

Laboratorio práctico: Desarrollar una aplicación para el censo de salud



Tiempo estimado necesario: 40 minutos

Lo que aprenderás

En esta interfaz de análisis de datos de salud, aprenderás a construir formularios interactivos utilizando HTML y emplear JavaScript para recopilar y gestionar datos de pacientes.

Este proyecto te ayudará a entender la manipulación del DOM y técnicas de búsqueda basadas en una condición de salud. Además, adquirirás habilidades en la generación dinámica de informes dentro de una página web, mostrando información estadística derivada de los datos. Este ejercicio práctico también enfatiza la aplicación de la toma de decisiones basada en datos en la analítica de salud.

Los conocimientos y habilidades adquiridos al trabajar en este proyecto de interfaz de análisis de datos de salud formarán una base sólida para tu proyecto final.

Objetivos de aprendizaje

Después de completar este laboratorio, podrás:

- **Interfaz de entrada de datos interactiva:** Desarrollar una comprensión del desarrollo web front-end creando una interfaz interactiva para recopilar datos de pacientes. Aprenderás a usar formularios HTML y elementos de entrada, validar la entrada del usuario y manejar varios tipos de entrada de datos (texto, botones de opción, entradas numéricas y menús desplegables).
- **Procesamiento y análisis de datos:** Aprender técnicas de gestión y búsqueda de datos utilizando JavaScript. Explorarás conceptos de manipulación de datos como almacenar datos en arreglos de objetos y filtrar datos según condiciones.
- **Generación dinámica de informes:** Generarás y mostrarás informes de manera dinámica dentro de una página web. Esta acción implica actualizar el contenido HTML basado en los datos procesados, presentando información estadística de manera organizada y comprensible, y manipulando el DOM para reflejar cambios al instante.
- **Interacción del usuario y manejo de eventos:** Practicar la programación impulsada por eventos y la interacción del usuario. Aprenderás a manejar eventos desencadenados por el usuario (clics en botones y enlaces de navegación) y responder con consultas de búsqueda apropiadas.

Requisitos previos

- Conocimientos básicos de HTML y GitHub.
- Comprensión básica de arreglos, métodos de arreglos, cadenas, objetos y funciones.
- Navegador web con consola (Chrome DevTools, Firefox Console, etc.).

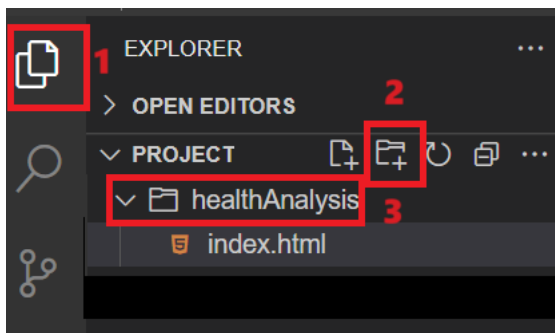
Configurando el entorno

1. Necesitas crear un repositorio en línea en GitHub y nombrarlo `heath_census`.

¡Importante! Asegúrate de que tu repositorio sea público.

2. Crea la estructura de archivos y carpetas según las instrucciones.

- En la ventana de la derecha, haz clic en **EXPLORER**, como se muestra en el número 1 de la captura de pantalla a continuación.
- Luego haz clic en la carpeta del proyecto y haz clic en el ícono de carpeta resaltado en rojo en el número 2 de la captura de pantalla.
- Ingresar el nombre de la carpeta como **healthAnalysis**. Esto creará una carpeta para ti.
- Luego selecciona la carpeta **healthAnalysis** mostrada en el número 3, haz clic derecho y selecciona **New File**.
- Ingresar el nombre del archivo **index.html** y haz clic en **OK**. Esto creará tu archivo HTML.



3. Crea el archivo JavaScript según las instrucciones:

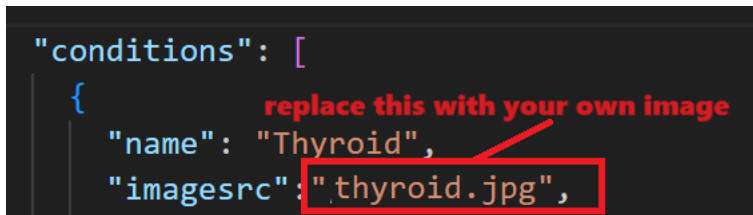
- Selecciona la carpeta **healthAnalysis** nuevamente, haz clic derecho y selecciona **New File**.
- Ingresar el nombre del archivo **health_analysis.js** y haz clic en **OK**. Esto creará tu archivo JavaScript.

4. Para este proyecto, también necesitas crear un archivo JSON. Sigue las instrucciones dadas:

- Nuevamente, haz clic derecho en la carpeta **healthAnalysis** y selecciona **New File**.
- Ingresar el nombre del archivo **health_analysis.json** y haz clic en **OK**. Esto creará tu archivo JSON.

5. Haz clic en este enlace [health_analysis.json](#) y copia los datos, luego pégalos en el archivo **health_analysis.json**, y luego guárdalo. Puedes recuperar estos datos JSON para encontrar detalles sobre síntomas, prevención y tratamiento relacionados con una condición de salud particular.

