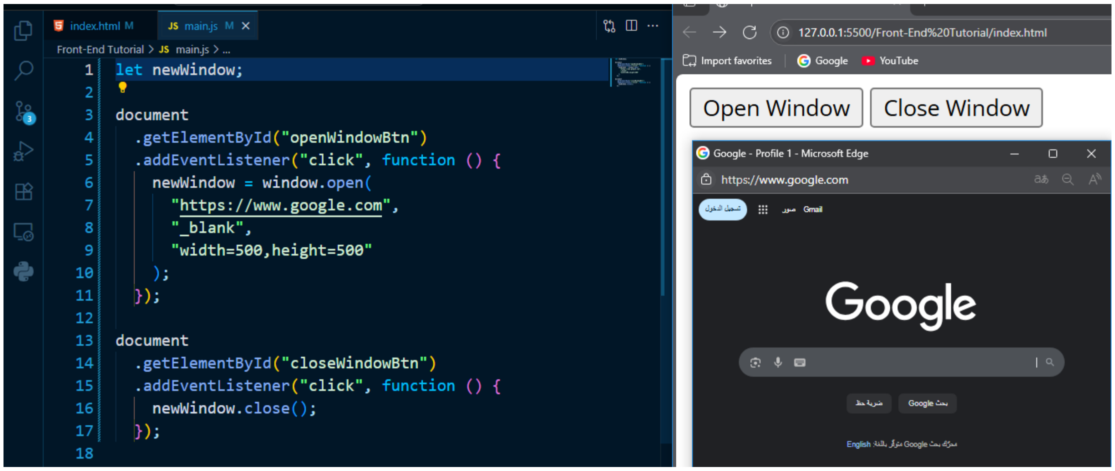
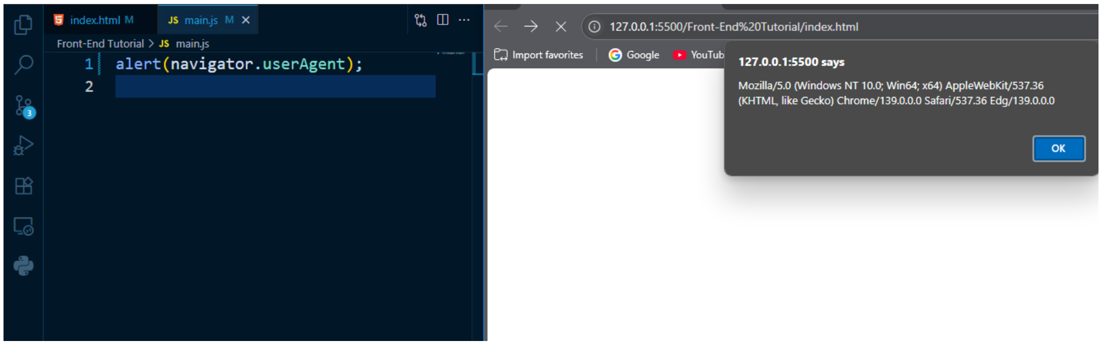
BOM (Browser Object Model)

1. Open a new window with a specific URL and size, then close it after 3 seconds using JavaScript.


index.html M
main.js M
Front-End Tutorial > JS main.js > ...
127.0.0.1:5500/Front-End%20Tutorial/index.html
Import favorites
Google
YouTube
Open Window
Close Window
Google - Profile 1 - Microsoft Edge
https://www.google.com
Gmail
Google

```
let newWindow;

document
  .getElementById("openWindowBtn")
  .addEventListener("click", function () {
    newWindow = window.open(
      "https://www.google.com",
      "_blank",
      "width=500,height=500"
    );
  });

document
  .getElementById("closeWindowBtn")
  .addEventListener("click", function () {
    newWindow.close();
  });

```

2. Display the browser's user agent string in an alert.

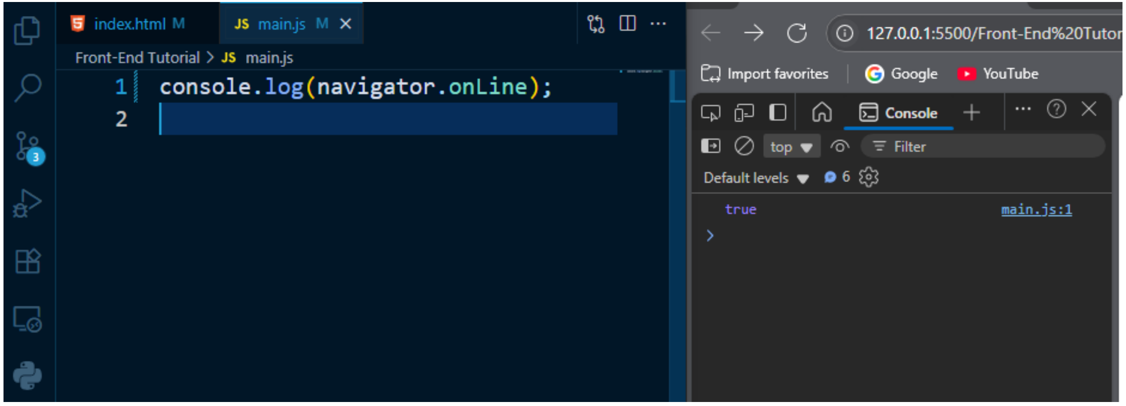

index.html M
main.js M
Front-End Tutorial > JS main.js
127.0.0.1:5500/Front-End%20Tutorial/index.html
Import favorites
Google
YouTub
OK

```
alert(navigator.userAgent);

```

127.0.0.1:5500 says

Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/139.0.0.0 Safari/537.36 Edg/139.0.0.0

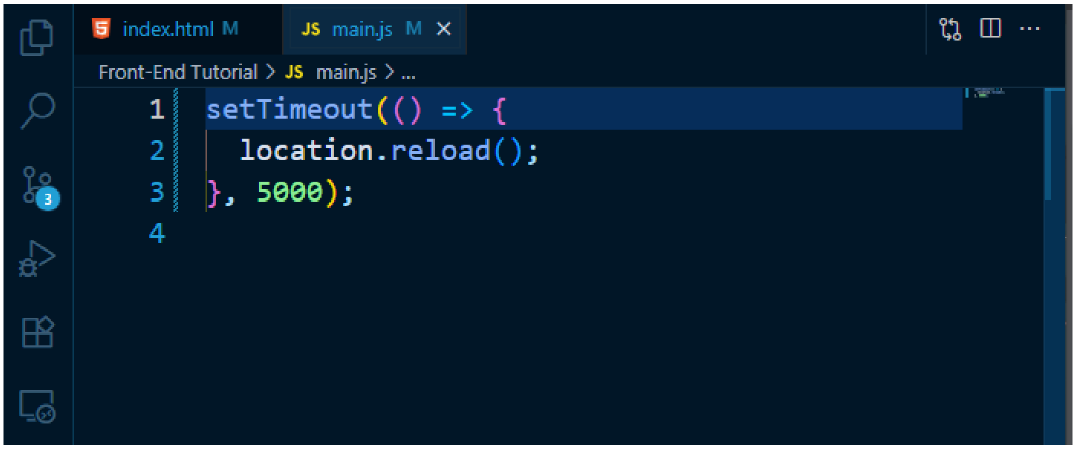
3. Use `navigator` to detect if the browser is online or offline and log the result.



```
console.log(navigator.onLine);

