comunidad	Nombres de las comunidades de las diferentes parroquias del Ecuador	text (50)
Tb_cultivo	id_cultivo	Clave primaria, identificador único, autoincrementable	serial (8)
	tipo_cultivo	Este campo contiene los diferentes tipos de cultivo	text (25)
TB_conyugue	id_datos_conyugue	Clave primaria, identificador único, autoincrementable	serial (8)

TABLA	CAMPO	DESCRIPCIÓN	TIPO
	id_nivel_instruccion	Clave foránea de la tabla nivel_instruccion.	integer
	id_sexo	Clave foránea de la tabla sexo.	integer
	id_actividad_principal	Clave foránea de la tabla actividad_principal.	integer
	cedula_conyugue	Documento de identificación único del conyugue	text (13)
	apellidos_conyugue	Apellidos del conyugue del usuario de riego	text (50)
	nombres_conyugue	Nombres del conyugue del usuario de riego	text (50)
	edad_conyugue	Edad del conyugue del usuario de riego	integer
	fecha_reg	Almacena la fecha del sistema en que se ingresa un nuevo registro para auditoria e histórico	date
Tb_datos_ficha	id_datos_ficha	Clave primaria, identificador único, autoincrementable	serial (8)
	id_familia	Clave foránea de la tabla familia.	integer
	id_usuario	Clave foránea de la tabla usuario.	integer
	nombre_encuestador	Nombres y Apellidos de la persona encargada de realizar la encuesta a los diferentes usuarios de riego	text (50)

TABLA	CAMPO	DESCRIPCIÓN	TIPO
	fecha_encuesta	Fecha en la que fue realizado el registro de la información en el sistema padrón de usuario	date
Tb_datos_hijo	id_datos_hijo	Clave primaria, identificador único, autoincrementable	serial (8)
	id_datos_conyugue	Clave foránea de la tabla datos_conyugue.	integer
	id_sexo	Clave foránea de la tabla sexo.	integer
	id_familia	Clave foránea de la tabla familia.	integer
	id_actividad_principal	Clave foránea de la tabla actividad_principal.	integer
	id_nivel_instruccion	Clave foránea de la tabla nivel_instruccion.	integer
	id_datos_personales	Clave foránea de la tabla datos_personales.	integer
	cedula_hijo	Numero de cedula del hijo registrado por el usuario de riego	text (13)
	apellidos_hijo	Apellidos del Hijo registrado por el usuario de riego	text (50)
	nombres_hijo	Nombres del Hijo registrado por el usuario de riego	text (50)
	edad_hijo	Cantidad de años que tiene el Hijo registrado por el usuario de riego	integer

TABLA	CAMPO	DESCRIPCIÓN	TIPO
	dependiente	Si o No depende económicamente del usuario de riego	text (2)
	fecha_reg	Almacena la fecha del sistema en que se ingresa un nuevo registro para auditoria e histórico	date
Tb_datos_personales	id_datos_hijo	Clave primaria, identificador único, autoincrementable	serial (8)
	id_datos_conyugue	Clave foránea de la tabla datos_conyugue.	integer
	id_sexo	Clave foránea de la tabla sexo.	integer
	id_familia	Clave foránea de la tabla familia.	integer
	id_actividad_principal	Clave foránea de la tabla actividad_principal.	integer
	id_nivel_instruccion	Clave foránea de la tabla nivel_instruccion.	integer
	id_datos_personales	Clave foránea de la tabla datos_personales.	integer
	cedula_hijo	Numero de cedula del hijo registrado por el usuario de riego	text (13)
	apellidos_hijo	Apellidos del Hijo registrado por el usuario de riego	text (50)
	nombres_hijo	Nombres del Hijo registrado por el usuario de riego	text (50)
	edad_hijo	Cantidad de años que tiene el Hijo	integer

TABLA	CAMPO	DESCRIPCIÓN	TIPO
		registrado por el usuario de riego	
	dependiente	Si o No depende económicamente del usuario de riego	text (2)
	fecha_reg	Almacena la fecha del sistema en que se ingresa un nuevo registro para auditoria e histórico	date
Tb_estado_civil	id_estado_civil	Clave primaria, identificador único, autoincrementable	serial (8)
	tipo_estado_civil	Tipo de estado civil de una persona	text (15)
Tb_familia	id_familia	Clave primaria, identificador único, autoincrementable	serial (8)
	id_datos_personales	Clave foránea de la tabla datos_personales.	integer
	id_datos_conyugue	Clave foránea de la tabla datos_conyugue.	integer
Tb_frecuencia_riego	id_frecuencia_riego	Clave primaria, identificador único, autoincrementable	serial (4)
	tipo_frecuencia	Medida establecida para frecuencia de uso del sistema de riego	text (15)
Tb_funcion	id_funcion	Clave primaria, identificador único, autoincrementable	serial (8)
	funcion	Función del sistema Padrón de usuarios (Ejemplo: Llenar formulario).	text (35)

TABLA	CAMPO	DESCRIPCIÓN	TIPO
Tb_ganado	id_ganado	Clave primaria, identificador único, autoincrementable	serial (8)
	tipo_ganado	Animal de tipo ganado.	text (25)
Tb_inventario_ganado	id_inventario_ganado	Clave primaria, identificador único, autoincrementable	serial (8)
	id_familia	Clave foránea de la tabla familia.	integer
	id_ganado	Clave foránea de la tabla ganado.	integer
	num_ganado	Cantidad de animales tipo ganado que posee una familia	integer
	fecha_reg	Almacena la fecha del sistema en que se ingresa un nuevo registro para auditoria e histórico	date
Tb_lote	id_lote	Clave primaria, identificador único, autoincrementable	serial (8)
	id_acequia	Clave foránea de la tabla acequia.	integer
	id_ovalo	Clave foránea de la tabla ovalo.	integer
	id_comunidad	Clave foránea de la tabla comunidad.	integer
	id_sector	Clave foránea de la tabla sector.	integer
	id_organizacion_riego	Clave foránea de la tabla organización_riego.	integer
	codigo_lote	Identificador único de un lote mediante sistema de	text (30)

