UTMACH

UNIDAD ACADÉMICA DE INGENIERÍA CIVIL

CARRERA DE INGENIERÍA DE SISTEMAS

DESARROLLO DE APLICACIÓN WEB PARA GESTIÓN DE EMPLEADOS USANDO METODOLOGÍA SNAIL Y LENGUAJE DE PROGRAMACIÓN JAVA.NET

AGUILAR NOLE LUIS FERNANDO
INGENIERO DE SISTEMAS

MACHALA
2019

UTMACH

UNIDAD ACADÉMICA DE INGENIERÍA CIVIL

CARRERA DE INGENIERÍA DE SISTEMAS

DESARROLLO DE APLICACIÓN WEB PARA GESTIÓN DE EMPLEADOS USANDO METODOLOGÍA SNAIL Y LENGUAJE DE PROGRAMACIÓN JAVA.NET

AGUILAR NOLE LUIS FERNANDO
INGENIERO DE SISTEMAS

MACHALA
2019

UTMACH

UNIDAD ACADÉMICA DE INGENIERÍA CIVIL

CARRERA DE INGENIERÍA DE SISTEMAS

EXAMEN COMPLEXIVO

DESARROLLO DE APLICACIÓN WEB PARA GESTIÓN DE EMPLEADOS USANDO METODOLOGÍA SNAIL Y LENGUAJE DE PROGRAMACIÓN JAVA.NET

AGUILAR NOLE LUIS FERNANDO
INGENIERO DE SISTEMAS

HONORES TAPIA JOOFRE ANTONIO

MACHALA, 01 DE FEBRERO DE 2019

MACHALA
01 de febrero de 2019

Nota de aceptación:

Quienes suscriben, en nuestra condición de evaluadores del trabajo de titulación denominado DESARROLLO DE APLICACIÓN WEB PARA GESTIÓN DE EMPLEADOS USANDO METODOLOGÍA SNAIL Y LENGUAJE DE PROGRAMACIÓN JAVA.NET, hacemos constar que luego de haber revisado el manuscrito del precitado trabajo, consideramos que reúne las condiciones académicas para continuar con la fase de evaluación correspondiente.

HONORES TAPIA JOOFRE ANTONIO
0704811751
TUTOR - ESPECIALISTA 1

MOLINA RIOS JIMMY ROLANDO
0703691980
ESPECIALISTA 2

JUMBO CASTILLO FREDDY ANIBAL
0704167949
ESPECIALISTA 3

Fecha de impresión: viernes 01 de febrero de 2019 - 13:45

URKUND

Urkund Analysis Result

Analysed Document: Caso practico Luis Aguilar.docx (D46888844)
Submitted: 1/17/2019 1:05:00 AM
Submitted By: lfaguilarn_est@utmachala.edu.ec
Significance: 6 %

Sources included in the report:

TrabajoComplexivo-Castillo Jimmy .docx (D40217812)
Tesis Marco Tunala.docx (D40657823)

Instances where selected sources appear:

8

CLÁUSULA DE CESIÓN DE DERECHO DE PUBLICACIÓN EN EL REPOSITORIO DIGITAL INSTITUCIONAL

El que suscribe, AGUILAR NOLE LUIS FERNANDO, en calidad de autor del siguiente trabajo escrito titulado DESARROLLO DE APLICACIÓN WEB PARA GESTIÓN DE EMPLEADOS USANDO METODOLOGÍA SNAIL Y LENGUAJE DE PROGRAMACIÓN JAVA.NET, otorga a la Universidad Técnica de Machala, de forma gratuita y no exclusiva, los derechos de reproducción, distribución y comunicación pública de la obra, que constituye un trabajo de autoría propia, sobre la cual tiene potestad para otorgar los derechos contenidos en esta licencia.

El autor declara que el contenido que se publicará es de carácter académico y se enmarca en las dispociones definidas por la Universidad Técnica de Machala.

Se autoriza a transformar la obra, únicamente cuando sea necesario, y a realizar las adaptaciones pertinentes para permitir su preservación, distribución y publicación en el Repositorio Digital Institucional de la Universidad Técnica de Machala.

El autor como garante de la autoría de la obra y en relación a la misma, declara que la universidad se encuentra libre de todo tipo de responsabilidad sobre el contenido de la obra y que asume la responsabilidad frente a cualquier reclamo o demanda por parte de terceros de manera exclusiva.

Aceptando esta licencia, se cede a la Universidad Técnica de Machala el derecho exclusivo de archivar, reproducir, convertir, comunicar y/o distribuir la obra mundialmente en formato electrónico y digital a través de su Repositorio Digital Institucional, siempre y cuando no se lo haga para obtener beneficio económico.

Machala, 01 de febrero de 2019

AGUILAR NOLE LUIS FERNANDO
0706022480

DEDICATORIA

Primeramente, a Dios, por brindarme la salud y fuerza necesaria para lograr cumplir todas mis metas propuestas durante la duración del periodo de mi carrera.

A mis padres, aquellos que me dieron la vida y siempre están ahí cuando se los necesita, tanto en momentos malos como en los buenos, resaltando todo su apoyo, consejos y ánimos entregados hacia mí día tras día.

A mi tutor, Ing. Joofre Honores, que, a través de sus consejos, orientaciones y amplios conocimientos, supo cómo prepararme y superar muchos obstáculos durante el proceso de mi trabajo de titulación.

Finalmente, pero no menos importante, a la Universidad Técnica de Machala, que en conjunto con la Unidad Académica de Ingeniería Civil y la carrera de Ingeniería de Sistemas, agradecerles por brindarme la oportunidad de forjarme como profesional y prepararme para desenvolverme en el campo laboral.

Sr. Aguilar Nole Luis Fernando.

AGRADECIMIENTO

Agradezco, primeramente, ante todo a Dios, el cual durante todo el transcurso de mi vida me ha dado fuerza, salud y me ha guiado por el camino del bien tanto en las cosas que me he propuesto realizar y en las decisiones que se me han presentado en mi convivir diario.

