UNIVERSIDAD NACIONAL DEL ALTIPLANO

FACULTAD DE INGENIERÍA ESTADÍSTICA E INFORMÁTICA

ESCUELA PROFESIONAL DE INGENIERÍA ESTADÍSTICA E INFORMÁTICA

SISTEMA DE INFORMACIÓN PARA LA AUTOMATIZACIÓN DE LABORATORIOS EN LA UNIVERSIDAD NACIONAL DEL ALTIPLANO PUNO - 2016

TESIS

PRESENTADA POR:

Bach. JORGE VICTOR BALBOA AGUILAR

Bach. YAMYS SADAM HUANACUNI CHURASACARI

PARA OPTAR EL TÍTULO PROFESIONAL DE:

INGENIERO ESTADÍSTICO E INFORMÁTICO

PUNO – PERÚ

2017

UNIVERSIDAD NACIONAL DEL ALTIPLANO

FACULTAD DE INGENIERÍA ESTADÍSTICA E INFORMÁTICA

ESCUELA PROFESIONAL DE INGENIERÍA ESTADÍSTICA E INFORMÁTICA

SISTEMA DE INFORMACIÓN PARA LA AUTOMATIZACIÓN DE LABORATORIOS EN LA UNIVERSIDAD NACIONAL DEL ALTIPLANO PUNO - 2016

TESIS PRESENTADA POR:

Bach. JORGE VICTOR BALBOA AGUILAR

Bach. YAMYS SADAM HUANACUNI CHURASACARI

PARA OPTAR EL TÍTULO PROFESIONAL DE:

INGENIERO ESTADÍSTICO E INFORMÁTICO

APROBADA POR EL JURADO REVISOR CONFORMADO POR:

PRESIDENTE	:	M.Sc. ERNESTO NAYER TUMI FIGUEROA
PRIMER MIEMBRO	:	D.Sc. PERCY HUATA PANCA
SEGUNDO MIEMBRO	:	D.Sc. JOSE PANFILO TITO LIPA
DIRECTOR / ASESOR	:	D.Sc. LEONEL COYLA IDME

Área	:	Informática
Tema	:	Sistema de Información y Base de Datos
Fecha de Sustentación	:	20/12/2017

DEDICATORIA

A dios, por nunca abandonarme en los momentos más difíciles de mi vida, por darme fuerzas cada día para alcanzar mis sueños.

Con mucho respeto y cariño dedico a los seres más importantes en mi vida, mis padres, Miguel Ángel y María Amanda, quienes sin escatimar esfuerzo alguno, han sacrificado gran parte de sus vidas para formarme y educarme, a quienes nunca podré pagar todos sus desvelos ni aun con las riquezas más grandes del mundo, por ello, solo deseo expresarles que mis ideales, esfuerzo y logros han sido también suyos. A mi hermano Genaro Mario, a mis hermanitas Carmen Rosa y Roxana Nataly, por todo su apoyo que siempre me dieron para salir adelante en mi vida personal y profesional.

Los quiero mucho.

JORGE VICTOR

A mis padres Manuel y Juana, por ser el pilar fundamental en todo lo que soy, en toda mi educación, tanto académica como de la vida, por su apoyo incondicional perfectamente mantenido a través del tiempo.

A mi hermana Brenda, por estar conmigo y apoyarme siempre, te quiero mucho.

A mi sobrino Matias, para que veas en mí un ejemplo a seguir.

Todo este trabajo ha sido posible gracias a ellos.

YAMYS SADAM

AGRADECIMIENTO

Sin duda cuando se trata de agradecer el amor, los valores, el impulso, la motivación, el cuidado, la protección, los desvelos y el sacrificio; A nuestros queridos padres quienes a lo largo de toda nuestras vidas nos apoyaron y motivaron en nuestra formación profesional, creyeron en nosotros en todo momento y no dudaron de nuestras habilidades.

Nuestro reconocimiento y agradecimiento a la Universidad Nacional del Altiplano, por ser la casa de estudio que nos brindó la oportunidad de formarnos profesionalmente, con los conocimientos y enseñanzas durante los cinco años de formación, de *forma especial a la Escuela Profesional del Ingeniería Estadística e Informática.*

A nuestros docentes de la Escuela Profesional de Ingeniería Estadística e Informática a quienes les debemos gran parte de nuestros conocimientos, gracias a su paciencia y enseñanza en las sesiones de aprendizaje, nuestro cariño, respeto y admiración por cada uno de ellos.

Un agradecimiento muy grande a nuestros jurados M.Sc. Ernesto Nayer Tumi Figueroa, D.Sc. Percy Huata Panca, D.Sc. Jose Panfilo Tito Lipa, y en especial a nuestro director de tesis D.Sc. Leonel Coyla Idme, quien con su experiencia, paciencia, motivación, asesoramiento y apoyo eficaz hicieron posible la conclusión de la presente tesis.

Y a todas aquellas personas y amigos, que de una y otra forma estuvieron a nuestro lado apoyándonos en todo momento.

¡A todos mil gracias!

JORGE VICTOR BALBOA AGUILAR

YAMYS SADAM HUANACUNI CHURASACARI

ÍNDICE GENERAL

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

ÍNDICE DE ACRÓNIMOS

UNAP	**:** Universidad Nacional del Altiplano Puno
FINESI	**:** Facultad de Ingeniería Estadística e Informática
SI	**:** Sistema de Información
N°	**:** Numero
CI	**:** CodeIgniter
HTML	**:** Lenguaje de Marcado de Híper-Texto
MVC	**:** Modelo Vista Controlador
SUBLIME TEXT	**:** Editor de códigos y texto
MYSQL	**:** Sistema de gestión de base de datos relacional
WORKBENCH	**:** Herramienta visual para diseñar base de datos
XAMPP	**:** servidor web de plataforma, software libre
HTML	**:** Lenguaje de Marcado de Hiper Texto
UML	**:** Lenguaje Unificado de Modelado
PHP	**:** Pre-procesador de Híper-texto
Jquery	**:** Biblioteca multiplataforma de JavaScript
ISO	**:** Organización de Estándares Internacionales
AJAX	**:** Acrónimo de Asynchronous JavaScript And XM

RESUMEN

Es tendencia en el desarrollo de tecnologías de la información, la creación de sistemas de información, porque nos ofrece muchas ventajas de administración, mantenimiento y adaptabilidad. Además, el problema de continuar utilizando métodos antiguos de administración es que nace la necesidad de implementar un sistema para automatizar, y así brindar un servicio ágil, eficiente y seguro en todo el proceso de la administración de laboratorios en la Universidad Nacional del Altiplano Puno. El objetivo principal de la presente investigación fue ayudar a la mejor administración de laboratorios, en el cual se realizó el análisis y diseño del sistema de información a implementar, basándose en los requerimientos de los laboratorios. Para el desarrollo del sistema de información se ha empleado la metodología SCRUM la cual se adecua mejor al desarrollo del sistema por su agilidad y que permite la programación a bajo costo. Para el modelado del sistema de información se utilizó la metodología UML (Lenguaje Unificado de Modelado). Para la validación del sistema que se desarrolló se utilizó la ficha de evaluación de calidad del producto estándar ISO-9126. Por lo que se concluye que el sistema de información ayudo a mejorar la administración de laboratorios, a través de una interface web amigable, fiable, segura y rápida; para lo cual se utilizó herramientas de software que permitieron acceder de forma fiable a la base de datos para realizar los procesos de registro, modificación, búsqueda y emisión de reportes.

Palabras Clave: Sistemas de información, Automatización, Procesos, Aplicación y Laboratorios.

ABSTRACT

It is a trend in the development of information technologies, the creation of information systems, because it offers us many advantages of administration, maintenance and adaptability. In addition, the problem of continuing to use old methods of administration is that the need arose to implement a system to automate, and thus provide an agile, efficient and safe service throughout the process of laboratory administration at the National University of Puno Altiplano. The main objective of this research was to help the best laboratory administration, in which the analysis and design of the information system to be implemented was carried out, based on the requirements of the laboratories. For the development of the information system, the SCRUM methodology has been used, which is better suited to the development of the system due to its agility and which allows programming at low cost. For the modeling of the information system, the UML (Unified Modeling Language) methodology was used. For the validation of the system that was developed, the quality evaluation form of the standard product ISO-9126 was used. So it is concluded that the information system helped to improve the administration of laboratories, through a friendly, reliable, safe and fast web interface; for which software tools were used that allowed to access the database in a reliable way to perform the processes of registration, modification, search and issuance of reports.

Keywords: Information systems, Automation, Processes, Application and Laboratories.

CAPÍTULO I

INTRODUCCIÓN

Los sistemas de información están cambiando la forma de operar de las organizaciones actuales. A través de su uso se logran importantes mejoras, pues automatizan los procesos operativos de las empresas, proporcionan información de apoyo al proceso de toma de decisiones y lo más importante, facilitan el logro de ventajas competitivas a través de su implantación en las empresas.

El crecimiento de la tecnología en los últimos años, ha tenido una incursión significativa en nuestro país, haciendo de las tecnologías un medio eficiente y masivo para las empresas e instituciones. Es por ello que la plataforma de investigación del Vicerrectorado de Investigación, brindaron a los autores de esta tesis la oportunidad de desarrollar un Sistema de Información utilizando la tecnología web.

Es necesario que las instituciones tanto públicas como privadas adapten sus procesos a las nuevas tendencias y retos que presentan los nuevos avances tecnológicos, esto hace que sea necesario que estas instituciones busquen una

solución, automatización y mejora de sus procesos con el fin de no quedar rezagados y brindar un mejor servicio a sus clientes.

El diseño del sistema se realizó un proceso previo a la recolección de información de los datos de los jefes de laboratorios, equipos y materiales de la distintas escuelas profesionales de la Universidad Nacional del Altiplano Puno, una vez recolectada la información se procedió a la implementación del sistema de información, para lo cual se utilizó PHP y JavaScript como lenguaje de programación, MySQL como gestor de base de datos, HTML y CSS para el diseño del sistema.

En el presente trabajo se presenta una solución tecnológica al proceso de administración de laboratorios en la Universidad Nacional del Altiplano Puno, para que dicho proceso se realizó de manera automatizada. La investigación se ha elaborado de acuerdo a la organización de información.

1.1. Planteamiento del Problema

En la actualidad, la ventaja que tienen los sistemas de información es que pueden hacer las tareas más rápidas, más cómodas y más fáciles para los usuarios, ya que reducen tiempo, costo y esfuerzo. Es así que la importancia de contar con sistemas de información en organizaciones ayuda a agilizar los procesos con los cuales se desenvuelve la institución, lo que conlleva a mejorar el rendimiento y desempeño de la misma. Si no se cuenta con un sistema que administre y almacene los procesos que se realizan, ocasiona la falta de evidencias y reportes que son de gran importancia para el desarrollo competitivo con las demás instituciones del país.

La Universidad Nacional del Altiplano Puno, no cuenta con un sistema de información automatizada para la administración de equipos y/o materiales de cada uno los laboratorios de las distintas escuelas profesionales que permita ayudar mejor la administración. Por lo que en la actualidad se tiene desinformación y se ocasiona confusión de datos, reincidencia de información, pérdida de datos al momento de realizar el intercambio de información al utilizar dispositivos de almacenamiento, no existe un registro exclusivo y falta de actualización automática en los equipos y materiales de los laboratorios de la Universidad Nacional del Altiplano Puno.

Desde el punto de vista del egresado de la Escuela Profesional de Ingeniería Estadística e Informática y con el afán de solucionar el problema, plasmamos nuestros conocimientos para implementar un sistema de información que mejore la administración de laboratorios en la Universidad Nacional del Altiplano Puno, Así como también, realizar el análisis y diseño del sistema de información a implementar, basándose en los requerimientos de los laboratorios, diseñar un modelo de base de datos acorde a los requerimientos de almacenamiento y manipulación de datos, Definir la arquitectura y herramientas con la cual se implementara el sistema de información, diseñar una interfaz gráfica intuitiva, que permita al usuario interactuar con el sistema con facilidad, implementar y realizar las pruebas correspondientes del sistema de información. De acuerdo a la problemática anterior se formula la siguiente interrogante:

1.2. Formulación del Problema

¿Cómo un sistema de información ayudará la mejor administración de laboratorios en la Universidad Nacional del Altiplano?

1.3. Hipótesis de la Investigación

El sistema de información ayudó en la mejor administración de laboratorios en la Universidad Nacional del Altiplano Puno.

1.4. Justificación de la Investigación

En la actualidad los sistemas información son muy utilizados en los institutos y universidades, esto gracias a su facilidad de acceso al sistema y como también se puede actualizar múltiples usuarios concurrentes, esto es posible ya que solo requiere un sistema para la gestión académica.

La universidad Nacional del altiplano, no cuenta con un sistema de apoyo para el control de la información de laboratorios de las distintas escuelas profesionales, es por esta razón que planteamos la siguiente solución, un “Sistema de información para la automatización de laboratorios en la Universidad Nacional del Altiplano Puno”, el sistema desarrollado permite el apoyo operacional y el control de la gestión administrativa de los laboratorios de escuelas profesionales.

El sistema cuenta con un acceso de tecnología vía web que tendrá un módulo de seguridad, en la cual se administrará las cuentas y las claves necesarias para el acceso al sistema, con esto podemos transparentar los procesos de cada usuario.

1.5. Objetivos de la Investigación

1.5.1. Objetivo General

Implementar un sistema de información que permita ayudar la mejor administración de laboratorios en la Universidad Nacional del Altiplano Puno.

1.5.2. Objetivos Específicos

- Diseñar e implementar el sistema de información y realizar las pruebas correspondientes.
- Validar la implementación del sistema con el ISO 9126, para el uso y fiabilidad del software.
- Evaluar el rendimiento y la aceptación del sistema de información en los usuarios.

CAPÍTULO II

REVISIÓN DE LITERATURA

2.1. Marco Teórico

2.1.1. Antecedentes de la Investigación

Tesis Internacionales

Ramírez Rodríguez, C., & Vélez Sabando, G. (2015). Concluye que el levantamiento de información permitió determinar las necesidades del Hotel-Laboratorio así como también identificar los procesos que se realizaban, A través de ello la información obtenida facilitó el desarrollo del contenido estático de la aplicación web. Mediante la realización de pruebas en el sistema permitió comprobar el correcto funcionamiento de la aplicación y finalmente evidencian si existe o no la mejora en los tiempos de los proceso realizados.

Tesis Nacionales

Córdova Forero, J. (2014). Se logró comprender en forma correcta y clara el proceso de matrículas y pagos del Centro de informática,

identificar todos los requerimientos funcionales y no funcionales asociados a los procesos de matrícula y pagos en el Centro de Informática, mediante las proyecciones realizadas sobre las matriculas se lograría que la Universidad obtenga ganancias de más de un millón de soles.

Talledo Farfan, J. (2015). Concluye que la implementación del sistema de información del laboratorio clínico con tecnología web al Centro de Salud Miguel Checa-Sojo, distrito del Miguel Checa-Sullana, mejoró la atención integral a los pacientes. A través del sistema permitió al personal asistencial, la búsqueda rápida y eficaz de los resultados del análisis clínico de los pacientes que acuden al centro de salud. Mediante la aplicación de herramienta orientada a objetos de software open source, es posible desarrollar e implementar el sistema de laboratorio clínico en el Centro de Salud de Miguel Checa- Sojo.

Tesis Locales

Torres Cruz, F. (2016). Concluye que el desarrollo de una plataforma ágil demostrando el uso se aminora el tiempo de inscripción, registro, sorteo, revisión, corrección y dictamen de los proyectos de tesis de pregrado optimizando de esta manera todos los procedimientos. La eficiencia de la plataforma fue calificada por los usuarios quienes afirman que PILAR agilizo óptimamente los procedimientos. Como también la seguridad de la plataforma es indicada para evitar cualquier vulnerabilidad de la información aquí consignada.

Quispe Puma, J. (2017). Concluye que con el desarrollo e implementación del sistema de automatización para los procesos de administración y búsqueda de la biblioteca especializada en la Institución Educativa Secundaria Mariano Hilario Cornejo “Comercio 32” Juliaca 2010, la atención es más rápida e eficiente en el proceso de registro, búsqueda y emisión de libros disponibles con un nivel de aceptación de 110 puntos, de acuerdo a la ficha de evaluación aplicada a los operadores del Sistema de automatización. A través de una interface web utilizando herramientas de software libre como PHP y DREAMWEAVER CS3.0, herramientas que permitieron acceder de forma fiable a la base de datos para realizar los procesos de registro, búsqueda y emisión de libros disponibles. Se validó el funcionamiento del sistema con la ficha de evaluación ISO – 9126.

2.2. Marco Conceptual

2.2.1. Sistemas de Información

(Laudon 2015) Un sistema de información es un conjunto de componentes interrelacionados que reúne (u obtiene), procesa, almacena, y distribuye para apoyar la toma de decisiones y el control en una organización.

Un sistema de información (SI) es un conjunto de elementos interrelacionados con el propósito de prestar atención a las demandad de información de una organización, para elevar el nivel de conocimientos que permitan un mejor apoyo a la toma de decisiones y desarrollo de acciones. Un sistema de información realiza cuatro actividades básicas:

Entrada de información: Es el proceso mediante el cual el sistema de información toma los datos que requiere para procesar la información. Las entradas pueden ser manuales o automáticas. Las manuales son las que se proporcionan en forma directa por el usuario, mientras que las automáticas son datos e información que provienen o son tomadas de otros sistemas o módulos. Este último se denomina interfaces automáticas. Las unidades típicas de entrada de datos a las computadoras son las terminales, las cintas magnéticas, las unidades de diskettes, los códigos de barras, los escáneres, la voz, los monitores sensibles al taco, el teclado y el mouse, entre otras.

Almacenamiento de información: El almacenamiento es una de las actividades o capacidades más importantes que tiene una computadora, ya que a través de esta propiedad el sistema puede recordar la información guardada en la sección o proceso anterior. Esta información suele ser almacenada en estructuras de información denominadas archivos. La unidad típica de almacenamiento son los discos magnéticos o discos duros, los discos flexibles o diskettes y los discos compactos (CD-ROM).

Procesamiento de información: Es la capacidad del sistema de información para efectuar cálculos de acuerdo con una secuencia de operaciones preestablecidas. Estos cálculos pueden efectuarse con cálculos introducidos recientemente en el sistema o bien con datos que están almacenados. Esta característica de los sistemas permite la transformación de datos fuente en información que puede ser utilizada en

la toma de decisiones, lo que hace posible, entre otras cosas, que en un tomador de decisiones genere una proyección financiera a partir de los que contiene un estado de resultados o un balance general de una año base.

Salida de información: La salida es la capacidad de un sistema de información para sacar la información procesada o bien datos de entrada al exterior. Las unidades típicas de salida son las impresoras, terminales, la voz, los graficadores y los plotters, entre otros. Es importante aclarar que la salida de un sistema de información puede constituir la entrada a otro sistema de información o módulo. En este caso, también existe un interface automática de salida.