TABLA	CAMPO	DESCRIPCIÓN	TIPO
		información geográfica	
	area_total_h	Superficie total de un lote en hectáreas	numeric (10,2)
	area_total_m	Superficie total de un lote en metros cuadrados	numeric (10,2)
	area_riego_h	Superficie de riego de un lote en hectáreas	numeric (10,2)
	area_riego_m	Superficie de riego de un lote en metros cuadrados	numeric (10,2)
	fecha_reg	Almacena la fecha del sistema en que se ingresa un nuevo registro para auditoria e histórico	date
Tb_metoo_riego	id_metodo_riego	Clave primaria, identificador único, autoincrementable	serial (8)
	tipo_metodo	Tipo de Método de riego (Inundación, aspersión o goteo)	text (15)
Tb_nivel_instruccion	id_nivel_instruccion	Clave primaria, identificador único, autoincrementable	serial (4)
	tipo_instruccion	Nivel de instrucción educativa.	text (15)
Tb_organizacion_riego	id_organizacion_riego	Clave primaria, identificador único, autoincrementable	serial (8)
	nom_organizacion_riego	Nombre de una organización de riego a la que pertenece un lote	text (60)
Tb_ovalo	id_ovalo	Clave primaria, identificador único, autoincrementable	serial (8)

TABLA	CAMPO	DESCRIPCIÓN	TIPO
	nom_ovalo	Nombre de un Óvalo.	text (50)
Tb_parroquia	id_parroquia	Clave primaria, identificador único, autoincrementable	serial (8)
	id_canton	Clave foránea de la tabla canton.	integer
	nom_parroquia	Nombre de una parroquia.	text (50)
Tb_porcentaje_riego	id_porcentaje_riego	Clave primaria, identificador único, autoincrementable	serial (8)
	id_lote	Clave foránea de la tabla lote.	integer
	id_tipo_aspersor	Clave foránea de la tabla tipo_aspersor.	integer
	id_metodo_riego	Clave foránea de la tabla método_riego.	integer
	porcentaje	Porcentaje de cada método de riego que posee un lote	integer
	num_aspersores_sitio	La cantidad de aspersores que puede tener un lote en caso de que su método de riego se realice por aspersión.	integer
	fecha_reg	Almacena la fecha del sistema en que se ingresa un nuevo registro para auditoria e histórico	date
Tb_produccion_agricola	id_produccion_agricola	Clave primaria, identificador único, autoincrementable	serial (8)
	id_lote	Clave foránea de la tabla lote.	integer

TABLA	CAMPO	DESCRIPCIÓN	TIPO
	id_cultivo	Clave foránea de la tabla cultivo.	integer
	superficie_cultivo	Superficie que ocupa un tipo de cultivo dentro de un lote	numeric (10,2)
	fecha_reg	Almacena la fecha del sistema en que se ingresa un nuevo registro para auditoria e histórico	date
Tb_produccion_leche	id_produccion_leche	Clave primaria, identificador único, autoincrementable	serial (8)
	id_familia	Clave foránea de la tabla familia.	integer
	producción_diaria_leche	Cantidad de litros/hato de producción de leche diaria.	numeric (10,2)
	fecha_reg	Almacena la fecha del sistema en que se ingresa un nuevo registro para auditoria e histórico	date
Tb_produccion_pecuario	id_produccion_pecuario	Clave primaria, identificador único, autoincrementable	serial (8)
	id_familia	Clave foránea de la tabla familia.	integer
	id_animal_pecuario	Clave foránea de la tabla animal_pecuario.	integer
	num_animal	Cantidad de un tipo de animal pecuario que posee una familia.	integer
	fecha_reg	Almacena la fecha del sistema en que	date

TABLA	CAMPO	DESCRIPCIÓN	TIPO
		se ingresa un nuevo registro para auditoria e histórico	
Tb_provincia	id_provincia	Clave primaria, identificador único, autoincrementable	serial (8)
	nom_provincia	Nombre de una provincia del Ecuador	text (50)
Tb_registro_usuario	id_registro_usuario	Clave primaria, identificador único, autoincrementable	serial (8)
	apellido_usuario	Apellidos de un usuario de riego	text (50)
	nombre_usuario	Nombres de un usuario de riego	text (50)
	e_email	Dirección de correo electrónico de un usuario de riego	text (50)
	cedula_usuario	Numero de cedula de usuario de riego	text (13)
Tb_roles_funciones	id_roles_funciones	Clave primaria, identificador único, autoincrementable	serial (8)
	id_funcion	Clave foránea de la tabla función.	integer
	id_tipo_usuario	Clave foránea de la tabla tipo_usuario.	integer
	estado	Permite habilitar o deshabilitar cada función del sistema	boolean
Tb_sector	id_sector	Clave primaria, identificador único, autoincrementable	serial (8)
	nom_sector	Sectores para la ubicación de un lote	text (50)
Tb_sexo	id_sexo	Clave primaria, identificador único, autoincrementable	serial (4)

TABLA	CAMPO	DESCRIPCIÓN	TIPO
	tipo_sexo	Tipo de sexo (Masculino, Femenino, Otro)	text (15)
Tb_tenencia_tierra	id_tenencia_tierra	Clave primaria, identificador único, autoincrementable	serial (8)
	id_lote	Clave foránea de la tabla lote.	integer
	id_familia	Clave foránea de la tabla familia.	integer
	estado_tierra	Puede ser propio o en arriendo	text (12)
	propietario	Número de cédula del dueño legal del lote	text (13)
	fecha_reg	Almacena la fecha del sistema en que se ingresa un nuevo registro para auditoria e histórico	date
Tb_tipo_aspersor	id_tipo_aspersor	Clave primaria, identificador único, autoincrementable	serial (4)
	tipo_aspersor	Tipo de aspersor que puede ocupar un lote	text (8)
Tb_tipo_tarifa	id_tipo_tarifa	Clave primaria, identificador único, autoincrementable	serial (4)
	tipo_tarifa	Tipo de tarifa para el uso de agua de riego	text (15)
Tb_tipo_usuario	id_tipo_usuario	Clave primaria, identificador único, autoincrementable	serial (8)
	tipo_usuario	Perfil definido de usuario con diferentes roles y características.	text (40)

TABLA	CAMPO	DESCRIPCIÓN	TIPO
Tb_turno_riego	id_turno_riego	Clave primaria, identificador único, autoincrementable	serial (8)
	id_lote	Clave foránea de la tabla lote.	integer
	id_frecuencia_riego	Clave foránea de la tabla frecuencia_riego.	integer
	num_dias	Número de días que un usuario hace uso del riego por cada lote.	integer
	horas_turno	Número de Horas en función de num_dias que un usuario hace uso del riego por cada lote	integer
	fecha_reg	Almacena la fecha del sistema en que se ingresa un nuevo registro para auditoria e histórico	date
Tb_usuario	id_usuario	Clave primaria, identificador único, autoincrementable	serial (8)
	id_tipo_usuario	Clave foránea de la tabla tipo_usuario.	integer
	id_registro_usuario	Clave foránea de la tabla registro_usuario.	integer
	nom_usuario	Nombre de identificación corto para validar el ingreso al sistema	text (50)
	contrasena	Código secreto de caracteres alfanuméricos para validar el ingreso al sistema	text (32)

TABLA	CAMPO	DESCRIPCIÓN	TIPO
Tb_valor_tarifa	id_valor_tarifa	Clave primaria, identificador único, autoincrementable	serial (8)
	id_lote	Clave foránea de la tabla lote.	integer
	id_tipo_tarifa	Clave foránea de la tabla tipo_tarifa.	integer
	valor_tarifa	Valor en dólares a cancelar por el tipo de tarifa de riego	numeric (10,2)
	fecha_reg	Almacena la fecha del sistema en que se ingresa un nuevo registro para auditoria e histórico	date

Nota: Esta tabla contiene todos los datos de las tablas en la BBDD.