Agradezco a mi familia, los cuales son los seres más importantes en mi vida, ellos supieron criarme con los mejores valores y me han brindado sus apoyos tantos emocionales como económicos.

Finalmente agradezco a la Universidad Técnica de Machala por darme la oportunidad de cursar mis estudios en una buena institución educativa, al igual agradezco a mis profesores que me acompañaron durante todo el lapso de mi formación como profesional, especialmente a mi tutor Ing. Joofre Honores, por su dedicación, conocimientos y apoyo hacia mí, durante sus tutorías.

Sr. Aguilar Nole Luis Fernando

RESUMEN

DESARROLLO DE APLICACIÓN WEB PARA GESTIÓN DE EMPLEADOS USANDO METODOLOGÍA SNAIL Y LENGUAJE DE PROGRAMACIÓN JAVA.NET

Aguilar Nole Luis Fernando, 0706022480

El presente proyecto detalla el DESARROLLO DE APLICACIÓN WEB PARA GESTIÓN DE EMPLEADOS USANDO METODOLOGÍA SNAIL Y LENGUAJE DE PROGRAMACIÓN JAVA.NET, el cual tiene como objetivo, el gestionamiento y control de sus empleados por parte de la empresa multinacional “XYZ”. Para el diseño y desarrollo de la aplicación se utilizó el framework de desarrollo web en JAVA.NET; el Sistema Gestor de Base de Datos PostgreSQL, el modelo de arquitectura de software modelo-vista-controlador, permitiendo al administrador del sitio, gestionar las asistencias, horarios, salarios, cargos, sanciones y recepción del nuevo personal; además de gestionar datos de la empresa. Las herramientas usadas en este proyecto poseen características dinámicas y junto con una robusta documentación ayudaron a desarrollar de una manera rápida la aplicación web. Como metodología de desarrollo de software se planteó la SNAIL (Software Nativo de Arquitectura Iterativa Lógica) metodología idónea por promover el desarrollo ágil, permitir la resolución a corto tiempo y muy adaptable a un limitado equipo de trabajo; además cabe destacar que todas las herramientas utilizadas son de tipo Open Source que otorgan al proyecto un reducido costo, rentabilidad y gran soporte en documentación. Al final se obtuvo una aplicación web con comunicación a base de datos relacional que permite gestionar la información de la empresa y sus empleados.

PALABRAS CLAVES: Aplicación Web, Arquitectura Modelo-Vista-Controlador, Base de datos relacional PostgreSQL, Java.Net, Metodología Híbrida SNAIL.

ABSTRACT

DEVELOPMENT OF WEB APPLICATION FOR EMPLOYEE MANAGEMENT USING SNAIL METHODOLOGY AND JAVA.NET PROGRAMMING LANGUAGE

Aguilar Nole Luis Fernando, 0706022480

This project details the DEVELOPMENT OF WEB APPLICATION FOR MANAGEMENT OF EMPLOYEES USING SNAIL METHODOLOGY AND JAVA.NET PROGRAMMING LANGUAGE, which has as objective, the management and control of its employees by the multinational company "XYZ". For the design and development of the application, the web development framework was used in JAVA.NET; the PostgreSQL Database Management System, the model of software architecture model-view-controller, allowing the site administrator to manage assistance, schedules, salaries, charges, sanctions and reception of new personnel; In addition to managing company data. The tools used in this project have dynamic characteristics and together with a robust documentation they helped to develop the web application in a fast way. As software development methodology, the SNAIL (Native Software of Logical Iterative Architecture) was proposed, an ideal methodology to promote agile development, allow short-term resolution and very adaptable to a limited work team; It should also be noted that all the tools used are of Open Source type that give the project a low cost, profitability and great documentation support. In the end, a web application with relational database communication was obtained that allows managing the information of the company and its employees.

KEYWORDS: Web Application, Model-View-Controller Architecture, PostgreSQL relational database, Java.Net, SNAIL Hybrid Methodology.

CONTENIDO

pág.

ÍNDICE DE TABLAS

ÍNDICE DE ILUSTRACIONES

1. INTRODUCCIÓN

Desde la aparición de los ordenadores, todos los programas o aplicaciones que permanecían instaladas pertenecían a la arquitectura stand-alone (ser único), es decir, software que funcionaban dentro de un mismo ordenador con objetivos donde solo un usuario y una computadora podían realizar, esto, hasta el nacimiento del internet y las redes, después de este acontecimiento fueron surgiendo nuevos modelos de arquitecturas, hasta llegar al punto de encontrar programas alojados en nubes, en los cuales millones de usuarios, tienen la posibilidad de interactuar entre sí, al mismo tiempo, sin importar el sitio físico, sin la necesidad que el usuario tenga conocimiento de esta infraestructura.[1]

Mientras se desarrollaban las redes de comunicación, nace la probabilidad de una multiconexión de clientes a servidores, de esta forma dando paso al intercambio de información a una gran velocidad y es aquí donde nace el concepto de los sistemas distribuidos siendo la arquitectura modelo-vista-controlador una de las nacientes ideas en el campo de GUI's, además de ser uno de los principales trabajos en describir y realizar aplicaciones software en términos de sus diferentes funciones.

La Arquitectura Modelo Vista Controlador (MVC), está conformada por tres capas; en la capa Modelo se encuentran los datos de la aplicación, en la capa Vista es donde se encuentras las GUI's (Interfaz Gráfica de Usuario) y por último la capa de Controladores es la encarga de conectar los datos con la interfaz gráfica de usuario, además de proporcionar las reglas y restricciones que dicha aplicación tendrá.[2]

El objetivo de este proyecto es desarrollar una aplicación web para la gestión de empleados haciendo uso de la metodología SNAIL y lenguaje de programación Java.Net, por ende, fue necesario recopilar información en distintas fuentes de datos como artículos científicos relacionados a dicho tema, una vez efectuado esto se inició con el análisis de las diversas tecnologías investigadas y seleccionar las más factibles para la resolución de dicho proyecto, aplicación que contendrá un menú dinámico dependientemente de los permisos y roles que el usuario ingresado posea, además de una interfaz intuitiva y amigable.

