UNIVERSIDAD TÉCNICA DE AMBATO

FACULTAD DE INGENIERÍA EN SISTEMAS ELECTRÓNICA E INDUSTRIAL

CARRERA DE INGENIERÍA EN SISTEMAS COMPUTACIONALES E INFORMÁTICOS

TEMA:

SISTEMA EMBEBIDO IoT PARA LA FACULTAD DE INGENIERÍA EN SISTEMAS, ELECTRÓNICA E INDUSTRIAL APLICANDO PROGRAMACIÓN LIMPIA Y PATRONES DE DISEÑO.

Trabajo de Graduación. Modalidad: Proyecto de Investigación, presentado previo la obtención del título de Ingeniero en Ingeniero en Sistemas Computacionales e Informáticos

LÍNEA DE INVESTIGACIÓN: Desarrollo de Software

AUTOR: Tisalema Guamanquispe Andrés Alejandro
TUTOR: Ing. Urvina Renato Mg.

Ambato - Ecuador
Agosto, 2017

CERTIFICACIÓN DEL TUTOR

En mi calidad de Tutor del Trabajo de Investigación sobre el Tema:

"SISTEMA EMBEBIDO IoT PARA LA FACULTAD DE INGENIERÍA EN SISTEMAS, ELECTRÓNICA E INDUSTRIAL APLICANDO PROGRAMACIÓN LIMPIA Y PATRONES DE DISEÑO", del señor Tisalema Guamanquispe Andrés Alejandro, estudiante de la Carrera de Ingeniería en Sistemas Computacionales e Informáticos, de la Facultad de Ingeniería en Sistemas, Electrónica e Industrial, de la Universidad Técnica de Ambato, considero que el informe investigativo reúne los requisitos suficientes para que continúe con los trámites y consiguiente aprobación de conformidad con el Art. 16 del Capítulo II, del Reglamento de Graduación para Obtener el Título Terminal de Tercer Nivel de la Universidad Técnica de Ambato

Ambato, agosto de 2017

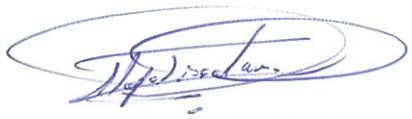
Ing. Urvina Renato Mg.

EL TUTOR

AUTORÍA DEL TRABAJO

El presente trabajo de investigación titulado: "SISTEMA EMBEBIDO IoT PARA LA FACULTAD DE INGENIERÍA EN SISTEMAS, ELECTRÓNICA E INDUSTRIAL APLICANDO PROGRAMACIÓN LIMPIA Y PATRONES DE DISEÑO". Es absolutamente original, auténtico y personal, en tal virtud, el contenido, efectos legales y académicos que se desprenden del mismo son de exclusiva responsabilidad del autor.

Ambato, agosto de 2017



Tisalema Guamanquispe Andrés Alejandro

CC: 1804433900

APROBACIÓN DEL TRIBUNAL DE GRADO

La Comisión Calificadora del presente trabajo conformada por los señores docentes Ing. Rubén Nogales e Ing. Dennis Chicaiza, revisó y aprobó el Informe Final del trabajo de graduación titulado "SISTEMA EMBEBIDO IoT PARA LA FACULTAD DE INGENIERÍA EN SISTEMAS, ELECTRÓNICA E INDUSTRIAL APLICANDO PROGRAMACIÓN LIMPIA Y PATRONES DE DISEÑO", presentado por el señor Tisalema Guamanquispe Andrés Alejandro de acuerdo al Art. 17 del Reglamento de Graduación para obtener el título Terminal de tercer nivel de la Universidad Técnica de Ambato.

Ing. Pilar Urrutia Mg.

PRESIDENTE DEL TRIBUNAL

Ing. Rubén Nogales Mg.
DOCENTE CALIFICADOR

Ing. Dennis Chicaiza Mg.
DOCENTE CALIFICADOR

DEDICATORIA

El presente trabajo se lo dedico a mis Padres, por su ejemplo de lucha constante, su apoyo incondicional y sobre todo por el amor que me han brindado en el transcurso de mi vida, y durante esta etapa académica.

A mis hermanas, Mónica, Mayra; y a quien considero mi hermano Eduardo, quienes me han brindado su apoyo, y han sido un ejemplo de esfuerzo y superación.

Tisalema Guamanquispe Andrés Alejandro

AGRADECIMIENTO

Quiero extender un sincero agradecimiento a Dios por brindarme la vida, la sabiduría y la perseverancia para poder culminar está meta de formación profesional en mi vida.

A mi Familia, por estar presentes en cada momento de mi vida con su amor, su ejemplo y apoyo incondicional.

A los docentes e investigadores de la Unidad Operativa de Investigación y Desarrollo de la Facultad de Ingeniería en Sistemas, Electrónica e Industrial, por su especial apoyo en el desarrollo del proyecto.

Tisalema Guamanquispe Andrés Alejandro

ÍNDICE

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

RESUMEN EJECUTIVO

La presente investigación comprende el desarrollo de un sistema embebido para el control y monitoreo del uso de aulas y laboratorios de la Facultad de Ingeniería en Sistemas, Electrónica e Industrial de la Universidad Técnica de Ambato, a través del diseño y desarrollo de sistemas embebido y web siguiendo una metodología ágil de desarrollo de software, identificando reglas de programación limpia y patrones de diseño que el software necesita. Así mismo se incorporó un sensor biométrico, con la finalidad de identificar al usuario por su huella dactilar, debido a que es el sistema más común en el proceso de identificación a personas. Una huella dactilar es una característica biométrica que se basa en la presencia de un conjunto de líneas genéticas llamadas crestas y valles. El sistema de control permite optimizar el proceso de prestamos de aulas y laboratorios a la comunidad de la facultad.

ABSTRACT

The present research comprises the development of an embedded system for the control of loans of classrooms and laboratories of the Faculty of Engineering in Systems, Electronics and Industrial of the Technical University of Ambato, through the design and development of embedded and web systems. In addition to identifying clean programming rules and design patterns that the software needs. Likewise a biometric sensor was incorporated, with the purpose to identify the user by its fingerprint. A fingerprint is a biometric feature that is based on the presence of a set of genetic lines called ridges and valleys. The control system allows optimize the loan process of classrooms and laboratories to the community of the faculty.

INTRODUCCIÓN

Las huellas dactilares son características que se manifiestan a partir del período de gestación del ser humano, son invariantes, propias y unívocas del individuo [1].
De este modo la operación de un sistema de reconocimiento de huellas dactilares está basado en [2]:

- Capturar un rasgo biométrico.
- Extraer un conjunto de características.
- Comparar con varios patrones almacenados.
- Identificar al usuario.

Los sensores biométricos son dispositivos embebidos compatibles con varios sistemas operativos como Windows o Linux a través de su software de control, de esta manera hace posible su integración con otros dispositivos embebidos de bajo costo como Raspberry Pi. Un sistema embebido es un sistema basado en uno o varios microprocesadores diseñado para realizar determinadas funciones, son programados previamente para realizar ciertas tareas El usuario final puede interactuar con él, pero no podrá cambiar su función principal [3].
En este contexto se busca la implementación de un sistema embebido IoT, su propósito es el control de seguridad electrónico para el acceso a aulas y laboratorios Facultad de Ingeniería en Sistemas, Electrónica e Industrial de la Universidad Técnica de Ambato, a través del reconocimiento de huellas dactilares en la nube.
El presente proyecto de investigación consta de cinco capítulos, los mismas que se detallan a continuación:

Capítulo 1, se realiza el planteamiento del problema del tema de investigación "Sistema Embebido IoT para el Control de Acceso a Aulas y Laboratorios", además de identificar sus limitaciones, justificación y sus objetivos.

Capítulo 2, presenta la fundamentación teórica definida por los diferentes conceptos, además de los casos de éxito o antecedentes investigativos nacionales e

internacionales relacionado al control de seguridad en instituciones y el desarrollo en dispositivos embebidos.

Capítulo 3, consta de la modalidad básica de la investigación, recolección y procesamiento de la información. Se describe las actividades a desarrollar para el cumplimiento del proyecto de investigación.

Capítulo 4, presenta el desarrollo de la propuesta, cumpliendo con la metodología de desarrollo seleccionada, siguiendo las actividades descritas en el capítulo 3, y además cumpliendo con los objetivos planteados en el capítulo 2.

Capítulo 5, consta de las conclusiones y recomendaciones del proyecto de investigación, además de las referencias bibliográficas y anexos del proyecto.

CAPÍTULO 1

El problema

1.1. Tema de Investigación

Sistema Embebido IoT para la Facultad de Ingeniería en Sistemas, Electrónica e Industrial aplicando Programación Limpia y Patrones de Diseño.

1.2. Planteamiento del problema

A través de los años el concepto de control de acceso ha tomado un papel trascendental como una solución para la seguridad de la información, partiendo de sistemas tradicionales a sistemas con tecnología avanzada. Las necesidades en el ámbito de la seguridad están orientadas a la integración de distintas tecnologías, que brinden soluciones integrales al control de acceso. Dentro de este contexto, no sólo las empresas corporativas están requiriendo de estas herramientas, sino también instituciones públicas como ministerios, centros educativos hasta las compañías del mundo industrial.

En Ecuador existen varios sistemas que tienen como objetivo el control, monitoreo y restricción de acceso a un área de uso personal, laboral o educativo. Estos sistemas muy difícilmente se pueden ajustar a la evolución y cambios en los requerimientos que presentan empresas e instituciones.

En base al Modelo de Evaluación Institucional de Universidades y Escuelas Politécnicas [4], aprobado por el Consejo de Evaluación, Acreditación y Aseguramiento de la Calidad de la Educación Superior (CEAACES) en el año 2015, para el proceso de acreditación.

Las Instituciones de Educación Superior (IES), se han preocupado en mejorar sus sistemas de acceso, partiendo de un control de registro manual a dispositivos de control de acceso biométrico como: BioEntry Plus, BioStation, BioDevice, Stylus 980, entre otros. Estos son dispositivos de código cerrado, costosos y que no presentan la flexibilidad que una institución requiere.

La Facultad de Ingeniería en Sistemas, Electrónica e Industrial de la Universidad Técnica de Ambato cuenta con un número de 20 aulas y 19 laboratorios, utilizados por docentes, estudiantes y el personal administrativo de la facultad.

En la facultad existen 3 oficinas que funcionan como administración y préstamo de laboratorios, 2 oficinas en el edificio central para la carrera de sistemas y 1 oficina en el edificio secundario para las carreras de electrónica e industrial. Desde estas oficinas se realiza el préstamo de laboratorios mediante un proceso manual a cargo de los auxiliares de laboratorios.
Este proceso es lento e inseguro debido a que su registro de uso se lo realiza a través de planillas impresas.
Además, para realizar un ingreso a aulas y laboratorios únicamente se requieren de llaves, lo cual implica que pueden ser prestadas, perdidas, robadas o copiadas siendo esta una vulnerabilidad notable para la seguridad del control de acceso de personas.

1.3. Delimitación

- Área Académica: Software.
- Línea de Investigación: Desarrollo de Software.
- Sublínea de investigación: Aplicaciones distribuidas.
- Delimitación espacial: Unidad Operativa de Investigación y Desarrollo de la Facultad de Ingeniería en Sistemas, Electrónica e Industrial.
- Delimitación temporal: La presente investigación se desarrollará en los 6 meses posteriores a la aprobación del proyecto por parte del Honorable Consejo Directivo de la Facultad de Ingeniería en Sistemas, Electrónica e Industrial.

1.4. Justificación

La importancia académica teórica del presente trabajo se refleja a través del desarrollo de un Sistema de control de acceso a aulas y laboratorios, se contará con un registro de ingresos y salidas, mismo que se almacenará en una base de datos para su administración. El desarrollo del proyecto será complementado mediante la aplicación de programación limpia y patrones de diseño de software, asegurando así calidad y flexibilidad al software.
La automatización del proceso de préstamos tendrá como beneficiarios directos al personal docente y estudiantil y beneficiarios indirectos al personal administrativo de la Facultad de Ingeniería en Sistemas, Electrónica e Industrial, debido a que la información de uso de aulas y laboratorios será almacenada para su

administración y control, a través de reportes estadísticos. Después de analizar la problemática que se presenta al momento de acceder al uso de aulas y laboratorios, observando los requerimientos que exige un sistema de control acceso para su funcionamiento como fluido eléctrico constante y estable, además de acceso a la red, sin tomar en cuenta factores externos, se indica que el proyecto es técnicamente factible de realizar e implementar. Se indica que el sistema de control de acceso es económicamente factible a través del proyecto de investigación "Plataforma CLOUDIOT de Control y Monitoreo del Uso de Equipamiento y Programas Informáticos en Aulas y Laboratorios" aprobado por HCU según resolución 2496-CU-P-2015, y según el reglamento para la gestión de proyectos de investigación, capítulo IV, artículo 31.

> "Los desembolsos de los Proyectos de Investigación se lo realizarán una vez que los contratos hayan sido suscritos y serán del 100 % cuando sean destinados para la adquisición de suministros y materiales, servicios generales, y bienes muebles.".

Además, se indica que el proyecto es legalmente factible en base a la norma de control interno para las entidades, organismos del sector público y de las personas jurídicas de derecho privado que dispongan de recursos públicos, capítulo VI, artículo 102.

> "El control interno de las entidades, organismo del sector público y personas jurídicas de derecho privado que dispongan de recursos públicos para alcanzarla misión institucional, deberá contribuir al cumplimiento de los siguientes objetivos:
>
> Promover la eficiencia, eficacia y economía de las operaciones bajo principios éticos y de transparencia.
>
> Garantizar la confiabilidad, integridad y oportunidad de la información.
>
> Cumplir con las disposiciones legales y la normativa de la entidad para otorgar bienes y servicios públicos de calidad.
>
> Proteger y conservar el patrimonio público contra pérdida, despilfarro, uso indebido, irregularidad o acto ilegal."

1.5. Objetivos

1.5.1. General

Implementar un Sistema Embebido IoT para el Control de Acceso a Aulas y Laboratorios para la Facultad de Ingeniería en Sistemas, Electrónica e Industrial

aplicando Programación Limpia y Patrones de Diseño de Software.

1.5.2. Específicos

- Determinar las características que debe cumplir el desarrollo de software para que su código sea considerado limpio.
- Desarrollar un Sistema Embebido IoT para el control de acceso a aulas y laboratorios en la Facultad de Ingeniería en Sistemas, Electrónica e Industrial aplicando programación limpia y patrones de diseño basado en una metodología ágil de ingeniería de software.
- Integrar el trabajo al proyecto "Plataforma CLOUDIOT de Control y Monitoreo del Uso de Equipamiento y Programas Informáticos en Aulas y Laboratorios" aprobado por HCU según resolución 2496-CU-P-2015.

CAPÍTULO 2

Marco Teórico

2.1. Antecendentes Investigativos

Un estudio realizado por la Universidad Henri Poincaré, en el año 2013, [5], a través del refinamiento de un sistema médico, que consiste en un Marcapasos Cardiaco. Se aplicaron métricas de programación limpia como una alternativa para asegurar la calidad y corrección de sistemas altamente críticos. Como contribución, el estudio proporcionó una metodología para el desarrollo de sistemas médicos de alta confianza, el refinamiento de estos sistemas proporciona la capacidad de dividir los bloques funcionales del sistema en bloques más simples sin cambiar el comportamiento del sistema, además de poder montar nuevos componentes de una manera rentable.

En el año 2014 en el Departamento de Electrónica de la Universidad de Bharathiar, India, en la investigación [6], se desarrolló una técnica para la captura de una imagen en un sistema embebido basado en Raspberry Pi. El proyecto consiste en la implementación de una cámara NoIR compatible con Raspberry Pi para realizar la captura de una imagen, la cual es almacenada en una base de datos, para su comparación e identificación utilizando algoritmos de reconocimiento facial.

En 2015, se desarrolló el proyecto [7], en el cual desarrollaron un sistema para conectar una puerta al internet. Se utilizó una cámara USB y un sensor PIR para la detección de personas trabajando en conjunto con un dispositivo embebido Raspberry Pi. El Raspberry Pi fue equipado con un módem USB 3G para una conexión inalámbrica.

En Ecuador se realizaron varios estudios que mencionan la importancia de los sistemas de control en una institución, además de las ventajas de utilizar patrones de diseño en el desarrollo de software citados a continuación.

En un análisis del funcionamiento y elaboración de Patrones de Diseño para aplicaciones Web [8], indicó que el uso de Patrones de Diseño permite aumentar la calidad del software, basándose en la experiencia de la resolución de problemas ya conocidos y reduciendo tiempos. Para la obtención de resultados óptimos es importante determinar el patrón a utilizar para resolver un determinado

problema.
En la investigación [9], detalla que la implementación de patrones de diseño en una aplicación web cuenta con varias ventajas, en lo que se refiere al costo y tiempo de desarrollo del proyecto. Centrándose en el patrón de diseño Modelo Vista Controlador, haciendo fácil el mantenimiento e implementación de nuevos requerimientos.
La investigación [10], se llegó a una conclusión que, en el diseño de aplicaciones, es fundamental el anticiparse a nuevos requisitos y cambios, sin correr riesgos de pérdida de recursos y tiempo de desarrollo. Es necesario la aplicación de Patrones de Diseño, brindando así una mayor flexibilidad a las aplicaciones.
A partir del desarrollo de un sistema de control de acceso a laboratorios [11], se concluyó que la migración de un sistema manual de acceso es de vital importancia, gracias a lo cual se puede tener control de entradas y salidas del personal completamente automatizados. Fue utilizada la tecnología NFC, ofreciendo un alto nivel de seguridad de acceso de estudiantes, maestros y personal administrativo a los laboratorios.

2.2. Fundamentación teórica

2.2.1. Biometría

La biometría es considerada como un método que tiene como objetivo el medir ciertas características del cuerpo humano con el fin de identificar a una persona de otra. Para esto se debe elegir una característica dotada de una fuerte variabilidad de un individuo a otro [12]. Dentro de los principales características humanas utilizadas en la biometría se encuentran:

- Cara
- Huellas dactilares
- Geometría de la mano
- Iris del ojo
- Voz
- Radiografía dental
- ADN.

2.2.2. Sistemas Embebidos

Un sistema embebido es un sistema basado en uno o varios microprocesadores diseñado para realizar determinadas funciones, son programados previamente para realizar ciertas tareas, el usuario final puede interactuar con él, pero no podrá cambiar su función principal [3]. El término embebido o empotrado hace referencia al hecho que la electrónica o el sistema electrónico de control es una parte fundamental del sistema en que se encuentra.

2.2.3. Dispositivos Embebidos

Un dispositivo embebido está integrado por circuitos integrados programables, memoria flash o ROM y un software embebido. El software se utiliza para controlar los elementos electrónicos y usualmente se ejecuta sobre un microprocesador, un microcontrolador, un procesador digital de señal (DSP), un controlador lógico programable (PLC) y a veces en una computadora de propósitos generales adaptada para fines específicos [13].

2.2.4. Internet of Things (IoT)

El Internet de las Cosas (IoT) consiste en la integración de sensores y dispositivos conectados a Internet a través de una red. Los sensores y dispositivos son fácilmente integrables en hogares, entornos de trabajo y lugares públicos debido a su tamaño y costo. Estos dispositivos permiten medir desde la temperatura en una habitación hasta el tráfico vehicular en una ciudad.

2.2.5. Programación Limpia

La programación limpia se describe como el arte para escribir código limpio se debe contar con varias características como:
Conocimiento
Disciplina
Sentido del código.
Actitudes que nos permiten modificar el código, transformarlo de código malo, código con errores y fallos a código limpio.
La lógica del código debe ser fácil de seguir para hacer que los bugs (errores de programación) sean difícil de esconder. Las dependencias deben ser mínimas para un fácil mantenimiento. El manejo de errores debe ser completo de acuerdo a una articulada estrategia. El rendimiento debe estar cerca del óptimo para evitar que

otras personas ensucien el código buscando posibles mejoras de rendimiento. El código limpio hace una sola cosa y la hace bien. (Bjarne Stroustrup).

2.2.5.1. Nombrado Correcto

El nombrado es sumamente importante en la programación, nombres para describir a estructuras, variables, funciones y métodos deben cumplir una serie de características que nos ayuden a que mediante un nombrado correcto el código sea más limpio.

2.2.5.2. Reglas de Nombramiento

Las clases y objetos deben tener un sustantivo o una frase sustantiva como nombre. Evitar la utilización de nombres ambiguos.
El nombre de una clase debe describir correctamente su responsabilidad. Las funciones deben tener verbos o frases verbales como nombre. Evitar el uso de prefijos para el nombramiento de variables. No usar una palabra para dos propósitos.
Utilizar nombres técnicos.

2.2.6. Patrones de Diseño

El termino patrón fue definido por primera vez en el libro A Pattern Language "Cada patrón describe un problema que ocurre una y otra vez en nuestro entorno, para describir después el núcleo de la solución a ese problema, de tal manera que esa solución pueda
ser usada más de un millón de veces sin hacerlo ni siquiera dos veces de la misma forma." [14]. En el proceso de desarrollo de software un patrón es una solución a una serie de problemas que se repite de manera recurrente.
Estas soluciones no son necesariamente líneas de código, ni modelos específicos, sino más bien es una descripción general de cómo resolver un problema [9, 10].
Los patrones de diseño de software generalmente se clasifican en:
Patrones de diseño Creacionales.
Patrones de diseño Estructurales.
Patrones de diseño de Comportamiento.

