**Desarrollo de Software Web**

**Actividad #2**

Héctor David Medina Vergara

Facultad de Ingeniería, Universidad de Cartagena

Ingeniería de Software

John Arrieta

2023

**TABLA DE CONTENIDO**

**PRESENTACIÓN** 1

**CONTENIDO DEL TRABAJO** 2

**INTRODUCCIÓN** 3

**OBJETIVOS** 4

**JUSTIFICACIÓN** 5

**DESARROLLO** 6

**ARGUMENTACIÓN INDIVIDUAL**17

**BIBLIOGRAFÍA** 18

**INTRODUCCIÓN**

Una de las metodologías de diseño de software más comunes en la creación de aplicaciones web es la arquitectura Model View Controller (MVC). MVC permite que los componentes de una aplicación web distingan claramente sus funciones entre sí, lo que facilita el mantenimiento y la escalabilidad a largo plazo.

Este artículo muestra una aplicación web creada con JavaServer Pages y la arquitectura MVC. Uno de los métodos más populares para crear páginas web dinámicas en Java es JSP.

Este documento tiene como objetivo presentar una comprensión clara de cómo se puede usar la arquitectura MVC en una aplicación web utilizando JSP, así como también cómo se pueden implementar varias funcionalidades de una aplicación en los diversos componentes de la arquitectura. Además, se mostrará cómo el desarrollo y mantenimiento de aplicaciones web puede facilitarse mediante la separación de funciones que ofrece la arquitectura MVC.

**OBJETIVOS**

**Objetivo General:**

* Desarrollar una aplicación web utilizando la arquitectura Modelo-Vista-Controlador (MVC) y la tecnología JavaServer Pages (JSP).

**Objetivos Específicos:**

* Diseñar y desarrollar los diferentes componentes de la aplicación web (modelo, vista, controlador) siguiendo los principios de la arquitectura MVC.
* Implementar diferentes funcionalidades en la aplicación web, como la creación, edición, eliminación y marcado de tareas como completadas.
* Realizar pruebas exhaustivas para garantizar el correcto funcionamiento de la aplicación y la validación de los datos ingresados por el usuario.

**JUSTIFICACIÓN**

Una estrategia de diseño común utilizada en la creación de aplicaciones web es la arquitectura Model-View-Controller (MVC). La arquitectura MVC ofrece una clara separación de funciones entre las diversas partes de una aplicación web, lo que facilita la escalabilidad y el mantenimiento del código a lo largo del tiempo. Sin embargo, JavaServer Pages (JSP) es una tecnología que utiliza Java como lenguaje de programación para crear páginas web dinámicas.

Con el fin de mostrar cómo se pueden combinar estas herramientas para crear aplicaciones web dinámicas y escalables, este trabajo tiene como objetivo desarrollar una aplicación web utilizando la arquitectura MVC y la tecnología JSP. La aplicación web que se creará permitirá a los usuarios gestionar tareas y realizar diversas operaciones.

La creación de esta aplicación web brindará la oportunidad de conocer más a fondo la tecnología JSP y la arquitectura MVC, así como también cómo se pueden utilizar estas ideas para crear aplicaciones web dinámicas y escalables.

En conclusión, este trabajo ofrece la oportunidad de mostrar cómo se pueden aplicar las ideas teóricas a la creación de aplicaciones web prácticas y evaluar la eficiencia de las herramientas empleadas en el proceso.

**DESARROLLO**

El patrón Modelo-Vista-Controlador (MVC) es un enfoque arquitectónico utilizado en el desarrollo de aplicaciones web. Proporciona una estructura organizada y separada de responsabilidades para el diseño y desarrollo de aplicaciones, lo que facilita la modularidad, la reutilización de código y el mantenimiento a largo plazo.

1. **Modelo:** El modelo representa los datos y la lógica de negocio de la aplicación. Se encarga de manejar la interacción con la base de datos, procesar la información y aplicar reglas de negocio. En resumen, el modelo encapsula la funcionalidad principal de la aplicación.
2. **Vista:** La vista se encarga de la presentación de la interfaz de usuario. Es responsable de mostrar los datos al usuario de una manera comprensible y atractiva. En el contexto web, la vista suele ser una página HTML generada dinámicamente que muestra los resultados y permite la interacción con el usuario.
3. **Controlador:** El controlador actúa como intermediario entre el modelo y la vista. Recibe las solicitudes del usuario y coordina las acciones necesarias. Procesa la entrada del usuario, actualiza el modelo según sea necesario y selecciona la vista adecuada para presentar los resultados. El controlador se encarga de la lógica de enrutamiento y toma decisiones basadas en la interacción del usuario.