```

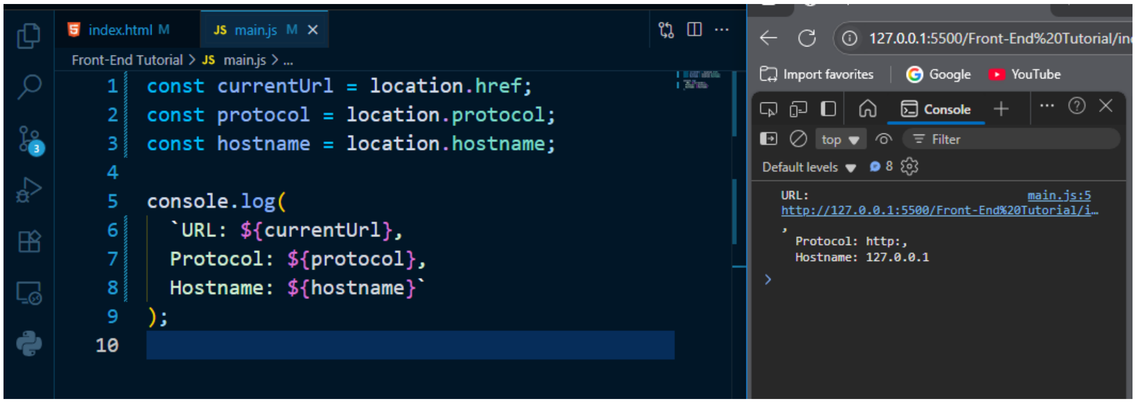
```
true                                    main.js:1
>
```

4. Write code to reload the current page after 5 seconds.



```
setTimeout(() => {
  location.reload();
}, 5000);
```

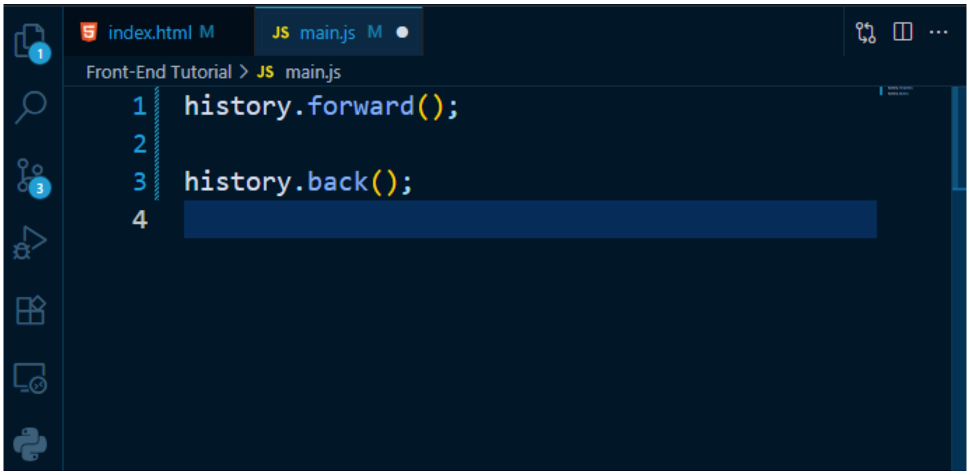
5. Get the current page URL, protocol, and hostname using `location` and log them.



```
const currentUrl = location.href;
const protocol = location.protocol;
const hostname = location.hostname;

console.log(
  `URL: ${currentUrl},
  Protocol: ${protocol},
  Hostname: ${hostname}`
);
```

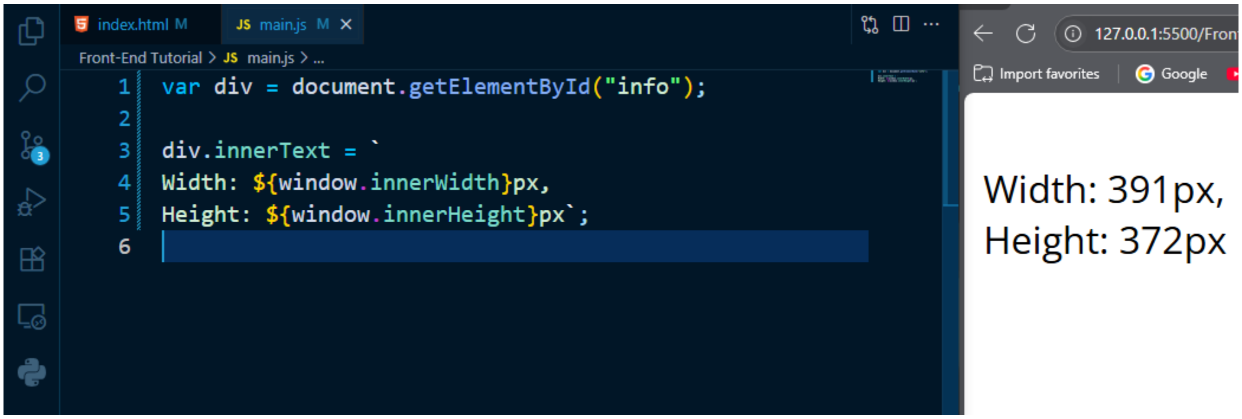
```
URL:                                         main.js:5
http://127.0.0.1:5500/Front-End%20Tutorial/i…
,
  Protocol: http:,
  Hostname: 127.0.0.1
>
```

6. Use `history` to go back one page, then forward one page (write the code, don't actually run it).



```
history.forward();

history.back();
```

