# Filtros de Búsqueda CameJobs

# Integrantes:

# Eduardo Enrique Carracedo Leuro y Breiner Cardona Ortiz

# Informe del proyecto de avance de proyecto CameJobs

# Ficha: 2999167

Contenido

[Filtros de Búsqueda CameJobs 3](#_Toc181646353)

[1. Introducción 3](#_Toc181646354)

[2. Propósito del Código 4](#_Toc181646355)

[3. Tecnologías Utilizadas 5](#_Toc181646356)

[HTML: 5](#_Toc181646357)

[CSS: 5](#_Toc181646358)

[JavaScript: 5](#_Toc181646359)

[4. Estructura del Código 5](#_Toc181646360)

[HTML: 5](#_Toc181646361)

[JavaScript: 5](#_Toc181646362)

[5. Funcionamiento del Filtrado 7](#_Toc181646363)

[Proceso de Filtrado: 7](#_Toc181646364)

[modalidadMatch: 7](#_Toc181646365)

[experienciaMatch: 7](#_Toc181646366)

[localizacionMatch: 7](#_Toc181646367)

[searchMatch: 7](#_Toc181646368)

[Visualización de Resultados: 8](#_Toc181646369)

[6. Conclusión 9](#_Toc181646370)

## 1. Introducción

El presente documento describe la implementación de un sistema de filtrado de usuarios utilizado en la plataforma CameJobs. Este sistema permite a los usuarios buscar y filtrar perfiles de empresas y trabajadores según diversos criterios, mejorando así la experiencia de búsqueda y facilitando la conexión entre empleadores y empleados potenciales.

## 2. Propósito del Código

El código tiene como finalidad proporcionar una interfaz de usuario que permita aplicar filtros en la búsqueda de perfiles laborales. Los usuarios pueden filtrar por diferentes categorías, tales como:

* Modalidad de trabajo (diurno/nocturno)
* Horario (diferentes duraciones de horas)
* Nivel de experiencia (sin experiencia, prácticas, intermedio, director, ejecutivo)
* Localización (diferentes regiones geográficas)
* Nombre del usuario (búsqueda textual)

Estos filtros ayudan a los usuarios a encontrar resultados más relevantes y específicos de acuerdo a sus necesidades.

## 3. Tecnologías Utilizadas

El sistema está desarrollado utilizando las siguientes tecnologías:

### HTML:

Para estructurar la página web y crear los elementos de la interfaz de usuario.

### CSS:

Para aplicar estilos visuales a los elementos HTML (aunque no se detalla en este informe).

### JavaScript:

Para implementar la lógica de filtrado, manejando la interacción del usuario y actualizando dinámicamente la lista de usuarios en función de los criterios seleccionados.

## 4. Estructura del Código

### HTML:

La estructura HTML incluye un encabezado con un logotipo y una barra de navegación, así como un contenedor principal que contiene los filtros y la lista de usuarios. Cada usuario está representado como un elemento de lista (<li>) que contiene atributos de datos para facilitar el filtrado.

### JavaScript:

La lógica de filtrado se gestiona principalmente en el archivo script.js, donde se define la función applyFilters(). Esta función se ejecuta cada vez que un usuario realiza un cambio en los filtros seleccionados.

En applyFilters(), se obtienen los valores de los filtros y se accede a la lista de usuarios. Se utiliza un bucle para iterar sobre cada elemento de la lista (<li>), extrayendo sus atributos de datos.

Se realizan verificaciones de coincidencia para cada filtro. Si un usuario cumple con todos los criterios, se muestra; de lo contrario, se oculta.

Además, el código incluye la función updateSalary(), que se activa al mover el control deslizante. Esta función actualiza en tiempo real el valor del salario mostrado en la interfaz.

La función setFilter() se utiliza para establecer valores específicos en los filtros y aplicar automáticamente la lógica de filtrado.

## 5. Funcionamiento del Filtrado

### Proceso de Filtrado:

Al seleccionar un valor en cualquiera de los filtros (modalidad, horario, experiencia, localización), se activa la función applyFilters(). Esta función obtiene los valores seleccionados de los elementos de entrada y el campo de texto para búsqueda.

Para cada usuario en la lista, se extraen los valores de sus atributos de datos. Se comprueba si los valores seleccionados coinciden con los datos del usuario:

### modalidadMatch:

Verifica si el valor de modalidad seleccionado es igual al valor del atributo data-modalidad del usuario. Si no hay modalidad seleccionada, se considera una coincidencia.

### experienciaMatch:

Realiza una verificación similar para el nivel de experiencia.

### localizacionMatch:

Comprueba si la localización seleccionada coincide con el atributo data-localizacion.

### searchMatch:

Verifica si el nombre del usuario incluye la cadena de búsqueda ingresada en el campo de texto, comparando en minúsculas para asegurar que la búsqueda sea insensible a mayúsculas.

### Visualización de Resultados:

Si un usuario cumple con todos los criterios de filtrado, su elemento de lista se muestra (se establece style.display en block). Si no coincide, se oculta (se establece style.display en none).

El control deslizante de salario permite a los usuarios ajustar el salario máximo que están dispuestos a considerar. Cada vez que el usuario mueve el deslizador, la función updateSalary() se activa, actualizando el texto que muestra el salario actual seleccionado.

La función setFilter() permite a los desarrolladores o al sistema establecer filtros automáticamente (por ejemplo, al cargar la página con filtros predefinidos) y aplicar la lógica de filtrado inmediatamente después de cada cambio.

## 6. Conclusión

El sistema de filtrado de usuarios implementado en CameJobs permite a los usuarios realizar búsquedas eficientes y precisas, adaptándose a las necesidades específicas de cada usuario. Al integrar HTML, CSS y JavaScript, se logra una experiencia de usuario interactiva y dinámica que mejora la funcionalidad de la plataforma.

Este código es una parte del sistema de búsqueda de CameJobs cuya función es esencial para conectar a empleadores y candidatos de manera mas eficiente y rápida.