Universidad de la Cuenca del Plata

Autorización Definitiva. Decreto Poder Ejecutivo Nacional N°091/2006

Trabajo Práctico Nº 1

CARRERA: Ing. en Sistemas de Información.

MATERIA: Paradigmas y Lenguajes de Programación III.

COMISIÓN: “U” (única) “A” PROFESOR: Encina, Agustin.

ESTUDIANTE: Stupniki, Hernan.

FECHA: 02-09-2025.

Análisis y Diseño de la Aplicación Web “Bienes Raíces”

Introducción

La aplicación web “Bienes Raíces” fue desarrollada con el objetivo de crear un portal inmobiliario que permita gestionar propiedades para la venta. Utilizando tecnologías como HTML5, Sass (SCSS) y herramientas como Node.js y Gulp, se construyó una página con un diseño limpio y una estructura modular que permite un mantenimiento eficiente y una fácil escalabilidad.

El enfoque utilizado responde a la necesidad de aplicar buenas prácticas de desarrollo web, incorporando un flujo de trabajo que contempla la separación entre estructura, estilos y automatización, logrando un producto final profesional.

Objetivos del proyecto

El proyecto tiene como propósito principal presentar una aplicación web modular que utilice HTML5 para la estructura, Sass (SCSS) para la presentación, y un proceso de compilación y optimización basado en Node.js y Gulp.

Se busca que cada vista esté correctamente estructurada, que los estilos se encuentren organizados en módulos reutilizables y que la automatización permita obtener archivos minificados y optimizados para garantizar tiempos de carga rápidos y un mejor posicionamiento en buscadores.

Objetivos específicos

- Desarrollar una plataforma que permita mostrar propiedades de bienes raíces de manera clara y visualmente atractiva.
- Implementar una estructura modular en los estilos usando **Sass (SCSS)** para garantizar escalabilidad.
- Automatizar el proceso de desarrollo con **Gulp**, minimizando los tiempos de compilación y optimización.
- Asegurar que la aplicación sea completamente **responsiva**, adaptándose a diferentes dispositivos (móviles y escritorios).

Requisitos

- El sitio debe ser **estático** y **responsivo**.
- Debe contener al menos las siguientes secciones: **Página principal**, **Listado de propiedades**, **Detalle de propiedad**, **Formulario de compra**.
- Los **archivos HTML** deben estar correctamente estructurados y **validar** según los estándares de HTML5.
- Los **estilos deben ser gestionados** mediante **Sass** para asegurar una fácil escalabilidad y mantenimiento.

- El proyecto debe contar con una **automatización de tareas** usando **Gulp** para compilar **Sass**, minificar **CSS** y **JS**, y optimizar **imágenes**.

- El código debe estar **documentado** y organizado siguiendo buenas prácticas de desarrollo web.

Tecnologías utilizadas

HTML5

HTML es el lenguaje de marcado fundamental de la web, responsable de definir la estructura y jerarquía del contenido. En esta aplicación, se utiliza HTML5 debido a que incorpora etiquetas semánticas como `<header>`, `<nav>`, `<section>`, `<article>` y `<footer>`.

Estas etiquetas no solo facilitan la lectura del código por parte de los desarrolladores, sino que además aportan a la accesibilidad, ya que permiten a los lectores de pantalla y a los motores de búsqueda interpretar correctamente el contenido.

Se utiliza HTML5 porque asegura una mayor compatibilidad con estándares modernos y porque mejora el SEO y la accesibilidad de la aplicación.

Sass (SCSS)

Sass es un preprocesador de CSS, y en este proyecto se utiliza la sintaxis SCSS por su cercanía con CSS tradicional y su mayor legibilidad.

Se emplea porque extiende las capacidades de CSS, permitiendo:

- **Variables**, para definir colores, fuentes o tamaños de manera centralizada. Esto facilita mantener coherencia y modificar estilos globales de forma simple.
- **Anidación (nesting)**, que mejora la organización de los selectores y evita repeticiones innecesarias.
- **Mixins**, que permiten reutilizar bloques de código, como definiciones de grillas o breakpoints de responsive design.
- **Funciones y operadores**, que posibilitan realizar cálculos dentro de las hojas de estilo.
- **Modularización en parciales**, dividiendo los estilos en diferentes archivos (`_variables.scss`, `_mixins.scss`, `_layout.scss`, etc.), lo que promueve el principio de responsabilidad única y facilita el trabajo colaborativo.

La elección de Sass se justifica porque simplifica la escritura de estilos, incrementa la mantenibilidad del código y permite escalar el proyecto de forma ordenada.

Node.js y npm

Node.js es un entorno de ejecución de JavaScript que se utiliza en el lado del servidor o, en este caso, en la terminal del desarrollador. npm, su gestor de paquetes, facilita la instalación de librerías necesarias para automatizar y optimizar el proyecto.

Se utiliza porque permite instalar y gestionar dependencias de manera estandarizada, garantizando que cualquier desarrollador pueda reproducir el entorno de trabajo con un simple `npm install`.

En este proyecto, Node y npm son la base que permite integrar herramientas como Gulp y sus plugins para compilar Sass, minificar CSS y JS, y optimizar imágenes.

Gulp y sus plugins

Gulp es un *task runner*, es decir, un sistema para definir y ejecutar tareas automatizadas.

Se utiliza porque evita procesos manuales repetitivos, garantizando consistencia y eficiencia. Gracias a Gulp se automatiza la compilación de SCSS, la minificación de CSS y JavaScript, la optimización de imágenes y la recarga automática en el navegador.