3.7. Métodos más importantes

3.7.1. Interface MetodoDao

La Interface MetodoDao clase se compara a una plantilla, es decir una lista de acciones que obligatoriamente deben ser implementadas. Esta clase contiene Métodos de tipo CRUD que se implementan en las clases DAO ubicadas en el modelo del sistema.

Interface MetodoDAO

```
package Modelo.Interfaces;

import java.util.List;

public interface MetodoDao <miTabla> {

    public void registrar(miTabla miMetodo) throws Exception;
    public void modificar(miTabla miMetodo) throws Exception;
    public void eliminar(miTabla miMetodo) throws Exception;
    public miTabla leerPor(Object obj) throws Exception;
    public List<miTabla> listarPorObj(Object obj)throws Exception;
    public List<miTabla> listar() throws Exception;

}
```

Figura 29 Interface MetodoDAO

Elaborado por: Norman Chuquirima, Miguel Silva. (2018)

3.7.2. Clase UsuarioDao

La clase UsuarioDao hereda de la clase conexión la cual está permitida a usar los atributos de dicha clase, además implementa la interface MetodoDao, es decir que obligatoriamente sobrescribe los métodos CRUD.

Clase usuario DAO

```
public class UsuarioDao extends ConexionBaseDatos implements MetodoDao<UsuarioVo> {

    private static final String consultaRegistro ="INSERT INTO tb_usuario(id_tipo_usuario, id_registro_usuario, nom_usuario, contrasena) VALUES(?,?,?,?)";
    private static final String consultaModificar ="UPDATE tb_usuario SET nom_usuario = ?, contrasena = ?  WHERE id_registro_usuario = ?";
    private static final String consultaEliminar ="DELETE FROM tb_usuario WHERE nom_provincia = ?";
    private static final String consultaListar ="SELECT * FROM tb_usuario";
    private static final String consultaListarPor ="SELECT * FROM tb_usuario WHERE nom_provincia = ?";
    private static final String consultaLeerPor ="SELECT * FROM tb_usuario WHERE nom_usuario = ? ";
    private static final String consultaLeerPorId ="SELECT * FROM tb_usuario WHERE id_registro_usuario = ? ";
    private static final String consultaLeerPor2 ="SELECT * FROM tb_usuario WHERE nom_usuario = ? and contrasena = ?";
```

Figura 30 Clase usuario DAO

Elaborado por: Norman Chuquirima, Miguel Silva. (2018)

3.7.3. Método registrar ()

El método registrar permite guardar información en la base de datos mediante una consulta SQL, este método obtiene los datos mediante la Clase UsuarioVo que contiene los métodos getter y setter; Este método permite registrar un nuevo usuario en el sistema al cual se le otorga roles y permisos para la interacción con el sistema.

Método Registrar

```
@Override
public void registrar(UsuarioVo miMetodo) throws Exception {
    try {
        this.conectar();
        PreparedStatement st = this.conex.prepareStatement(consultaRegistro);
        st.setInt(1, miMetodo.getId_tipo_usuario());
        st.setInt(2, miMetodo.getId_registro_usuario());
        st.setString(3, miMetodo.getNom_usuario());
        st.setString(4, miMetodo.getContrasena());
        st.executeUpdate();

    } catch (Exception e) {
        throw e;
    }finally{
        this.desconectar();
    }
}
```

Figura 31 Método Registrar

Elaborado por: Norman Chuquirima, Miguel Silva. (2018)

3.7.4. Clase UsuarioVo

La Clase UsuarioVo (Value Object) contiene constructores y métodos getter y setter que permiten la interacción entre la capa MVC, el Sistema Padrón de usuario posee una clase del tipo Value Object por cada tabla del modelo de la base de datos. Donde cada variable es un campo, y el tipo de dato es equivalente al tipo de dato generado por la base de datos.

Clase Usuario VO

```
public class UsuarioVo {
    Integer id_usuario;
    Integer id_tipo_usuario;
    Integer id_registro_usuario;
    String nom_usuario;
    String contrasena;

    public UsuarioVo() {
    }

    public UsuarioVo(Integer id_usuario, Integer id_tipo_usuario, Integer id_registro_usuario, String nom_usuario, String contrasena) {
        this.id_usuario = id_usuario;
        this.id_tipo_usuario = id_tipo_usuario;
        this.id_registro_usuario = id_registro_usuario;
        this.nom_usuario = nom_usuario;
        this.contrasena = contrasena;
    }
```

Figura 32 Clase Usuario VO

Elaborado por: Norman Chuquirima, Miguel Silva. (2018)

3.7.5. Método validarDosDecimales()

Este método pertenece a la capa Controlador, y permite validar el número de decimales que ingresa el usuario en el objeto visual de la Capa Vista o Interfaz gráfica de usuario. Recibe como parámetro un objeto del tipo jTextField, y ocupa un evento del tipo KeyTyped.

Métodos para validar dos decimales

```
public void validarDosDecimales (JTextField a){
    a.addKeyListener(new KeyAdapter() {
        @Override
            public void keyTyped(KeyEvent e){
                if(a.getText().split("\\.").length == 2 && a.getText().split("\\.")[1].length() >= 2 ){
                    e.consume();
                }
            }
    });
}
```

Figura 33 Métodos para validar dos decimales

Elaborado por: Norman Chuquirima, Miguel Silva. (2018)

3.7.6. Método limitarRangoEnteros()

Este método pertenece a la capa Controlador, y permite validar el ingreso de un numero entero, con un límite inferior y superior, este método se encarga que el usuario ingrese exactamente el número de teléfono móvil, y el número de teléfono local. Evitando que los datos de contacto del usuario sean erróneos. El método limitarRangoDecimales recibe como parámetro un objeto del tipo JTextField, el número inferior de caracteres permitido, el número superior de caracteres permitido, la ubicación del mensaje de advertencia si esto no se cumple, y el mensaje que se desplegara en la interfaz gráfica de usuario.

