

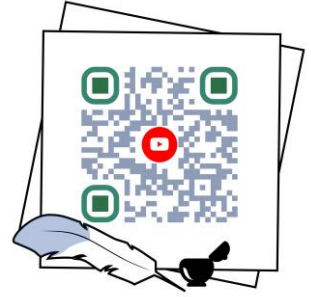


samantha y sus peludos

David Sebastian Enciso Lopez david.enciso@pi.edu.co

Cristian De Los Rios cristian.de.los.rios@pi.edu.co

Nicole Pajarito nicole.pajarito@pi.edu.co



Resumen – Este proyecto tiene como objetivo crear un sistema de información con conexión a base de datos desde un sitio web basados en solución LAMP (Linux – Apache – MySQL – PHP), mediante la virtualización de un servidor el cual va a tener instalado un sistema operativo basado en Ubuntu Server abordando las diferentes temáticas vistas en programación 2 que nos darán como resultado un muy buen sitio web en donde los amantes de las mascotas puedan encontrar los mejores servicios, productos y accesorios para sus peluditos. Esta aplicación y sitio web nos mostrara diferentes tipos de servicios como guardería, vacunación, consultas, productos alimenticios y accesorios entre otros que podrán ser consultados por los usuarios luego de realizar el login mediante el ingreso de un usuario y contraseña, se tiene como una prioridad crear un sitio web agradable e intuitivo para el uso del público en general.

Palabras claves – Ubuntu server, VirtualBox, PHP, MySQL, HTML, LAMP

Abstract – This project aims to create an information system with a connection to a database from a website based on a LAMP solution (Linux - Apache - MySQL - PHP), through the virtualization of a server which will have an operating system installed based on in Ubuntu Server addressing the different topics seen in programming 2 that will result in a very good website where pet lovers can find the best services, products and accessories for their furry friends. This application and website will show us different types of services such as daycare, vaccination, consultations, food products and accessories among others that can be consulted by users after logging in by entering a username and password, it is a priority. create a pleasant and intuitive website for the use of the general public.

I. INTRODUCCION

Este proyecto pretende realizar el desarrollo de una aplicación y sitio web que permitirá mostrar información, servicios, productos, y accesorios con una conexión a bases de datos con MySQL y un servidor web para los más consentidos de la casa

nuestros peluditos y brindar una sensación de bienestar y satisfacción no solo para ellos sino también para los demás integrantes de la familia. Durante el desarrollo de este proyecto se realiza la instalación de VirtualBox que nos

permite la creación de máquinas virtuales allí se realiza la configuración de un servidor con sistema operativo Ubuntu server. Una vez configurada la máquina virtual se realiza la instalación de MySQL y PHP MyAdmin para poder gestionar nuestro sistema de información.

II. DEFINICIONES

¿Qué es Linux – sabor Ubuntu? Características, beneficios y versiones

Linux es un sistema operativo que fue creado o convido desde la base de Unix con una licencia publica general lo que permite que este código pueda ser copiado, modificado y/o redistribuido entre la comunidad en general. Este sistema operativo es muy ligero, pero se pueden ir agregando programas y/o servicios de acuerdo con la necesidad del cliente. Inicio en 1980 con la premisa de ser un software libre, Linus Torvalds y Richard Stallman han sido los mayores contribuyentes en este sistema dentro de los múltiples “sabores de Linux encontramos Ubuntu el cual es el más usado en el mundo y nace con objetivo de poder llevar Linux a todo el mundo ya que este era un sistema operativo de difícil manejo y se podían requerir algunos conocimientos técnico para su operación, encontramos diversas versiones en las cuales se destaca la 22.04 LTS dentro de las más recientes. Entre sus principales características encontramos:

- Código abierto
- Variedad de distribuciones
- multiusuario y multitarea
- estabilidad y seguridad
- personalizable
- mayor control de los procesos.

Ventajas:

- seguridad
- Personalizable
- gratuito
- rendimiento y fiabilidad.

¿Qué es Apache? Características, beneficios y versiones

Apache es un servidor web multiplataforma, gratuito y de código abierto. Este servidor web es uno de los servidores web más utilizados en el mundo y actualmente lo utilizan el 43% de los sitios web. El nombre Apache se refiere a una tribu nativa americana conocida por su resistencia y tácticas de guerra. Es muy popular entre los programadores debido a su modularidad y actualizaciones constantes de la comunidad. Apache tiene una estructura basada en módulos lo que permite activar y/o desactivar algunas funciones adicionales encontramos la versión más reciente 2.4 y sus principales beneficios son:

- Soporte
- Multiplataforma
- Simplicidad
- Seguridad
- Flexibilidad.

¿Qué es MYSQL? Características, beneficios y versiones

Es un robusto motor de bases de datos relacionales y uno de los más populares en el mundo el cual ofrece una gran cantidad de funciones que permite que la gestión y manejo de los datos sea mucho más eficiente. Nació bajo de sun Microsystems bajo la premisa de código libre y abierto, pero cuando Sun fue adquirido por oracle paso a tener 2 opciones de licencia una que es paga y otra que es de código abierto, en sus características y ventajas encontramos:

- Sirve para el manejo de bases de datos relacionales.
- Implementa varios motores de almacenamiento con características y velocidades diferentes.
- Es software libre para uso en la versión de comunidad.
- Dispone de la arquitectura cliente/servidor.
- Sistema ligero, de fácil uso y mantenimiento.
- Es robusto, seguro y confiable.
- Fácil instalación.
- Open source.

Versión 8.0.34 es la más reciente disponible.

¿Qué es PHP?

PHP es el acrónimo de “Hypertext PreProcessor” es un lenguaje de programación comúnmente usado para el desarrollo de aplicaciones en la web y la creación de sitios web ya que es de código abierto y libre, fácil de usar y en un constante mejoramiento, facilitando la conexión entre los servidores y la interfaz de usuario. En sus características se puede resaltar que:

- Es código abierto y gratuito.
- Lenguaje orientado a objetos haciendo que sea más fluido el procesamiento.
- Código limpio y estable. Permite separación de códigos.
- Permite el desarrollo de páginas web complejas y dinámicas.

Beneficios:

- Se puede usar con cualquier sistema operativo virtual y servidor.
- HTML, imágenes, PDFs, o archivos flash pueden generarse de forma dinámica con este lenguaje.
- Soporta una gran cantidad de bases de datos diferentes.
- Es un lenguaje ideal para la creación de sitios web basados en bases de datos.
- Cuenta con un soporte completo para la comunicación del servidor con otros protocolos.
- Es un lenguaje sencillo de aprender.

Se encuentra la versión 8.2 como la más reciente.

¿Qué es el modelo de arquitectura tres capas?

Es una arquitectura que tiene una capa intermedia durante el proceso en la cual cada capa es un proceso y se divide en:

Capa usuario: es la que proporciona al usuario una interfaz gráfica en la cual podrá ver toda la información y los datos acá es donde se solicita y se recibe los servicios de la capa del mismo nivel o capa intermedia.

Capa de negocio (intermedia): su principal función es hacer el puente entre la capa de usuario y capa de datos ya que entre estas dos no se genera conexión, acá se deben complementar las tareas del negocio como la validación en bases de datos.

