Manual de Programador

JG PET
Juan Sebastian Alzate Gomez
Justine Bravo Gomez
11/29/2024

Resumen-- Este documento describe el desarrollo de un sitio web para la veterinaria JG PET, diseñado para gestionar productos, vender artículos relacionados con el cuidado de mascotas y ofrecer servicios de atención veterinaria. El sistema permite a los administradores realizar operaciones como agregar, editar y gestionar inventario, mientras que los usuarios pueden explorar productos, realizar compras y programar citas. Este proyecto busca optimizar los procesos administrativos y mejorar la experiencia del cliente, siguiendo los estándares de desarrollo web.

Palabras clave: gestión de inventario, ventas en línea, atención veterinaria, Django, experiencia del usuario.

I. Introducción

Este documento presenta el desarrollo de un sitio web para la veterinaria JG PET, una plataforma diseñada para optimizar la gestión de productos, ventas y servicios veterinarios. La herramienta está dirigida a mejorar la eficiencia administrativa y la experiencia del usuario, permitiendo a los administradores manejar inventarios y a los clientes acceder fácilmente a productos y servicios. Este artículo detalla los aspectos técnicos y funcionales del sistema, proporcionando una base para entender su diseño y propósito.

II. Desarrollo de Contenidos

En esta sección se presentan los detalles técnicos y funcionales del sitio web desarrollado para la veterinaria JG PET. Cada apartado aborda un componente clave del sistema, estructurado de manera clara y secuencial para facilitar la comprensión.

A. Descripción del problema

El sitio web se compone de dos roles principales: administradores y usuarios. Los administradores gestionan el inventario, productos y servicios, mientras que los usuarios tienen acceso a la compra de productos y agendamiento de citas para sus mascotas.

B. Estructura técnica

El sistema está desarrollado en Django, utilizando Python como lenguaje principal, junto con HTML, CSS y JavaScript para el diseño y la interacción del usuario. La base de datos se gestiona con MySQL, garantizando un manejo eficiente y seguro de la información

C. Funcionalidades clave

1. Gestión de Productos: Agregar, editar y eliminar productos.
2. Inventario: Visualización y control del stock disponible.
3. Citas Veterinarias: Solicitud y programación de servicios para mascotas.
4. Compras en Línea: Carrito de compras para adquirir productos fácilmente.

III. METODOLOGIA O ENFOQUE DEL DESARROLLO

El desarrollo del sitio web para la veterinaria JG PET siguió un enfoque estructurado y secuencial, comenzando con la recopilación y análisis de los requerimientos del sistema. A partir de esta etapa inicial, se definieron las funcionalidades necesarias, asegurando que todas las necesidades del usuario estuvieran contempladas.

1. Planificación y diseño de la base de datos:

Se diseñó la base de datos utilizando MySQL, organizando la información en tablas según los requerimientos del sistema para garantizar un manejo eficiente y estructurado de los datos.

2. Planificación y diseño de la base de datos:

- Se descargó y configuró Django como framework principal, instalando las librerías necesarias y definiendo la estructura inicial del proyecto.
- En el archivo “settings.py”, se estableció la conexión entre el proyecto y la base de datos MySQL. Las tablas fueron replicadas en el archivo “models.py” y sincronizadas mediante comandos como “makemigrations” y “migrate”.

3. Diseño del sistema:

- Se inició con un diseño básico para definir la estructura visual del sistema, incluyendo la creación de logotipos, banners y una interfaz intuitiva para la página de inicio.

- El diseño fue pensado para ser dinámico, permitiendo la modificación de elementos como logotipos, nombres de la empresa e información principal directamente desde el sistema, sin necesidad de intervención en el código.

4. Desarrollo por módulos:

- Se creó un módulo independiente para cada funcionalidad del sistema (gestión de productos, citas, inventario, etc.).
- Cada módulo incluyó vistas (views.py) para definir la lógica de negocio, rutas (urls.py) para enlazar la lógica con las pantallas, y plantillas HTML (templates/) para la interfaz de usuario.
- Se utilizaron carpetas específicas para gestionar recursos estáticos como imágenes (static/), estilos CSS y scripts JavaScript.

4. Pruebas y corrección de errores:

- Durante el desarrollo, cualquier error encontrado fue solucionado recurriendo a recursos como videos tutoriales, inteligencia artificial, o soluciones compartidas por otros programadores.
- Se realizaron pruebas unitarias y funcionales para garantizar el correcto funcionamiento de cada módulo y la interacción entre ellos.

5. Iteración final:

Una vez que las funcionalidades principales y el diseño del sistema estuvieron completados, se realizaron ajustes finales para garantizar la usabilidad, eficiencia y cumplimiento de los objetivos del proyecto.

