

APP DEVELOPMENT MANUAL

PRACTICAL EXERCISES:

1. Using react native, build a cross platform application for a BMI calculator.
2. Build a cross platform application for a simple expense manager which allows entering expenses and income on each day and displays category wise weekly income and expense.
3. Develop a cross platform application to convert units from imperial system to metric system (km to miles, kg to pounds etc.,)
4. Design and develop a cross platform application for day to day task (to-do) management.
5. Design an android application using Cordova for a user login screen with username, password, reset button and a submit button. Also, include header image and a label. Use layout managers.
6. Design and develop an android application using Apache Cordova to find and display the current location of the user.
7. Write programs using Java to create Android application having Databases
 - For a simple library application.
 - For displaying books available, books lend, book reservation. Assume that student information is available in a database which has been stored in a database server.

Ex1 Using react native, build a cross platform application for a BMI Calculator.

Aim:

To develop a cross platform application for a BMI Calculator using react native JS.

Algorithm:

STEP1: Create index file in HTML to define the structure of application

STEP2: To build that application as appealing and user friendly style the sheet by using cascading style sheet.

STEP3: To compute the Body Mass Index value develop a JavaScript file for Back-end and computation.

STEP4: Run the application in web browser.

STEP5: Get the input from the user such as weight and height of a person.

STEP6: Calculate BMI Value.

Index.HTML:

```
<html>

<head>

    <title>BMI Calculator</title>

    <meta name="viewport" content="width=device-width, initial-scale=1.0, user-
scalable=no">

    <link rel="stylesheet" type="text/css" href="stylesheet.css">

    <link href="https://fonts.googleapis.com/css?family=Open+Sans" rel="stylesheet">

</head>

<body>

    <div class="container">

        <div class="panel">

            <h2 class="text-center">Check your BMI</h2>

            <p id="introText" class="text-center">Enter your weight and height below to
check your BMI results</p>

            <form>

                <div id="weightInput">

                    <p>Put your weight in here (KG)</p>

                    <input id="weight" type="number" pattern="[0-9]*" name="a" />

                </div>

                <div id="heightInput">

                    <p>And your height in here (CM)</p>

                    <input id="height" type="number" pattern="[0-9]*" name="b"/>

                </div>

                <button type="button" class="btn" onclick="calculate()">Calculate BMI</button>

            </form>

            <div id="results" class="text-center">Your BMI results will appear here</div>

        </div>

    </div>

    <script src="main.js"></script>
```

```
</body></html>
```

Main.JS:

```
var weight, height, measure, bmi, error ;
```

```
function calculate() {  
    weight = document.getElementById("weight").value;  
    height = document.getElementById("height").value;  
    error = "Please enter some values";  
    height /= 100;  
    height *= height;  
    bmi = weight/height;  
    bmi = bmi.toFixed(1);  
    if (bmi <= 18.4) {  
        measure = "Your BMI is " + bmi + " which means " + "you are Underweight";  
    } else if (bmi >= 18.5 && bmi <= 24.9) {  
        measure = "Your BMI is " + bmi + " which means " + "You are Normal";  
    } else if (bmi >= 25 && bmi <= 29.9) {  
        measure = "Your BMI is " + bmi + " which means " + "You are Overweight";  
    } else if (bmi >= 30) {  
        measure = "Your BMI is " + bmi + " which means " + "You are Obese";  
    }  
    if (weight === 0 ) {  
        document.getElementById("results").innerHTML = error;  
    } else if (height === 0){  
        document.getElementById("results").innerHTML = error;  
    }  
    else {  
        document.getElementById("results").innerHTML = measure;  
    }  
    if (weight < 0) {  
        document.getElementById("results").innerHTML = "Negative Values not Allowed";  
    }  
}
```

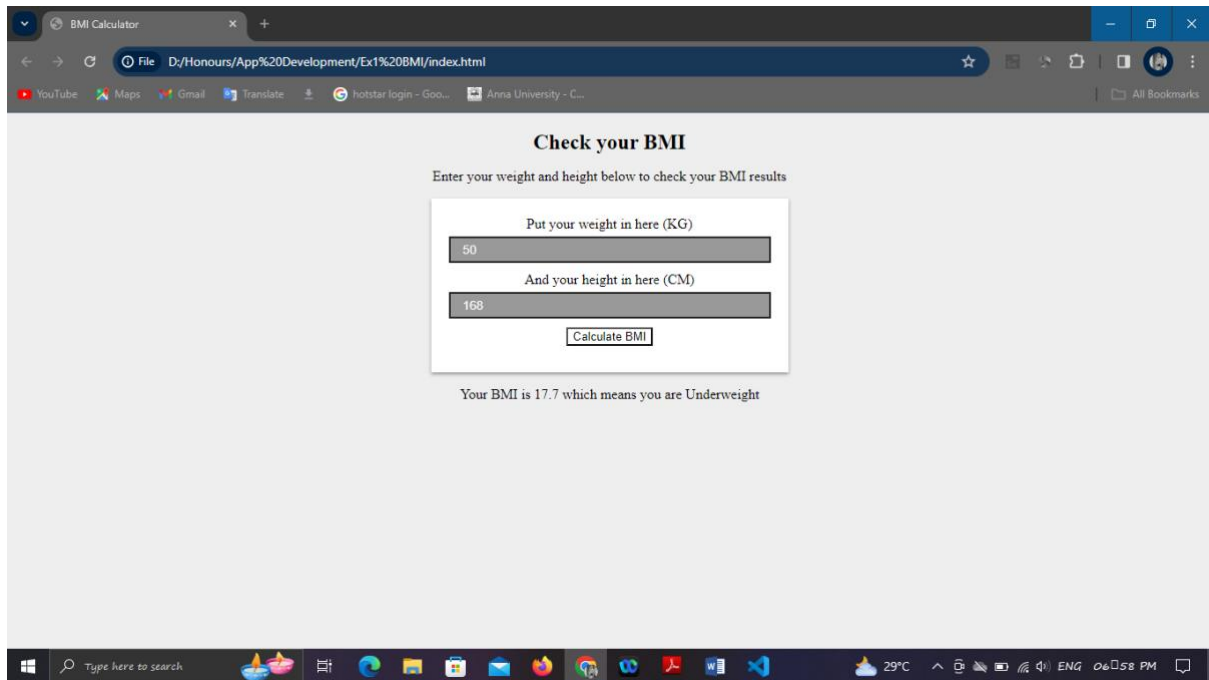
```
}}
```

StyleSheet.CSS:

```
body {  
    background-color: #eee;  
    text-align: center;  
}button {  
    background-color: rgba(0,0,0,0);  
    outline: none;}  
h2 {  
    margin-bottom: 10px;  
}  
.btn {  
    margin-bottom: 10px;  
    margin-left: auto;  
    margin-right: auto;  
    display: block;  
}input {  
    width: 100%;  
    padding: 5px 14px;  
    border: 2px solid #201f1d;  
    margin-bottom: 10px;  
    background-color: rgba(0,0,0,.4);  
    outline: none;  
    color: #fff;  
    font-size: 14px;  
}input:focus {  
    border: 2px solid rgba(0,0,0,.4);  
    background-color: rgb(60,60,60);  
}form {
```

```
background-color: #fff;
padding: 20px;
color: #000;
box-shadow: 0 2px 4px 0 rgba(0,0,0,.4);
}form p {
margin-bottom: 5px;
}form p:nth-child(1){
margin-top: 0;
}.panel {
margin-top: 20px;
float: none;
}@media (min-width: 768px) {
.panel {
width: 50%;
margin-left: auto;
margin-right: auto;
}}
@media (min-width: 1024px) {
.panel {
width: 30%;
margin-left: auto;
margin-right: auto;
}}
```

