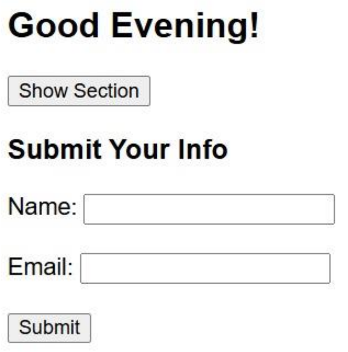
AeroAspire-SDE Intern
Gokul Krishna S

Week 1 – Day 2 (September 24)

Task/Assignment :

- 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.

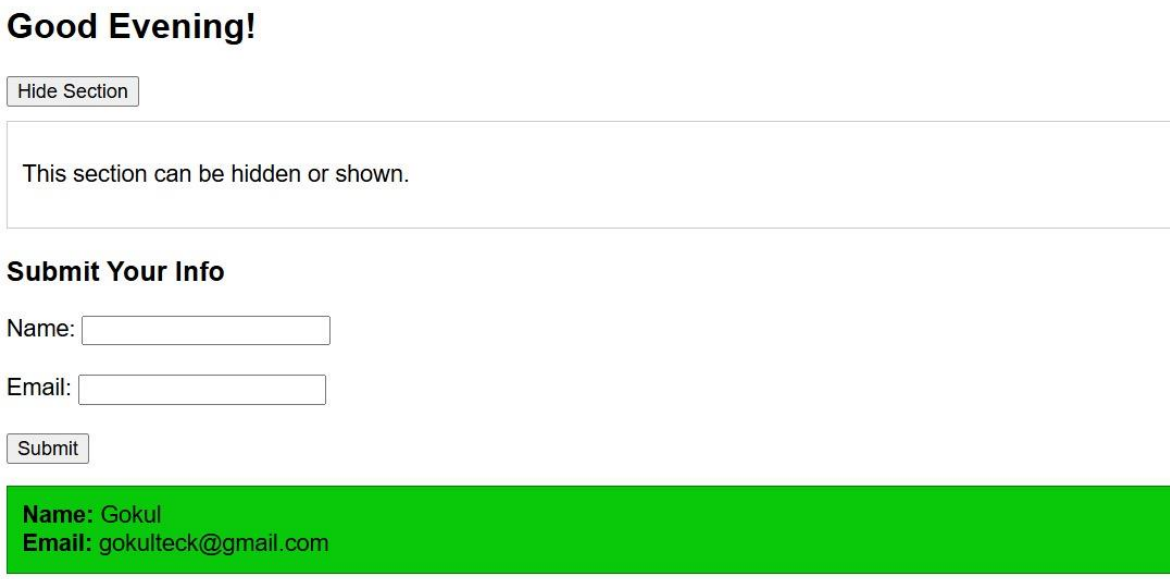


Good Evening!

Submit Your Info

Name:

Email:



Good Evening!

This section can be hidden or shown.

Submit Your Info

Name:

Email:

Name: Gokul
Email: gokulteck@gmail.com

Questions/Reflections :

1. Difference between var, let, const ?

- **var** – function-scoped, can be redeclared and updated.
- **let** – block-scoped, can be updated but not redeclared in the same scope.
- **const** – block-scoped, cannot be updated or redeclared; must be initialized.

2. What is event listener?

- Event listeners wait for an event (like click, input, or submit) on an element.
- addEventListener(event, function) attaches a function that runs when the event occurs.

3. Walk through the event flow when you click the button:

- **Event capturing** – the event goes from document down to the target element.
- **Event bubbling** – the event propagates from the target element up to document.

4. How does JS know which DOM element, what listener, what callback?

- JavaScript knows which DOM element, which listener, and which callback through references and event binding
- The browser detects the event (click, input, submit, etc.) on the element.
- JS checks the element’s listener list to see if any function is bound to this event.
- JS executes the callback function, optionally passing an event object **(e)** containing details like target, type, coordinates, etc.

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

- **Event Capturing -** The event starts at the top of the DOM and moves down through parent elements until it reaches the target element.
- **Event Bubbling -** After the event reaches the target, it bubbles back up, triggering handlers on each ancestor element along the way.

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

Use console Methods :

- console.log(value) → print variables
- console.error(error) → print errors in red
- console.warn(message) → warning in yellow
- console.table(array) → show arrays/objects in table format
- console.dir(object) → inspect DOM objects

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

- event.preventDefault() is a method on the event object (event or e) in JavaScript.
- It prevents the browser's default behavior for that event from happening.

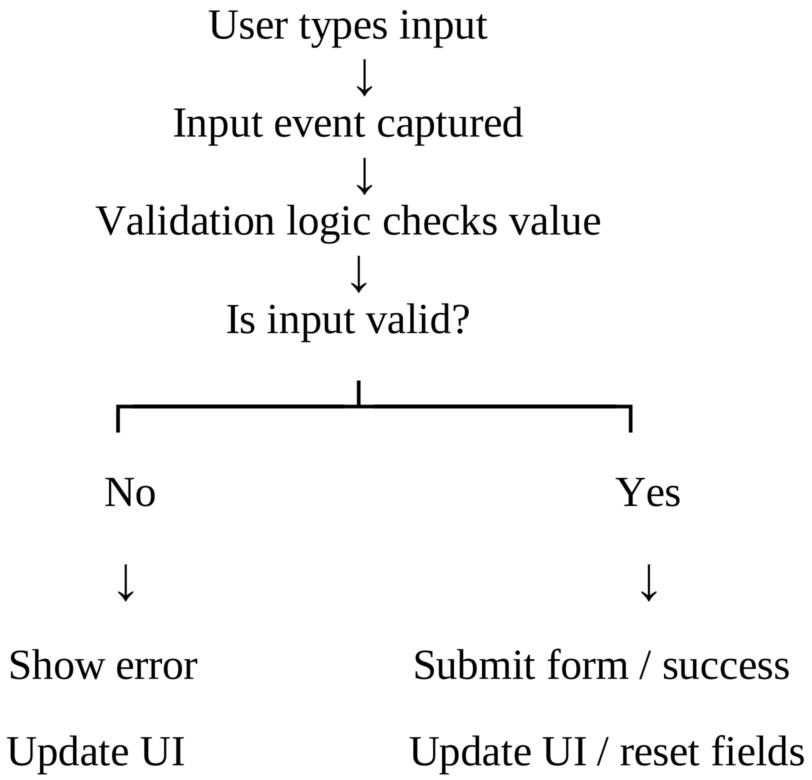
8. How to validate fields? How do you validate form fields (required, email format, etc.) in JS?

- **Prevent Default Submission** – Use event.preventDefault() to stop the form from submitting so you can validate first.
- **Check Required Fields** – Ensure mandatory fields are not empty using .value.trim().
- **Validate Length** – Check minimum or maximum character length for text inputs.
- **Validate Format** – Use regular expressions (regex) to check email, phone number, or password formats.
- **Provide Feedback** – Display error messages in the UI and only submit/reset the form if all validations pass.

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

- **User Input** – The user types data into form fields, triggering events like input or change.

- **Event Trigger** – JavaScript listens for events (e.g., submit) to know when to validate.
- **Validation** – JS checks input values against rules: required fields, format, length, etc.
- **Decision / Logic** – If validation fails, stop submission; if it passes, proceed to next actions.
- **UI Feedback** – Display error messages for invalid input or success messages/reset form for valid input.


User types input
↓
Input event captured
↓
Validation logic checks value
↓
Is input valid?
No
Yes
↓
↓
Show error
Update UI
Submit form / success
Update UI / reset fields