2.2.6.1. Patrones Creacionales

Se encuentran relacionados con el proceso de creación de clases y objetos, encargados de abstraer el proceso de instanciación o creación de objetos, ayudan

a que el sistema sea independiente de cómo sus objetos son creados, integrados y representados [15].
Abstract Factory.- Proporciona un interfaz para crear o trabajar con familias de objetos, de manera que sea transparente el tipo de familia concreta que se esté utilizando.
Builder.- Abstrae el proceso de creación de un objeto complejo, de forma que el mismo proceso pueda crear diferentes representaciones.
Factory Method.- Define una interfaz centralizando la creación de objetos, haciendo que las subclases sean quienes decidan que objeto instanciar.
Prototype.- Crea nuevos objetos clonandolos de una instancia prototipada ya existente.
Singleton.- Garantiza que una clase tendrá una única instancia y proporciona un punto de acceso global a ella.

2.2.6.2. Patrones Estructurales

Tratan de la composición de clases y objetos, se ocupan de cómo las clases y objetos se agrupan, para formar estructuras más grandes [15]. Los patrones en esta categoría se encargan de lograr que los cambios en los requisitos de la aplicación no ocasionen cambios en las relaciones entre los objetos [16].
Adapter.- Convierte una interfaz, y la adapta según requerimientos para que pueda ser utilizada por una clase que de otra forma no lo haría.
Bridge.- Desacopla o desvincula una abstracción de su implementación de manera que cada una de ellas pueda variar de forma independiente.
Composite.- Permite combinar objetos compuestos en estructuras para tratarlos como simples.
Decorator.- Añade dinámicamente nuevas funcionalidades a una clase, proporcionando una mayor flexibilidad.
Facade.- Proporciona una interfaz unificada simple para poder acceder una interfaz o conjunto de interfaces de un subsistema.
Flyweight.- Reduce la redundancia cuando gran cantidad de objetos poseen idéntica información.
Proxy.- Proporciona un representante de un objeto para controlar el acceso a este.

2.2.6.3. Patrones Comportamentales

Caracterizan las formas en que las clases o los objetos interactúan y distribuyen la responsabilidad. Son los encargados de las opciones de comportamiento de la aplicación, permitiendo que el comportamiento varíe en tiempo de ejecución, sin

estos patrones cada comportamiento tendría que diseñarse e implementarse por separado [15].
Chain of Responsibility.- Permite establecer la línea que deben llevar los mensajes para que los objetos realicen la tarea indicada.
Command.- Encapsula una operación o petición en un objeto, permitiendo su ejecución sin la necesidad de conocer su contenido.
Interpreter.- Dado un lenguaje, define una gramática para dicho lenguaje, así como las herramientas necesarias para interpretarlo.
Iterator.- Permite realizar recorridos sobre objetos compuestos independientemente de la implementación de estos.
Mediator.- Define in objeto que coordine la comunicación entre objetos de distintas clases, pero que funcionan como un conjunto.
Memento.- Permite volver a estados anteriores del sistema.
Observer.- Define una dependencia de uno a muchos entre objetos, de forma que cuando un objeto cambie de estado se notifique y actualicen automáticamente todos los objetos que dependen de él.
State.- Permite que un objeto modifique su comportamiento cada vez que cambie su estado interno.
Strategy.- Permite disponer de varios métodos para resolver un problema y elegir cual utilizar en tiempo de ejecución.
Template Method.- Define en una operación el esqueleto de un algoritmo, delegando en las subclases algunos de los pasos, esto permite que las subclases redefinan ciertos pasos de un algoritmo sin cambiar su estructura.
Visitor.- Permite definir nuevas operaciones sobre una jerarquía de clases sin codificar las clases sobre las que opera.

2.2.7. Java

2.2.7.1. Plataforma J2EE

La plataforma Java EE está destinada a desarrollar aplicaciones empresariales distribuidas, con una arquitectura multicapa, escritas en el lenguaje de programación Java y que se ejecutan en un servidor de aplicaciones como Tomcat, weblogic, jboss, etc.
Es un conjunto de especificaciones que permiten soluciones para el desarrollo, despliegue y gestión de aplicaciones multicapa centradas en servidor.

2.2.7.2. Especificación

Una especificación no es más que el detalle de cada una de las tecnologías dentro de la plataforma Java EE. Un conjunto de reglas que dictan como debe desarrollarse ese producto de tal forma que se pueda garantizar que una aplicación desarrollada siguiendo las especificaciones de Java EE pueda desplegarse y ejecutarse.
Tecnologías de la plataforma Java EE Enterprise JavaBeans (EJB).
Java Servlet
JavaServer Page (JSP)
JavaServer Pages Standard Tag Library (JSTL).
JavaServer Faces (JSF)
Java Message Service (JMS).
Java Transaction API (JTA).
JavaMail API y JavaBeans Activation Framework (JAF).
Tecnologías XML (JAXP, JAX-RPC, JAX-WS, JAXB, SAAJ, JAXR)
JPA, JDBC API
Java Naming and Directory Interface (JNDI)
Java Authentication and Authorization Service (JAAS)

2.2.7.3. Patrones de Diseño J2EE

Para el desarrollo de aplicaciones J2EE de una forma técnica siguiendo un proceso de desarrollo normalizado, tales como:
Proceso Unificado.- Es uno de los procesos más reconocidos definido por Jacobson, Booch y Rumbaugh. Este proceso se dice que es "dirigido por casos de uso, centrado en la arquitectura, iterativo e incremental." [17].

2.2.8. Hardware Libre

Aquel hardware cuyo diseño se hace disponible públicamente para que cualquier persona lo pueda estudiar, modificar, distribuir, materializar y vender, tanto el original como otros objetos basados en ese diseño. El Hardware Libre nos ofrece un sin número de ventajas, tales como: La posibilidad de controlar la tecnología mediante la utilización de elementos de calidad, estandarizados, infraestructura abierta, contenido y herramientas no restringidas compartiendo así el conocimiento libremente [18].

2.2.9. Metodologías ágiles

El desarrollo ágil de software refiere a métodos de Ingeniería del Software basados en el desarrollo iterativo e incremental, estas metodologías son imprescindibles en un mundo en el que nos exponemos a cambios recurrentemente. Siempre hay que tener en cuenta como programadores que lo que es la última tendencia hoy puede que no exista mañana y por esto existe la metodología ágil donde los requisitos y soluciones evolucionan mediante la colaboración de grupos auto organizados y multidisciplinarios [19].
De entre todos los métodos de desarrollo ágil, estos son los 2 que actualmente dominan el panorama:

2.2.9.1. SCRUM

Scrum es un proceso de la Metodología Ágil que se usa para minimizar los riesgos durante la realización de un proyecto, pero de manera colaborativa.
Entre las ventajas se encuentran la productividad, calidad y que se realiza un seguimiento diario de los avances del proyecto, logrando que los integrantes estén unidos, comunicados y que el cliente vaya viendo los avances. La profundidad de las tareas que se asignan en SCRUM tiende a ser incremental, caso que coincide exactamente con el devenir normal de un desarrollo [19].

2.2.9.2. XP o Extreme Programming

La "programación extrema" es un proceso de la Metodología Ágil que se aplica en equipos con muy pocos programadores quienes llevan muy pocos procesos en paralelo. Consiste entonces en diseñar, implementar y programar lo más rápido posible, hasta en casos se recomienda saltar la documentación y los procedimientos tradicionales. Se fundamenta en la capacidad del equipo para comunicarse entre sí y las ganas de aprender de los errores propios inherentes en un programador [19].
La gran ventaja de XP es su increíble capacidad de respuesta ante imprevistos, aunque por diseño es una metodología que no construye para el largo plazo y para la cual es difícil documentar.

2.2.9.3. Crystal

La filosofía de Crystal define el desarrollo como un juego cooperativo de invención y comunicación cuya meta principal es entregar software útil, que funcione, y su objetivo secundario, preparar el próximo juego.

Existen dos reglas que aplican para toda la familia Crystal. La primera indica que los ciclos donde se crean los incrementos no deben exceder cuatro meses; la segunda, que es necesario realizar un taller de reflexión después de cada entrega para afinar la metodología [20].

2.2.9.4. Desarrollo orientado a funcionalidades (Feature-Driven Development - FDD)

FDD tiene como rasgo característico la planeación y el diseño por adelantado. En consecuencia, el modelo de objetos, la lista de características y la planeación se hacen al inicio del proyecto. Las iteraciones son incrementos con características identificadas.
Su ciclo de vida está compuesto por cinco etapas: el desarrollo de un modelo general, la construcción de la lista de características, la planeación por característica, el diseño por característica y la construcción por característica [20].

2.3. Propuesta de Solución

A través del desarrollo del Sistema Embebido de Control de Ingreso a Aulas y Laboratorios de la Facultad de Ingeniería en Sistemas, Electrónica e Industrial se busca principalmente automatizar la asignación de laboratorios al personal docente y el proceso de préstamos de aulas y laboratorios a estudiantes y personal administrativo de la facultad, generando un registro de ingresos y salidas que permitirá su administración y control.

CAPÍTULO 3

Metodología

3.1. Modalidad Básica de la investigación

La realización del presente proyecto de investigación se basará en la investigación de campo, la cual analiza el problema partiendo de hechos reales para la obtención de información y requerimientos que evidencien los objetivos.
La investigación bibliográfica se realizará mediante el uso de libros e internet, artículos científicos, publicaciones, tesis; a través de la cual se obtendrá información que ayude en todo lo que concierne al desarrollo de la fundamentación teórica en la cual se basó la investigación.
La investigación aplicada se realizará al aplicar los conocimientos adquiridos a lo largo de la carrera para la elaboración del presente proyecto de tesis.

3.2. Población y muestra

La presente investigación por su característica no requiere población y muestra.

3.3. Recolección de información

Partiendo de que el objetivo de la investigación, es el desarrollo de un Sistema para el control de acceso a aulas y laboratorios en la Facultad de Ingeniería en Sistemas, Electrónica e Industrial, se realizará una investigación exhaustiva a través de la recopilación bibliográfica en documentos tales como: artículos científicos, libros y tesis correspondientes al tema, los mismos que nos proporcionarán la información necesaria para la estructuración del presente tema de investigación.
Además, se recolectará información proporcionada por el personal de la Facultad con el fin de obtener información útil para la investigación.

3.4. Procesamiento y análisis de datos

Una vez recolectada la información se procederá a su respectivo análisis obteniendo resultados satisfactorios los cuales serán de gran importancia para la formulación de la propuesta. Los datos serán analizados y procesados en relación

al problema para poder establecer las respectivas conclusiones asegurando que los datos sean lo más reales posibles.

3.5. Desarrollo del Proyecto

1. Evaluación y revisión de las características que debe cumplir el código de programación para que sea considerado limpio.

2. Análisis comparativo entre los patrones de diseño de software más empleados en el proceso de desarrollo de software.

3. Definición de la metodología de desarrollo de software a utilizar.

4. Desarrollo del aplicativo Embebido IoT para el Control de Ingreso a Aulas y Laboratorios de la Facultad de Ingeniería en Sistemas, Electrónica e Industrial aplicando reglas de programación limpia.
   - Análisis de los procesos para el control de ingreso de aulas y laboratorios de la Facultad de Ingeniería en Sistemas, Electrónica e Industrial.
   - Diseño del aplicativo Embebido IoT para el Control de Ingreso a Aulas y Laboratorios de la Facultad de Ingeniería en Sistemas, Electrónica e Industrial aplicando reglas de programación limpia.
   - Evaluación del funcionamiento y control de pruebas para la validación del Sistema.

5. Desarrollo del Sistema Web para la administración de los datos generados por el aplicativo Embebido IoT.
   - Proceso de Análisis de requerimientos para la administración del aplicativo.
   - Diseño del Sistema Web, aplicando patrones de diseño de software.
   - Evaluación del funcionamiento y control de pruebas.

6. Integración del Sistema Embebido IoT para el Control de Ingreso a Aulas y Laboratorios de la Facultad de Ingeniería en Sistemas, Electrónica e Industrial al proyecto “Plataforma CLOUDIOT de Control y Monitoreo del Uso de Equipamiento y Programas Informáticos en Aulas y Laboratorios”.

CAPÍTULO 4

Desarrollo de la propuesta

El presente capítulo se enfoca en el desarrollo del sistema de control, a partir de una evaluación y determinación de estándares de programación limpia y patrones de diseño que fueron aplicados al sistema propuesto. La metodología de desarrollo de software que se utilizó fue seleccionada a través de un estudio comparativo entre varias metodologías ágiles de desarrollo de software.

4.1. Estándares de Programación Limpia

> "Cualquier persona puede escribir código que una computadora pueda entender. Buenos programadores escriben código que los humanos pueden entender" - Martin Fowler

Código limpio.- es código legible y funcional. El código limpio es una percepción del lector en base a su experiencia.
Los programadores no escriben código limpio. Los programadores experimentados pueden mirar el código fuente de un programa y afirmar si es legible o no, además de desarrollar una opinión, de si el código es eficiente, se encuentra bien estructurado e incluso si es simple [21].
Según la investigación realizada por la Corporación de Investigación ABB de Suiza, 2015 [22], a través de un análisis del rendimiento del código de programación de sistemas informáticos desarrollados en el ámbito académico en diferentes lenguajes de programación como Java, .Net y C, además de la investigación realizada en Canadá por la Universidad de Tecnología de Holanda en el año 2015 [23], en la cual se habla acerca de los malos hábitos que existen en el desarrollo de software. A través de una refactorización del código de programación aplicando reglas de programación limpia se buscó mejorar el rendimiento de varios sistemas informáticos.
Se ha realizado una síntesis de las experiencias de investigación, que muestran un análisis de los beneficios de la utilización de reglas de programación limpia en el desarrollo de software, reflejadas en las tablas 1, 2 y 3.

Tabla 1: Reglas de Programación Limpia (a).

Fuente: Autor.

	Reglas	Propósito	Beneficios
Nombrado	Nombres que revelen las intenciones. Evitar la desinformación. Distinciones con sentido. Nombres que se puedan pronunciar. Nombres que se puedan buscar.	El nombre de una variable, función o clase debe responder a las preguntas: - ¿Por qué existe? - ¿Qué hace? - ¿Cómo se usa? Facilita la comprensión y la modificación del código. Es correcta la utilización de prefijos, mientras la distinción entre las variables tenga sentido. Las constantes numéricas y variables de una sola letra son difíciles de localizar en el código. El nombre de los métodos debe ser un verbo, el de las clases no.	Facilita la ayuda del IDE de programación que se utiliza. Mejora la legibilidad del código de programación
Funciones	Tamaño reducido. Una sola tarea. Responsabilidad única. Nombres descriptivos.	Las funciones deben ser breves, con una longitud aproximada de 20 líneas de código. Los bloques en instrucciones if – else, while, etc. Deben tener aproximadamente una línea de longitud, generalmente la invocación de una función. Las funciones deben hacer una cosa. Deben hacerlo bien y debe ser lo único que hagan. Las funciones que realizan más de una tarea se las divide en secciones. Las funciones deben tener aproximadamente 2 argumentos, evitar el uso de un número mayor de argumentos.	La correcta aplicación de estas reglas permitirá la correcta creación de funciones. Las funciones serán breves, con nombres correctos y bien organizadas.

Tabla 2: Reglas de Programación Limpia (b).

Fuente: Autor.

	Reglas	Propósito	Beneficios
Comentarios	Comentarios de calidad. Comentarios legales. Comentarios informativos. Explicar la intención del comentario. Advertencia de consecuencias del código.	Por estándares corporativos de creación de código muchas veces nos obligan a crear comentarios por motivos legales. Comentarios que proporcionan información básica son útiles. Los comentarios deben ser actualizados, relevantes y precisos. No usar comentarios si se puede usar una función o una variable.	Los comentarios permiten al código ser más expresivo y mostrar sus intenciones. Los comentarios son alternativas que tiene el desarrollador para expresarse en el código de programación. El código claro y simple con muy pocos comentarios es muy superior al código complejo con multitud de comentarios.
Estructuras de datos	Abstracción de los datos. Cumplir con la ley de Demeter. Ocultar la estructura.	El objetivo de la abstracción de datos no es mostrar los detalles de los datos, sino expresarlos en términos abstractos. Un objeto no debe mostrar su estructura interna.	El control de objetos permite mostrar el comportamiento de los mismos, pero ocultar sus datos, de esta manera facilita la inserción de nuevos tipos de objetos sin la necesidad de cambiar los comportamientos existentes. Brinda flexibilidad al código de programación permitiendo añadir nuevos tipos de datos o en otros casos añadir nuevos comportamientos.

Tabla 3: Reglas de Programación Limpia (c).

Fuente: Autor.

	Reglas	Propósito	Beneficios
Procesamiento de errores	Uso de excepciones. Instrucción Try – Catch – Finally. Redacción de mensajes de error informativos.	Tratar de forma independiente la lógica de negocio del control de errores, de esta manera el código de programación es mejor. Un método no debe devolver null, mucho menos recibir como parámetro null.	El control de errores brinda la posibilidad de crear código limpio que sea legible y al mismo tiempo robusto. Aumenta la capacidad de mantenimiento del código de programación.
Pruebas de unidad	Cumplimiento de las tres leyes del DGP – Desarrollo Guiado por Pruebas. Realizar pruebas Limpias. Utilización de un solo estándar. Cumplimiento de las reglas F.I.R.S.T.	El cumplimiento de las leyes del DGP conlleva a la creación de las pruebas y el código de producción en forma conjunta. Las pruebas deben ser limpias, de esta manera se garantiza el control del funcionamiento del código. Las pruebas de unidad deben ser legibles, claras y simples, con el menor número de expresiones posibles. Cada función de prueba debe probar un único concepto, de esta manera se evita funciones de pruebas extensas. La aplicación de las reglas F.I.R.S.T. proporciona las siguientes características a las pruebas de unidad: - Rapidez. - Independencia. - Repetición. - Validación Automática. - Puntualidad.	Las pruebas de unidad comprenden un proceso de vital importancia en el desarrollo limpio, brindado al código flexibilidad, mantenibilidad y reutilización.

4.2. Patrones de Diseño de Software

En base al análisis de expertos realizado en la investigación [24], se determinó los patrones de diseño más utilizados en el desarrollo de aplicaciones web, tabla 4.

Tabla 4: Patrones GOF empleados en aplicaciones Web.

Fuente: Clean Code: A Handbook of Agile Software Craftsmanship, [24].

Categoría GOF	Patrón de Diseño
Creacionales	1. Abstract Factory 2. Builder 3. Factory Method 4. Singleton
Estructurales	1. Decorator 2. Facade
De Comportamiento	1. Iterator 2. Observer 3. Strategy 4. Template Method

En este contexto, es importante conocer el propósito, la aplicabilidad y los beneficios que brindan estos patrones de diseño al software. Esto se lo realizó a través de un análisis comparativo, permitiendo de esta forma la selección de los patrones de diseño que se aplicaron en el desarrollo según sus características,

tablas 5 y 6.

Tabla 5: Tabla Comparativa Patrones de Diseño (a).

Fuente: Autor.

Nombre	Propósito	Aplicabilidad	Beneficios
Abstract Factory	Proporciona una interfaz para crear familias de objetos relacionados o dependientes sin especificar sus clases concretas.	1.- Cuando un sistema debe ser independiente de como los objetos son creados, de cómo estas compuestos y representados. 2.- Cuando se desea proporcionar una biblioteca de clases de objetos, y se desea habilitar solo sus interfaces, no sus implementaciones.	1.- Controla las clases de objetos que crea una aplicación. Aísla las clases de implementación, permitiendo la manipulación solo de sus interfaces abstractas. 2.- Cuando los objetos de una de una sola familia están diseñados para trabajar conjuntamente, Abstract Factory controla el uso de objetos de una sola familia a la vez.
Builder	Abstrae el proceso de creación de un objeto complejo, de forma que el mismo proceso pueda crear diferentes representaciones	1.- Cuando el algoritmo para crear un objeto complejo debe ser independiente de las partes que lo componen y su ensamblado. 2.- Cuando el objeto que se construye debe permitir diferentes representaciones.	1.- Mejora la modularidad de la aplicación, encapsulando la forma en la que se construye un objeto complejo. El código se escribe una sola vez, y se reutiliza para construir variantes del objeto. 2.- Permite realizar un control integro, paso a paso en el proceso de construcción de un objeto.
Factory Method	Define una interfaz para crear un objeto, pero permite a una subclase definir qué clase instanciar.	1.- Cuando una clase no puede anticipar la clase de objetos que desea crear. 2.- Cuando se desea que una subclase especifique los objetos que crea. 3.- Cuando se desea conocer responsabilidades específicas que una subclase heredó.	1.- Permite eliminar la necesidad de vincular clases específicas de una aplicación dentro del código. El código solo se ocupa de la interfaz del producto.
Singleton	Asegura que una clase sea instanciada una sola vez.	1.- Cuando debe haber exactamente una sola instancia de una clase, y esta debe ser accesible desde cualquier punto de acceso conocido. 2.- Cuando esa única instancia debe ser heredada por subclases y subprocesos sin la necesidad de modificar su código.	1.- Singleton encapsula una única instancia, de esta forma controla su acceso. 2.- Menor uso de espacios de memoria, evitando la utilización de variables globales para almacenar instancias. 3.- Mejora el proceso de configuración de una aplicación y su flexibilidad en tiempo de ejecución. 4.- Controla el número de instancias que utiliza una aplicación.
Decorator	Agrega responsabilidades adicionales a los objetos de forma dinámica, proporcionan flexibilidad a las subclases ampliando su funcionalidad.	1.- Añade responsabilidades a objetos de forma dinámica y transparente, sin afectar a otros objetos. 2.- Para revocar responsabilidades a objetos. 3.-	1.- El patrón de diseño Decorator proporciona una mayor flexibilidad en herencia, las responsabilidades pueden ser añadidas o revocadas en tiempo de ejecución, se puede definir una clase simple y agregar funcionalidad en forma incremental.