Ventajas de utilizar el patrón MVC en el desarrollo de aplicaciones web:

* **Separación de responsabilidades:** MVC promueve una separación clara de las responsabilidades de cada componente (modelo, vista y controlador). Esto facilita la colaboración en equipo, el mantenimiento y la reutilización de código.
* **Modularidad:** El enfoque MVC permite dividir la aplicación en módulos independientes. Esto facilita la escalabilidad y el desarrollo paralelo, ya que diferentes equipos pueden trabajar en diferentes partes del proyecto sin interferir entre sí.
* **Mayor legibilidad y mantenibilidad:** Al separar la lógica de negocio (modelo) de la presentación (vista) y la gestión de interacciones (controlador), el código se vuelve más claro y más fácil de mantener. Esto agiliza el proceso de depuración y facilita las actualizaciones y mejoras futuras.
* **Reutilización de código:** La estructura modular del MVC permite la reutilización de componentes en diferentes partes de la aplicación o incluso en proyectos futuros. Esto ahorra tiempo y esfuerzo en el desarrollo, evitando la duplicación de código y promoviendo una mayor eficiencia.
* **Flexibilidad:** Al utilizar el patrón MVC, es más sencillo realizar cambios en la interfaz de usuario (vista) sin afectar el modelo y el controlador subyacentes. Esto permite una mayor flexibilidad en el diseño y la adaptación a las necesidades cambiantes de los usuarios.

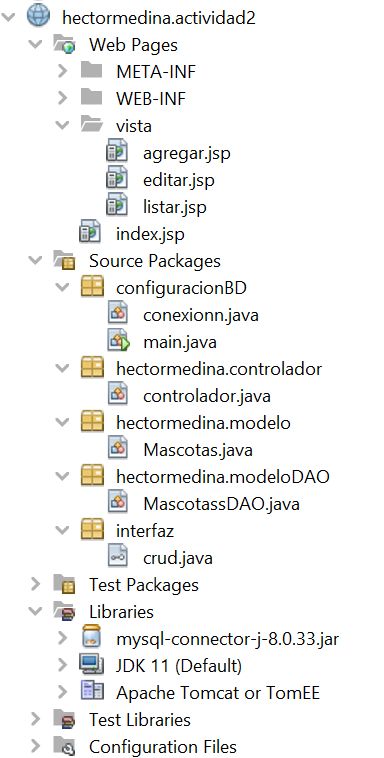
**JAVASERVER PAGES (JSP)** es una tecnología de servidor utilizada en el desarrollo de aplicaciones web basadas en Java. Permite la creación de páginas dinámicas que combinan código Java con contenido HTML, lo que facilita la generación de contenido web de manera dinámica y personalizada.

En JSP, las páginas son preprocesadas y se convierten en servlets Java antes de ser enviadas al cliente. Esto significa que el código Java incrustado en las páginas JSP se ejecuta en el servidor, generando el contenido HTML resultante que se envía al navegador del cliente.

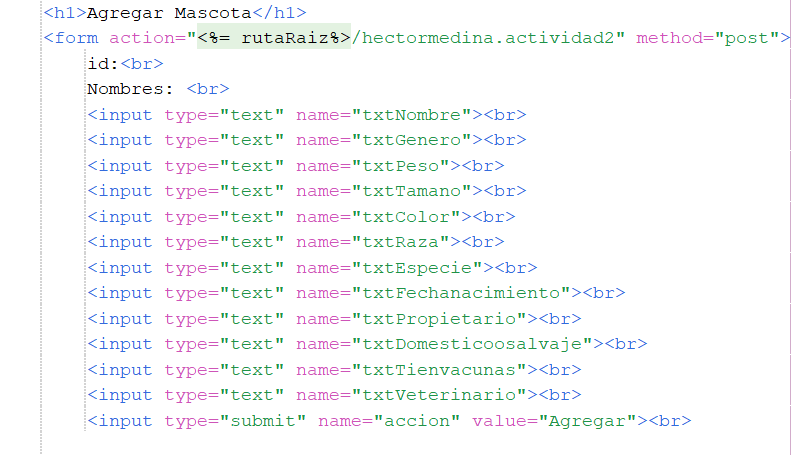
**Las características clave de JSP incluyen:**