Capa de Datos: en esta se encarga de las tareas típicas como la creación, modificación, consulta y/o borrado, también es encargado de gestionar las peticiones que vienen de la capa de negocio.

La implementación de este modelo de arquitectura nos genera grandes beneficios como:

- Modularidad.
- Separación de responsabilidades.
- Reutilización.

- Escalabilidad.
- Flexibilidad.

Modelo Vista Controlador

Esta pensado en la división de las partes que conforman un aplicativo y asegurando su buen mantenimiento, el modelo es el encargado del manejo de datos, la vista es la finalmente se muestra al usuario por medio de una interfaz gráfica y el controlador es el encargado de la manipulación del modelo para mostrar información por medio de la vista. En el proyecto se mostrará la vista desde la creación del sitio web y se aplica en el proyecto de la siguiente manera la vista será el sitio web, el modelo será la base de datos y el controlador seria la manipulación del sistema de información.

III. REQUERIMIENTOS FUNCIONALES.

¿Qué son los requerimientos funcionales?

Son las descripciones del comportamiento que debe tener una solución de software y la información que debe manejar, expresan las cualidades que debe tener la solución para satisfacer los requerimientos del proyecto, se debe proporcionar información suficientemente detallada para el correcto funcionamiento e implementación de la solución.

Historias de Usuario:

HISTORIA DE USUARIO				
ID. HISTORIA DE USUARIO	ROL	CARACTERISTICAS/ FUNCIONALIDAD	RAZON/RESUELTO	CRITERIO DE ACEPTACION
1	Administrador	Quiero ingresar productos	Para mostrar a los clientes los nuevos productos	Opcion de ingresar nuevos productos
2	Empleado	Deseo consultar los productos disponibles	Para poder ofertar a los clientes los productos disponibles	Opcion para consultar los productos
3	Cliente	Quiero consultar los servicios que ofrece la tienda	Para poder conocer los servicios de mi preferencia	Opcion de consultar los servicios
4	Empleado	Quiero ingresar mascotas	Poder tener el control de las mascotas que se atienden	Opcion de ingresar mascotas
5	Visitante	Deseo conocer el producto mas economico	Quiero comprar pero no cuento con mucho dinero	Filtro para ordenar los productos por precio del mas economico al mas costoso
6	Administrador	Quiero eliminar un producto	Para eliminar un cliente que ya no se encuentra disponible	Opcion de eliminar producto
7	Vendedor	Quiero conocer los productos de mayor Stock	Para no pedir mas productos de esas referencias	opcion de ordenar el Stock de mayor a menor
8	Administrador	Quiero modificar los datos de un empleado	Por que al momento de ingresarlo cometi un error	Opcion de modificar a los empleados
9	Cliente	Quiero conocer los alimentos ofertados	Deseo comprar alimento para perro	opcion de filtrar productos por nombre
10	Administrador	Quiero ofrecer descuento sobre los productos	Para generar fidelizacion con la tienda	Opcion de generar descuento en el precio del producto

Figura 1. Historias de usuario.Creación propia

Product Version: Apache NetBeans IDE 19
Java: 21.0.1; Java HotSpot(TM) 64-Bit Server VM 21.0.1+12-LTS-29
Runtime: Java(TM) SE Runtime Environment 21.0.1+12-LTS-29
System: Windows 11 version 10.0 running on amd64; UTF-8; es_CO (nb)
User directory: C:\Users\EQUIPO\AppData\Roaming\NetBeans\19
Cache directory: C:\Users\EQUIPO\AppData\Local\NetBeans\Cache\19

Figura 2. Requerimientos funcionales. Creación propia

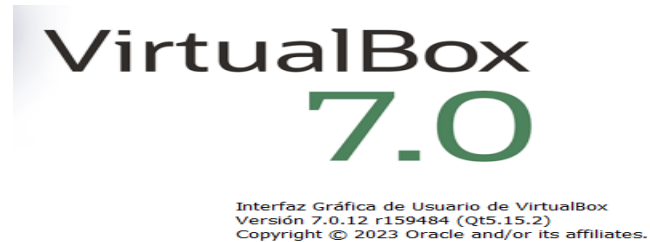


Figura 1.

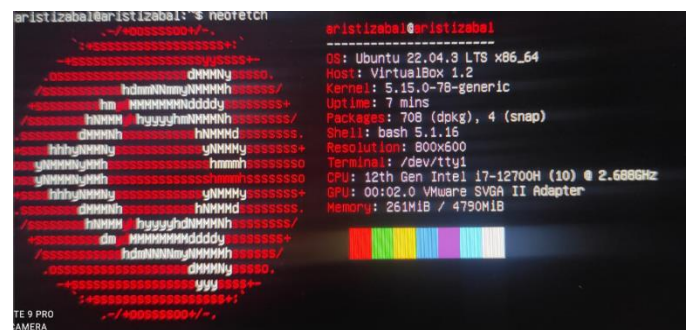


Figura 2.

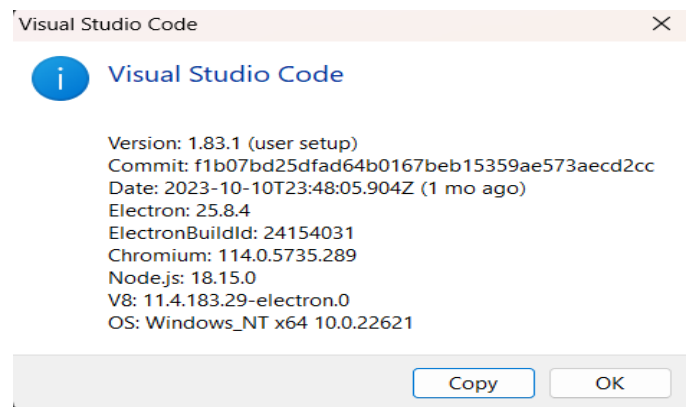


Figura 3.

¿Qué son los requerimientos NO funcionales?

Son las restricciones de un sistema en el que se definen sus atributos de calidad, estos ayudan a garantizar que el sistema cumpla las necesidades del usuario. Podemos encontrar algunos requerimientos no funcionales como>

- Seguridad: el sistema debe tener una clave de acceso para ser protegido de accesos no autorizados
- Disponibilidad: debe estar disponible siempre que se requiera.
- Compatibilidad: debe ser compatible con otros sistemas y no presentar novedad.
- Mantenimiento: debe ser actualizable y de fácil mantenimiento.

