**// Multistepform**

```
<!DOCTYPE html>
<html>
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<link href="https://fonts.googleapis.com/css?family=Raleway" rel="stylesheet">
<style>
* {
  box-sizing: border-box;
}

body {
  background-color: #f1f1f1;
}

#regForm {
  background-color: #ffffff;
  margin: 100px auto;
  font-family: Raleway;
  padding: 40px;
  width: 70%;
  min-width: 300px;
}

h1 {
  text-align: center;
}

input {
  padding: 10px;
  width: 100%;
  font-size: 17px;
```

```
  font-family: Raleway;
  border: 1px solid #aaaaaa;
}

/* Mark input boxes that gets an error on validation: */
input.invalid {
  background-color: #ffdddd;
}
input.valid {
 border:1px green solid;
}

/* Hide all steps by default: */
.tab {
  display: none;
  /*visibility:hidden;*/
}
.error{
color:red;
}
.noerror {
  display: none;
}

button {
  background-color: #4CAF50;
  color: #ffffff;
  border: none;
  padding: 10px 20px;
  font-size: 17px;
  font-family: Raleway;
```

```
  cursor: pointer;
}

button:hover {
  opacity: 0.8;
}

#prevBtn {
  background-color: #bbbbbb;
}

/* Make circles that indicate the steps of the form: */
.step {
  height: 15px;
  width: 15px;
  margin: 0 2px;
  background-color: #bbbbbb;
  border: none;
  border-radius: 50%;
  display: inline-block;
  opacity: 0.5;
}

.step.active {
  opacity: 1;
}

/* Mark the steps that are finished and valid: */
.step.finish {
  background-color: #4CAF50;
}
```

```
</style>
<body>

<form id="regForm">
  <h1>Register:</h1>
  <!-- One "tab" for each step in the form: -->
  <div class="tab">Name:
    <p><input placeholder="First name..." oninput="validatefname(this)" name="fname"></p>
          <p id="nameerror"></p>
    <p><input placeholder="Last name..." oninput="this.className = ''" name="lname"></p>
  </div>
  <div class="tab">Contact Info:
    <p><input placeholder="E-mail..." oninput="this.className = ''" name="email"></p>
    <p><input placeholder="Phone..." oninput="this.className = ''" name="phone"></p>
  </div>
  <div class="tab">Birthday:
    <p><input placeholder="dd" oninput="this.className = ''" name="dd"></p>
    <p><input placeholder="mm" oninput="this.className = ''" name="nn"></p>
    <p><input placeholder="yyyy" oninput="this.className = ''" name="yyyy"></p>
  </div>
  <div class="tab">Login Info:
    <p><input placeholder="Username..." oninput="this.className = ''" name="uname"></p>
    <p><input placeholder="Password..." oninput="this.className = ''" name="pword"
type="password"></p>
  </div>
  <div style="overflow:auto;">
    <div style="float:right;">
      <button type="button" id="prevBtn" onclick="nextPrev(-1)">Previous</button>
      <button type="button" id="nextBtn" onclick="nextPrev(1)">Next</button>
    </div>
  </div>
```

```
  <!-- Circles which indicates the steps of the form: -->
  <div style="text-align:center;margin-top:40px;">
    <span class="step"></span>
    <span class="step"></span>
    <span class="step"></span>
    <span class="step"></span>
  </div>
</form>

<script>
var currentTab = 0; // Current tab is set to be the first tab (0)
showTab(currentTab); // Display the current tab
var validfname=false;

function validatefname(name){
var errorele=document.getElementById("nameerror");
if((name.value).length>2){
            validfname=true;
            name.className="valid";
            errorele.className="noerror";
            }
else{
            name.className="invalid";
            errorele.className="error";
            errorele.innerHTML="Enter a First Name with more than two characters";
            }
}
function showTab(n) {
  // This function will display the specified tab of the form...