- Necesitas elegir una imagen en línea apropiada relacionada con el tema “Tiroides, diabetes y presión arterial”. Por favor, sube la imagen a la carpeta donde están almacenados tus archivos HTML y JS. Reemplaza “thyroid.jpg” con el nombre de la imagen subida como se muestra en la captura de pantalla a continuación:



- Por ejemplo, si has subido una imagen llamada “thyroid-my-img.png”, cambia la línea mostrada en la captura de pantalla anterior a:

```
"imagesrc": "thyroid-my-img.png",
```

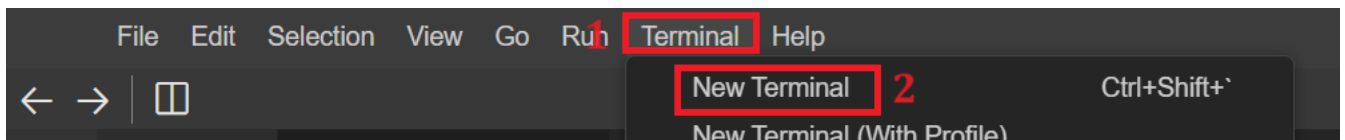
6. Crea el archivo HTML de contacto según las instrucciones:

- Selecciona la carpeta **healthAnalysis** nuevamente, haz clic derecho y selecciona **New File**.
- Ingresas el nombre del archivo **health_contact.html** y haz clic en **OK**. Esto creará tu archivo HTML de contacto.

Realizar comandos de Git

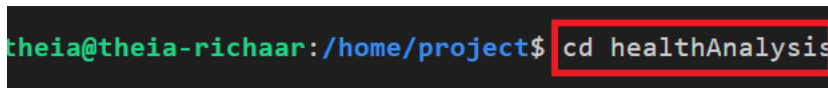
1. Ahora realiza comandos de git según las instrucciones dadas:

- Haz clic en la terminal mostrada en el número 1 y luego selecciona **Nueva Terminal** como se muestra en el número 2 en la captura de pantalla dada.



- Luego, necesitas entrar en la carpeta **healthAnalysis**. Para esto, escribe el comando dado en la terminal y presiona **Enter**.

```
cd healthAnalysis
```



- Luego inicializa esta carpeta como un repositorio git escribiendo el comando dado en la terminal.

```
git init
```

- Realiza **git add** para agregar los últimos archivos y carpetas escribiendo el comando dado en la terminal en el entorno git.

```
git add --a
```

- Luego realiza **git commit** en la terminal. Al realizar **git commit**, la terminal puede mostrar un mensaje para configurar tu **git config --global** para **user.name** y **user.email**. Si es así, entonces también necesitas ejecutar el comando **git config** para **user.name** y **user.email** como se indica.

```
git config --global user.email "you@example.com"
```

```
git config --global user.name "Your Name"
```

Nota: Reemplaza los datos entre comillas con tus propios detalles.

Luego realiza el comando de commit como se indica:

```
git commit -m "message"
```

- A continuación, realiza `git push` simplemente escribiendo el comando dado en la terminal uno tras otro.

```
git remote add origin2 <git-repo-url>
```

Nota: Reemplaza toda la `<git-repo-url>` con la URL de tu repositorio de GitHub, como `git remote add origin2 https://github.com/tuNombreDeCuenta/tuNombreDeRepositorio`

- Luego ejecuta el comando dado en la terminal para enviar el contenido de tu archivo al repositorio de GitHub y presiona **Enter**.

```
git push origin2
```

- Al enviar los archivos usando el comando `git push`, te pedirá que ingreses el nombre de usuario de tu cuenta de GitHub en la terminal. Ingresa tu nombre de usuario y luego presiona enter. Ahora, también te pedirá tu contraseña, necesitas pegar tu **Token de Acceso Personal** aquí que generaste en el primer laboratorio.

Nota: Al pegar esto en la terminal, no se mostrará por razones de seguridad, pero ya está allí. Simplemente presiona enter y se enviarán tus archivos y carpetas al repositorio de GitHub.

- Se enviarán todos los archivos directamente a tu repositorio de GitHub.

Nota: Después de pegar el código, guarda tu archivo.

Recordar

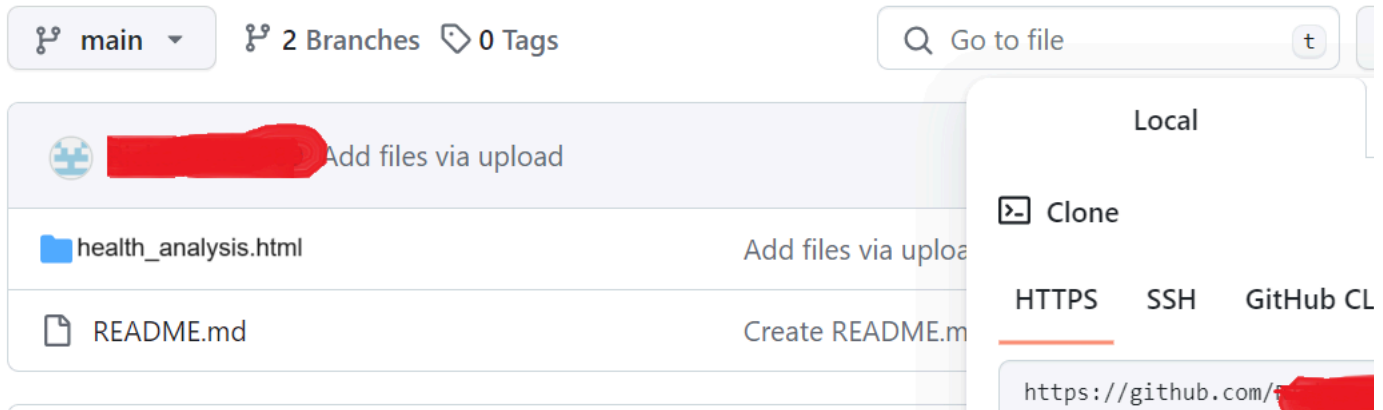
¡**IMPORTANTE!** Si cierras la sesión, necesitarás clonar el repositorio de GitHub de este proyecto después de abrir el **Entorno de Red de Habilidades**. Para ello, debes seguir los pasos:

- Abre una terminal dentro del **Entorno de Red de Habilidades** y escribe este comando:

```
git clone <git-repo-link>
```

Nota: Reemplaza **<git-repo-link>** con el enlace real de tu repositorio de GitHub.

- Puedes obtener el enlace de tu repositorio haciendo clic en **código**, como se muestra en la captura de pantalla a continuación en el número 1.
- Luego copia el enlace URL que se muestra en el número 2.



- Realiza los comandos `git add`, `git commit` y `git push` para subir todos tus archivos y carpetas más recientes en el repositorio de GitHub.

Nota:

- Después de clonar, solo necesitas usar el comando `git push origin` para subir todo tu trabajo más reciente.
- Si estás trabajando continuamente y aún no has realizado el `git clone`, entonces usa el comando `git push origin2` para subir.

Tarea 1: Creación de página web para *index.html*

En esta tarea, crearás una estructura básica de plantilla HTML en un archivo agregando el código a continuación.

- Esta estructura HTML configura una página web titulada **Datos de Análisis de Salud** con un encabezado y un elemento `<div>` vacío destinado a mostrar contenido relacionado con la salud de manera dinámica utilizando JavaScript.
- Incluye el código dado a continuación en tu archivo **index.html**. Las explicaciones de este código son las siguientes:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Health Analysis Data</title>
  <link rel="stylesheet" href="./health_analysis.css">
</head>
<body>
  <nav><h1>Health Analysis Census</h1>
  <ul>
    <li><a href="./index.html" id="home">Inicio </a></li>
    <li><a href="./health_contact.html" id="contact">Contáctanos </a></li>
    <li><input type="text" id="conditionInput" placeholder="Ingresa una condición de salud"> </li>
    <li><button id='btnSearch'>Buscar</button>
  </li>
</ul>
</nav>
<div class="container">
  <div class="analysisForm">
    <h1>Análisis de Datos de Salud</h1>
    <div>
      <label for="name">Nombre:</label>
      <input type="text" id="name">
    </div>
    <div>
      <label>Género:</label>
      <label for="male">Masculino</label>
      <input type="radio" name="gender" id="male" value="Male">
      <label for="female">Femenino</label>
      <input type="radio" name="gender" id="female" value="Female">
    </div>
    <div>
      <label for="age">Edad:</label>
      <input type="number" id="age">
    </div>
    <div>
      <label for="condition">Condición:</label>
      <select id="condition">
        <option value="">Seleccionar condición</option>
        <option value="Diabetes">Diabetes</option>
        <option value="Tiroides">Tiroides</option>
        <option value="Presión Arterial Alta">Presión Arterial Alta</option>
      </select>
    </div>
    <button id="addPatient">Agregar Paciente</button>
    <h2>Informe de Análisis</h2>
    <div id="report"></div>
  </div>
  <div class="searchCondition">
    <div id="result"></div>
  </div>
</div>
<script src="./health_analysis.js"></script>
</body>
</html>
```

- La etiqueta <nav> significa una sección de navegación dentro del documento HTML.
- La etiqueta <head> presenta el título “Datos de Análisis de Salud” dentro de la sección de navegación.
- Una lista desordenada contiene una lista de elementos :
 - uno para el enlace de inicio utilizando una etiqueta de anclaje
 - otro es un cuadro de entrada para una barra de búsqueda
 - un tercero es para un botón de búsqueda para buscar datos relacionados con una condición de salud particular
- El formulario de entrada constituye un formulario para ingresar datos relacionados con la salud. Los campos incluyen:
 - Un campo de entrada de texto para capturar el nombre del paciente
 - Botones de radio para seleccionar el género del paciente (masculino/femenino)
 - Un campo de entrada específicamente para ingresar la edad del paciente
 - Un menú desplegable <select> que permite la selección de la condición médica del paciente
 - Un botón etiquetado **Agregar Paciente** para enviar los datos para su procesamiento y análisis
- Una etiqueta <h2> presenta el encabezado “Informe de Análisis.”
- El ID report identifica un elemento <div> vacío. Esta acción mostrará dinámicamente los informes de análisis generados en función de los datos ingresados por el usuario.
- Un elemento <div> vacío más identificado por el ID result. Esta acción mostrará dinámicamente los detalles generados para una condición de salud particular para mostrar sus síntomas y métodos de prevención.
- Se te ha dado una línea de código disponible en el número de línea 12. El hipervínculo tiene un atributo href establecido en “./health_contact.html”, lo que indica que al hacer clic, redirigirá al usuario a la página “health_contact.html” ubicada en el directorio actual.