Tabla 6: Tabla Comparativa Patrones de Diseño (b).

Fuente: Autor.

Nombre	Propósito	Aplicabilidad	Beneficios
Facade	Define una interfaz superior unificada a un conjunto de interfaces en un subsistema.	1.- Cuando se desea proporcionar una interfaz sencilla a un subsistema complejo. Una fachada proporciona una vista más simple a un subsistema complejo, brindando al sistema independencia y portabilidad. 2.- Puede ser un punto de conexión entre subsistemas.	1.- Mejora la usabilidad del sistema. 2.- Reduce las dependencias de compilación, importante en sistemas de software de gran escala. Este patrón de diseño simplifica al sistema y lo hace más portable.
Iterator	Brinda una forma para acceder a objetos compuestos independientemente de su implementación.	1.- Cuando se desea acceder al contenido de un objeto compuesto sin exponer su implementación 2.- Cuando se desea una interfaz uniforme para diferentes estructuras compuestas	1.- Permite soportar la variación en el recorrido de una interfaz, facilitando el cambio del algoritmo simplemente reemplazando la instancia del iterador por otra diferente.
Observer	Define una dependencia de uno a muchos entre objetos, de forma que cuando un objeto cambie de estado se notifique y actualicen automáticamente todos los objetos que dependen de él.	1.- Cuando un cambio en un objeto depende o requiere el cambio de otros objetos. 2.- Para encapsular objetos dependientes en objetos separados, esto permite su variación y reutilización independientemente.	1.- Permite la variabilidad de las interfaces y objetos de forma independiente.
Strategy	Encapsula y define a un grupo de algoritmos. Este patrón permite la variación de los algoritmos independientemente de los clientes que lo utilicen.	1.- Cuando es necesario encapsular un algoritmo para que el cliente no conozca los datos que este utiliza. 2.- Es una forma de configurar una clase con uno de muchos comportamientos	1.- Es una alternativa a la herencia o subclases, de modo que permite la reutilización de clases facilitando el cambio, la comprensión y la extensión del código.
Template Method	Define en una operación el esqueleto de un algoritmo, delegando en las subclases algunos de los pasos, esto permite que las subclases redefinan ciertos pasos de un algoritmo sin cambiar su estructura.	1.- Cuando se debe factorizar o agrupar varias subclases con un comportamiento similar en una clase común, evitando la duplicidad de código. 2.- Permite definir métodos de plantilla que llame a operaciones en puntos específicos, de esta forma controla las extensiones de las subclases.	1.- Es un patrón de diseño fundamental para la reutilización de código, permitiendo descomponer el comportamiento común de las clases. 2.- Permite una estructura de control en la que una clase padre llama a las operaciones de una subclase. Controla el llamado de las operaciones.

Basado en la realidad del proyecto, y en las tablas comparativas de patrones de diseño 5 y 6 se consideraron criterios de valoración en función de las cualidades con las que el software propuesto debe contar. Los criterios a evaluar son:

- Usabilidad
- Escalabilidad
- Abstracción
- Simplificación de código
- Reutilización de código
- Transparencia de uso
- Encapsulamiento

Tabla 7: Criterios de Valoración Patrones de Diseño.

Fuente: Autor.

Patrón de Diseño / Criterio	Abstract Factory	Builder	Factory Method	Singleton	Decorator	Facade	Iterator	Observer	Strategy	Template Method
Usabilidad		x		x		x		x		x
Escalabilidad					x					
Abstracción	x	x					x		x	
Simplificación de código				x		x				
Reutilización de código		x	x		x			x	x	x
Transparencia de uso	x			x	x		x			
Encapsulamiento	x	x	x	x				x	x	
Total	**3**	**4**	**2**	**4**	**3**	**2**	**2**	**3**	**3**	**2**

Los patrones de diseño Builder y Singleton son los que mayor cualidades aportan al software según los criterios de evaluación de la tabla 7.

4.3. Definición de la metodología de desarrollo de software

Para selección de la metodología de desarrollo a utilizar, se consideraron criterios en función a los conocimientos que el equipo del trabajo tiene de las metodologías que se evaluaron. Los criterios a evaluar son:

- Tipo de metodología de desarrollo
- Grado de conocimiento
- Adaptable a cambios
- Documentación adecuada

- Proceso de desarrollo con iteraciones
- Tiempo corto en las iteraciones
- Equipo de trabajo pequeño
- Adaptable a proyectos a corto plazo.

Según la investigación [25], los autores establecen pesos para cada criterio de evaluación.

- 20 % para el Grado de conocimiento
- 15 % para Adaptable a cambios y documentación adecuada
- 10 % para los criterios considerados por el equipo de trabajo

Se determinó la metodología a utilizar, a través de la elaboración de una tabla comparativa, tabla 8.

Tabla 8: Tabla Comparativa de Metodologías Ágiles.

Fuente: Autor.

Criterio	%	Scrum	XP	Kanban
Tipo de metodología de desarrollo	10	10	10	10
Grado de conocimiento	20	10	15	5
Adaptable a cambios	15	10	12	10
Documentación adecuada	15	10	5	10
Iteraciones	10	10	10	10
Tiempo corto en las iteraciones	10	5	10	5
Equipo de trabajo pequeño	10	5	10	10
Adaptable a proyectos a corto plazo	10	5	10	10
Total	**100**	**65**	**82**	**70**

La metodología de desarrollo XP, según a los criterios de evaluación en función de los conocimientos del equipo de trabajo, se determino que la metodología Extreme Programming (XP) es la más apropiada a utilizar.

4.4. Metodología XP en el Desarrollo de Software

XP al ser parte de las metodologías de desarrollo ágil, se basa en iteraciones utilizadas para proyectos que cuentan con un equipo pequeño y cuyo plazo de entrega es en el menor tiempo posible. Esta metodología se basa en la simplicidad, la comunicación y la reutilización de código. [20, 26].
Roles XP

- Programador: Andrés Tisalema.
- Cliente: Uodide.
- Encargado de pruebas - Tester: Ing. Carlos Vargas.
- Encargado de seguimiento - Tracker: Ing. Jesús Guaman.
- Entrenador - Coach: Ing. Rubén Nogales.
- Gestor - Big boss: Ing. Renato Urvina.

Las fases de la Metodología XP son las siguientes:

- Exploración.
- Planificación de la entrega.
- Iteraciones.
- Producción.
- Pruebas.

4.5. Desarrollo del Sistema de Control basado en la Metodología Ágil XP

4.5.1. Fase 1. Exploración

En la fase de exploración se definen las historias de usuarios, además se analiza cada uno de los procesos y requerimientos que debe cumplir el sistema propuesto. Los procesos de acuerdo a cada uno de los actores que participan o interactúan con el sistema son los siguientes:

- Personal Laboratorista
    - Registro de Facultades.

- Registro de Carreras.
- Registro de Categorías.
- Registro del Personal.
- Registro de Asignaturas.
- Registro de Horarios.
- Registro de Jornadas.
- Registro de Dependencias.
- Registro de Dispositivos.
- Registro de Oficinas.
- Registro de Aulas y Laboratorios.
- Registro de Distributivos de Aulas y Laboratorios.
- Enrolamiento o registro de usuarios.
- Prestamos de aulas y laboratorios al personal de la Fisei.

▪ Personal Docente

- Proceso de validación de identidad.
- Registro de ingreso a aulas o laboratorios en horarios de clases.
- Registro de préstamo de aulas o laboratorios.

▪ Personal de Investigación

- Registro de préstamo de aulas o laboratorios.
- Registro de ingreso a oficinas de investigación.

▪ Personal Docente/Investigador

- Registro de ingreso a aulas o laboratorios en horarios de clases.
- Registro de préstamo de aulas o laboratorios.
- Registro de ingreso a oficinas de investigación.

▪ Personal Estudiantil

- Registro de préstamo de aulas o laboratorios.

Las historias de usuarios describen los procesos del sistema en 3 o 4 líneas de una forma general y en lenguaje no técnico. Las historias de usuarios reemplazan al documento de requerimientos y se interpretan como tareas.
Proceso: Personal Laboratorista

- Registro de Facultades.

Tabla 9: Historia de Usuario: Registro de Facultades.

Fuente: Autor.

Historia de Usuario	
Número: 1	**Usuario:** Laboratorista
Nombre historia: Registro de Facultades	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Media
Iteración asignada: 1	
Programador responsable: Andrés Tisalema	
Descripción: Debe permitir el registro de los datos pertenecientes a las facultades, datos como: Código, nombre y descripción de la facultad.	
Observaciones: Los datos del usuario deben ser almacenados en la nube.	

Tabla 10: Criterios de Aceptación: Registro de Facultades.

Fuente: Autor.

Criterios de Aceptación	
Dado:	Soy un usuario de tipo laboratorista
Cuando:	Realizo el registro de los datos pertenecientes a una facultad y presiono el botón guardar.
Entonces:	Los datos serán almacenados en la base de datos en la nube.

Criterios de Aceptación	
Dado:	Soy un usuario de tipo laboratorista
Cuando:	Realizo cambios a los datos pertenecientes a una facultad y presiono el botón guardar.
Entonces:	Los cambios se actualizarán en la base de datos.

Criterios de Aceptación	
Dado:	Soy un usuario de tipo laboratorista
Cuando:	Presiono el botón guardar en un registro perteneciente a una facultad
Entonces:	Los datos de la facultad serán eliminados de la base de datos.

- Registro de Carreras.

Tabla 11: Historia de Usuario: Registro de Carreras.

Fuente: Autor.

Historia de Usuario	
Número: 2	**Usuario:** Laboratorista
Nombre historia: Registro de Carreras	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Media
Iteración asignada: 1	
Programador responsable: Andrés Tisalema	
Descripción: Debe permitir el registro de los datos pertenecientes a las carreras. Los datos que se solicitarán son: Código, nombre y descripción de la carrera, además de la facultad a la que pertenece.	
Observaciones: Los datos del usuario deben ser almacenados en la nube.	

Tabla 12: Criterios de Aceptación: Registro de Carreras.

Fuente: Autor.

Criterios de Aceptación	
Dado:	Soy un usuario de tipo laboratorista
Cuando:	Realizo el registro de los datos pertenecientes a una carrera y presiono el botón guardar.
Entonces:	Los datos serán almacenados en la base de datos en la nube.

Criterios de Aceptación	
Dado:	Soy un usuario de tipo laboratorista
Cuando:	Realizo cambios a los datos pertenecientes a una carrera y presiono el botón guardar.
Entonces:	Los cambios se actualizarán en la base de datos.

Criterios de Aceptación	
Dado:	Soy un usuario de tipo laboratorista
Cuando:	Presiono el botón guardar en un registro perteneciente a una carrera
Entonces:	Los datos de la carrera serán eliminados de la base de datos.

- Registro de Categorías.

Tabla 13: Historia de Usuario: Registro de Categorías.

Fuente: Autor.

Historia de Usuario	
Número: 3	**Usuario:** Laboratorista
Nombre historia: Registro de Categorías	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Media
Iteración asignada: 1	
Programador responsable: Andrés Tisalema	
Descripción: Debe permitir el registro de los datos pertenecientes a las categorías del personal. Los datos que se solicitarán son: Código, nombre, y descripción de la categoría.	
Observaciones: Los datos del usuario deben ser almacenados en la nube.	

Tabla 14: Criterios de Aceptación: Registro de Categorías.

Fuente: Autor.

Criterios de Aceptación	
Dado:	Soy un usuario de tipo laboratorista
Cuando:	Realizo el registro de los datos pertenecientes a una categoría del personal y presiono el botón guardar.
Entonces:	Los datos serán almacenados en la base de datos en la nube.

Criterios de Aceptación	
Dado:	Soy un usuario de tipo laboratorista
Cuando:	Realizo cambios a los datos pertenecientes a una categoría del personal y presiono el botón guardar.
Entonces:	Los cambios se actualizarán en la base de datos.

Criterios de Aceptación	
Dado:	Soy un usuario de tipo laboratorista
Cuando:	Presiono el botón guardar en un registro perteneciente a una categoría del personal
Entonces:	Los datos de la categoría del personal serán eliminados de la base de datos.

- Registro del Personal.

Tabla 15: Historia de Usuario: Registro del Personal.

Fuente: Autor.

Historia de Usuario	
Número: 4	**Usuario:** Laboratorista
Nombre historia: Registro del Personal	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Media
Iteración asignada: 1	
Programador responsable: Andrés Tisalema	
Descripción: Debe permitir el registro de los datos pertenecientes al personal, datos como: Código, cédula de identidad, nombre, título de tercer nivel, tipo de personal y la carrera a la que pertenece.	
Observaciones: Los datos del usuario deben ser almacenados en la nube.	

Tabla 16: Criterios de Aceptación: Registro del Personal.

Fuente: Autor.

Criterios de Aceptación	
Dado:	Soy un usuario de tipo laboratorista
Cuando:	Realizo el registro de los datos pertenecientes al personal y presiono el botón guardar.
Entonces:	Los datos serán almacenados en la base de datos en la nube.

Criterios de Aceptación	
Dado:	Soy un usuario de tipo laboratorista
Cuando:	Realizo cambios a los datos pertenecientes al personal y presiono el botón guardar.
Entonces:	Los cambios se actualizarán en la base de datos.

Criterios de Aceptación	
Dado:	Soy un usuario de tipo laboratorista
Cuando:	Presiono el botón guardar en un registro perteneciente al personal
Entonces:	Los datos del personal serán eliminados de la base de datos.

- Registro de Asignaturas.

Tabla 17: Historia de Usuario: Registro de Asignaturas.

Fuente: Autor.

Historia de Usuario	
Número: 5	**Usuario:** Laboratorista
Nombre historia: Registro de Asignaturas	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Media
Iteración asignada: 1	
Programador responsable: Andrés Tisalema	
Descripción: Debe permitir el registro de los datos pertenecientes a las asignaturas. Los datos que se solicitarán son: Código, nombre, y la carrera a la que pertenece.	
Observaciones: Los datos del usuario deben ser almacenados en la nube.	

Tabla 18: Criterios de Aceptación: Registro de Asignaturas.

Fuente: Autor.

Criterios de Aceptación	
Dado:	Soy un usuario de tipo laboratorista
Cuando:	Realizo el registro de los datos pertenecientes a una asignatura y presiono el botón guardar.
Entonces:	Los datos serán almacenados en la base de datos en la nube.

Criterios de Aceptación	
Dado:	Soy un usuario de tipo laboratorista
Cuando:	Realizo cambios a los datos pertenecientes a una asignatura y presiono el botón guardar.
Entonces:	Los cambios se actualizarán en la base de datos.

Criterios de Aceptación	
Dado:	Soy un usuario de tipo laboratorista
Cuando:	Presiono el botón guardar en un registro perteneciente a una asignatura
Entonces:	Los datos de la asignatura serán eliminados de la base de datos.

- Registro de Horarios.

Tabla 19: Historia de Usuario: Registro de Horarios.

Fuente: Autor.

Historia de Usuario	
Número: 6	**Usuario:** Laboratorista
Nombre historia: Registro de Horarios	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Media
Iteración asignada: 1	
Programador responsable: Andrés Tisalema	
Descripción: Debe permitir el registro de los datos pertenecientes a los horarios del personal docente y docente/investigador. Los datos que se solicitarán son: Código, el día del horario, hora de inicio, hora de finalización, la asignatura y el docente o docente/investigador al cual pertenece el horario.	
Observaciones: Los datos del usuario deben ser almacenados en la nube.	

Tabla 20: Criterios de Aceptación: Registro de Horarios.

Fuente: Autor.

Criterios de Aceptación	
Dado:	Soy un usuario de tipo laboratorista
Cuando:	Realizo el registro de los datos pertenecientes a un horario y presiono el botón guardar.
Entonces:	Los datos serán almacenados en la base de datos en la nube.

Criterios de Aceptación	
Dado:	Soy un usuario de tipo laboratorista
Cuando:	Realizo cambios a los datos pertenecientes a un horario y presiono el botón guardar.
Entonces:	Los cambios se actualizarán en la base de datos.

Criterios de Aceptación	
Dado:	Soy un usuario de tipo laboratorista
Cuando:	Presiono el botón guardar en un registro perteneciente a un horario
Entonces:	Los datos del horario serán eliminados de la base de datos.

- Registro de Jornadas.

Tabla 21: Historia de Usuario: Registro de Jornadas.

Fuente: Autor.

Historia de Usuario	
Número: 7	**Usuario:** Laboratorista
Nombre historia: Registro de Jornadas	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Media
Iteración asignada: 1	
Programador responsable: Andrés Tisalema	
Descripción: Debe permitir el registro de los datos pertenecientes a las jornadas del personal de investigación. Los datos que se solicitarán son: Código, el día de la jornada, el número de horas y el investigador al cual pertenece la jornada.	
Observaciones: Los datos del usuario deben ser almacenados en la nube.	

Tabla 22: Criterios de Aceptación: Registro de Jornadas.

Fuente: Autor.

Criterios de Aceptación	
Dado:	Soy un usuario de tipo laboratorista
Cuando:	Realizo el registro de los datos pertenecientes a una jornada y presiono el botón guardar.
Entonces:	Los datos serán almacenados en la base de datos en la nube.

Criterios de Aceptación	
Dado:	Soy un usuario de tipo laboratorista
Cuando:	Realizo cambios a los datos pertenecientes a una jornada y presiono el botón guardar.
Entonces:	Los cambios se actualizarán en la base de datos.

Criterios de Aceptación	
Dado:	Soy un usuario de tipo laboratorista
Cuando:	Presiono el botón guardar en un registro perteneciente a una jornada
Entonces:	Los datos de la jornada serán eliminados de la base de datos.

- Registro de Dependencias.

Tabla 23: Historia de Usuario: Registro de Dependencias.

Fuente: Autor.

Historia de Usuario	
Número: 8	**Usuario:** Laboratorista
Nombre historia: Registro de Dependencias	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Media
Iteración asignada: 1	
Programador responsable: Andrés Tisalema	
Descripción: Debe permitir el registro de los datos pertenecientes a las dependencias. Los datos que se solicitarán son: Código, nombre, ubicación y la carrera a la cual pertenece la dependencia.	
Observaciones: Los datos del usuario deben ser almacenados en la nube.	

Tabla 24: Criterios de Aceptación: Registro de Dependencias.

Fuente: Autor.

Criterios de Aceptación	
Dado:	Soy un usuario de tipo laboratorista
Cuando:	Realizo el registro de los datos pertenecientes a una dependencia y presiono el botón guardar.
Entonces:	Los datos serán almacenados en la base de datos en la nube.

Criterios de Aceptación	
Dado:	Soy un usuario de tipo laboratorista
Cuando:	Realizo cambios a los datos pertenecientes a una dependencia y presiono el botón guardar.
Entonces:	Los cambios se actualizarán en la base de datos.

Criterios de Aceptación	
Dado:	Soy un usuario de tipo laboratorista
Cuando:	Presiono el botón guardar en un registro perteneciente a una dependencia
Entonces:	Los datos de la dependencia serán eliminados de la base de datos.

- Registro de Dispositivos.

Tabla 25: Historia de Usuario: Registro de Dispositivos.

Fuente: Autor.

Historia de Usuario	
Número: 9	**Usuario:** Laboratorista
Nombre historia: Registro de Dispositivos	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Media
Iteración asignada: 2	
Programador responsable: Andrés Tisalema	
Descripción: Debe permitir el registro de los datos pertenecientes a los dispositivos. Los datos que se solicitarán son: Código, la IP del dispositivo, nombre y el estado en el que se encuentra el dispositivo.	
Observaciones: Los datos del usuario deben ser almacenados en la nube.	

Tabla 26: Criterios de Aceptación: Registro de Dispositivos.

Fuente: Autor.

Criterios de Aceptación	
Dado:	Soy un usuario de tipo laboratorista
Cuando:	Realizo el registro de los datos pertenecientes a un dispositivo y presiono el botón guardar.
Entonces:	Los datos serán almacenados en la base de datos en la nube.

Criterios de Aceptación	
Dado:	Soy un usuario de tipo laboratorista
Cuando:	Realizo cambios a los datos pertenecientes a un dispositivo y presiono el botón guardar.
Entonces:	Los cambios se actualizarán en la base de datos.

Criterios de Aceptación	
Dado:	Soy un usuario de tipo laboratorista
Cuando:	Presiono el botón guardar en un registro perteneciente a un dispositivo
Entonces:	Los datos del dispositivo serán eliminados de la base de datos.

- Registro de Oficinas.

Tabla 27: Historia de Usuario: Registro de Oficinas.

Fuente: Autor.

Historia de Usuario	
Número: 10	**Usuario:** Laboratorista
Nombre historia: Registro de Oficinas	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Media
Iteración asignada: 2	
Programador responsable: Andrés Tisalema	
Descripción: Debe permitir el registro de los datos pertenecientes a las oficinas. Los datos que se solicitarán son: Código, nombre, ubicación y la facultad a la cual pertenece la oficina.	
Observaciones: Los datos del usuario deben ser almacenados en la nube.	