  var x = document.getElementsByClassName("tab");
  x[n].style.display = "block";
```

```
  //... and fix the Previous/Next buttons:
  if (n == 0) {
    document.getElementById("prevBtn").style.display = "none";
  } else {
    document.getElementById("prevBtn").style.display = "inline";
  }
  if (n == (x.length - 1)) {
    document.getElementById("nextBtn").innerHTML = "Submit";
  } else {
    document.getElementById("nextBtn").innerHTML = "Next";
  }
  //... and run a function that will display the correct step indicator:
  fixStepIndicator(n)
}

function nextPrev(n) {
  // This function will figure out which tab to display
  var x = document.getElementsByClassName("tab");
  // Exit the function if any field in the current tab is invalid:
  if (n == 1 && !validateForm()) return false;
  // Hide the current tab:
  x[currentTab].style.display = "none";
  // Increase or decrease the current tab by 1:
  currentTab = currentTab + n;
  // if you have reached the end of the form...
  if (currentTab >= x.length) {
    // ... the form gets submitted:
    document.getElementById("regForm").submit();
    return false;
  }
  // Otherwise, display the correct tab:
```

```
  showTab(currentTab);
}

function validateForm() {
  // This function deals with validation of the form fields
  var x, y, i, valid = true;
  x = document.getElementsByClassName("tab");
  y = x[currentTab].getElementsByTagName("input");
  // A loop that checks every input field in the current tab:
  for (i = 0; i < y.length; i++) {
    // If a field is empty...
    if (y[i].value == "") {
      // add an "invalid" class to the field:
      y[i].className += " invalid";
      // and set the current valid status to false
      valid = false;
    }
  }
  // If the valid status is true, mark the step as finished and valid:
  if (valid) {
    document.getElementsByClassName("step")[currentTab].className += " finish";
  }
  return valid; // return the valid status
}

function fixStepIndicator(n) {
  // This function removes the "active" class of all steps...
  var i, x = document.getElementsByClassName("step");
  for (i = 0; i < x.length; i++) {
    x[i].className = x[i].className.replace(" active", "");
  }
```

```
  //... and adds the "active" class on the current step:
  x[n].className += " active";
}
</script>

</body>
</html>
```

**//Basic Validation program**

**//frmvalidateevents.js**

```
window.onload = init;
function init() {
   // Attach "onclick" handler to "reset" button
   document.getElementById("reset").onclick = clearDisplay;
   // Set initial focus
   document.getElementById("name").focus();
}
 //document.forms (Collection of all forms in the document)
 //document.forms[0] (Refers to the first form in the document)

//1.[index]        Returns the element in <form> with the specified index (starts at 0).
  //document.forms[0].elements[0].value;
  //document.forms.item(0).elements[0].value;
// document.getElementById("eform").elements[0].value;
//document.forms.namedItem("eform").elements[0].value;

//2.item(index)  Returns the element in <form> with the specified index (starts at 0).
//document.getElementById("eform").elements.item(0).value;

//3.namedItem(id)        Returns the element in <form> with the specified id.
```

```
//document.getElementById("eform").elements.namedItem("name").value;


function validateForm(thisForm) {
  with(thisForm) {
            //The with statement extends the scope chain for a statement
    return (isNotEmpty(name.value, name, "Please enter your name!", nameError)
        && isNumeric(zipcode.value, zipcode, "Please enter a 5-digit zip code!", zipcodeError)
        && isLengthMinMax(zipcode.value, 5, 5, zipcode, "Please enter a 5-digit zip code!",
zipcodeError)
        && isSelected(country.value, country, "Please make a selection!", countryError)
        && isChecked("gender", null, "Please check a gender!", genderError)
        && isChecked("color", null, "Please check a color!", colorError)
        && isNumeric(phone.value, phone, "Please enter a valid phone number!", phoneError)
        && isValidEmail(email.value, email, "Enter a valid email!", emailError)
        && isValidPassword(password.value, password, "Password shall be 6-8 characters!",
passwordError)
        && verifyPassword(password.value, pwVerified.value, pwVerified, "Different from new
password!", pwVerifiedError));
  }
}


function showMessageAndFocus(isValid, focusInputElm, errMsg, errElm) {


  if (!isValid) {
    // Show errMsg on errElm, if provided.