IV. DICCIONARIO DE DATOS.

Nombre	Tipo de dato	Descripción
Scanner	Scanner	Para leer el dato ingresado por el usuario
java.sql.Connection*	Biblioteca	Gestiona la conexión con la base de dato
java.sql.DriverManager*	Biblioteca	Permite registrar y obtener controladores de bases de datos.
javax.swing.*	Biblioteca	Biblioteca de entorno gráfico
java.sql.PreparedStatement*	Biblioteca	Ejecuta consultas SQL pre- compiladas de manera segura.
java.sql.ResultSet*	Biblioteca	Representa el conjunto de resultados de una consulta SQL
java.sql.SQLException*	Biblioteca	Maneja excepciones relacionadas con operaciones de base de datos.
java.util.Scanner	Biblioteca	Lee la entrada del usuario desde la consola
DB_URL	Conexión	Conexión con mysql IP; 192.168.10.18
DB_USER	USER	nombre de usuario utilizado para autenticarse en la base de datos. En este caso, el usuario es "politecnico".
DB_PASSWORD	Password	La contraseña asociada al nombre de usuario. En este caso, la contraseña es "123456".
opcion	String	Equipo que ganó la final
Mostrarmenuprincipal	switch	Menú principal

ingresarCliente	Método	Método para ingresar nuevo cliente
eliminarCliente	Método	Método para eliminar cliente.
consultarClientes	Método	Método para consultar un cliente
actualizarCliente	Método	Método para actualizar un cliente
ingresarEmpleado	Método	Método para ingresar un nuevo empleado.
eliminarEmpleado	Método	Método para eliminar empleado.
consultarEmpleados	Método	Método para consultar un empleado.
actualizarEmpleado	Método	Método para actualizar un empleado.
ingresarProveedor	Método	Método para ingresar un proveedor.
eliminarProveedor	Método	Método para eliminar un proveedor.
consultarProveedores	Método	Método para consultar un proveedor.
actualizarProveedor	Método	Método para actualizar un proveedor.
eliminarMascota	Método	Método para ingresar una mascota.
consultarMascota	Método	Método para eliminar una mascota.
ingresarMascota	Método	Método para consultar una mascota.
actualizarMascota	Método	Método para actualizar una mascota
System.exit	Salida	Ejecuta el salir del menú
catch	Opción	maneja excepciones relacionadas con operaciones en la base de datos (SQL)
if	Condicional	rowCount > 0

else	Condicional	Si no pasa algo, pasa otra condición.
clienteID	INT	ID del cliente en la tabla
nombre	String	Nombre del cliente
edad	int	Edad del cliente,
estadoCivil	String	Estado civil del cliente,
direccion	String	Dirección del cliente
nombre	String	Nombre empleado
codigoEmpleado	String	Código del empleado
area	String	Departamento de trabajo
sede	String	Sede de trabajo
nombre	String	Nombre del proveedor
telefono	String	Teléfono del vendedor
correo	String	Correo electrónico del vendedor
direccion	String	Dirección del vendedor
intentarInicioSesion	Método	Método para iniciar sesión
Mostrarmenuprincipal	Método	Método para mostrar menú principal
ProveedorID	Int	Id del proveedor
anioPublicacion	int	Fecha publicación del libro
cantidadInventario	int	Cantidad en Stock
menuClientes	Método	Método para el menú de clientes
menuEmpleados	Método	Método para el menú de empleados
resultSet	while	Condicional para saber si el libro está

V. ESQUEMA BASES DE DATOS

peluditos Clientes ClienteID : int Nombre : varchar(255) Edad : int EstadoCivil : varchar(50)	peluditos Empleados EmpleadoID : int Nombre : varchar(255) CodigoEmpleado : varchar(10) Area : varchar(50) Sede : varchar(50)
peluditos Productos idproducto : int nombreproducto : varchar(255) Precio : decimal(10,2) CantidadStock : int disponible : tinyint(1)	peluditos Mascotas IdMascota : varchar(15) Nombre : varchar(255) Raza : varchar(15) Color : varchar(255) NombreDueno : varchar(255)

Figura 4.

```
-- phpMyAdmin SQL Dump
-- version 5.1.1deb5ubuntu1
-- https://www.phpmyadmin.net/
--
-- Servidor: localhost:3306
-- Tiempo de generación: 28-11-2023 a las 06:44:41
-- Versión del servidor: 8.0.35-0ubuntu0.22.04.1
-- Versión de PHP: 8.1.2-1ubuntu2.14

SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";
START TRANSACTION;
SET time_zone = "+00:00";

/*!40101 SET
@OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_C
LIENT */;
/*!40101 SET
@OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_
RESULTS */;
/*!40101 SET
@OLD_COLLATION_CONNECTION=@@COLLATION_CONNE
CTION */;
/*!40101 SET NAMES utf8mb4 */;

--
-- Base de datos: `peluditos`
--
```



```

-- -----
--
-- Estructura de tabla para la tabla
`Clientes`
--
CREATE TABLE `Clientes` (
  `ClienteID` int NOT NULL,
  `Nombre` varchar(255) NOT NULL,
  `Edad` int DEFAULT NULL,
  `EstadoCivil` varchar(50) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4
COLLATE=utf8mb4_0900_ai_ci;

--
-- Volcado de datos para la tabla
`Clientes`
--

INSERT INTO `Clientes` (`ClienteID`,
`Nombre`, `Edad`, `EstadoCivil`) VALUES
(1, 'David', 33, 'Casado'),
(2, 'Yesica', 30, 'soltera'),
(3, 'Alejandra', 10, 'Soltera');

-- -----
--
-- Estructura de tabla para la tabla
`Empleados`
--
CREATE TABLE `Empleados` (
  `EmpleadoID` int NOT NULL,
  `Nombre` varchar(255) NOT NULL,
  `CodigoEmpleado` varchar(10) NOT NULL,
  `Area` varchar(50) DEFAULT NULL,
  `Sede` varchar(50) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4
COLLATE=utf8mb4_0900_ai_ci;

-- -----
--

```

```

-- Estructura de tabla para la tabla
`Mascotas`
--
CREATE TABLE `Mascotas` (
  `IdMascota` varchar(15) NOT NULL,
  `Nombre` varchar(255) NOT NULL,
  `Raza` varchar(15) DEFAULT NULL,
  `Color` varchar(255) DEFAULT NULL,
  `NombreDueno` varchar(255) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4
COLLATE=utf8mb4_0900_ai_ci;

-- -----
--
-- Estructura de tabla para la tabla
`Productos`
--
CREATE TABLE `Productos` (
  `idproducto` int NOT NULL,
  `nombreproducto` varchar(255) NOT NULL,
  `Precio` decimal(10,2) DEFAULT NULL,
  `CantidadStock` int DEFAULT NULL,
  `disponible` tinyint(1) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4
COLLATE=utf8mb4_0900_ai_ci;

--
-- Índices para tablas volcadas
--
-- Indices de la tabla `Clientes`
--
ALTER TABLE `Clientes`
  ADD PRIMARY KEY (`ClienteID`);

--
-- Indices de la tabla `Empleados`
--
ALTER TABLE `Empleados`
  ADD PRIMARY KEY (`EmpleadoID`);

--
-- Indices de la tabla `Mascotas`

```

```
--
ALTER TABLE `Mascotas`
  ADD PRIMARY KEY (`IdMascota`);

--
-- Indices de la tabla `Productos`
--
ALTER TABLE `Productos`
  ADD PRIMARY KEY (`idproducto`);

--
-- AUTO_INCREMENT de las tablas volcadas
--
--
-- AUTO_INCREMENT de la tabla `Clientes`
--
ALTER TABLE `Clientes`
  MODIFY `ClienteID` int NOT NULL
  AUTO_INCREMENT, AUTO_INCREMENT=4;

--
-- AUTO_INCREMENT de la tabla `Empleados`
--
ALTER TABLE `Empleados`
  MODIFY `EmpleadoID` int NOT NULL
  AUTO_INCREMENT;

--
-- AUTO_INCREMENT de la tabla `Productos`
--
ALTER TABLE `Productos`
  MODIFY `idproducto` int NOT NULL
  AUTO_INCREMENT;
COMMIT;

/*!40101 SET
CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLI
ENT */;
/*!40101 SET
CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RE
SULTS */;
/*!40101 SET
COLLATION_CONNECTION=@OLD_COLLATION_CONNECT
ION */;
```

VI. PROCESO VISTA JAVA Y CONEXIÓN CON SERVIDOR

Generamos la conexión entre Java (NetBeans) y un servidor funcional en Ubuntu Server 22.04 generando sincronización de los datos anteriormente establecidos en las tablas del punto V. Esquema base de datos.