Método Limitar Rango Enteros

```
public void limitarRangoEnteros(JTextField a, int inferior, int superior, JLabel b, String mensaje){
        if(!a.getText().equals("")){
            int numerorango;
             numerorango = Integer.parseInt(a.getText());
                if (numerorango < inferior || numerorango > superior) {
                    b.setText(mensaje);
                    b.setVisible(true);
                }else{
                    b.setVisible(false);
                }
        }else{
            if(b.getText().equals(mensaje)){
              b.setText("");
            }
        }
}
```

Figura 34 Método Limitar Rango Enteros

Elaborado por: Norman Chuquirima, Miguel Silva. (2018)

3.8. Método validarDiasHorasXFrecuencia ()

Este método pertenece a la capa Controlador, permite registrar datos acerca de la frecuencia de riego, es decir el tiempo en días y horas que utiliza el servicio de riego para cada lote que posee.

Método validar días horas por frecuencias

```
public void validarDiasHorasXFrecuencia(){
    try {
        if(!VentanaFormulario.frecuenciacbx.getSelectedItem().equals("PERMANENTE")){
        int horasXDia;
        int numDias = Integer.parseInt(VentanaFormulario.numDiastxt.getText());
        horasXDia = 24 * numDias;

        if(VentanaFormulario.frecuenciacbx.getSelectedItem().equals("SEMANAL")){
            limitarRangoEnteros(VentanaFormulario.numDiastxt, 1, 7, VentanaFormulario.numDiasTurnoinf,
            "* El Número de Días debe estar en un rango entre 1 y 7");
            limitarRangoEnteros(VentanaFormulario.horasTurnotxt, 1,horasXDia, VentanaFormulario.horasTurnoinf,
            "* El Número de Horas debe estar en un rango entre "+ 1 +" y "+ horasXDia);
        }

        if(VentanaFormulario.frecuenciacbx.getSelectedItem().equals("QUINCENAL")){
            limitarRangoEnteros(VentanaFormulario.numDiastxt, 1, 15, VentanaFormulario.numDiasTurnoinf,
            "* El Número de Días debe estar en un rango entre 1 y 15");
            limitarRangoEnteros(VentanaFormulario.horasTurnotxt, 1,horasXDia, VentanaFormulario.horasTurnoinf,
            "* El Número de Horas debe estar en un rango entre "+ 1 +" y "+ horasXDia);
        }

        if(VentanaFormulario.frecuenciacbx.getSelectedItem().equals("MENSUAL")){
            limitarRangoEnteros(VentanaFormulario.numDiastxt, 1, 30, VentanaFormulario.numDiasTurnoinf,
            "* El Número de Días debe estar en un rango entre 1 y 30");
            limitarRangoEnteros(VentanaFormulario.horasTurnotxt, 1,horasXDia, VentanaFormulario.horasTurnoinf,
            "* El Número de Horas debe estar en un rango entre "+ 1 +" y "+ horasXDia);
        }
    }
    } catch (NumberFormatException e) {
    }
}
```

Figura 35 Método validar días horas por frecuencias

Elaborado por: Norman Chuquirima, Miguel Silva. (2018)

3.8.1. Método validarLotesAl100Porciento ()

Este método pertenece a la capa controlador, se encarga de validar que cada lote que cuente con un Área de riego registre al menos un método de riego y que la suma de porcentajes de estos métodos no sea menor, o sobrepase el porcentaje 100%

Los métodos de riego son (Inundación, Aspersión o goteo), se puede seleccionar uno, dos o tres métodos por lote, pero la suma de estos métodos debe ser siempre 100%.

Método validar lotes al 100 porciento

```
public void validarLotesAl100Pociento(){
    ArrayList<String> ComboLoteMetodoRiego = new ArrayList<>();
        for(int i = 0 ; i < modeloComboLoteMetodoRiego.getSize()-1;i++){
            ComboLoteMetodoRiego.add(i, (String) VentanaFormulario.loteMetodoRiegocbx.getItemAt(i+1));
        }
        for(int i = 0 ; i < ComboLoteMetodoRiego.size();i++){
            for(int j = 0 ; j < VentanaFormulario.tablaMetodoRiego.getRowCount();j++){
                if(ComboLoteMetodoRiego.get(i).equals(VentanaFormulario.tablaMetodoRiego.getValueAt(j, 3))){
                    if(VentanaFormulario.tablaMetodoRiego.getValueAt(j, 8).equals("100")){
                        ComboLoteMetodoRiego.remove(i);
                    }
                }
            }
        }
        if(!ComboLoteMetodoRiego.isEmpty()){
            VentanaFormulario.tablaMetodoRiegoinf.setText("El Lote: " + ComboLoteMetodoRiego + " no registra el 100%");
            VentanaFormulario.tablaMetodoRiegoinf.setVisible(true);
        }else{
            VentanaFormulario.tablaMetodoRiegoinf.setVisible(false);
        }
}
```

Figura 36 Método validar lotes al 100 porciento

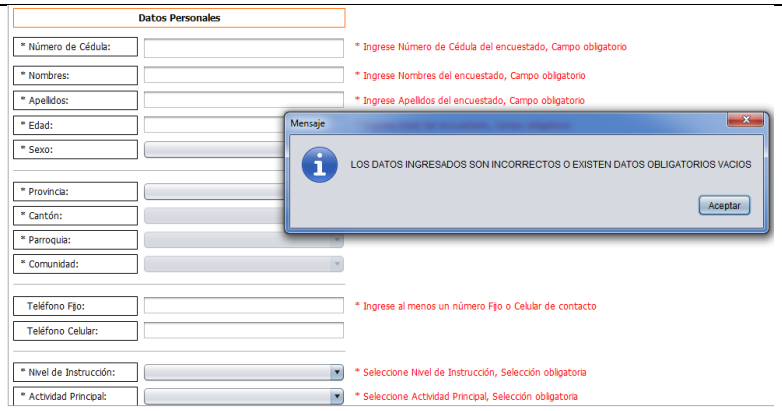
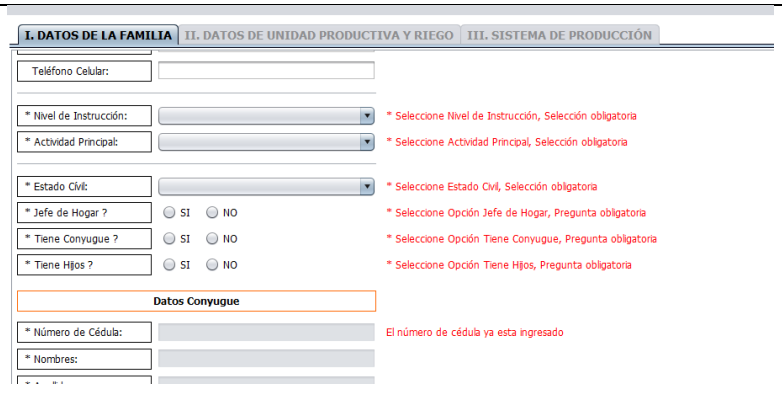
Elaborado por: Norman Chuquirima, Miguel Silva. (2018)