Tabla 28: Criterios de Aceptación: Registro de Oficinas.

Fuente: Autor.

Criterios de Aceptación	
Dado:	Soy un usuario de tipo laboratorista
Cuando:	Realizo el registro de los datos pertenecientes a una oficina y presiono el botón guardar.
Entonces:	Los datos serán almacenados en la base de datos en la nube.

Criterios de Aceptación	
Dado:	Soy un usuario de tipo laboratorista
Cuando:	Realizo cambios a los datos pertenecientes a una oficina y presiono el botón guardar.
Entonces:	Los cambios se actualizarán en la base de datos.

Criterios de Aceptación	
Dado:	Soy un usuario de tipo laboratorista
Cuando:	Presiono el botón guardar en un registro perteneciente a una oficina
Entonces:	Los datos de la oficina serán eliminados de la base de datos.

- Registro de Aulas y Laboratorios.

Tabla 29: Historia de Usuario: Registro de Aulas y Laboratorios.

Fuente: Autor.

Historia de Usuario	
Número: 11	**Usuario:** Laboratorista
Nombre historia: Registro de Aulas y Laboratorios	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Media
Iteración asignada: 2	
Programador responsable: Andrés Tisalema	
Descripción: Debe permitir el registro de los datos pertenecientes a las aulas y los laboratorios. Los datos que se solicitarán son: Código, nombre, número de equipos, capacidad, ubicación y la facultad a la cual pertenece el aula o laboratorio.	
Observaciones: Los datos del usuario deben ser almacenados en la nube.	

Tabla 30: Criterios de Aceptación: Registro de Aulas y Laboratorios.

Fuente: Autor.

Criterios de Aceptación	
Dado:	Soy un usuario de tipo laboratorista
Cuando:	Realizo el registro de los datos pertenecientes a un aula/laboratorio y presiono el botón guardar.
Entonces:	Los datos serán almacenados en la base de datos en la nube.

Criterios de Aceptación	
Dado:	Soy un usuario de tipo laboratorista
Cuando:	Realizo cambios a los datos pertenecientes a un aula/laboratorio y presiono el botón guardar.
Entonces:	Los cambios se actualizarán en la base de datos.

Criterios de Aceptación	
Dado:	Soy un usuario de tipo laboratorista
Cuando:	Presiono el botón guardar en un registro perteneciente a un aula/laboratorio
Entonces:	Los datos del aula/laboratorio serán eliminados de la base de datos.

- Registro de Distributivos de Aulas y Laboratorios.

Tabla 31: Historia de Usuario: Registro de Distributivo de aulas y laboratorios.

Fuente: Autor.

Historia de Usuario	
Número: 12	**Usuario:** Laboratorista
Nombre historia: Registro del Distributivo de aulas y laboratorios	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Media
Iteración asignada: 2	
Programador responsable: Andrés Tisalema	
Descripción: Debe permitir el registro de los datos pertenecientes al distributivo de aulas y laboratorios. Los datos que se solicitarán son: Código, aula o laboratorio asignado y su horario respectivo.	
Observaciones: Los datos del usuario deben ser almacenados en la nube.	

Tabla 32: Criterios de Aceptación: Registro de Distributivo de aulas y laboratorios.

Fuente: Autor.

Criterios de Aceptación	
Dado:	Soy un usuario de tipo laboratorista
Cuando:	Realizo el registro de los datos pertenecientes a un distributivo y presiono el botón guardar.
Entonces:	Los datos serán almacenados en la base de datos en la nube.

Criterios de Aceptación	
Dado:	Soy un usuario de tipo laboratorista
Cuando:	Realizo cambios a los datos pertenecientes a un distributivo y presiono el botón guardar.
Entonces:	Los cambios se actualizarán en la base de datos.

Criterios de Aceptación	
Dado:	Soy un usuario de tipo laboratorista
Cuando:	Presiono el botón guardar en un registro perteneciente a un distributivo
Entonces:	Los datos del distributivo serán eliminados de la base de datos.

- Enrolamiento o registro de usuarios.

Tabla 33: Historia de Usuario: Enrolamiento o registro de usuarios.

Fuente: Autor.

Historia de Usuario	
Número: 13	**Usuario:** Laboratorista
Nombre historia: Enrolamiento o registro de usuarios	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Alta
Iteración asignada: 3	
Programador responsable: Andrés Tisalema	
Descripción: El sistema debe solicitar la cedula del usuario y verificar si pertenece a la comunidad universitaria, si el usuario pertenece a la universidad debe solicitar su huella dactilar para el proceso de enrolamiento. El sistema debe mostrar a través de un indicador la calidad de la huella capturada.	
Observaciones: Los datos del usuario deben ser almacenados en la nube.	

Tabla 34: Criterios de Aceptación: Enrolamiento o registro de usuarios.

Fuente: Autor.

Criterios de Aceptación	
Dado:	Soy un usuario de tipo laboratorista
Cuando:	Realizo el proceso de enrolamiento y presiono el botón registrar.
Entonces:	Los datos serán almacenados en la base de datos en la nube.

- Prestamos de aulas y laboratorios al personal de la Fisei.

Tabla 35: Historia de Usuario: Préstamos de aulas y laboratorios al personal de la Fisei.

Fuente: Autor.

Historia de Usuario	
Número: 14	**Usuario:** Laboratorista
Nombre historia: Prestamos de aulas y laboratorios al personal de la Fisei.	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Alta
Iteración asignada: 3	
Programador responsable: Andrés Tisalema	
Descripción: El sistema web debe permitir realizar el proceso de préstamos de aulas y laboratorios disponibles al personal de la Fisei que lo solicite, a través del ingreso de los datos personales del solicitante, además de su huella dactilar para la verificación y apertura del aula o laboratorio.	
Observaciones: El aula o laboratorio solicitado debe encontrarse disponible para su préstamo.	

Tabla 36: Criterios de Aceptación: Préstamos de aulas y laboratorios al personal de la Fisei.

Fuente: Autor.

Criterios de Aceptación	
Dado:	Soy un usuario de tipo laboratorista
Cuando:	Realizo el préstamo de aulas/laboratorios y presiono el botón registrar.
Entonces:	Los datos serán almacenados en la base de datos en la nube.

Proceso: Personal Docente

- Proceso de validación de identidad.

Tabla 37: Historia de Usuario: Proceso de validación de identidad.

Fuente: Autor.

Historia de Usuario	
Número: 1	**Usuario:** Docente
Nombre historia: Proceso de validación de identidad	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Alta
Iteración asignada: 4	
Programador responsable: Andrés Tisalema	
Descripción: El sistema debe comparar la huella dactilar del usuario con las huellas almacenadas en la base de datos en la nube. Si la huella es encontrada, el usuario es identificado.	
Observaciones:	

Tabla 38: Criterios de Aceptación: Proceso de validación de identidad.

Fuente: Autor.

Criterios de Aceptación	
Dado:	Soy un usuario de tipo docente
Cuando:	Registro mi huella dactilar en el sensor hamster.
Entonces:	Si tengo permisos de ingreso, el sistema imprimirá el mensaje de bienvenido.

Criterios de Aceptación	
Dado:	Soy un usuario de tipo docente
Cuando:	Registro mi huella dactilar en el sensor hamster.
Entonces:	Si no tengo permisos de ingreso, el sistema imprimirá el mensaje de No autorizado.

- Registro de ingreso a aulas o laboratorios en horarios de clases.

Tabla 39: Historia de Usuario: Registro de ingreso a aulas y laboratorios.

Fuente: Autor.

Historia de Usuario	
Número: 2	**Usuario:** Docente
Nombre historia: Registro de ingreso a aulas y laboratorios	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Alta
Iteración asignada: 4	
Programador responsable: Andrés Tisalema	
Descripción: El sistema debe solicitar la huella dactilar del docente y validar su identidad, además de contrastar su horario de clases con el aula o laboratorio asignado. Se debe registrar su ingreso y habilitar su acceso al aula o laboratorio.	
Observaciones: El registro de ingreso debe ser almacenado en la nube.	

Tabla 40: Criterios de Aceptación: Registro de ingreso a aulas y laboratorios.

Fuente: Autor.

Criterios de Aceptación	
Dado:	Soy un usuario de tipo docente
Cuando:	Registro mi huella dactilar en el sensor hamster.
Entonces:	Si tengo permisos de ingreso, se almacenará el registro en la base de datos

Criterios de Aceptación	
Dado:	Soy un usuario de tipo docente
Cuando:	Registro mi huella dactilar en el sensor hámster.
Entonces:	Si no tengo permisos de ingreso, no se almacenará ningún registro.

- Registro de préstamo de aulas o laboratorios.

Tabla 41: Historia de Usuario: Registro de préstamo de aulas y laboratorios.

Fuente: Autor.

Historia de Usuario	
Número: 3	**Usuario:** Docente
Nombre historia: Registro del préstamo de aulas y laboratorios	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Baja
Iteración asignada: 4	
Programador responsable: Andrés Tisalema	
Descripción: El sistema debe solicitar la huella dactilar del docente, y verificar si su huella consta en el registro de préstamos de aulas y laboratorios. Se debe registrar su ingreso y habilitar su acceso al aula o laboratorio.	
Observaciones: Los datos del docente deben constar en el registro de préstamos, el registro de ingreso debe almacenarse en la nube.	

Tabla 42: Criterios de Aceptación: Registro de préstamo de aulas y laboratorios.

Fuente: Autor.

Criterios de Aceptación	
Dado:	Soy un usuario de tipo docente
Cuando:	Registro mi huella dactilar en el sensor hámster.
Entonces:	Si solicité el aula o laboratorio, el registro se almacenará en la base de datos.

Criterios de Aceptación	
Dado:	Soy un usuario de tipo docente
Cuando:	Registro mi huella dactilar en el sensor hámster.
Entonces:	Si no solicité el aula o laboratorio, el registro no se almacenará en la base de datos.

Proceso: Personal de Investigación

- Registro de ingreso a oficinas de investigación.

Tabla 43: Historia de Usuario: Registro de ingreso a oficinas de investigación.

Fuente: Autor.

Historia de Usuario	
Número: 1	**Usuario:** Investigador
Nombre historia: Registro de ingreso a oficinas de investigación	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Alta
Iteración asignada: 4	
Programador responsable: Andrés Tisalema	
Descripción: El sistema debe solicitar la huella dactilar del investigador y validar su identidad, además de verificar su oficina asignada. Se debe registrar su ingreso y habilitar su acceso a la oficina de investigación.	
Observaciones: El registro de ingreso debe ser almacenado en la nube.	

Tabla 44: Criterios de Aceptación: Registro de ingreso a oficinas de investigación.

Fuente: Autor.

Criterios de Aceptación	
Dado:	Soy un usuario de tipo investigador
Cuando:	Registro mi huella dactilar en el sensor hámster.
Entonces:	Si estoy asignado a la oficina, el registro se almacenará en la base de datos.

Criterios de Aceptación	
Dado:	Soy un usuario de tipo investigador
Cuando:	Registro mi huella dactilar en el sensor hámster.
Entonces:	Si no estoy asignado a la oficina, el registro no se almacenará en la base de datos.

- Registro de préstamo de aulas o laboratorios.

Tabla 45: Historia de Usuario: Registro de préstamo de aulas y laboratorios.

Fuente: Autor.

Historia de Usuario	
Número: 2	**Usuario:** Investigador
Nombre historia: Registro de préstamo de aulas y laboratorios	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Alta
Iteración asignada: 4	
Programador responsable: Andrés Tisalema	
Descripción: El sistema debe solicitar la huella dactilar del investigador, y verificar si su huella consta en el registro de préstamos de aulas y laboratorios. Se debe registrar su ingreso y habilitar su acceso al aula o laboratorio.	
Observaciones: Los datos del investigador deben constar en el registro de préstamos, el registro de ingreso debe almacenarse en la nube.	

Tabla 46: Criterios de Aceptación: Registro de préstamo de aulas y laboratorios.

Fuente: Autor.

Criterios de Aceptación	
Dado:	Soy un usuario de tipo investigador
Cuando:	Registro mi huella dactilar en el sensor hámster.
Entonces:	Si solicité el aula o laboratorio, el registro se almacenará en la base de datos.

Criterios de Aceptación	
Dado:	Soy un usuario de tipo investigador
Cuando:	Registro mi huella dactilar en el sensor hámster.
Entonces:	Si no solicité el aula o laboratorio, el registro no se almacenará en la base de datos.

Proceso: Personal Docente/Investigador

- Registro de ingreso a aulas o laboratorios en horarios de clases.

Tabla 47: Historia de Usuario: Registro de ingreso a aulas y laboratorios.

Fuente: Autor.

Historia de Usuario	
Número: 1	**Usuario:** Docente/Investigador
Nombre historia: Registro de ingreso a aulas y laboratorios	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Alta
Iteración asignada: 4	
Programador responsable: Andrés Tisalema	
Descripción: El sistema debe solicitar la huella dactilar del docente/investigador y validar su identidad, además de contrastar su horario de clases con el aula o laboratorio asignado. Se debe registrar su ingreso y habilitar su acceso al aula o laboratorio.	
Observaciones: El registro de ingreso debe ser almacenado en la nube.	

Tabla 48: Criterios de Aceptación: Registro de ingreso a aulas y laboratorios.

Fuente: Autor.

Criterios de Aceptación	
Dado:	Soy un usuario de tipo docente/investigador
Cuando:	Registro mi huella dactilar en el sensor hamster.
Entonces:	Si tengo permisos de ingreso, se almacenará el registro en la base de datos

Criterios de Aceptación	
Dado:	Soy un usuario de tipo docente/investigador
Cuando:	Registro mi huella dactilar en el sensor hámster.
Entonces:	Si no tengo permisos de ingreso, no se almacenará ningún registro.

- Registro de préstamo de aulas o laboratorios.

Tabla 49: Historia de Usuario: Registro de préstamo de aulas y laboratorios.

Fuente: Autor.

Historia de Usuario	
Número: 2	**Usuario:** Docente/Investigador
Nombre historia: Registro de préstamo de aulas y laboratorios	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Alta
Iteración asignada: 4	
Programador responsable: Andrés Tisalema	
Descripción: El sistema debe solicitar la huella dactilar del docente/investigador, y verificar si su huella consta en el registro de préstamos de aulas y laboratorios. Se debe registrar su ingreso y habilitar su acceso al aula o laboratorio.	
Observaciones: Los datos del docente/investigador deben constar en el registro de préstamos, el registro de ingreso debe almacenarse en la nube.	

Tabla 50: Criterios de Aceptación: Registro de préstamo de aulas y laboratorios.

Fuente: Autor.

Criterios de Aceptación	
Dado:	Soy un usuario de tipo docente/investigador
Cuando:	Registro mi huella dactilar en el sensor hámster.
Entonces:	Si solicité el aula o laboratorio, el registro se almacenará en la base de datos.

Criterios de Aceptación	
Dado:	Soy un usuario de tipo docente/investigador
Cuando:	Registro mi huella dactilar en el sensor hámster.
Entonces:	Si no solicité el aula o laboratorio, el registro no se almacenará en la base de datos.

- Registro de ingreso a oficinas de investigación.

Tabla 51: Historia de Usuario: Registro de ingreso a oficinas de investigación.

Fuente: Autor.

Historia de Usuario	
Número: 3	**Usuario:** Docente/Investigador
Nombre historia: Registro de ingreso a oficinas de investigación	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Alta
Iteración asignada: 4	
Programador responsable: Andrés Tisalema	
Descripción: El sistema debe solicitar la huella dactilar del docente/investigador y validar su identidad, además de verificar su oficina asignada. Se debe registrar su ingreso y habilitar su acceso a la oficina de investigación.	
Observaciones: El registro de ingreso debe ser almacenado en la nube.	

Tabla 52: Criterios de Aceptación: Registro de ingreso a oficinas de investigación.

Fuente: Autor.

Criterios de Aceptación	
Dado:	Soy un usuario de tipo docente/investigador
Cuando:	Registro mi huella dactilar en el sensor hámster.
Entonces:	Si estoy asignado a la oficina, el registro se almacenará en la base de datos.

Criterios de Aceptación	
Dado:	Soy un usuario de tipo docente/investigador
Cuando:	Registro mi huella dactilar en el sensor hámster.
Entonces:	Si no estoy asignado a la oficina, el registro no se almacenará en la base de datos.

Proceso: Personal Estudiantil

- Registro de préstamo de aulas o laboratorios.

Tabla 53: Historia de Usuario: Registro de préstamo de aulas y laboratorios.

Fuente: Autor.

Historia de Usuario	
Número: 1	**Usuario:** Estudiante
Nombre historia: Registro de préstamo de aulas y laboratorios	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Alta
Iteración asignada: 4	
Programador responsable: Andrés Tisalema	
Descripción: El sistema debe solicitar la huella dactilar del estudiante, y verificar si su huella consta en el registro de préstamos de aulas y laboratorios. Se debe registrar su ingreso y habilitar su acceso al aula o laboratorio.	
Observaciones: Los datos del estudiante deben constar en el registro de préstamos, el registro de ingreso debe almacenarse en la nube.	

Tabla 54: Criterios de Aceptación: Registro de préstamo de aulas y laboratorios.

Fuente: Autor.

Criterios de Aceptación	
Dado:	Soy un usuario de tipo estudiante
Cuando:	Registro mi huella dactilar en el sensor hámster.
Entonces:	Si solicité el aula o laboratorio, el registro se almacenará en la base de datos.

Criterios de Aceptación	
Dado:	Soy un usuario de tipo estudiante
Cuando:	Registro mi huella dactilar en el sensor hámster.
Entonces:	Si no solicité el aula o laboratorio, el registro no se almacenará en la base de datos.

4.5.2. Fase 2. Planificación de la Entrega

En esta fase, se realizó una estimación del tiempo de duración de cada una de las historias de usuarios, pertenecientes a los procesos que conforma el sistema de control.

Proceso: Personal Laboratorista

Tabla 55: Estimación de Historias de Usuarios del Personal Laboratorista.

Fuente: Autor.

PROCESO	N°	HISTORIA DE USUARIO	TIEMPO ESTIMADO		
			SEMANAS ESTIMADAS	DÍAS ESTIMADOS	HORAS ESTIMADAS
LABORATORISTA	1	Registro de Facultades.	0,4	2	8
	2	Registro de Carreras	0,4	2	8
	3	Registro de Categorías	0,4	2	8
	4	Registro del Personal	0,4	2	8
	5	Registro de Asignaturas	0,4	2	8
	6	Registro de Horarios	0,4	2	8
	7	Registro de Jornadas	0,4	2	8
	8	Registro de Dependencias	0,4	2	8
	9	Registro de Dispositivos	0,4	2	8
	10	Registro de Oficinas	0,4	2	8
	11	Registro de Aulas y Laboratorios	0,4	2	8
	12	Registro del Distributivo de aulas y laboratorios	0,4	2	8
	13	Enrolamiento o registro de usuarios	1	5	8
	14	Prestamos de aulas y laboratorios al personal de la Fisei.	1	5	8
	TIEMPO TOTAL ESTIMADO		**6,8**	**34**	**70**

Proceso: Personal Docente

Tabla 56: Estimación de Historias de Usuarios del Personal Docente.

Fuente: Autor.

PROCESO	N°	HISTORIA DE USUARIO	TIEMPO ESTIMADO		
			SEMANAS ESTIMADAS	DÍAS ESTIMADOS	HORAS ESTIMADAS
DOCENTE	1	Proceso de validación de identidad	0,4	2	8
	2	Registro de ingreso a aulas y laboratorios.	0,4	2	8
	3	Registro de préstamo de aulas y laboratorios.	0,6	3	8
	TIEMPO TOTAL ESTIMADO		**1,4**	**7**	**24**

Proceso: Personal de Investigación

Tabla 57: Estimación de Historias de Usuarios del Personal de Investigación.

Fuente: Autor.

PROCESO	N°	HISTORIA DE USUARIO	TIEMPO ESTIMADO		
			SEMANAS ESTIMADAS	DÍAS ESTIMADOS	HORAS ESTIMADAS
INVESTIGADOR	1	Registro de ingreso a oficinas de investigación.	0,4	2	8
	2	Registro de préstamo de aulas y laboratorios.	1	5	8
	TIEMPO TOTAL ESTIMADO		1,4	7	16

Proceso: Personal Docente/Investigador

Tabla 58: Estimación de Historias de Usuarios del Personal Docente/Investigador.

Fuente: Autor.

PROCESO	N°	HISTORIA DE USUARIO	TIEMPO ESTIMADO		
			SEMANAS ESTIMADAS	DÍAS ESTIMADOS	HORAS ESTIMADAS
DOCENTE / INVESTIGADOR	1	Registro de ingreso a aulas y laboratorios.	0,4	2	8
	2	Registro de préstamo de aulas y laboratorios.	0,4	2	8
	3	Registro de ingreso a oficinas de investigación.	0,4	2	8
	TIEMPO TOTAL ESTIMADO		1,2	6	24

Proceso: Personal Estudiantil

Tabla 59: Estimación de Historias de Usuarios del Personal Estudiantil.

Fuente: Autor.