1.1 Marco Contextual

La mayor preocupación de todo gerente o jefe de área o departamento ha sido, es y será la supervisión y control del personal a su cargo, dicha supervisión es llevada en forma de registro de entradas y salidas de cada empleado, de tal forma que la empresa pueda tener un mayor control del desempeño laboral y de las actividades con las cuales los empleados deben de cumplir.

La empresa multinacional “XYZ”, en la cual actualmente están laborando más de 500 empleados, los cuales tienen un control de asistencia, horarios, salarios, cargos, sanciones y la recepción del nuevo personal, dichos controles no se encuentran automatizados actualmente, por lo cual, las instalaciones de dicha empresa será el sitio donde se dará solución al problema de este caso práctico.

1.2 Problema

La empresa multinacional “XYZ” necesita gestionar correctamente su personal de forma automatizada para no incurrir en multas, dado que los controles implementados se realizan de forma manual, donde cada empleado tiene que firmar de manera diaria dependientemente del departamento donde labore, pero dicha acción no asegura que el empleado ingreso al departamento a realizar sus labores, de la misma manera se encuentran las sanciones y salarios, por estos motivos, la empresa necesita un sistema web que gestione dichos controles en tiempo real, con el único objetivo de no incurrir en multas innecesarias.

1.3 Objetivo general

Desarrollar una aplicación web para la gestión de empleados usando metodología SNAIL y lenguaje de programación Java.Net

2. DESARROLLO

2.1 Marco teórico

2.1.1 Arquitectura: Modelo, Vista, Controlador (MVC)

Esta arquitectura es muy utilizada por los desarrolladores ya que posee la ventaja de separar o dividir los componentes de la aplicación que se esté desarrollando, dando a cada uno una determinada responsabilidad, [3] esto proporciona la ventaja de que, si se realiza un cambio en uno de los componentes, dicho cambio no afectará el código de otro componente. [4]

La ventaja de usar esta arquitectura es que se puede realizar de forma modular y sus vistas muestran información actualizada siempre.[5]

2.1.2 Apache

El servidor Web Apache al ser de código abierto, además de tener la ventaja de ser multiplataforma, escalable y dinámico[6]; es el más factible y elegido para trabajar a nivel mundial, cabe resaltar que fue diseñado de manera especial para la transferencia de archivos, ya que utiliza el protocolo de comunicación http[7]

2.1.3 CentOS 7

Es uno de los sistemas operativos perteneciente a Linux, creado con el único objetivo de servir como servidor web[6], cabe resaltar que dicho sistema operativo es de código abierto y seguro, haciéndolo preferente por grandes y medianas empresas.[8]

2.1.4 Base de Datos relacional PostgreSQL

PostgreSQL se encuentra entre las 10 DBMS más usadas en todo el mundo, ya sea dentro del área de las telecomunicaciones como big data, cloud computing, entre otras, así mismo dicho DBMS es muy utilizado dentro del área científica, gracias a la característica que posee de almacenar una abundante cantidad de información, además de la gran velocidad a la que responde cada consulta. [9]

2.1.5 Lenguaje de programación JAVA.NET

Java.Net es un paquete o API (application programming interface o Interfaz de programación de aplicaciones en su traducción al español) de Java, con el cual es posible la conexión e intercambio de información por medio de la red. Gracias a esta API se puede establecer la comunicación entre dos o más computadoras sin importar la distancia física que existe.[10]

2.1.6 Framework Bootstrap 4

Marco de trabajo creado con el único fin de establecer diseños responsivos para el usuario, ya que posee una infinidad de técnicas y modelos que facilitan el trabajo al programador en cuanto al diseño.[6] Gracias a este marco de trabajo es posible crear aplicaciones web que se adapten a distintos tipos de dispositivos como tablets, celulares, entre otros.[11]

2.1.7 Lenguaje de programación HTML (HyperText Markup Language)

HTML (Lenguaje de Marcas de Hipertexto) es un lenguaje de programación para páginas web muy simple, gracias a esto, este lenguaje es muy usado para dicho fin, cabe resaltar que HTML tiene sus bases en el protocolo HTTP, gracias a este protocolo es posible el envío de información desde un servidor y dicha información puede ser mostrada en un archivo HTML desde un navegador.[12]

2.1.8 HTTP

Protocolo basado en el paradigma petición-respuesta, creado con la finalidad de transferir información de manera eficiente solicitada entre computadoras, puede ser entre servidores web y clientes.[13]

2.1.9 Java 8.

Es la versión más actual que posee Java, esta versión posee una variedad de mejoras en comparación a la versión anterior (Java 7). Java es un lenguaje que está fundamentado en el paradigma orientado a objetos, gracias a esto es posible la creación de una sintaxis elegante y

limpia de tal forma haciendo que Java sean uno de los lenguajes más usados para la investigación y educación. [14]

2.1.10 Estructura y manejo de datos.

La estructura de datos es la forma más sencilla que tiene una computadora para organizar grandes cantidades de datos, más aun cuando éstos son complejos, dicha acción es realizada mediante diferentes tipos de estructuras con el único fin de hacer más eficiente el control de datos.[15]

2.1.11 Metodología Híbrida SNAIL (Software Nativo de Arquitectura Iterativa Lógica)

Las metodologías de desarrollo hibridas, se caracterizan por tener practicas existentes tanto en metodologías tradicionales como en las agiles, gracias a esto obtiene una gran ventaja, por lo cual seleccionar una metodología hibrida es lo más adecuado, más aún cuando dicha metodología fue creada para el desarrollo de aplicaciones web, SNAIL está basada en lo sencillez, comunicación y planificación del código a desarrollarse. Dicha metodología posee un modelo de fases en espiral, convirtiéndola así, en una metodología ligera, pero abarca gran parte del ciclo clásico del desarrollo de una aplicación web.