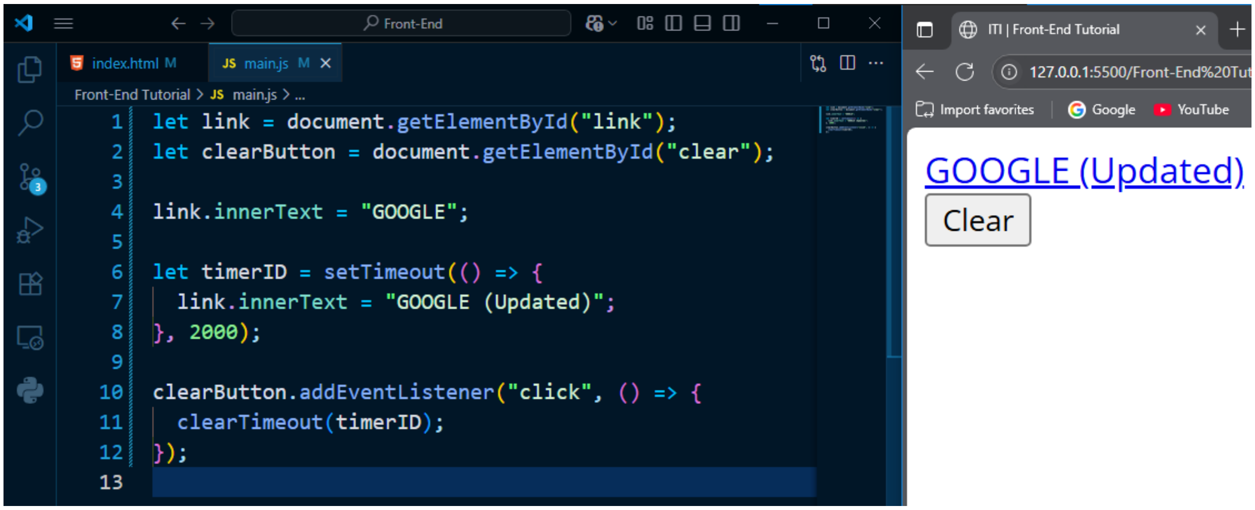
7. Show the screen width and height in a div -change the div content- on the page.



```javascript
var div = document.getElementById("info");

div.innerText = `
Width: ${window.innerWidth}px,
Height: ${window.innerHeight}px`;
```

```
Width: 391px,
Height: 372px
```

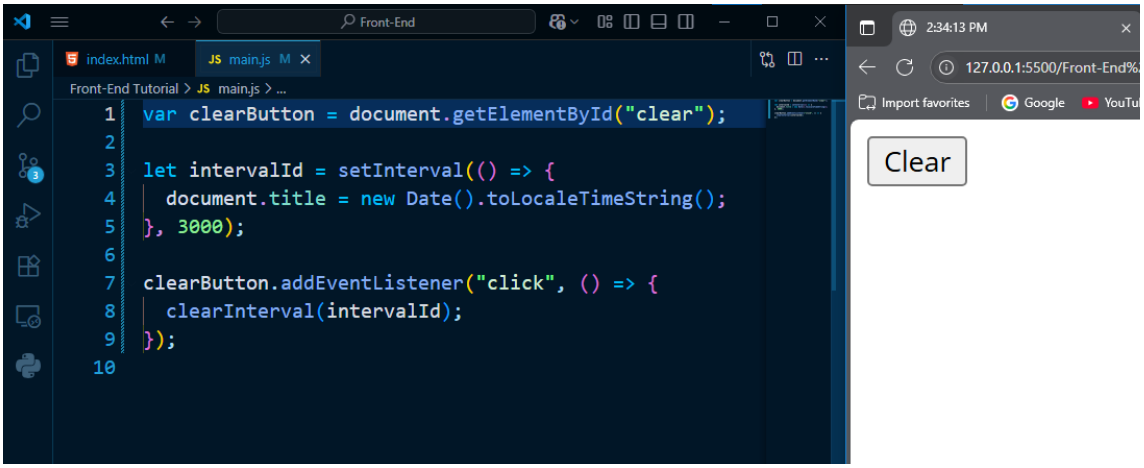
8. Use `setTimeout` to change the content of a hyperlink -a- with a new content after 2 seconds, and provide a button to cancel it.



```javascript
let link = document.getElementById("link");
let clearButton = document.getElementById("clear");

link.innerText = "GOOGLE";

let timerID = setTimeout(() => {
  link.innerText = "GOOGLE (Updated)";
}, 2000);

clearButton.addEventListener("click", () => {
  clearTimeout(timerID);
});
```

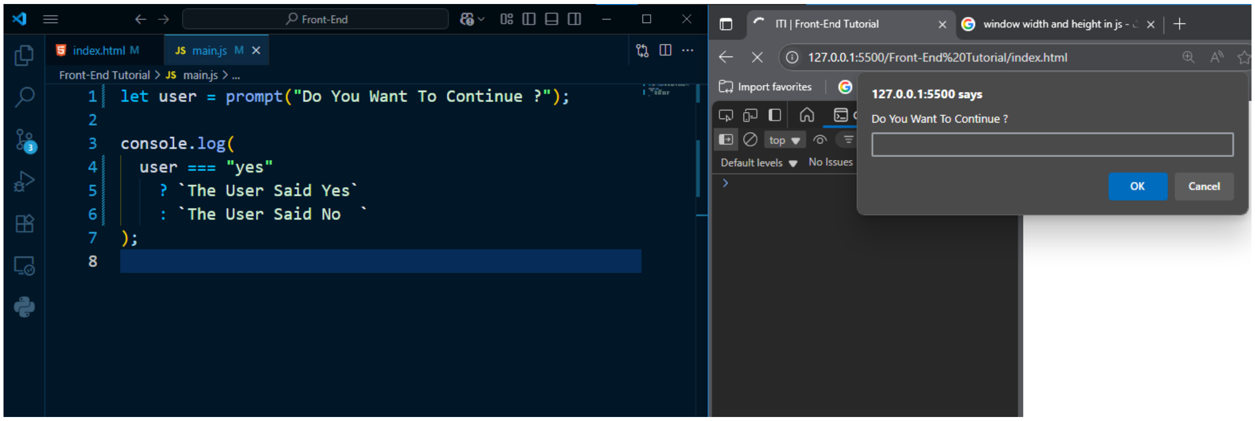
```
GOOGLE (Updated)
```

9. Use `setInterval` to update the time on the page title every second, and a button to stop it.



```
var clearButton = document.getElementById("clear");

let intervalId = setInterval(() => {
  document.title = new Date().toLocaleTimeString();
}, 3000);

clearButton.addEventListener("click", () => {
  clearInterval(intervalId);
});
```

10. Show a confirm dialog asking "Do you want to continue?" and log the user's choice("user said yes", "user said no").



```
let user = prompt("Do You Want To Continue ?");

console.log(
  user === "yes"
    ? `The User Said Yes`
    : `The User Said No  `
);
```