IV. INTERRACCIONES CLAVE DEL SISTEMA

El sistema soporta diversos escenarios, incluyendo la gestión de productos, el manejo de citas veterinarias y la compra en línea. Cada funcionalidad está diseñada para optimizar las tareas del administrador y mejorar la experiencia del cliente. Los casos de uso completos están disponibles en la documentación interna del proyecto.

V. RESULTADOS

Tras la implementación del sistema para la veterinaria JG PET, se lograron los siguientes beneficios clave:

1. Mejora en la eficiencia administrativa:

El sistema ha permitido a los administradores gestionar el inventario de productos de forma más eficiente, reduciendo el tiempo empleado en tareas manuales y mejorando la precisión de la información. Además, la capacidad de agregar, editar y eliminar productos de manera rápida ha optimizado los procesos internos.

2. Optimización en la gestión de citas:

La posibilidad de que los clientes agenden citas de forma autónoma ha reducido la carga administrativa en el personal, permitiendo que se concentren en otras tareas. Esto ha aumentado la eficiencia operativa y mejorado la disponibilidad de horarios para los clientes.

3. Mejora en la experiencia del cliente

Los usuarios ahora tienen una interfaz intuitiva para realizar compras en línea, visualizar productos y programar citas, lo que ha aumentado la satisfacción del cliente. Además, el sistema permite un acceso fácil y rápido a la información sobre los productos disponibles, mejorando la experiencia de compra.

4. Mayor control sobre el inventario:

El sistema ofrece una visibilidad completa del stock disponible, lo que ha permitido un mejor manejo de los productos y la prevención de desabastecimientos. Los administradores ahora pueden gestionar el inventario en tiempo real y recibir alertas cuando los niveles de stock son bajos.

Estos beneficios contribuyen al objetivo de optimizar los procesos internos, mejorar la atención al cliente y proporcionar un sistema flexible y adaptable a las necesidades de la veterinaria.

VI. BIBLIOTECAS Y HERRAMIENTAS UTILIZADAS

En esta sección se detallan las bibliotecas y herramientas utilizadas para el desarrollo del sistema, explicando su propósito y cómo contribuyen al correcto funcionamiento del sitio web de la veterinaria JG PET.

1. Django

Propósito: Framework web utilizado para desarrollar el backend del sistema. Django facilita la creación de aplicaciones web robustas, permitiendo gestionar base de datos, formularios, autenticación y más, de manera sencilla y eficiente.

Función: Se utilizó para la creación de vistas, controladores, manejo de formularios y comunicación con la base de datos MySQL.

2. Python

Propósito: Lenguaje de programación principal utilizado en el desarrollo del backend del sistema. Python es conocido por su simplicidad y versatilidad, siendo ideal para aplicaciones web rápidas y escalables.

Función: Se empleó para la lógica del backend, scripts de migración y gestión de bases de dato.

3. MySQL

Propósito: Sistema de gestión de bases de datos relacional utilizado para almacenar los datos del sistema, como la información de productos, clientes, citas y transacciones.

Función: Permite una gestión eficiente de los datos, garantizando su integridad y disponibilidad.

4. CSS

Propósito: Lenguaje de estilos utilizado para la presentación visual del sistema. Función: Se utilizó para

darle estilo a las páginas web, logrando una experiencia de usuario atractiva y funcional.

5. Java Script
   Propósito: Lenguaje de programación utilizado para implementar funcionalidades interactivas en el frontend.
   Función: Se utilizó para mejorar la interacción del usuario,
   Como validación de formularios y manipulación dinámica del contenido sin recargar la pagina.
6. Pathlib
   Propósito: pathlib es una librería de Python que ofrece clases para la manipulación y operación con rutas de archivos y directorios. La clase Path es el componente principal de esta librería y permite realizar operaciones de manera más sencilla y legible que utilizando cadenas de texto convencionales.
   Función: Se utiliza para crear, consultar y manipular rutas de archivos y directorios de forma independiente del sistema operativo, lo que hace que el código sea más portátil entre diferentes plataformas (Windows, Linux, macOS). Ejemplos de operaciones que se pueden hacer con Path incluyen la obtención de la ruta absoluta de un archivo, la comprobación de si un archivo o directorio existe, la creación de directorios y más.