```
<li><a href="./health_contact.html" id="contact">Contáctanos </a></li>
```
- Aplica CSS de acuerdo con tu diseño y tema de color. Para esto, crea un archivo CSS y nómbralo **health_analysis.css** seleccionando la carpeta **healthAnalysis** y haciendo clic derecho para seleccionar **Nuevo Archivo**.

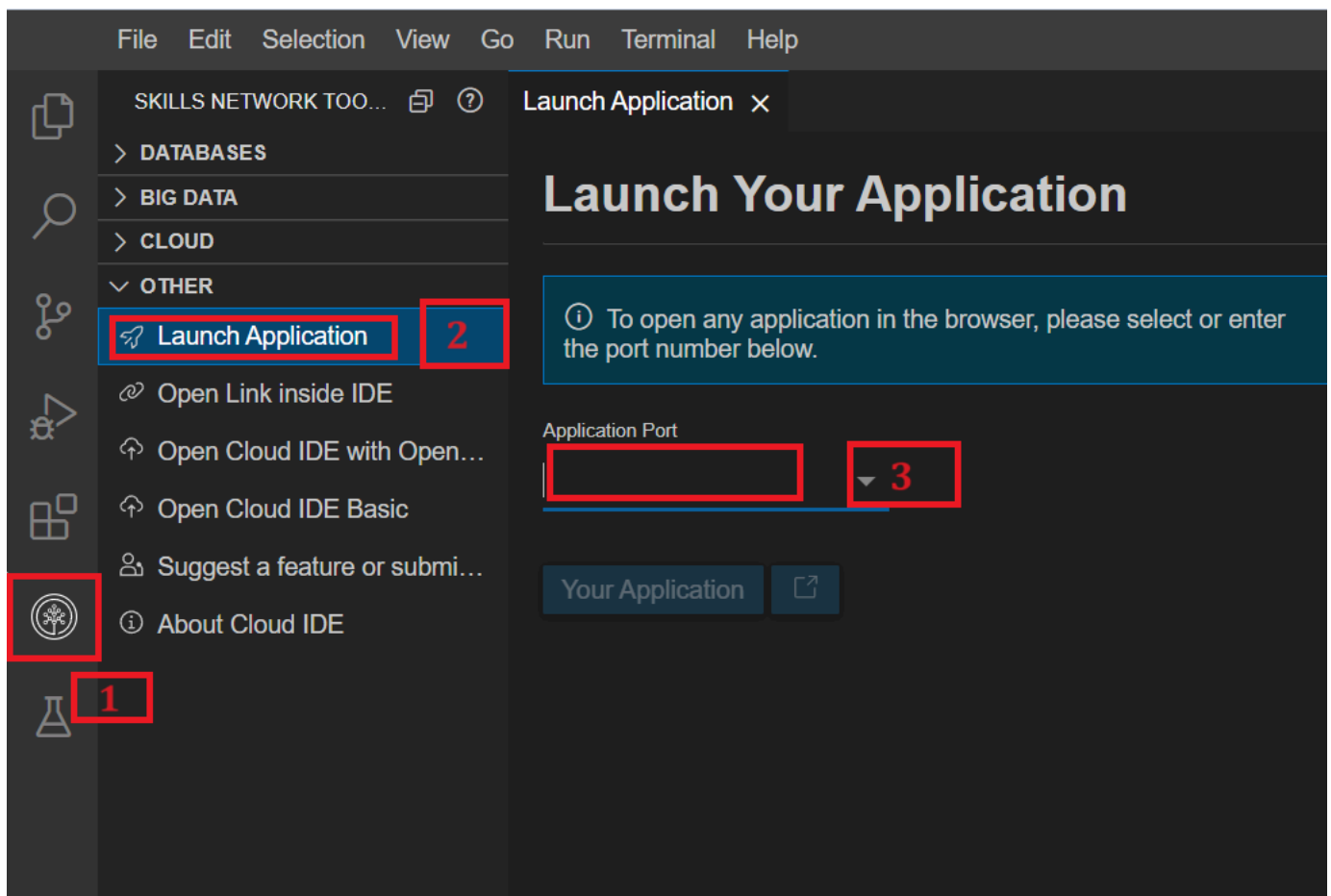
También puedes usar el CSS proporcionado en este enlace [health_analysis.css](#)

Tarea 2: Verifica la salida

1. Para ver tu página HTML en un navegador, utiliza la extensión integrada Live Server. Selecciona el archivo **index.html** dentro de la carpeta del proyecto y haz clic derecho en ese archivo. Elige ‘Abrir con Live Server’.
2. Aparecerá una notificación en la esquina inferior derecha, indicando que el servidor se inició en el puerto 5500.



3. Haz clic en el ícono de **Skills Network** en el lado izquierdo de la pantalla que se muestra en el número 1 de la captura de pantalla a continuación. Se abrirá la **Caja de Herramientas de Skills Network**.
- Luego haz clic en **Lanzar Aplicación** que se muestra en el número 2.
- Ingresa el número de puerto como 5500 en el número 3.
- Haz clic en este botón .



4. Se abrirá tu navegador predeterminado y verás una carpeta llamada **healthAnalysis**. Haz clic en el nombre de esa carpeta y se abrirá automáticamente tu página principal.

Nota: La razón por la que la página principal se abrió automáticamente es porque el nombre de tu archivo HTML principal es **index.html**, que es el nombre de archivo predeterminado que se selecciona automáticamente.

5. Se abrirá la página principal y verás la salida mostrada en la captura de pantalla a continuación.

Health Analysis Census

Healthcare Data Analysis

Name:

Gender: ☒ Male ☐ Female ☐

Age:

Condition:

Analysis Report

6. Luego ejecuta los comandos `git init`, `git add`, `git commit` y `git push` para subir tu código más reciente a tu repositorio de GitHub.

Nota: Asegúrate de guardar tus archivos y ejecutar los comandos de git, para que tu trabajo esté actualizado en tu repositorio de GitHub.

Tarea 3: Definiendo variables

- Ahora, inicializaremos las variables. Incluye el código dado en el archivo **health_analysis.js**.