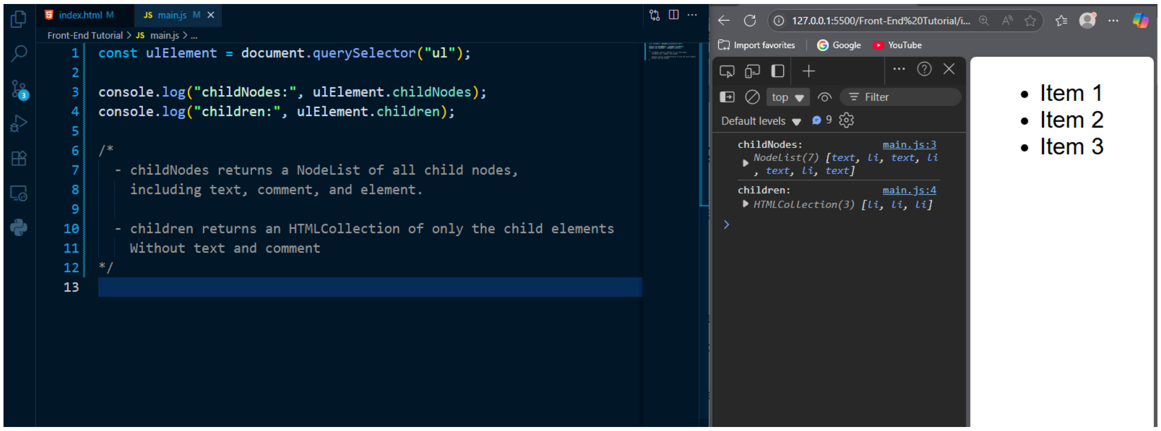
127.0.0.1:5500 says

Do You Want To Continue ?

Front-End
index.html M
main.js M
Front-End Tutorial > main.js > ...
ITI | Front-End Tutorial
window width and height in js -
127.0.0.1:5500/Front-End%20Tutorial/index.html
Import favorites
top
Default levels
No Issues
OK
Cancel

DOM Traversal (Nodes, Elements, Collections)

11. Select the first `<ul>` in the document and log its `childNodes` and `children` properties. Explain the difference in a comment.



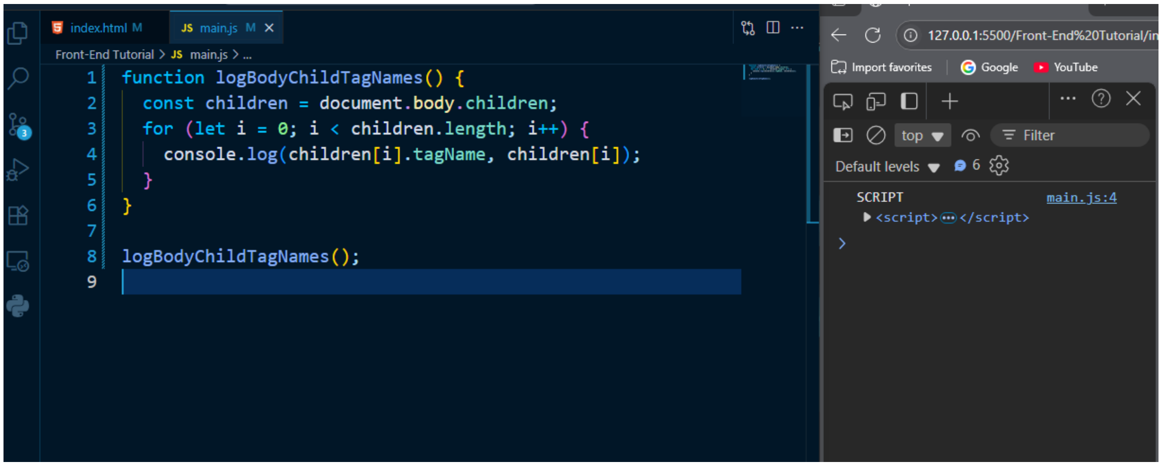
index.html M
main.js M
Front-End Tutorial > main.js > ...
127.0.0.1:5500/Front-End%20Tutorial/i...
Import favorites
Google
YouTube
top
Filter
Default levels
9

```
const ulElement = document.querySelector("ul");

console.log("childNodes:", ulElement.childNodes);
console.log("children:", ulElement.children);

/*
  - childNodes returns a NodeList of all child nodes,
    including text, comment, and element.

  - children returns an HTMLCollection of only the child elements
    Without text and comment
*/
```

```
childNodes:                                      main.js:3
▶ NodeList(7) [text, li, text, li
  , text, li, text]
children:                                        main.js:4
▶ HTMLCollection(3) [li, li, li]
```

- Item 1
- Item 2
- Item 3

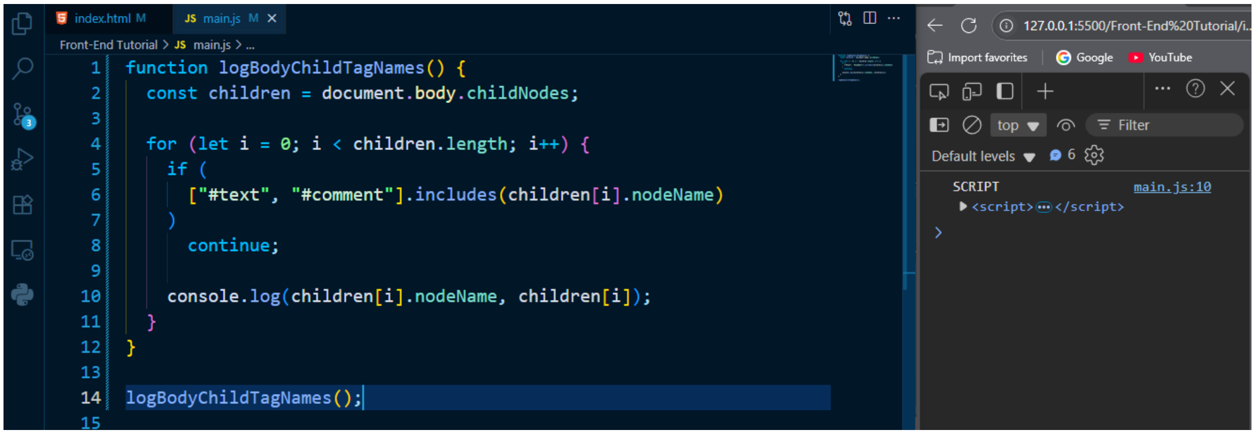
12. Write a function that logs the tag names of all direct child elements of `<body>`.



```
function logBodyChildTagNames() {
  const children = document.body.children;
  for (let i = 0; i < children.length; i++) {
    console.log(children[i].tagName, children[i]);
  }
}

logBodyChildTagNames();
```

```
SCRIPT                                main.js:4
▶ <script>⋯</script>
```