* **Integración con Java:** JSP permite la incorporación directa de código Java dentro de las páginas, lo que proporciona un amplio rango de funcionalidades y acceso a bibliotecas y APIs de Java. Esto facilita la manipulación de datos, la interacción con bases de datos y la implementación de lógica empresarial en las aplicaciones web.
* **Separación de la lógica de presentación:** JSP promueve la separación de la lógica de presentación del código Java subyacente. Los desarrolladores pueden utilizar etiquetas y directivas JSP para encapsular el código Java y generar contenido HTML de manera más legible y mantenible.
* **Reutilización de componentes:** JSP permite la creación de fragmentos reutilizables de código llamados "etiquetas personalizadas" o "taglibs". Estos componentes encapsulan funcionalidades comunes y facilitan el desarrollo modular y la reutilización de código en diferentes partes de la aplicación.
* **Soporte para tecnologías web:** JSP se integra fácilmente con otras tecnologías web, como servlets, JavaBeans, CSS y JavaScript. Esto permite el desarrollo de aplicaciones web completas y altamente interactivas utilizando JSP como capa de presentación.
* **Amplia compatibilidad:** JSP es compatible con una amplia gama de servidores de aplicaciones y frameworks de desarrollo web en el ecosistema Java. Esto proporciona flexibilidad y portabilidad al desarrollar aplicaciones web utilizando JSP.