SNAIL no tiene mucha diferencia en cuanto a metodologías ya creadas anteriormente para el desarrollo de aplicaciones web, ya que las fases de dichas metodologías son muy similares a las fases de la metodología SNAIL. [16]

2.2 Solución del problema

2.2.1 Materiales

Los recursos o materiales empleados para el desarrollo de este proyecto son:

Tabla 1: Recursos empleados

Materiales	
Lenguaje de desarrollo	Java.Net (Java 8)
Marco de trabajo (Framework)	Hibernate 9 y Spring
Gestor de base de datos	PostgreSQL 9.4
Metodología empleada	SNAIL
Aplicaciones	Aplicación de escritorio: Java 8, Aplicación Web: HTML

Fuente: Luis Fernando Aguilar Nole

2.2.2 Metodología Híbrida de desarrollo SNAIL

2.2.2.1 Fase 1: Análisis de requerimientos

En esta fase se llevaron a cabo consultas no formales a los futuros usuarios con el objetivo de tener de forma clara todos y cada uno de los requerimientos, ya que dichos requerimientos son el punto de inicio para el desarrollo de la aplicación web, misma que se puede apreciar en el siguiente apartado (Anexo A).

Después, se plantearon prototipos de interfaz gráfica de usuarios, cabe destacar que dichos prototipos posiblemente cambien a medida que pase el tiempo, todo depende de las peticiones o cambios que desee el usuario, estos prototipos son creados para que el usuario tenga una idea remota de su futuro sistema (Ver Anexo B).

2.2.2.2 Fase 2: Planificación

En esta fase se proporcionó un marco de trabajo el cual permitió hacer estimaciones razonables de recursos y entregables.

Estas estimaciones se hicieron dentro de un marco de tiempo limitado al iniciar un proyecto de software, y fueron actualizadas constantemente a medida que se avanzó el proyecto. También, las estimaciones definieron los escenarios del mejor caso, y peor caso, de modo que los resultados del proyecto fueron favorables.

2.2.2.3 Fase 3: Diseño

En esta fase se procedió a crear el código en el lenguaje de programación Java con el framework Hibernate y Spring, en cuanto a la base de datos, se la gestiono con PgAdmin ya que es el mejor gestor para base de datos PostgreSQL, las imágenes fueron realizadas en Photoshop 2019.

2.2.2.4 Fase 4: Programación

Ilustración 1: Método getEmpleado

```
public class EmpleadoModel {
    @RequestMapping()
    //Obtener Empleado
    public List<Empleado> getEmpleado(){
        List<Empleado> lst = new ArrayList<Empleado>();
        Session s = HibernateUtil.getSessionFactory().openSession();
        try{
            lst = s.createCriteria(Empleado.class).list();
            s.getTransaction().commit();
        }catch(Exception e){
            e.printStackTrace();
//              s.getTransaction().rollback();
        }
        return lst;
    }
```

Fuente: Luis Fernando Aguilar Nole

Este método fue creado con el único fin de obtener los datos de todos empleados y almacenarlos en un TAD (Tipo Abstracto de Datos) Lista, para de esta forma tener un mejor manejo de dichos datos.

Ilustración 2: Método crearEmpleado (Empleado)

```
//Crear Empleado
public void crearEmpleado(Empleado u){
    Session s = HibernateUtil.getSessionFactory().openSession();
    try{
        s.beginTransaction();
        s.save(u);
        s.getTransaction().commit();
    }catch(Exception e){
        e.printStackTrace();
        s.getTransaction().rollback();
    }
}
```

Fuente: Luis Fernando Aguilar Nole

Método cuyo objetivo es crear un empleado mediante el uso de una clase, ya que con dicha acción se reduce el uso de la base de datos, haciendo únicamente una consulta la cual almacenará todos los datos de un empleado.

Ilustración 3: Método EliminarEmpleado (String)

```
//Eliminar
public void eliminarEmpleado(Empleado u){
    Session s = HibernateUtil.getSessionFactory().openSession();
    try{
        s.beginTransaction();
        s.delete(u);
        s.getTransaction().commit();
    }catch(Exception e){
        e.printStackTrace();
        s.getTransaction().rollback();
    }
}
```

Fuente: Luis Fernando Aguilar Nole

Este método es el encargado de realizar un “Delete” dentro de la base de datos, ya que exige un dato de entrada, el cual debe ser el identificador del registro al cual se desea eliminar.

Ilustración 4: Método getEmpleado(String)

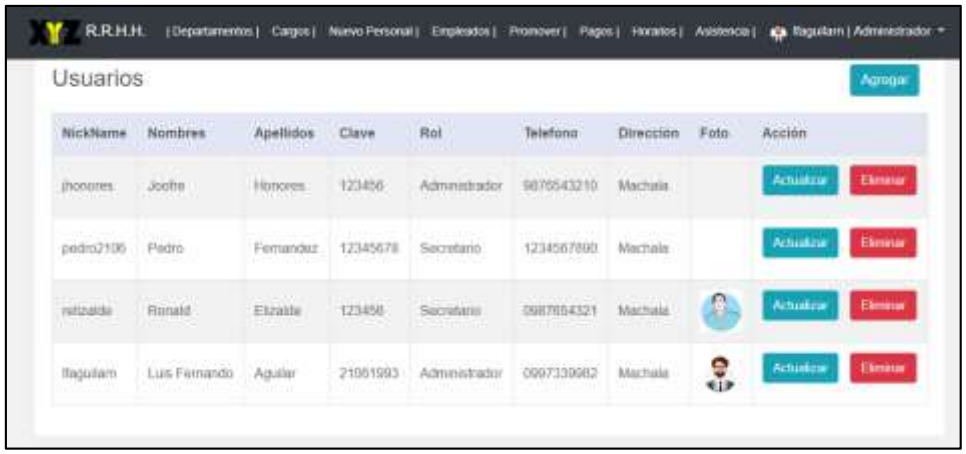
```
public Empleado getEmpleado(String username){
    Session s = HibernateUtil.getSessionFactory().getCurrentSession();
    Empleado user = new Empleado();
    try {
        s.beginTransaction();
        user =(Empleado) s.get(Empleado.class,username);
        s.getTransaction().commit();
    } catch (Exception e) {
        e.printStackTrace();
        s.getTransaction().rollback();
    }
    return user;
}
```

Fuente: Luis Fernando Aguilar Nole

Método encargado de la obtención de un solo empleado, mediante el ingreso de su identificador, dicho método hace uso de la base de datos de forma eficaz, tal es el caso que la base de datos no sufre cargos por este método.