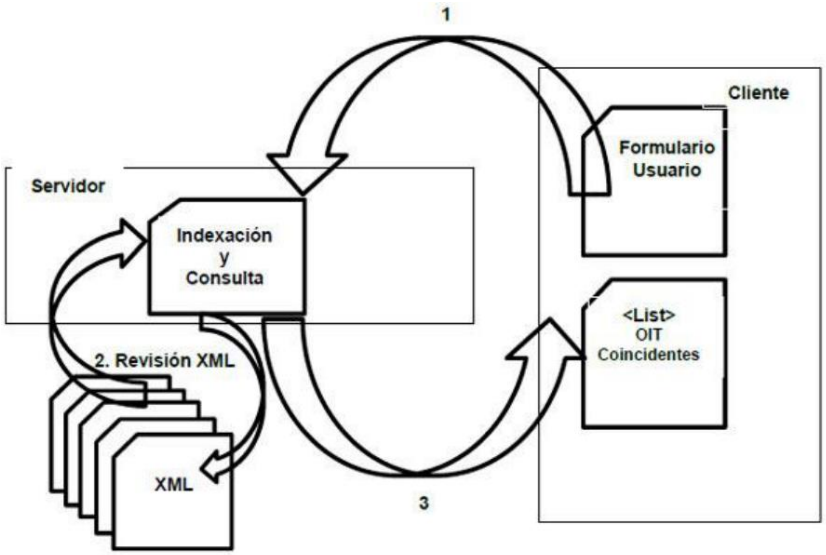
2.2.2. Sistemas de Búsqueda

(Tramullas 2011) Un sistema o motor de búsqueda es el mecanismo por la cual la información almacenada puede ser recuperada puede ser recuperada por el usurario, mediante una interfaz provista para comunicarle con la base de datos y realizar operaciones para extraer la información que se solicita. En un laboratorio digital no es cuestionable la utilización de un sistema de búsqueda.

El principal servicio es la consulta de información y como no se puede tener toda la colección en línea al mismo tiempo, por muy pequeña que sea, siempre será necesario hacer una revisión y extracción de los materiales que cumplen con los intereses del usuario. En muchos casos se requiere más de un sistema de información para una colección de cinco documentos grandes, que en una colección de 20 documentos pequeños

bien definidos. Los científicos de la información y los laboratorios han estudiado por décadas los hábitos de búsqueda de los usuarios. Recientemente, estos estudios se refieren a los sistemas de información tradicionales, tales como encontrar un patrón para encontrar las necesidades de información, o como hacer sistemas sencillos para la búsqueda de información en servicios en línea de laboratorios u otras base de datos.

Figura N° 1: Sistema de búsqueda



Fuente: Tramullas 2011

2.2.3. Ingeniería de Software

La ingeniería de software es una disciplina formada por un conjunto de métodos, herramientas y técnicas que se utilizan en el desarrollo de programas informáticos. Esta disciplina trasciende la actividad de programación, que es el pilar fundamental a la hora de crear una aplicación.

Por lo tanto, la ingeniería de software incluye el análisis previo de la situación, el diseño del proyecto, el desarrollo del software, las pruebas necesarias para confirmar su correcto funcionamiento y la implementación del sistema.

El desarrollo del software implica del software implica lo que se conoce como ciclo de vida del software, que está formado por cuatro etapas: concepción, elaboración, construcción y transición. La concepción fija el alcance del proyecto y se desarrolla el modelo de negocio; la elaboración define el plan del proyecto, detalla las características y fundamenta la arquitectura; la construcción es el desarrollo del producto; y la transición es la transferencia del producto terminado a los usuarios.

2.2.4. Software

Es el conjunto de los programas de cómputo, procedimientos, reglas, documentación y datos asociados, que forman parte de las operaciones de un sistema de computación (Estándar 729 del IEEE).

Se conoce también software al equipo lógico o soporte lógico de un sistema informático, que comprende el conjunto de los componentes lógicos necesarios que hace posible la realización de tareas específicas, en contraposición a los componentes físicos que son llamados hardware. Los componentes lógicos incluyen, entre muchos otros, las aplicaciones informáticas, tales como el procesador de texto, software de sistema y sistema operativo. (https://es.wikipedia.org/wiki/Software).

2.2.5. Programación Orientada a Objetos

(Roldan 2016) La programación orientada a objetos (*POO*, u *OOP* según sus siglas en inglés) es un paradigma de programación que usa objetos en sus interacciones, para diseñar aplicaciones y programas informáticos.

La programación orientada a objetos es una de las formas más populares de programar y viene teniendo gran acogida en el desarrollo de proyectos de software desde los últimos años. Esta acogida se debe a sus grandes capacidades y ventajas frente a las antiguas formas de programar. Todo objeto necesita interactuar con su medio por lo cual se definen cinco propiedades fundamentales:

- **Abstracción de datos:** Nos permite la creación de nuevos tipos de datos a partir de los ya existentes en un lenguaje determinado.
- **Encapsulación:** Protección de los datos privados que no pueden ser accesados por objetos externos.
- **Herencia:** Propiedad que permite la creación de un objeto hijo de uno superior heredando las mismas características.
- **Polimorfismo:** Propiedad mediante la cual un grupo de objetos reaccionan de diferente manera a un mismo mensaje.
- **Reutilización:** Posibilidad de volver a utilizar el código existente sin mucho esfuerzo.

2.2.6. Metodología Orientada a Objetos

La metodología orientada a objetos ha derivado de las metodologías anteriores a éste. Así como los métodos de diseño estructurado realizados guían a los desarrolladores que tratan de construir sistemas complejos utilizando algoritmos como sus bloques fundamentales de construcción, similarmente los métodos de diseño orientado a objetos han evolucionado para ayudar a los desarrolladores a explotar el poder de los lenguajes de programación basados en objetos y orientados a objetos, utilizando las clases y objetos como bloques de construcción básicos.

Actualmente el modelo de objetos ha sido influenciado por un número de factores no sólo de la Programación Orientada a Objetos, POO (Object Oriented Programming, OOP por sus siglas en inglés). Además, el modelo de objetos ha probado ser un concepto uniforme en las ciencias de la computación, aplicable no sólo a los lenguajes de programación sino también al diseño de interfaces de usuario, bases de datos y arquitectura de computadoras por completo. La razón de ello es, simplemente, que una orientación a objetos nos ayuda a hacer frente a la inherente complejidad de muchos tipos de sistemas.

2.2.7. Análisis y Diseño de Sistemas Orientado a Objetos

(Roldán 2016) Para el desarrollo de software orientado a objetos no basta usar un lenguaje orientado a objetos. También se necesitará realizar un análisis y diseño orientado a *objetos*. El modelamiento visual es la clave para realizar el análisis OO (*orientado*

a objeto). Desde los inicios del desarrollo de software OO han existido diferentes metodologías para hacer esto del modelamiento, pero sin lugar a duda, el Lenguaje de Modelamiento Unificado (UML) puso fin a la guerra de metodologías. Según los mismos diseñadores del lenguaje UML, éste tiene como fin modelar cualquier tipo de sistemas (no solamente de software) usando los conceptos de la orientación a objetos. Y además, este lenguaje debe ser entendible para los humanos y máquinas. Actualmente en la industria del desarrollo de software tenemos al UML como un estándar para el modelamiento de sistemas OO. Fue la empresa Racional que creó estas definiciones y especificaciones del estándar UML, y lo abrió al mercado. La misma empresa creó uno de los programas más conocidos hoy en día para este fin; el Racional Rose, pero también existen otros programas como el Poseidon que trae licencias del tipo community edition que permiten su uso libremente.

El UML consta de todos los elementos y diagramas que permiten modelar los sistemas en base al paradigma orientado a objetos. Los modelos orientados a objetos cuando se construyen en forma correcta, son fáciles de comunicar, cambiar, expandir, validar y verificar. Este modelamiento en UML es flexible al cambio y permite crear componentes plenamente reutilizables.

2.2.8. Lenguaje de Modelado Unificado UML

Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema. UML ofrece un estándar para describir un "plano" del sistema (modelo), incluyendo aspectos conceptuales tales como

procesos, funciones del sistema, y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y compuestos reciclados. Es importante remarcar que UML es un "lenguaje de modelado" para especificar o para describir métodos o procesos. Se utiliza para definir un sistema, para detallar los artefactos en el sistema y para documentar y construir. En otras palabras, es el lenguaje en el que está descrito el modelo.

El Lenguaje de Modelado Unificado preescribe un conjunto de notaciones y diagramas estándar para modelar sistemas orientados a objetos, y describe la semántica esencial de lo que estos diagramas y símbolos significan. Mientras que ha habido muchas notaciones y métodos usados para el diseño orientado a objetos, ahora los modeladores sólo tienen que aprender una única notación.

2.2.9. Diagramas UML

La finalidad de los diagramas es presentar diversas perspectivas de un sistema, a las cuales se les conoce como modelo. Recordemos que un modelo es una representación simplificada de la realidad; el modelo UML describe lo que supuestamente hará un sistema, pero no dice cómo implementar dicho sistema. A continuación se describirán los diagramas más comunes del UML y los conceptos que representan:

2.2.9.1. Diagrama de Clases

El propósito de este diagrama es el de representar los objetos fundamentales del sistema, es decir los que percibe el usuario y con los

que espera tratar para completar su tarea en vez de objetos del sistema o de un modelo de programación. Un diagrama de clases está compuesto por los siguientes elementos.

Clase: Es la unidad básica que encapsula toda la información de un Objeto (un objeto es una instancia de una clase). A través de ella podemos modelar el entorno en estudio. Los atributos o características de las clases pueden ser de tres tipos, según el grado de comunicación y visibilidad de ellos con el entorno, estos son:

Tabla N° 1: Tipos de atributos de la clase

Nombre	Descripción
Públicos (+)	Indican que el atributo será visible tanto fuera como dentro de la clase, es decir, es accesible desde todos lados.
Privados (-)	Indican que el atributo solo será accesible desde dentro de la clase (solo sus métodos lo pueden acceder).
Protegidos (#)	Indican que el atributo no será accesible desde afuera de la clase, pero si podrá ser accesado por métodos de la clase además de la subclases que se deriven.

Fuente: Elaboración propia

Relaciones entre clases: Una relación es un término general que abarca los tipos específicos de conexiones lógicas que se pueden encontrar en los diagramas de clases y objetos. UML presenta las siguientes relaciones:

Herencia: Indica que una subclase hereda los métodos y atributos especificados por una Súper Clase, por ende la Subclase además de

poseer sus propios métodos y atributos, poseerá las características y atributos visibles de la Súper Clase).

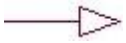
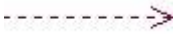
Agregación: Es un tipo especial de asociación que representa una relación estructural entre las clases donde el llamado agregado indica el todo y el componente es una parte del mismo.

Composición: Es un tipo de agregación donde la relación de posesión es tan fuerte como para marcar otro tipo de relación.

Asociación: Relación estructural que describe un conjunto de conexiones entre objetos de forma bidireccional.

Dependencia: Es una relación de uso, es decir una clase usa a otra, que la necesita para su cometido. Se representa con una flecha discontinua que va desde la clase utilizadora a la clase utilizada. Con la dependencia se muestra que un cambio en la clase utilizada puede afectar el funcionamiento de la clase utilizadora, pero no al contrario.

Tabla N° 2: Símbolo de relaciones de clases

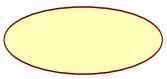
Nombre	Símbolo
Herencia	
Agregación	
Composición	
Asociación	
Dependencia	

Fuente: Elaboración propia

2.2.9.2. Diagrama de Casos de Uso

Un caso de uso es una descripción de las acciones de un sistema desde el punto de vista del usuario. Es una herramienta valiosa dado que es una técnica de aciertos y errores para obtener los requerimientos del sistema, justamente desde el punto de vista del usuario. Los diagramas de caso de uso modelan la funcionalidad del sistema usando actores y casos de uso. Los casos de uso son servicios o funciones provistas por el sistema para sus usuarios. Los diagramas de caso de uso modelan la funcionalidad del sistema usando actores y casos de uso. Los casos de uso son servicios o funciones provistas por el sistema para sus usuarios.

Tabla N° 3: Elementos de casos de uso

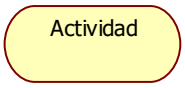
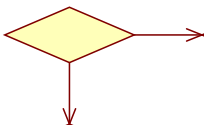
Nombre	Símbolo	Descripción
Actor		Es un rol que un usuario juega con respecto al sistema. Es importante destacar el uso de la palabra rol, pues con esto se especifica que un Actor no necesariamente representa a una persona en particular, sino más bien la labor que realiza frente al sistema.
Caso de uso		Es una operación y/o tarea específica que se realiza tras una orden de algún agente externo, sea desde una petición de un actor o bien desde la invocación desde otro caso de uso.
Relaciones		**Asociación:** Es el tipo de relación más básica que indica la invocación desde un actor o caso de uso a otra operación.
		Dependencia: Es una forma muy particular de relación entre clases, en la cual una clase depende de otra, es decir, se crea.
		**Generalización:** Este tipo de relación es uno de los más utilizados, cumple una doble función dependiendo de su estereotipo, que puede ser de Uso o de Herencia.

Fuente: Elaboración propia

2.2.9.3. Diagrama de Actividades

Diagrama de actividades ilustra la naturaleza dinámica de un sistema mediante el modelado del flujo ocurrente de actividad en actividad. Una actividad representa una operación en alguna clase del sistema y que resulta en un cambio en el estado del sistema. Típicamente, los diagramas de actividad son utilizados para modelar el flujo de trabajo interno de una operación.

Tabla N° 4: Elementos de diagrama de actividades

Nombre	Símbolo	Descripción
Acción		Nodo de Actividad primitiva ejecutable de asignación o computación.
Nodo de Inicio		Nodo de control que indica el inicio de un flujo de control cuando una actividad es invocada.
Nodo fin de actividad		Nodo de control que indica el fin de datos los flojos dentro de una muestra el fin de la actividad.
Flujo de control		Eje de actividades para flujo de control que conecta dos acciones. Usado para indicar secuencia.
Nodo de decisión		Nodo de control que selecciona entre dos o más flujos de salida.

Fuente: Elaboración propia

2.2.9.4. Diagrama de Secuencia

Los diagramas de secuencia se lo realizan por cada caso de uso, nos permite tener claro que métodos se va a implementar. Los diagramas de clases y los de objetos representan información estática. No obstante,

en un sistema funcional, los objetos interactúan entre sí, y tales interacciones suceden con el tiempo. El diagrama de secuencias UML muestra la mecánica de la interacción con base en tiempos.

2.2.9.5. Diagrama de Colaboración

El diagrama de colaboraciones describe las interacciones entre los objetos en términos de mensajes secuenciados. Los diagramas de colaboración representan una combinación de información tomada de los diagramas de clases, de secuencias y de casos de uso, describiendo el comportamiento, tanto de la estructura estática, como de la estructura dinámica de un sistema.

2.2.9.6. Diagrama de Estados

(Rumbaugh, Jacobson, & Booch, 2004). En cualquier momento, un objeto se encuentra en un estado particular, la luz está encendida o apagada, el auto en movimiento o detenido, la persona leyendo o cantando, etc. El diagrama de estados UML captura esa pequeña realidad. Está representado principalmente por los siguientes elementos.

- Estado
- Elemento
- Transición

2.2.9.7. Diagrama de Componentes

(Rumbaugh, Jacobson, & Booch, 2004) Un diagrama de componentes muestra dependencias entre los componentes. Cada

componente ofrece algunas interfaces y utiliza otras. Si las dependencias entre componentes se hacen a través de interfaces, los componentes se pueden sustituir por otros componentes que realicen las mismas interfaces.

2.2.9.8. Diagrama de Distribución

El diagrama de distribución UML muestra la arquitectura física de un sistema informático. Puede representar a los equipos y a los dispositivos, y también mostrar sus interconexiones y el software que se encontrará en cada máquina.

2.2.9.9. Diagrama de Despliegue

Son aquellos que muestran las relaciones físicas entre los componentes de software y hardware en el sistema de desarrollo, es decir cómo se encuentra y se mueven los componentes y los objetos.

Un diagrama de despliegue modela la arquitectura en tiempo de ejecución de un sistema mostrando la configuración de los elementos de hardware y mostrando como los elementos y artefactos del software se trazan en esos nodos.

2.2.10. Pruebas o Métricas del Software

Las métricas de software tienen un papel decisivo en la obtención de un producto de alta calidad, porque determinan mediante estadísticas basadas en la experiencia, el avance del software y el cumplimiento de parámetros requeridos. Siempre habrá elementos cualitativos para la

creación de software. El problema estriba en que la valoración cualitativa puede no ser suficiente. Un ingeniero del software necesita criterios objetivos para guiarse en el diseño de datos, de la arquitectura, de las interfaces y de los componentes. El verificador necesita una referencia cuantitativa que le ayude en la selección de los casos de prueba y de sus objetivos. Las métricas técnicas facilitan una base para que el análisis, diseño, codificación y prueba puedan ser conducidos más objetivamente y valorados más cuantitativamente.

La palabra métrica, es muy común asociarla con las palabras medición y medida, aunque estas tres son distintas. La medición “es el proceso por el cual los números o símbolos son asignados a atributos o entidades en el mundo real tal como son descritos de acuerdo a reglas claramente definidas”.

La prueba de software es un elemento crítico para la garantía de calidad del software y representa una visión final de las especificaciones del diseño y la codificación.

El objetivo es diseñar pruebas que sistemáticamente saquen a la Luz diferentes clases de errores, haciéndolos con la menor cantidad de tiempo y esfuerzo. Las normas que puedan servir para objetos de prueba son:

- La prueba es un proceso de ejecución de un programa con la intensión de descubrir un error.

- Un buen caso de prueba es aquel que tiene una alta probabilidad de mostrar un error no descubierto hasta entonces.
- Una prueba tiene éxito si se descubre un error no detectado hasta entonces.

2.2.11. Organización Internacional de Estandarización 9126

Esta norma internacional fue publicada en 1992, la cual es usada para la evaluación de la calidad de software, La norma ISO/IEC 9126 permite especificar y evaluar la calidad del software desde diferentes criterios asociados con adquisición, requerimientos, desarrollo, uso, evaluación, soporte, mantenimiento, aseguramiento de la calidad y auditoria de software.

El objetivo de esta norma ISO/IEC 9126 es proporcionar un modelo de calidad que sirve como elemento central en un proceso de evaluación. El modelo de calidad que propone la norma puede aplicarse a cualquier tipo de software incluido el desarrollo para el ámbito educativo.

Figura N° 2: Norma de evaluación ISO/IEC 9126

Fuente: Estándar ISO/IEC 9126

Características Generales:

a. Funcionalidad: Es la capacidad de software de cumplir y proveer las funciones para satisfacer las necesidades explicitas e implícitas cuando es utilizado en condiciones específicas. Se divide en 5 criterios:

Adecuación: La capacidad del software para proporcionar un conjunto apropiado de funciones para tareas específicas y objetivos de los usuarios.

Exactitud: La capacidad del software para proporcionar los resultados o efectos correctos y con el grado de precisión acordado.

Interoperabilidad: La capacidad del software para interactuar con uno o más sistemas especificados.

Seguridad: La capacidad del software para proteger la información y los datos de manera que los usuarios o los sistemas no autorizados no puedan acceder a ellos para realizar operaciones, y la capacidad de aceptar el acceso a los datos de los usuarios o sistemas autorizados.

Conformidad: La capacidad del software de cumplir los estándares referentes a la funcionalidad.

b. Confiabilidad: Es la capacidad para asegurar un nivel de funcionamiento adecuado cuando es utilizado en condiciones específicas, se divide en 4 criterios:

Madurez: La capacidad que tiene el software para evitar fallas cuando encuentra errores.

Tolerancia a errores: La capacidad que tiene el software para mantener un nivel de rendimiento determinado en caso de errores en el software.

Recuperabilidad: la capacidad que tiene el software para restablecer un determinado nivel de rendimiento y recuperar los datos afectados directamente en el caso de una falla.