PROCESO	N°	HISTORIA DE USUARIO	TIEMPO ESTIMADO		
			SEMANAS ESTIMADAS	DÍAS ESTIMADOS	HORAS ESTIMADAS
ESTUDIANTE	1	Registro de préstamo de aulas y laboratorios.	0,4	2	8
	TIEMPO TOTAL ESTIMADO		0,4	2	8

4.5.3. Fase 3. Iteraciones

Esta fase se encuentra compuesta por 2 o más iteraciones, cada iteración debe tener una duración menor a 3 semanas. El resultado final de una iteración es un prototipo entregable y funcional del sistema a desarrollar.

Tabla 60: Iteraciones Historias de Usuarios (a).

Fuente: Autor.

ITERACIÓN	N°	HISTORIA DE USUARIO	PRIORIDAD (Entrega)	ACTIVIDAD (Nueva - Ajuste)	DEPENDENCIA (N° Historia Usuario)	RIESGO (Alta - Media - Baja)	ESTADO DE DESARROLLO	PRUEBAS
Primera	1	Registro de Facultades.	1	Nueva	NA	Media	Completo	Aprobado
	2	Registro de Carreras.	2	Nueva	1	Media	Completo	Aprobado
	3	Registro de Categorías.	1	Nueva	NA	Media	Completo	Aprobado
	4	Registro de Personal.	3	Nueva	1, 2, 3	Media	Completo	Aprobado
	5	Registro de Asignaturas.	2	Nueva	1, 2	Media	Completo	Aprobado
	6	Registro de Horarios.	4	Nueva	1, 2, 3, 4, 5	Media	Completo	Aprobado
	7	Registro de Jornadas.	3	Nueva	1, 2, 3, 4	Media	Completo	Aprobado
	8	Registro de Dependencias.	2	Nueva	1, 2	Media	Completo	Aprobado
Segunda	9	Registro de Dispositivos.	1	Nueva	NA	Media	Completo	Aprobado
	10	Registro de Oficinas.	2	Nueva	1	Media	Completo	Aprobado
	11	Registro de Aulas y Laboratorios.	2	Nueva	1	Media	Completo	Aprobado
	12	Registro de Distributivos de Aulas y Laboratorios.	3	Nueva	1, 6	Media	Completo	Aprobado

Tabla 61: Iteraciones Historias de Usuarios (b).

Fuente: Autor.

ITERACIÓN	N°	HISTORIA DE USUARIO	PRIORIDAD (Entrega)	ACTIVIDAD (Nueva - Ajuste)	DEPENDENCIA (N° Historia Usuario)	RIESGO (Alta - Media - Baja)	ESTADO DE DESARROLLO	PRUEBAS
Tercera	13	Enrolamiento o registro de usuarios.	1	Nueva	4	Alta	Completo	Aprobado
	14	Prestamos de Aulas y Laboratorios al personal de la Fisei.	2	Nueva	4, 11, 12	Alta	Completo	Aprobado
Cuarta	15	Proceso de validación de identidad.	1	Nueva	NA	Alta	Completo	Aprobado
	16	Registro de ingreso a aulas o laboratorios en horarios de clase – Docentes.	2	Nueva	15	Alta	Completo	Aprobado
	17	Registro de préstamos de aulas o laboratorios – Docentes.	3	Nueva	14, 15	Alta	Completo	Aprobado
	18	Registro de ingreso a oficinas de investigación – Investigadores.	2	Nueva	15	Alta	Completo	Aprobado
	19	Registro de préstamos de aulas o laboratorios – Investigadores.	2	Nueva	15	Alta	Completo	Aprobado
	20	Registro de ingreso a aulas o laboratorios en horarios de clase – Docente/Investigador.	2	Nueva	15	Alta	Completo	Aprobado
	21	Registro de préstamos de aulas o laboratorios – Docente/Investigador.	3	Nueva	14, 15	Alta	Completo	Aprobado
	22	Registro de ingreso a oficinas de investigación – Docente/Investigador.	2	Nueva	15	Alta	Completo	Aprobado
	23	Registro de préstamos de aulas o laboratorios – Estudiantes.	3	Nueva	14, 15	Alta	Completo	Aprobado

4.5.3.1. Plan de Entrega

Tabla 62: Plan de Entrega (a).

Fuente: Autor.

PROCESO/MÓDULO	N°	HISTORIA DE USUARIO	TIEMPO ESTIMADO			ITERACIÓN ASIGNADA				ENTREGA ASIGNADA			
			Semanas Estimadas	Días Estimados	Horas Estimadas	1	2	3	4	1	2	3	4
LABORATORISTA	1	Registro de Facultades	0,4	2	8	X				X			
	2	Registro de Carreras	0,4	2	8	X					X		
	3	Registro de Categorías	0,4	2	8	X				X			
	4	Registro de Personal	0,4	2	8	X						X	
	5	Registro de Asignaturas	0,4	2	8	X					X		
	6	Registro de Horarios	0,4	2	8	X							X
	7	Registro de Jornadas	0,4	2	8	X						X	
	8	Registro de Dependencias	0,4	2	8	X					X		
	9	Registro de Dispositivos	0,4	2	8		X			X			
	10	Registro de Oficinas	0,4	2	8		X				X		
	11	Registro de Aulas y Laboratorios	0,4	2	8		X				X		
	12	Registro del Distributivo de Aulas y Laboratorios	0,4	2	8		X					X	
	13	Enrolamiento o registro de usuarios	1	5	8			X		X			
	14	Prestamos de aulas y laboratorios al personal	1	5	8			X			X		

Tabla 63: Plan de Entrega (b).

Fuente: Autor.

PROCESO/MÓDULO	N°	HISTORIA DE USUARIO	TIEMPO ESTIMADO			ITERACIÓN ASIGNADA				ENTREGA ASIGNADA			
			Semanas Estimadas	Días Estimados	Horas Estimadas	1	2	3	4	1	2	3	4
DOCENTE	15	Proceso de validación de identidad	0,4	2	8				X	X			
	16	Registro de ingreso a aulas y laboratorios	0,4	2	8				X		X		
	17	Registro de préstamos de aulas y laboratorios	0,6	3	8				X			X	
INVESTIGADOR	18	Registro de ingreso a oficinas de investigación	0,4	2	8				X		X		
	19	Registro de préstamos de aulas y laboratorios	1	5	8				X		X		

Tabla 64: Plan de Entrega (c).

Fuente: Autor.

PROCESO / MÓDULO	N°	HISTORIA DE USUARIO	TIEMPO ESTIMADO			ITERACIÓN ASIGNADA				ENTREGA ASIGNADA			
			Semanas Estimadas	Días Estimados	Horas Estimadas	1	2	3	4	1	2	3	4
DOCENTE / INVESTIGADOR	20	Registro de ingreso a aulas y laboratorios	0,4	2	8				X		X		
	21	Registro de préstamos de aulas y laboratorios	0,4	2	8				X			X	
	23	Registro de ingreso a oficinas de investigación	0,4	2	8				X		X		
ESTUDIANTE	23	Registro de préstamos de aulas y laboratorios	0,4	2	8				X			X	
TIEMPO TOTAL ESTIMADO:			11,2	56	142	3,2	1,6	2	4,4	11,2			

4.5.4. Fase 4. Producción

En la fase de producción se desarrolló el software embebido para el Raspberry Pi y el sistema web de administración.

El sistema embebido IoT para el control de ingreso a aulas y laboratorios, consta de la integración de los sistemas embebidos SecuGen Hamster Plus y Raspberry Pi, administrados por una aplicación de control desarrollada en Java, el gestor de base de datos en la nube, y el sistema web, figura 1.

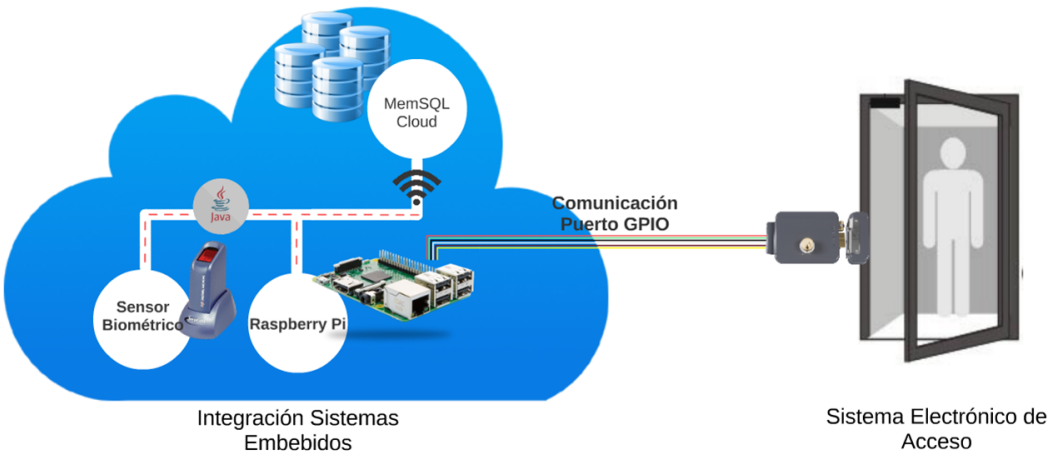


Figura 1: Esquema General del Sistema Embebido IoT.
Fuente: Autor.

4.5.4.1. Diseño de la base de datos.

Para realizar el diseño de la base de datos se determinó su propósito, a partir de un listado con la información que el usuario necesita.

Una vez definidas las tablas, registros y relaciones de la base de datos, se diseñó el modelo relacional utilizando el software MySQL Workbench 6.3, figura 2.

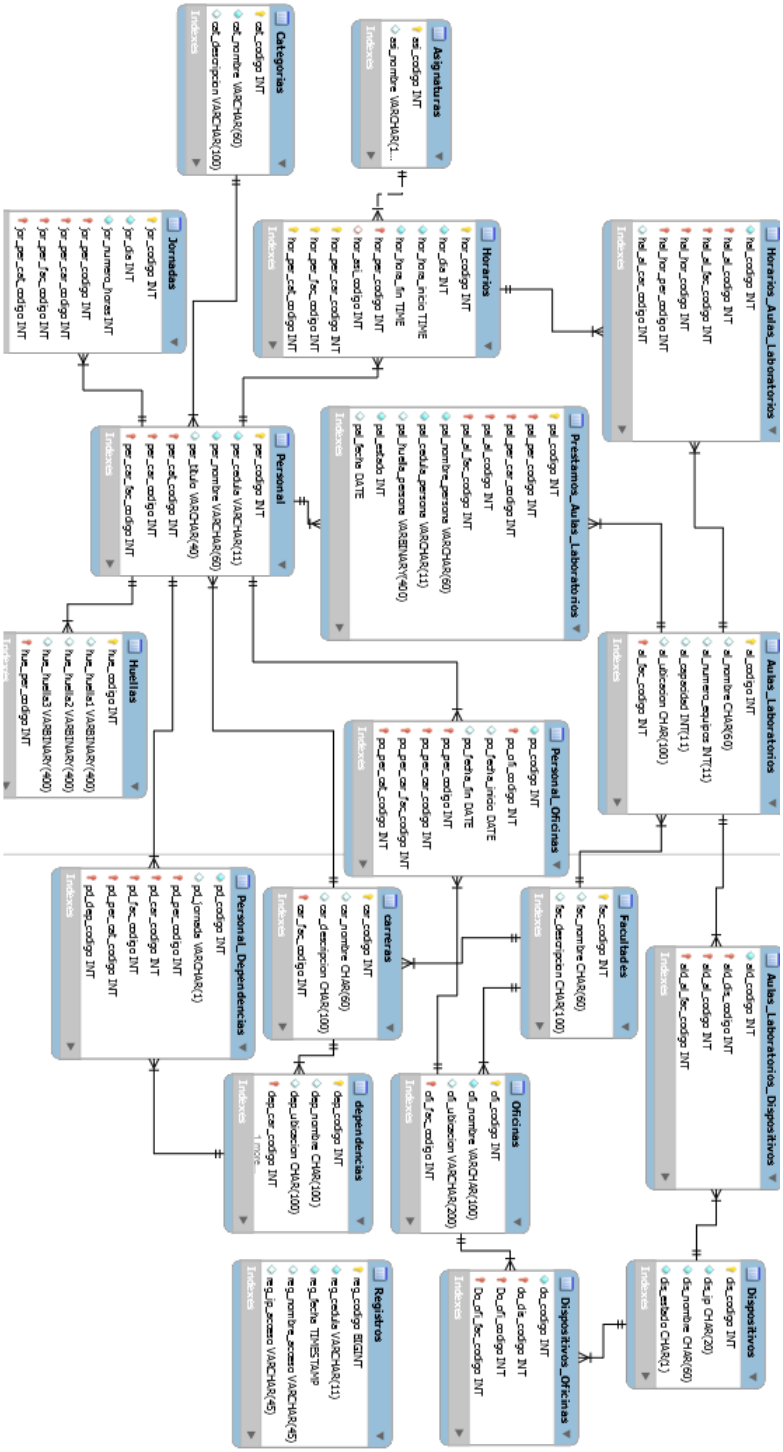


Figura 2: Modelo relacional de la base de datos.
Fuente: Autor.

La base de datos consta de 19 tablas, cada una con sus entidades y relaciones

correspondientes al proceso de control de acceso a aulas y laboratorios de la FISEI. Para definir e identificar el tipo de información que se almacenará en cada una de las tablas de la base de datos, se generó un diccionario de datos.
Los requerimientos funcionales del sistema embebido, además del sistema web, se encuentran definidos en las historias de usuarios. Para su representación gráfica se utilizaron diagramas de casos de uso, debido a su facilidad de interpretación. Los diagramas de casos de uso permiten representar la interacción entre los usuarios o actores y el sistema en desarrollo. Cada caso de uso debe ser relacionado mínimo con un actor, y debe alcanzar una única meta o tarea.
Control de ingreso a aulas y laboratorios - Personal Docente.

Tabla 65: Caso de Uso del Ingreso a Aulas y Laboratorios FISEI.

Fuente: Autor.

Nombre:	Registro Ingreso Aulas/Laboratorios Docentes
Actores:	Docente
Precondiciones El usuario debe estar registrado, y debe pertenecer al personal docente.	
Flujo Normal **1.** El sistema solicita la huella dactilar del docente. **2.** El docente ingresa su huella dactilar. **3.** El sistema valida la identidad del docente. **4.** El sistema verifica y valida el aula o laboratorio asignado al horario de clases del docente. **5.** Se registra el ingreso del docente al aula o laboratorio. **6.** El sistema habilita el acceso al aula o laboratorio.	
Flujo Alternativo **1.** El sistema verifica y valida que el aula o laboratorio no se encuentra asignado al horario de clases del docente. **2.** El sistema presenta un mensaje de notificación. **3.** El sistema regresa a su estado inicial.	
Poscondiciones Se permitió el acceso del docente al aula o laboratorio exitosamente.	

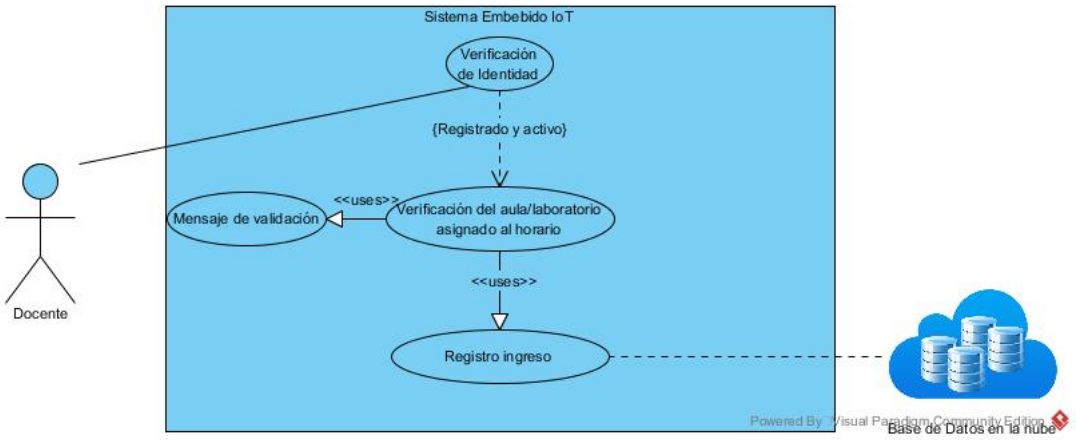


Figura 3: Diagrama de Casos de Uso del Ingreso a Aulas y Laboratorios FISEI. Fuente: Autor.

Control de préstamos de aulas y laboratorios - Personal Docente, Administrativo y Estudiantil.

Tabla 66: Caso de Uso del Prestamo a Aulas y Laboratorios FISEI.

Fuente: Autor.

Nombre:	Prestamos Aulas/Laboratorios Estudiantes
Actores:	Estudiante, Docente, Administrativo
Precondiciones El usuario debe estar registrado para el proceso de préstamos.	
Flujo Normal **1.** El sistema solicita la huella dactilar del usuario. **2.** El usuario ingresa su huella dactilar. **3.** El sistema verifica si la huella del usuario se encuentra registrada para préstamos. **4.** El sistema verifica y valida el aula o laboratorio asignado al usuario. **5.** Se registra el ingreso del usuario al aula o laboratorio. **6.** El sistema habilita el acceso al aula o laboratorio.	
Flujo Alternativo **1.** El sistema verifica que el aula o laboratorio no se encuentra asignado al usuario. **2.** El sistema presenta un mensaje de notificación. **3.** El sistema regresa a su estado inicial.	
Poscondiciones Se permitió el acceso del usuario al aula o laboratorio exitosamente.	

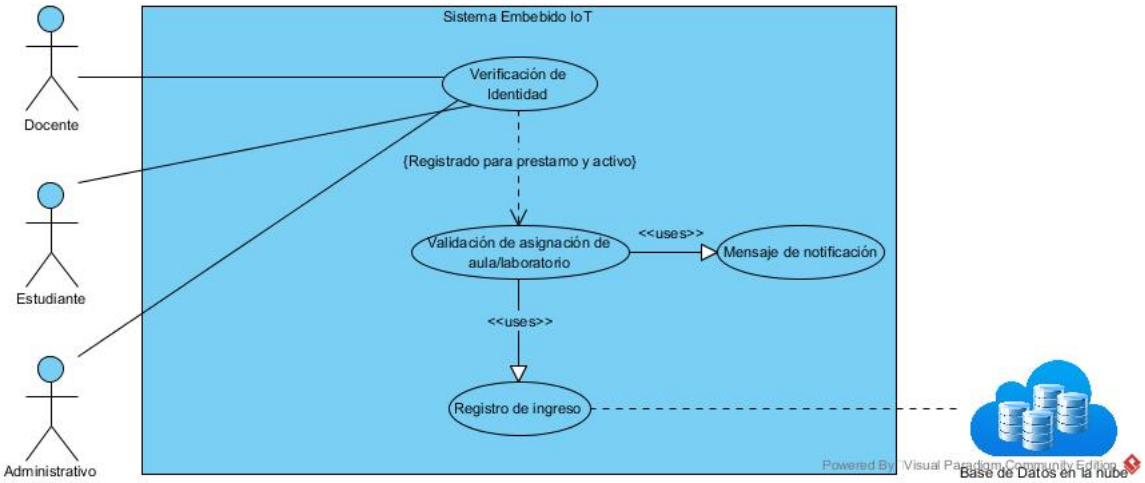


Figura 4: Diagrama de Casos de Uso del Prestamo a Aulas y Laboratorios FISEI. Fuente: Autor.

Control de ingreso a las oficinas de investigación - Personal de Investigación.

Tabla 67: Caso de Uso del Ingreso a Oficinas de Investigación FISEI.

Fuente: Autor.

Nombre:	Ingreso a Oficinas de Investigación
Actores:	Investigador, Docente/Investigador
Precondiciones El usuario debe estar registrado, y debe pertenecer al personal de Investigación o Docente/Investigador.	
Flujo Normal **1.** El sistema solicita la huella dactilar del usuario. **2.** El usuario ingresa su huella dactilar. **3.** El sistema valida la identidad del usuario. **4.** El sistema verifica y valida la oficia asignada al usuario. **5.** Se registra el ingreso del usuario a la oficina. **6.** El sistema habilita el acceso a la oficina.	
Flujo Alternativo **1.** El sistema verifica que la oficina no se encuentra asignada al usuario. **2.** El sistema presenta un mensaje de notificación. **3.** El sistema regresa a su estado inicial.	
Poscondiciones Se permitió el acceso del usuario a la oficina de investigación.	

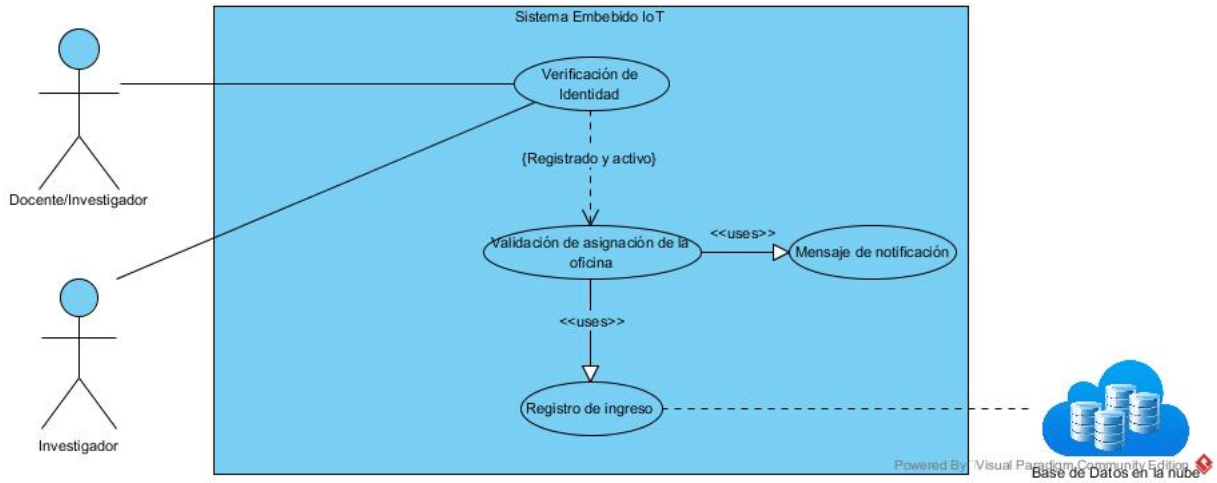


Figura 5: Diagrama de Casos de Uso del Ingreso a Oficinas de Investigación FISEI.