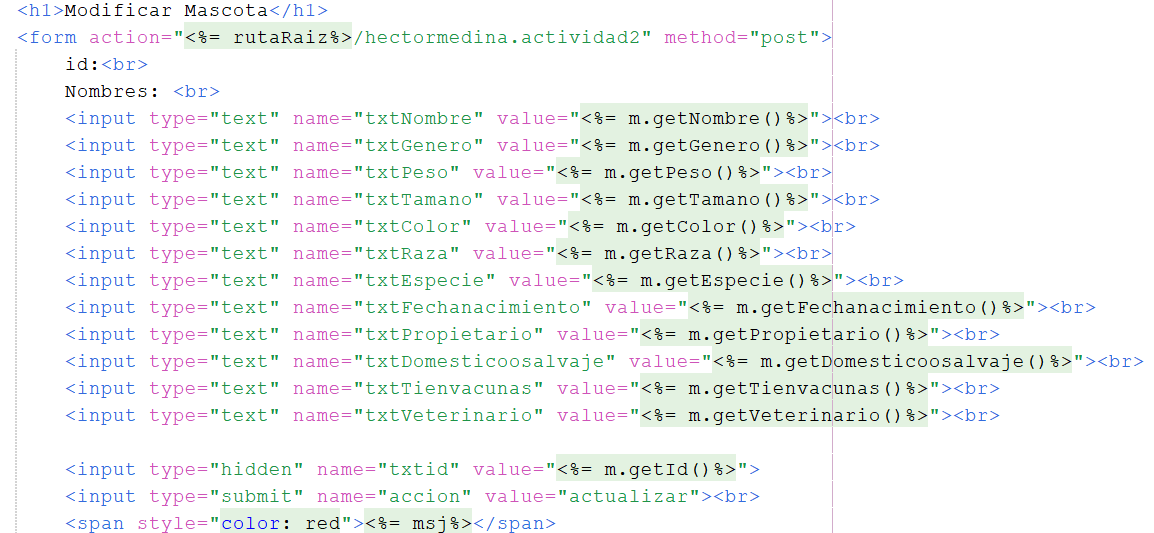
**ESTRUCTURA DEL PROYECTO**



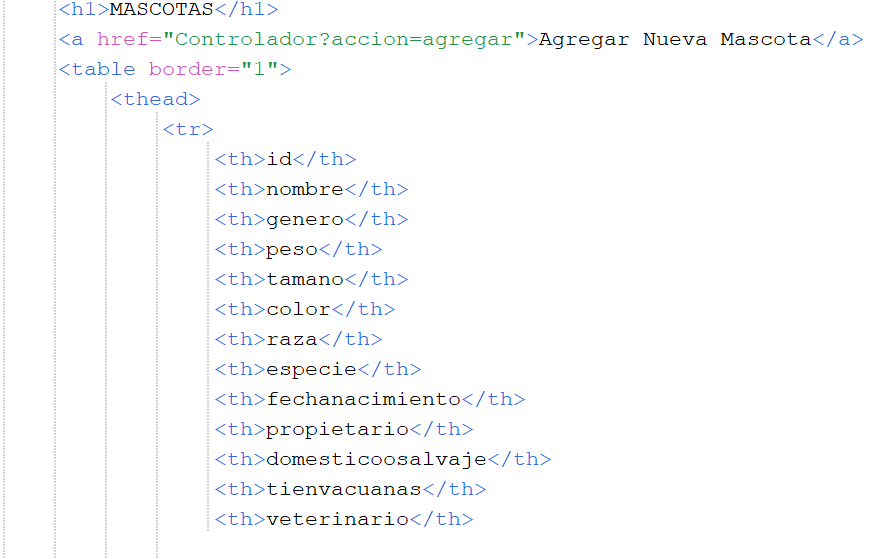
**agregar.jsp:** Esta clase corresponde a la vista utilizada para agregar nuevos elementos a la aplicación. Los usuarios podrán completar el formulario y enviar los datos al servidor para su procesamiento. Se muestra una breve muestra del código:



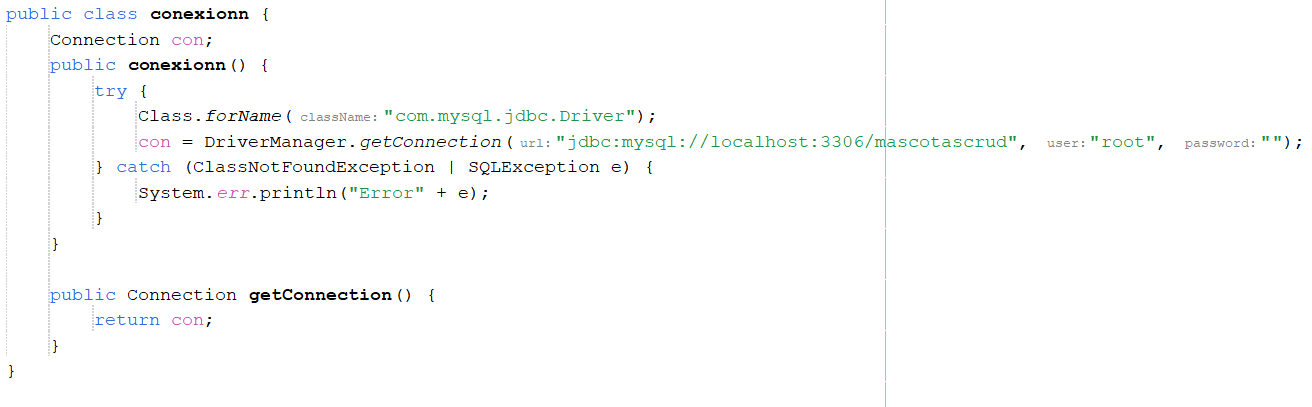
**editar.jsp:** En esta clase, se presenta la vista para editar elementos existentes en la aplicación. Al igual que agregar.jsp, suele contener un formulario con los campos prellenados con la información actual del elemento a editar. Los usuarios podrán modificar los valores y enviar el formulario al servidor para guardar los cambios realizados. Se muestra breve muestra del código:



**listar.jsp:** Esta clase se encarga de mostrar una lista de elementos en la aplicación. Se muestra breve muestra del código:



**conexionn.java:** Esta clase se encarga de establecer la conexión con la base de datos. Dentro de esta clase, se definen los parámetros necesarios para establecer la conexión, como la URL de la base de datos, el nombre de usuario y la contraseña. Breve muestra del código:



**controlador.java:** Esta clase, como su nombre indica, representa el controlador en el patrón Modelo-Vista-Controlador (MVC). Es responsable de manejar las solicitudes y las interacciones del usuario, coordinando las acciones entre la vista y el modelo.