Conformidad: la capacidad del software de cumplir a estándares, convenciones y regulaciones relacionadas a la fiabilidad.

c. Usabilidad: Es la capacidad del software de ser entendido, aprendido, y usado en forma fácil y atractiva. La usabilidad está determinada por los usuarios finales y los usuarios indirectos del software, dirigidos a todos los ambientes, a la preparación del uso y el resultado obtenido.

Entendimiento: La capacidad que tiene el software para permitir al usuario que entienda si el software es adecuado, y como debe de utilizarse para determinadas tareas y bajo ciertas condiciones de uso.

Aprendizaje: La forma como el software permite al usuario aprender su uso. También es importante considerar la documentación.

Operabilidad: La manera como el software permite al usuario operarlo y controlarlo.

Atractividad: La presentación del software debe ser atractivo al usuario. Esto se refiere a las cualidades del software para hacer más agradable al usuario.

Conformidad: La capacidad del software de cumplir los estándares o normas relacionadas a su usabilidad.

d. Eficiencia: La eficiencia del software es la forma del desempeño adecuado, de acuerdo al número de recursos utilizados según las condiciones planteadas.

Comportamiento temporal: La capacidad del software para proporcionar tiempos de respuestas y de procesamiento apropiados cuando realiza sus funciones bajo condiciones determinadas.

Utilización de recursos: La capacidad del software para utilizar cantidades y tipos de recursos apropiados cuando el software realiza su función bajo determinadas condiciones establecidas.

Conformidad: La capacidad que tiene el software para cumplir con los estándares o convenciones relacionadas a la eficiencia.

e. Capacidad de mantenimiento: Es la cualidad que tiene el software para ser modificado. Incluyendo correcciones o mejoras del software, a cambio en el entorno, y especificaciones de requerimientos funcionales.

Analizabilidad: La capacidad del software de diagnosticar sus deficiencias o causas de fallos o de identificar las partes que deben ser modificadas.

Cambiabilidad: La capacidad del software para que la implementación de una modificación se pueda realizar, incluye también codificación, diseño y documentación de cambios.

Estabilidad: La forma como el software evita efectos inesperados para modificaciones del mismo.

Facilidad: La forma como el software permite realizar pruebas a las modificaciones sin poner el riesgo los datos.

Conformidad: La capacidad que tiene el software para cumplir con los estándares de facilidad de mantenimiento.

f. Portabilidad: La capacidad que tiene el software para ser trasladado de un entorno a otro. Esta referido por los siguientes sub atributos:

Adaptabilidad: La capacidad del software para ser adaptado para ambientes determinados sin realizar acciones o aplicar medios, más que los proporcionados para este propósito para el software considerado.

Facilidad de instalación: La capacidad del software para ser instalado en un entorno específico o por el usuario final.

Coexistencia: La capacidad que tiene el software para coexistir con otro o varios software, la forma de compartir recursos comunes con otro software o dispositivo.

Reemplazibilidad: La capacidad que tiene el software para ser remplazado por otro software del mismo tipo, y para el mismo objetivo.

Conformidad: La capacidad que tiene el software para cumplir con los estándares relacionados a la portabilidad.

g. **Calidad de uso:** Es la calidad del software que el usuario final refleja, la forma como el usuario final logra realizar los procesos con satisfacción, eficiencia y exactitud.

Eficacia: La capacidad del software para permitir a los usuarios finales realizar los procesos con exactitud e integridad.

Productividad: La forma como el software permite a los usuarios emplear cantidades apropiadas de recursos, en relación a la eficacia lograda en un contexto específico de uso.

Seguridad: Se refiere al que el software no tenga niveles de riesgo para causar daño a las personas, instituciones, software, propiedad intelectual o entorno.

Satisfacción: La satisfacción es la respuesta del usuario a la interacción con el software, e incluye las actitudes hacia el uso del mismo.

2.2.12. Modelo de Proceso del Software

Metodologia Scrum

Scrum es un proceso en el que se aplican de manera regular un conjunto de buenas prácticas para trabajar colaborativamente, en equipo, y obtener el mejor resultado posible de un proyecto. Estas prácticas se apoyan unas a otras y su selección tiene origen en un estudio de la manera de trabajar de equipos altamente productivos.

En Scrum se realizan entregas parciales y regulares del producto final, priorizadas por el beneficio que aportan al receptor del proyecto. Por ello, Scrum está especialmente indicado para proyectos en entornos complejos, donde se necesita obtener resultados pronto, donde los requisitos son cambiantes o poco definidos, donde la innovación, la competitividad, la flexibilidad y la productividad son fundamentales.

Scrum es un proceso ágil para desarrollar software que fue aplicado por primera vez por Ken Schwaber y Jeff Sutherland., quienes lo documentaron en detalle en el libro Agile Software Development with Scrum. Esta metodología centra su atención en las actividades de Gerencia y no especifica prácticas de Ingeniería. Fomenta el surgimiento de equipos auto dirigido cooperativo y aplica inspecciones frecuentes como mecanismo de control. Scrum parte de la base de que los procesos definidos funcionan bien sólo si las entradas están perfectamente definidas y el ruido, ambigüedad o cambio es muy pequeño. Por lo tanto, resulta ideal para proyectos con requerimientos inestables, ya que fomenta el surgimiento de los mismos.

El ciclo de vida definido por Scrum es incremental iterativo y se caracteriza por ser muy adaptable.

Principales características:

- Equipos auto dirigidos.
- Utiliza reglas para crear un entorno ágil de administración de proyectos.
- No prescribe prácticas específicas de ingeniería.

- Los requerimientos se capturan como ítems de la lista Product Backlog.
- El producto se construye en una serie de Sprints de un mes de duración.

Principales elementos de la metodología

Herramientas

- Product Backlog
- Sprint Backlog

Prácticas

- Sprints
- Sprint Planning Meeting
- Daily Meetings
- Sprint Review Meeting
- Design Review Meeting
- Stabilization Sprints
- Meta Scrums

Roles y responsabilidades

- Scrum Master
- Product Owner
- Scrum Team
- Customer
- Management

Figura N° 3: Roles, artefactos y eventos principales de SCRUM

Fuente: Deemer et al., (2009)

2.2.13. Base de Datos

Una Base de Datos, llamado también banco de datos es un conjunto de Información relacionada que se encuentra agrupada o estructurada. Desde el punto de vista informático, la base de datos está formada por un conjunto de datos almacenados en medios de almacenamientos y que se permite el acceso a ellos a través de un conjunto de programas que manipulen ese conjunto de datos. Cada base de datos se compone de una o más tablas que guarda un conjunto de datos. Cada tabla tiene una o más columnas y filas (cada fila de la tabla conforma un registro).

Figura N° 4: Base de datos

Fuente: Pressman

(Elmasri & Shamkant 2011) Una base de datos es una colección de datos relacionados. Con la palabra datos nos referimos a los hechos (datos) conocidos que se pueden grabar y que tienen un significado implícito.

Esto último se sostiene el concepto, donde se expresa que una base de datos es un conjunto exhaustivo no redundante de datos estructurados organizados independientemente de su utilización y su implementación en máquinas accesibles en tiempo real y compatible con usuarios concurrentes con necesidad de información diferente y no predicable en tiempo.

(Casillas 2005) Indica que en el diseño de la Base de Datos conviene descomponer el proceso del diseño en varias etapas; en cada una se obtiene un resultado intermedio que sirve de punto de partida de la etapa siguiente: “la etapa de diseño conceptual nos permite concentrarnos únicamente en la problemática de la estructuración de la

información, sin tener que preocuparnos al mismo tiempo de resolver cuestiones tecnológicas. El resultado de la etapa del diseño conceptual se expresa mediante algún modelo de datos de alto nivel, uno de los más empleados es el modelo entidad - interrelación".

También refiere que la etapa del diseño lógico lo define como: "parte de la etapa del diseño conceptual, que se transforma de forma que se adapte a la tecnología que se debe emplear, es preciso que se ajuste al modelo del SGBD con el que se desea implementar la BD. Esta etapa obtendrá un conjunto de relaciones con sus atributos, claves primarias y claves foráneas". La etapa del diseño físico menciona que: "es donde se transforma la estructura obtenida en la etapa del diseño lógico, con el objetivo de conseguir una mayor eficiencia, además se completa con aspectos de implementación física que dependerán SGBD. Los aspectos de implementación física que hay que completar consisten normalmente en la elección de las estructuras físicas de implementación de las relaciones".

2.2.13.1. Estructura de una Base de Datos.

(Aguilar 2011) Las bases de datos poseen una estructura según donde existen:

- **Independencia de datos y tratamiento:** Se entiende que el cambio de los datos no implica cambio de los programas y viceversa dando un menor costo en operaciones de mantenimiento.

- **Coherencia de resultados:** Aquí se logran reducir la redundancia la cual es evaluada por medio de acciones lógicamente únicas y se evita la inconsistencia.

- **Disponibilidad de datos:** Se llega a mejorar la disponibilidad de datos debido a que no hay un dueño necesario de los datos y al guardado de las descripciones.

- **Restricciones:** Se cumplen algunas normas tales como las restricciones de seguridad para evitar el acceso a usuarios no autorizados y prevenir operaciones no deseadas o no programadas.

2.2.13.2. Arquitectura de la Base de Datos

Según Elmasri & Shamkant existen hasta tres niveles en la arquitectura de una base de datos según refiere, siendo los siguientes:

- **Nivel físico:** Este nivel tiene un esquema interno, que describe la estructura de almacenamiento físico de la base de datos. El esquema interno utiliza un modelo de datos físico y describe todos los detalles del almacenamiento de datos y las rutas de acceso a la base de datos.

- **Nivel conceptual:** Este nivel tiene un esquema conceptual, que describe la estructura de toda la base de datos para una comunidad de usuario. El esquema conceptual oculta los detalles de las estructuras de almacenamiento físico y se concentra en describir las entidades, los tipos de datos, las relaciones, las operaciones de

los usuarios y las restricciones. Normalmente, el esquema conceptual se describe en un modelo de datos representativo cuando se implementa un sistema de base de datos.

- **Nivel de vista:** Nivel de vista o extremo incluye una cierta cantidad de esquemas externos o vistas de usuario. Un esquema externo describe la parte de la base de datos en la que un grupo de usuarios en particular está interesado y le oculta el resto de la base de datos.

2.2.14. Sistema de Gestión de la Base de Datos

(Silberschatz, Korth&Sudarshanm) Se conceptualiza que los sistemas de base de datos se diseñan para gestionar grandes cantidades de información. Esto implica la definición de estructuras para almacenar la información como la provisión de mecanismos para la manipulación de la información. Además, los sistemas de base de datos deben proporcionar la fiabilidad de la información almacenada, a pesar de las caídas del sistema o los intentos de acceso sin autorización.

Sistema de gestión de base de datos es aquel que permite el almacenamiento, manipulación y consulta de datos organizada en uno o varios ficheros. En el modelo más extendido (base de datos relacional) la base de datos consiste, de cara al usuario, en un conjunto de tablas, entre las que se establecen relaciones.

Objetivos de la SGBD.

(Gutiérrez 2010) Existen varios objetivos que deben cumplir como explica, en los Sistemas de Gestión de Base de Datos, definidos en la presentación de Gustavo Alfonso con maestra en ciencias en México en el 2010, del cual detallaremos las principales:

- **Abstracción de la información:** Los SGBD ahorran a los usuarios detalles acerca del almacenamiento físico de los datos. Da lo mismo si una base de datos ocupa uno o cientos de archivos, este hecho se hace transparente al usuario.

- **Independencia:** La independencia de la base de datos consiste en la capacidad de modificar el esquema (físico o lógico) de una base de datos sin tener que realizar cambios en las aplicaciones que se sirven de ella.

- **Consistencia:** En aquellos casos en los que no se ha logrado eliminar la redundancia, será necesario vigilar que aquella información que aparece repetida se actualice de forma coherente, es decir, que todos los datos repetidos se actualicen de forma simultánea. El sistema no debería aceptar datos de un conductor menor de edad. En los SGBD existen herramientas que facilitan la programación de este tipo de condiciones.

- **Seguridad:** La información almacenada en una base de datos puede llegar a temer un gran valor. Los SGBD deben garantizar que esta información se encuentra segura de permisos a usuarios

y grupo de usuarios, que permiten otorgar diversas categorías de permisos.

- **Manejo de transacciones:** Una transacción es una ejecución de una sola operación. Los SGBD proveen mecanismos para programar las modificaciones de las bases de datos de una forma mucho más simple que si no se dispusiera de ellos.

- **Tiempo en respuesta:** Lógicamente, es deseable minimizar el tiempo que el SGBD demora en proporcionar la información solicitada y en almacenar los cambios realizados.

2.2.15. Programación Web

La programación Web, parte de las siglas WWW, que significan World Wide Web. Para realizar una página con la programación Web, se deben tener claros, tres conceptos fundamentales los cuales son, el URL (Uniform Resource Locators), es un sistema con el cual se localiza un recurso dentro de la red, este recurso puede ser una página web, un servicio o cualquier otra cosa. En resumen el URL no es más que un nombre, que identifica una computadora, dentro de esa computadora un archivo que indica el camino al recurso que se solicita.

2.2.15.1. HTML

HyperText Markup Language (lenguaje de marcas de hipertexto), hace referencia al lenguaje de marcado para la elaboración de páginas web. Es un estándar que sirve de referencia del software que conecta con la elaboración de páginas web en sus diferentes versiones, define una

estructura básica y un código (denominado código HTML) para la definición de contenido de una página web, como texto, imágenes, videos, juegos, entre otros. Es un estándar a cargo del Consorcio WWW, organización dedicada a la estandarización de casi todas las tecnologías ligadas a la web, sobre todo en lo referente a su escritura e interpretación. Se considera el lenguaje web más importante siendo su invención crucial en la aparición, desarrollo y expansión de la WWW. Es el estándar que se ha impuesto en la visualización de páginas web y es el que todos los navegadores actuales han adoptado.

2.2.15.2. PHP

Es un lenguaje de programación de propósito general de código del lado del servidor originalmente diseñado para el desarrollo web de contenido dinámico. Fue uno de los primeros lenguajes de programación del lado del servidor que se podían incorporar directamente en el documento HTML en lugar de llamar a un archivo externo que procese los datos. El código es interpretado por un servidor web con un módulo de procesador de PHP que genera la página web resultante. PHP ha evolucionado por lo que ahora incluye también una interfaz de línea de comandos que puede ser usada en aplicaciones gráficas independientes. Puede ser usado en la mayoría de los servidores web al igual que en casi todos los sistemas operativos y plataformas sin ningún costo.

El objetivo de PHP es brindarles a los creadores de sitios web la posibilidad de desarrollar sitios dinámicos en forma sencilla y rápida, aunque en verdad veremos que las posibilidades y funcionalidades de

PHP son muy superiores al simple hecho de solo hacer una página web dinámica.

2.2.15.3. MySQL

MySQL es un sistema de gestión de base de datos racional desarrollado bajo una licencia dual comercial GPL/Licencia, lo que la considera una de las la base datos open source más popular del mundo; MySQL es muy utilizado en aplicaciones web. Las innovaciones de MySQL están fundamentadas en que este sistema es el único que permite escoger entre múltiples motores de almacenamiento para cada tabla, además, MySQL te permite realizar agrupaciones de transacciones, reuniendo múltiples transacciones de varias conexiones para el incremento del número de transacciones por segundo.

MySQL como sistema de administración de bases de datos permite que esta base pueda ser desde una simple lista de compras a una galería de pinturas, o el vasto volumen de información en una red corporativa; además, la base de datos relacional archiva datos en tablas separadas en vez de colocar todos los datos en un gran archivo, lo que permite velocidad y flexibilidad.

2.2.15.4. JAVASCRIPT

Javascript es un lenguaje con muchas posibilidades, utilizado para crear pequeños programas que luego son insertados en una página web y en programas más grandes, orientados a objetos mucho más complejos.

Con Javascript podemos crear diferentes efectos e interactuar con nuestros usuarios.

Este lenguaje posee varias características, entre ellas podemos mencionar que es un lenguaje basado en acciones que posee menos restricciones. Además, es un lenguaje que utiliza Windows y sistemas X-Windows, gran parte de la programación en este lenguaje está centrada en describir objetos, escribir funciones que respondan a movimientos del mouse, aperturas, utilización de teclas, cargas de páginas entre otros.

La ventaja de JavaScript es que al estar alojado en el ordenador del usuario los efectos son muy rápidos y dinámicos. Al ser un lenguaje de programación permite toda la potencia de la programación como uso de variables, condicionales, bucles, etc. También podemos citar algún inconveniente: por ejemplo si el usuario tiene desactivado JavaScript en su navegador, no se mostrarán los efectos. No obstante, hoy día la mayoría de los usuarios navegan por la web con JavaScript activado.

2.2.15.5. AJAX

AJAX, acrónimo de Asynchronous JavaScript And XML (JavaScript asíncrono y XML), es una técnica de desarrollo web para crear aplicaciones interactivas o RIA (Rich Internet Applications). Estas aplicaciones se ejecutan en el cliente, es decir, en el navegador de los usuarios mientras se mantiene la comunicación asíncrona con el servidor en segundo plano. De esta forma es posible realizar cambios sobre las páginas sin necesidad de recargarlas, mejorando la interactividad, velocidad y usabilidad en las aplicaciones.

AJAX es una tecnología asíncrona, en el sentido de que los datos adicionales se solicitan al servidor y se cargan en segundo plano sin interferir con la visualización ni el comportamiento de la página, aunque existe la posibilidad de configurar las peticiones como síncronas de tal forma que la interactividad de la página se detiene hasta la espera de la respuesta por parte del servidor.

El principal objetivo de AJAX, es intercambiar información entre el servidor y el cliente (navegadores) sin la necesidad de recargar la página. De esta forma, ganamos en usabilidad, experiencia y productividad del usuario final.

2.2.15.6. JQUERY

Esta es la librería web más popular disponible en estos días. La librería jQuery es gratuita y fue diseñada para simplificar la creación de sitios web modernos. Facilita la selección de elementos HTML, la creación de animaciones y efectos, también controla eventos y ayuda a implementar Ajax en nuestras aplicaciones.

La librería jQuery se encuentra en un archivo pequeño que se puede descargar desde **www.jquery.com** y luego incluir en nuestros documentos usando la etiqueta **<script>**. Provee un API sencilla que cualquiera puede aprender y rápidamente aplicar a sus proyectos. Una vez que el archivo provisto por jQuery es incluido en nuestro documento, ya estamos listos para aprovechar los métodos simples incorporados por la librería y convertir nuestra web estática en una moderna y practica aplicación.

JQuery tiene la ventaja de proveer soporte para viejos navegadores y vuelve simple tareas cotidianas. Puede ser utilizado junto con HTML5 o como una forma simple de reemplazar funciones de HTML5 en navegadores que no están preparados para esta tecnología. (Gauchat,J. 2012).

2.2.15.7. BOOTSTRAP

Twitter Bootstrap es un framework o conjunto de herramientas de software libre para diseño de sitios y aplicaciones web. Contiene plantillas de diseño con tipografía, formularios, botones, cuadros, menus de navegación y otros elementos de diseño basado en HTML y CSS, asi como, extensiones de Java Script opcionales adicionales.

Bootstrap fue desarrollado por Mark Otto y Jacbod Thornton de Twiterr, como un marco de trabajo (framework) para fomentar la consistencia a través de herramientas internas. Antes de Bootstrap, se usaba varias librerías para el desarrollo de interfaces de usuario, las cuales guiaban a inconsistencias y a una carga de trabajo alta en su mantenimiento.

