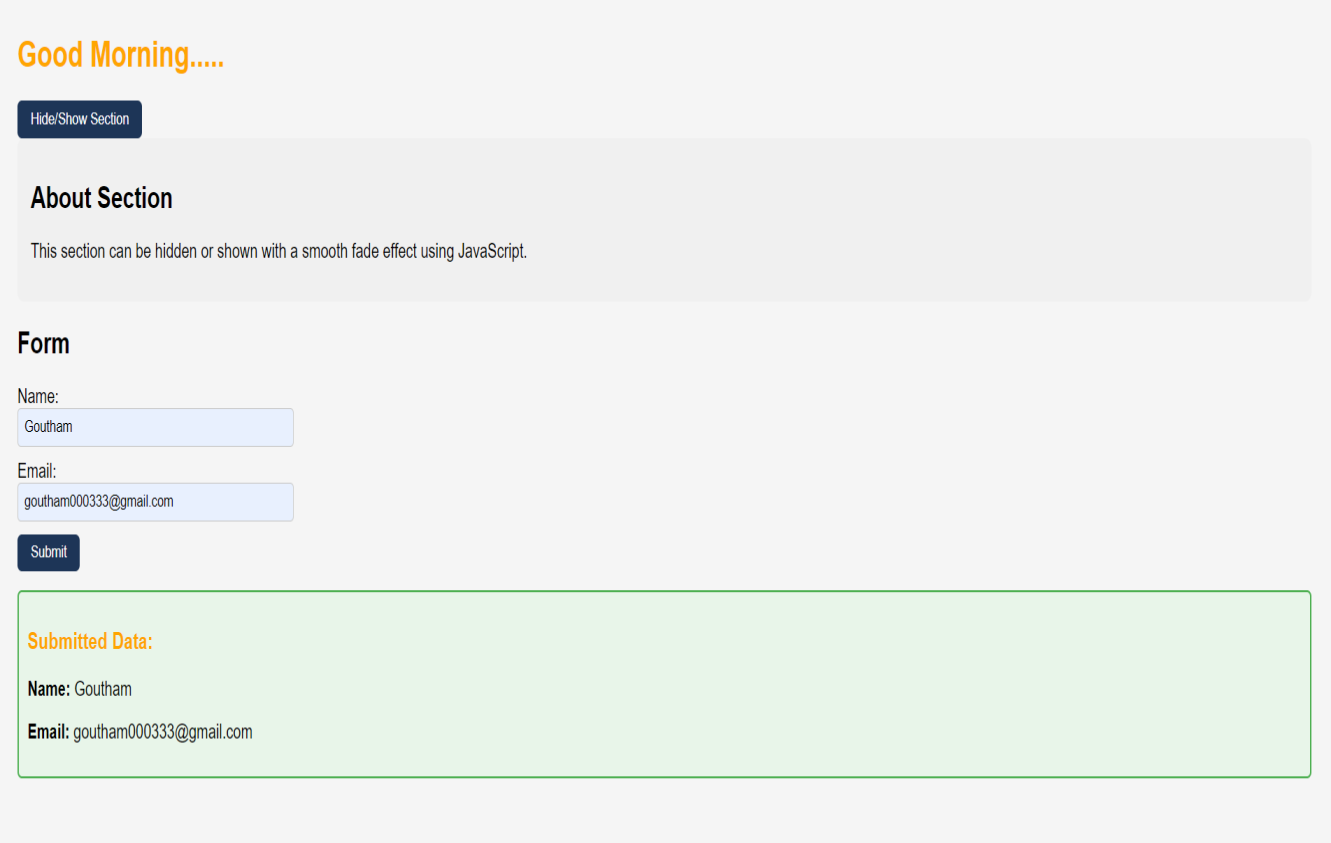
AeroAspire-SDE Intern

Goutham V

Week 1 – Day3 (24rd September)

Task:

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.



Day 2 Task Output: Dynamic greeting, hide/show section, and form submission with live display and validation.

The page shows a greeting at the top that changes every few seconds. There is an About section that can be hidden or shown using a button with a smooth effect. Below that, there is a form to enter Name and Email, and when submitted, the data is shown on the page without reloading, with validation to make sure the fields are filled.

Reflection,

1. Difference between let, const, var?

- var is function-scoped and can be updated or re-declared. It's hoisted, meaning it will be moved to the top of its scope.
- let is block-scoped, can be updated but not re-declared in the same scope. It's also hoisted, but stays in the "temporal dead zone" until initialized.
- const is block-scoped and cannot be updated or re-declared. It must be initialized when declared, and its value cannot change.

2. What is event listener?

- An event listener is a function in JavaScript that waits for a specific event (like click,keydown, etc.) on a DOM element and then executes some action when the event happens.
- You can attach event listeners to an element using `addEventListener()`.

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

-When a button is clicked:

- The event begins at the window and travels down the DOM tree (capturing phase).
- It reaches the target element (in this case, the button).
- JS checks if there's an `addEventListener` defined for the event type on the clicked element and runs its callback function.
- If no listener is set on the target, it bubbles back up the DOM tree, checking parent elements for listeners as it goes.

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

- **Capturing:** The event starts from the outermost parent and travels down to the target element.
- **Bubbling:** The event starts from the target element and goes up through its ancestors.
- By default, bubbling is enabled in JavaScript, so events are handled starting from the innermost element and move outward

.

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

- Use the Console panel to check error messages.
- Add breakpoints in the Sources panel to pause code and inspect variables.
- Step through code execution line by line.
- Use the Elements panel to check HTML structure and applied styles.
- Use the Network panel to monitor requests if the error is related to data fetching.

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

-The event.preventDefault() method stops the browser from performing its default action for a given event. For example, it can stop a link from navigating to another page or keep a form from submitting. Use it when you want to provide custom functionality instead of the browser's default behavior, often during form validation or AJAX submissions..

7. How to validate fields?

- Check if required fields are not empty.
- Confirm if inputs meet certain patterns (like email format).
- Add custom checks for passwords, phone numbers, etc.
- Show error messages next to invalid fields and block form submission if needed.

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

-You validate form fields in JavaScript by:

- Adding event listeners for form submission.
- Checking each field value against your validation rules (e.g., required, pattern matching for emails).
- Displaying error messages and preventing the form from submitting if conditions aren't met.
- Using regular expressions for formats like emails or passwords

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

-When a user enters input in a form, the data is captured and then immediately validated using JavaScript, typically on form submission or on input change. The code checks that the fields meet criteria like being filled or matching a format. If validation fails, error messages are shown instantly on the UI. If successful, the form can be submitted, and any feedback (like “Thank you!”) is displayed to the user, ensuring a smooth and interactive experience.