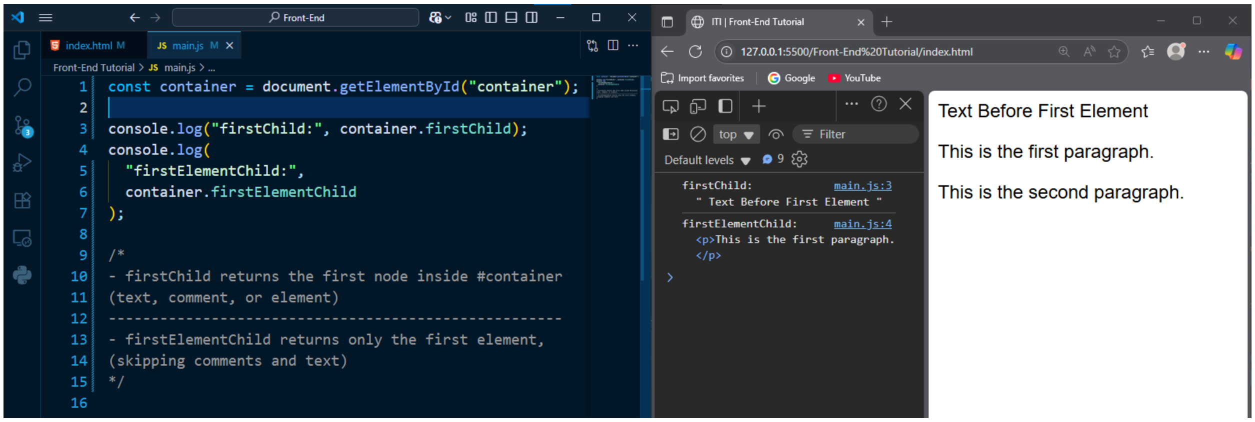
13. Given a parent element, loop through its `childNodes` and log only the nodes that are elements (not text/comments).


index.html M
main.js M
Front-End Tutorial > main.js > ...
127.0.0.1:5500/Front-End%20Tutorial/i...
Import favorites
Google
YouTube
top
Filter
Default levels
6

```javascript
function logBodyChildTagNames() {
  const children = document.body.childNodes;

  for (let i = 0; i < children.length; i++) {
    if (
      ["#text", "#comment"].includes(children[i].nodeName)
    )
      continue;

    console.log(children[i].nodeName, children[i]);
  }
}

logBodyChildTagNames();
```

```
SCRIPT                                 main.js:10
  ▶ <script>…</script>
>
```

14. Use `firstChild` and `firstElementChild` on a container and explain the difference in a comment.



```javascript
const container = document.getElementById("container");

console.log("firstChild:", container.firstChild);
console.log(
  "firstElementChild:",
  container.firstElementChild
);

/*
- firstChild returns the first node inside #container
(text, comment, or element)
----------------------------------------------------
- firstElementChild returns only the first element,
(skipping comments and text)
*/
```

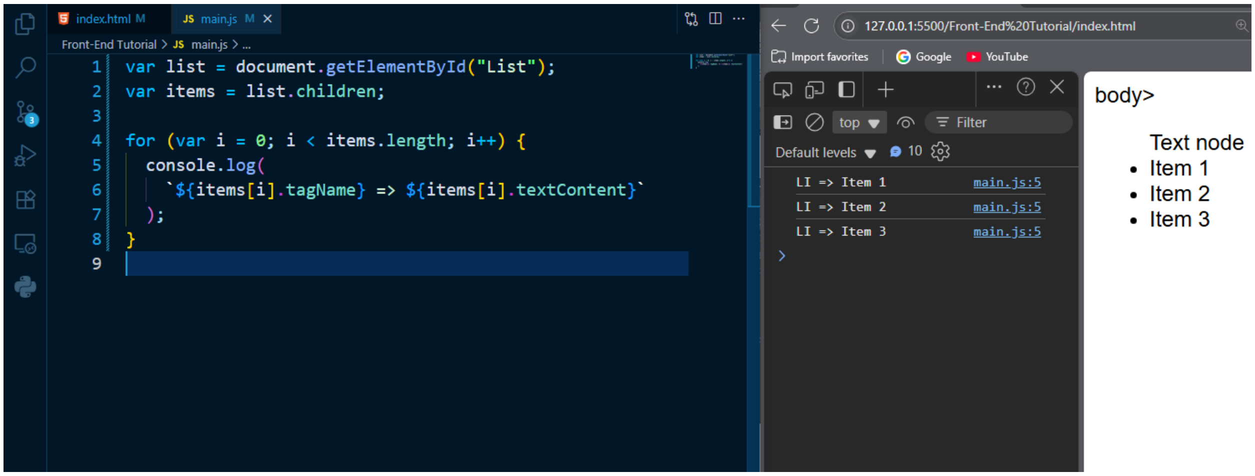
```
firstChild:                    main.js:3
  " Text Before First Element "
firstElementChild:             main.js:4
  <p>This is the first paragraph.
  </p>
>
```

Text Before First Element

This is the first paragraph.

This is the second paragraph.

15. Write code to get all `<li>` elements inside a `<ul>` using `children` and explain the difference.


index.html M
main.js M
Front-End Tutorial > main.js > ...
127.0.0.1:5500/Front-End%20Tutorial/index.html
Import favorites
Google
YouTube
top
Filter
Default levels
10

```javascript
var list = document.getElementById("List");
var items = list.children;

for (var i = 0; i < items.length; i++) {
  console.log(
    `${items[i].tagName} => ${items[i].textContent}`
  );
}
```