Según el desarrollador de Twitter Mark Otto, frente a esos desafíos: "...un pequeño grupo de desarrolladores nos reunimos a diseñar y construir una nueva herramienta interna y vimos una oportunidad de hacer más. A través de ese proceso, nos vimos construyendo algo mucho más sustancial que otra herramienta interna. Meses después terminamos con una primera versión de Bootstrap como una manera de documentar y compartir bienes y patrones de diseño comunes dentro de la compañía."

El primer desarrollo en condiciones reales ocurrió durante la primera "Semana de Hackeo" (Hackweek) de Twitter." Mark Otto mostró a algunos colegas como acelerar el desarrollo de sus proyectos con la ayuda de la herramienta de trabajo. Como resultado, decenas de temas se han introducido en el marco de trabajo.

2.2.15.8. CSS

La sigla CSS corresponde a la expresión inglesa Cascading Style Sheets, que puede traducirse como “Hojas de estilo en cascada”. El concepto se utiliza en el ámbito de la informática para referirse a un lenguaje empleado en el diseño gráfico. El lenguaje CSS permite presentar, de manera estructurada, un documento que fue escrito en un lenguaje de marcado. Se usa especialmente en el diseño visual de un sitio web cuando las páginas se hallan escritas en XML o HTML. El CSS se desarrolló en distintos niveles.

El CCS1 ya no se emplea, mientras que el CSS2 funciona como recomendación. El CSS3, que se divide en varios módulos, es el lenguaje que se está tomando como estándar. Lo que hace el CSS es encargarse de la descripción de las formas y de la sintaxis del lenguaje de marcado. De esta manera describe cómo se tienen que renderizar los elementos que aparecen en pantalla.

2.2.15.9. FRAMEWORK

La palabra inglesa "framework" (infraestructura, armazón, marco) define, en términos generales, un conjunto estandarizado de conceptos,

prácticas y criterios para enfocar un tipo de problemática particular que sirve como referencia, para enfrentar y resolver nuevos problemas de índole similar. En el desarrollo de software, un framework o infraestructura digital, es una estructura conceptual y tecnológica de soporte definido, normalmente con artefactos o módulos concretos de software, que puede servir de base para la organización y desarrollo de software. Típicamente, puede incluir soporte de programas, bibliotecas, y un lenguaje interpretado, entre otras herramientas, para así ayudar a desarrollar y unir los diferentes componentes de un proyecto. Representa una arquitectura de software que modela las relaciones generales de las entidades del dominio, y provee una estructura y una especial metodología de trabajo, la cual extiende o utiliza las aplicaciones del dominio.

2.2.15.10. SUBLIME TEXT

Es un editor de código multiplataforma, ligero y con pocas concesiones a las florituras. Es una herramienta concebida para programar sin distracciones. Su interfaz de color oscuro y la riqueza de coloreado de la sintaxis, centra nuestra atención completamente. Permite tener varios documentos abiertos mediante pestañas, e incluso emplear varios paneles para aquellos que utilicen más de un monitor. Dispone de modo de pantalla completa, para aprovechar al máximo el espacio visual disponible de la pantalla.

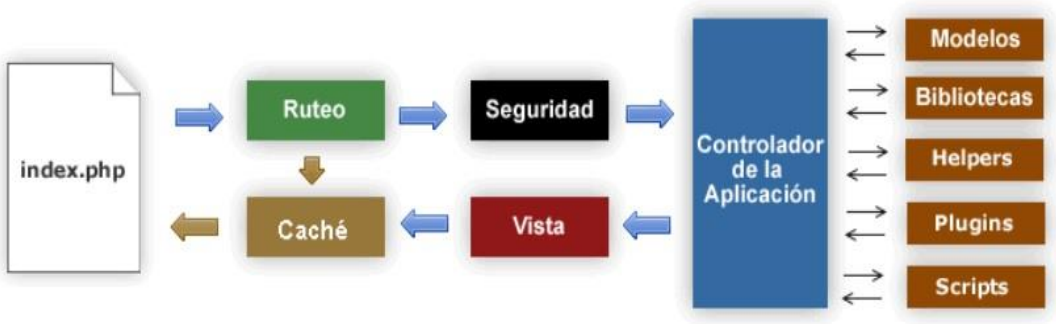
2.2.15.11. CODEIGNITER

CodeIgniter es un conjunto de herramientas para personas que construyen su aplicación web usando PHP. Su objetivo es permitir

desarrollar proyectos mucho más rápido de lo que podría si lo escribiese desde cero, proveyendole un rico juego de librerías para tareas comunmente necesarias, así como una interface simple y estructura lógica para acceder a esas librerías. También permite creativamente enfocarse en su proyecto minimizando la cantidad de código necesaria para una tarea dada. CodeIgniter se encuentra bajo una licencia open source Apache/BSD-style, así que lo puede usar donde le parezca. Para más información por favor lea el acuerdo de licencia.

CodeIgniter usa el acercamiento Modelo-Vista-Controlador, que permite una buena separación entre lógica y presentación. Esto es particularmente bueno para proyectos, en los cuales, diseñadores están trabajando con sus archivos de plantilla, ya que el código en esos archivos será mínimo. Describimos MVC en más detalle en su propia página.

Figura N° 5: Diagrama de flujo de la aplicación



Fuente: CodeIgniter Guía Usuario

Modelo – Vista – Controlador

CodeIgniter está basado en el patrón de desarrollo Modelo-Vista-Controlador. MVC es una aproximación al software que separa la lógica de la aplicación de la presentación. En la práctica, permite que sus

páginas web contengan mínima codificación ya que la presentación es separada del código PHP.

- **Modelo:** Representa la estructura de datos. Típicamente sus clases de modelo contendrán funciones que lo ayudarán a recuperar, insertar y actualizar información en su base de datos.

- **Vista:** Es la información que es presentada al usuario. La Vista normalmente será una página web, pero en CodeIgniter, una vista también puede ser un fragmento de una página como un encabezado o un píe de página. También puede ser una página RSS, o cualquier otro tipo de "página".

- **Controlador:** Sirve como un intermediario entre el Modelo, la Vista y cualquier otro recurso necesario para procesar la petición HTTP y generar una página web.

CodeIgniter tiene un enfoque bastante flexible del MVC, ya que los Modelos no son requeridos. Si no necesita agregar separación, o descubre que mantener los modelos requiera más complejidad que quería, puede ignorarlos y construir su aplicación mínimamente usando Controladores y Vista.

2.3. Definición de Términos Básicos

Arquitectura: Una arquitectura es un entramado de componentes funcionales, aprovechando diferentes estándares, convenciones, reglas y procesos, permite integrar una amplia gama de productos y servicios

informáticos, de manera que puedan ser utilizadas eficazmente dentro de la organización.

Automatización: Es la realización de una tarea del hombre con la ayuda de una computadora diseñada y acondicionada para una tarea especificada.

Administrador: Es la persona o equipo de personas profesionales responsables del control y manejo del sistema de base de datos, generalmente tienen experiencia en DBMS, diseño de bases de datos, sistemas operativos, comunicación de datos, hardware y programación.

Proceso: un conjunto de actividades, material y/o flujo de información que transforma un conjunto de entradas en resultados definidos.

Sistema: Es el conjunto de elementos relacionados entre sí con un propósito determinado (único). Es un conjunto u ordenación de elementos organizados para llevar a cabo algún método, procedimiento o control mediante el procesamiento de información. En el sentido más amplio, un sistema es simplemente un conjunto de componentes que interactúan para alcanzar algún objetivo.

Sistema de información: Es un conjunto de procedimientos organizados que, cuando se ejecutan, proporcionan información para la toma de decisiones y/o el control de la organización. Es un conjunto de personas, datos y procedimientos que funcionan en conjunto; el énfasis en sistemas significa que los variados componentes buscan un objetivo común para apoyar las actividades de la organización.

Lenguaje de programación: Es cualquier lenguaje según el cual se expresan o se escriben las instrucciones (programas) de procesamiento de la computadora. Lenguajes en el cual se escriben las instrucciones que controlan el movimiento y procesado de datos.

Programación: Conjunto de instrucciones escritas en un lenguaje particular, que representa la resolución del problema. En otros términos se puede decir que un "programa" es la elaboración de un algoritmo efectuado en lenguaje para ordenador.

Tiempo: Sucesión de fenómenos que va irreversiblemente del pasado al futuro.

Interfaz del Sistema: Son las especificaciones funcionales del sistema, los cuales son representados mediante pantallas y/o menús, que permitirán al usuario interactuar con el sistema.

Sistema: Conjunto de componentes interrelacionados e interactuantes para llevar a cabo una misión conjunta. Admite ciertos elementos de entrada y produce ciertos elementos de salida en un proceso organizado.

Página web: Es un documento electrónico que contiene información específica de un tema en particular y que es almacenado en algún sistema de cómputo que se encuentre conectado a la red mundial de información denominada Internet, de tal forma que este documento pueda ser consultado por cualquiera persona que se conecte a esta red mundial de comunicaciones y que cuente con los permisos apropiados para hacerlo.

CAPÍTULO III

MATERIALES Y MÉTODOS

3.1. Ubicación Geográfica del Estudio

El desarrollo del presente trabajo se llevó a cabo en el Vicerrectorado de Investigación de la Universidad Nacional del Altiplano Puno.

3.2. Población y Muestra del Estudio

3.2.1. Población

Nuestra población para efectos de investigación está conformada por todas las personas relacionadas con los laboratorios de las diferentes escuelas profesionales.

3.2.2. Muestra

La técnica de muestro que se utilizó para el presente trabajo de investigación fue el Muestro Aleatorio Simple (MAS). Para lo cual aplicamos la siguiente formula.

$$n_0 = \frac{Z^2 PQ}{E^2}$$

Donde:

P = 0.5 (Probabilidad de éxito).

Q = 0.5 (Probabilidad de fracaso).

E = 0.05 (Margen de error).

Z = 95% = 1.96 (Nivel de confianza).

N = 105 (Tamaño de la población).

$$n_0 = \frac{(1.96)^2(0.5)(0.5)}{(0.05)^2}$$

$$n_0 = 384.16 \approx 384$$

El tamaño de muestra se considera n_0 siempre que la fracción de muestreo sea menor o igual del 5% es decir si $f \leq 0.05$.

$$f = \frac{n_0}{N} = \frac{384}{105} = 3.66$$

El tamaño de muestra final será:

$$n = \frac{n_0}{1 + \frac{n_0 - 1}{N}} = \frac{384}{1 + \frac{384 - 1}{105}}$$

$$\boldsymbol{n = 82.63 \approx 83}$$

Realizando el redondeo al entero inmediato se obtuvo n = 83 jefes de laboratorios y laboratoristas, el cual fue el tamaño de muestra para el trabajo de investigación.

3.3. Operacionalización de Variables

Tabla N° 5: Operacionalización de Variables

VARIABLE	DIMENSION	INDICADORES	ESCALA
Eficacia del Sistema de Información.	Utilidad del Sistema (ISO)	1. Mejora en el flujo del trabajo. 2. Reducción del tiempo de proceso. 3. Calidad de interfaces. 4. Reducción de errores. 5. Disponibilidad del sistema. 6. Presenta la información necesaria. 7. Prevención de errores. 8. Velocidad del sistema.	• Inaceptable • Mínimamente aceptable • Aceptable • Cumple los requisitos • Excede los requisitos
	Satisfacción de Usuario (ENCUESTA)	1. Navegación del sistema. 2. Acceso al sistema. 3. Seguridad del sistema. 4. Precisión de la información. 5. Consistencia de los datos de ingreso. 6. Calidad de soporte. 7. Tiempo de respuesta de la plataforma. 8. Pantallas de ayuda.	• Mala • Regular • Bueno

Fuente: Elaboración Propia

3.4. Recopilación de la Información

Para la recolección de la información de requerimientos del sistema se realizó mediante entrevistas y encuestas a los jefes de laboratorio y laboratoristas que hacen uso del sistema de información.

Para la recopilación de datos de usabilidad del sistema se utilizó los instrumentos que fueron aplicados en los usuarios del sistema de información en la Universidad Nacional del Altiplano, como:

- La ficha de evaluación del ISO-9126 que ve la calidad de software.
- Encuesta de satisfacción del usuario.

3.5. Arquitectura del Software

La arquitectura de software se refiere a la estructuración del sistema que, idealmente, se crea en etapas tempranas del desarrollo. Esta estructuración representa un diseño de alto nivel del sistema que tiene dos propósitos primarios: satisfacer los atributos de calidad y servir como guía en el desarrollo. Al igual que en la ingeniería civil, las decisiones críticas relativas al diseño general de un sistema de software complejo deben de hacerse desde un principio. El no crear este diseño desde etapas tempranas del desarrollo puede limitar severamente el que el producto final satisfaga las necesidades de los clientes.

3.5.1. Ciclo de Desarrollo de la Arquitectura

Dentro de un proyecto de desarrollo, e independientemente de la metodología que se utilice, se puede hablar de “desarrollo de la

arquitectura de software". Este desarrollo, que precede a la construcción del sistema, está dividido en las siguientes etapas: requerimientos, diseño, documentación y evaluación.

A continuación, se describen dichas etapas.

Requerimientos: La etapa de requerimientos se enfoca en la captura, documentación y priorización de requerimientos que influencian la arquitectura. Como se mencionó anteriormente, los atributos de calidad juegan un papel preponderante dentro de estos requerimientos, así que esta etapa hace énfasis en ellos.

Diseño: La etapa de diseño es la etapa central en relación con la arquitectura y probablemente la más compleja. Durante esta etapa se definen las estructuras que componen la arquitectura. La creación de estas estructuras se hace en base a patrones de diseño, tácticas de diseño y elecciones tecnológicas.

Documentación: Una vez creado el diseño de la arquitectura, es necesario poder comunicarlo a otros involucrados dentro del desarrollo. La comunicación exitosa del diseño muchas veces depende de que dicho diseño sea documentado de forma apropiada. La documentación de una arquitectura involucra la representación de varias de sus estructuras que son representadas a través de distintas vistas.

Evaluación: Dado que la arquitectura de software juega un papel crítico en el desarrollo, es conveniente evaluar el diseño una vez que este ha sido documentado con el fin de identificar posibles problemas y riesgos.

La ventaja de evaluar el diseño es que es una actividad que se puede realizar de manera temprana (aún antes de codificar), y que el costo de corrección de los defectos identificados a través de la evaluación es mucho menor al costo que tendría el corregir estos defectos una vez que el sistema ha sido construido.

3.6. Modelo y Metodología de Desarrollo de Software

El propósito no es de escoger una metodología porque sea mejor, pues el empleo de una u otra es de acuerdo al tipo de proyecto, a los recursos que serán utilizados y a la facilidad de interacción con el usuario.

Tabla N° 6: Diferencias entre Metodologías Tradicionales y Ágiles

Metodologías Tradicionales	Metodologías Agiles
Basada en normas provenientes de estándares seguidos por el entorno de desarrollo.	Basadas en heurísticas provenientes de prácticas de producción de código.
Cierta resistencia a los cambios.	Especialmente preparados para cambios durante el proyecto.
Impuestas externamente.	Impuestas internamente (por el equipo de desarrollo)
Proceso mucho más controlado, con numerosas políticas/normas.	Proceso menos controlado, con pocos principios.
Existe un contrato prefijado.	No existe contrato tradicional o al menos es bastante flexible.
El cliente interactúa con el equipo de desarrollo mediante reuniones.	El cliente es parte del equipo de desarrollo.
Grupos grandes y posibilidades distribuidos.	Grupos pequeños (<10 integrantes) y trabajando en el mismo sitio.
Más artefactos.	Pocos artefactos.
Más roles.	Pocos roles.
La arquitectura del software es esencial y se expresa mediante modelos	Menos énfasis en la arquitectura del software.

Fuente: https://arevalomaria.wordpress.com/2011/11/15/

La investigación se centra en el desarrollo de un sistema de información y los recursos que priman son los relacionados a los factores tiempo y costo. En cuanto a los costos lo deseable es que sea lo más reducido posible y, en relación al tiempo, se desea que la mayor inversión de éste sea para la construcción del sistema.

3.7. Diseño del Sistema

Para el diseño de la aplicación el equipo de trabajo siguió las recomendaciones de la metodología SCRUM, tratando de evitar las soluciones complejas, y se trabajó en varias iteraciones, previendo las modificaciones futuras para la optimización de la plataforma. Otro aspecto importante en el diseño, fue la constante reestructuración del código. El principal objetivo de la reestructuración fue evitar la duplicación del código poniendo en práctica la normalización de datos simplificarlo y hacerlo más flexible para facilitar los posteriores cambios. Esto se hizo constantemente en la programación de la aplicación.

Ya que también el modelo de desarrollo SCRUM es organizado y a la vez es riguroso. Además este es utilizado como base en otros modelos, este se define por su enfoque metodológico que ordena rigurosamente en los SPRINTS, de forma tal que el inicio de una nueva tarea se pueden ir desarrollando otras tareas en simultaneo (Sprint Planning Meeting), sin descuidar una de la otra de esta forma cualquier error de diseño detectado se puede corregir inmediatamente antes de finalizar la tarea (Scrum Daily Meeting) de esta manera controlando todo el proceso con revisiones al finalizar cada tarea encomendada (Sprint Review), dándole la seguridad,

optimización y calidad de desarrollo observando cada detalle de las etapas de desarrollo y dando la oportunidad reiniciar el ciclo para mejorar la calidad del producto (Sprint Retrospective) así como también las historias de usuario y el tablero de kanban.

3.8. Requerimientos del Sistema

3.8.1. Requerimientos Funcionales

- Debe existir una validación de datos para el acceso al sistema, este le dará mayor seguridad a la información.
- El sistema debe estar en la capacidad de registrar nuevos usuarios, laboratorios y equipos.
- El sistema debe estar en la capacidad de poder editar datos como de los usuarios, laboratorios y equipos en caso hubiese errores.
- Por necesidad debe existir la posibilidad de consultar datos.

3.8.2. Requerimientos no Funcionales

- Interfaz amigable para un fácil entendimiento del software.
- Disponibilidad del sistema todos los días.
- Estabilidad del sistema (soporta varios usuarios a la vez).
- Rendimiento del sistema que brindara un servicio óptimo ya que es diseñado para que funcione en un ambiente web.

CAPÍTULO IV

RESULTADOS Y DISCUSIÓN

4.1. Análisis y Diseño

4.1.1. Descripción

Para el adecuado desarrollo del software del sistema, se presentó una secuencia de procedimientos, que permitió llevar en forma ordenada para un mejor entendimiento de las diversas etapas, de la ejecución del presente trabajo de investigación, bajo las especificaciones de Ingeniería de Software, y con la ayuda del Lenguaje Unificado de Modelado (UML).

4.1.2. Análisis de Requisitos del Sistema

El desarrollo del software no consiste únicamente en la codificación, aunque sea una de las fases más importantes, sino que existen múltiples tareas que deben ser desarrolladas si se desea un sistema robusto y de fácil mantenimiento y con garantías de finalización en el tiempo y costo estimados.

Para determinar las tareas es necesario asegurar un proceso de desarrollo de software, adaptándolo según se estime conveniente al proyecto en el cual se aplique. Una de estas tareas es permitir conocer en todo momento el estado en el que se encuentra el sistema durante el desarrollo.

4.1.3. Análisis de Roles

Hay que tener en cuenta que los desarrolladores del proyecto eran dos personas por lo que los roles definidos en Scrum fueron ocupados por los ejecutores, en algunos casos por el asesor y director de proyecto.