Dentro de la clase controlador.java, se implementan métodos y lógica para controlar el flujo de la aplicación. Esto puede incluir la recepción de datos enviados por el usuario desde la vista, validación de la información, invocación de métodos en el modelo para realizar operaciones de negocio y preparación de datos para ser presentados nuevamente en la vista.

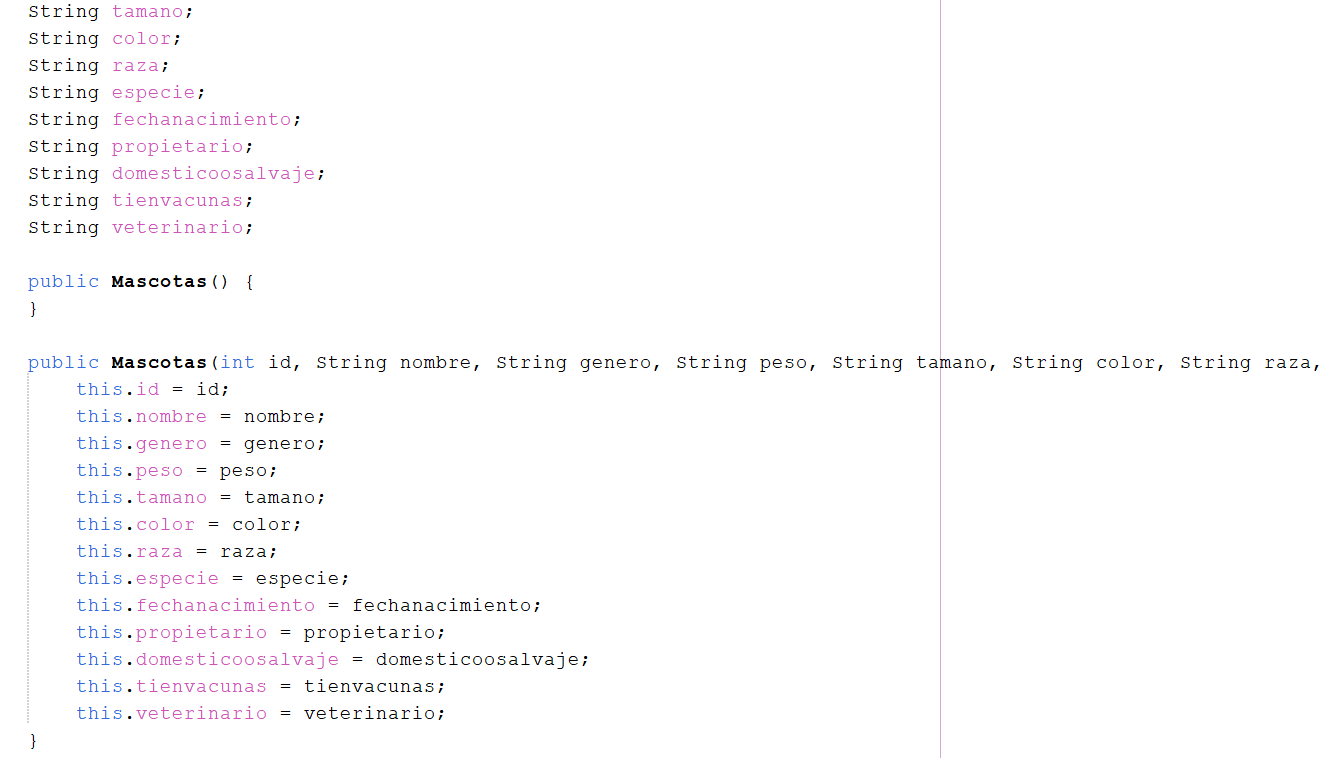
El controlador actúa como intermediario entre la vista y el modelo, asegurando una separación clara de responsabilidades. Se encarga de recibir eventos o acciones del usuario desde la vista, procesarlos y coordinar las acciones correspondientes en el modelo, y luego actualizar la vista para mostrar los resultados al usuario. Breve muestra del codigo:



**Mascotas.java:** Esta clase representa el modelo de la aplicación. En el patrón Modelo-Vista-Controlador (MVC), el modelo se encarga de encapsular la lógica de negocio y los datos de la aplicación.

La clase Mascotas.java se refiere a una entidad llamada "Mascotas" que representa objetos relacionados con las mascotas. Contiene atributos y métodos que representen las características y el comportamiento de una mascota, como nombre, edad, raza, tipo, etc.