Output:



The screenshot shows a web browser window with the title "BMI Calculator". The address bar displays the file path "D:/Honours/App%20Development/Ex1%20BMI/index.html". The browser's bookmark bar includes links to YouTube, Maps, Gmail, Translate, hotstar login - Goo..., and Anna University - C... The main content area of the browser displays a web application titled "Check your BMI". Below the title is a subtitle "Enter your weight and height below to check your BMI results". The application features a white rectangular form with two input fields: "Put your weight in here (KG)" with the value "50" and "And your height in here (CM)" with the value "168". A "Calculate BMI" button is positioned below the height input field. Below the form, the text "Your BMI is 17.7 which means you are Underweight" is displayed. The Windows taskbar at the bottom shows the search bar, task view button, and several application icons, including Chrome, Word, and Excel. The system tray on the right indicates a temperature of 29°C, network status, and the time 06:58 PM.

Check your BMI

Enter your weight and height below to check your BMI results

Put your weight in here (KG)

50

And your height in here (CM)

168

Calculate BMI

Your BMI is 17.7 which means you are Underweight

Result:

Thus the application to calculate BMI value was executed and verified successfully.

Ex2 Build a cross platform application for a simple expense manager which allows entering expenses and income on each day and displays category wise weekly income and expense.

Aim:

To develop a cross platform application for simple expense manager using JavaScript Frameworks.

Algorithm:

STEP1: Develop an index file for defining the structure of application.

STEP2: To make an application as appealing one style the sheets by using CSS file.

STEP3: Use MangoDB database to store the expense details.

STEP4: Run the application in any web browser.

STEP5: Get the input from the user.

STEP6: Calculate income and expense and display category wise.

STEP7: Output will be displayed.

Source Code:

Index.html:

```
<!DOCTYPE html>

<html lang="en">

  <head>

    <meta charset="UTF-8" />

    <meta name="viewport" content="width=device-width, initial-scale=1.0" />

    <meta http-equiv="X-UA-Compatible" content="ie=edge" />

    <link rel="stylesheet" href="style.css" />

    <title>Expense Tracker</title>

  </head>

  <body>

    <h2>Expense Tracker</h2>

    <div class="container">

      <h4>Your Balance</h4>

      <h1 id="balance">$0.00</h1>

    <div class="inc-exp-container">

      <div>

        <h4>Income</h4>

        <p id="money-plus" class="money plus">+$0.00</p>

      </div>

      <div>

        <h4>Expense</h4>

        <p id="money-minus" class="money minus">-$0.00</p>

      </div>

    </div>

  </body>

</html>
```

```
</div>
<h3>History</h3>
<ul id="list" class="list">
  <!-- <li class="minus">
    Cash <span>-$400</span><button class="delete-btn">x</button>
  </li> -->
</ul>
<h3>Add new transaction</h3>
<form id="form">
  <div class="form-control">
    <label for="text">Text</label>
    <input type="text" id="text" placeholder="Enter text..." />
  </div>
  <div class="form-control">
    <label for="amount">
      >Amount <br />
      (negative - expense, positive - income)</label>
    >
    <input type="number" id="amount" placeholder="Enter amount..." />
  </div>
  <button class="btn">Add transaction</button>
</form>
</div>
<script src="script.js"></script>
</body>
</html>
```

Script.js:

```
const balance = document.getElementById('balance');
const money_plus = document.getElementById('money-plus');
const money_minus = document.getElementById('money-minus');
const list = document.getElementById('list');
const form = document.getElementById('form');
const text = document.getElementById('text');
const amount = document.getElementById('amount');

// const dummyTransactions = [
//   { id: 1, text: 'Flower', amount: -20 },
//   { id: 2, text: 'Salary', amount: 300 },
//   { id: 3, text: 'Book', amount: -10 },
//   { id: 4, text: 'Camera', amount: 150 }
// ];

const localStorageTransactions = JSON.parse(
  localStorage.getItem('transactions')
);

let transactions =
  localStorage.getItem('transactions') !== null ? localStorageTransactions : [];

// Add transaction
function addTransaction(e) {
  e.preventDefault();

  if (text.value.trim() === '' || amount.value.trim() === '') {
    alert('Please add a text and amount');
  } else {
    const transaction = {
      id: generateID(),
```

```

        text: text.value,
        amount: +amount.value
    };
    transactions.push(transaction);
    addTransactionDOM(transaction);
    updateValues();
    updateLocalStorage();
    text.value = "";
    amount.value = "";
}
} // Generate random ID
function generateID() {
    return Math.floor(Math.random() * 1000000000);
} // Add transactions to DOM list
function addTransactionDOM(transaction) {
    // Get sign
    const sign = transaction.amount < 0 ? '-' : '+';
    const item = document.createElement('li');
    // Add class based on value
    item.classList.add(transaction.amount < 0 ? 'minus' : 'plus');
    item.innerHTML = `
        ${transaction.text} <span>${sign}${Math.abs(
            transaction.amount
        )}</span> <button class="delete-btn" onclick="removeTransaction(${
            transaction.id
        })">x</button>
    `;

```

```

    list.appendChild(item);
  } // Update the balance, income and expense
  function updateValues() {
    const amounts = transactions.map(transaction => transaction.amount);
    const total = amounts.reduce((acc, item) => (acc += item), 0).toFixed(2);
    const income = amounts
      .filter(item => item > 0)
      .reduce((acc, item) => (acc += item), 0)
      .toFixed(2);
    const expense = (
      amounts.filter(item => item < 0).reduce((acc, item) => (acc += item), 0) *
      -1
    ).toFixed(2);
    balance.innerText = `$$ {total}`;
    money_plus.innerText = `$$ {income}`;
    money_minus.innerText = `$$ {expense}`;
  } // Remove transaction by ID
  function removeTransaction(id) {
    transactions = transactions.filter(transaction => transaction.id !== id);
    updateLocalStorage();
    init();
  } // Update local storage transactions
  function updateLocalStorage() {
    localStorage.setItem('transactions', JSON.stringify(transactions));
  }