- Programador: El código fuente fue hecho por los ejecutores del proyecto de investigación.
- Arquitectura de Software: Mi asesor y director de tesis, quien apoyo en el modelado de la Plataforma.
- Cliente: El desarrollador del proyecto y los encargados de administrar el sistema fueron los que hicieron las pruebas funcionales para validar su implementación.
- Encargado de pruebas (Tester): El personal del Vicerrectorado de Investigación quien ayudó a escribir las pruebas funcionales. Ejecutó las pruebas regularmente, e informó los resultados y apreciaciones al desarrollador.

- Encargado de seguimientos (Tracker): El personal del Vicerrectorado de investigación quien proporcionó realimentación al equipo y realizó el seguimiento del progreso de cada iteración.

- Entrenador (Coach): Los ejecutores somos los responsables del proceso global, también de realizar las guías para las prácticas Scrum además responsable de que se siguiera el proceso correctamente.

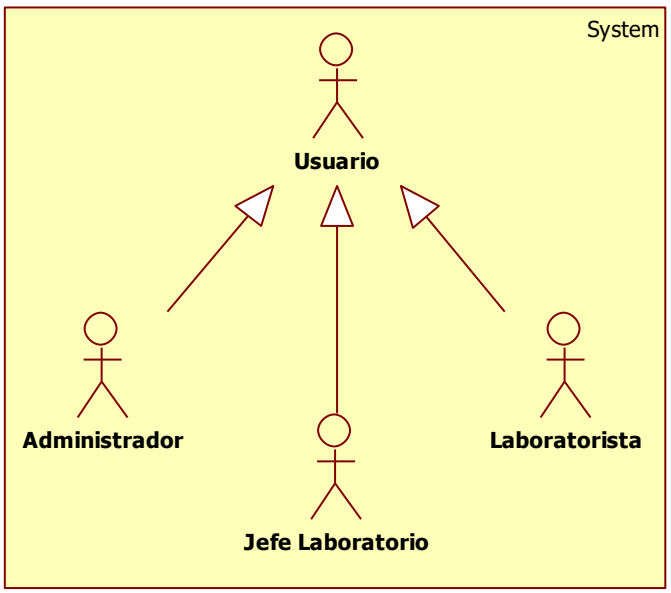
4.2. Modelamiento de la Plataforma Mediante UML

Para entender el modelamiento en el diseño de la plataforma se ha utilizado el lenguaje unificado de Modelado UML, la misma que nos ha permitido obtener los diagramas de casos de uso y secuencia.

4.3. Modelado de Requisitos

4.3.1. Actores del Sistema

Figura N° 6: Actores del sistema



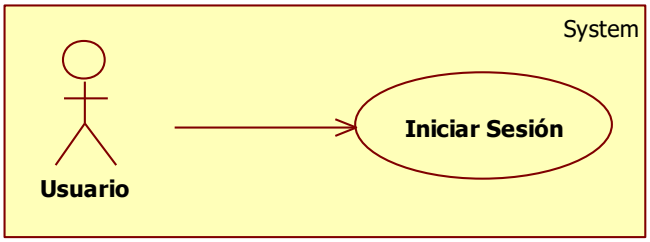
Fuente: Elaboración propia

En la figura N° 6 se puede observar el personal designado como usuario.

4.3.2. Modelo de Casos de Uso

4.3.2.1. Caso de uso: Inicio de Sesión

Figura N° 7: Diagrama de casos de uso Inicio de Sesión



Fuente: Elaboración propia

En la figura N° 7 se muestra el primer privilegio que posee el usuario dentro del sistema.

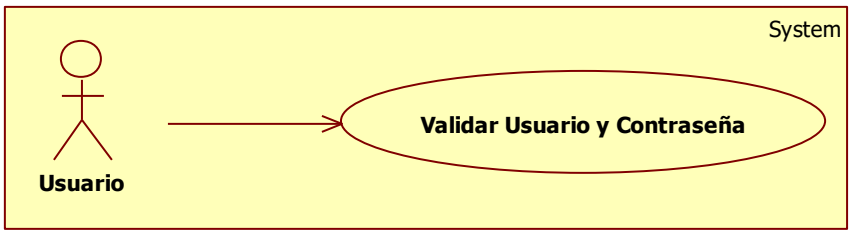
Tabla N° 7: Descripción del caso de uso Inicio de Sesión

Nombre	Inicio de Sesión
Actor	Usuario
Propósito	Ingresar al sistema de laboratorio
Pre-Condición	Acceder a la página principal del sistema web mediante URL.
Flujo Normal de los Eventos	
Actor	**Sistema**
1. Ingresar a la página principal. 2. Ingresar usuario y contraseña.	3. Validar usuario y contraseña. 4. Mostrar página Personal.
Post Condición	Ingresar a opciones del sistema.
Excepciones	Mostrar mensaje de error por falla de Inicio de Sesión.

Fuente: Elaboración propia

4.3.2.2. Caso de uso: Validar Usuario y Contraseña

Figura N° 8: Diagrama de casos de uso validar usuario y contraseña



Fuente: Elaboración propia

En la figura N° 8 se muestra el privilegio que posee el usuario (administrador, jefe de laboratorio y laboratorista) dentro del sistema para la validación del usuario y contraseña.

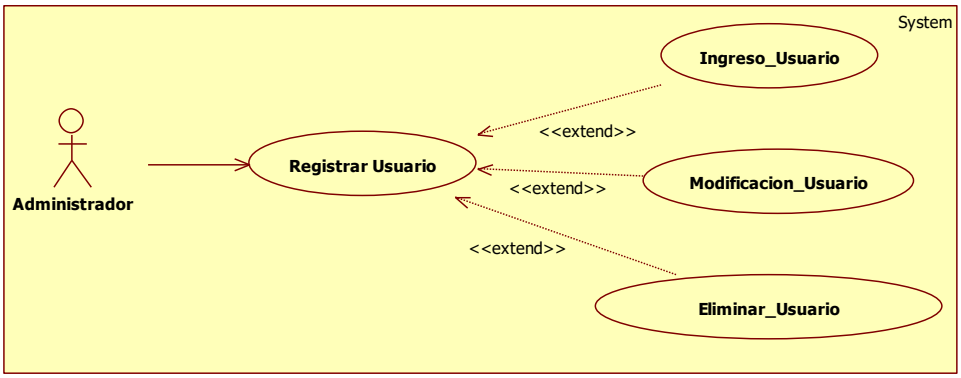
Tabla N° 8: Descripción de caso de uso validar usuario y contraseña

Nombre	Validar usuario y contraseña
Actor	Usuario
Propósito	Validar existencia de relación usuario y contraseña
Pre-Condición	1. Acceder a Página de inicio de sesión del sistema mediante URL. 2. Ingresar Usuario y contraseña.
Flujo Normal de los Eventos	
Sistema	
Validar relación usuario y contraseña.	
Post Condición	Ingresar a opciones del sistema.
Excepciones	Mostrar mensaje de error usuario y contraseña.

Fuente: Elaboración propia

4.3.2.3. Caso de uso: Registrar Usuario

Figura N° 9: Diagrama de Casos de uso Registrar Usuario



Fuente: Elaboración propia

En la figura N° 09 se muestra como el administrador realiza las acciones de registrar, editar, eliminar y buscar un usuario.

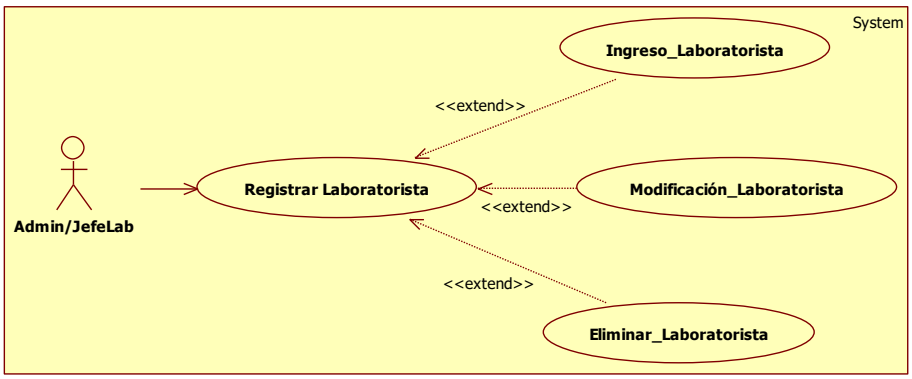
Tabla N° 9: Descripción de Caso de uso Registrar Usuario

Nombre	Registrar, editar y eliminar usuario.
Actor	Usuario (Administrador)
Propósito	El administrador ingresa, modifica y elimina usuarios de inicio de sesión.
Pre-Condición	Tener DNI del usuario
Flujo normal de los Eventos	
Actor	**Sistema**
1. Acceder al menú Administrador. 2. Seleccionar opción “Registrar / Editar / Eliminar usuario”. 3. Ingresar datos personales, usuario y contraseña en el formulario que muestra el sistema para su registro. 4. Ingresar el número de DNI para modificar y eliminar un usuario.	5. Validar datos para registrar, editar y eliminar usuarios. 6. Registrar, editar y eliminar datos.
Post Condición	Ingresar a otras opciones del menú usuario.
Excepciones	El sistema mostrara mensaje de error si no se registró, edito y elimino correctamente, o algún error que impida el proceso.

Fuente: Elaboración Propia

4.3.2.4. Caso de Uso: Registrar Laboratorista

Figura N° 10: Diagrama de Caso de uso Registrar Laboratorista



Fuente: Elaboración propia

En la figura N° 10 se muestra como los usuarios (Administrador y Jefe de laboratorio) realizan las acciones de registrar, editar, eliminar y buscar un laboratorista.

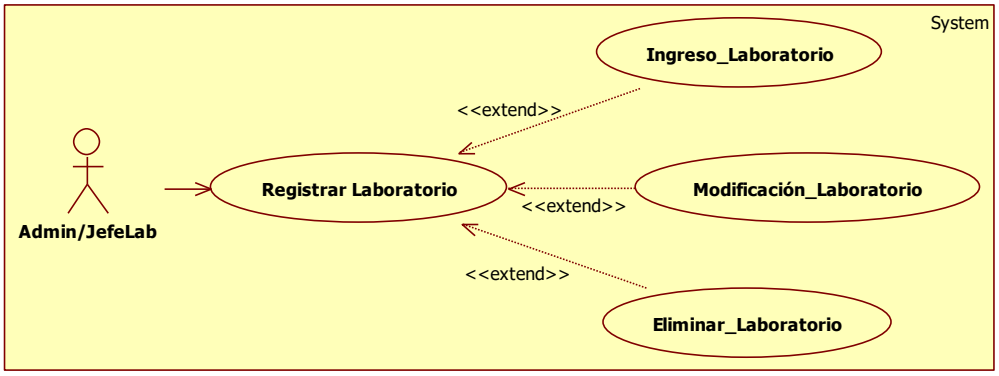
Tabla N° 10: Descripción de Caso de uso Jefe de Laboratorio

Nombre	Registrar, editar y eliminar usuario.
Actor	Usuario (Administrador y Jefe de laboratorio)
Propósito	El Administrador y Jefe de laboratorio ingresan, modifican y eliminan.
Pre-Condición	Tener DNI del laboratorista
Flujo normal de los Eventos	
Actor	**Sistema**
1. Acceder al menú Administrador y/o Jefe de laboratorio. 2. Seleccionar opción “Registrar / Editar / Eliminar usuario”. 3. Ingresar datos personales, usuario y contraseña en el formulario que muestra el sistema para su registro. 4. Ingresar el número de DNI para modificar y eliminar un Laboratorista.	5. Validar datos para registrar, editar y eliminar Laboratorista. 6. Registrar, editar y eliminar datos del Laboratorista.
Post Condición	Ingresar a otras opciones del menú Laboratorista.
Excepciones	El sistema mostrara mensaje de error si no se registró, edito y elimino correctamente, o algún error que impida el proceso.

Fuente: Elaboración propia

4.3.2.5. Caso de Uso: Registro de Laboratorios

Figura N° 11: Diagrama de Casos de Uso Registro de Laboratorios



Fuente: Elaboración propia

En la figura N° 11 se muestra como los usuarios (Administrador, Jefe de laboratorio y Laboratorista) realizan las acciones de registrar, editar, eliminar y buscar Laboratorios.

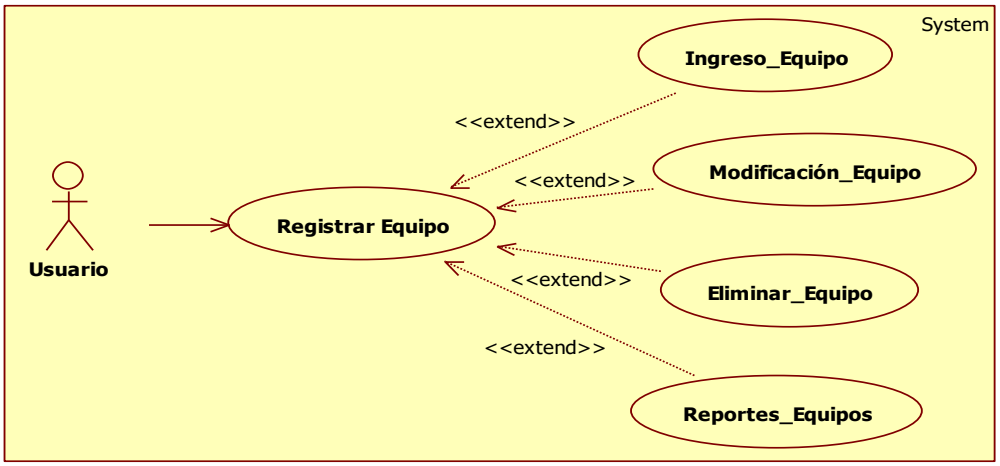
Tabla N° 11: Descripción de Caso de uso Laboratorio

Nombre	Registro de Laboratorios.
Actor	Usuario
Propósito	El usuario registra los Laboratorios de cada Escuela Profesional.
Pre-Condición	La Escuela Profesional debe pertenecer a la Universidad.
Flujo normal de los Eventos	
Actor	**Sistema**
1. Acceder al menú Usuarios. 2. Seleccionar opción “Registrar / Editar / Eliminar Laboratorios”. 3. Ingresar Nombre de la Facultad, Escuela Profesional, Nombre del Laboratorio, encargado del Laboratorio en el formulario que muestra el sistema para su registro.	4. Validar datos para registrar, editar y eliminar Laboratorios. 5. Registrar, editar y eliminar datos del Laboratorio.
Post Condición	Ingresar a otras opciones del menú Usuario.
Excepciones	El sistema mostrara mensaje de error si no se registró correctamente.

Fuente: Elaboración propia

4.3.2.6. Caso de Uso: Registro de Equipos

Figura N° 12: Diagrama de Casos de Uso Registro de Equipos



Fuente: Elaboración propia

En la figura N° 12 se muestra como los usuarios (Administrador, Jefe de laboratorio y Laboratorista) realizan las acciones de registrar, editar, eliminar y buscar Equipos.

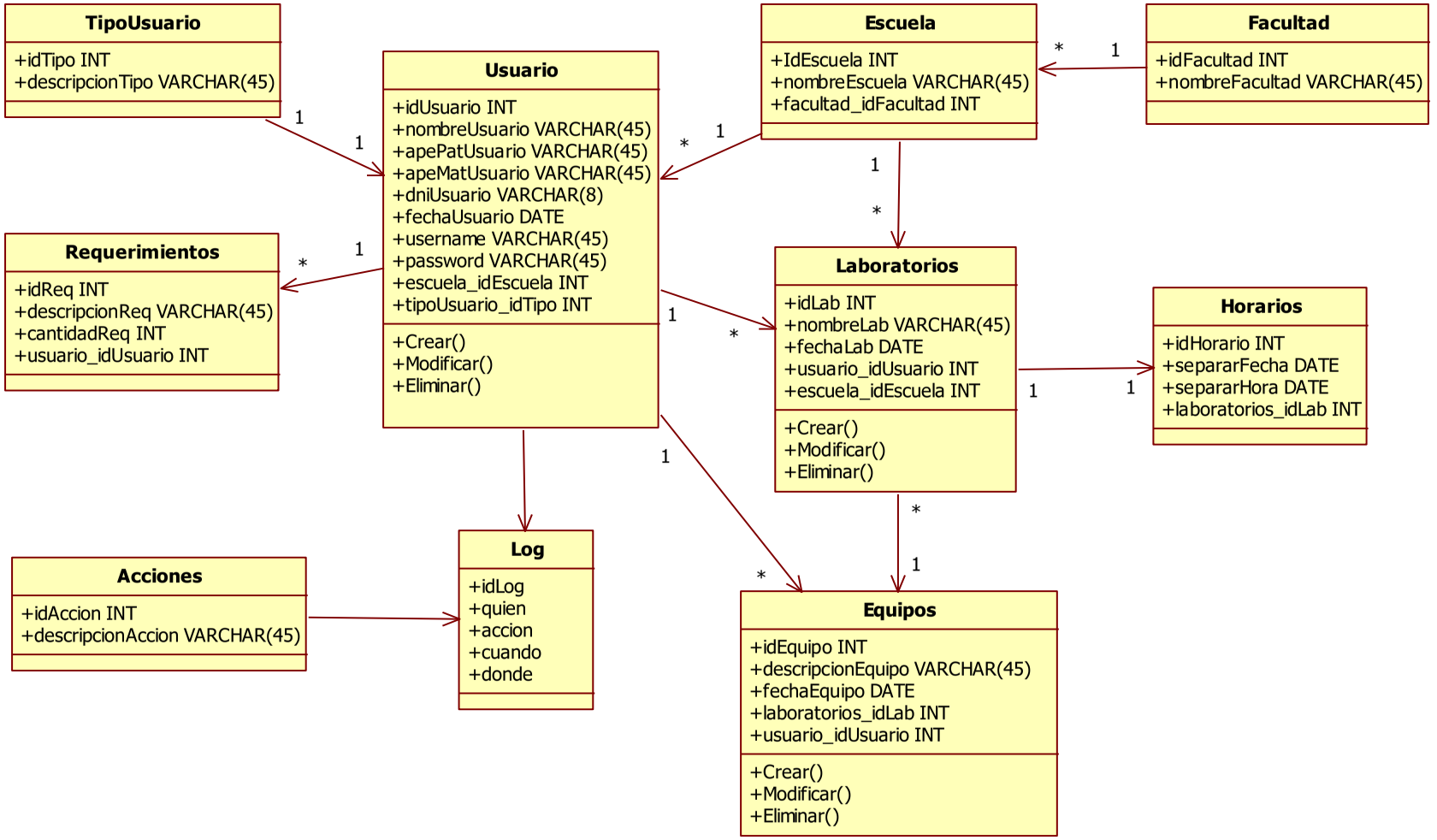
Tabla N° 12: Descripción de Caso de uso Equipos

Nombre	Registro de Equipos.
Actor	Usuario
Propósito	El usuario registra los Equipos de cada Escuela Profesional.
Pre-Condición	La Escuela Profesional debe pertenecer a la Universidad.
Flujo normal de los Eventos	
Actor	**Sistema**
1. Acceder al menú Usuarios. 2. Seleccionar opción “Registrar / Editar / Eliminar Equipos”. 3. Ingresar Nombre del Equipo, Laboratorio, Facultad, Escuela Profesional, encargado del Laboratorio en el formulario que muestra el sistema para su registro. 4. Ingresar el código del equipo para modificar y eliminar un Equipo.	5. Validar datos para registrar, editar y eliminar Equipos. 6. Registrar, editar y eliminar datos del Equipo.
Post Condición	Ingresar a otras opciones del menú Usuario.
Excepciones	El sistema mostrara mensaje de error si no se registró correctamente.