7. {% load base64_filters %}
   Propósito: Esta línea carga un archivo de filtros personalizados en Django, en este caso, base64_filters. Los filtros en Django son funciones que permiten modificar o formatear los datos dentro de las plantillas de manera sencilla.
   Función: El archivo base64_filters probablemente contiene filtros personalizados que se utilizan para convertir datos en formato Base64 o realizar otras manipulaciones relacionadas con la codificación.
8. {% load static %}
   Propósito: Esta línea carga el sistema de archivos estáticos de Django. Los archivos estáticos incluyen archivos como CSS, JavaScript, imágenes y otros recursos que no cambian, pero son necesarios para el funcionamiento de la página web.
   Función: Django proporciona el tag {% static %} para obtener la URL de los archivos estáticos configurados en tu proyecto. Esto es útil cuando necesitas incluir recursos como archivos CSS, imágenes o scripts en tus plantillas.
9. Librería os
   Propósito: El módulo os en Python proporciona una manera de interactuar con el sistema operativo en el que el programa se está ejecutando. Permite realizar tareas como la manipulación de archivos y directorios, y la obtención de información sobre el sistema.
10. Librería sys
    Propósito: El módulo sys proporciona acceso a algunas variables y funciones específicas del entorno de ejecución de Python. A través de sys, se puede interactuar con los argumentos pasados al script, manipular el flujo de entrada/salida, y gestionar el entorno de ejecución.
11. django.shortcuts
    Propósito: Esta es una de las funciones más utilizadas dentro de django.shortcuts. Permite devolver una respuesta HTTP con una plantilla renderizada. Es útil cuando se necesita combinar una plantilla HTML con datos dinámicos.
    Parámetros:
    - request: Objeto que contiene la solicitud HTTP. template_name: El nombre del archivo de plantilla (HTML) que se desea renderizar.
    - context: Un diccionario que contiene las variables que se deben pasar a la plantilla.
12. django.contrib
    Es un conjunto de aplicaciones integradas que se incluyen de forma predeterminada con Django. Estas aplicaciones proporcionan funcionalidades que cubren muchos aspectos comunes en el desarrollo de aplicaciones web, como la gestión de usuarios, la administración de la base de datos, la carga de archivos estáticos, la seguridad, entre otras.
    - django.contrib.admin
      Propósito: Proporciona una interfaz de administración web para gestionar los modelos del sistema de manera eficiente.
      Función: Se utiliza para permitir que los administradores gestionen fácilmente los productos, clientes y otras entidades desde una interfaz gráfica.
    - django.contrib.auth
      Propósito: Maneja la autenticación y autorización de usuarios.
      Función: Se utiliza para gestionar el registro, inicio de sesión, y permisos de los usuarios, asegurando que solo los usuarios autorizados accedan a ciertas funcionalidades.
    - django.contrib.sessions
      Propósito: Gestiona el almacenamiento de datos entre solicitudes utilizando sesiones.
      Función: Permite mantener la información del usuario a lo largo de diferentes interacciones con el sitio, como el contenido del carrito de compras.
    - django.contrib.messages
      Propósito: Ofrece un sistema para mostrar mensajes de retroalimentación a los usuarios.

Función: Se usa para mostrar mensajes de éxito, error o advertencia a los usuarios, mejorando la interacción con el sistema.

- django.contrib.staticfiles

  Propósito: Gestiona y sirve archivos estáticos (CSS, JavaScript, imágenes). Función: Se utiliza para almacenar y acceder a archivos estáticos del sistema, garantizando que los recursos visuales se carguen correctamente.

- django.contrib.sites

  Propósito: Permite gestionar múltiples sitios web en un mismo proyecto. Función: Se utiliza para asociar contenido con diferentes dominios, permitiendo la administración de varios sitios desde una única base de datos.

13. base64

    Propósito: Esta biblioteca permite realizar codificación y decodificación de datos en formato Base64.

    Función: La función b64encode se utiliza para convertir datos binarios (como imágenes o archivos) en una representación de texto, que puede ser transmitida de forma segura a través de canales que solo permiten caracteres ASCII, como en URLs o en correos electrónicos. Se emplea comúnmente para manejar imágenes, archivos o datos sensibles que necesitan ser transportados como texto.

14. Stripe

    Propósito: Plataforma de pagos en línea que permite procesar pagos de clientes.

    Función: Se integró para gestionar el proceso de compras en línea de productos y servicios, permitiendo a los clientes realizar pagos de manera segura.

VI. Conclusiones y trabajo a futuro

El desarrollo del sitio web para la veterinaria JG PET ha logrado optimizar los procesos de gestión de productos, ventas y atención veterinaria, proporcionando una plataforma intuitiva tanto para los administradores como para los usuarios. El sistema ha mejorado significativamente la eficiencia administrativa, reduciendo tiempos de gestión y errores humanos, mientras que los clientes pueden realizar compras y agendar citas de manera rápida y sencilla.

Como trabajo futuro, se propone la integración de un sistema de notificaciones automáticas para recordar a los clientes sobre citas, productos recomendados o promociones. También se contempla la ampliación del carrito de compras, permitiendo una mayor personalización en los pagos, incluyendo métodos alternativos y un sistema de seguimiento de pedidos en tiempo real.