```

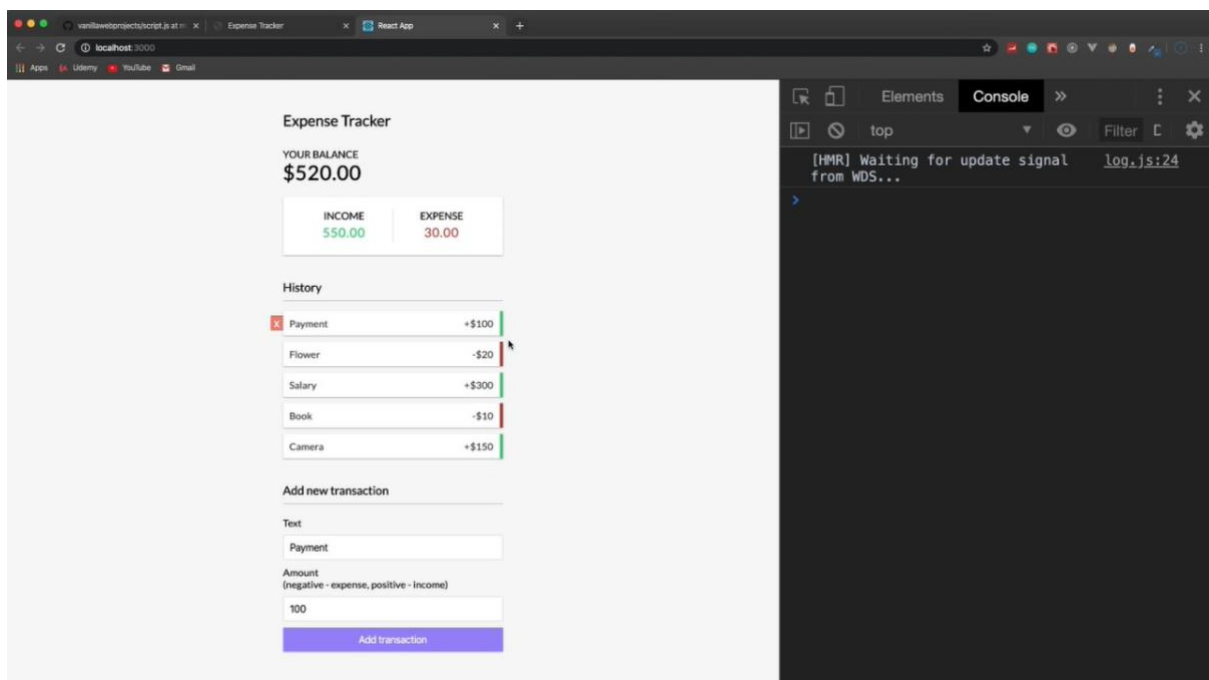
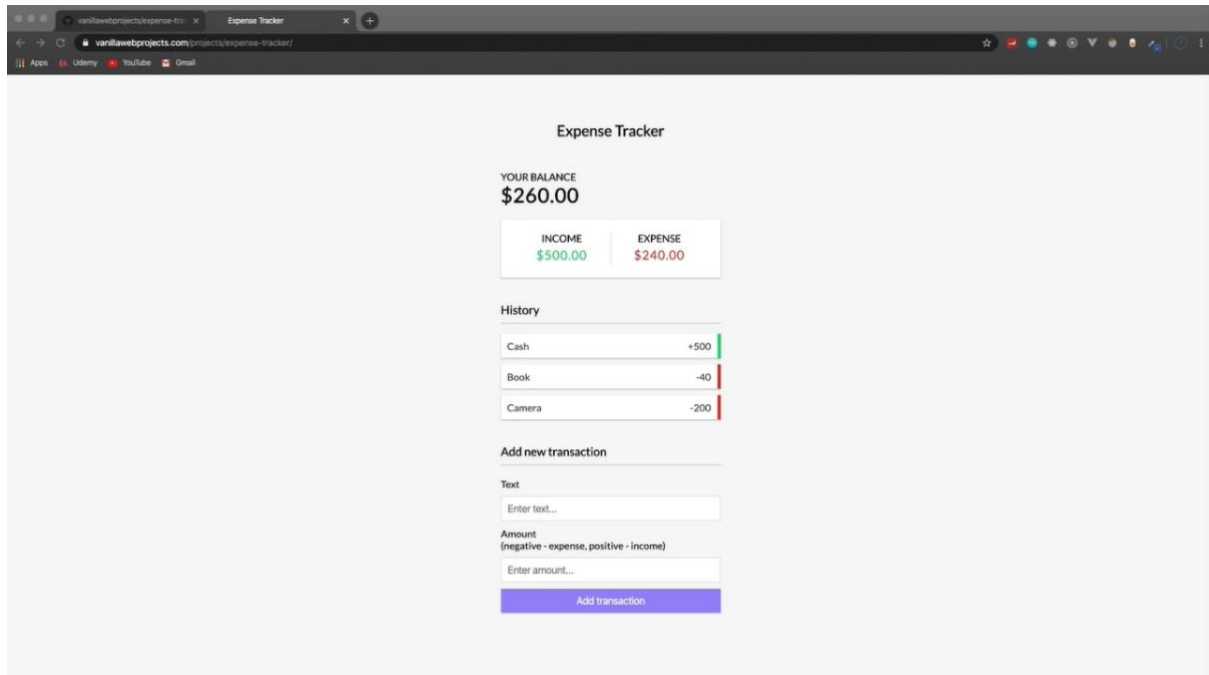
```
// Init app
function init() {
  list.innerHTML = "";
  transactions.forEach(addTransactionDOM);
  updateValues();
} init();
form.addEventListener('submit', addTransaction);
```

Style.css:

```
@import url('https://fonts.googleapis.com/css?family=Lato&display=swap');
:root {
  --box-shadow: 0 1px 3px rgba(0, 0, 0, 0.12), 0 1px 2px rgba(0, 0, 0, 0.24);
} * {
  box-sizing: border-box;
} body {
  background-color: #f7f7f7;
  display: flex;
  flex-direction: column;
  align-items: center;
  justify-content: center;
  min-height: 100vh;
  margin: 0;
  font-family: 'Lato', sans-serif;
} .container {
  margin: 30px auto;
  width: 350px;
} h1 {
```

```
letter-spacing: 1px;
margin: 0;
}h3 {
border-bottom: 1px solid #bbb;
padding-bottom: 10px;
margin: 40px 0 10px;
}h4 {
margin: 0;
text-transform: uppercase;
}.inc-exp-container {
background-color: #fff;
box-shadow: var(--box-shadow);
padding: 20px;
display: flex;
justify-content: space-between;
margin: 20px 0;
}.inc-exp-container > div {
flex: 1;
text-align: center;
}
```


Output:



Result:

Thus the application to manage the simple expenses was executed and verified successfully.

Ex3 Develop a cross platform application to convert units from imperial system to metric system (km to miles, kg to pounds etc.,)

Aim:

To build a cross platform application to convert units from imperial system to metric system using HTML, CSS, JS.

Algorithm:

STEP1: Develop an index file for defining the structure of application.

STEP2: To make an application as appealing one style the sheets by using CSS file.

STEP3: To compute the conversion between imperial and metric system develop a JavaScript file.

STEP4: Run the application in any web browser.

STEP5: Get the input from the user.

STEP6: Calculate the value.

STEP7: Output will be displayed.

SOURCE CODE:

Index.html:

```
<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

  <meta http-equiv="X-UA-Compatible" content="IE=edge">

  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <title>Unit Converter</title>

  <link rel="stylesheet"
href="https://cdnjs.cloudflare.com/ajax/libs/normalize/8.0.1/normalize.css">

  <link rel="stylesheet" href="index.css">

  <link rel="preconnect" href="https://fonts.googleapis.com">

  <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>

  <link
href="https://fonts.googleapis.com/css2?family=Inter:wght@400;600;800&display=swap"
rel="stylesheet">

</head>

<body>

  <main>

    <div class="up-section">

      <h1>Metric/Imperial Unit Conversion</h1>

      <div class="input-section">

        <input type="number" id="input-el">

      </div>

      <div class="button-container">

        <div class="input-section">

          <button id="convert-btn">Convert</button>
```

```
<div class ="separator"></div>

<button id="reset-btn">Reset</button>

</div>

</div>

<div class ="down-section">
  <div class = "conversion">
    <h3>Length (Meter/Feet)</h3>
    <p id = "convert-text-1"></p>
  </div>

  <div class = "conversion">
    <h3>Volume (Liters/Gallons)</h3>
    <p id = "convert-text-2"></p>
  </div>

  <div class = "conversion">
    <h3>Mass (Kilograms/Pounds)</h3>
    <p id = "convert-text-3"></p>
  </div>
</div>

</main>

<script src="index.js"></script>

<footer class = "copyright" >
  <p>
    @ 2022
    <a href="https://github.com/vinh-nguyen-code" target="_blank">Vinh NGUYEN</a>
  </p>
</footer>

</body>

</html>
```

