Experiment No: 07

Aim: Program using javaScript to valid the email address entered by the user (check the presence of "@" and "." character. If this character is missing, the script should display an alert box reporting the error and ask the user to re-enter it again).

Prerequisites: It is required that students should knows the basic HTML, style sheet, and javascript systax and Methods.

Objectives:
- To validate an email address entered by the users using javascript.
- To ensure the emial contains both "@" and "." characters.
- To display an error message if the email is invalid and ask the user to reeneter.
- To make the Form visually attractive using CSS styling.

Theory:

What is email validation ?

- Email validation is a process of checking whether the entered email address is syntatically correct.
- A valid email moot cantains
  "@" symbol
  "." symbol.

Role of javascript:

- java scripts interacts with the DOM (Document object Mode) to access the email field value.
- if check if "@" and "." are present using.
- ifC email.index ("@") ==> -1 || email. indexOf(".") === -1) {...}
- if missing, an allert box display an error massage Role of css.
- Css is to used to make the form attractive and user friendly
- Features added: Gradient background.
  Centered form container
  Rounded input field and buttons
  Hover effects.

Explanation of programs:

1. HTML part.
   - A form is created with a text input field (email) and a submit button.
   - On submission, the javascript function validateEmail () is called.

2. CSS part.
   - Background uses a blue-purple gradient.
   - A container div act as a card with shadow and rounded corners.
   - Input field glow when focused.
   - submit botton changes colour on hover.

3. Java Script part.

- The function fetches the entered value using.
- Var email = document.getElementById("Email").value;
- It check for the presence of "@" and ".".
- if missing → error alert + input field cleared + cursor refocused.
- If present → success alert.

Conclusion:

- The program successfully validates the email address using javascript.
- The demonstrate DOM manupulation by accessing and modifying input values.
- CSS styling enhance the visual appeal, making the form interactive and user-friendly.
- The approach is useful in real-life applications where user input validation is critical (e.g. login forms, sign-up pages).

Experiment No: 08

Aim: Program based on document object model to change the background colour of the webpage automatically after every 5 seconds.

Objectives:

- To understand the concepts of Document Object Model (DOM) in javascript.
- To demonstrate how javascript can dynamically manipulate HTML elements.
- To create a program that change the background color of the webpage automatically at regular intervals.

Theory:

What is DOM ?

- The document object model (DOM) is a programming interface for web documents.
- It represents the structure of an HTML or XML document as a tree of objects.
- With DOM, programming languages like javascripts can:
  - Access HTML elements,
  - Modify their properties,
  - changes styles,
  - Add or delete content dynamically.

DOM and Javascript :

- The browser automatically creates a DOM tree when n HTML pages loads.
- Javascript can interact with this tree using method like :

  document. getElementById ()

  document. getElementByTagName ()

  document. body (direct access to the <body> tags).

Background colour change using DOM.

- In CSS, background colour is controlled using the property background-color.
- With DOM, this property can be modified dynamically.
- document. body. Style. background Color = "red";
- Using setInterval (), we can make this change happen automatically after a fixed time interval.

Explanation of programs :

1. HTML Section.
   - The <h2> tag is used to display a message in the center of the page.
2. JavaScript Section.
   - Colors[] : An array of 6 color codes is created.
   - Index : A variable to keep track of which colour is currently being applied.
   - function changeBackground ();

     Change the body background colour using

Update the index to move to the next color.

- setInterval (changeBackground , 5000);
  (call the function every 5000 milliseconds (5 seconds).

3. DOM Usage :
   - The < body > Tag is accessed as an object using document. body.
   - Its CSS property Background color is modified dynamically.

Output :

- When the program is run in a browser :

1. The pages opens with an initial background color.
2. After 5 seconds, the background colour change automatically.
3. This process continues, cyclings through the list of predefined colors.

Conclusion :

- The program successfully demonstrates how to use the Document object model (DOM) in javascript to manipulate webpage property dynamically.
- It show the ability to access and modify HTML elements (in this case, <body>) using DOM methods.
- The concept of setInterval () helps in automating task at regular intervals, which can be useful in creating interactive and dynamic webapplication.

Experiment No:9

Aim: program to making use of react Hooks that dioplays four buttons namely. "Red", "Blue", "Green", "Yellow". On clicking any of these buttons the code displays the message that you have selected that particular colour.

Implementation Steps:

Step 1: Install and Configure React.

Prerequistes:

Make sure the following components are installed:

1. Node.js (includes npm)
   - Download and install from: https://nodejs.org
   - verify installation: using command node -v or npm -v.

Step 2: Install create React App (Optional but recommended)

open command line and type command.

npm install -g create-react-app (Here replace my-app withyour project name.)

Step 3: Navigate into the projects folder.

cd my-app.

Step 4: Start the development server.

npm start.

This will launch your app in the browser at http://localhost:3000/.

How it works - explanation (step by-step).

1. useState ('') creates a state variable selected and setter setselected. Initially it's an empty string (no colour chosen).
2. handleClick (color) calls setSelected (color) to update state whenever a button is clicked. React rerender the components with the new selected value.
3. The message area renders conditionlly.
   - if selected is non-empty it shows you have selected <selected>
   - otherwise it shows "No color selected yet",
4. Styling changes (background colour of the message box) are done inline via style = {{ BackgroundColor: bgColor }}, Where bgColor is selected.tolowercase () (e.g. "red") so the message box visually matches the selected colour.
5. Buttons are use aria-pressed and .active styles for accessibility and visual feedback.

Run and Test!

1. In project folder:

   npm run dev

2. open http://localhost:5173 (or the URL Vite shows).
3. Click each buttons and confirm message updates and message box color changes.

Extra improvements you can try.

- Add keyboard navigation (left / Right arrow to move between buttons).
- Add Transition / animation when changing message.
- Replace color strings with hext values for finer control.
- Save last-selected colour to localStorage so it persists across reloads,
- Convert to TypeScript (--template reacts when creating with vite).

Troubleshooting:

- If npm create vite@latest Fails, try npx create vite@latest colorr-picker -- template reacts updatets npm.
- If node not found after install, restart your terminal or system.
- If port 5173 is in use, vite will suggest another port or run: npm run dev -- -- port 3000.