Para iniciar, se creó un usuario: politecnico y una contraseña: 123456. Que, en caso de ser errónea en la vista de Java, informe que la contraseña es errónea y permita un total de 3 intentos para acceder:

```
-----
Usuario: 123
Contraseña: 456

Inicio de sesión fallido. Intentos restantes: 2
```

Figura 5.

Y, en caso de ser correcta permita continuar, mostrando la visualización del menú principal:

```
Usuario: politecnico
Contraseña: 123456

POLITECNICO INTERNACIONAL | PROGRAMACION II

Sistema de información para la gestión de Clientes, Empleados, Mascotas, Productos.

Samantha y sus peluditos - Menú Principal

1. GESTION DE CLIENTES
2. GESTION DE EMPLEADOS
3. GESTION DE MASCOTAS
4. GESTION DE PRODUCTO
5. SALIR DE LA APP

CREDITOS
Desarrollado por:
David Sebastian Enciso - david.enciso@pi.edu.co,
Cristian De Los Rios - cristian.de.los.rios@pi.edu.co y
Nicole Pajarito - nicole.pajarito@pi.edu.co
Samantha y sus peluditos | Bogotá D.C, Colombia | 2023

Selecciona una opción:
```

Figura 6.

Está su menú principal con opciones del 1 al 5, si seleccionamos 1, podremos gestionar los clientes:

```
Selecciona una opción:
1

1. Ingresar Cliente
2. Eliminar Cliente
3. Consultar Cliente
4. Actualizar Cliente
5. Volver
6. Salir

Selecciona una opción:
```

Figura 7.

Esto abrirá un submenú en el tendremos 6 opciones para elegir, las 4 iniciales para administrar “Clientes”, otra para volver al menú principal y la última para salir de la aplicación directamente.

Si elegimos la primera opción, nos permite agregar un nuevo cliente:

```

Selecciona una opción:
1
Nombre del cliente: Pablito
Edad: 18
Estado Civil: Soltero
Cliente ingresado con éxito.

```

Figura 8.

Proceso de inserción que podemos validar además en el servidor:

```

mysql> select * from Clientes;
+-----+-----+-----+-----+
| ClienteID | Nombre | Edad | EstadoCivil |
+-----+-----+-----+-----+
| 1 | David | 33 | Casado |
| 2 | Yesica | 30 | soltera |
| 3 | Alejandra | 10 | Soltera |
| 4 | Pablito | 18 | Soltero |
+-----+-----+-----+-----+

```

Figura 9.

De igual forma, se pueden eliminar, actualizar y consultar los “Clientes” tanto desde Java como directamente desde la base de datos.

```

Selecciona una opción:
3
ID: 1, Nombre: David, Edad: 33, Estado Civil: Casado
ID: 2, Nombre: Yesica, Edad: 30, Estado Civil: soltera
ID: 3, Nombre: Alejandra, Edad: 10, Estado Civil: Soltera
ID: 4, Nombre: Pablito, Edad: 18, Estado Civil: Soltero

```

Figura 10.

Seleccionando la opción 5 “volver”, regresamos al menú principal, donde podemos generar nuevas opciones:

```

Selecciona una opción:
5
POLITECNICO INTERNACIONAL | PROGRAMACION II
Sistema de información para la gestión de Clientes, Empleados, Mascotas, Productos.
Samantha y sus peluditos - Menú Principal
1. GESTION DE CLIENTES
2. GESTION DE EMPLEADOS
3. GESTION DE MASCOTAS
4. GESTION DE PRODUCTO
5. SALIR DE LA APP

```

Figura 11.

Al igual que con “Clientes”, podemos gestionar cualquier otra tabla mediante Java, por ejemplo, administrar la opción 4 “Gestionar producto”:

```

Selecciona una opción:
4
1. Ingresar Producto
2. Eliminar Producto
3. Consultar Producto
4. Actualizar Producto
5. Volver
6. Salir
Selecciona una opción:

```

Figura 12.

Allí ingresaremos ahora un nuevo producto:

```

Selecciona una opción:
1
Nombre del producto: Collar para perros.
Precio: 35.000
Cantidad en stock: 45
¿Disponible? (true/false): true
Producto ingresado con éxito.

```

Figura 13.

Lo consultamos desde Java:

```

3
ID: 1, Nombre: Collar, Precio: 20000.0, Cantidad en Stock: 37, Disponible: true
ID: 2, Nombre: Collar para perros., Precio: 35000.0, Cantidad en Stock: 45, Disponible: true

```

Y lo validamos desde la base de datos en Mysql:

```

mysql> select * from Productos;
+-----+-----+-----+-----+-----+
| idproducto | nombreproducto | Precio | CantidadStock | disponible |
+-----+-----+-----+-----+-----+
| 1 | Collar | 20000.00 | 37 | 1 |
| 2 | Collar para perros. | 35000.00 | 45 | 1 |
+-----+-----+-----+-----+-----+

```

Figura 14.

Desde Java también podemos modificar:

```

Selecciona una opción:
4
ID del producto a actualizar: 1
Nuevo nombre del producto: Shampu perros.
Nuevo precio: 46.500
Nueva cantidad en stock: 14
¿Nuevo estado de disponibilidad? (true/false): false
Producto actualizado con éxito.

```

Figura 15.

Y podemos validar su actualización también en Mysql:

```

mysql> select * from Productos;
+-----+-----+-----+-----+-----+
| idproducto | nombreproducto | Precio | CantidadStock | disponible |
+-----+-----+-----+-----+-----+
| 1 | Shampu perros. | 46500.00 | 14 | 0 |
| 2 | Collar para perros. | 35000.00 | 45 | 1 |
+-----+-----+-----+-----+-----+
2 rows in set (0.00 sec)

```

Figura 16.

En lo anterior, podemos validar un proceso con inicio de sesión, además de un CRUD, con menú principal y submenú para la creación, eliminación, consulta y actualización de datos de una base de datos.