Fuente: Autor.

4.5.4.2. Desarrollo del Software Embebido

El algoritmo desarrollado en lenguaje de programación Java, cumple con el proceso de validación y verificación del sistema de control de acceso propuesto, figura 6.

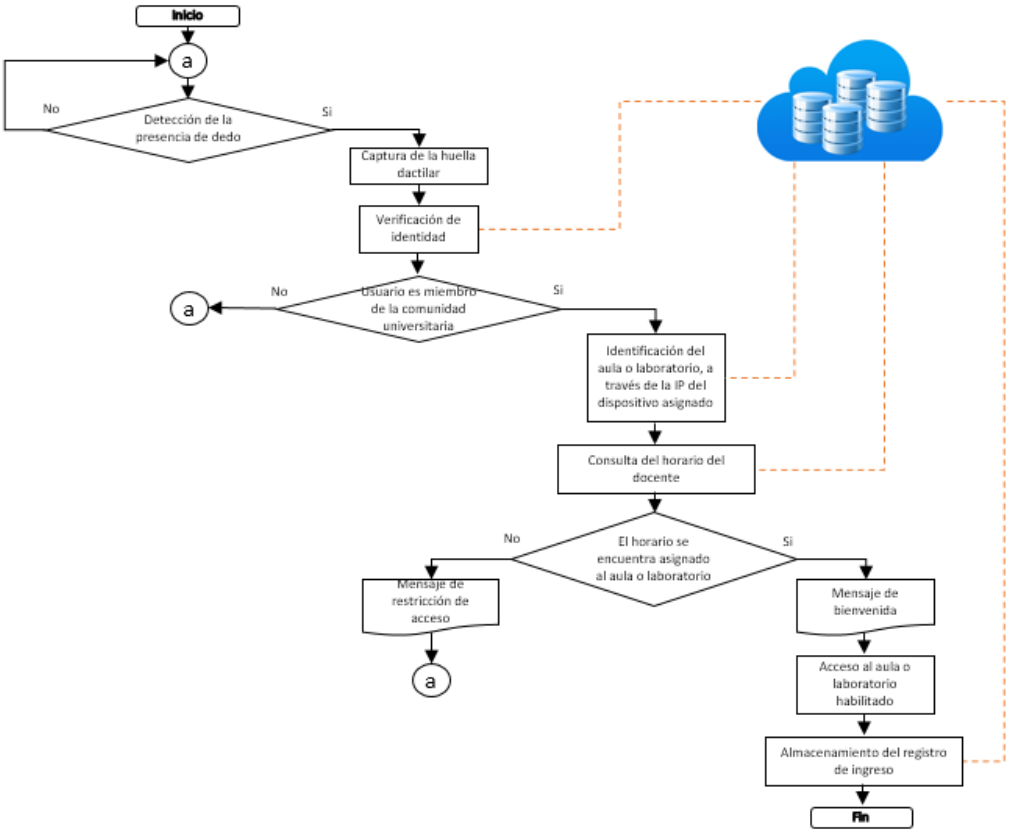


Figura 6: Proceso de verificación de ingreso a aulas y laboratorios.
Fuente: Autor.

El estado inicial del algoritmo comprende la auto detección de la presencia del dedo del usuario, al detectar su presencia, el sensor biométrico SecuGen Hamster Plus capturará la huella dactilar y realizará el proceso de verificación de la identidad. Se utilizó el algoritmo SG400 propio del sensor, definido en una función de emparejamiento que compara la huella capturada con las huellas ya almacenadas en la nube. Una vez identificado el usuario se contrasta su horario con el aula o laboratorio asignado. El resultado exitoso del proceso es la habilitación del acceso al aula o laboratorio, por medio del envío de un pulso eléctrico a través del puerto GPIO del Raspberry Pi hacia la cerradura eléctrica de la puerta.

4.5.4.3. Desarrollo del Sistema Web

La estructura física de la Aplicación Web se encuentra representada a través del diagrama de distribución, mostrado en la figura 7.

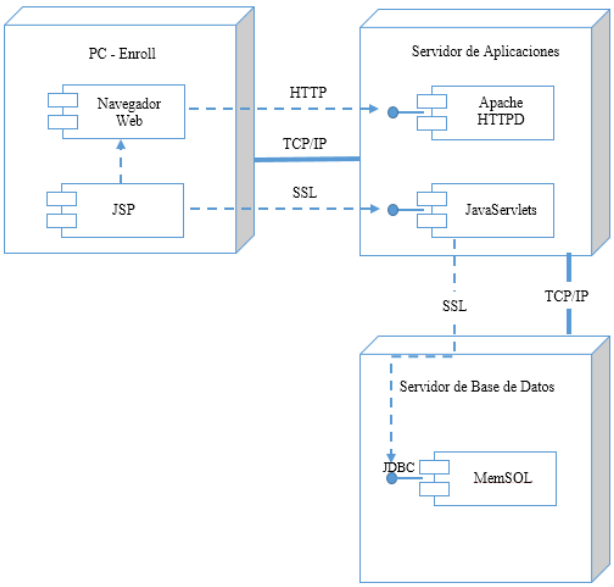


Figura 7: Diagrama de Distribución de la Aplicación Web.
Fuente: Autor.

4.5.4.4. Diseño de Interfaces de Usuario

Los prototipos de interfaz de usuarios se emplean para validar el diseño de la interfaz de usuario, enfocado a los requerimientos del usuario final.
En este contexto, se utilizó la herramienta de código abierto Pencil 3.0.1 para el desarrollo de los prototipos de interfaces de usuario de baja fidelidad y no ejecutables, con el objetivo de validar el diseño general de las interfaces.
El prototipo de la página maestra se muestra en la figura 8, este prototipo es el diseño de la interfaz general del sistema web.

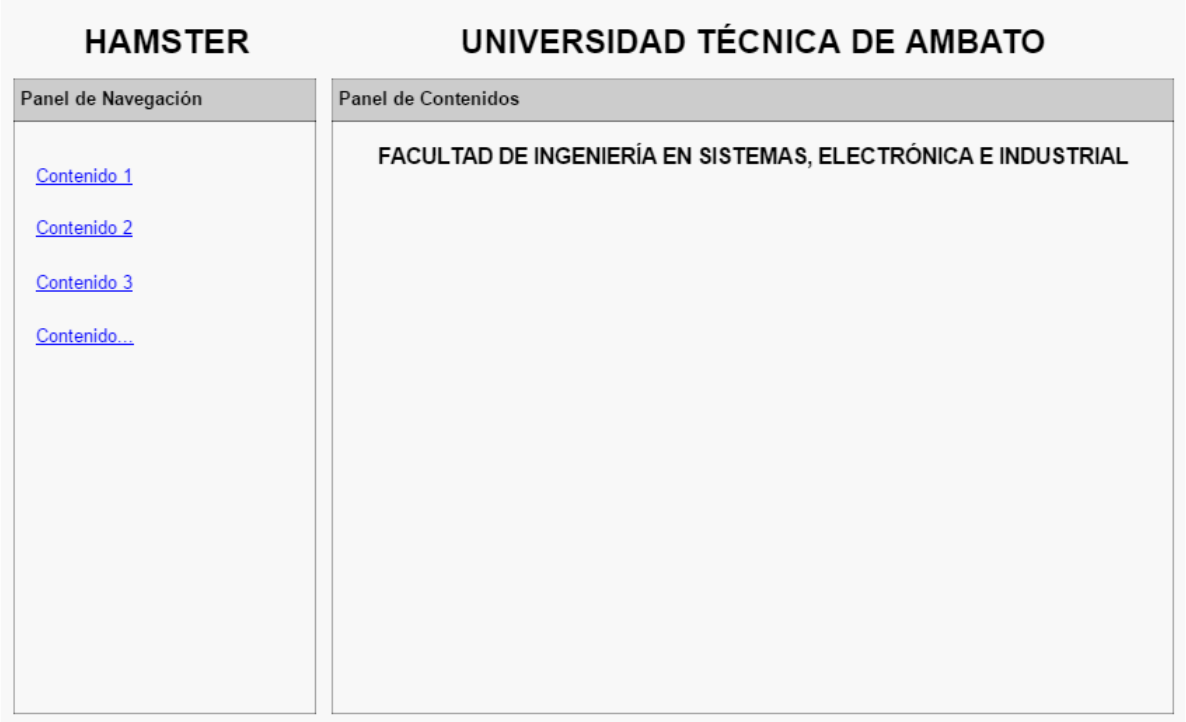


Figura 8: Prototipo de interfaz de usuario general del sistema web.
Fuente: Autor.

En las figuras 9 y 10 se muestran respectivamente, el prototipo de la interfaz de usuario para el proceso de ingreso, modificación y eliminación de parámetros y datos necesarios para el sistema web, y el prototipo que cumple el proceso de consultas de la información almacenada en la base de datos en la nube.

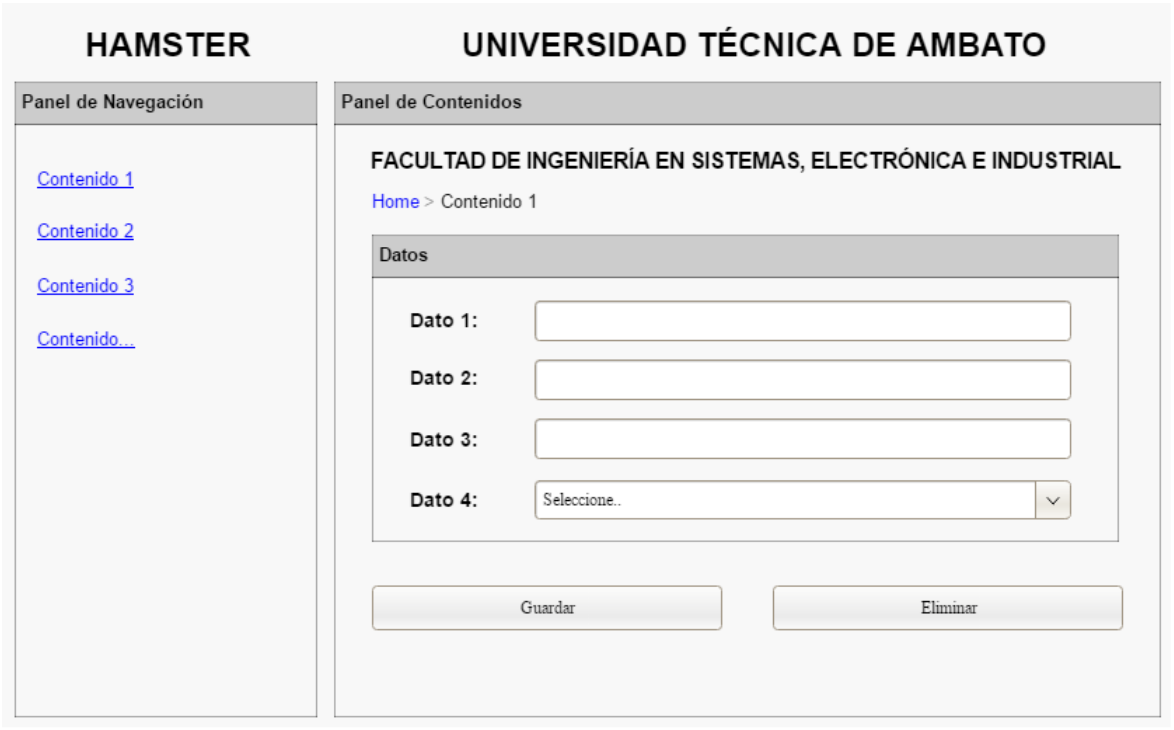


Figura 9: Prototipo de interfaz de usuario de ingreso de parámetros.
Fuente: Autor.

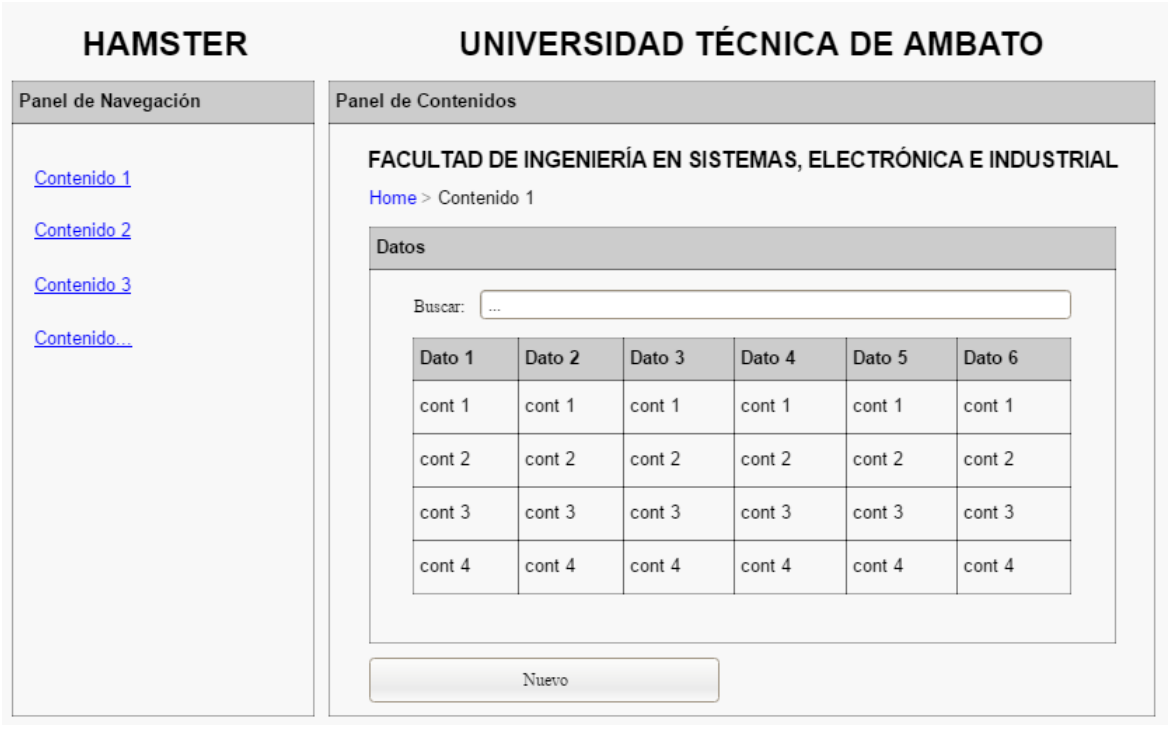


Figura 10: Prototipo de interfaz de usuario de consultas.
Fuente: Autor.

Las figuras 11 y 12, representan los procesos principales del sistema web.

El proceso de enrolamiento, que requiere de la información personal del usuario a ser enrolado, además de su huella dactilar.

Figura 11: Prototipo de interfaz de usuario del proceso de enrolamiento.
Fuente: Autor.

El proceso de prestamos de aulas y laboratorios, que lo realiza el laboratorista encargado de las aulas y laboratorios hacia los estudiantes, docentes y personal universitario que lo solicite.

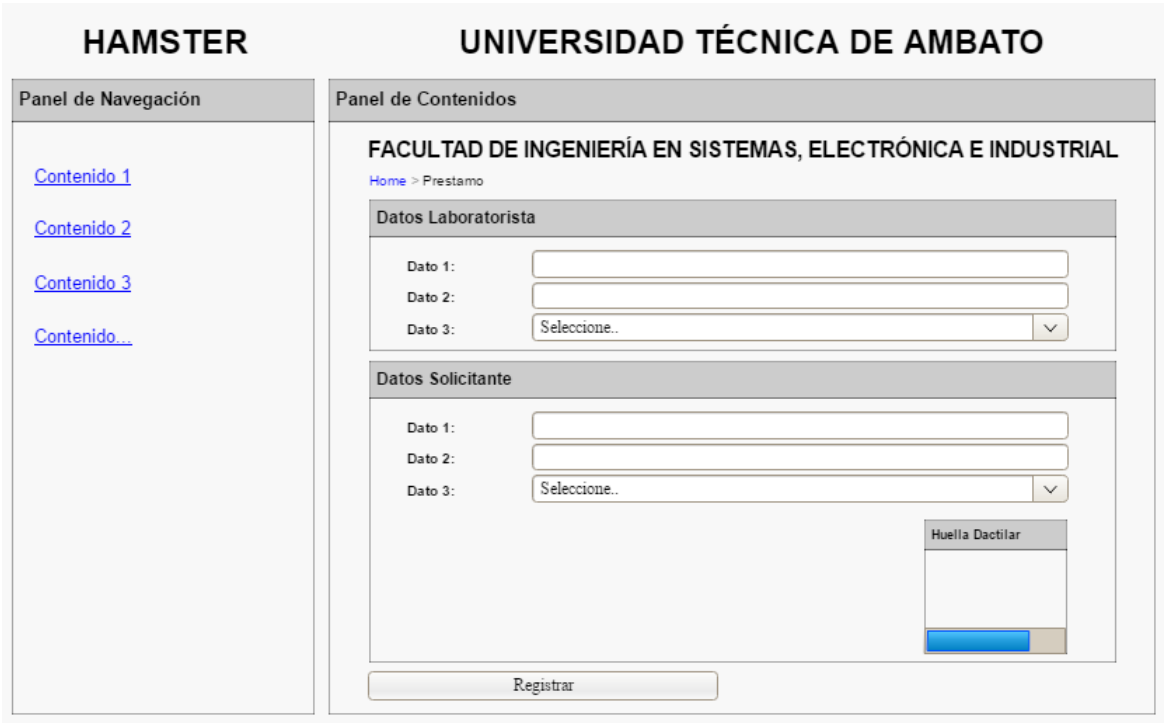


Figura 12: Prototipo de interfaz de usuario del proceso de prestamos de aulas y laboratorios.

Fuente: Autor.

El proceso de captura y obtención de la imagen debe cumplir con los siguientes parámetros:

- Calidad de la Huella.
- Posición de la Huella.

La calidad de la huella es un indicador que se muestra por medio de un porcentaje que va de 0 a 100 %, dependiendo de la calidad de la imagen de la huella dactilar capturada por el sensor SecuGen Hamster Plus. En la figura 13, se muestra los indicadores de calidad de una huella capturada en la Aplicación Web.

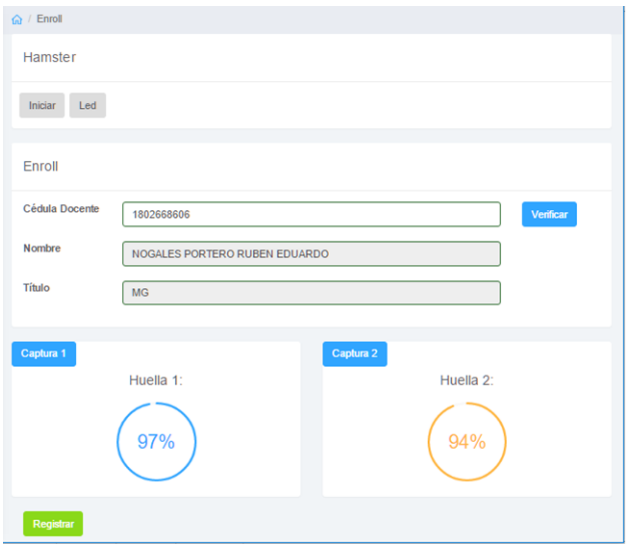


Figura 13: Proceso de enrolamiento de la Aplicación Web.
Fuente: Autor.

La posición en la que se captura la huella dactilar determinará su calidad, figura 14.

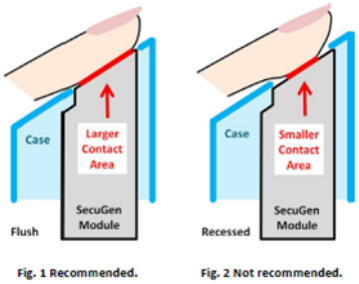


Figura 14: Posición recomendada para la toma de un a huella dactilar.
Fuente: Case Design for Optimal Fingerprint Scanning and Optimal Auto-On Finger Detection, [27].

Si la captura de la imagen es exitosa, se extraen las minucias de la huella dactilar, y se genera su template. La función GetImageEx() captura la imagen dactilar del dedo. La función GetImageQuality() evalúa la calidad de la imagen capturada. El proceso de enrolamiento concluye con la inserción de los datos a la Base de Datos en la nube, se almacenará la cédula del usuario, y los datos biométricos de los dos template generados.

4.5.4.5. Aplicación de patrones de diseño al Sistema Web

A partir del análisis comparativo ya realizado entre los 10 patrones de diseño más utilizados en el desarrollo de software, se seleccionó el patrón Singleton para ser

aplicado en el software propuesto según las características que presenta, figura 15.