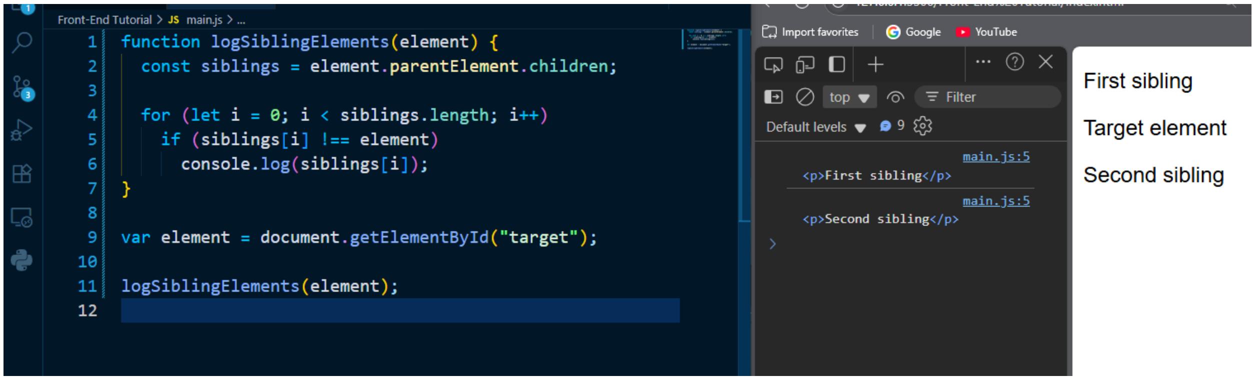
```
LI => Item 1    main.js:5
LI => Item 2    main.js:5
LI => Item 3    main.js:5
>
```

body>

Text node

- Item 1
- Item 2
- Item 3

16. Write a function that logs all sibling elements of a given element (excluding itself).



```javascript
function logSiblingElements(element) {
  const siblings = element.parentElement.children;

  for (let i = 0; i < siblings.length; i++)
    if (siblings[i] !== element)
      console.log(siblings[i]);
}

var element = document.getElementById("target");

logSiblingElements(element);
```

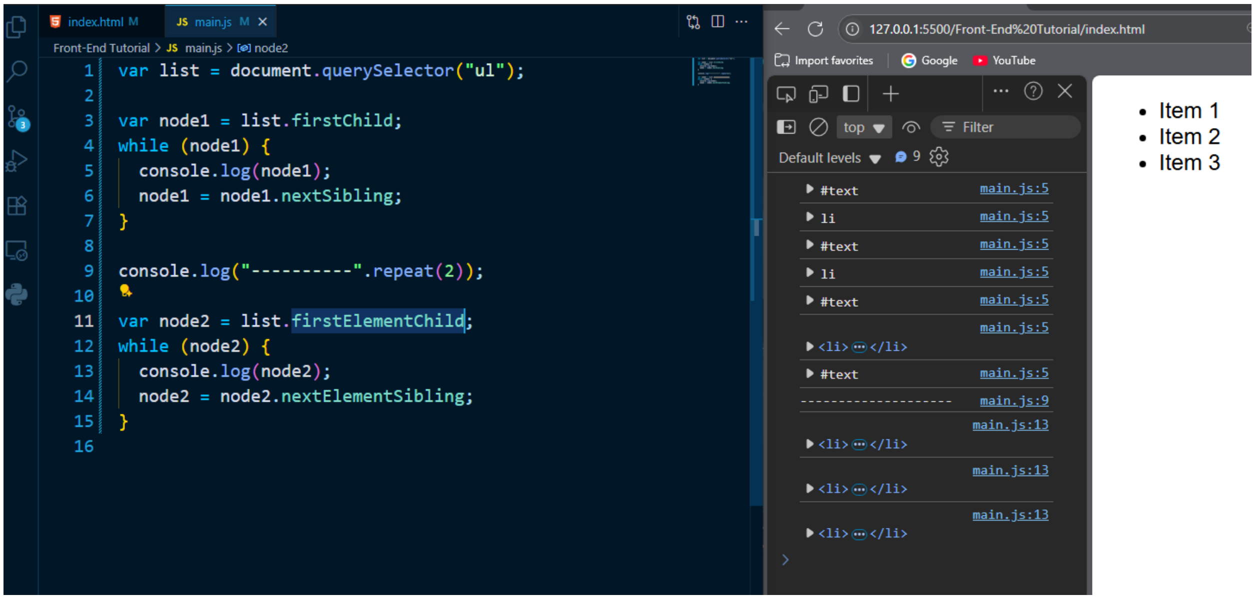
```
                          main.js:5
  <p>First sibling</p>
                          main.js:5
  <p>Second sibling</p>
>
```

First sibling

Target element

Second sibling

17. Use `nextSibling` and `nextElementSibling` to traverse from the first child to the last child of a `<ul>`, logging each node.

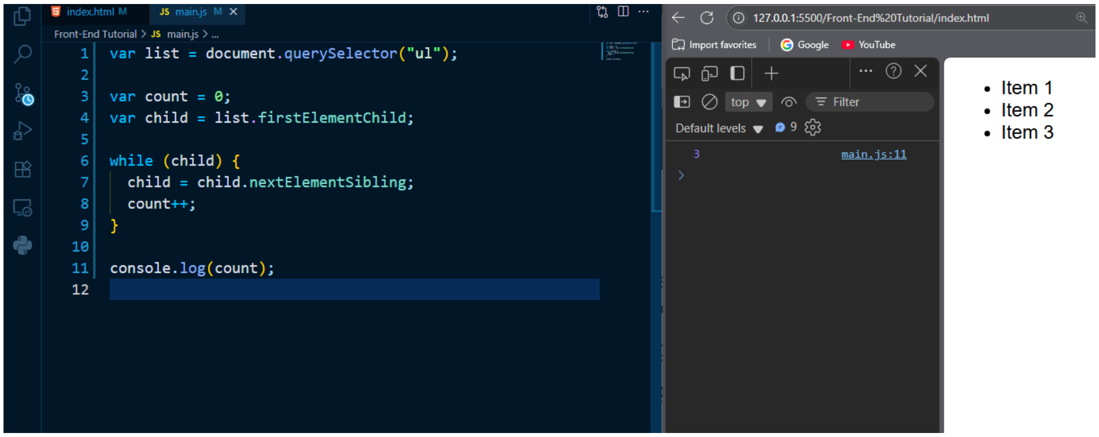

index.html M
main.js M
Front-End Tutorial > main.js > node2
127.0.0.1:5500/Front-End%20Tutorial/index.html
Import favorites
Google
YouTube
top
Filter
Default levels
9

```
var list = document.querySelector("ul");

var node1 = list.firstChild;
while (node1) {
  console.log(node1);
  node1 = node1.nextSibling;
}

console.log("----------".repeat(2));

var node2 = list.firstElementChild;
while (node2) {
  console.log(node2);
  node2 = node2.nextElementSibling;
}
```

```
▶ #text                 main.js:5
▶ li                    main.js:5
▶ #text                 main.js:5
▶ li                    main.js:5
▶ #text                 main.js:5
                        main.js:5
▶ <li>…</li>
▶ #text                 main.js:5
--------------------    main.js:9
                        main.js:13
▶ <li>…</li>
                        main.js:13
▶ <li>…</li>
                        main.js:13
▶ <li>…</li>
>
```