Index.JS:

```

const convertBtn = document.getElementById("convert-btn")
const resetBtn = document.getElementById("reset-btn")

const inputEl = document.getElementById("input-el")
const convertText1 = document.getElementById("convert-text-1")
const convertText2 = document.getElementById("convert-text-2")
const convertText3 = document.getElementById("convert-text-3")
convertBtn.addEventListener("click",function(){
  if(inputEl.value !== ""){
    convertText1.innerHTML =
      `${inputEl.value} meters = ${(inputEl.value*3.2808).toFixed(3)} feet | ${inputEl.value}
      feet = ${(inputEl.value/3.2808).toFixed(3)} meters
    `
    convertText2.innerHTML =
      `${inputEl.value} liters = ${(inputEl.value*0.264172).toFixed(3)} gallons |
      ${inputEl.value} gallons = ${(inputEl.value/0.264172).toFixed(3)} liters
    `
    convertText3.innerHTML = `${inputEl.value} kilos =
    ${(inputEl.value*2.20462).toFixed(3)} pounds | ${inputEl.value} pounds =
    ${(inputEl.value/2.20462).toFixed(3)} kilos
    `
  })
resetBtn.addEventListener("click",function(){
  window.location.reload();
})

```

Index.CSS:

```

body {
  background: #1C1C1C;
  font-family: 'Inter', sans-serif;
  font-size: 14px;
}main{
  display:flex;
  flex-direction: column;
  justify-content: center;
  align-items:center;

```

```
    margin-top: 5vh;
}.copyright{
    font-size: 0.8em;
    text-align: center;
    color:white
}.copyright a {
    color:white
}.copyright a:hover {
    color:lightgreen
}.up-section{
    background: #6943FF;
    width: 500px;
    height: 250px;
    display: grid;
    grid-template-rows: 1fr 1fr 1fr;
}h1 {
    color:white;
    font-size: 24px;
    text-align: center;
    line-height: 60px;
    margin-bottom: 0;
}.input-section{
    display: flex;
    min-width: 0;
    justify-content: center;
    align-items: center;
    margin:0
}input{
    background: transparent;
    border: solid 2px #B295FF;
```

```
border-radius: 5px;
box-sizing: border-box;
font-weight: 800;
font-size: 58px;
color:white;
text-align: center;
width:120px;
}input::-webkit-outer-spin-button,
input::-webkit-inner-spin-button {
-webkit-appearance: none;
margin: 0;
}
```

```
.button-container{
display: flex;
justify-content: center;
align-items: center;
}
button{
background: white;
padding: 9px 27px ;
border: none;
box-shadow: 0px 1px 2px rgba(0, 0, 0, 0.05);
border-radius: 5px;
}button:hover{
cursor: pointer;
}button:active{
transform: scale(1.02);
transition: 0.04s;
}
```

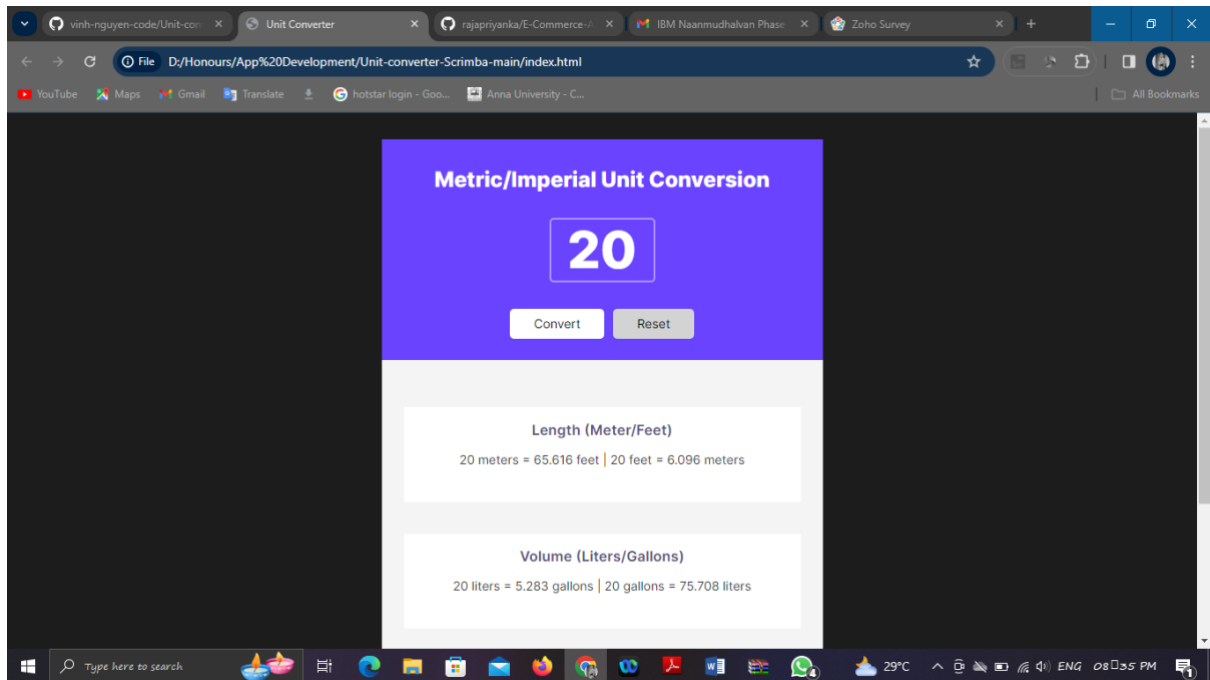


```
.separator{
    width: 10px;
}

#reset-btn{
    background: lightgray;
}.down-section{
background: #F4F4F4;
width: 500px;
height: 430px;
display: flex;
padding-top: 35px;
padding-bottom: 35px;
flex-direction: column;
justify-content:space-around ;
align-items: center
}

.conversion{
    background: white;
    width: 90%;
    height: 25%;
    text-align: center;
}.conversion h3{
    color: #5A537B;
    font-weight: 600;
}.conversion p{
    color: #353535;
}
```

Output:



Result:

Thus the application to convert from imperial to metric system was executed and verified successfully.

Ex4 Design and develop a cross platform application for day to day task (to-do) management.

Aim:

To build an application for day to day task management.

Algorithm:

STEP1: Develop an index file for defining the structure of application.

STEP2: To make an application as appealing one style the sheets by using CSS file.

STEP3: To store the to do list in a day to day life, develop a JavaScript file.

STEP4: Run the application in any web browser.

STEP5: Get the input from the user.

STEP6: Store the activities which are need to be.

STEP7: Output will be displayed.