Figura 15: Esquema de la aplicación de patrones de disñeo.
Fuente: Autor.

4.5.4.6. Patrón Singleton

El patrón de diseño Singleton permite una única instanciación de un objeto, asegurando así un punto de acceso global al objeto creado, figura 16.

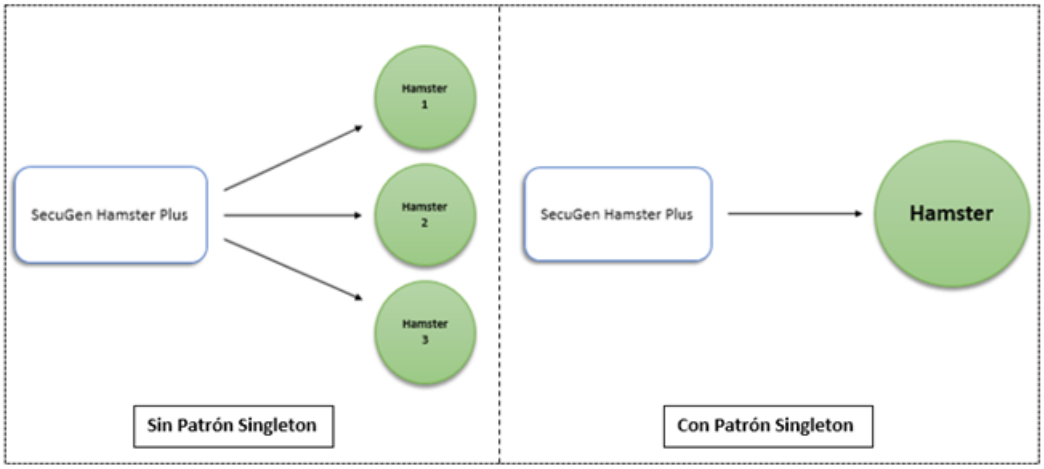


Figura 16: Patrón de diseño Singleton.
Fuente: Autor.

Para implementar el patrón Singleton se creó un constructor privado que solo puede ser accedido desde la misma clase, se creó también una instancia privada de la clase, además de un método estático que permite el acceso a la instancia que inicializa el sensor SecuGen Hamster Plus, figuras 17 y 18.

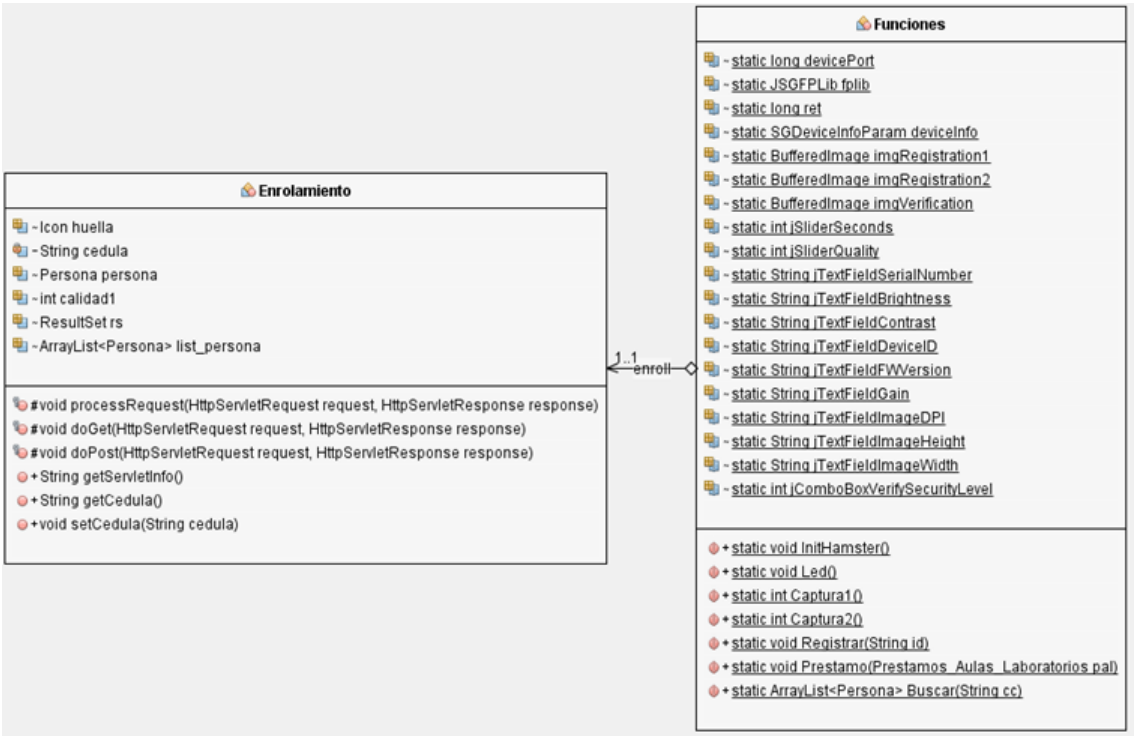


Figura 17: Diagrama de clases sin la implementación del Patrón Singleton.
Fuente: Autor.

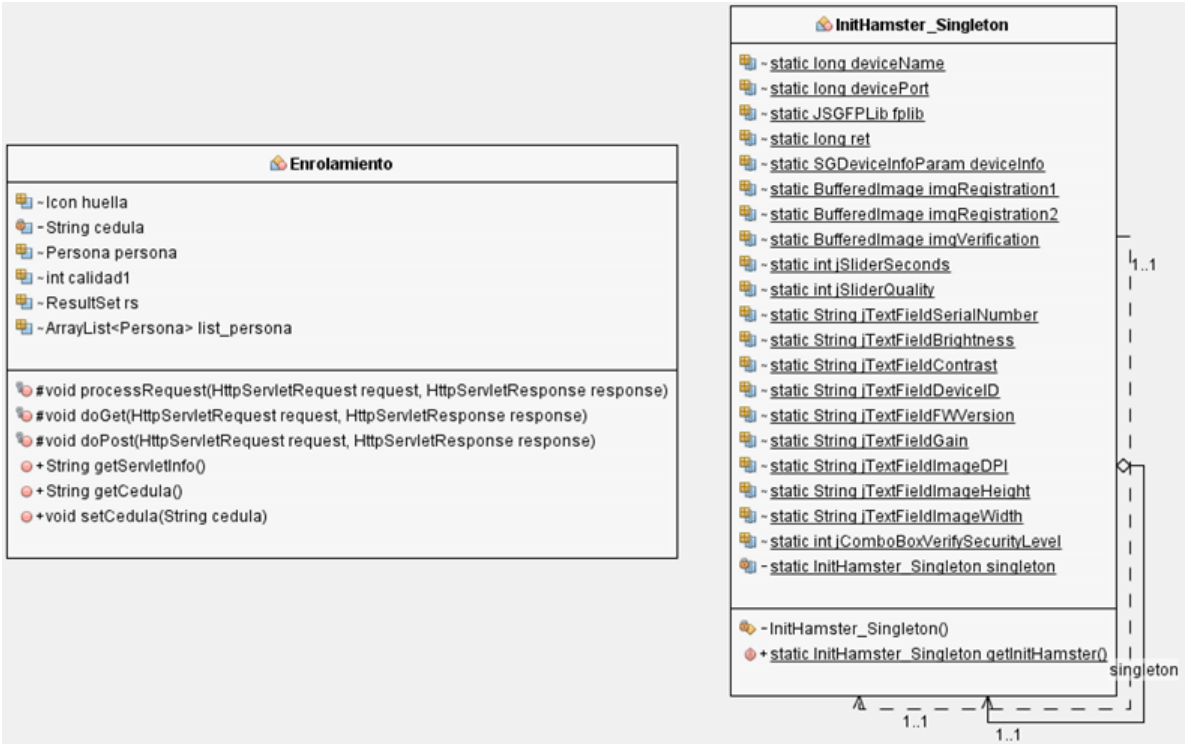


Figura 18: Diagrama de clases con la implementación del Patrón Singleton.
Fuente: Autor.

4.5.5. Fase 5. Pruebas

Previo a la implementación del sistema embebido en un entorno de producción, se requiere la validación del funcionamiento del software propuesto a través de un proceso de pruebas.

4.5.5.1. Pruebas de Caja Negra

El proceso de pruebas de caja negra se desarrolla en un entorno del lado del cliente, con la participación del usuario final. El objetivo principar de las pruebas de caja negra es verificar el registro, procesamiento y recuperación de datos, además del cumplimiento apropiado de las reglas de negocio.
Para la ejecución del proceso de pruebas, se configuró el servidor web GlassFish, en el cual se pueden ejecutar aplicaciones web desarrolladas en Java.

- Para habilitar GlassFish en CentOS, se requieren instalar varios paquetes y librerías.
- Es necesario descargar, e instalar Java JDK o el paquete Java SE Development Kit, figura 19.

```
[root@localhost opt]# sudo yum install java-1.7.0-openjdk
Complementos cargados:fastestmirror
Loading mirror speeds from cached hostfile
 * base: mirror.ueb.edu.ec
 * extras: mirror.ueb.edu.ec
 * updates: mirror.ueb.edu.ec
Resolviendo dependencias
--> Ejecutando prueba de transacción
---> Paquete java-1.7.0-openjdk.x86_64 1:1.7.0.141-2.6.10.1.el7_3 debe ser instala
do
--> Procesando dependencias: java-1.7.0-openjdk-headless = 1:1.7.0.141-2.6.10.1.el
7_3 para el paquete: 1:java-1.7.0-openjdk-1.7.0.141-2.6.10.1.el7_3.x86_64
--> Procesando dependencias: xorg-x11-fonts-Type1 para el paquete: 1:java-1.7.0-op
enjdk-1.7.0.141-2.6.10.1.el7_3.x86_64
```

Figura 19: Instalación del paquete Java JDK.
Fuente: Autor.

- A continuación se crea un usuario, debido a que GlassFish no se ejecuta en el entorno del usuario root, figura 20.

```
[root@localhost opt]# adduser \
> --comment 'GlassFish User' \
> --home-dir /home/glassfish \
> glassfish
[root@localhost opt]# su -
Último inicio de sesión:mar jun 20 01:19:36 -05 2017de 192.168.1.7en pts/0
[root@localhost ~]# su - glassfish
```

Figura 20: Creación de usuario GlassFish.
Fuente: Autor.

- Se descarga e instala la versión más estable de GlassFish. Se debe crear un systemd service, generando un fichero llamado glassfish.service, figura 21.

Figura 21: Instalación y configuración de GlassFish.
Fuente: Autor.

- Como ultimo punto, se inicia el servicio de GlassFish con los comandos mostrados en la figura 22.

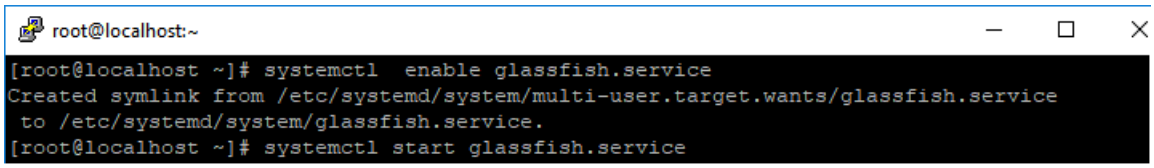


Figura 22: Habilitación de GlassFish en el servidor web.
Fuente: Autor.

Una vez que ha sido configurado y publicado el sistema web en GlassFish, puede ser accedido a través de un navegador web con la Url o dirección asignada por el servidor web.

Dispositivo Embebido Raspberry Pi.

La puesta a punta del dispositivo embebido Raspberry Pi se realizó, a través de la instalación del SDK FDx Pro. Estas librerías brindaran un nivel de acceso para la administración y comunicación del sensor biométrico SecuGen Hamster Plus FDU03 con el sistema operativo Raspbian del Raspberry Pi.
El proceso de instalación inicia con el comando «make install», en la carpeta en la cual se encuentran las librerías, figura 23.

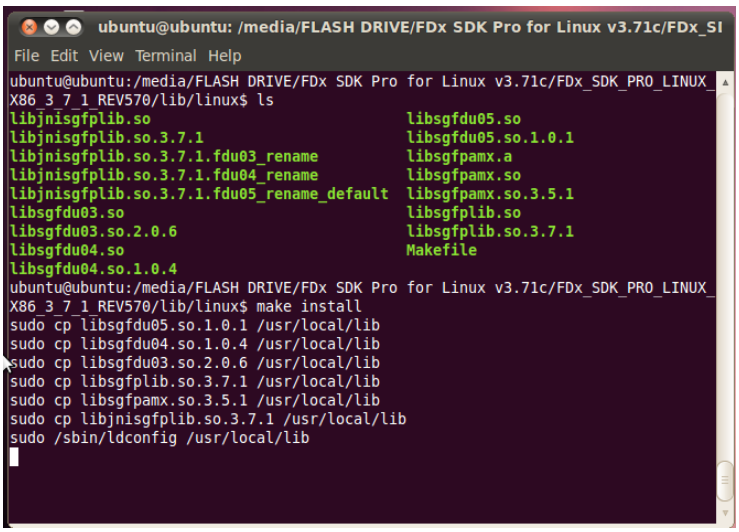


Figura 23: Instalación del SDK.
Fuente: Autor.

A continuación se crean grupos de usuarios, los cuales tendrán acceso al sensor biométrico. El comando para la creación de grupos es «groupadd NombreGrupo», es necesario agregar los usuarios al grupo creado a través del comando «gpasswd -a Usuario NombreGrupo», figura 24.

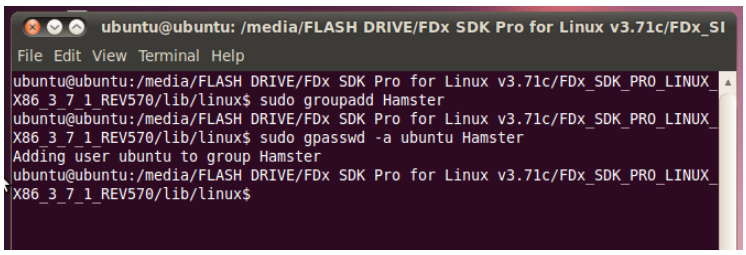


Figura 24: Creación del Grupo de Acceso.
Fuente: Autor.

Es importante identificar y definir el modelo del sensor biométrico que el sistema operativo va identificar, figura 25.

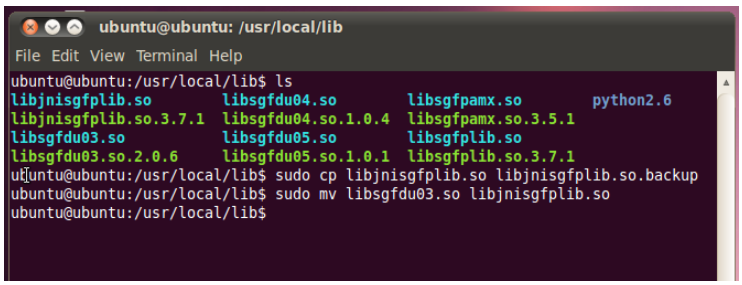


Figura 25: Renombrar la librería según modelo del sensor.
Fuente: Autor.

4.5.5.2. Casos de Pruebas Funcionales

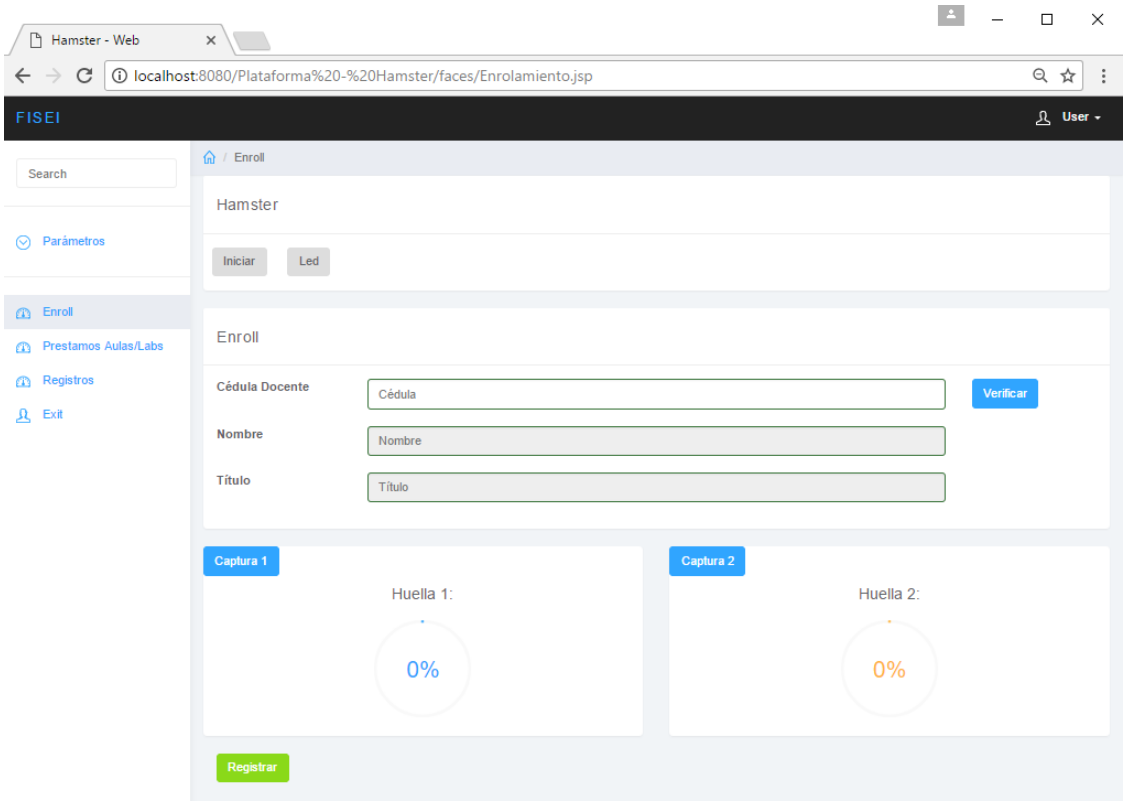


Figura 26: Sistema Web - Proceso de enrolamiento.
Fuente: Autor.

Tabla 68: Caso de Prueba: Proceso Enrolamiento.

Fuente: Autor.

Criterio	Observaciones
ID	CP-01
Nombre	Proceso enrolamiento usuarios
Descripción	Se probará la respuesta del sistema para el proceso de enrolamiento del personal de la facultad.
Precondiciones	El usuario debe pertenecer al personal universitario.
Pasos y condiciones ejecución	1. El sistema solicita la C.I. del usuario. 2. El sistema valida que el usuario pertenezca a la Universidad. 3. El sistema solicita la huella dactilar del usuario para su registro. 4. La calidad de la huella dactilar obtenida debe ser mayor al 60%.
Resultado esperado	Se realiza el registro de los datos personales, además de la huella dactilar del usuario en la base de datos en la nube.
Estado	Ejecutado
Resultado obtenido	El proceso de enrolamiento se realizó con éxito, los datos fueron almacenados en la nube.
Errores asociados	Ninguno.
Responsable diseño	Andrés Tisalema
Responsable ejecución	Andrés Tisalema
Comentarios	

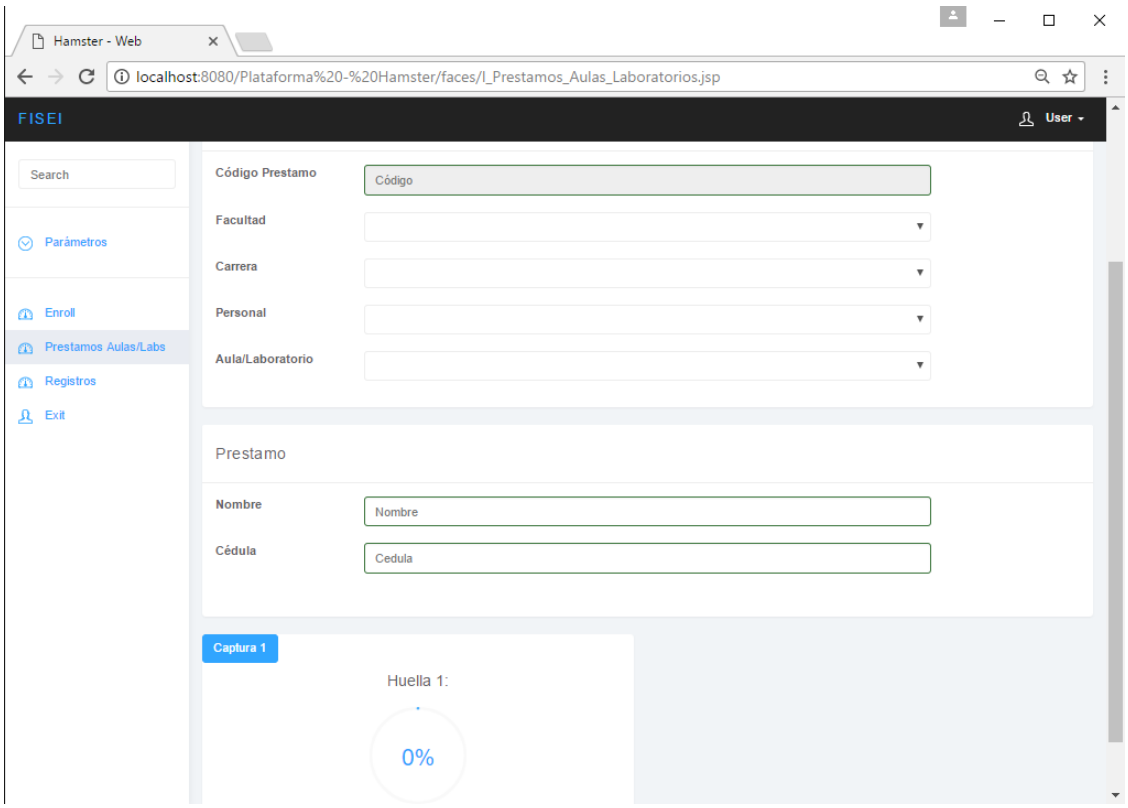


Figura 27: Sistema Web - Proceso de prestamos de aulas y laboratorios.
Fuente: Autor.