```
const addPatientButton = document.getElementById("addPatient");
const report = document.getElementById("report");
const btnSearch = document.getElementById('btnSearch');
const patients = [];
```

- **addPatientButton**: El botón utilizado para agregar datos de pacientes
- **report**: El elemento HTML donde verás los informes de análisis mostrados
- **btnSearch**: El nombre de la variable del botón que muestra los resultados de búsqueda al hacer clic
- También se crea un array vacío llamado **patients** para almacenar los datos de los pacientes recolectados.

Tarea 4: Creando la función que añade los detalles del paciente

Incluye el código dado a continuación, que tiene la función `addPatient()`. Esta función captura los datos ingresados por el usuario desde los elementos del formulario HTML: nombre, género, edad y condición médica. Asegura que todos los campos tengan entradas válidas.

```
function addPatient() {
  const name = document.getElementById("name").value;
  const gender = document.querySelector('input[name="gender"]:checked');
  const age = document.getElementById("age").value;
  const condition = document.getElementById("condition").value;
  if (name && gender && age && condition) {
    patients.push({ name, gender: gender.value, age, condition });
    resetForm();
    generateReport();
  }
}
```

Esta función recupera los detalles del paciente en el formulario, tales como nombre, género, edad y condición. Por ejemplo, la variable `name` se define como `const name = document.getElementById("name").value;`

Además,

- agrega los detalles del paciente al array `patients[]`, que almacena todos los datos de pacientes ingresados utilizando el método `push()`
- restablece los campos del formulario utilizando el método `resetForm()` para limpiar los campos de entrada para la próxima entrada
- activa el método `generateReport()` para actualizar y mostrar el informe de análisis basado en los datos del paciente recién agregado

Tarea 5: Crear una función para restablecer los valores del formulario

Crea una función llamada `resetForm()`. Esta función borra los valores de los campos de nombre, género, edad y condición en el formulario HTML al establecerlos como cadenas vacías o desmarcados para los botones de opción, restableciendo efectivamente el formulario a su estado inicial. Así, está listo para la entrada de nuevos datos.

Incluye este código después de la función `addPatient()`.

```
function resetForm() {
  document.getElementById("name").value = "";
  document.querySelector('input[name="gender"]:checked').checked = false;
  document.getElementById("age").value = "";
  document.getElementById("condition").value = "";
}
```

El código anterior asigna un valor vacío a todos los campos para limpiar los detalles ingresados anteriormente.

Tarea 6: Crea la función que genera el informe

Si anteriormente cerraste la sesión después de ingresar al **Entorno de Red de Habilidades**, debes clonar tu repositorio de GitHub para este proyecto. De lo contrario, continúa con el entorno en el que ya estás trabajando.

- Crea una función llamada `generateReport()` para generar informes. Incluye el código después de la función de formulario `resetForm()`.