VII. CRUD Y CONEXIÓN LAMP

Mediante MySql se generó la creación de una base llamada example_database, que contiene 3 tablas:

```
mysql> show tables;
+-----+
| Tables_in_example_database |
+-----+
| mascotas                    |
| todo_list                   |
| users                       |
+-----+
```

Figura 22.

A continuación, la descripción de cada tabla generada:
Tabla “mascotas”:

```
mysql> describe mascotas;
+-----+
| Field      | Type          | Null | Key | Default | Extra          |
+-----+
| item_id    | int           | NO   | PRI | NULL    | auto_increment |
| id_mascota | varchar(50)   | NO   |     | NULL    |                |
| nombre     | varchar(100)  | NO   |     | NULL    |                |
| raza       | varchar(50)   | NO   |     | NULL    |                |
| color      | varchar(50)   | NO   |     | NULL    |                |
| nombre_dueno | varchar(100) | NO   |     | NULL    |                |
+-----+
```

Figura 23.

Tabla todo_list:

```
mysql> describe todo_list;
+-----+
| Field      | Type          | Null | Key | Default | Extra          |
+-----+
| item_id    | int           | NO   | PRI | NULL    | auto_increment |
| nombre     | varchar(255)  | NO   |     | NULL    |                |
| edad       | int           | NO   |     | NULL    |                |
| estado_civil | varchar(255) | NO   |     | NULL    |                |
+-----+
```

Figura 24.

Tabla users:

```
mysql> describe users;
+-----+
| Field      | Type          | Null | Key | Default | Extra          |
+-----+
| user_id    | int           | NO   | PRI | NULL    | auto_incremer |
| username   | varchar(50)   | NO   | UNI | NULL    |                |
| password   | varchar(255)  | NO   |     | NULL    |                |
+-----+
```

Figura 25.

Esta última almacenará los datos de los usuarios creados y sus respectivas contraseñas para ser usados en el sitio web y su inicio de sesión.

Para generar la conexión, se realizó la creación de un nuevo usuario con roles en Ubuntu Server llamado **example_user**. Esto mediante los comandos:

```
CREATE USER 'example_user'@'%' IDENTIFIED BY 'password';
GRANT ALL PRIVILEGES ON *.* TO 'example_user'@'%' WITH GRANT OPTION;
FLUSH PRIVILEGES;
```

Luego se creó el Index de la página y su anexo llamado “Crud2” el cual inicia con las siguientes líneas de código para conectar con el usuario creado anteriormente y las tablas ya generadas:

```
<?php
$user = "example_user";
$password = "password";
$database = "example_database";
$clientTable = "todo_list";
$petTable = "mascotas";
```

Se crearon funciones para leer y mostrar en pantalla los datos ingresados por teclado, tanto para la tabla mascotas, como para la tabla de clientes llamada todo_list:

```
try {
    $db = new
PDO("mysql:host=localhost;dbname=$database"
, $user, $password);
    $db->setAttribute(PDO::ATTR_ERRMODE,
PDO::ERRMODE_EXCEPTION);

    // Función para leer y mostrar todos
los elementos de la tabla de clientes
function readClientData($db, $table)
{
    echo "<h2>Clientes</h2>";
    echo "<table border='1'>";
    echo
"<tr><th>Nombre</th><th>Edad</th><th>Estado
Civil</th><th>Acciones</th></tr>";
    $query = $db->query("SELECT * FROM
$table");
    while ($row = $query-
>fetch(PDO::FETCH_ASSOC)) {
        echo "<tr>
            <td>{$row['nombre']}</td>
            <td>{$row['edad']}
años</td>
            <td>{$row['estado_civil']}<
/td>
            <td>
                <a
href='?action=edit&table=todo_list&id={$row
['item_id']}'><button>Editar</button></a>
```

```

                <a
href='?action=delete&table=todo_list&id={$r
ow['item_id']}'><button>Eliminar</button></
a>

                </td>
            </tr>";
        }
        echo "</table>";
    }
}

```

Esta sería la visualización en el código.

Y su visualización en el sitio web, con CSS incluido sería:

Clientes			
Nombre	Edad	Estado Civil	Acciones
Nicole	18 años	Soltero	Editar Eliminar

Mascotas					
ID Mascota	Nombre	Raza	Color	Nombre del Dueño	Acciones
1	Pepa	French	Marrón	Hugo	Editar Eliminar

Figura 26.27.

Donde visualmente en el mismo campo aparece la opción para “Editar” y “Eliminar” los datos almacenados.

De igual forma, se comparte parte del código usado para editar y eliminar los campos:

```

// Función para actualizar un cliente
en la tabla
function updateClientData($db, $table,
$id, $nombre, $edad, $estadoCivil)
{
    $stmt = $db->prepare("UPDATE $table
SET nombre = :nombre, edad = :edad,
estado_civil = :estadoCivil WHERE item_id =
:id");
    $stmt->bindParam(':nombre',
$nombre);
    $stmt->bindParam(':edad', $edad,
PDO::PARAM_INT);
    $stmt->bindParam(':estadoCivil',
$estadoCivil);
    $stmt->bindParam(':id', $id);
    $stmt->execute();
}

```

El anterior para actualizar y el siguiente para eliminar:

```

// Función para eliminar un cliente de
la tabla
function deleteClientData($db, $table,
$id)
{
    $stmt = $db->prepare("DELETE FROM
$table WHERE item_id = :id");
    $stmt->bindParam(':id', $id);
    $stmt->execute();
}

// Función para eliminar una mascota de
la tabla
function deletePetData($db, $table,
$id)
{
    $stmt = $db->prepare("DELETE FROM
$table WHERE item_id = :id");
    $stmt->bindParam(':id', $id);
    $stmt->execute();
}

```

También se manejó el siguiente código para el procesamiento de inserción y actualización de datos:

```

// Procesamiento del formulario de
inserción o actualización
if ($_SERVER['REQUEST_METHOD'] ===
'POST') {
    if (isset($_POST['insert'])) {
        if ($_POST['table'] ===
'todo_list') {
            $nombre = $_POST['nombre'];
            $edad = $_POST['edad'];
            $estadoCivil =
$_POST['estado_civil'];
            insertClientData($db,
$clientTable, $nombre, $edad,
$estadoCivil);
        } elseif ($_POST['table'] ===
'mascotas') {
            $idMascota =
$_POST['id_mascota'];
            $nombre = $_POST['nombre'];
            $raza = $_POST['raza'];
            $color = $_POST['color'];

```

```

        $nombreDueno =
$_POST['nombre_dueno'];
        insertPetData($db,
$petTable, $idMascota, $nombre, $raza,
$color, $nombreDueno);
    }
} elseif (isset($_POST['update']))
{
    if ($_POST['table'] ===
'todo_list') {
        $id = $_POST['id'];
        $nombre = $_POST['nombre'];
        $edad = $_POST['edad'];
        $estadoCivil =
$_POST['estado_civil'];
        updateClientData($db,
$clientTable, $id, $nombre, $edad,
$estadoCivil);
    } elseif ($_POST['table'] ===
'mascotas') {
        $id = $_POST['id'];
        $idMascota =
$_POST['id_mascota'];
        $nombre = $_POST['nombre'];
        $raza = $_POST['raza'];
        $color = $_POST['color'];
        $nombreDueno =
$_POST['nombre_dueno'];
        updatePetData($db,
$petTable, $id, $idMascota, $nombre, $raza,
$color, $nombreDueno);
    }
}
} catch (PDOException $e) {
    print "Error!: " . $e->getMessage() .
"<br/>";
    die();
}
?>

```

Estas fueron las páginas trabajadas en HTML y PHP:

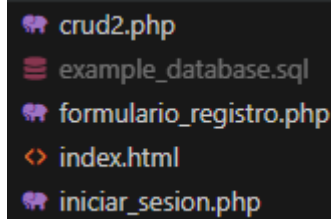


Figura 28.