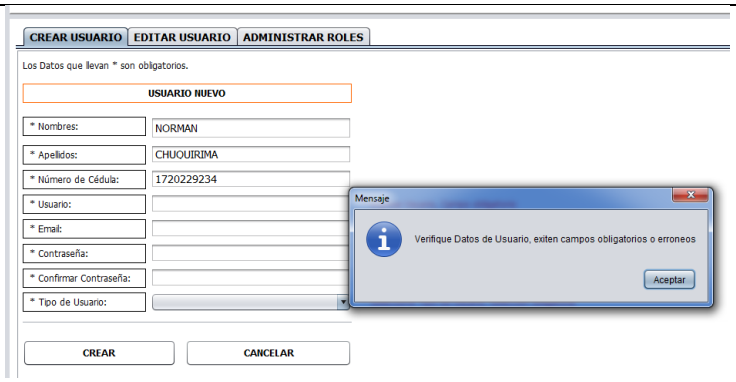
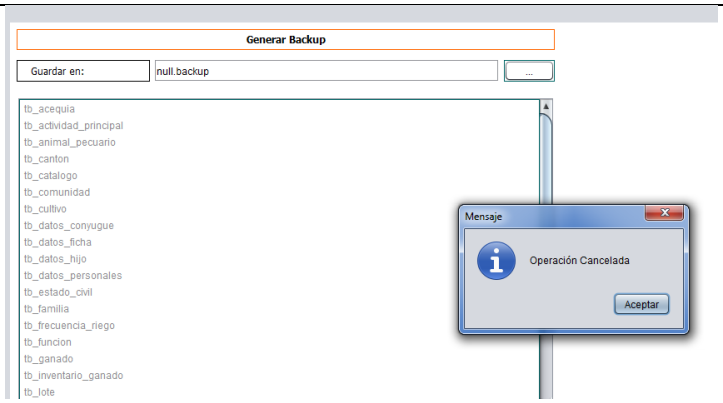
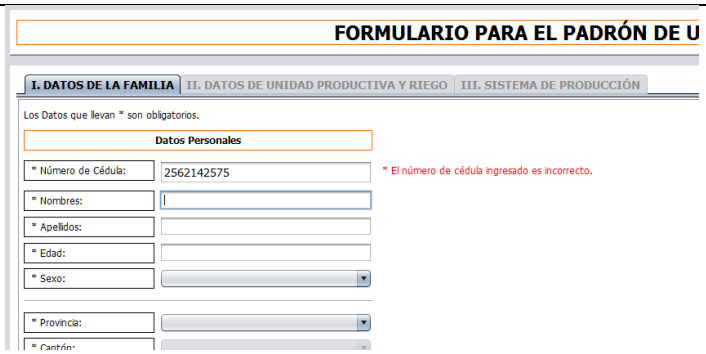
3.9. Pruebas y Resultados

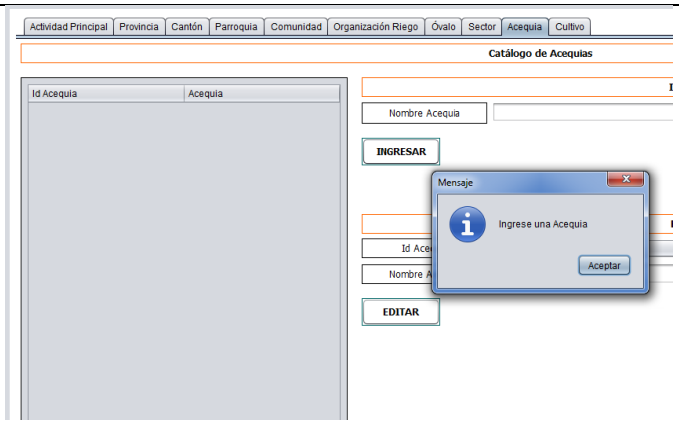
3.9.1. Pruebas de Caja Negra

Las pruebas de caja negra son aquellas que se enfocan directamente en el exterior del módulo, sin importar el código, son pruebas funcionales en las que se trata de encontrar fallas en las que este no se atiende a su especificación, como ser interfaz con el usuario, apariencia de los menús, control de las teclas, entre otras.(Natsys, 2017)

Tabla 12 Pruebas de caja negra

ACTIVIDAD	ACCIÓN	RESULTADO
Inicio de sesión	El usuario ingresa mal sus datos	
Llenar formulario	El usuario no ingresó los datos	
Llenar formulario	El usuario no llenó bien el formulario o dejó espacios en blanco	

Registro de Usuarios	Los datos ingresados no son correctos	
Reporte de Datos	El sistema envía una alerta de falta de datos	
Cancelar BackUp	Se cancela la generación del BackUp	
Ingreso de Formulario	Se ingresa datos incorrectos y el sistema envía un	

	mensaje de error	
Administrar Catálogo	Mensaje de error al no ingresar los datos	

Nota: Esta tabla muestra las pruebas de caja negra

3.9.2. Pruebas de Carga

- **Requisitos Principales para la prueba**
  - Puerto 8080 habilitado
  - Dirección o Path de la página web
  - Servidor Glasfish levantado
  - Apache jMeter configurado
  - Java versión 6 o superior
  - Características del servidor.
  - Sistema Operativo: Windows 10 pro de 64 bits
  - Procesador: Intel Core i7-4700MQ de 2.40 GHz
  - Memoria RAM: 16 GB
  - Disco Duro: 1TB

- **Descripción de la prueba.**

Simulación o prueba de carga con 500 usuarios concurrentes para el Sistema Web Padrón de Usuarios, con el fin de evaluar el performance, detectar procesos erróneos o detectar bugs que están ocupando de forma incorrecta los recursos del procesador.

- **Pruebas de carga en JMeter**

En la Figura 36 se configura varios parámetros para iniciar la prueba de carga, la cual contiene el grupo de usuarios concurrentes, los módulos del sistema web para la simulación, además de los tipos de resultados.

