Experiment 07

# **Aim:** Perform from the following to develop interactive web pages using JavaScript: a. Error handling, Validations, b. Arrays, String, Date

# **Requirements:** Chrome, VsCode

## **Theory:**

## **1) Error handling**

# JavaScript Errors - Throw and Try to Catch

* The try statement lets you test a block of code for errors.
* The catch statement lets you handle the error.
* The throw statement lets you create custom errors.
* The finally statement lets you execute code, after try and catch, regardless of the result.

## JavaScript try and catch

The try statement allows you to define a block of code to be tested for errors while it is being executed. The catch statement allows you to define a block of code to be executed, if an error occurs in the try block.

The JavaScript statements try and catch come in pairs:

try {  
  *Block of code to try*  
}  
catch(*err*) {  
  *Block of code to handle errors*  
}

**2) Validations**

HTML form validation can be done by JavaScript. If a form field (fname) is empty, this function alerts a message, and returns false, to prevent the form from being submitted:

### JavaScript Example

function validateForm() {  
  let x = document.forms["myForm"]["fname"].value;  
  if (x == "") {  
    alert("Name must be filled out");  
    return false;  
  }  
}

**3) Arrays**

In array is a special variable, which can hold more than one value at a time. If you have a list of items (a list of car names, for example), storing the cars in single variables could look like this:

let car1 = "Saab";  
let car2 = "Volvo";  
let car3 = "BMW";

However, what if you want to loop through the cars and find a specific one? And what if you had not 3 cars, but 300? The solution is an array! An array can hold many values under a single name, and you can access the values by referring to an index number. Using an array literal is the easiest way to create a JavaScript Array.

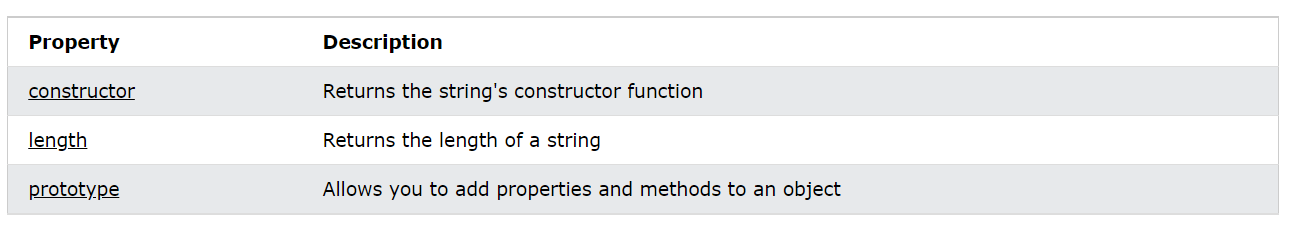
Syntax:

const array\_name = [item1, item2, ...];

**3) Strings**

Primitive values, like "John Doe", cannot have properties or methods (because they are not objects).

But with JavaScript, methods and properties are also available to primitive values, because JavaScript treats primitive values as objects when executing methods and properties.