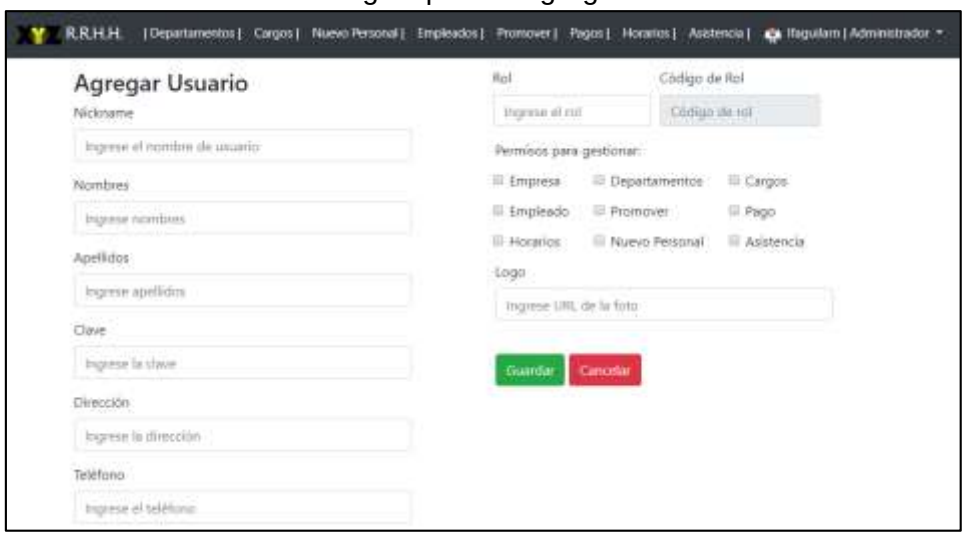
2.2.2.5 Fase 5: Pruebas

Ilustración 5: Página Usuarios



Página en la cual es posible la visión de todos los usuarios del sistema, así como de los roles que posee cada uno, además del beneficio de la actualización y la eliminación por cada registro, cabe destacar que solo al usuario que es Administrador le es posible ver dicha página.

Ilustración 6: Página para la agregación de usuarios



Página en la cual es posible realizar el registro de nuevos usuarios, así como de asignarle permisos y roles.

2.3 Resultados

Posterior a la utilización de varios argumentos descritos en este proyecto, los resultados son los siguientes:

En primera instancia se encuentra la realización de la aplicación web, la misma que será usada para la administración de este sitio web, cabe destacar que esto fue creado haciendo uso del framework Hibernate 9 y Spring, dentro del lenguaje de programación Java.Net, utilizando una arquitectura de 3 capas (MVC).

La aplicación web posee como seguridad de ingreso un login (Ver anexo C), dentro del cual el usuario deberá ingresar su nickname (nombre de usuario) y su contraseña, datos con los cuales el sistema hará el respectivo procedimiento para su ingreso al sistema y el menú se mostrará según los permisos y roles que dicho usuario tenga.

Si el usuario ingresado es un Administrador, entonces podrá apreciar el menú completo (ver Anexo D), dicho menú consta de todas las opciones que fue pedida por el usuario (Ver anexo E):

- Administración de departamentos.
- Administración de usuarios.
- Administración de cargos.
- Administración de postulaciones.
- Administración de empleados.
- Administración de promover.
- Administración de pagos.
- Administración de horarios.
- Administración de asistencia.

Cabe destacar que se hizo uso de estructuras de datos TAD (Tipo Abstracto de Datos), haciendo más sencillo el manejo de los algoritmos de listas. El TAD lista se implementó en cada uno de los módulos de administración, ya que dicha estructura hace más veloz el manejo e intercambio de datos.

3. CONCLUSIONES

Basándose en la investigación de artículos científicos y a la implementación de la metodología hibrida SNAIL, se consiguió el cumplimiento de los requerimientos que el cado practico propuso, dando como resultado o producto final una aplicación web que ayude en la administración de la asistencia, pagos, sanciones, horarios y entre otros controles más de los empleados de la empresa multinacional “XYZ”

Los TAD lista fueron utilizados en la resolución de este caso práctico, con el objetivo de optimizar el manejo e intercambio de datos.

La aplicación web permite un mejor control de la asistencia, pagos, sanciones, horarios y entre otros controles más de los empleados, y el poder abrir la página en un dispositivo móvil gracias al diseño responsivo que posee dicha aplicación, esto le hace más sencillo al administrador, tener un mayor control acerca de las actividades de sus empleados.

REFERENCIAS

[1] N. Lisbeth Hernandez Quintero and E. Anderson Smith Florez Fuente, “Computación En La Nube Cloud Computing,” p. 102, 2010.

[2] A. T. Espinosa, J. G. C. Sagredo, M. M. Reyes, and M. de L. L. García, “Automatización de la codificación del patrón modelo vista controlador (MVC) en proyectos orientados a la Web,” *Cienc. ergo-sum*, vol. 19, no. 3, pp. 239–250, 2012.

[3] J. R. Molina Ríos, N. M. Loja Mora, M. P. Zea Ordóñez, and E. L. Loaiza Sojos, “Evaluación de los Frameworks en el Desarrollo de Aplicaciones Web con Python,” *Rev. Latinoam. Ing. Softw.*, vol. 4, no. 4, p. 201, 2016.

[4] R. Cubana *et al.*, “Propuesta de arquitectura cliente de la aplicación de interfaz de usuario del sistema xavia ris 2 . 0 Proposal for client architecture for user interface application of systems ris xavia 2 . 0,” vol. 8, no. 1, pp. 30–45, 2016.