El modelo se utiliza para almacenar y manipular la información de la aplicación, y puede interactuar con la capa de persistencia, como una base de datos, para realizar operaciones de consulta, inserción, actualización o eliminación de datos. También puede contener lógica de negocio, como validaciones, cálculos y reglas específicas relacionadas con las mascotas. Breve muestra del codigo.



**MascotassDAO.java:** Esta clase pertenece al patrón de diseño DAO (Data Access Object) y se encarga de manejar el acceso a los datos relacionados con las mascotas en la aplicación. El objetivo principal de esta clase es proporcionar una capa de abstracción entre el modelo y la capa de persistencia, como una base de datos.

La clase MascotassDAO.java contiene los métodos para realizar operaciones CRUD (editar, listar, agregar, eliminar) en la base de datos para la entidad "Mascotas". Estos métodos pueden incluir la inserción de nuevos registros, la consulta de registros existentes, la actualización de registros y la eliminación de registros relacionados con las mascotas.

El objetivo principal de la clase MascotassDAO.java es abstraer los detalles específicos de la base de datos y proporcionar una interfaz limpia y coherente para que el modelo pueda interactuar con la capa de persistencia. Breve muestra del codigo:



Hemos investigado la creación de una aplicación MVC usando JSP. Separar la lógica comercial, la presentación y el control de una aplicación web en componentes claros y organizados es posible gracias al patrón Modelo-Vista-Controlador.

JSP, una tecnología de servidor, nos permite usar HTML y el poder de Java para crear páginas dinámicas y personalizadas. Simplifica el desarrollo y mantenimiento de aplicaciones web fiables al permitir la integración con otras tecnologías web y fomentar la reutilización de componentes.

El modelo contiene la lógica comercial y los datos de la aplicación, y el controlador está a cargo de coordinar las interacciones entre la vista y el modelo. La interfaz de usuario debe presentarse de manera atractiva y comprensible para la vista.

La reutilización de componentes, una mayor legibilidad y capacidad de mantenimiento del código, una separación clara de responsabilidades, flexibilidad de diseño y una mayor eficiencia de desarrollo son solo algunos de los beneficios del uso de este patrón y tecnología.

Podemos desarrollar aplicaciones web escalables, modulares y fáciles de mantener si nos adherimos a las mejores prácticas de desarrollo de MVC con JSP. Esta estrategia nos brinda la capacidad de brindar una experiencia de usuario atractiva y, al mismo tiempo, nos brinda la adaptabilidad para satisfacer las necesidades y requisitos cambiantes de los usuarios.

En conclusión, el patrón MVC y JSP funcionan bien juntos para brindarnos una base sólida para crear aplicaciones web eficientes y de alta calidad.

**SÍNTESIS Y ARGUMENTACIÓN INDIVIDUAL**

En base a todo lo mencionado anteriormente podemos decir que, una herramienta para desarrollar aplicaciones web ya sean dinámicas o de manera escalable es una aplicación MVC que utiliza tecnología JSP. Este conjunto de herramientas es crucial porque permite una clara división del trabajo y facilita el escalado y el mantenimiento del código a largo plazo.

La creación de una amplia gama de aplicaciones web, como sistemas de gestión de contenido, sitios web de comercio electrónico y sistemas de gestión de recursos humanos, se puede realizar a diario utilizando una aplicación MVC con tecnología JSP. Estos programas pueden ser utilizados por empresas y organizaciones para administrar información y completar una variedad de tareas.

En resumen, una aplicación MVC compatible con JSP es una herramienta útil para desarrollar aplicaciones webs en una variedad de escenarios del mundo real. Su importancia se deriva de la clara división del trabajo, la simplicidad del mantenimiento y la escalabilidad, lo que conduce a soluciones más rentables y efectivas.

**BIBLIOGRAFÍA**

Santana, L. (2020, septiembre 7). *El modelo de arquitectura View Controller (MVC).* freeCodeCamp.org. https://www.freecodecamp.org/espanol/news/el-modelo-de-arquitectura-view-controller-pattern/

Osmosis Latina (s.f.). Modelo Vista Controlador (MVC).   
De. https://javaweb.osmosislatina.com/curso/mvc.htm

https://www.it.uc3m.es/jaf/cw/practicas/7-mvc/