- Item 1
- Item 2
- Item 3

18. Count how many element children a given node has (not using `children.length`).

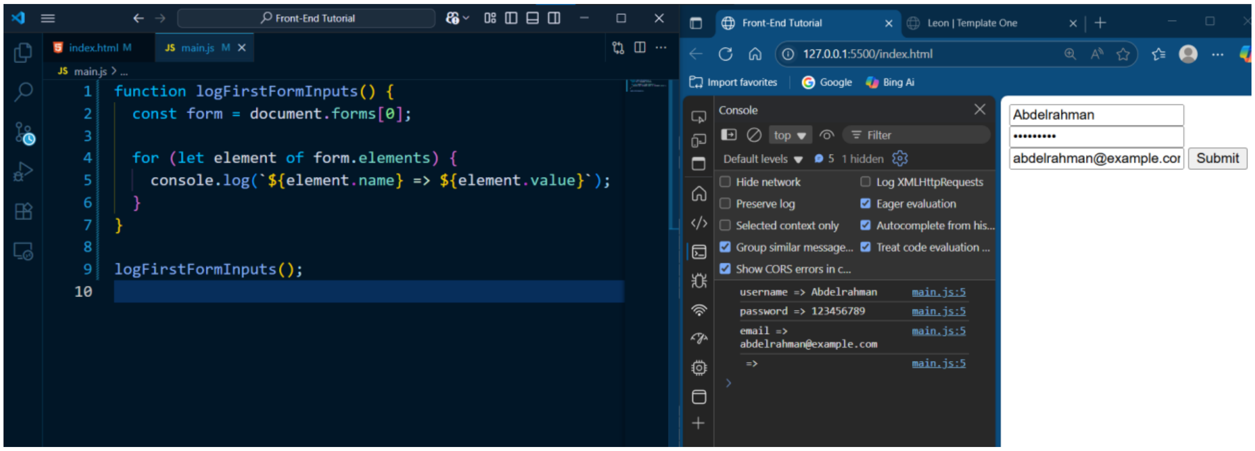

index.html M
main.js M
Front-End Tutorial > main.js > ...
127.0.0.1:5500/Front-End%20Tutorial/index.html
Import favorites
Google
YouTube
top
Filter
Default levels
9

```
var list = document.querySelector("ul");

var count = 0;
var child = list.firstElementChild;

while (child) {
  child = child.nextElementSibling;
  count++;
}

console.log(count);
```

```
3                       main.js:11
>
```

- Item 1
- Item 2
- Item 3

19. Write a function that takes the first form element and logs the names and values of all its input fields using the `elements` collection.



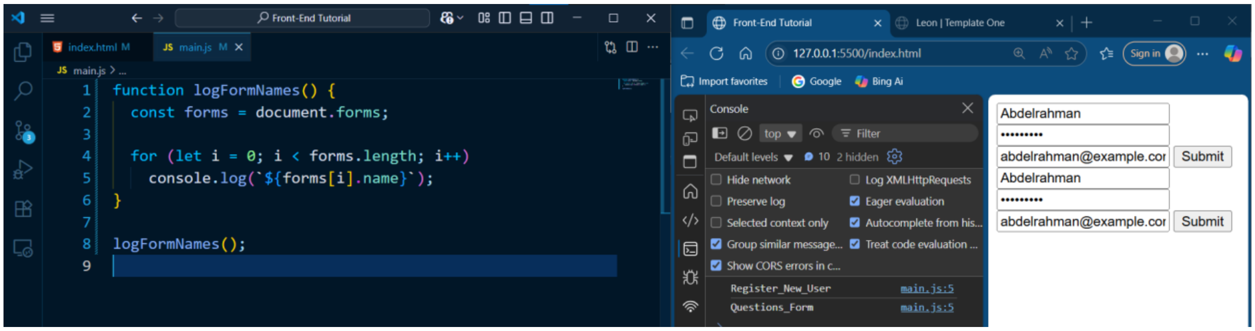
```javascript
function logFirstFormInputs() {
  const form = document.forms[0];

  for (let element of form.elements) {
    console.log(`${element.name} => ${element.value}`);
  }
}

logFirstFormInputs();
```

☐ Hide network ☐ Log XMLHttpRequests
☐ Preserve log ☑ Eager evaluation
☐ Selected context only ☑ Autocomplete from his...
☑ Group similar message... ☑ Treat code evaluation ...
☑ Show CORS errors in c...

```
username => Abdelrahman              main.js:5
password => 123456789                main.js:5
email =>                             main.js:5
abdelrahman@example.com
 =>                                  main.js:5
```

Abdelrahman
•••••••••
abdelrahman@example.cor

20. Access all forms in the document using `document.forms` and log their names.



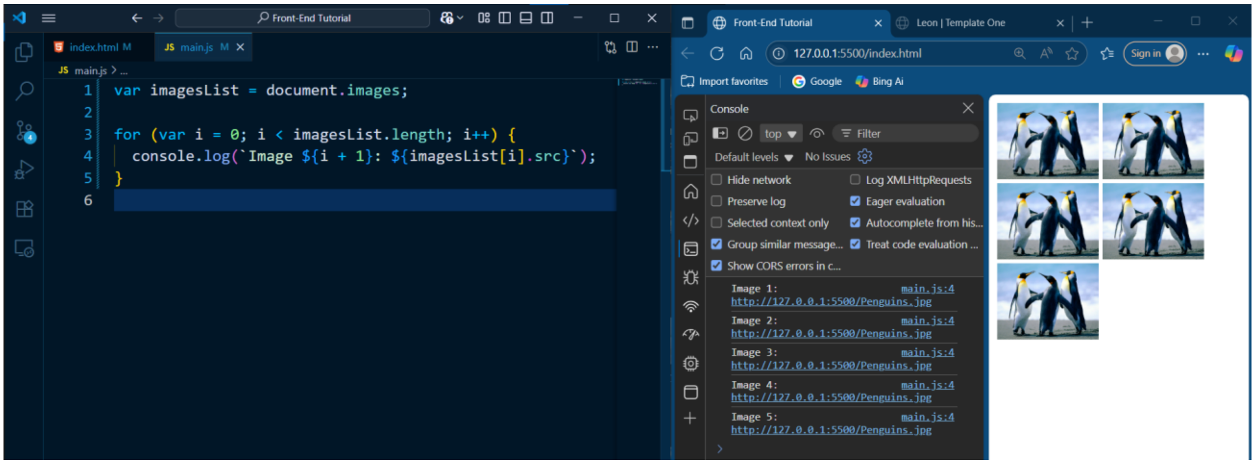
```javascript
function logFormNames() {
  const forms = document.forms;

  for (let i = 0; i < forms.length; i++)
    console.log(`${forms[i].name}`);
}

logFormNames();
```