Plan de pruebas

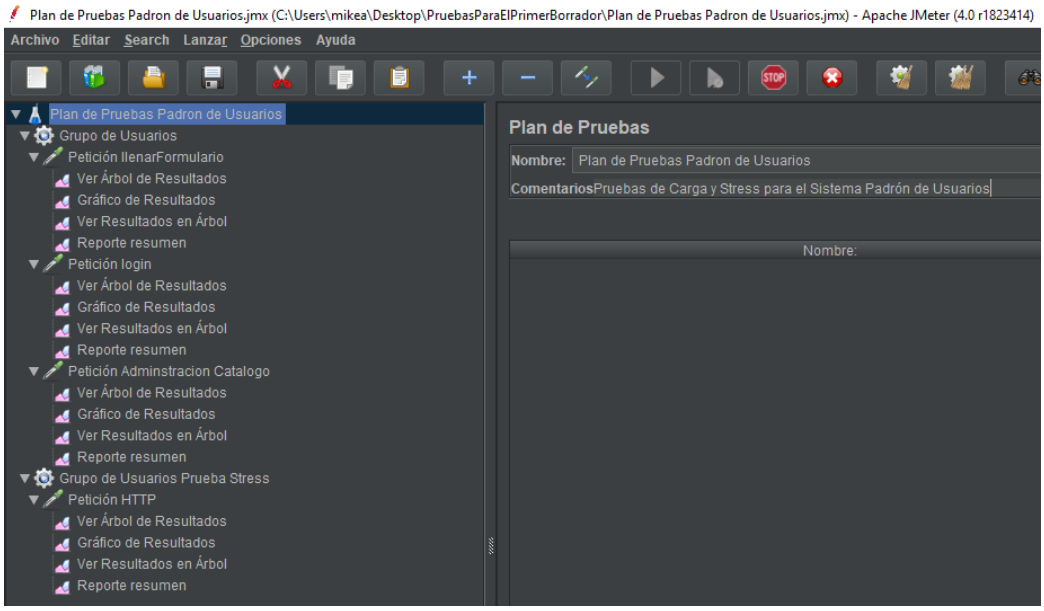


Figura 37 Plan de pruebas

Elaborado por: Norman Chuquirima, Miguel Silva. (2018)

En la Figura 37 presenta las diferentes configuraciones de la cantidad de usuarios concurrentes a simular para probar la carga esperada del sistema, en este caso la muestra es de 500 usuarios, los cuales hacen un request cada uno en un periodo de subida de 1 segundo, sin embargo de recomienda realizar las pruebas en periodos largos de tiempo.

Configuración de la prueba con 500 usuarios

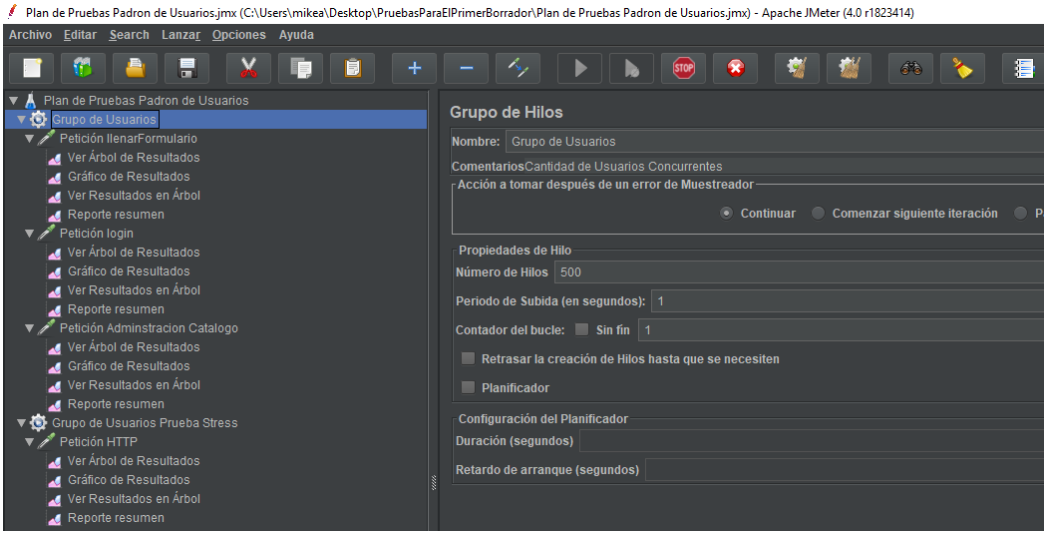


Figura 38 Configuración de prueba con 500 usuarios

Elaborado por: Norman Chuquirima, Miguel Silva. (2018)

En la Figura 38 muestra la configuración del tipo de request a utilizar en la prueba, en este caso se utiliza el HTTP request para los módulos del sistema web, muestra la configuración del nombre del servidor o la ip, el número de puerto y el método a utilizar, en este caso es el método GET. Se define el Path para la prueba (llenarFormulario).

URL del sistema

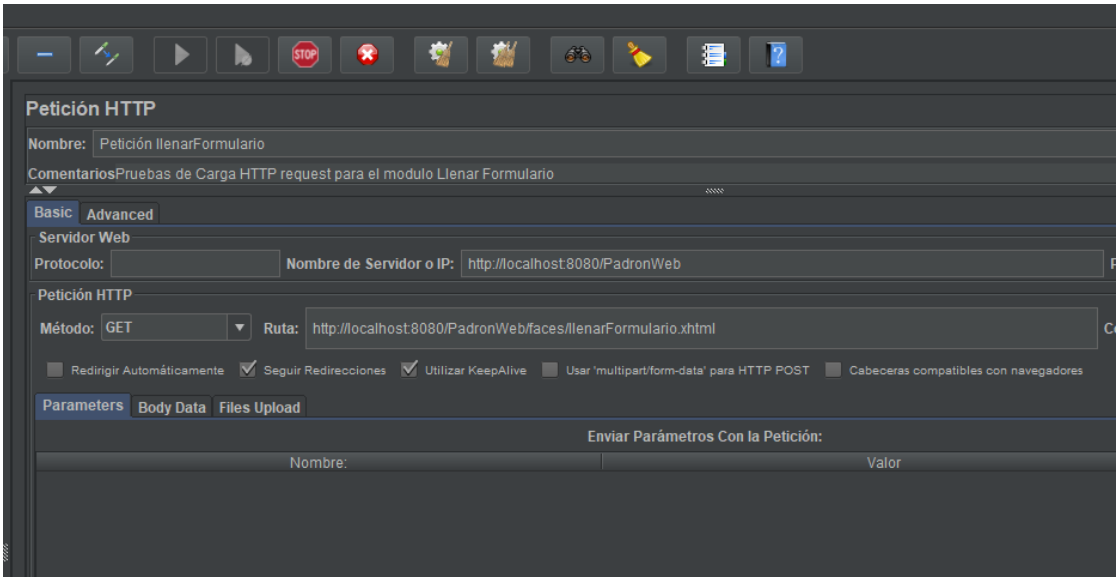


Figura 39 Insertar la URL del sistema Web

Elaborado por: Norman Chuquirima, Miguel Silva. (2018)

- **Reportes de Rendimiento y Estrés**
    - Árbol de resultados
    - Gráfico de resultados
    - Resumen de reporte

La Figura 39 despliega información de la prueba de carga por cada usuario simulado, el tiempo de carga, tiempo de la conexión, código de respuesta = 200 que indica que el método GET de HTTP request se realizó con éxito.