Fuente: Elaboración propia

4.4. Diseño del Sistema

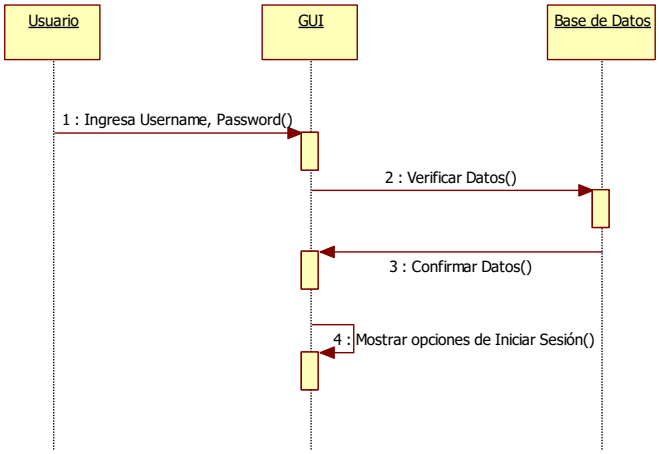
4.4.1. Diagrama de Clases



4.5. Diagrama de secuencia

4.5.1. Diagrama de secuencia: Inicio de sesión

Figura N° 13: Diagrama de secuencia de inicio de sesión

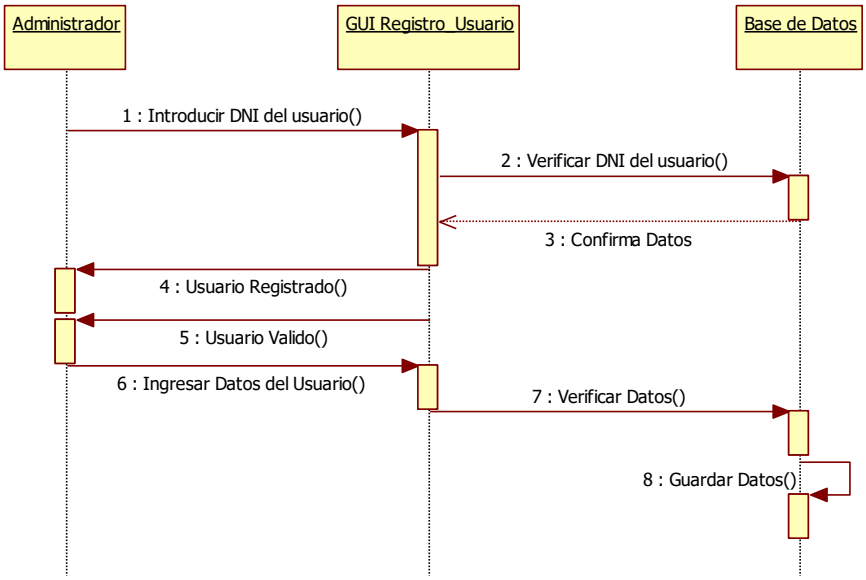


Fuente: Elaboración propia

En el diagrama inicio de sesión el usuario ingresa su cuenta (username y password), para verificar los datos el sistema se conecta con la base de datos, si estos son correctos se accederá al panel del usuario, caso contrario enviara otras opciones de Iniciar Sesión.

4.5.2. Diagrama de secuencia: Registro de un usuario

Figura N° 14: Diagrama de secuencia de registro de un usuario

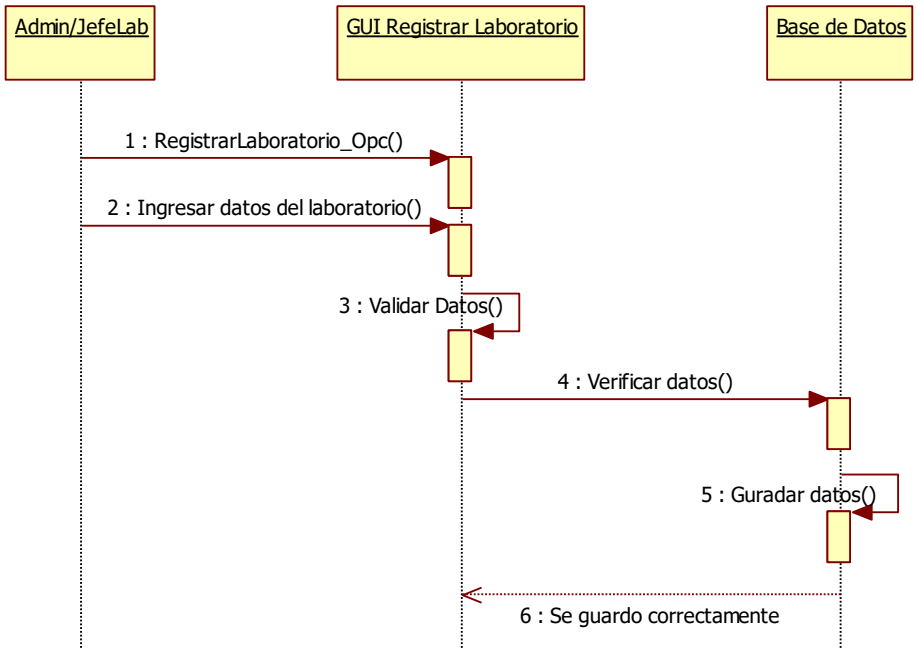


Fuente: Elaboración propia

En el diagrama Registro de usuario el administrador introduce en número de DNI del usuario, el sistema verifica y confirma si está o no registrado, en caso estar registrado enviara un mensaje “usuario valido”, caso contrario se deberá ingresar sus datos personales en el formulario que el sistema muestra para su verificación y registro.

4.5.3. Diagrama de secuencia: Registro de Laboratorios

Figura N° 15: Diagrama de secuencia de registro de laboratorios

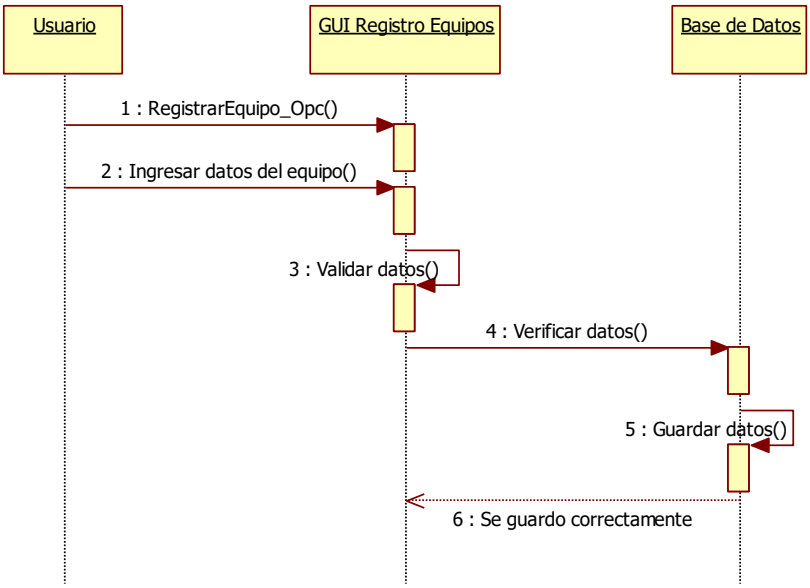


Fuente: Elaboración propia

En el diagrama Registro de laboratorios el administrador o jefe de laboratorio ingresa los datos de los laboratorios en el formulario que el sistema muestra para validar, verificar y guardar en la base de datos.

4.5.4. Diagrama de secuencia: Registro de Equipos

Figura N° 16: Diagrama de secuencia de registro de equipos



Fuente: Elaboración propia

En el diagrama Registro de equipos el usuario ingresa los datos de los equipos en el formulario que el sistema muestra para validar, verificar y guardar en la base de datos.

4.6. Desarrollo del Software del Sistema

El desarrollo del software se realizó con el lenguaje de programación PHP y el lenguaje de marcado HTML en el editor de código Sublime Text, divididos en dos áreas, una es la parte del código y el otro es donde muestra la interfaz la cual está siendo diseñada en la parte del código donde se hace las modificaciones lógicas del software en PHP o en HTML y en la parte del interfaz se puede hacer también modificaciones en mayor cantidad de código puede ser insertado botones, áreas de texto, creación de tablas y todo lo necesario para la creación del software.

Una aplicación PHP es una aplicación dirigida por eventos. El usuario controla la aplicación que se ejecutara provocando eventos. Por ejemplo: cada acción que realiza el usuario como abrir una aplicación provoca un evento. El lenguaje utilizado para escribir procedimientos se denomina JavaScript pueden incluir sentencias, órdenes y funciones. A continuación se presenta un procedimiento para la ejecución para un script el cual sucede cada vez que se ejecute una aplicación o un script PHP y que pida datos a la base de datos.

4.7. Pruebas del Software

Al momento de las ejecuciones reiteradas del sistema no se tuvo inconvenientes sin embargo no está demás explicar que como todo software este se encuentra predispuesto a cualquier tipo de cambio que el usuario requiera con la finalidad de poder optimizar el mismo, esto incluye también las actualizaciones que por tiempo de vida de uso se deben de realizar periódicamente.

- **Diseño de las Interfaces Internas del Programa**

  Este depende de los datos que debe fluir entre los módulos y las características del lenguaje de programación en el que se implementa el software. En general el proceso de análisis contiene mucha información requerida para el diseño de interfaces.

- **Diseño de la Interface Hombre – Maquina**

  Las categorías de diseño de interfaz hombre – máquina a usar son: Interacción general, visualización de información y la entrada de datos.

- **Interacción General**

  La interfaz muestra un formato para la selección del menú, visualización de datos, etc. Ofrece respuestas significativas que garanticen la comunicación en los dos sentidos (entre operador y la interfaz). Reducen la cantidad de información que se debe memorizar para las operaciones.

4.8. Instalación

Durante la instalación no se presentó ningún tipo de problemas, porque esta plataforma fue desarrollada en un entorno web.

4.9. Resultados de la Prueba según el ISO – 9126

Tabla N° 13: Cuadro de decisiones ISO - 9126

Clasificación	Intervalo	Decisión
A) Inaceptable	[27 - 54 >	
B) Mínimamente aceptable	[54 - 81 >	
C) Aceptable	[81 - 95 >	
D) Cumple los requisitos	[95 - 122 >	**103**
E) Excede los requisitos	[122 - 135]	

Fuente: Cuadro de decisiones ISO – 9126

Según los resultados el promedio de 105, 108, 100, 99, 103 y 104 nos resulta 103, indicando que cumple con los requisitos según ISO-9126. Todo esto se muestra en el ANEXO I.

Decisión:

De acuerdo a los resultados de la calidad de software se concluyó que el sistema de información para la automatización de laboratorios en la Universidad Nacional del Altiplano Puno, 2016, cumple los requisitos con un promedio de 103 puntos de un total de 135 puntos que se considera en el cuadro de decisiones del ISO-9126.

4.10. Cuadro Comparativo según el Tiempo

Tabla N° 14: Cuadro comparativo expresado en minutos con o sin sistema

CONDICION	CON SISTEMA	SIN SISTEMA
Registro de datos	3 min	12 min
Búsqueda de datos	2 min	10 min
Salida de Reportes	3 min	12 min
Total	8 min	34 min

Fuente: Cuadro comparativo en minutos con o sin sistema

Decisión:

De acuerdo al cuadro comparativo expresado en minutos de un registro, búsqueda de datos con sistema y sin sistema, se aprecia que utilizando el sistema demora un promedio de 8 minutos por dato mientras que sin el sistema es de 34 minutos por dato.

4.11. Gráficos de Satisfacción de Usuario

GRAFICO N° 01: Calificación de la interfaz y el acceso al sistema de información de laboratorios.

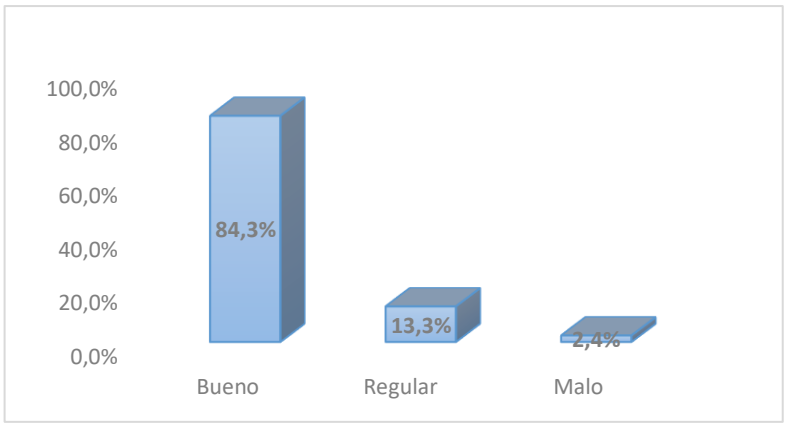


GRAFICO N° 02: Calificación de la fiabilidad del sistema de información de laboratorios.

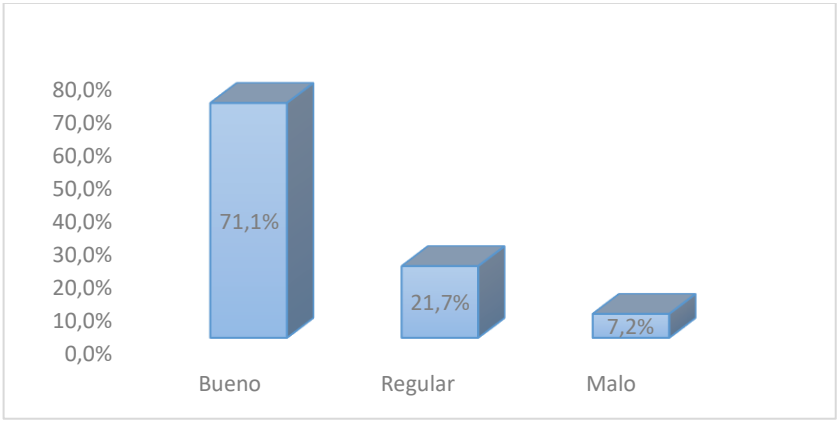


GRAFICO N° 03: Calificación de la seguridad del sistema de información de laboratorios.

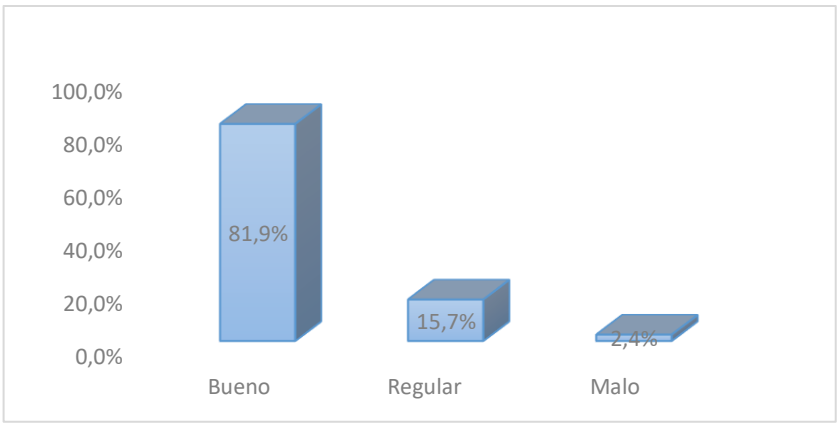


GRAFICO N° 04: Calificación de la rapidez del sistema de información de laboratorios.

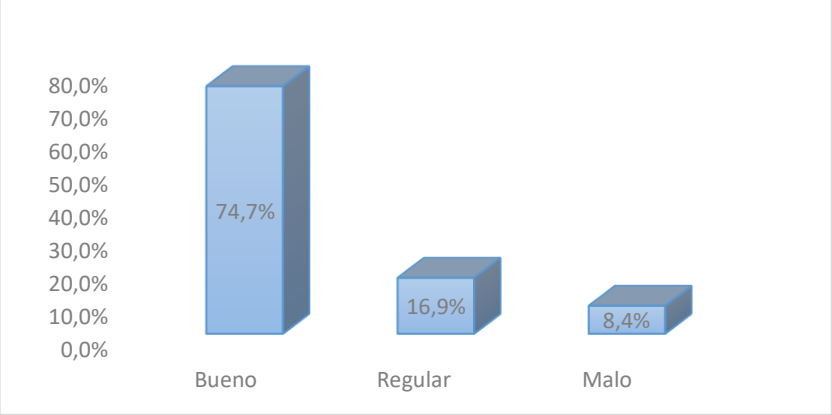
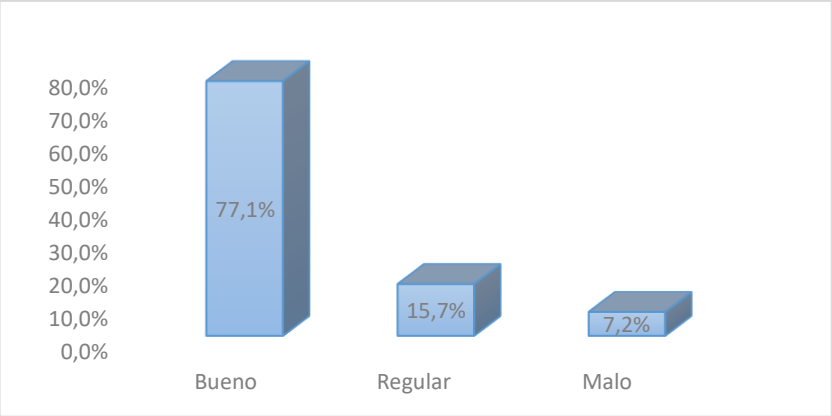


GRAFICO N° 05: Calificación del sistema de información de laboratorios.



4.12. Contraste de Hipótesis

Hipótesis Nula (Ho):

No existe una relación significativa entre el nivel de satisfacción del sistema de información y el tiempo de procesamiento de los datos en la implementación del sistema de información para la administración de laboratorios en la universidad nacional del altiplano en el año 2017.

Hipótesis Alterna (Ha):

Existe una relación significativa entre el nivel de satisfacción del sistema de información y el tiempo de procesamiento de los datos en la implementación del sistema de información para la administración de laboratorios en la universidad nacional del altiplano en el año 2017.

Nivel de significancia

Se considera un nivel de 95% de confiabilidad y una significancia del (α = 0.05) para establecer relación entre las variables evaluadas.

Prueba estadística: Correlación de Pearson

CUADRO N° 01: Correlación de Pearson del nivel de satisfacción y el tiempo de procesamiento de los datos en el año 2017.

		NIVEL DE SATISFACCION DEL SISTEMA WEB	TIEMPO DE PROCESAMIENTO DE LOS DATOS
NIVEL DE SATISFACCION DEL SISTEMA WEB	Correlación de Pearson	1	,7030
	Sig. (bilateral)		**,000**
	N	100	100
TIEMPO DE PROCESAMIENTO DE LOS DATOS	Correlación de Pearson	,711**	1
	Sig. (bilateral)	,000	
	N	100	100

**. La correlación es significativa en el nivel 0,01 (bilateral)

FUENTE: Elaboración en SPSS (Versión 22) a partir de los datos.

Como el P (**0.000**) muy inferior o menor al nivel de significancia α (0.050) entonces la prueba es significativa, por lo tanto, concluimos que tenemos suficiente evidencia para determinar que existe una relación significativa entre el

nivel de satisfacción del sistema web y el tiempo de procesamiento de los datos en la implementación del sistema de información para la administración de laboratorios en la universidad nacional del altiplano en el año 2017.