[5] J. M. Suárez and L. E. Gutiérrez, “Tipificación de Dominios de Requerimientos para la Aplicación de Patrones Arquitectónicos,” *Inf. Tecnol.*, vol. 27, no. 4, pp. 193–202, 2016.

[6] G. Vivian, L. Fernando, and I. De Sistemas, “DESARROLLO DE UN SISTEMA WEB Y MÓVIL BASADO EN LA TECNOLOGÍA CLIENTE SERVIDOR PARA LA GESTIÓN DE ASISTENCIA LABORAL,” 2018.

[7] M. Baş Seyyar, F. Ö. Çatak, and E. Gül, “Detection of attack-targeted scans from the Apache HTTP Server access logs,” *Appl. Comput. Informatics*, vol. 14, no. 1, pp. 28–36, 2018.

[8] M. R. López Vallejo, “Hacking ético. Vulnerabilidad de Sistemas Operativos en el acceso por contraseñas,” *Rev. Publicando*, no. 101, pp. 31–51, 2017.

[9] R. V Quijije-diaz, “Ciencias Informáticos Artículo Científico,” vol. 3, pp. 771–779, 2017.

[10] A. B. García, R. L. Espinosa, and J. M. M. Espinosa, “Enfoque semántico para el descubrimiento de recursos sensible al contexto sobre contenidos académicos estructurados con OAI-PMH y Java.Net,” *Comput. y Sist.*, vol. 20, no. 1, pp. 127–142, 2016.

[11] D. M. Tartar and V. R. Sharon, “Responsive web design diseño multidispositivo para mejorar la experiencia de usuario,” *Dermatol. Online J.*, vol. 23, no. 1, 2017.

[12] M. Zea and J. Molina, “Metodologías de Desarrollo de Aplicaciones WEB,” vol. 11, pp. 245–270, 2017.

[13] P. Wei, Z. Hong, and M. Shi, “Performance analysis of HTTP and FTP based on OPNET,” *2016 IEEE/ACIS 15th Int. Conf. Comput. Inf. Sci. ICIS 2016 - Proc.*, pp. 1–4, 2016.

[14] P. D. E. Especies and D. E. C. En, “Análisis semántico de programas escritos en java,” 2004.

[15] A. Kilgarriff, D. McCarthy, P. Rychlý, and J. Pomikálek, *Joyanes, Estructura de datos en Java*, vol. 2011, no. June. 2011.

[16] J. R. M. Ríos, M. P. Z. Ordóñez, F. F. R. Castillo, N. M. L. Mora, M. R. V. Pardo, and J. A. H. Tapia, *Metodologia SNAIL*, vol. 91. 2017.

ANEXOS

Anexo A. Requisitos funcionales y no funcionales

Tabla 2: RF01 – Administración de usuarios

Identificador del requisito	RF01
Nombre de requisito	Administración de usuarios.
Tipo	Funcional
Descripción	El usuario autorizado tendrá la posibilidad de realizar la administración de usuarios, así como de sus permisos y restricciones.
Actores	Usuario
Prioridad	Alta/Esencial
Condición de éxito	El usuario fue creado con éxito
Condición de fracaso	Los campos no son válidos al gestionar el usuario

Fuente: Luis Fernando Aguilar Nole.

Tabla 3: RF02 - Administración de roles

Identificador del requisito	RF02
Nombre de requisito	Administración de roles.
Tipo	Funcional
Descripción	El usuario autorizado podrá realizar la administración de roles, con lo cual podrá de forma posterior asignar dichos roles a los usuarios que se registren en el sistema.
Actores	Administrador
Prioridad	Alta/Esencial
Condición de éxito	El rol fue creado con éxito.
Condición de fracaso	Los campos no son válidos al gestionar el rol.

Fuente: Luis Fernando Aguilar Nole.

Tabla 4: RF03 – Control de ingreso al sistema (Login)

Identificador del requisito	RF03
Nombre de requisito	Control de ingreso al sistema (Login)
Tipo	Funcional
Descripción	El sistema asegurará el ingreso solo de usuarios autorizados mediante un nickname (nombre de usuario) y una contraseña.
Actores	Usuario
Prioridad	Alta/Esencial
Condición de éxito	Ingreso a la aplicación web con sus respectivos roles
Condición de fracaso	Nombre de usuario o contraseña incorrectos

Fuente: Luis Fernando Aguilar Nole.

Tabla 5: RF04 - Administración de departamentos

Identificador del requisito	RF04
Nombre de requisito	Administración de departamentos.
Tipo	Funcional
Descripción	El usuario autorizado podrá realizar la administración de departamentos con un empleado responsable de dicho departamento
Actores	Usuario
Prioridad	Alta/Esencial
Condición de éxito	El departamento fue creado con éxito
Condición de fracaso	Campos no validos al gestionar el departamento.

Fuente: Luis Fernando Aguilar Nole.

Tabla 6: RF05 - Administración de cargos

Identificador del requisito	RF05
Nombre de requisito	Administración de cargos.
Tipo	Funcional
Descripción	El usuario autorizado podrá realizar la administración de los cargos con un departamento asociado y un sueldo que se le asigna el responsable de dicho departamento
Actores	Usuario
Prioridad	Alta/Esencial
Condición de éxito	El cargo fue creado con éxito
Condición de fracaso	Campos no validos al gestionar el cargo.

Fuente: Luis Fernando Aguilar Nole.

Tabla 7: RF06 - Administración de postulantes

Identificador del requisito	RF06
Nombre de requisito	Administración de postulantes.
Tipo	Funcional
Descripción	El usuario autorizado podrá realizar la administración de los postulantes con el cargo al cual están postulando.
Actores	Usuario
Prioridad	Alta/Esencial
Condición de éxito	El postulante fue creado con éxito
Condición de fracaso	Campos no validos al gestionar el postulante.

Fuente: Luis Fernando Aguilar Nole.

Tabla 8: RF07 - Administración de horarios y responsables para receptar pruebas.