    if (errElm !== undefined && errElm !== null
        && errMsg !== undefined && errMsg !== null) {
      errElm.innerHTML = errMsg;
    }
    // Set focus on Input Element for correcting error, if provided.
    if (focusInputElm !== undefined && focusInputElm !== null) {
      focusInputElm.className = "error";
```

```
      focusInputElm.focus();
    }
  } else {
    // Clear previous error message on errElm, if provided.
    if (errElm !== undefined && errElm !== null) {
      errElm.innerHTML = "";
    }
    if (focusInputElm !== undefined && focusInputElm !== null) {
      focusInputElm.className = "";
    }
  }
}

/* Validate the inputValue is not empty (and not null). */
function isNotEmpty(inputValue, focusInputElm, errMsg, errElm) {

  var isValid = (inputValue !== null)
          && (inputValue.trim() !== "");
  showMessageAndFocus(isValid, focusInputElm, errMsg, errElm);
  return isValid;
}

/* Return true if the input value contains only digits (at least one) */
function isNumeric(inputValue, focusInputElm, errMsg, errElm) {
  var isValid = (inputValue !== null
          && inputValue.trim().match(/^\d+$/) !== null);
  showMessageAndFocus(isValid, focusInputElm, errMsg, errElm);
  return isValid;
}

/* Return true if the input value contains only letters (at least one) */
```

```
function isAlphabetic(inputValue, focusInputElm, errMsg, errElm) {
  var isValid = (inputValue !== null
          && inputValue.trim().match(/^[a-zA-Z]+$/) !== null) ;
  showMessageAndFocus(isValid, focusInputElm, errMsg, errElm);
  return isValid;
}

/* Return true if the input value contains only digits and letters (at least one) */
function isAlphanumeric(inputValue, focusInputElm, errMsg, errElm) {
  var isValid = (inputValue !== null
          && inputValue.trim().match(/^[0-9a-zA-Z]+$/) !== null);
  showMessageAndFocus(isValid, focusInputElm, errMsg, errElm);
  return isValid;
}

/* Return true if the input length is between minLength and maxLength */
function isLengthMinMax(inputValue, minLength, maxLength, focusInputElm, errMsg, errElm) {
  var inputValue = inputValue.trim();
  var isValid = (inputValue.length >= minLength) && (inputValue.length <= maxLength);
  showMessageAndFocus(isValid, focusInputElm, errMsg, errElm);
  return isValid;
}

// Return true if the input value is a valid email address
function isValidEmail(inputValue, focusInputElm, errMsg, errElm) {
 var isValid = (inputValue !== null)
       && (inputValue.trim().match(/^[-$]+\w+[-$]+@manipal\.(edu|in)$/) !== null);

  showMessageAndFocus(isValid, focusInputElm, errMsg, errElm);
  return isValid;
}
```

```
/* Return true if selection is made (not default of "") in <select> input */
function isSelected(inputValue, focusInputElm, errMsg, errElm) {
  // You need to set the default value of <select>'s <option> to "".
  var isValid = (inputValue !== "");
  showMessageAndFocus(isValid, focusInputElm, errMsg, errElm);
  return isValid;
}

/* Return true if the one of the checkboxes or radio buttons is checked
 * Need to check all elements of the "names" */
function isChecked(inputName, focusInputElm, errMsg, errElm) {
  var inputElements = document.getElementsByName(inputName);
  var isChecked = false;
  for (var i = 0; i < inputElements.length; i++) {
    if (inputElements[i].checked) {
      isChecked = true;  // found one element checked
      break;
    }
  }
  showMessageAndFocus(isChecked, focusInputElm, errMsg, errElm);
  return isChecked;
}


// Validate password, 6-8 characters of [a-zA-Z0-9_]
function isValidPassword(inputValue, focusInputElm, errMsg, errElm) {
  var isValid = (inputValue !== null)
          && (inputValue.trim().match(/^\w{6,8}$/) !== null);
  showMessageAndFocus(isValid, focusInputElm, errMsg, errElm);
  return isValid;
```

```
}



// Verify password.