☐ Hide network ☐ Log XMLHttpRequests
☐ Preserve log ☑ Eager evaluation
☐ Selected context only ☑ Autocomplete from his...
☑ Group similar message... ☑ Treat code evaluation ...
☑ Show CORS errors in c...

```
Register_New_User                    main.js:5
Questions_Form                       main.js:5
```

Abdelrahman
•••••••••
abdelrahman@example.cor
Abdelrahman
•••••••••
abdelrahman@example.cor

21. Access all images in the document using `document.images` and log their `src` attributes.

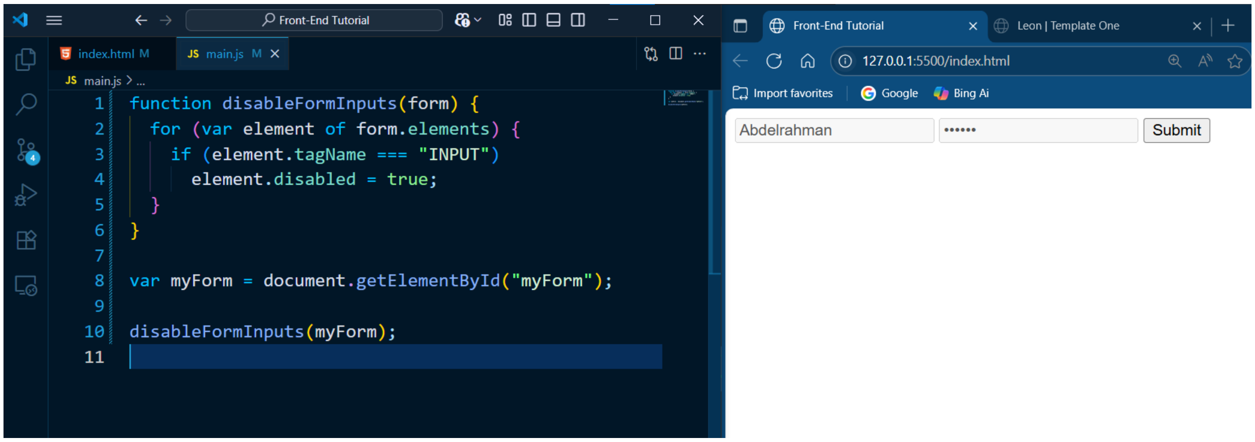


```javascript
var imagesList = document.images;

for (var i = 0; i < imagesList.length; i++) {
  console.log(`Image ${i + 1}: ${imagesList[i].src}`);
}
```

☐ Hide network ☐ Log XMLHttpRequests
☐ Preserve log ☑ Eager evaluation
☐ Selected context only ☑ Autocomplete from his...
☑ Group similar message... ☑ Treat code evaluation ...
☑ Show CORS errors in c...

```
Image 1:                             main.js:4
http://127.0.0.1:5500/Penguins.jpg
Image 2:                             main.js:4
http://127.0.0.1:5500/Penguins.jpg
Image 3:                             main.js:4
http://127.0.0.1:5500/Penguins.jpg
Image 4:                             main.js:4
http://127.0.0.1:5500/Penguins.jpg
Image 5:                             main.js:4
http://127.0.0.1:5500/Penguins.jpg
```

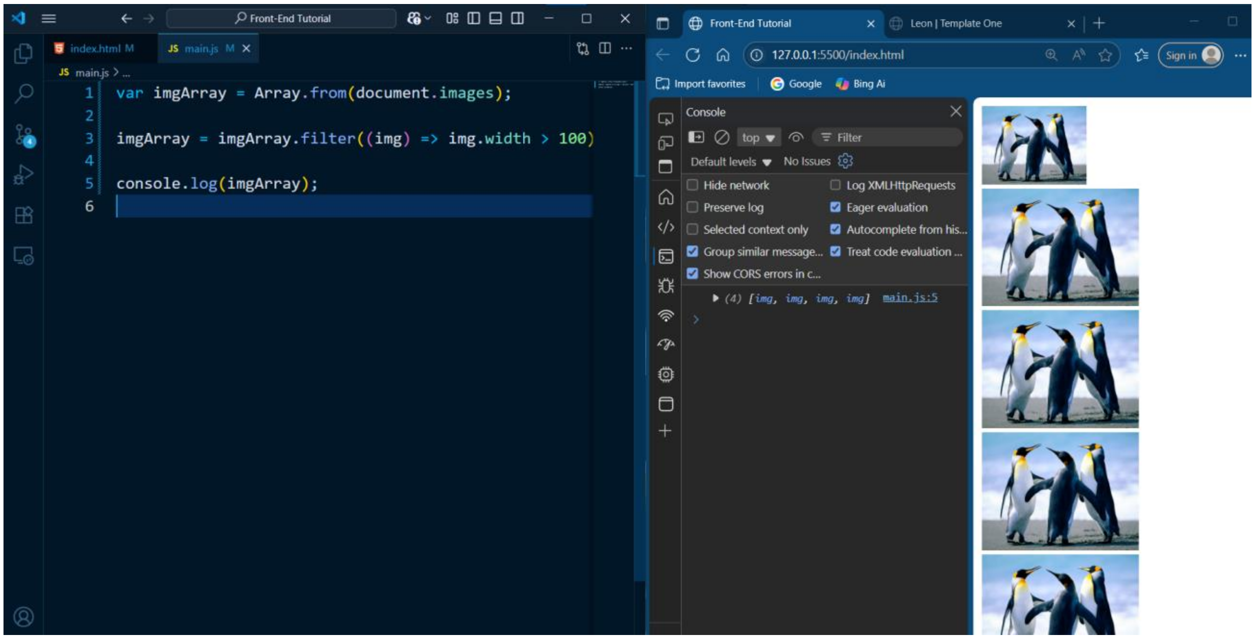
22. Write a function that takes a form and disables all its input fields using the `elements` collection.



```javascript
function disableFormInputs(form) {
  for (var element of form.elements) {
    if (element.tagName === "INPUT")
      element.disabled = true;
  }
}

var myForm = document.getElementById("myForm");

disableFormInputs(myForm);

```

Abdelrahman ••••••

23. Use `Array.from` to convert `document.images` to an array and filter images with width > 100px.



```javascript
var imgArray = Array.from(document.images);

imgArray = imgArray.filter((img) => img.width > 100)

console.log(imgArray);

```

☐ Hide network ☐ Log XMLHttpRequests
☐ Preserve log ☑ Eager evaluation
☐ Selected context only ☑ Autocomplete from his...
☑ Group similar message... ☑ Treat code evaluation ...
☑ Show CORS errors in c...

```
▶ (4) [img, img, img, img]   main.js:5
>
```