```
function generateReport() {
  const numPatients = patients.length;
  const conditionsCount = {
    Diabetes: 0,
    Thyroid: 0,
    "High Blood Pressure": 0,
  };
  const genderConditionsCount = {
    Male: {
      Diabetes: 0,
      Thyroid: 0,
      "High Blood Pressure": 0,
    },
    Female: {
      Diabetes: 0,
      Thyroid: 0,
      "High Blood Pressure": 0,
    },
  };
  for (const patient of patients) {
    conditionsCount[patient.condition]++;
    genderConditionsCount[patient.gender][patient.condition]++;
  }
  report.innerHTML = `Number of patients: ${numPatients}<br><br>`;
  report.innerHTML += `Conditions Breakdown:<br>`;
  for (const condition in conditionsCount) {
    report.innerHTML += `${condition}: ${conditionsCount[condition]}<br>`;
  }
  report.innerHTML += `<br>Gender-Based Conditions:<br>`;
  for (const gender in genderConditionsCount) {
    report.innerHTML += `${gender}<br>`;
    for (const condition in genderConditionsCount[gender]) {
      report.innerHTML += `&nbsp;&nbsp; ${condition}: ${genderConditionsCount[gender][condition]}<br>`;
    }
  }
}
addPatientButton.addEventListener("click", addPatient);
```

- Esta función `generateReport()` calcula y construye un informe de análisis basado en los datos de pacientes recopilados almacenados en el arreglo `patients[]`. Aquí hay un desglose:
 - Inicialización:
 - `numPatients` Representa el total de pacientes almacenados en el arreglo `patients[]`
 - `conditionsCount` Una estructura de datos (objeto) que inicializa contadores para condiciones médicas específicas (Diabetes, Tiroides, Presión Arterial Alta), inicialmente configurados en cero.
 - `genderConditionsCount` Un objeto anidado con contadores de condiciones específicas por género (masculino y femenino) para cada condición médica, también inicializados en cero para cada condición.
 - Bucle de procesamiento de datos:
 - Itera a través del arreglo `patients[]`: Utiliza un bucle `for...of` para iterar a través de los datos de cada paciente dentro del arreglo `patients[]`.
 - Incrementar conteos de condiciones: Incrementa el conteo para la condición médica específica de cada paciente en el objeto `conditionsCount`.
 - Actualización de conteos de condiciones por género: Aumenta el conteo de cada condición médica dentro de la categoría de género respectiva en el objeto `genderConditionsCount` basado en el género y la condición del paciente.
 - Actualización de HTML:
 - Actualizar elemento del informe: Actualiza dinámicamente el contenido HTML dentro del elemento `report` designado.
 - Mostrar total de pacientes: Muestra el número total de pacientes.
 - Desglose de condiciones: Enumera los conteos para cada condición médica en el objeto `conditionsCount`.
 - Mostrar condiciones por género: Ilustra los conteos de cada condición categorizados por género en el objeto `genderConditionsCount`, mostrando la distribución de condiciones entre hombres y mujeres por separado.
 - Listener de eventos:
 - Ahora, necesitas configurar el listener de eventos usando `addPatientButton.addEventListener("click", addPatient)` para agregar detalles del paciente cuando el usuario haga clic en el botón **Agregar Paciente**.

Ve a tu navegador donde se ejecuta tu código, ingresa los detalles y haz clic en el botón **Agregar Paciente**. Debería generar datos, como se muestra en la captura de pantalla a continuación.

Analysis Report

Number of patients: 1

Conditions Breakdown:

Diabetes: 0

Thyroid: 1

High Blood Pressure: 0

Gender-Based Conditions:

Male:

Diabetes: 0

Thyroid: 1

High Blood Pressure: 0

Female:

Diabetes: 0

Thyroid: 0

High Blood Pressure: 0

También verás la funcionalidad de la función **ResetForm** que restablece todo el formulario después de que el usuario hace clic en el botón **Agregar Paciente**.

Tarea 7: Crear una función para la solicitud de búsqueda

Esta función de JavaScript `searchCondition()` está diseñada para funcionar dentro de una página web para recuperar información sobre condiciones de salud basada en la entrada del usuario. Incluye el código a continuación después de la función `resetForm()` de la tarea anterior en tu archivo de JavaScript.

```
function searchCondition() {
  const input = document.getElementById('conditionInput').value.toLowerCase();
  const resultDiv = document.getElementById('result');
  resultDiv.innerHTML = '';
  fetch('health_analysis.json')
    .then(response => response.json())
    .then(data => {
      const condition = data.conditions.find(item => item.name.toLowerCase() === input);
      if (condition) {
        const symptoms = condition.symptoms.join(', ');
        const prevention = condition.prevention.join(', ');
        const treatment = condition.treatment;
        resultDiv.innerHTML += `<h2>${condition.name}</h2>`;
        resultDiv.innerHTML += ``;
        resultDiv.innerHTML += `<p><strong>Symptoms:</strong> ${symptoms}</p>`;
        resultDiv.innerHTML += `<p><strong>Prevention:</strong> ${prevention}</p>`;
        resultDiv.innerHTML += `<p><strong>Treatment:</strong> ${treatment}</p>`;
      } else {
        resultDiv.innerHTML = 'Condition not found.';
      }
    })
    .catch(error => {
      console.error('Error:', error);
      resultDiv.innerHTML = 'An error occurred while fetching data.';
    });
}
btnSearch.addEventListener('click', searchCondition);
```

Esta función obtiene los datos de la condición de salud del archivo **health.json** y busca una condición que coincida con la entrada del usuario. Luego, muestra los detalles de la condición o un mensaje de error en un elemento HTML designado (`resultDiv`).

El código anterior incluye:

- `const input = document.getElementById('conditionInput').value.toLowerCase();` Esto recupera el valor ingresado en el campo de entrada con el ID `conditionInput`. Convierte el texto ingresado a minúsculas para asegurar una comparación que no distinga entre mayúsculas y minúsculas.
- `const resultDiv = document.getElementById('result');` `resultDiv.innerHTML = '';` Esto recupera el elemento HTML con el ID `'result'`. Limpia cualquier contenido previo dentro de este elemento HTML.
- `fetch('health.json')` Este método de API inicia una solicitud de recuperación al archivo llamado `'health.json'`. Se asume que hay un archivo JSON llamado `'health.json'` en el mismo directorio que el archivo HTML.
- `.then(response => response.json())` Convierte la respuesta recuperada en formato JSON.

- `.then(data => { /* ... */ })` Esto maneja los datos JSON recuperados. Busca una condición de salud que coincida con la entrada del usuario.
- `const condition = data.conditions.find(item => item.name.toLowerCase() === input);` Esto busca dentro de los datos JSON una condición de salud cuyo nombre coincida con la entrada ingresada.
- `if (condition) { /* ... */ } else { /* ... */ }` Este código verifica si hay una condición coincidente. Si se encuentra, construye contenido HTML para mostrar detalles sobre la condición (nombre, síntomas, prevención, tratamiento) dentro del `resultDiv`. Si el sistema no puede encontrar una condición coincidente, muestra un mensaje de 'Condición no encontrada' dentro del `resultDiv`.
- `.catch(error => { /* ... */ })` Esto maneja cualquier error que pueda ocurrir durante la solicitud de recuperación o el procesamiento de datos. Si ocurre un error, lo registra en la consola y muestra un mensaje de error dentro del `resultDiv`.
- Supongamos que has ingresado **Tiroides** en la barra de búsqueda. Después de hacer clic en el botón de búsqueda, mostrará la información proporcionada de **health_analysis.json**.

Thyroid



Symptoms: Fatigue, Weight gain or loss, Dry skin, Muscle weakness, Irregular menstrual periods

Prevention: Eat a balanced diet, Exercise regularly, Get regular check-ups

Treatment: Medication like levothyroxine may be prescribed by a doctor.

Tarea 8: Creación de página web para *health_contact.html*

- Ahora necesitas agregar contenido en **health_contact.html**. Copia el código proporcionado en el archivo de contacto html.