Identificador del requisito	RF07
Nombre de requisito	Administración de horarios y responsables para receptar pruebas.
Tipo	Funcional
Descripción	El usuario autorizado podrá realizar la administración de los horarios y responsables para receptar pruebas.
Actores	Usuario
Prioridad	Alta/Esencial
Condición de éxito	El horario y responsable fue establecido con éxito
Condición de fracaso	Campos no validos al gestionar el horario o responsable.

Fuente: Luis Fernando Aguilar Nole.

Tabla 9: RF08 - Administración de empleados.

Identificador del requisito	RF08
Nombre de requisito	Administración de empleados.
Tipo	Funcional
Descripción	El usuario autorizado podrá realizar la administración de los empleados.
Actores	Usuario
Prioridad	Alta/Esencial
Condición de éxito	El empleado fue creado con éxito.
Condición de fracaso	Campos no validos al gestionar el empleado.

Fuente: Luis Fernando Aguilar Nole.

Tabla 10: RF09 – Reportes de actividades mensuales o quincenales de empleados

Identificador del requisito	RF10
Nombre de requisito	Reportes de actividades mensuales o quincenales de empleados
Tipo	Funcional
Descripción	Esta opción permitirá al administrador realizar reportes mensuales o quincenales de las actividades realizadas por los empleados.
Actores	Usuario
Prioridad	Alta/Esencial
Condición de éxito	El reporte fue generado con éxito.
Condición de fracaso	Datos insuficientes para la generación del reporte.

Fuente: Luis Fernando Aguilar Nole.

Tabla 11: RF10 - Administración de asistencia de empleados

Identificador del requisito	RF10
Nombre de requisito	Administración de asistencia de empleados.
Tipo	Funcional
Descripción	El sistema web permitirá el control de la asistencia de los empleados.
Actores	Usuario
Prioridad	Alta/Esencial
Condición de éxito	La asistencia del empleado fue registrada con éxito.
Condición de fracaso	Campos no validos al registrar la asistencia del empleado.

Fuente: Luis Fernando Aguilar Nole.

Tabla 12: RNF01 – Uso de la arquitectura MVC para la solución del proyecto

Identificador del requisito	RNF01
Nombre de requisito	Uso de la arquitectura MVC para la solución del proyecto
Tipo	Funcional
Descripción	La aplicación web será desarrollada mediante el uso de la arquitectura Modelo-Vista-Controlador
Actores	Usuario
Prioridad	Alta/Esencial

Fuente: Luis Fernando Aguilar Nole.

Tabla 13: RNF02 – Uso de TAD Lista

Identificador del requisito	RNF02
Nombre de requisito	Uso de TAD Lista
Tipo	Funcional
Descripción	La aplicación web será desarrollada mediante el uso de TAD en forma de lista.
Actores	Usuario
Prioridad	Alta/Esencial

Fuente: Luis Fernando Aguilar Nole.

Tabla 14: RNF03 – Control de ingreso de datos

Identificador del requisito	RNF 03
Nombre de requisito	Control de ingreso de datos
Tipo	Funcional
Descripción	La aplicación web deberá validar cada ingreso de datos, así como la salida de los mismos.
Actores	Usuario
Prioridad	Alta/Esencial

Fuente: Luis Fernando Aguilar Nole.

Tabla 15: RNF04 – Diseño responsivo

Identificador del requisito	RNF 04
Nombre de requisito	Diseño responsivo
Tipo	Funcional
Descripción	La aplicación web es posible apreciarla desde cualquier dispositivo con conexión a internet.
Actores	Usuario
Prioridad	Alta/Esencial

Fuente: Luis Fernando Aguilar Nole.

Tabla 16: RNF05 – Administración de permisos

Identificador del requisito	RNF 05
Nombre de requisito	Administración de permisos
Tipo	Funcional
Descripción	El control de ingreso al sistema delegara el rol y los permisos al usuario según lo que el administrador le haya otorgado.
Actores	Usuario
Prioridad	Alta/Esencial

Fuente: Luis Fernando Aguilar Nole.

Anexo B. Prototipo Control de ingreso al sistema (Login)

Ilustración 7: Control de ingreso al sistema (Login)