Asimismo existe una correlación de Pearson de 0.711 lo cual nos indica que existe una correlación directa y muy buena, es decir que los usuarios que están totalmente satisfechos con el sistema, asimismo mencionan que el tiempo de procesamiento de los datos es eficaz y buena.

CAPÍTULO V

CONCLUSIONES

- Se diseñó e implemento un sistema de información para la automatización de laboratorios para los procesos de administración, a través de una interface web amigable, fiable, segura y rápida; para lo cual se utilizó herramientas de software que permitieron acceder de forma fiable a la base de datos para realizar los procesos de registro, modificación, búsqueda y emisión de reportes.

- Se validó el funcionamiento del sistema de información para los procesos de administración de los laboratorios en la Universidad Nacional del Altiplano Puno, con la ficha de evaluación ISO – 9126 en una escala del 27 – 135 puntos, dando como resultado 103 puntos, que significa que el sistema de información cumple con los requisitos de calidad para su funcionamiento.

- Se evaluó el rendimiento y la aceptación del sistema de información con la ficha de encuesta dando como resultado, que un 80% de encuestados manifiestan que es bueno el sistema de información, por lo que un 15% considera que el sistema es regular y tan solo 5% considera que es malo.

CAPÍTULO VI

RECOMENDACIONES

- Se recomienda contar con respaldos periódicamente para mantener en resguardo una copia actualizada de la base de datos, evitando así pérdida de información.

- Se recomienda adquisición de un servidor, con prestaciones mayores en lo que respecta velocidad de proceso, prevención contra apagones, dispositivos de tapes backups, almacenamiento mayor a (2 Terabytes a más).

- Se recomienda en posteriores trabajos de investigación agregar otros componentes al sistema que es necesario para la administración de personal de los laboratorios de las diferentes escuelas profesionales de la Universidad Nacional del Altiplano.

- Se recomienda un constante mantenimiento al sistema y al hardware para su óptima operatividad ya que esto ocasionaría rentabilidad en el futuro, mejoras en el conocimiento de las funciones y además agiliza los procesos de los laboratorios.

CAPÍTULO VII

REFERENCIAS BIBLIOGRÁFICAS

Alegsa. (2013). *Sistema. (En línea). Consultado, 18 de abr. 2017. Formato HTML. Disponible en http://www.alegsa.com.ar/Dic/sistema%20.php*

Alvarez Cedeño, M., Astudillo Sarmiento, R., & Zambrano Herrera, A. (2010). *Sistema Integrado para la automatización de un laboratorio clínico orientado a la Web (Tesis Pregrado). Universidad Politécnica Salesiana, Ecuador.*

Córdova Forero, J. (2014). *Implementación de un Sistema de Matrículas y Pagos para el Centro de Informática de la Universidad César Vallejo (Tesis Pregrado). Universidad de San Martín de Porres.*

Definición de AJAX *Última revisión 25 de Junio del 2016 y disponible en https://es.wikipedia.org/wiki/AJAX*

Definición de Bootstrap *Última revisión 14 de Agosto del 2016 y disponible en https://es.wikipedia.org/wiki/Bootstrap_(framework)*

Definición de Javascript *accesado el 20 de enero de 2012 y disponible en http://www.librosweb.es/javascript/pdf/introduccion_javascript.pd*

Definición Metodología SCRUM *Disponible en https://proyectosagiles.org/que-es-scrum/*

Definición de MySQL *Disponible en https://iiemd.com/mysql/que-es-mysql*

Gauchat, J. D. (2012). *El gran libro de HTML5, CSS3 y Javascript. (Primera edición). Barcelona.*

Jihuallanca Ccoa, N. (2016). *Sistema de consultad de la unidad de pensiones y liquidaciones de la Universidad Nacional del Altiplano Puno. (Tesis Pregrado) Universidad Nacional del Altiplano Puno.*

Julian, P. (2017). *¿Qué es PHP? accesado el 24 de mayo del 2017 y disponible en https://definicion.de/css/*

Kendall, E. y Kendall, J. (2005). *Análisis y diseño de sistemas. (Sexta edición). México. Pearson Education.*

La norma ISO/IEC *9126 accesado el 05 de febrero de 2012 y disponible en http://iso25000.com/index.php/iso-iec-9126.html*

Laudon (2015). *Sistemas de Información.*

https://es.slideshare.net/jaimendoza

Miguel, A. (2001). *¿Qué es PHP? accesado el 01 de enero del 2012 y disponible en http://www.desarrolloweb.com/articulos/392.php*

Peña, A. A. (2006). *Ingeniería de software - guía para crear sistemas de información. (Primera edición). México.*

Pressman, R. S. (2011). *Ingeniería del software un enfoque práctico. (Séptima edición). México.*

Quispe Puma, J. (2017). *Sistema de automatización para los procesos de administración y búsqueda de la biblioteca especializada de la Institución Educativa Secundaria Mariano Hilario Cornejo 32 de la Ciudad de Juliaca 2010. (Tesis Pregrado) Universidad Nacional del Altiplano Puno.*

Ramírez Rodríguez, C., & Vélez Sabando, G. (2015). *Automatización del registro y control de los procesos de hospedaje, restaurante y eventos del hotel-laboratorio "el higuerón" de la espam MFL (Tesis Pregrado). Escuela Superior Politécnica Agropecuaria de Manabí Manuel Félix López, Calceta.*

Riehle, Dirk (2000), *Framework Design: A Role Modeling Approach, Swiss Federal Institute of Technology*

Roldán, A (2016). *Programación Orientada a Objetos http://www.ciberaula.com/articulo/tecnologia_orientada_objetos*

Talledo Farfan, J. (2015). *Desarrollo de un sistema de información de laboratorios clínico con tecnología web para el Centro de Salud de Miguel Checa, Distrito de Miguel Checa – Sullana en el año 2015 (Tesis Pregrado) Universidad Nacional de Piura.*

Torres Cruz, F. (2016). *Plataforma web basada en cloud computing para el seguimiento de proyectos de tesis de pregrado UNA Puno 2016. (Tesis Pregrado). Universidad Nacional del Altiplano Puno.*

ANEXOS

ANEXO 1

FICHA DE EVALUACIÓN DE LA CALIDAD DEL PRODUCTO ESTÁNDAR ISO-9126

INDICADORES	PUNTUACIÓN				
	1	2	3	4	5
1. FUNCIONALIDAD					
Adecuación: La capacidad del producto software para proporcionar un conjunto apropiado de funciones para tareas específicas y objetivos de los usuarios.					
Exactitud: La capacidad del producto software para proporcionar los resultados o efectos correctos y con el grado de precisión acordado.					
Interoperabilidad: La capacidad del producto software para interactuar con uno o más plataformas especificados.					
Seguridad: Referido a la capacidad del producto software para proteger la información y los datos.					
Conformidad: La capacidad del producto software para adaptarse a los estándares, convenciones o regulaciones en leyes y prescripciones relativos a la funcionalidad.					
2. FIABILIDAD					
Madurez: La capacidad del producto software para evitar fallos provocados por errores en el software.					
Tolerancia a fallos: La capacidad del producto software para mantener un nivel de rendimiento determinado en caso de defectos en el software o incumplimiento de su interfaz.					
Recuperabilidad: La capacidad del producto software para restablecer un determinado nivel de rendimiento y recuperar los datos afectados directamente en caso de ocurrir un fallo.					
Conformidad: La capacidad del producto software para adaptarse a estándares, convenciones y regulaciones referidas a la fiabilidad.					
3. USABILIDAD					
Comprensibilidad: La capacidad del producto software para permitir al usuario que entienda si el software es adecuado, y como debe utilizarse para determinadas tareas y bajo ciertas condiciones de uso.					
Facilidad de aprendizaje: La capacidad del producto software para permitir al usuario aprender su aplicación.					
Atracción: La capacidad del producto software para atraer al usuario.					
Conformidad: La capacidad del producto software para adaptarse a estándares, convenciones, guías de estilo y regulaciones relacionadas con la usabilidad.					
Operabilidad: La capacidad del producto software para permitir que el usuario lo opere y lo controle.					
4. EFICIENCIA					
Comportamiento temporal: La capacidad del producto software para proporcionar tiempos de respuesta y de procesamiento apropiados cuando realiza sus funciones bajo condiciones determinadas.					
Utilización de recursos: La capacidad del producto software para utilizar cantidades y tipos de recursos apropiados cuando el software realiza su función bajo determinadas condiciones.					
Conformidad: La capacidad del producto software para adaptarse a estándares o convenciones relacionadas con la eficiencia.					
5. MANTENIBILIDAD					
Analizabilidad: Capacidad del producto software de diagnosticar sus deficiencias o causas de fallos, o de identificar las partes que deben ser modificadas.					
Cambiabilidad: Capacidad del producto software de permitir implementar una modificación especificada. La implementación incluye los cambios en el diseño, el código y la documentación.					
Estabilidad: Capacidad del producto software de evitar los efectos inesperados de las modificaciones.					
Facilidad de prueba: Capacidad del producto software de permitir validar las partes modificadas.					
Conformidad: Capacidad del producto software de cumplir los estándares o convenciones relativas a la mantenibilidad.					
INDICADORES	**PUNTUACION**				
	1	**2**	**3**	**4**	**5**
6. PORTABILIDAD					
Adaptabilidad: La capacidad del producto software para ser adaptado para ambientes determinados sin realizar acciones o aplicar medios, más que los proporcionados para este propósito para el software considerado.					
Facilidad de instalación: La capacidad del producto software para ser instalado en un ambiente determinado.					
Coexistencia: La capacidad del producto software para coexistir con otro software independiente en un ambiente común compartiendo recursos.					
Reemplazabilidad: La capacidad del producto software para ser utilizado en lugar de otro producto de software para el mismo propósito en el mismo ambiente.					
Conformidad: La capacidad del producto software para adaptarse a estándares relacionados con la portabilidad.					
SUB TOTALES					
TOTAL					

Fuente: Ficha de evaluación ISO-9126

Cuadro de decisiones ISO - 9126

Clasificación	Intervalo	Decisión
A) Inaceptable	[27 - 54 >	
B) Mínimamente aceptable	[54 - 81 >	
C) Aceptable	[81 - 95 >	
D) Cumple los requisitos	[95 - 122 >	
E) Excede los requisitos	[122 - 135]	

Fuente: Cuadro de decisiones ISO – 9126

Cuadro de decisiones ISO - 9126

Clasificación	Intervalo	Decisión
F) Inaceptable	[27 - 54 >	
G) Mínimamente aceptable	[54 - 81 >	
H) Aceptable	[81 - 95 >	
I) Cumple los requisitos	[95 - 122 >	
J) Excede los requisitos	[122 - 135]	

Fuente: Cuadro de decisiones ISO – 9126

ANEXO 2

ENCUESTA DE SATISFACCIÓN DE USUARIO

Cuestionario dirigido a jefes de laboratorio y laboratoristas de la Universidad Nacional del Altiplano.

Marque con una equis [x] la opción que usted considere adecuada.

1. **¿Cómo califica la interfaz y el acceso al sistema de información de laboratorios?**
   [ ] Bueno
   [ ] Regular
   [ ] Malo
2. **¿Cómo califica la fiabilidad del sistema de información de laboratorios?**
   [ ] Bueno
   [ ] Regular
   [ ] Malo
3. **¿Cómo califica la seguridad del sistema de información de laboratorios?**
   [ ] Bueno
   [ ] Regular
   [ ] Malo
4. **¿Cómo califica la rapidez del sistema de información de laboratorios?**
   [ ] Bueno
   [ ] Regular
   [ ] Malo
5. **¿Cómo califica el sistema de información de laboratorios?**
   [ ] Bueno
   [ ] Regular
   [ ] Malo

ANEXO 3

UNIVERSIDAD NACIONAL DEL ALTIPLANO

ESCUELA PROFESIONAL DE INGENIERIA ESTADISTICA E INFORMATICA

MANUAL DEL USUARIO

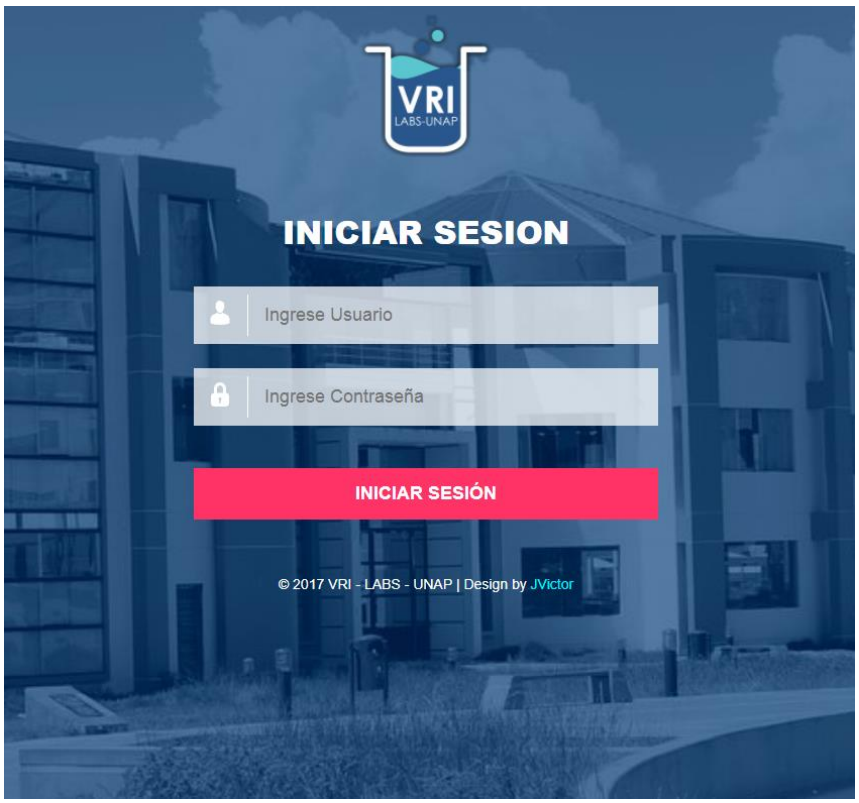
En este documento se presenta el Manual de Usuario del sistema de información para la automatización de laboratorios, que tiene como objetivo proporcionar una guía práctica a los usuarios, para el dominio y aplicación adecuada del sistema, también se precisa cada una de las alternativas del menú principal según el rol de cada usuario, así como las directrices necesarias y las acciones a realizar en cada pantalla.

INGRESO AL SISTEMA

1.1. Activación de cuenta

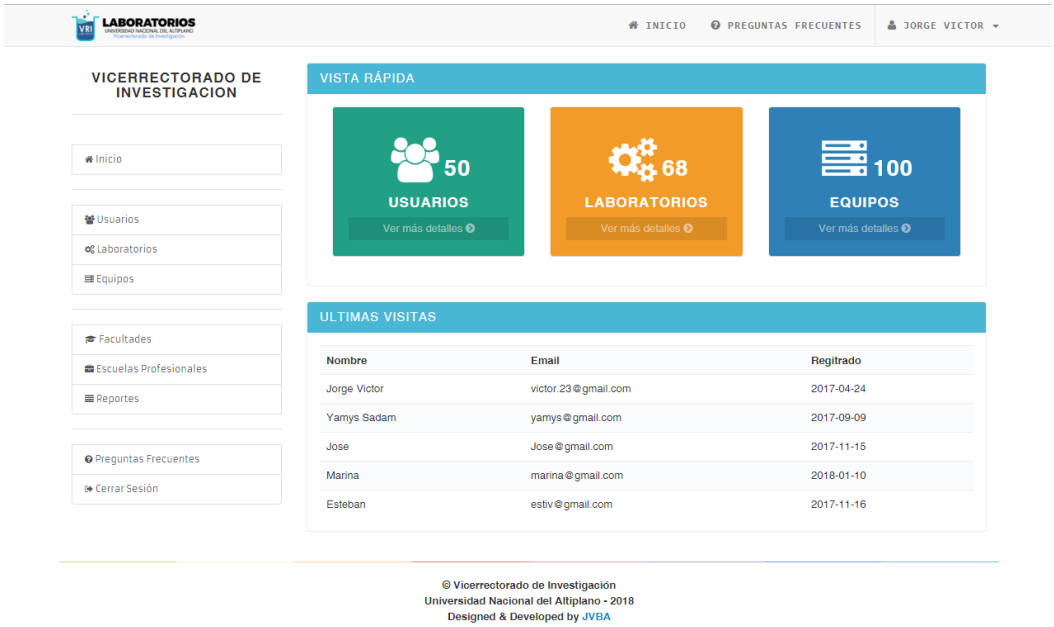
Para activar una cuenta, el usuario deberá estar anticipadamente registrado en el sistema, para ello el administrador debe ingresar los datos del usuario en el panel de control previa validación de requisitos.

Al intentar ingresar al sistema, lo primero que se le mostrara será la ventana de inicio de sesión den donde deberá ingresar la cuenta de usuario y contraseña.



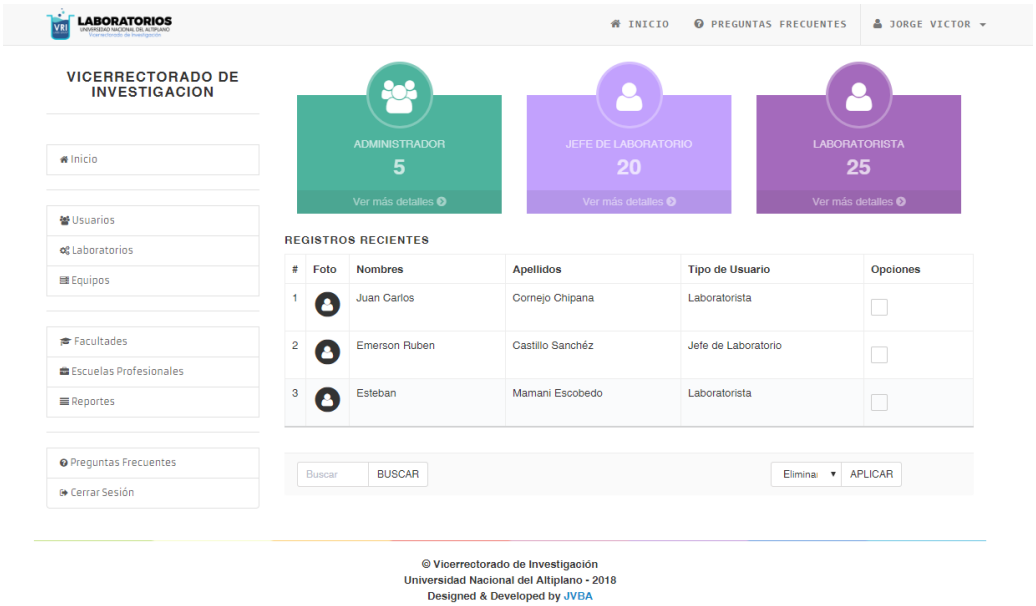
1.2. Ventana del panel de control

Esta ventana permite visualizar todas las opciones disponibles para los respectivos usuarios.