Y a continuación el formulario para el registro de usuarios:

```

<?php
if ($_SERVER['REQUEST_METHOD'] === 'POST')
{
    $user = "example_user";
    $password = "password";
    $database = "example_database";
    $table = "users";

    try {
        $db = new
PDO("mysql:host=localhost;dbname=$database"
, $user, $password);
        $db-
>setAttribute(PDO::ATTR_ERRMODE,
PDO::ERRMODE_EXCEPTION);

        $username = $_POST['username'];
        $password = $_POST['password'];

        // Hash de la contraseña
        $hashedPassword =
password_hash($password, PASSWORD_DEFAULT);

        // Insertar usuario en la base de
datos
        $query = $db->prepare("INSERT INTO
$table (username, password) VALUES
(:username, :password)");
        $query->bindParam(':username',
$username);
        $query->bindParam(':password',
$hashedPassword);
        $query->execute();

        echo '<script>alert("Usuario
registrado exitosamente.");</script>';
    }
}

```

```

    } catch (PDOException $e) {
        echo '<script>alert("Error de
conexión a la base de datos: ' . $e-
>getMessage() . '");</script>';
    }
}
?>
<form method="post" action="">
    <h2>Registro de Usuario</h2>

    <label for="username">Usuario:</label>
    <input type="text" id="username"
name="username" required>

    <label
for="password">Contraseña:</label>
    <input type="password" id="password"
name="password" required>

    <button
type="submit">Registrarse</button>
</form>

```

En el código de “iniciar_sesion”, se valida si el usuario ya está autenticado y registrado, pues de ser así, será redirigido a la página de “Todolist”, es decir que ya podrá realizar el CRUD:

```

<?php
session_start();

// Verificar si el usuario ya está
autenticado, redirigir a Todolist si es así
if (isset($_SESSION['user_id'])) {
    header("Location: crud2.php");
    exit();
}

if ($_SERVER['REQUEST_METHOD'] === 'POST')
{
    $user = "example_user";
    $password = "password";
    $database = "example_database";
    $table = "users";

    try {

```

```

        $db = new
PDO("mysql:host=localhost;dbname=$database"
, $user, $password);
        $db-
>setAttribute(PDO::ATTR_ERRMODE,
PDO::ERRMODE_EXCEPTION);

        // Obtener datos del formulario
$username = $_POST['username'];
$password = $_POST['password'];

        // Consulta para obtener el usuario
$query = $db->prepare("SELECT *
FROM $table WHERE username = :username");
        $query->bindParam(':username',
$username);
        $query->execute();
        $userRow = $query-
>fetch(PDO::FETCH_ASSOC);

        // Verificar la contraseña
utilizando password_verify
        if ($userRow &&
password_verify($password,
$userRow['password'])) {
            // Credenciales válidas,
establecer la sesión y redirigir a Todolist
            $_SESSION['user_id'] =
$userRow['user_id'];
            header("Location: crud2.php");
            exit();
        } else {
            $error = "Usuario o contraseña
incorrectos";
        }

    } catch (PDOException $e) {
        $error = "Error de conexión a la
base de datos";
    }
}
?>

```

VIII. MOCKUPS

Menú de navegación:

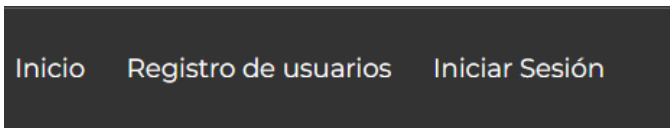


Figura 29.

Índex:

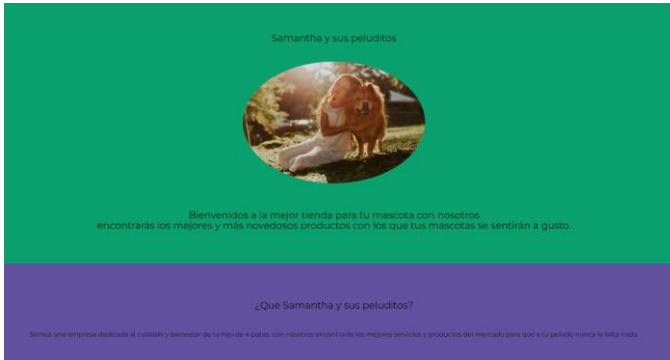


Figura 30.

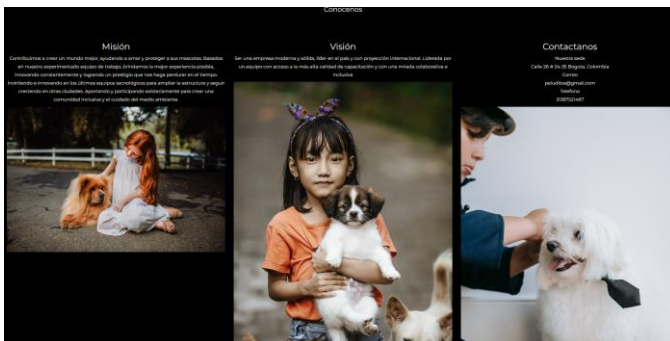


Figura 31.

Footer:



Figura 32.

Pantalla de registro de usuario:

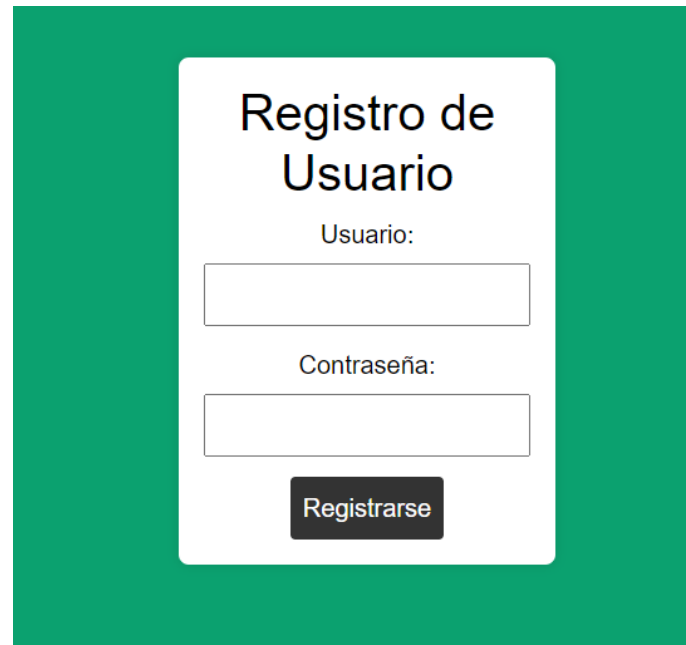


Figura 33.