SOURCE CODE:

Index.html:

```
<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <title>Document</title>

  <link rel="stylesheet" href="index.css">

</head>

<body>

  <div id="myDIV" class="header">

    <h2>My To Do List</h2>

    <input type="text" id="myInput" placeholder="Title...">

    <span onclick="newElement()" class="addBtn">Add</span>

  </div>

  <ul id="myUL">

    <li>Hit the gym</li>

    <li class="checked">Pay bills</li>

    <li>Meet George</li>

    <li>Buy eggs</li>

    <li>Read a book</li>

    <li>Organize office</li>

  </ul>

  <script>

    // Create a "close" button and append it to each list item

var myNodelist = document.getElementsByTagName("LI");
var i;
for (i = 0; i < myNodelist.length; i++) {
```

```

var span = document.createElement("SPAN");
var txt = document.createTextNode("\u00D7");
span.className = "close";
span.appendChild(txt);
myNodelist[i].appendChild(span);
}

// Click on a close button to hide the current list item
var close = document.getElementsByClassName("close");
var i;
for (i = 0; i < close.length; i++) {
    close[i].onclick = function() {
        var div = this.parentElement;
        div.style.display = "none";
    }
}

// Add a "checked" symbol when clicking on a list item
var list = document.querySelector('ul');
list.addEventListener('click', function(ev) {
    if (ev.target.tagName === 'LI') {
        ev.target.classList.toggle('checked');
    }
}, false);

// Create a new list item when clicking on the "Add" button
function newElement() {
    var li = document.createElement("li");
    var inputValue = document.getElementById("myInput").value;
    var t = document.createTextNode(inputValue);
    li.appendChild(t);
    if (inputValue === "") {
        alert("You must write something!");
    }
}

```

```

    } else {
        document.getElementById("myUL").appendChild(li);
    }
    document.getElementById("myInput").value = "";
    var span = document.createElement("SPAN");
    var txt = document.createTextNode("\u00D7");
    span.className = "close";
    span.appendChild(txt);
    li.appendChild(span);
    for (i = 0; i < close.length; i++) {
        close[i].onclick = function() {
            var div = this.parentElement;
            div.style.display = "none";
        }
    }
}
</script>
</body>
</html>

```

Index.CSS:

```

/* Include the padding and border in an element's total width and height */
* {
    box-sizing: border-box;
}/* Remove margins and padding from the list */
ul {
    margin: 0;
    padding: 0;
}/* Style the list items */
ul li {

```

```
cursor: pointer;
position: relative;
padding: 12px 8px 12px 40px;
background: #eee;
font-size: 18px;
transition: 0.2s;
/* make the list items unselectable */
-webkit-user-select: none;
-moz-user-select: none;
-ms-user-select: none;
user-select: none;
}/* Set all odd list items to a different color (zebra-stripes) */
ul li:nth-child(odd) {
    background: #f9f9f9;
}/* Darker background-color on hover */
ul li:hover {
    background: #ddd;
}/* When clicked on, add a background color and strike out text */
ul li.checked {
    background: #888;
    color: #fff;
    text-decoration: line-through;
}/* Add a "checked" mark when clicked on */
ul li.checked::before {
    content: "";
    position: absolute;
    border-color: #fff;
    border-style: solid;
    border-width: 0 2px 2px 0;
    top: 10px;
```



```
left: 16px;

transform: rotate(45deg);

height: 15px;

width: 7px;

}/* Style the close button */

.close {

  position: absolute;

  right: 0;

  top: 0;

  padding: 12px 16px 12px 16px;

}.close:hover {

  background-color: #f44336;

  color: white;

}/* Style the header */

.header {

  background-color: #f44336;

  padding: 30px 40px;

  color: white;

  text-align: center;

}/* Clear floats after the header */

.header:after {

  content: "";

  display: table;

  clear: both;

}/* Style the input */

input {

  margin: 0;

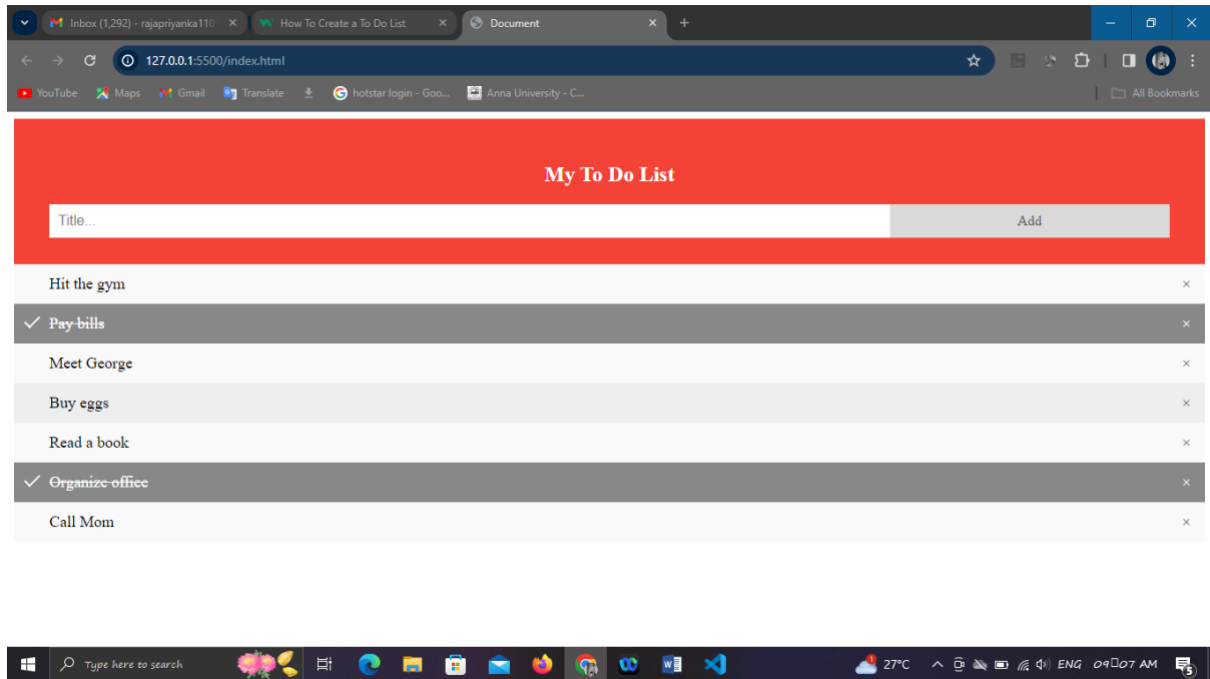
  border: none;

  border-radius: 0;

  width: 75%;
```

```
padding: 10px;
float: left;
font-size: 16px;
}/* Style the "Add" button */
.addBtn {
padding: 10px;
width: 25%;
background: #d9d9d9;
color: #555;
float: left;
text-align: center;
font-size: 16px;
cursor: pointer;
transition: 0.3s;
border-radius: 0;
}.addBtn:hover {
background-color: #bbb;
}
```

Output:



.

Result:

Thus the application to-do list was executed and verified successfully.

Ex5 Design an android application using Cordova for a user login screen with username, password, reset button and a submit button. Also, include header image and a label. Use layout managers.

Aim:

To build an android application for a user login screen by using cordova.

Algorithm:

STEP1: Develop an index file for defining the structure of application.

STEP2: To make an application as appealing one style the sheets by using CSS file.

STEP3: Use MangoDB database to store the users details.

STEP4: Run the application in any web browser.

STEP5: Get the input from the user.

