AeroAspire Intern

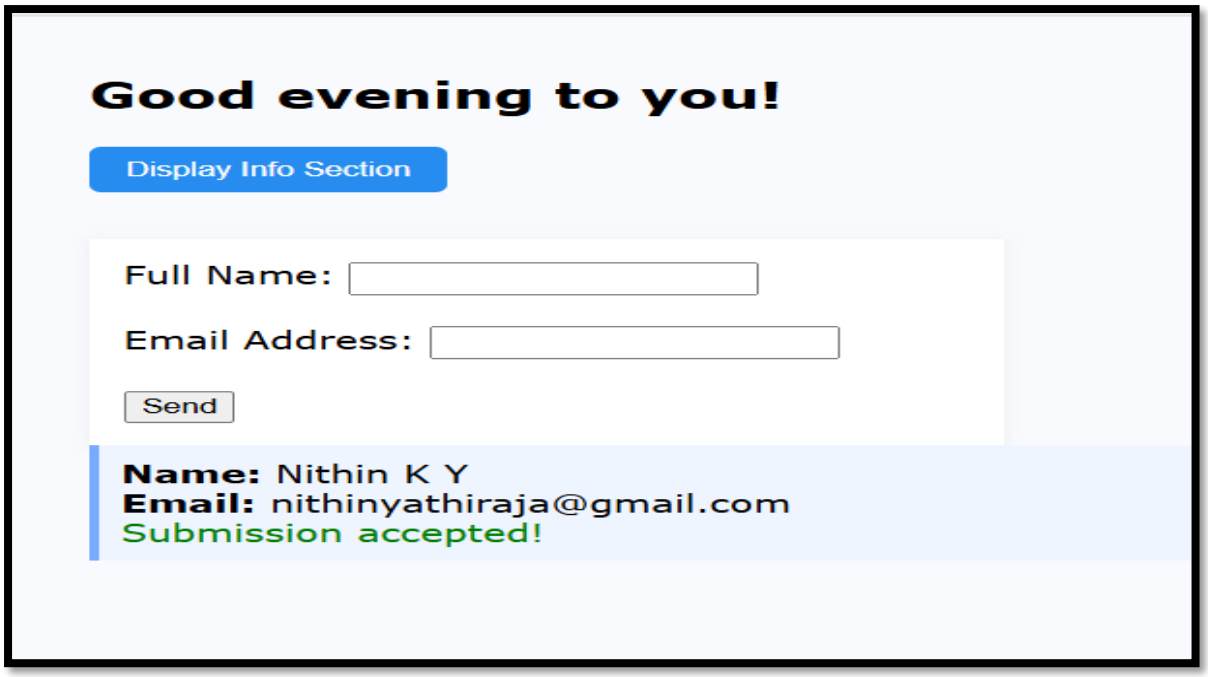
Nithin K Y

Day 2:- Add script to show greeting based on current time; add button to hide/show a section. Create form with name/email; show data on page on submit without reload; input validation.

In this we can see that I have not entered anything in the name column so

It is showing as enter at least 2 alphabetic letters

If the user enters nothing, or enters symbols or single letters, the validation fails.



In this case the input is entered in a right way so there is no error .

Questions:- 1. Difference between let, const, var?

- **var** is either globally scoped (if declared outside a function) or function scoped (if declared inside a function).
- **let** and **Const** are block scoped, which means they are only accessible within the block ({}) where they are defined.
- **Var** can be reassigned and redeclared within its scope
- **Let** can be reassigned but not redeclared within its block scope
- **Const** cannot be reassigned or redeclared; its value is constant after initialization

2. What is event listener?

- Events are actions such as user clicks, mouse movements, or key presses.
- The event listener "listens" for these events on a target element.

- When the event happens, the event listener runs a function you specify. They are used to :-
- They separate JavaScript from HTML, making code cleaner.
- You can attach multiple event listeners to a single element.
- They allow dynamic, interactive web pages that respond to user actions.

3. Walk through the event flow when you click the button: how does JS know which DOM element, what listener, what callback?

Capturing → The event travels from top (window, document, body) down to the button.
Target → The event reaches the button itself and runs its handlers.
Bubbling → The event travels back up the DOM tree (button → body → document → window).

When you click, the browser creates an event object with the target element (the button). It knows the path through the DOM and checks each element's stored event listeners (from addEventListener). For every matching listener in the right phase (capturing, target, bubbling), the browser calls the linked callback function in order.

4. What is the difference between event capturing vs bubbling?

- Event Capturing:
  The event starts from the outermost ancestor (the root)
  and travels down to the target element. Event listeners set with {capture: true} listen during this phase. It means the event is "captured" first by ancestors before reaching the target.
- Event Bubbling:
  The event starts at the target element where the event occurred and then bubbles up through the ancestors back to the root. Event listeners set with

the default or {capture: false} listen during this phase. This means the target handles the event first, then its parents.

5. How could you debug JS errors in browser dev tools?

1. Open DevTools: Press F12 (or Ctrl+Shift+I / Cmd+Option+I) and go to the Console to view error messages.
2. Add Breakpoints: In the Sources (or Debugger) panel, click the line number in your JS code to pause execution at specific points.
3. Step Through Code: Use "Step Over", "Step Into", and "Step Out" buttons to execute code line-by-line and inspect behavior.
4. Inspect Variables: Watch variable values and call stack to understand program state during pause.
5. Use console.log(): Print variable values and checkpoints to the Console for quick insight.
6. Use debugger Statement: Insert debugger; in code to automatically pause execution when DevTools are open.

6. What is event.preventDefault(), why/when you use it?

event.preventDefault() stops the browser's default behavior (like a form submitting or a link navigating). It's used when you want to handle the action with your own JavaScript instead of letting the browser do it automatically.

7.How to validate fields? How do you validate form fields (required, email format, etc.) in JS?

In JavaScript, you validate form fields by checking their values before submission—like ensuring required fields are not empty, verifying email with a regex pattern, and checking number or length limits. These checks are usually

done on the form’s submit event, and if any field is invalid, you use event.preventDefault() to stop submission and show an error message.

8.Describe data flow from user input → validation → UI feedback.

When a user enters input, the value is captured from the form field and passed to validation logic. The validation checks rules (e.g., required, format, length), and if the input is invalid, submission is blocked and an error message is shown near the field or in the UI. If the input is valid, the form proceeds, and the user sees success feedback or the data is sent to the server.