Árbol de resultados

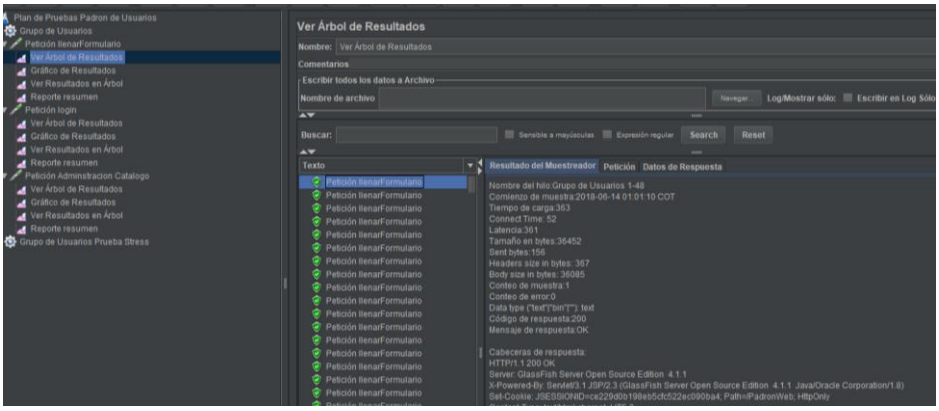


Figura 40 Árbol de resultados

Elaborado por: Norman Chuquirima, Miguel Silva. (2018)

La Figura 40 muestra que con 500 usuarios el sistema web se comporta de la manera esperada ya que el rendimiento muestra un pico al iniciar la prueba porque existen demasiados request y luego baja ya que el sistema comienza a resolverlos.

Gráfico de resultados

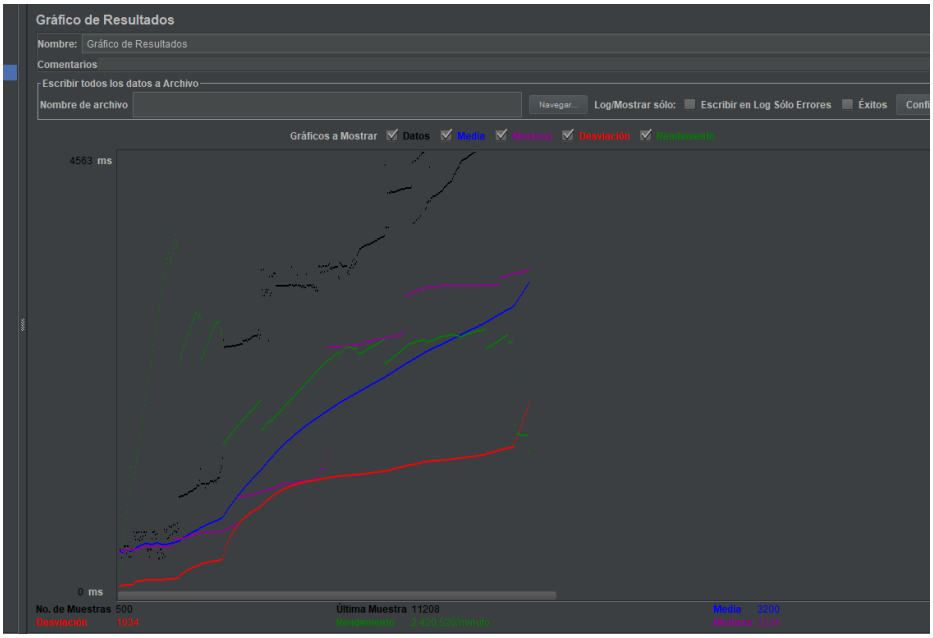


Figura 41 Gráfico de resultados

Elaborado por: Norman Chuquirima, Miguel Silva. (2018)

La Figura 41 presenta el número de muestras que se han tomado, el tiempo promedio que ha tomado realizar la prueba, el mínimo y máximo tiempo de respuesta del sistema web, el rendimiento del sistema, el porcentaje de error.

Resumen de resultados

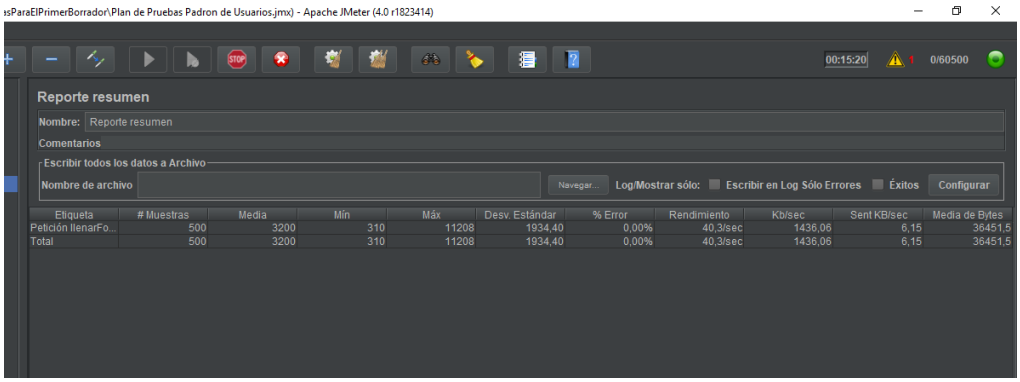


Figura 42 Resumen de Resultados

Elaborado por: Norman Chuquirima, Miguel Silva. (2018)

Conclusiones

- Con un sistema local de Padrón de usuarios, el centro de investigación de la Leche CILEC de Universidad Politécnica Salesiana "Sede Cayambe" puede poner en marcha el registro de información de los usuarios de las diferentes juntas de riego del cantón Cayambe y Pedro Moncayo que no cuentan con acceso al servicio de internet.

- El uso del modelo de diseño MVC, combinado con el patrón de modelo DAO y el patrón de modelo VO, facilitaron el desarrollo del sistema pudiendo reusar el código para las operaciones CRUD de la base de datos.

- El uso de JSF con el framework PrimeFaces ayudo a cumplir el objetivo de una interfaz gráfica amigable, sencilla e intuitiva para el usuario, y facilito el desarrollo del sistema.

- El uso de la metodología XP, facilito entender los requerimientos funcionales cambiantes del cliente, debido a las constantes reuniones realizadas en el periodo de realización del sistema, obteniendo un producto final de fácil usabilidad, pensando en los usuarios campesinos y adultos mayores, a los cuales se les dificultó el manejo de ordenadores, sistema operativo y aplicaciones.