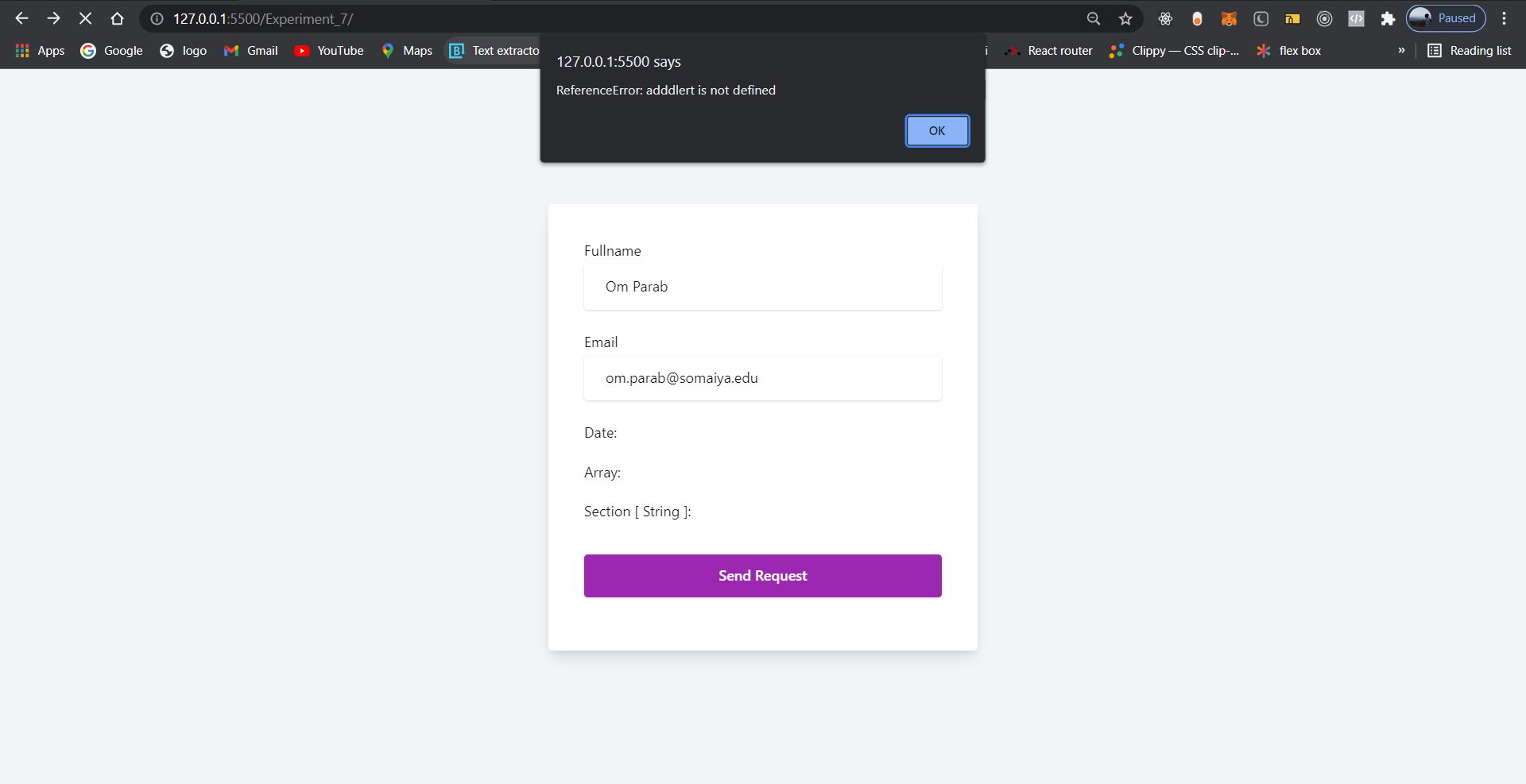
****

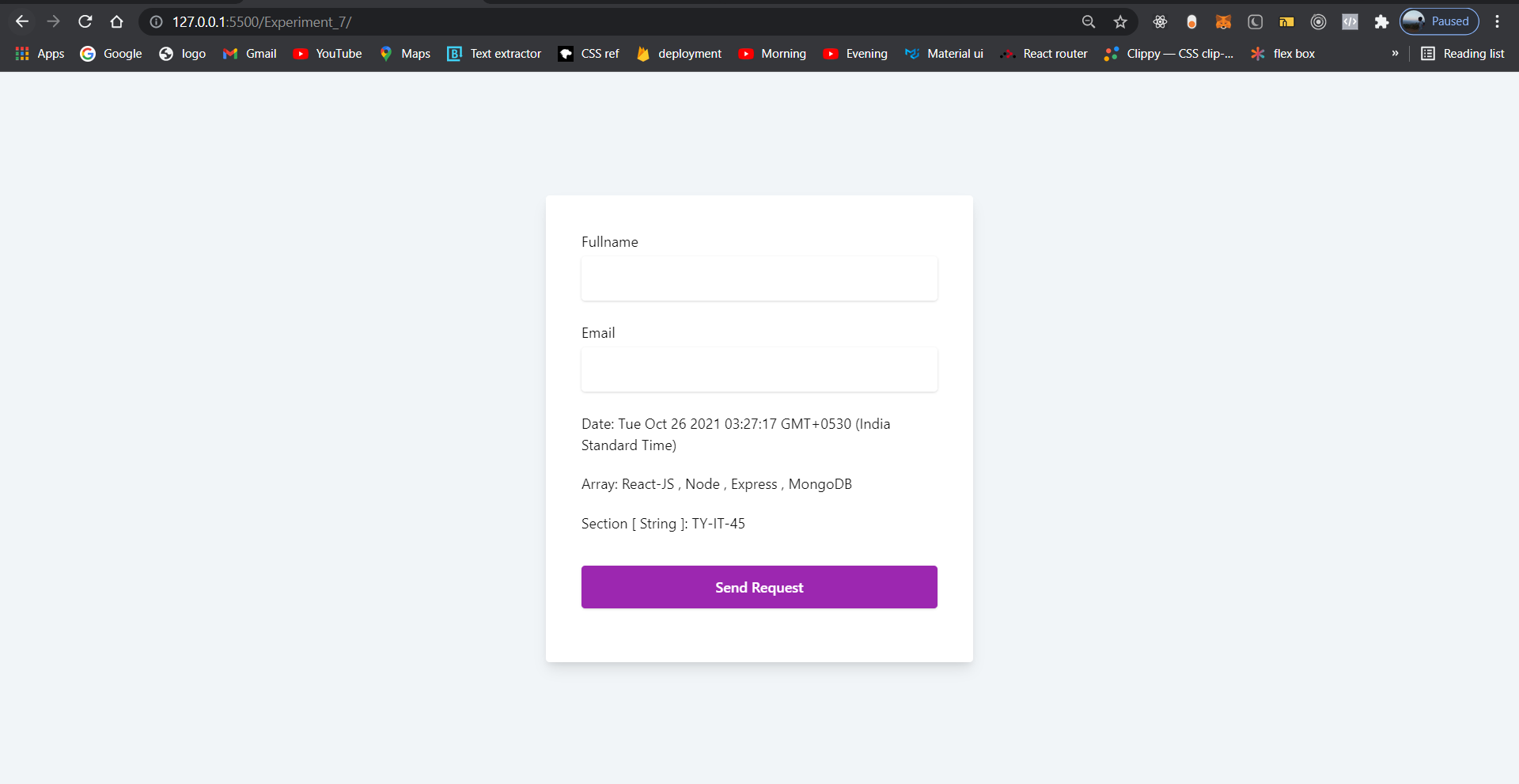
**3) Date**

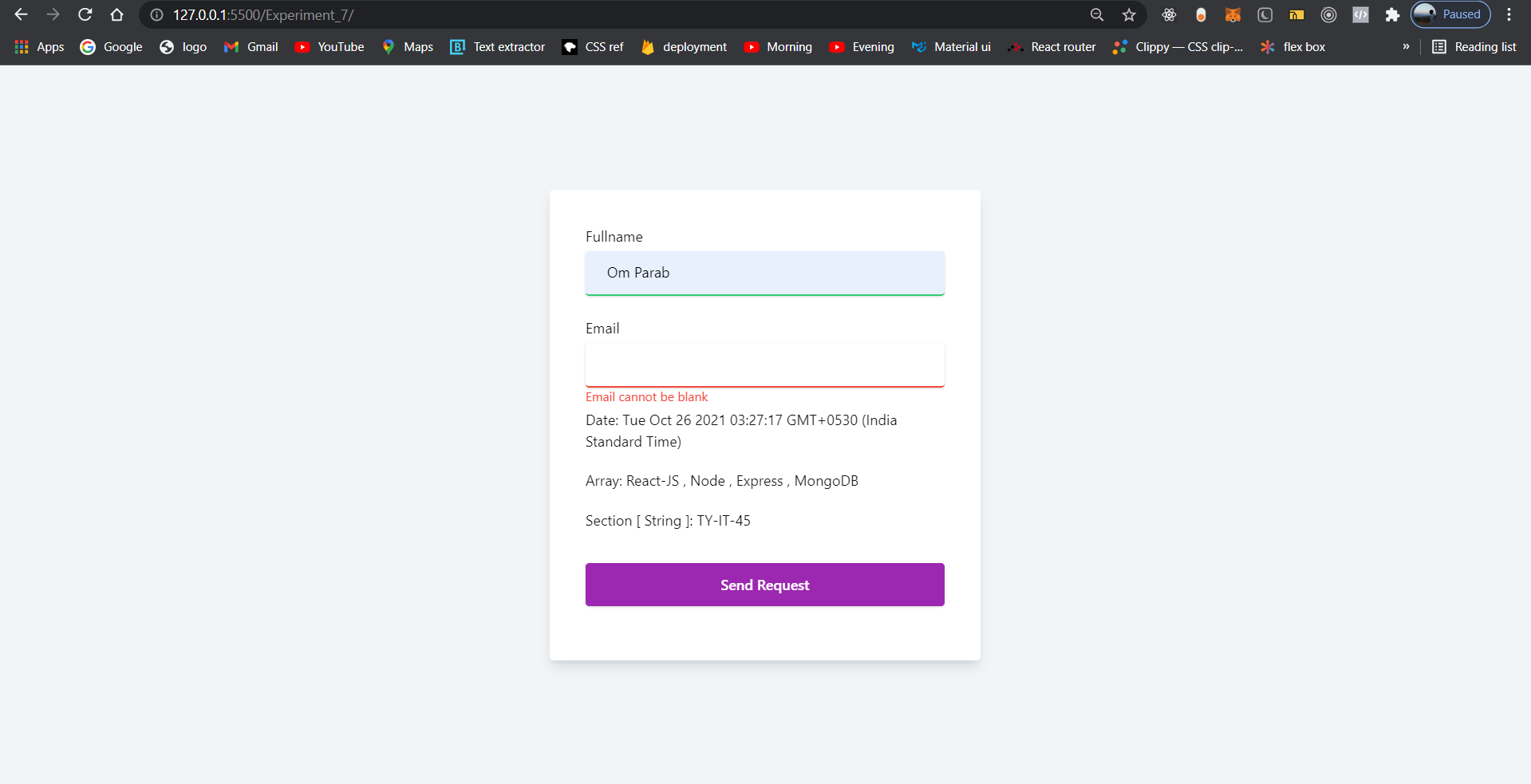
Date objects are created with the new Date() constructor.

new Date()  
new Date(year, month, day, hours, minutes, seconds, milliseconds)  
new Date(milliseconds)  
new Date(date string)

# **Output:**







# **Conclusion:** Using JavaScript we can do error handling and form validations very easily. In Js arrays, string, date objects are very useful for making interactive ui design

**References**: <https://www.w3schools.com>