```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <title>Contáctenos</title>
  <link rel="stylesheet" href="./health_analysis.css">
  <style>
    body{
      text-align: center;
    }
    /* Algunos estilos básicos para fines de demostración */
    #contactForm {
      max-width: 400px;
      margin: 0 auto;
    }
    #contactForm input,
    #contactForm textarea {
      width: 100%;
      margin-bottom: 15px;
      padding: 8px;
    }
    #contactForm button {
      padding: 10px 20px;
      background-color: #007bff;
      color: white;
      border: none;
      cursor: pointer;
    }
    #contactForm button:hover {
      background-color: #0056b3;
    }
  </style>
</head>
<body>
  <nav><h1>Censo de Análisis de Salud</h1>
  <ul>
    <li><a href="./index.html" id="home">Inicio </a></li>
    <li><a href="./health_contact.html" id="contact">Contáctenos </a></li>
  </ul>
</nav>
  <h1>Contáctenos</h1>
  <p>No dude en contactarnos para saber más sobre sus condiciones y métodos de tratamiento.</p>
  <form id="contactForm">
    <div>
```

```

        <label for="name">Nombre</label>
        <input type="text" id="name" name="name" required>
    </div>
    <div>
        <label for="email">Correo electrónico</label>
        <input type="email" id="email" name="email" required>
    </div>
    <div>
        <label for="condition">Condición</label>
        <input type="text" id="condition" name="condition" required>
    </div>
    <div>
        <label for="message">Mensaje</label>
        <textarea id="message" name="message" rows="5" required></textarea>
    </div>
    <button type="submit" onclick="thankyou()">Enviar</button>
</form>
<script>
    function thankyou(){
        alert('¡Gracias por contactarnos!')
    }
</script>
</body>
</html>

```

- Verifica la salida del código anterior que se mostrará como a continuación:

Contact Us

Feel free to contact us to know more about your conditions and for treatment methods.

Name

Email

Condition

Message

Submit

- Completa el formulario y haz clic en el botón **Enviar**. Aparecerá un cuadro emergente mostrando el mensaje ¡**Gracias por contactarnos!**.

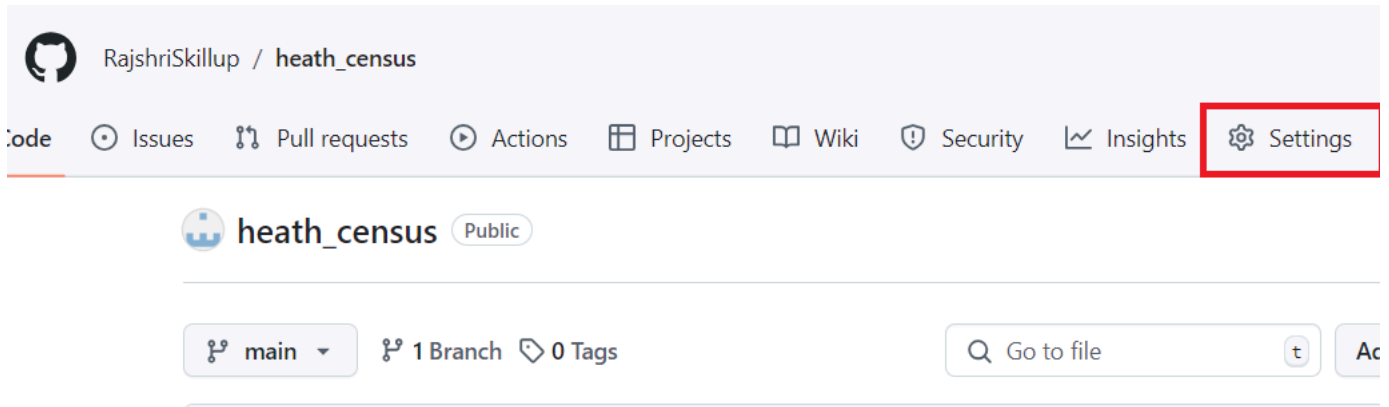
127.0.0.1:5500 says

Thank you for contacting us!

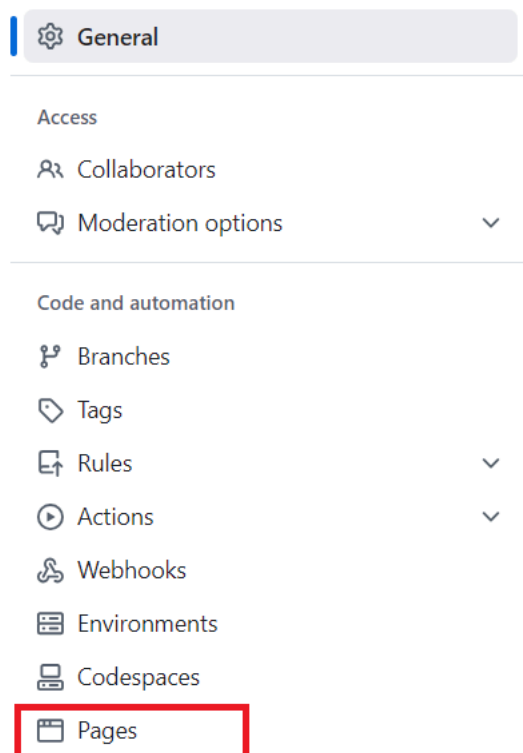
OK

Tarea 9: Realizar comandos git y generar páginas

1. Necesitas guardar todos tus archivos y realizar comandos git para ellos.
2. Realiza los comandos git `add`, git `commit` y git `push` para actualizar los cambios dentro de la carpeta **healthAnalysis**. Usa tu repositorio de GitHub para una adecuada gestión del código.
3. Ve a tu repositorio de GitHub. Luego, navega al repositorio de tu sitio que creaste al inicio de este proyecto.
4. Bajo el nombre de tu repositorio, haz clic en **Configuración**. Si no puedes ver la pestaña **Configuración**, selecciona el menú desplegable y luego haz clic en **Configuración**.