Si ya existe el usuario o hay algún error, aparecerá ventana emergente notificando:

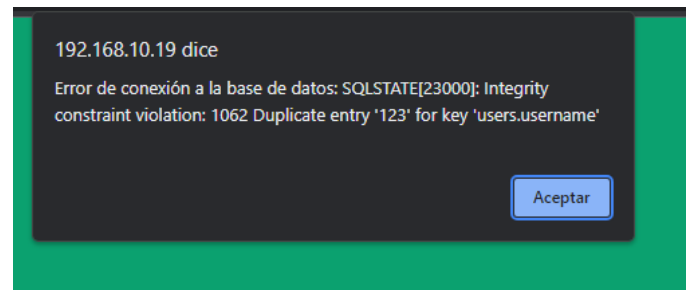


Figura 34.

Si se registra exitosamente, también será notificado:

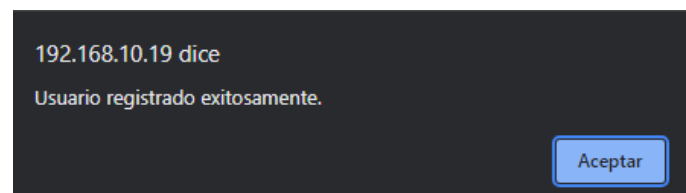


Figura 35.

Una vez registrado, podrá iniciar sesión.

Si el usuario no existe, aparece mensaje:

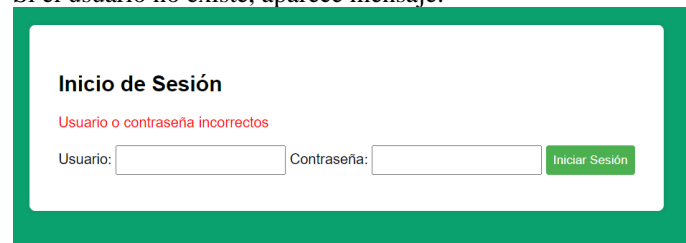


Figura 36.

Si el usuario sí está registrado, será remitido a las tablas para gestionar clientes y mascotas:

Figura 37.

Esta cuenta también con su propio menú superior:

En el que podrán intercambiar entre la gestión de clientes o mascotas:

Figura 38.

Se podrán gestionar los clientes para agregar, editar, eliminar o simplemente visualizar los existentes:

Figura 39.

Lo que se vaya agregando, quedará en la tabla superior, con opción de editar o eliminar:

Nombre	Edad	Estado Civil	Acciones
Nicole	18 años	Soltero	Editar Eliminar
Lupe	17 años	Soltera	Editar Eliminar

Figura 40.

A su vez, estará el botón para “agregar datos”:

Figura 41.

Esto aplica tanto para Clientes como para Mascotas y todos los procesos se reflejarán a su vez y de forma inmediata en el servidor:

Figura 42.

Nicole	25 años	Soltero	Editar Eliminar
--------	---------	---------	-----------------

Figura 43.

```
mysql> select * from mascotas;
+-----+-----+-----+-----+-----+-----+
| item_id | id_mascota | nombre | raza           | color | nombre_dueno |
+-----+-----+-----+-----+-----+-----+
| 1       | 1          | Pepa   | French         | Marrón | Hugo          |
| 2       | 2          | Yoshi  | Cocker Spaniel | Café   | Camilo        |
+-----+-----+-----+-----+-----+-----+
```

Figura 44.

```
mysql> select * from todo_list;
+-----+-----+-----+-----+
| item_id | nombre | edad | estado_civil |
+-----+-----+-----+-----+
| 5       | Nicole | 25   | Soltero      |
| 6       | Lupe   | 17   | Soltera      |
| 7       | David  | 25   | Casado       |
| 8       | Camilo | 25   | Soltero      |
+-----+-----+-----+-----+
```

Figura 45.

Este sería como tal el proceso de CRUD mediante LAMP, conexión entre una base de datos, servidor Ubuntu, código en HTML y PHP.

IX. DIAGRAMAS

Diagrama de clases:

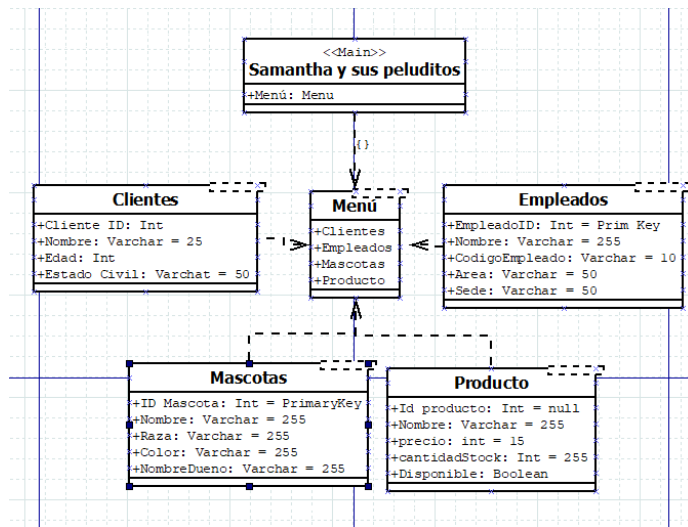


Figura 17.

Diagrama de secuencias:

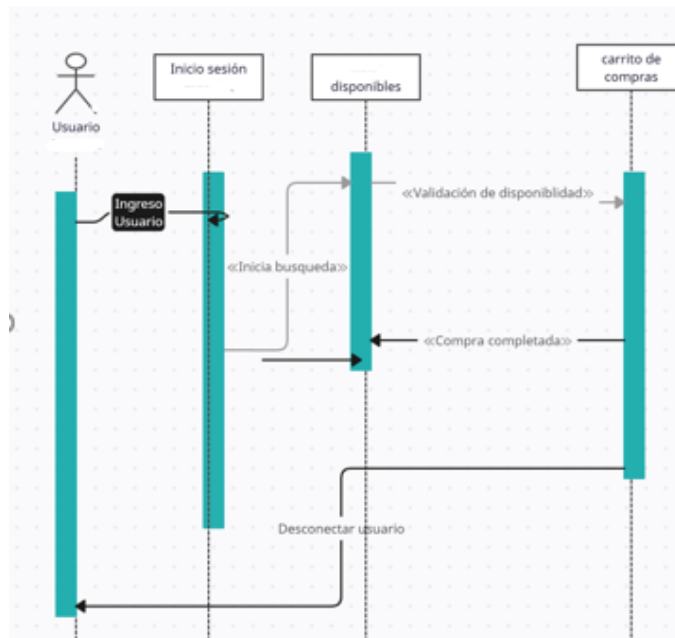


Figura 18.