Entre los plugins más relevantes utilizados se encuentran:

- **gulp-sass** y **sass**: compilan los archivos SCSS a CSS plano.
- **gulp-postcss** con **autoprefixer**: asegura compatibilidad con navegadores antiguos mediante la inserción automática de prefijos.
- **cssnano**: minifica el CSS reduciendo su tamaño.
- **gulp-sourcemaps**: facilita la depuración al vincular el CSS generado con el SCSS original.
- **gulp-imagemin** y **gulp-webp**: optimizan las imágenes reduciendo su peso y generando formatos modernos.
- **gulp-terser-js**: minifica JavaScript para mejorar su rendimiento en producción.
- **gulp-cache y gulp-clean**: gestionan la caché de imágenes y limpian archivos previos.

La elección de Gulp se justifica porque es una herramienta liviana, flexible y ampliamente utilizada, que permite personalizar el flujo de trabajo del desarrollador y optimizar los archivos finales.

Minificación y cssnano

La minificación es el proceso de reducir el tamaño de los archivos eliminando espacios, saltos de línea, comentarios y caracteres innecesarios, sin modificar su funcionamiento.

Se utiliza porque contribuye directamente a mejorar el rendimiento del sitio. Un archivo más liviano se descarga más rápido, reduciendo el tiempo de carga de la página y el consumo de ancho de banda.

En este proyecto, se utiliza **cssnano** como minificador de CSS. cssnano no solo elimina caracteres innecesarios, sino que también optimiza reglas redundantes y convierte valores largos en equivalentes más cortos (por ejemplo, #ffffff a #fff).

Se emplea **cssnano** porque garantiza que las hojas de estilo sean lo más ligeras posibles, mejorando la experiencia del usuario, el rendimiento en dispositivos móviles y el posicionamiento SEO.

Análisis de las vistas HTML y relación con la consigna

A continuación, se describe cómo cada archivo HTML se relaciona con la consigna original y los archivos creados en el proyecto.

1. Portada principal (index.html)

- **Consigna:** Se requiere un archivo llamado `index.html` como la portada principal.
- **Archivo Actual:** `index.html` (Este archivo cumple con la función de la portada principal, mostrando el encabezado, un breve saludo y enlaces a otras secciones).

Relación con la consigna: index.html → Portada principal

2. Sección que liste todos los productos en forma de tabla (listado_tabla.html)

- **Consigna:** Se requiere un archivo llamado listado_tabla.html para mostrar los productos en formato tabla.
- **Archivo Actual:** listado_tabla.html (Este archivo presenta un listado de propiedades en formato tabla, donde se incluyen imágenes, nombres, habitaciones, precios, estado y acciones).

Relación con la consigna: listado_tabla.html → Sección que liste todos los productos en forma de tabla

3. Sección que liste todos los productos en forma de box (listado_box.html)

- **Consigna:** Se requiere un archivo llamado **listado_box.html** para listar los productos en **formato box (cards)**.
- **Archivo Actual: anuncios.html** (Este archivo muestra las propiedades en formato **card** o **box**, con una visualización destacada de cada propiedad, su imagen, nombre y precio).

Relación con la consigna: anuncios.html → Sección que liste todos los productos en forma de box
Nota: Aunque el archivo se llama anuncios.html, cumple con la misma función que el archivo listado_box.html de la consigna.

4. Sección que muestre la ficha de un producto en particular (producto.html)

- **Consigna:** Se requiere un archivo llamado producto.html para mostrar la ficha de un producto en particular.
- **Archivo Actual:** propiedad.html (Este archivo muestra los detalles completos de una propiedad seleccionada, con una imagen destacada, características y un bloque de información con el precio y otras especificaciones).

Relación con la consigna: propiedad.html → Sección que muestre la ficha de un producto en particular

5. Sección para completar el formulario de compra (comprar.html)

- **Consigna:** Se requiere un archivo llamado comprar.html para incluir el formulario de compra.
- **Archivo Actual:** contacto.html (Este archivo contiene un formulario de compra donde el usuario puede ingresar los datos de contacto, dirección, medio de pago, etc. para concretar la compra de una propiedad).

Relación con la consigna: contacto.html → Sección para completar el formulario de compra

Conclusión

En resumen, aunque los nombres de archivo no coinciden exactamente con los solicitados en la consigna, cada archivo cumple con la misma función y las mismas responsabilidades dentro del proyecto. Se realizaron algunas modificaciones en los nombres para mejorar la legibilidad y la estructura del proyecto, asegurando que cada archivo esté correctamente organizado y refleje de manera precisa su propósito dentro del flujo de trabajo del proyecto.

Correspondencia de los archivos según la consigna:

- Portada principal: `index.html`
- Listado en tabla: `listado_tabla.html`
- Listado en boxes: `anuncios.html` → Renombrado a `listado_box.html` en la consigna.
- Ficha de producto: `propiedad.html` → Renombrado a `producto.html` en la consigna.
- Formulario de compra: `contacto.html` → Renombrado a `comprar.html` en la consigna.

Este informe describe cómo cada archivo HTML en el proyecto se relaciona con la consigna dada, haciendo las aclaraciones pertinentes sobre los nombres de archivo para garantizar que el propósito de cada uno sea claro.

BIBLIOGRAFÍA

Can i use:
https://caniuse.com/
Box sizing:
https://www.paulirish.com/2012/box-sizing-border-box-ftw/

ANEXOS

Github:

https://github.com/Hernanstupniki/TP1-Paradgimas-3