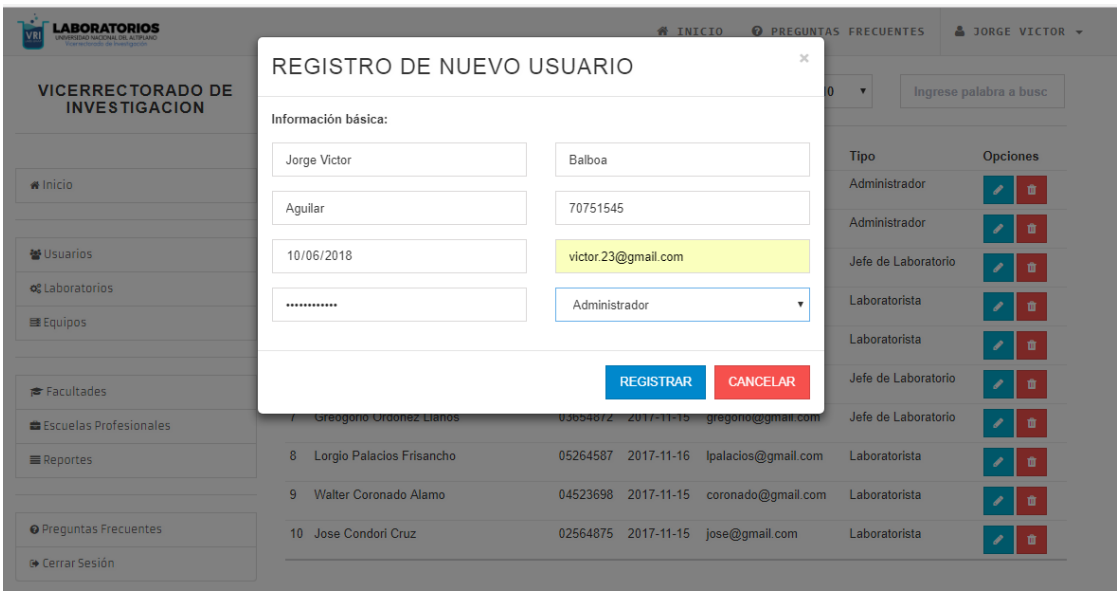
1.3. Panel de control usuarios

En la siguiente ventana se observa los tres tipos de usuarios, en donde cada uno de ellos tiene una tarea diferente. Esta ventana solo puede visualizar el administrador.



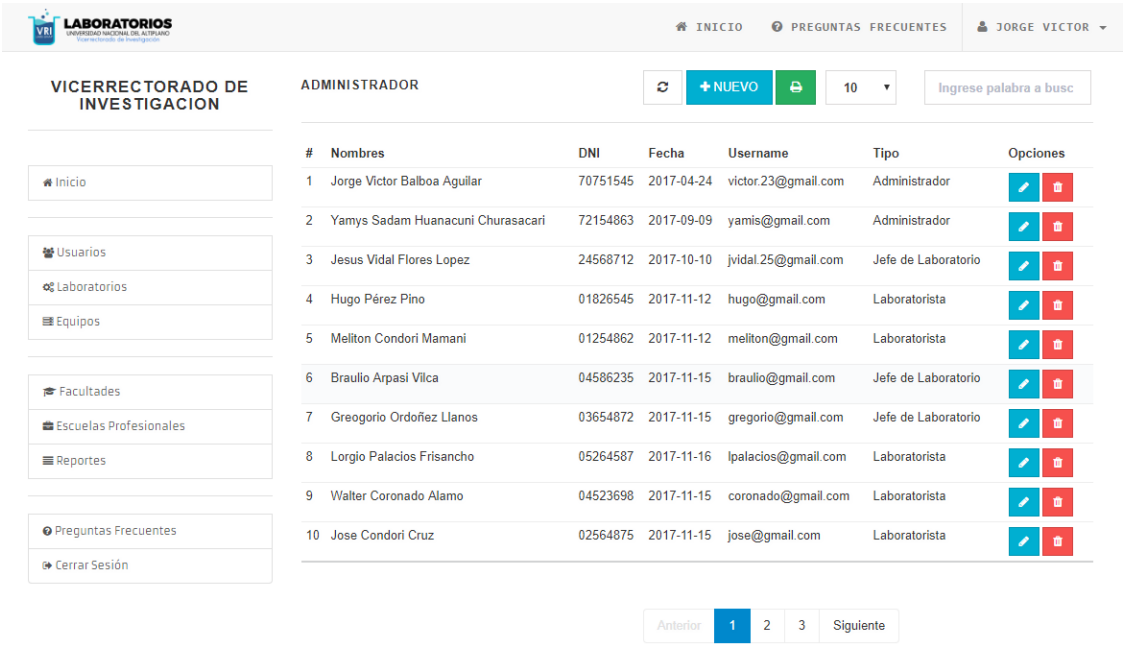
1.4. Registrar usuarios

Ingrese los datos requeridos por el formulario que muestra el sistema para el registro de un usuario, el cual lo puede realizar un administrador o jefe de laboratorio.



1.5. Usuarios registrados

En esta ventana se puede observar los usuarios registrados en el sistema de información.



1.6. Reporte de usuarios

En esta ventana se realiza el reporte de los usuarios registrados en el sistema, para esta acción deberá hacer clic en reportes.

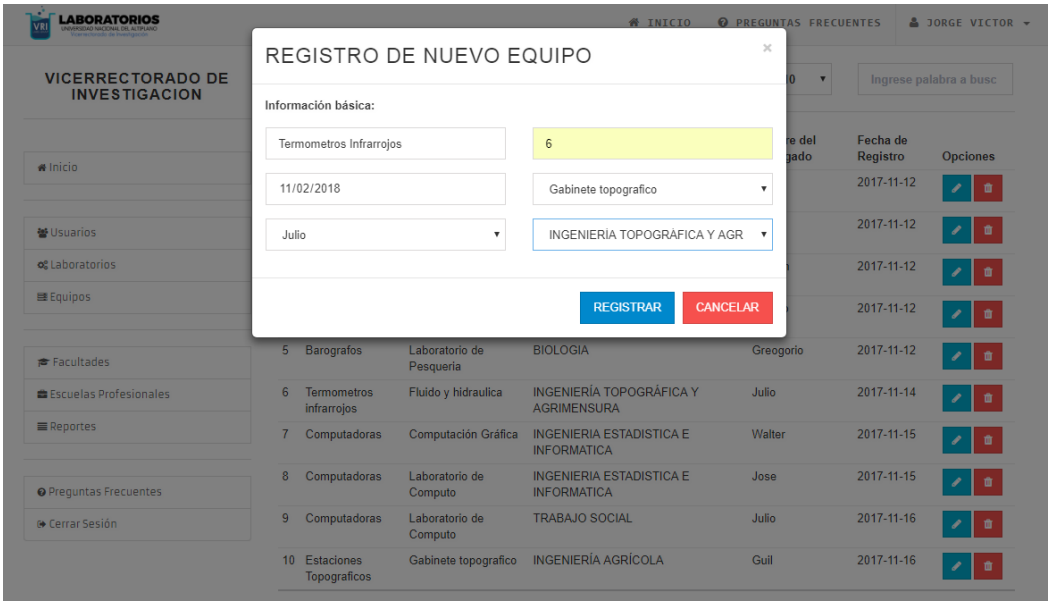
LABORATORIOS
UNIVERSIDAD NACIONAL DEL ALTIPLANO
Vicerrectorado de Investigación

Universidad Nacional del Altiplano
Vicerrectorado de Investigación
Reporte de Usuarios

N°	Nombres	Tipo Usuario	email	DNI	Fecha
1	Jorge Victor Balboa Aguilar	Administrador	victor.23@gmail.com	70751545	24/04/17
2	Yamys Sadam Huanacuni Churasacari	Administrador	yamis@gmail.com	72154863	09/09/17
3	Jesus Vidal Flores Lopez	Jefe de Laboratorio	jvidal.25@gmail.com	24568712	10/10/17
4	Hugo Perez Pino	Laboratorista	hugo@gmail.com	01826545	12/11/17
5	Meliton Condori Mamani	Laboratorista	meliton@gmail.com	01254862	12/11/17
6	Braulio Arpasi Vilca	Jefe de Laboratorio	braulio@gmail.com	04586235	15/11/17
7	Greogorio OrdoÃ±ez Llanos	Jefe de Laboratorio	gregorio@gmail.com	03654872	15/11/17
8	Lorgio Palacios Frisancho	Laboratorista	lpalacios@gmail.com	05264587	16/11/17
9	Walter Coronado Alamo	Laboratorista	coronado@gmail.com	04523698	15/11/17
10	Jose Condori Cruz	Laboratorista	jose@gmail.com	02564875	15/11/17

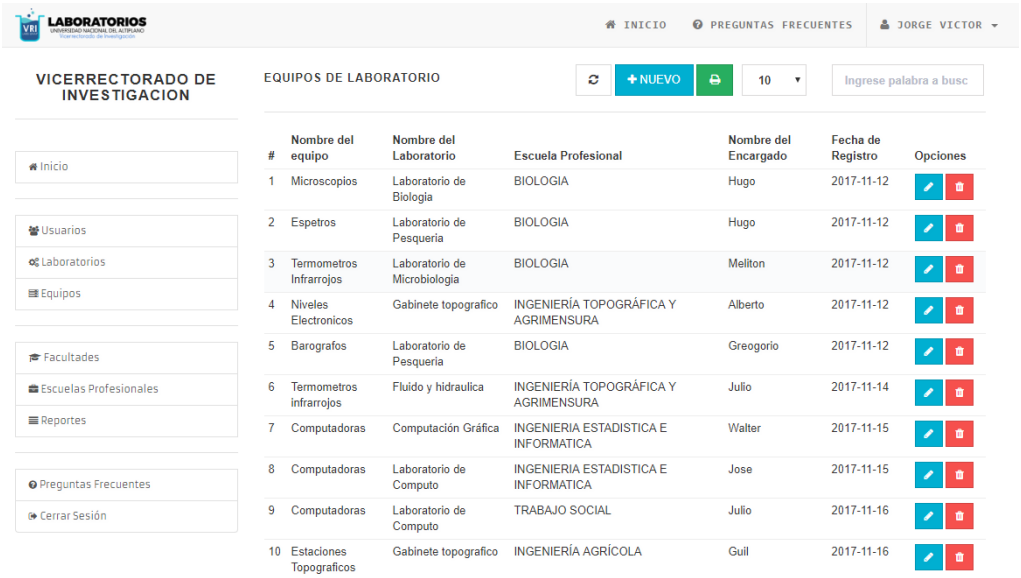
1.7. Registro de equipos

Ingrese los datos requeridos por el formulario que muestra el sistema para el registro de un equipo, el cual lo puede realizar un administrador, jefe de laboratorio o laboratorista.



1.8. Equipos registrados

En esta ventana se puede observar los equipos registrados en el sistema de información.



#	Nombre del equipo	Nombre del Laboratorio	Escuela Profesional	Nombre del Encargado	Fecha de Registro	Opciones
1	Microscopios	Laboratorio de Biologia	BIOLOGIA	Hugo	2017-11-12	
2	Espetros	Laboratorio de Pesqueria	BIOLOGIA	Hugo	2017-11-12	
3	Termometros Infrarrojos	Laboratorio de Microbiologia	BIOLOGIA	Meliton	2017-11-12	
4	Niveles Electronicos	Gabinete topografico	INGENIERÍA TOPOGRÁFICA Y AGRIMENSURA	Alberto	2017-11-12	
5	Barografos	Laboratorio de Pesqueria	BIOLOGIA	Greogorio	2017-11-12	
6	Termometros infrarrojos	Fluido y hidraulica	INGENIERÍA TOPOGRÁFICA Y AGRIMENSURA	Julio	2017-11-14	
7	Computadoras	Computación Gráfica	INGENIERIA ESTADISTICA E INFORMATICA	Walter	2017-11-15	
8	Computadoras	Laboratorio de Computo	INGENIERIA ESTADISTICA E INFORMATICA	Jose	2017-11-15	
9	Computadoras	Laboratorio de Computo	TRABAJO SOCIAL	Julio	2017-11-16	
10	Estaciones Topograficos	Gabinete topografico	INGENIERÍA AGRÍCOLA	Guil	2017-11-16	

1.9. Reporte de usuarios

En esta ventana se realiza el reporte de los equipos registrados en el sistema, para esta acción deberá hacer clic en reportes.

LABORATORIOS
UNIVERSIDAD NACIONAL DEL ALTIPLANO
Vicerrectorado de Investigación

Universidad Nacional del Altiplano
Vicerrectorado de Investigación
Reporte de Equipos

N°	Nombre del Equipo	Nombre del Laboratorio	Escuela Profesional	Encargado	Fecha
1	Microscopios	Laboratorio de Biologia	BIOLOGIA	Hugo	12/11/17
2	Espetros	Laboratorio de Pesqueria	BIOLOGIA	Hugo	12/11/17
3	Termometros Infrarrojos	Laboratorio de Microbiologia	BIOLOGIA	Meliton	12/11/17
4	Niveles Electronicos	Gabinete Topografico	INGENIERÍA TOPOGRÁFICA	Alberto	12/11/17
5	Barografos	Laboratorio de Pesqueria	BIOLOGIA	Greogorio	12/11/17
6	Termometros infrarrojos	Fluido e Hidraulica	INGENIERÍA TOPOGRÁFICA	Julio	14/11/17
7	Computadoras	Computación Gráfica	INGENIERIA ESTADISTICA E	Walter	15/11/17
8	Computadoras	Laboratorio de Computo	INGENIERIA ESTADISTICA E	Jose	15/11/17
9	Computadoras	Laboratorio de Computo	TRABAJO SOCIAL	Julio	16/11/17
10	Estaciones Topograficos	Gabinete topografico	INGENIERÍA AGRÍCOLA	Guil	16/11/17

ANEXO - IV

CÓDIGO FUENTE

```
<?php
/*******************************************************************************
 *      Sistema de Laboratorios
 *      Versión: 1.0
 *      Programador: Bach. Jorge Victor Balboa Aguilar
 *      Programador: Bach. Yamys Sadam Huanacuni Churasacari
 *      Architect: Bach. Jorge Victor Balboa Aguilar
 *
 *******************************************************************************/
define('ENVIRONMENT', 'development');
witch (ENVIRONMENT)
{
        case 'development':
                error_reporting(-1);
                ini_set('display_errors', 1);
        break;
        case 'testing':
        case 'production':
                ini_set('display_errors', 0);
                if (version_compare(PHP_VERSION, '5.3', '>='))
                {
                        error_reporting();
                }
                else {
                        error_reporting();
                }
        break;
        default:
                header('HTTP/1.1 503 Service Unavailable.', TRUE, 503);
                echo 'The application environment is not set correctly.';
                exit(1);
        }
```

```
33          if (defined('STDIN'))   {
34                  chdir(dirname(__FILE__));
35          }
36          if (($_temp = realpath($system_path)) !== FALSE) {
37                  $system_path = $_temp.'/';
38          }
39          else {
40                  $system_path = rtrim($system_path, '/').'/';
41          }
42          if ( ! is_dir($system_path))
43          {
44                  header('HTTP/1.1 503 Service Unavailable.', TRUE, 503);
45                  echo 'Your system folder path does not appear to be set
46  correctly. Please open the following file and correct this:
47  '.pathinfo(__FILE__, PATHINFO_BASENAME);
48                  exit(3);
49          }
50          define('SELF', pathinfo(__FILE__, PATHINFO_BASENAME));
51          define('BASEPATH', str_replace('\\', '/', $system_path));
52          define('FCPATH', dirname(__FILE__).'/');
53          define('SYSDIR', trim(strrchr(trim(BASEPATH, '/'), '/'), '/'));
54          if (is_dir($application_folder))          {
55                  if (($_temp = realpath($application_folder)) !== FALSE) {
56                          $application_folder = $_temp;
57                  }
58                  define('APPPATH',
59  $application_folder.DIRECTORY_SEPARATOR);
60          }
61          else {
62                  $view_folder = rtrim($view_folder,
63  '/\\').DIRECTORY_SEPARATOR;
64          }
65          define('VIEWPATH', $view_folder);
66  require_once BASEPATH.'core/CodeIgniter.php';
```

Implementación registro de equipos

```
1  <?php
2  defined('BASEPATH') OR exit('No direct script access allowed');
3  class Equipos extends CI_Controller {
4    public function __construct()
5    {
6        parent::__construct();
7        $this->load->model('equipos_model');
8        $this->load->library('fpdf_sislab');
9    }
10   public function index()
11   {
12     $data['list_lab'] = $this->equipos_model->get_laboratorios();
13     $data['list_usu'] = $this->equipos_model->get_usuarios();
14     $data['list_esc'] = $this->equipos_model->get_escuelas();
15     $data['contenido_a'] = 'equipos_view';
16     $this->load->view('template_a', $data);
17   }
18   public function ajax_list()
19   {
20       $list = $this->equipos_model->get_datatables();
21       $data = array();
22       $no = $_POST['start'];
23       foreach ($list as $equipos) {
24           $no++;
25           $row = array();
26           $row[] = $equipos->idEquipo;
27           $row[] = $equipos->nombre;
28           $row[] = $equipos->nombreLab;
29           $row[] = $equipos->nombreEsc;
30           $row[] = $equipos->nombreUsuario;
31           $row[] = $equipos->fechaEquipo;
32           $row[] = '<a class="btn btn-sm btn-square btn-info"
33 href="javascript:void()" title="Editar" onclick="editUser('."'".$equipos-
34 >idEquipo."'".')"><i class="fa fa-pencil"></i></a>
35
```

```
36              G<a class="btn btn-sm btn-square btn-danger"
37  href="javascript:void()" title="Eliminar" onclick="deleteUser('".$equipos-
38  >idEquipo."'.')"><i class="fa fa-trash"></i></a>';
39          $data[] = $row;
40      }
41          $output = array(
42                      "draw" => $_POST['draw'],
43                      "recordsTotal" => $this->equipos_model->count_all(),
44                      "recordsFiltered" => $this->equipos_model-
45  >count_filtered(),
46                      "data" => $data,
47              );
48          echo json_encode($output);
49   }
50     private function hash_password($password) {
51     return password_hash($password, PASSWORD_BCRYPT);
52   }
53   public function ajax_edit($id)
54   {
55      $data = $this->equipos_model->get_by_id($id);
56      echo json_encode($data);
57   }
58   public function ajax_add()
59   {
60      $this->_validate();
61      $data = array(
62                  'nombre' => $this->input->post('nombre'),
63                  'descripcionEquipo'  => $this->input-
64  >post('descripcionEquipo'),
65                  'fechaEquipo' => $this->input->post('fechaEquipo'),
66                  'laboratorios_idLab' => $this->input->post('nombreLab'),
67                  'usuario_idUsuario' => $this->input->post('nombreUsuario'),
68                  'escuela_id' => $this->input->post('nombreEsc')
69          );
70      $insert = $this->equipos_model->save($data);
71      echo json_encode(array("status" => TRUE));
```

```
72    }
73    public function ajax_update()
74    {
75        $this->_validate();
76        $data = array(
77                'nombre' => $this->input->post('nombre'),
78                'descripcionEquipo'  => $this->input-
79  >post('descripcionEquipo'),
80                'fechaEquipo' => $this->input->post('fechaEquipo'),
81                'laboratorios_idLab' => $this->input->post('nombreLab'),
82                'usuario_idUsuario' => $this->input->post('nombreUsuario'),
83                'escuela_id' => $this->input->post('nombreEsc')
84            );
85        $this->equipos_model->update(array('idEquipo' => $this->input-
86  >post('idEquipo')), $data);
87        echo json_encode(array("status" => TRUE));
88    }
89    public function ajax_delete($id)
90    {
91        $this->equipos_model->delete_by_id($id);
92        echo json_encode(array("status" => TRUE));
93    }
94    public function ajax_list_delete()
95     {
96         $list_id = $this->input->post('id');
97         foreach ($list_id as $id) {
98             $this->equipos_model->delete_by_id($id);
99         }
100        echo json_encode(array("status" => TRUE));
101    }
102 }
103 /* End of file users.php */
104 /* Location: ./application/controllers/users.php */
```