X. TECNOLOGÍAS USADAS:

Product Version: **Apache NetBeans IDE 17**

Updates: Updates available

Java: 15.0.2; Java HotSpot(TM) 64-Bit Server VM

15.0.2+7-27

Runtime: Java(TM) SE Runtime Environment 15.0.2+7-27

System: Windows 10 version 10.0 running on amd64; Cp1252; es_CO (nb)

Para el diagrama se ha utilizado:

dia.exe 0.97.2 // Un programa para dibujar diagramas estructurados. // (C) 1998-2009 The Free Software Foundation and the authors

Interfaz Gráfica de Usuario de VirtualBox

Versión 7.0.8 r156879 (Qt5.15.2)

Copyright © 2023 Oracle and/or its affiliates

Ubuntu-22.04-live-server-amd64

XI. LISTA DE IMÁGENES:

Figura 1. Requerimientos funcionales. - Creación propia

Figura 2. Requerimientos funcionales Ubuntu. -Creación propia

Figura 3. Requerimientos funcionales Visual Code. - Creación propia

Figura 4. EsquemaBD. -Creación propia

Figura 5. Inicio sesión erróneo. Creación propia

Figura 6. Inicio sesión exitoso, menú principal. -Creación propia

Figura 7. Menú principal opción 1 Clientes. -Creación propia

Figura 8. Clientes opción 1. -Creación propia

Figura 9. Visualización Clientes Mysql -Creación propia

Figura 10. Consulta clientes Java -Creación propia

Figura 11. Opción 5 "volver" -Creación propia

Figura 12. Opción 4 "Productos". -Creación propia

Figura 13. ingreso de nuevos Productos. -Creación propia

Figura 14. Consulta "Productos" en Mysql. -Creación propia

Figura 15. Actualizar "Productos" Java. -Creación propia

Figura 16. Validar actualización en Mysql. -Creación propia

Figura 17. Diagrama de clases. -Creación propia

Figura 18. Diagrama de secuencias -Creación propia

Figura 19. Foto ingranje 1. -Creación propia

Figura 20. Foto ingranje 2. -Creación propia

Figura 21. Foto ingranje 3. -Creación propia

Figura 22. show tables. -Creación propia

Figura 23. Tabla mascotas. -Creación propia

Figura 24. Tabla todo_list. -Creación propia

Figura 25. Tabla users. -Creación propia

Figura 26. Visualización con CSS clientes. -Creación propia

Figura 27. Visualización con Css Mascotas. -Creación propia
 Figura 28. Páginas usadas. -Creación propia
 Figura 29. Mockup menú. -Creación propia
 Figura 30. Mockup Index principal. -Creación propia
 Figura 31. Mockup Index misión, visión, contacto. -Creación propia
 Figura 32. Mockup Footer. -Creación propia
 Figura 33. Registro usuario. -Creación propia
 Figura 34. Alerta usuario existente/error. -Creación propia
 Figura 35. Alerta, usuario existoso. -Creación propia
 Figura 36. Contraseña o usuario erroneo. -Creación propia
 Figura 37. Gestión usuario/mascotas -Creación propia
 Figura 38. Menú usuario mascota. -Creación propia
 Figura 39. Clientes agregar/editar/ver/eliminar. -Creación propia
 Figura 40. Editar / eliminar clientes. -Creación propia
 Figura 41. Button agregar. -Creación propia
 Figura 42. Edición. -Creación propia
 Figura 43. Resultado edición. -Creación propia
 Figura 44. Mascotas en Mysql. -Creación propia
 Figura 45. Clientes en Mysql (Todo_list). -Creación propia

XII. REFERENCIAS:

[1] Software libre, Linux y Ubuntu.pdf (um.es)
 [2]B, G. (2018, August 31). ¿Qué es Apache? Descripción completa. Tutoriales Hostinger.
<https://www.hostinger.es/tutoriales/que-es-apache/>
 [4]Robledano, A. (2019, September 24). Qué es MySQL: Características y ventajas. OpenWebinars.net.
<https://openwebinars.net/blog/que-es-mysql/>
 [5]Qué es PHP. (n.d.). Desarrolloweb.com.
<https://desarrolloweb.com/articulos/392.php>
 [6]Licencia de plantilla - HTML Codex
 [7]Online Shop Website Template Free Download - HTML Codex
 [8]Arquitectura de 3 Capas: Simplificando el Diseño de Software - Desarrollo de software. (2023, September 15).
<http://desarrollo-de-software.com/arquitectura-de-3-capas-simplificando-el-diseno-de-software/>
 [9] El patrón modelo-vista-controlador: Arquitectura y frameworks explicados.(2021, June 28).FreeCodeCamp.org.
<https://www.freecodecamp.org/espanol/news/el-modelo-de-arquitectura-view-controller-pattern/>
 [10]¿Qué es el CSS? - Aprende sobre desarrollo web | MDN. (n.d.). Developer.mozilla.org.
https://developer.mozilla.org/es/docs/Learn/CSS/First_steps/What_is_CSS
 [11] HTML. (2020, January 3). Documentación Web de MDN.
<https://developer.mozilla.org/es/docs/Web/HTML>
 [12] ¿Qué es JavaScript? - Aprende sobre desarrollo web | MDN. (n.d.). Developer.mozilla.org.
https://developer.mozilla.org/es/docs/Learn/JavaScript/First_steps/What_is_JavaScript
 [13] La teoría del color según Newton, Goethe, Turner y otros grandes artistas. (n.d.). Ttamayo.com.

[14] <https://www.ttamayo.com/2019/07/la-teoria-del-color/>
 [15] <https://app.creately.com/d/XcyiIgyNMJx/edit/s/epQ38rwo1QD>

XIII. CRÉDITOS



Figura 19.

David Sebastian Enciso Lopez

Desarrollador junior, apasionado por el futbol y estudiante del politécnico internacional, dentro de mis proyectos realizados a la fecha se pueden destacar la elaboración de código para el funcionamiento de un cajero electrónico mediante consola, programa de consulta para la Liga Betplay femenina con interfaz gráfica bajo el modelo-vista-controlador, este último es un maravillo y novedoso proyecto para los amantes del futbol femenino colombiano.

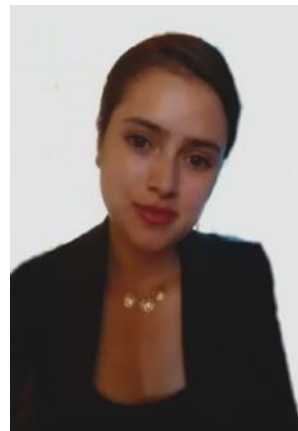


Figura 20.

Nicole Dayana Pajarito Santamaria

Mi nombre es Nicole Dayana Pajarito Santamaria, tengo 26 años, me gusta leer, correr, viajar, escuchar música, séptimo arte. Actualmente estoy estudiando una tecnología en Desarrollo de Software en el Politécnico Internacional. Este proyecto hace parte de Programación II, IV ciclo.



Figura 21.

Cristian Camilo De Los Rios Rodríguez.

Me gustan los video juegos, las películas, escuchar música y viajar por carretera.

Actualmente en proceso como tecnólogo de desarrollo, espero ser un excelente programador a futuro y poner en práctica todo lo aprendido en el transcurso de estos ciclos.