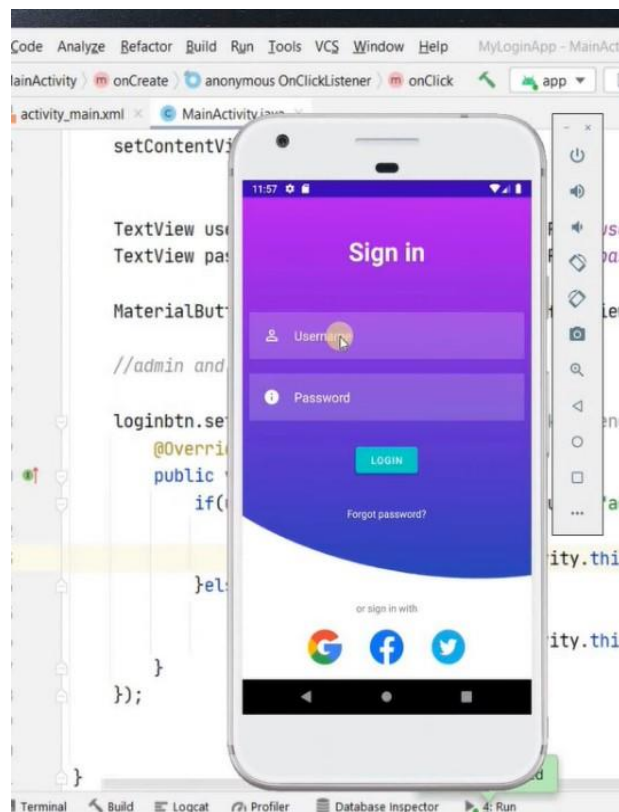
STEP6: Output will be displayed.

Source Code:

Android.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.myloginapp">
<application
    android:allowBackup="true"
    android:icon="@mipmap/ic_launcher"
    android:label="@string/app_name"
    android:roundIcon="@mipmap/ic_launcher_round"
    android:supportsRtl="true"
    android:theme="@style/Theme.MyLoginApp">
    <activity android:name=".MainActivity">
        <intent-filter>
            <action android:name="android.intent.action.MAIN" />
            <category android:name="android.intent.category.LAUNCHER" />
        </intent-filter>
    </activity>
</application>
</manifest>
```

Output:



Result:

Thus the application to create a simple user login was executed and verified successfully.

Ex6 Design and develop an android application using Apache Cordova to find and display the current location of the user.

Aim:

To build an android application find and the display the current location of the user using Apache Cordova.

Algorithm:

STEP1: Develop an xml file for defining the structure of application.

STEP2: To make a responsive page use JavaScript file.

STEP3: Run the application in any web browser.

STEP4: Get the input from the user.

STEP5: Output will be displayed.

Source Code:

Activity.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:gravity="center"
    android:orientation="vertical"
    tools:context=".MainActivity">
    <TextView
        android:id="@+id/addressText"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="User Address"
        android:textColor="@android:color/black"
        android:textSize="19sp" />
    <Button
        android:id="@+id/locationButton"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="16dp"
        android:text="Get Current Location"
        android:textAllCaps="false"
        android:textSize="19sp" />
</LinearLayout>
```

Android Manifest.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.technical.myapplication">

    <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION"/>
    <uses-permission android:name="android.permission.INTERNET"/>

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
            </intent-filter>
            <category android:name="android.intent.category.LAUNCHER" />
        </activity>
    </application>
</manifest>
```

Main Activity.java:

```
package com.technical.myapplication;
import androidx.annotation.NonNull;
import androidx.annotation.Nullable;
import androidx.appcompat.app.AppCompatActivity;
import androidx.core.app.ActivityCompat;
import android.Manifest;
import android.app.Activity;
import android.content.Context;
import android.content.Intent;
import android.content.IntentSender;
import android.content.pm.PackageManager;
import android.location.LocationManager;
import android.os.Build;
import android.os.Bundle;
import android.os.Looper;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;
import android.widget.Toast;
import com.google.android.gms.common.api.ApiException;
import com.google.android.gms.common.api.ResolvableApiException;
import com.google.android.gms.location.LocationCallback;
import com.google.android.gms.location.LocationRequest;
import com.google.android.gms.location.LocationResult;
import com.google.android.gms.location.LocationServices;
import com.google.android.gms.location.LocationSettingsRequest;
import com.google.android.gms.location.LocationSettingsResponse;
import com.google.android.gms.location.LocationSettingsStatusCodes;
import com.google.android.gms.tasks.OnCompleteListener;
import com.google.android.gms.tasks.Task;
public class MainActivity extends AppCompatActivity {
    private TextView AddressText;
    private Button LocationButton;
    private LocationRequest locationRequest;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        AddressText = findViewById(R.id.addressText);
        LocationButton = findViewById(R.id.locationButton);
        locationRequest = LocationRequest.create();
        locationRequest.setPriority(LocationRequest.PRIORITY_HIGH_ACCURACY);
        locationRequest.setInterval(5000);
        locationRequest.setFastestInterval(2000);
        LocationButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
```

```

        getLocation();
    }
}); }
@Override
public void onRequestPermissionsResult(int requestCode, @NonNull String[] permissions,
@NonNull int[] grantResults) {
    super.onRequestPermissionsResult(requestCode, permissions, grantResults);
    if (requestCode == 1){
        if (grantResults[0] == PackageManager.PERMISSION_GRANTED){
if (isGPSEnabled()) {
            getLocation();
        }else {
            turnOnGPS();
        }
    }
}
} }

@Override
protected void onActivityResult(int requestCode, int resultCode, @Nullable Intent data) {
    super.onActivityResult(requestCode, resultCode, data);
    if (requestCode == 2) {
        if (resultCode == Activity.RESULT_OK) {getLocation();
        }
    }
} private void getLocation() {
    if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.M) {
        if (ActivityCompat.checkSelfPermission(MainActivity.this,
Manifest.permission.ACCESS_FINE_LOCATION) ==
PackageManager.PERMISSION_GRANTED) {
            if (isGPSEnabled()) {

                LocationServices.getFusedLocationProviderClient(MainActivity.this)
                    .requestLocationUpdates(locationRequest, new LocationCallback() {
                        @Override
                        public void onLocationResult(@NonNull LocationResult
locationResult) {
                            super.onLocationResult(locationResult);
                            LocationServices.getFusedLocationProviderClient(MainActivity.this)
                                .removeLocationUpdates(this);

                            if (locationResult != null && locationResult.getLocations().size()
>0){

                                int index = locationResult.getLocations().size() - 1;
                                double latitude =
locationResult.getLocations().get(index).getLatitude();
                                double longitude =
locationResult.getLocations().get(index).getLongitude();

```

```

        AddressText.setText("Latitude: "+ latitude + "\n" +
"Longitude: "+ longitude);
    }
    }, Looper.getMainLooper());

    } else {
        turnOnGPS();
    }

    } else {
        requestPermissions(new
String[]{Manifest.permission.ACCESS_FINE_LOCATION}, 1);
    }
}

private void turnOnGPS() {
LocationSettingsRequest.Builder builder = new LocationSettingsRequest.Builder()
    .addLocationRequest(locationRequest);
    builder.setAlwaysShow(true);
    Task<LocationSettingsResponse> result =
    LocationServices.getSettingsClient(getApplicationContext())
        .checkLocationSettings(builder.build());
    result.addOnCompleteListener(new OnCompleteListener<LocationSettingsResponse>()
    {
        @Override
        public void onComplete(@NonNull Task<LocationSettingsResponse> task) {
    try {
        LocationSettingsResponse response = task.getResult(ApiException.class);
        Toast.makeText(MainActivity.this, "GPS is already tured on",
Toast.LENGTH_SHORT).show();
    } catch (ApiException e) {
        switch (e.getStatusCode()) {
            case LocationSettingsStatusCodes.RESOLUTION_REQUIRED:
                try {
                    ResolvableApiException resolvableApiException =
(ResolvableApiException) e;
                    resolvableApiException.startResolutionForResult(MainActivity.this, 2);
                } catch (IntentSender.SendIntentException ex) {
                    ex.printStackTrace();
                }
                break;
            case LocationSettingsStatusCodes.SETTINGS_CHANGE_UNAVAILABLE:
                //Device does not have location
                break;
        }
    }
}
}
}
}
}

private boolean isGPSEnabled() {
    LocationManager locationManager = null;
    boolean isEnabled = false;

```