5. Navega en la barra de navegación del lado izquierdo. En la sección **Código y Automatización** de la barra lateral, haz clic en **Páginas**.



6. Verás la página mostrada a continuación. Haz clic en el menú desplegable donde ves **Ninguno**, luego haz clic en **main**, y luego haz clic en el botón **Guardar**.

GitHub Pages

GitHub Pages is designed to host your personal, organization, or project pages from a GitHub repository.

Build and deployment

Source

Deploy from a branch ▾

Branch

GitHub Pages is currently disabled. Select a source below to enable GitHub Pages for this repository. [Learn more about configuring the publishing source for your site.](#)

None ▾

Save

Select branch

Q |

main

✓ None

access to your GitHub Pages site by publishing it privately. You can use documentation or knowledge base with members of your enterprise. You can [learn more about the visibility of your GitHub Pages site.](#)

7. Actualiza tu página nuevamente, y verás el enlace, resaltado a continuación.

GitHub Pages

GitHub Pages is designed to host your personal, organization, or project pages from a GitHub repository.

Your site is live at https://rajshriskillup.github.io/heath_census/

Last **deployed** by  RajshriSkillup 1 minute ago

Visit site

...

Nota: Si no puedes ver el enlace, por favor espera de (1-2) minutos y actualiza la página nuevamente.

8. Haz clic en el enlace generado arriba para ver tu sitio web en vivo. Dependiendo del tamaño de las imágenes que hayas utilizado, puede que tarden un poco en abrirse.

Nota: Puedes referirte al primer laboratorio de este curso para revisar todos los comandos git.

¡Felicidades! Has terminado el laboratorio práctico.

Resumen

- Estructura HTML:** El archivo HTML establece una estructura que comprende enlaces de navegación para opciones de análisis y elementos de formulario para ingresar datos del paciente, incluyendo nombre, género, edad y condición médica.
- Lógica de JavaScript:** El código JavaScript incrustado gestiona el almacenamiento de datos del paciente, proporcionando funciones para agregar pacientes, restablecer las entradas del formulario y generar un informe de análisis basado en la información ingresada.
- Capacidades de Análisis de Datos:** El código incluye funcionalidades para filtrar pacientes según condiciones específicas y grupos de edad activados por los enlaces de navegación. Estas funciones solicitan la entrada del usuario a través de mensajes emergentes y muestran los resultados del análisis correspondiente dentro de la página web.
- Manejo de Eventos:** Se configuran oyentes de eventos para responder a las interacciones del usuario con los enlaces de navegación y el botón “Agregar Paciente”, facilitando actualizaciones dinámicas y generación de análisis basados en los datos de salud proporcionados.

© IBM Corporation. Todos los derechos reservados.