function verifyPassword(pw, verifiedpw, focusInputElm, errMsg, errElm) {
  var isTheSame = (pw === verifiedpw);
  showMessageAndFocus(isTheSame, focusInputElm, errMsg, errElm);
  return isTheSame;
}

// The "onclick" handler for the "reset" button to clear the display
function clearDisplay() {
  var elms = document.getElementsByTagName("*");  // all tags
  for (var i = 0; i < elms.length; i++) {
    if ((elms[i].id).match(/Error$/)) {
      elms[i].innerHTML = "";
    }
    if (elms[i].className === "error") {  // assume only one class
      elms[i].className = "";
    }
  }
  // Set initial focus
  document.getElementById("name").focus();
}
```

**//validatcss.css**

```
/* for error messages */
.red {
  color: red;
}
```

```
/* for the error input text fields */
input.error {
  border: 1px red inset;
  padding: 2px;
}

table {
  border: 0;
}

td {
  margin: 0;
  padding: 3px 10px 3px 3px;
}
```

**//validform.html**

```
<!DOCTYPE html>
<html>
<head>
 <title>Test JavaScript Form Validation</title>
 <link rel="stylesheet" href="validatcss.css">
 <script type="text/javascript" src="frmvalidateevents.js" ></script>
</head>

<body>
 <h2>Test JavaScript Form Validataion</h2>

 <form method="get" action="" name="empfrm" id="eform" onsubmit="return validateForm(this)">
  <table>
  <tr>
```

```
    <td>Name<span class="red">*</span></td>
    <td>
     <input type="text" id="name" name="name" tabindex="1" placeholder="Enter" value="Default
Name"
                    />  
     <span id="nameError" class="red"></span>
    </td></tr>
   <tr>
    <td>Address</td>
    <td>
     <input type="text" id="address" name="address" tabindex="2" />  
     <span id="addressError" class="red"></span>
    </td></tr>
   <tr>
    <td>Zip Code<span class="red">*</span></td>
    <td>
     <input type="text" id="zipcode" name="zipcode" tabindex="3" />  
     <span id="zipcodeError" class="red"></span>
    </td></tr>
   <tr>
    <td>Country<span class="red">*</span></td>
    <td>
     <select id="country" name="country">
      <option value="" selected>Please select...</option>
      <option value="AA">AA</option>
      <option value="BB">BB</option>
      <option value="CC">CC</option>
     </select>  
     <span id="countryError" class="red"></span>
    </td></tr>
   <tr>
```

```
 <td>Gender<span class="red">*</span></td>
 <td>
  <input type="radio" name="gender" value="m" />Male
  <input type="radio" name="gender" value="f" />Female   
  <span id="genderError" class="red"></span>
 </td></tr>
<tr>
 <td>Preferences<span class="red">*</span></td>
 <td>
  <input type="checkbox" name="color" value="r" />Red
  <input type="checkbox" name="color" value="g" />Green
  <input type="checkbox" name="color" value="b" />Blue   
  <span id="colorError" class="red"></span>
 </td></tr>
<tr>
 <td>Phone<span class="red">*</span></td>
 <td>
  <input type="text" id="phone" name="phone" />  
  <span id="phoneError" class="red"></span>
 </td></tr>
<tr>
 <td>Email<span class="red">*</span></td>
 <td>
  <input type="text" id="email" name="email" />
  <span id="emailError" class="red"></span>
 </td></tr>
<tr>
 <td>password (6-8 characters)<span class="red">*</span></td>
 <td>
  <input type="password" id="password" name="password" />  
  <span id="passwordError" class="red"></span>
```

```
    </td></tr>
   <tr>
    <td>Verify password<span class="red">*</span></td>
    <td>
     <input type="password" id="pwVerified" name="pwVerified" />  
     <span id="pwVerifiedError" class="red"></span>
    </td></tr>
   <tr>
    <td> </td>
    <td>
     <input type="submit" value="SEND" id="submit"/> 
     <input type="reset" value="CLEAR" id="reset"/>  
    </td></tr>
   </table>
  </form>
</body>
</html>
```