```

        if (locationManager == null) {
            locationManager = (LocationManager)
getSystemService(Context.LOCATION_SERVICE);
        }

        isEnabled = locationManager.isProviderEnabled(LocationManager.GPS_PROVIDER);
        return isEnabled;
    }}

```

Build.gradle:

```

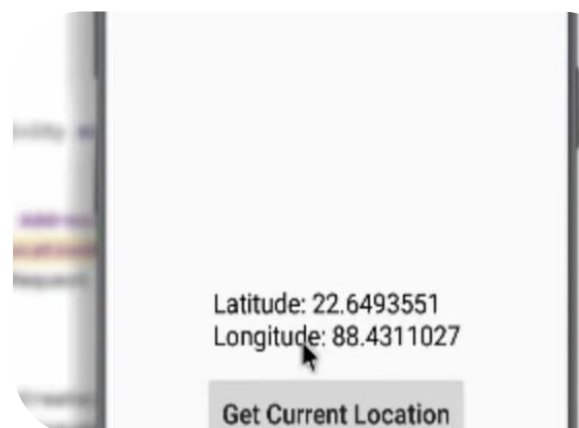
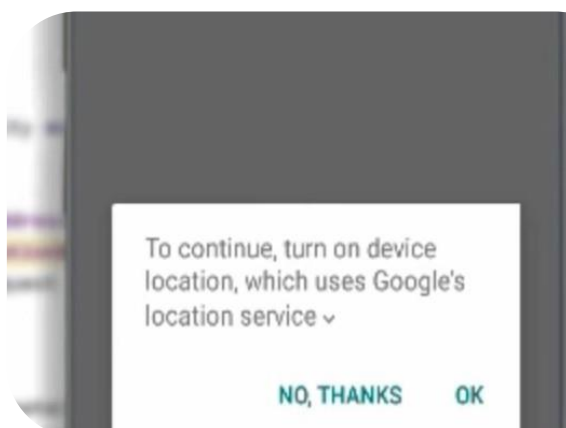
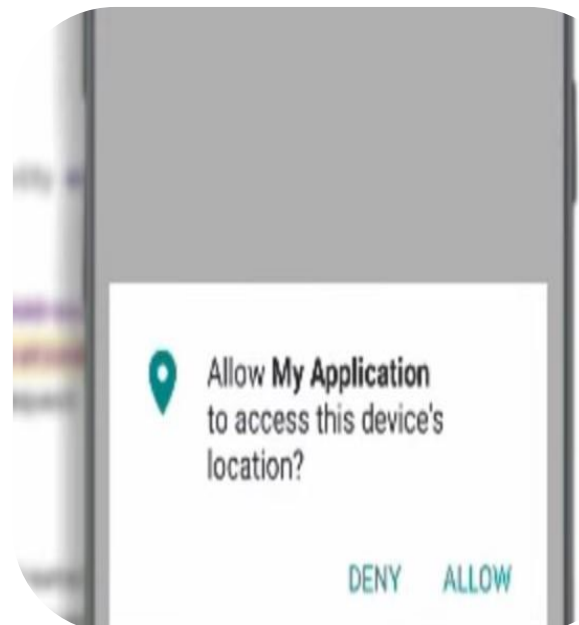
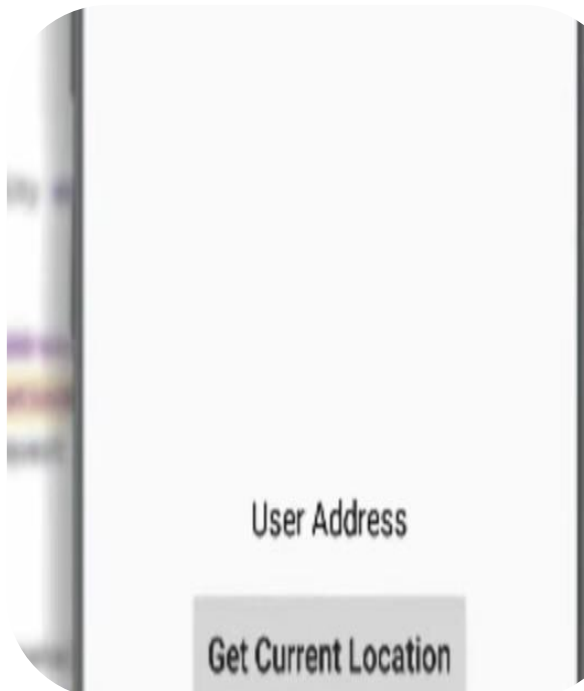
apply plugin: 'com.android.application'
android {
    compileSdkVersion 28
    buildToolsVersion "29.0.1"

    defaultConfig {
        applicationId "com.technical.myapplication"
        minSdkVersion 19
        targetSdkVersion 28
        versionCode 1
        versionName "1.0"

        testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"
    }buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android-optimize.txt'), 'proguard-
rules.pro'
        }
    }
}dependencies {
    implementation fileTree(dir: "libs", include: ["*.jar"])
    implementation 'androidx.appcompat:appcompat:1.2.0'
    implementation 'androidx.constraintlayout:constraintlayout:1.1.3'
    implementation 'com.google.android.material:material:1.1.0'
    implementation 'com.google.android.gms:play-services-location:18.0.0' //Location
dependency
    testImplementation 'junit:junit:4.12'
    androidTestImplementation 'androidx.test.ext:junit:1.1.2'
    androidTestImplementation 'androidx.test.espresso:espresso-core:3.3.0'
}

```

Output:



Result:

Thus the application to display the location of the user was executed and verified successfully.

**Ex7 Write programs using Java to create Android application having
Databases for a simple library application.**

Aim:

To develop a java program for library application using android studio.

Algorithm:

STEP1: Develop xml file for defining the structure of application.

STEP2: To make a responsive page use JavaScript file.

STEP3: Run the application in any web browser.

STEP4: Get the input from the user.

STEP5: Output will be displayed.

Source Code:

```
package com.iiitnr.libraryapp;

import android.content.Context;
import android.support.test.InstrumentationRegistry;
import android.support.test.runner.AndroidJUnit4;
import org.junit.Test;
import org.junit.runner.RunWith;
import static org.junit.Assert.*;

/**
 * Instrumented test, which will execute on an Android device.
 *
 * @see <a href="http://d.android.com/tools/testing">Testing
 documentation</a>
 */
@RunWith(AndroidJUnit4.class)
public class ExampleInstrumentedTest {
    @Test
    public void useAppContext() {
        // Context of the app under test.
        Context appContext = InstrumentationRegistry.getTargetContext();

        assertEquals("com.iiitnr.libraryapp", appContext.getPackageName());
    }
}
```

Database Queries:

```
-- phpMyAdmin SQL Dump
-- version 4.4.14
-- http://www.phpmyadmin.net
```

```
--
-- Host: 127.0.0.1
-- Generation Time: Jul 17, 2017 at 06:15 PM
-- Server version: 5.6.26
-- PHP Version: 5.5.28
SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";
SET time_zone = "+00:00";

/*!40101 SET
@OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;

/*!40101 SET
@OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS
*/;

/*!40101 SET
@OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION
*/;

/*!40101 SET NAMES utf8mb4 */;

--

-- Database: `library`

-----

--

-- Table structure for table `admin`

--

CREATE TABLE IF NOT EXISTS `admin` (
  `id` int(11) NOT NULL,
  `FullName` varchar(100) DEFAULT NULL,
  `AdminEmail` varchar(120) DEFAULT NULL,
  `UserName` varchar(100) NOT NULL,
  `Password` varchar(100) NOT NULL,
```

```
`updateDate` timestamp NOT NULL DEFAULT '0000-00-00 00:00:00' ON  
UPDATE CURRENT_TIMESTAMP
```

```
) ENGINE=InnoDB AUTO_INCREMENT=2 DEFAULT CHARSET=latin1;
```

```
--
```

```
-- Dumping data for table `admin`
```

```
--
```

```
INSERT INTO `admin` (`id`, `FullName`, `AdminEmail`, `UserName`,  
`Password`, `updateDate`) VALUES
```

```
(1, 'Anuj Kumar', 'phpgurukulofficial@gmail.com', 'admin',  
'f925916e2754e5e03f75dd58a5733251', '2017-07-16 18:11:42')
```

```
--
```

```
-- Table structure for table `tblauthors`
```

```
--
```

```
CREATE TABLE IF NOT EXISTS `tblauthors` (
```

```
`id` int(11) NOT NULL,
```

```
`AuthorName` varchar(159) DEFAULT NULL,
```

```
`creationDate` timestamp NULL DEFAULT CURRENT_TIMESTAMP,
```

```
`UpdationDate` timestamp NULL DEFAULT NULL ON UPDATE  
CURRENT_TIMESTAMP
```

```
) ENGINE=InnoDB AUTO_INCREMENT=10 DEFAULT CHARSET=latin1;
```

```
--
```

```
-- Dumping data for table `tblauthors`
```

```
--
```

```
INSERT INTO `tblauthors` (`id`, `AuthorName`, `creationDate`,  
`UpdationDate`) VALUES
```

```
(1, 'Anuj kumar', '2017-07-08 12:49:09', '2017-07-08 15:16:59'),
```

```
(2, 'Chetan Bhagatt', '2017-07-08 14:30:23', '2017-07-08 15:15:09'),
```

```
(3, 'Anita Desai', '2017-07-08 14:35:08', NULL),
```

```

(4, 'HC Verma', '2017-07-08 14:35:21', NULL),
(5, 'R.D. Sharma ', '2017-07-08 14:35:36', NULL),
(9, 'fwdfrwer', '2017-07-08 15:22:03', NULL);
-- Table structure for table `tblbooks`
--
CREATE TABLE IF NOT EXISTS `tblbooks` (
  `id` int(11) NOT NULL,
  `BookName` varchar(255) DEFAULT NULL,
  `CatId` int(11) DEFAULT NULL,
  `AuthorId` int(11) DEFAULT NULL,
  `ISBNNumber` int(11) DEFAULT NULL,
  `BookPrice` int(11) DEFAULT NULL,
  `RegDate` timestamp NULL DEFAULT CURRENT_TIMESTAMP,
  `UpdationDate` timestamp NULL DEFAULT NULL ON UPDATE
CURRENT_TIMESTAMP
) ENGINE=InnoDB AUTO_INCREMENT=4 DEFAULT CHARSET=latin1;
--
-- Dumping data for table `tblbooks`
--
INSERT INTO `tblbooks` (`id`, `BookName`, `CatId`, `AuthorId`,
`ISBNNumber`, `BookPrice`, `RegDate`, `UpdationDate`) VALUES
(1, 'PHP And MySql programming', 5, 1, 222333, 20, '2017-07-08 20:04:55',
'2017-07-15 05:54:41'),
(3, 'physics', 6, 4, 1111, 15, '2017-07-08 20:17:31', '2017-07-15 06:13:17');
(5, 1, 'SID009', '2017-07-15 10:59:26', NULL, 0, NULL),
(6, 3, 'SID011', '2017-07-15 18:02:55', NULL, 0, NULL);
-- -----
--
-- Table structure for table `tblstudents`

```

```

--
CREATE TABLE IF NOT EXISTS `tblstudents` (
  `id` int(11) NOT NULL,
  `StudentId` varchar(100) DEFAULT NULL,
  `FullName` varchar(120) DEFAULT NULL,
  `EmailId` varchar(120) DEFAULT NULL,
  `MobileNumber` char(11) DEFAULT NULL,
  `Password` varchar(120) DEFAULT NULL,
  `Status` int(1) DEFAULT NULL,
  `RegDate` timestamp NULL DEFAULT CURRENT_TIMESTAMP,
  `UpdationDate` timestamp NULL DEFAULT NULL ON UPDATE
CURRENT_TIMESTAMP
) ENGINE=InnoDB AUTO_INCREMENT=11 DEFAULT CHARSET=latin1;

--

-- Dumping data for table `tblstudents`

--

INSERT INTO `tblstudents` (`id`, `StudentId`, `FullName`, `EmailId`,
`MobileNumber`, `Password`, `Status`, `RegDate`, `UpdationDate`) VALUES
(1, 'SID002', 'Anuj kumar', 'anuj.lpu1@gmail.com', '9865472555',
'f925916e2754e5e03f75dd58a5733251', 1, '2017-07-11 15:37:05', '2017-07-15
18:26:21'),

-- Indexes for table `admin`

--

ALTER TABLE `admin`

  ADD PRIMARY KEY (`id`);--

-- Indexes for table `tblauthors`

--

ALTER TABLE `tblauthors`

  ADD PRIMARY KEY (`id`);

```

```
--  
-- Indexes for table `tblbooks`  
--  
ALTER TABLE `tblbooks`  
  ADD PRIMARY KEY (`id`);  
--  
-- Indexes for table `tblcategory`  
--  
ALTER TABLE `tblcategory`  
  ADD PRIMARY KEY (`id`);  
--  
-- Indexes for table `tblissuedbookdetails`  
--  
ALTER TABLE `tblissuedbookdetails`  
  ADD PRIMARY KEY (`id`);  
Indexes for table `tblstudents`  
  MODIFY `id` int(11) NOT NULL  
  AUTO_INCREMENT,AUTO_INCREMENT=8;  
--
```

Output:

Library App

USER REGISTRATION

User

Name

Enrollment No.

Library Card No.

Email ID

Password

Confirm Password

☐ I Agree that the information provided is correct

REGISTER

Already Registered ? Sign in

Library App

ID : 1465412
Title : RANDOM PROCESS AND STATISTICS
Type : Mathematics
Issue Date : 12/05/19
Due Date : 26/05/19

ID : 1753604
Title : DATA STRUCTURES IN JAVA
Type : CSE
Issue Date : 15/04/19
Due Date : 29/04/19

ID : 1254501
Title : GOOSEBUMPS
Type : Novel
Issue Date : 30/04/19
Due Date : 14/05/19

ID : 1356203
Title : MICROELECTRONIC CIRCUITS
Type : ECE
Issue Date : 11/05/19
Due Date : 25/05/19

Result:

Thus the application to display the location of the user was executed and verified successfully.