Tabla 69: Caso de Prueba: Solicitud Préstamo.

Fuente: Autor.

Criterio	Observaciones
ID	CP-02
Nombre	Proceso de solicitud de préstamos Aulas/Laboratorios
Descripción	Se probará la respuesta del sistema cuando el personal universitario solicite el préstamo de aulas o laboratorios.
Precondiciones	El usuario debe pertenecer al personal universitario.
Pasos y condiciones ejecución	1. El sistema solicita los datos del laboratorista encargado de las aulas y laboratorios. 2. El sistema solicita los datos personales del usuario. 3. El sistema valida que el usuario pertenezca a la Universidad. 4. El sistema solicita la huella dactilar del usuario para el préstamo. 5. La calidad de la huella dactilar obtenida debe ser mayor al 60%.
Resultado esperado	Se realiza el registro de los datos personales del usuario que solicita el préstamo y del laboratorista de turno en la nube.
Estado	Ejecutado
Resultado obtenido	Los datos se almacenaron en la nube para el proceso de préstamos de aulas y laboratorios.
Errores asociados	Ninguno.
Responsable diseño	Andrés Tisalema
Responsable ejecución	Andrés Tisalema
Comentarios	

Figura 28: Sistema Embebido - Proceso de registro de ingreso a aulas y laboratorios.

Fuente: Autor.

Tabla 70: Caso de Prueba: Registro ingreso docentes.

Fuente: Autor.

Criterio	Observaciones
ID	CP-03
Nombre	Registro Ingreso Aulas/Laboratorios Docentes
Descripción	Se probará la respuesta del sistema cuando el docente registre su entrada al aula o laboratorio.
Precondiciones	El usuario debe estar registrado, y debe pertenecer al personal docente.
Pasos y condiciones ejecución	1. El sistema solicita la huella dactilar del docente. 2. El docente ingresa su huella dactilar. 3. El sistema verifica y valida la identidad del usuario, además del aula y laboratorio asignado a su horario de clases.
Resultado esperado	Se registra el ingreso del docente, y se habilita el acceso al aula o laboratorio.
Estado	Ejecutado.
Resultado obtenido	Se habilitó el acceso al laboratorio y se registró el ingreso del docente.
Errores asociados	Ninguno.
Responsable diseño	Andrés Tisalema
Responsable ejecución	Andrés Tisalema
Comentarios	

Tabla 71: Caso de Prueba: Registro ingreso a prestamo de aulas/laboratorios.

Fuente: Autor.

Criterio	Observaciones
ID	CP-04
Nombre	Prestamos Aulas/Laboratorios
Descripción	Se probará la respuesta del sistema cuando el usuario registre su entrada al aula o laboratorio prestado.
Precondiciones	El usuario debe estar registrado para el proceso de préstamos.
Pasos y condiciones ejecución	1. El sistema solicita la huella dactilar del usuario. 2. El usuario ingresa su huella dactilar. 3. El sistema verifica y valida si la huella del usuario se encuentra registrada para el proceso de préstamos, además del aula y laboratorio asignado a su préstamo.
Resultado esperado	Se registra el ingreso del usuario, y se habilita el acceso al aula o laboratorio.
Estado	Ejecutado
Resultado obtenido	Se habilitó el acceso al laboratorio asignado para el préstamo y se registró el ingreso del usuario.
Errores asociados	Ninguno.
Responsable diseño	Andrés Tisalema
Responsable ejecución	Andrés Tisalema
Comentarios	

Tabla 72: Caso de Prueba: Ingreso oficinas de investigación.

Fuente: Autor.

Criterio	Observaciones
ID	CP-05
Nombre	Ingreso a oficinas de investigación
Descripción	Se probará la respuesta del sistema cuando el personal de Investigación registre su entrada a las oficinas de investigación.
Precondiciones	El usuario debe estar registrado, y debe pertenecer al personal de Investigación o Docente/Investigador.
Pasos y condiciones ejecución	1. El sistema solicita la huella dactilar del usuario. 2. El usuario ingresa su huella dactilar. 3. El sistema verifica y valida la identidad del usuario, además de la oficina a la cual pertenece.
Resultado esperado	Se registra el ingreso del usuario, y se habilita el acceso a la oficina.
Estado	Ejecutado
Resultado obtenido	Se habilitó el acceso a la oficina asignada al investigador y se registró su ingreso.
Errores asociados	Ninguno.
Responsable diseño	Andrés Tisalema
Responsable ejecución	Andrés Tisalema
Comentarios	

4.6. Integración del Sistema de Control al proyecto "Plataforma CLOUDIOT de Control y Monitoreo del Uso de Equipamiento y Programas Informáticos en Aulas y Laboratorios".

La integración del sistema embebido IoT permite el control y monitoreo del uso de aulas y laboratorios por parte del personal de la facultad. Información que se puede obtener a través de la aplicación web, figuras 29 y 30.

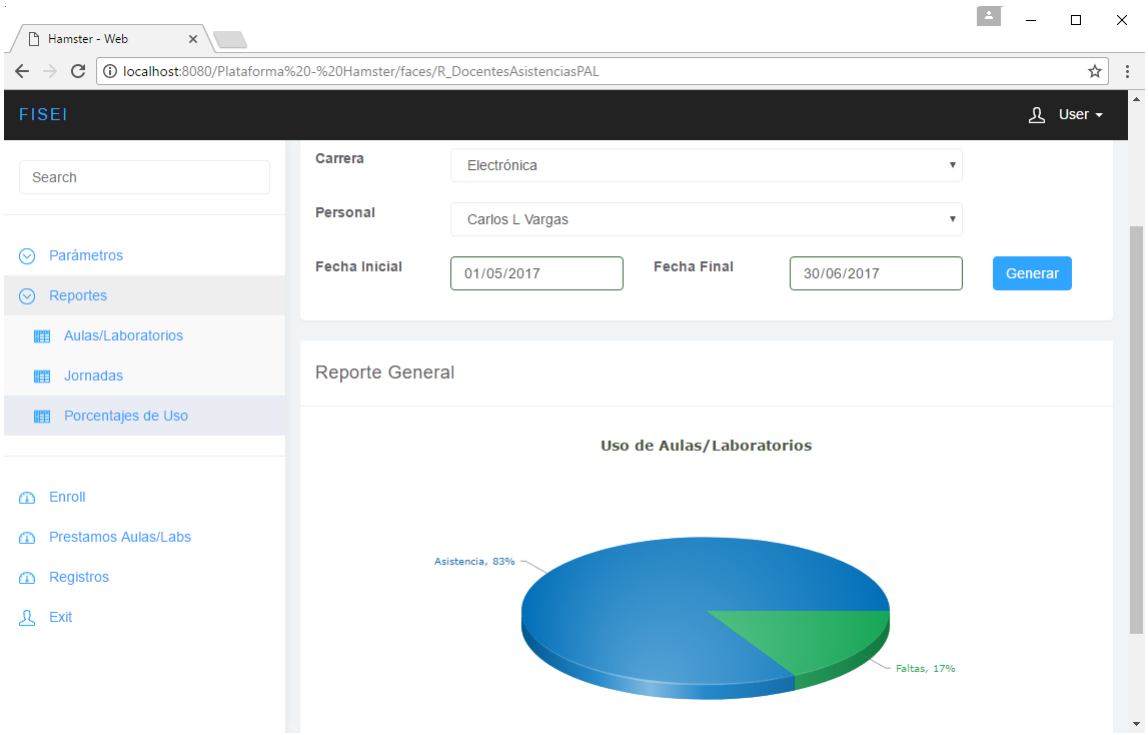


Figura 29: Sistema Web - Porcentaje del uso de aulas y laboratorios (a).
Fuente: Autor.

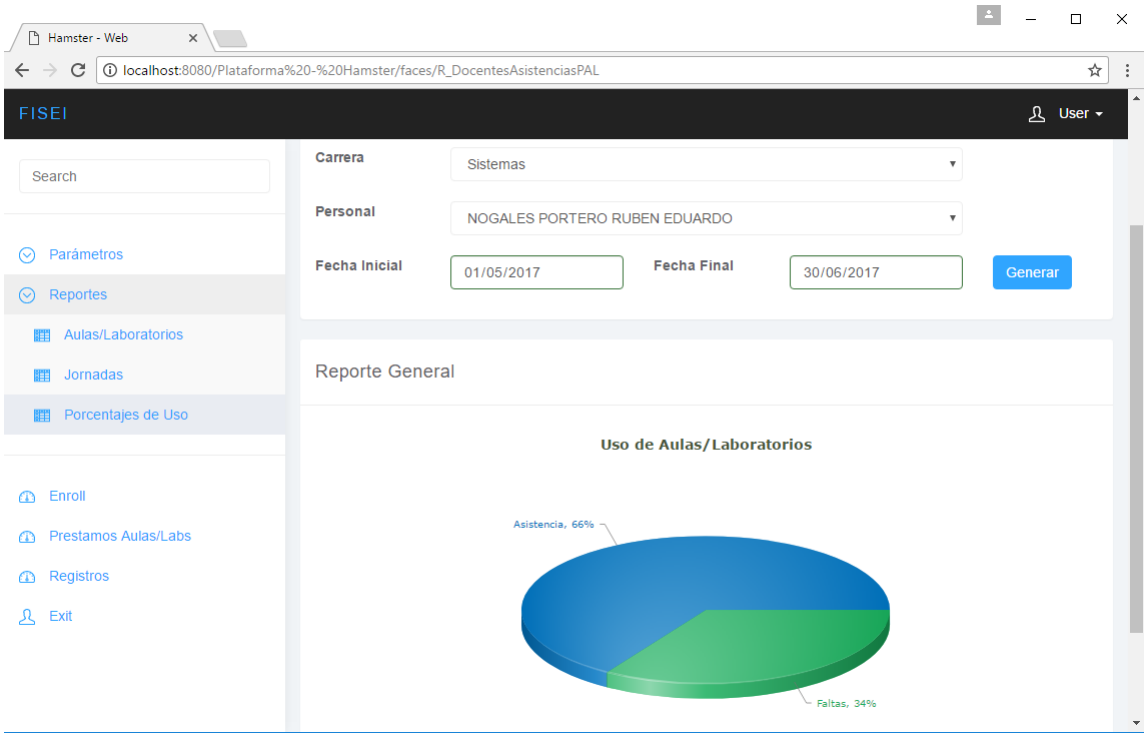


Figura 30: Sistema Web - Porcentaje del uso de aulas y laboratorios (b).
Fuente: Autor.

Tiempos de Respuesta.

El sistema de control, en los procesos de verificación de identidad y validación de permisos para el ingreso a aulas y laboratorios obtuvo los tiempos de respuesta, mostrados en la figura 31.

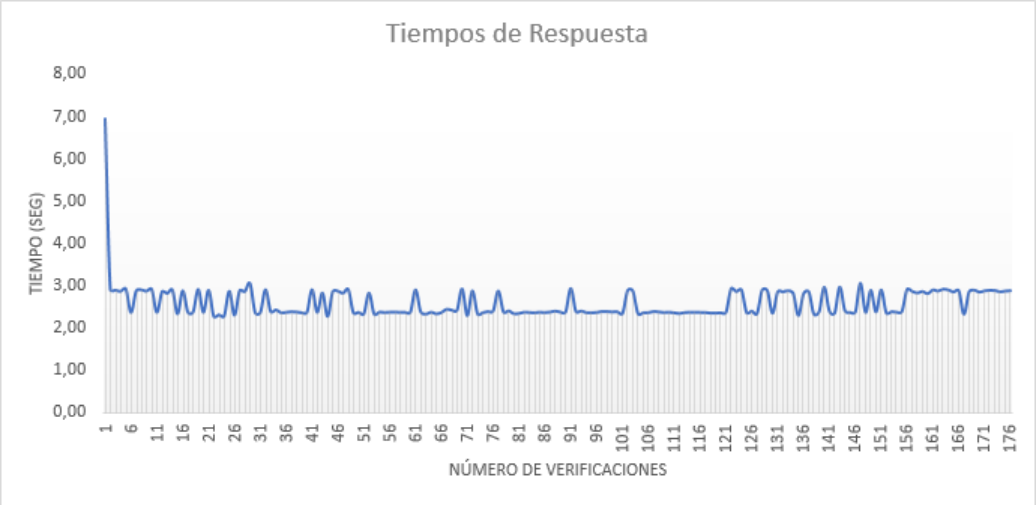


Figura 31: Tiempos de respuesta del sistema de control.
Fuente: Autor.

CAPÍTULO 5

Conclusiones y Recomendaciones

5.1. Conclusiones

- El código de programación en el desarrollo de software sin la utilización de reglas de programación no es flexible, no se adaptan a cambios de requerimientos. La aplicación de reglas de programación limpia al sistema de control, proporcionó características como: la mejora en la legibilidad del código de programación, el código es mucho más expresivo, claro y simple, una forma correcta en la creación de clases, métodos y funciones que utiliza el sistema de control.

- El sistema de control desarrollado optimiza el proceso de préstamos de aulas y laboratorios al personal de la facultad, remplazando al método manual, y eliminando retardos generados por dicho proceso.

- El sistema de control restringe el acceso a personas no autorizadas a aulas y laboratorios de la facultad, a través de la identificación del usuario por medio de su huella dactilar, almacenada conjuntamente con sus datos personales en una base de datos en la nube, mejorando el nivel de seguridad.

- Con un valor de retardo máximo de respuesta de 6,93 segundos, un valor mínimo de 2,27 segundos y un promedio aproximado de 2.59 segundos en el tiempo de respuesta, El sistema de control se muestra ágil en el control de acceso a aulas y laboratorios.

- La integración del sistema al proyecto de investigación, permite llevar un control y monitoreo del uso de aulas y laboratorios de la facultad. El sistema de control registra los siguientes procesos:

    1. Ingreso del personal docente a aulas y laboratorios asignados según su horario de clase.
    2. Solicitud de préstamo de aulas y laboratorios, por parte del personal perteneciente a la facultad.
    3. Ingreso a las oficinas de investigación de la facultad, por parte del personal de investigación y docentes investigadores.

En este contexto, el sistema es una fuente de consulta que permite medir el porcentaje de uso que tienen las aulas y laboratorios de la facultad.

5.2. Recomendaciones

- En el mundo del IoT (Internet of Things), servicios y aplicaciones son requeridos de una forma constante, y necesitan contar con un alto nivel de disponibilidad, es recomendable la utilización de bases de datos distribuidas, de manera que el sistema de control ofrezca una capacidad de almacenamiento unificado.

- Para la realización del proceso de almacenamiento unificado, se recomendaría como trabajo futuro la utilización de bases de datos embebida en los dispositivos raspberry pi. En la base de datos embebida se almacenarán los datos cuando el sistema no presente conexión a la nube, y cuando se establezca la conectividad se iniciará la sincronización a través de un proceso automático de replicación de datos desde la base de datos local o embebida hacia la base de datos en la nube.

- Para un posible trabajo futuro se recomendaría la migración del sistema embebido que se ejecuta en el dispositivo raspberry pi hacia el modelo de distribución de software como servicio, SaaS (Software as a Service). El sistema para el dispositivo raspberry pi podrá ser accedido a través de una API accesible por internet, eliminado el proceso de instalación en cada uno de los dispositivos raspberry pi, y agilizando el proceso de mantenimiento.

Bibliografia

[1] J. Hoyos, C. Madrigal, and J. Ramirez, "Diseño de un sistema biométrico de identificación usando sensores capacitivos para huellas dactilares.," *Universidad de Antioquia*, Oct. 2008.

[2] C. Yañez, L. Pro, and J. Gonzales, "Tecnologías biométricas aplicadas a la seguridad en las organizaciones," *Universidad Politécnica de Madrid*, Dec. 2009.

[3] L. Güette and M. Arocha, "Diseño de un sistema inmótico mediante el uso del sistema embebido intel galileo," *IV Simposio Científico y Tecnológico en Computación*, 2016.

[4] "Modelo de evaluación institucional de universidades y escuelas politécnicas," in *Consejo de Evaluación, Acreditación y Aseguramiento de la Calidad de la Educación Superior*, 2015.

[5] D. Mery and N. Kumar, "Formal specification of medical systems by proof-based refinement.," *ACM Trans. Embedded Comput. Syst.*, 2013.

[6] V. Kumar, G. Senthilumar, and K. Gopalakrishnan, "Embedded image capturing system using raspberry pi system," *International Journal Of Emerging Trends of Technology in Computer Science*, vol. 3, ISSN 2278-6856, 2014.

[7] S. Mansoor and J. S. U, "Raspberry pi based home door security throught 3g dongle," *International Journal of Engineering Research and General Science*, vol. 3, ISSN 2091-2730, 2015.

[8] F. S. Rodríguez, "Patrones de diseño para el desarrollo de software basados en la herramienta case genexus," *Universidad del Azuay, Titulación de Ingeniería*, 2008.

[9] J. G. Morocho, "Análisis del patrón modelo vista controlador implementado en lenguajes de software libre para el desarrollo de aplicaciones web. caso practico: Liceo de talentos stephen hawking," *Escuela Superior Politécnica de Chimborazo, Titulación de Ingeniería*, 2010.

[10] L. L. Taramuel, "Desarrollo e implementación de un aplicativo web para la gestión de concursos de la asociación de caballos de paso utilizando patrones de diseño modelo-vista-controlador," *Universidad Técnica Particula de Loja, Titulación de Ingeniería*, 2011.

[11] C. V. Ramírez, "Sistema de control de acceso de personal para los laboratorios de la carrera de ingeniería en sistemas computacionales de la universidad de guayaquil utilizando tecnoligía nfc," *Universidad de Guayaquil, Titulación de Ingeniería*, 2015.

[12] J. Cortés, F. Medina, and J. Muriel, "Sistemas de seguridad basados en biometría," *Scientia et Technica Año XVII*, pp. 98–102, Dec. 2010.

[13] H. Vega and J. Isidro, "El software embebido y los retos que implica su desarrollo," *Conciencia Tecnológica, Mexico*, pp. 42–45, Dec. 2010.

[14] C. Alexander, "A pattern lenguege," *Oxford University*, vol. 4, 1997.

[15] J. Gracia, "Patrones de diseño, ingeniería de software, análisis y diseño," *Universidad Católica de Salta*, 2010.

[16] E. Braude, "Ingeniería del software una perspectiva orientada a objetos," *EDITORIAL. Madrid, España*, 2003.

[17] I. Jacobson, G. Booch, and J. Rumbaugh, "The unified software development process," *Technische Universital Fachbereich Informatik*, vol. 1, 1999.

[18] J. Lazalde and A. Tórres, "Hardware: ecosistemas de innovación y producción basados en hardware libre (v.2.0). buen conocer - flok society, modelos sostenibles y políticas públicas para una economía social del conocimiento común y abierto en el ecuador.," *Universidad Iberoamericana, Campus Santa Fe, México.*, vol. 2, 2015.

[19] O. Pastrana, "Beneficios de aplicar metodologías Ágiles en el desarrollo de software," 2014.

[20] A. Navarro, J. Fernandéz, and J. Morales, "Revisión de metodologías ágiles para el desarrollo de software," *Prospect.*, vol. 11, no. 2, pp. 30–39, 2013.

[21] J. O. Coplien, *Clean Code: A Handbook of Agile Software Craftsmanship.* Pearson Education, Inc., 2009.

[22] M. Orion, "Testing legacy embedded code: Landing on a software engineering desert island," *IEEE*, 2015.

[23] E. Ammerlaan, W. Veninga, and A. Zaidman, "Old habits die hard: Why refactoring for understandability does not give immeediate benefits," *Saner 2015, Candá - IEEE*, 2015.

[24] C. Guerrero, J. Suarez, and L. Gutierrez, "Patrones de diseño gof (the gand of four) en el contexto de procesos de desarrollo de aplicaciones orientadas a la web," *SciELO*, 2013.

[25] E. RÌos and W. Suntaxi, "Desarrollo de un sistema inform·tico para los procesos de cosecha y post cosecha de la camaronera pampas de cayanca," *Escuela Politécnica Nacional, Quito, Titulación de Ingeniería*, 2008.

[26] A. Orjuela and M. Rojas, "Las metodologías de desarrollo Ágil como una oportunidad para la ingeniería del software educativo," *Revista Avances en Sistemas e Informática*, vol. 5, no. 2, pp. 159–171, 2008.

[27] S. Corporation, *Case Design for Optimal Fingerprint Scanning and Optimal Auto-On Finger Detection.*, 2011.

Anexos y Apéndices

Anexo A

Manual de Usuario del Sistema Embebido Cloud IoT.

Anexo B

Horarios del uso de laboratorios.