- El sistema web proporciona un mecanismo automatizado para las comunidades de riego, el cual facilita el manejo y administración del sistema de riego.

- Los miembros de los centros de investigación de la Universidad Politécnica Salesiana "Sede Cayambe" al realizar la migración al sistema web pueden evidenciar la optimización tanto de recursos y tiempos de gestión de datos.

Recomendaciones

- Tanto el sistema web como el sistema local de Padrón de usuarios poseen la opción de generar backup de la base de datos, se recomienda realizar respaldos periódicos para preservar la información y evitar pérdidas cada 6 meses.
- Tanto el sistema web, como el sistema local deben ser administrados por una persona con conocimientos básicos en informática y base de datos de los centros de investigación de la Universidad Politécnica Salesiana "Sede Cayambe".
- Se debe capacitar a los representantes de las diferentes juntas de riego acerca de los permisos y tipos de usuarios con los que se manejará el sistema, para una correcta administración al momento de crear y otorgar roles de usuario.
- Para un futuro se recomienda la creación e integración de varios módulos como: Datawarehouse, módulo de reportería gráfica que mejorarán el sistema web.
- El padrón de usuarios permite el registro de información de los habitantes de las comunidades, producción agrícola, pecuaria, inventario ganado, además de información de lote, sistema de riego, se recomienda agregar un módulo de facturación o cobranza que permita la administración y gestión de ingresos recolectados.

BIBLIOGRAFÍA

Balsamiq. Rápido, efectivo y divertido software de wireframing. | Balsamiq. (s. f.).Recuperado 12 de mayo de 2018, de https://balsamiq.com/

Beck, K., & Andres, C. (2004). *Extreme Programming Explained: Embrace Change*. Addison-Wesley Professional.

Beck, K., & Andres, C. (2007). *Extreme programming explained: embrace change* (2. ed., 6. printing). Boston: Addison-Wesley.

Bienvenido a NetBeans y www.netbeans.org, Portal del IDE Java de Código Abierto. (s. f.).Recuperado 26 de julio de 2017, de https://netbeans.org/index_es.html

Debrauwer, L., & Van der Heyde, F. (2016). *UML 2.5 iniciación, ejemplos y ejercicios corregidos*. Cornellà de Llobregat: ENI. Recuperado de http://sfx.urv.cat/urv?url_ver=Z39.88-2004&url_ctx_fmt=info:ofi/fmt:kev:mtx:ctx&ctx_enc=info:ofi/enc:UTF-8&ctx_ver=Z39.88-2004&rfr_id=info:sid/sfxit.com:azbook&sfx.ignore_date_threshold=1&rft.object_id=2240000000000447

Diferencias esenciales entre un wireframe, mockup y prototipo. (2016, septiembre 13).Recuperado 15 de mayo de 2018, de http://blog.aulaformativa.com/diferencias-esenciales-wireframe-mockup-prototipo/

Fernández Alarcón, V. (2006). *Desarrollo de sistemas de información una metodología basada en el modelado*. Barcelona: Edicions UPC. Recuperado de http://bibliotecnica.upc.es/edupc/locate4.asp?codi=OE043XXXN

Fernando Pech-May, Mario A. Gomez-Rodriguez, Luis A. de la Cruz-Diaz, & Salvador U. Lara-Jeronimo. (s. f.). *Desarrollo de Aplicaciones web con JPA, EJB, JSF y PrimeFaces* (p. 9). Carretera Balancán – Villahermosa, Balancán, Tabasco, México: Instituto Tecnológico Superior de los Ríos. Recuperado de http://www.tamps.cinvestav.mx/~fpech/sd/files/paper001.pdf

Flórez Fernández, H. A. (2012). *Programación orientada a objetos usando java*. Bogotá: Ecoe Ediciones. Recuperado de http://site.ebrary.com/id/10623628

Fuentes, J. R. L. (2015). *Desarrollo de Software ÁGIL: Extreme Programming y Scrum*. IT Campus Academy.

Gobierno de Pichincha, & Universidad Politécnica Salesiana. (2015). DOCUMENTO DE CARACTERIZACIÓN DE LA CUENCA DEL RÍO PISQUE EN EL PROYECTO "DETERMINACIÓN DE LA EFICIENCIA DEL USO DEL AGUA EN CULTIVO DE PASTOS CON SISTEMAS COMUNITARIOS DEL MARGEN IZQUIERDO DE LA CUENCA DEL RÍO PISQUE.

Jack 2.0. (2014, junio 23). Patrón de Diseño MVC (Modelo Vista Controlador) y DAO (Data Access Object).Recuperado 13 de mayo de 2018, de https://jossjack.wordpress.com/2014/06/22/patron-de-diseno-mvc-modelo-vista-controlador-y-dao-data-access-object/

JSF - EcuRed. (s. f.).Recuperado 22 de febrero de 2017, de https://www.ecured.cu/JSF

MNDEZ GONZLEZ, G. (2016). *APRENDE A DESARROLLAR CON SPRING FRAMEWORK*. S.l.: CREATESPACE INDEPENDENT P.

Mohedano, J., Saiz, J. M., & Salazar Román, P. (2012). *Iniciación a javascript*. Recuperado de http://www.digitaliapublishing.com/a/20926/

Natsys. (2017). *Testing: Convertite en un experto en QA*. Natsys.

Panda, D., Rahman, R., & Lane, D. (2007). *EJB 3 in action*. Greenwich, CT: Manning Publications Co.

P. Pablo Garrido Abenza. (s. f.). *Comenzando a programar con JAVA.*

PostgreSQL - EcuRed. (s. f.). Recuperado 26 de julio de 2017, de https://www.ecured.cu/PostgreSQL

Pressman, R. S., Ince, D., Ojeda Martín, R., & Joyanes Aguilar, L. (2004). *Ingeniería del software: un enfoque práctico*. Madrid: McGraw-Hill.

Qué es un diagrama de secuencia. (2017, enero 31).Recuperado 14 de junio de 2018, de https://www.lucidchart.com/pages/es/%C2%BFqu%C3%A9-es-un-diagrama-de-secuencia-en-uml

Ramirez, D. P. (2011). *Extreme programming (xp).* Place of publication not identified: Editorial Acad Mica Espa.

Sommerville, I., & Alfonso Galipienso, M. I. (2005). *Ingeniería del software*. Madrid: Pearson Educación.

Urbano López, M. del P. (2015). *Administración y auditoría de los servicios Web. IFCT0509*. Málaga: IC Editorial. Recuperado de http://public.eblib.com/choice/PublicFullRecord.aspx?p=5350015

Van Lancker, L. (2012). *HTML5: los fundamentos del lenguaje*. Cornellà de Llobregat (Barcelona): Ediciones ENI.