Fuente: Luis Fernando Aguilar Nole

Ilustración 8: Panel de administración

Fuente: Luis Fernando Aguilar Nole

Ilustración 9: Lista de Usuarios

Fuente: Luis Fernando Aguilar Nole

Anexo C: Login

Ilustración 10: Vista del Login

Fuente: Luis Fernando Aguilar Nole

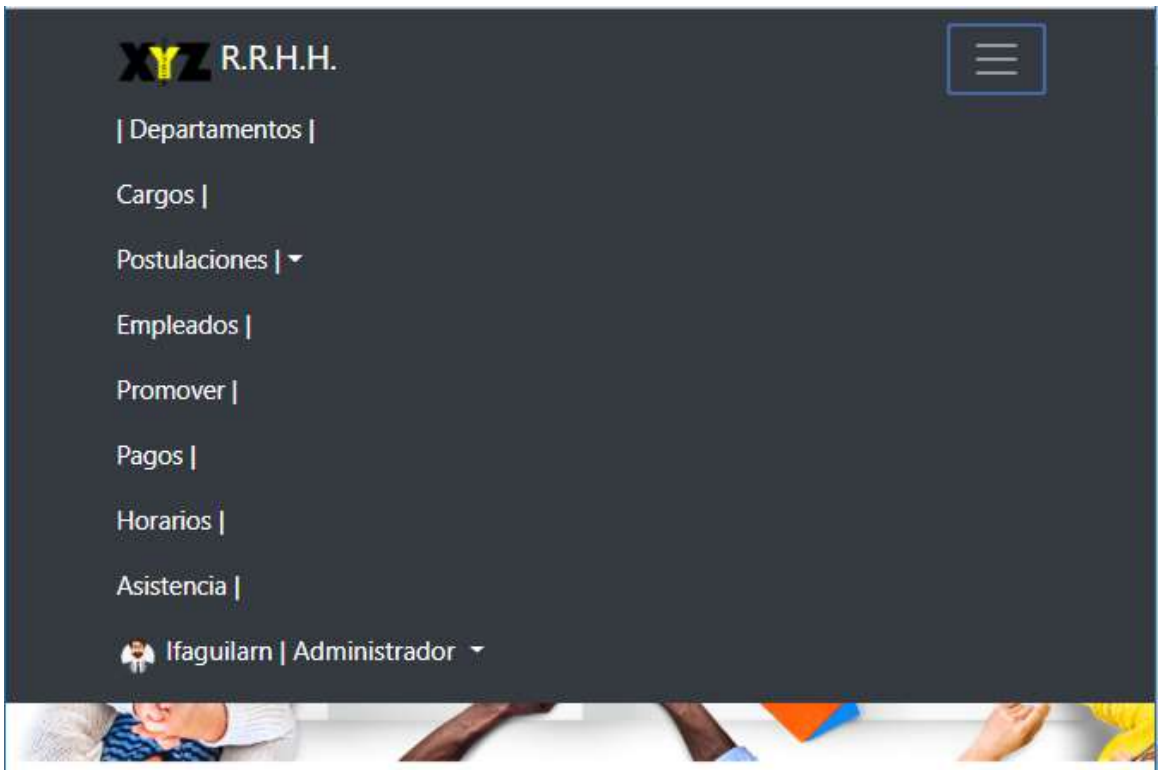
Anexo D: Menú Principal – Administrador

Ilustración 11: Menú principal

Fuente: Luis Fernando Aguilar Nole

Anexo E: Opciones del menú – Diseño Responsivo

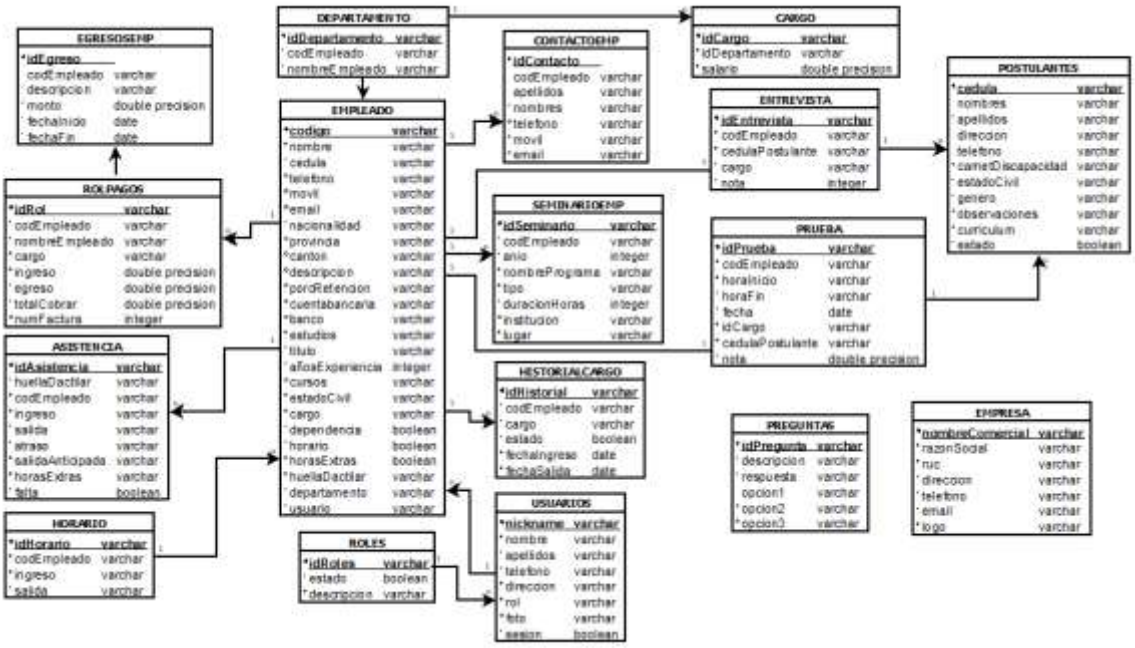
Ilustración 12: Diseño responsivo – Menú Principal



Fuente: Luis Fernando Aguilar Nole

Anexo F. Diagrama de base de datos relacional

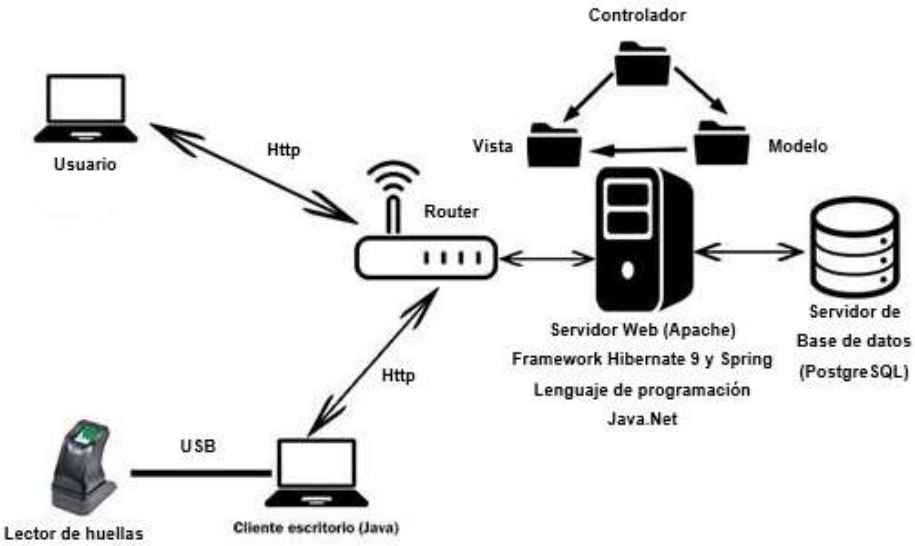
Ilustración 13: Diagrama relacional base de datos



Fuente: Luis Fernando Aguilar Nole

Anexo G. Diagrama de arquitectura MVC

Ilustración 14: Diseño arquitectura MVC



Fuente: Luis Fernando Aguilar Nole