Mini project

1.) Create analog clock by using HTML, CSS and JAVASCRIPT.

ANS)

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Analog Clock</title>
      <link rel ="stylesheet" href="mini pro clock style.css">

</head>
<body>
   <div class="hero">
       <div class="container">
          <div class="clock">
             <img src="./clock digital.png" alt="clock image">
             <div style="--clr:#ffffff; --h:72px;" id="hour"  class="hand" ><i></i></div>
             <div style="--clr:#00a6ff; --h:84px;" id="min" class="hand" ><i></i> </div>
             <div style="--clr:#ff3d58; --h:94px;" id="sec"  class="hand" ><i></i></div>
```

```
        </div>
      </div>
    </div>
    <script src="mini pro clock script.js"></script>
</body>
</html>
```

CSS

```
  *{
  margin: 0;
  padding: 0;
  font-family: 'Lucida Sans', 'Lucida Sans Regular',
'Lucida Grande', 'Lucida Sans Unicode', Geneva,
Verdana, sans-serif;
  box-sizing: border-box;
}
.hero{
  width: 100%;
  min-height: 100vh;
  background:linear-
gradient(45deg,#08001f,#30197d);
  color: #fff;
  position: relative;
}
.clock{
  width: 100%;
  height: 100%;
  /* background:rgba(235,0,255,0.11); */
  border-radius:10px;
  display: flex;
```

```
  align-items: center;
  justify-content: center;
}
.container{
  display: flex;
  width: 800px;
  height: 180px;
  position: absolute;
  top:50%;
  left:50%;
  transform:translate(-50%,-50%);
  justify-content: center;
  align-items: center;
}
.clock::before{
content: '';
width: 8px;
height: 8px;
position: absolute;
border-radius: 50%;
background-color: #fff;
justify-content: center;

}
.hand{
position: absolute;
display: flex;
justify-content: center;
align-items: flex-end;
```

```
}
.hand i{
position: absolute;
width: 4px;
height: var(--h);
background-color: var(--clr);
border-radius: 8px;
justify-content: center;
}
```

Java Script

```
    let hr = document.getElementById('hour');
let min = document.getElementById('min');
let sec = document.getElementById('sec');

function displayTime(){
  let date = new Date();

  let hh = date.getHours();
  let mm  = date.getMinutes();
  let ss  = date.getSeconds();

  let hRotation = 30*hh + mm/2;
  let mRotation = 6*mm;
  let sRotation = 6*ss;

  hr.style.transform = rotate(${hRotation}deg;
  min.style.transform =rotate(${mRotation}deg;
  sec.style.transform =rotate(${sRotation}deg);
```

```
}
setInterval(displayTime,1000);
```

Output:

2.) Create to-do-list application by using HTML, CSS and JAVASCRIPT.

ANS)

```
<!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">
```

```
    <link rel="stylesheet" href="style.css">
    <title>To Do List</title>
</head>
<body>
    <div class="container">
        <div class="input-container">
            <h1>To Do List</h1>
            <input type="text" placeholder="Make your
list" id="inputBox">
            <input type="button" value="Add"
id="addBtn">
        </div>

        <ul class="todoList" id="todoList">
            <!-- <li><p>Task1</p></li>
            <li>Task2</li>
            <li>Task3</li> -->
        </ul>
    </div>

    <script src="script.js"></script>
</body>
</html>
```

CSS

```
 @import
url('https://fonts.googleapis.com/css2?family=Popp
ins&display=swap');

*{
```

```
  margin: 0;
  padding: 0;
  box-sizing: border-box;
  font-family: 'Poppins', sans-serif;
}
body{
  background-color: rgb(232, 238, 238);
}
.container{
  display: flex;
  flex-direction: column;
  justify-content: center;
  align-items: center;
  margin-top: 20px;
}
.input-container{
  width: 100%;
  max-width: 500px;
  text-align: center;
  padding: 20px;
}
.input-container input{
  border: none;
  outline: none;
  padding: 12px;
  margin-block: 12px;
border-radius: 4px;
  font-size: 16px;
}
```

```
.input-container input[type="text"]{
  width: 70%;
}
.input-container input[type="button"]{
  background-color: orange;
  color: #fff;
  font-weight: 700;
  margin-left: 8px;
  cursor: pointer;
  padding: 12px 24px;
}
.input-container input[type="button"]:hover{
  background-color: #60b160;
}
ul{
  width: 70%;
  max-width: 450px;
  /* display: flex;
  justify-content: center;
  flex-direction: column;
align-items: center; */
}
ul li{
  list-style-type: none;
  cursor: pointer;
  margin-block-end: 12px;
  border-radius: 8px;
  border: 0.125px solid #a19f9f;
  padding: 6px 12px;
```

```
  background-color: #ffffff;
  display: flex;
  align-items: center;
  justify-content: space-between;
  transition: background-color 0.5s;
}
ul li:hover{
  background-color: #cbcaca;
}
ul li p{
  flex-grow: 1;
  padding: 2px;
}
.btn{
  border: none;
  outline: none;
  font-size: 16px;
  background: none;
  font-weight: 600;
  cursor: pointer;
  padding: 8px;
}
.deleteBtn{
  color: #ff0000;
}
.editBtn{
  color: #008000;
}
.editBtn{
```

```
  color: #008000;
}
Java script
 const inputBox =
document.getElementById('inputBox');
const addBtn =
document.getElementById('addBtn');
const todoList =
document.getElementById('todoList');

let editTodo = null;

// Function to add todo
const addTodo = () => {
    const inputText = inputBox.value.trim();
    if (inputText.length <= 0) {
        alert("You must write something in your
to do");
        return false;
    }

    if (addBtn.value === "Edit") {
        // Passing the original text to
editLocalTodos function before edit it in the
todoList
```

```
editLocalTodos(editTodo.target.previousEleme
ntSibling.innerHTML);

editTodo.target.previousElementSibling.innerH
TML = inputText;
        addBtn.value = "Add";
        inputBox.value = "";
    }
    else {
        //Creating p tag
        const li = document.createElement("li");
        const p = document.createElement("p");
        p.innerHTML = inputText;
        li.appendChild(p);
 // Creating Edit Btn
        const editBtn =
document.createElement("button");
        editBtn.innerText = "Edit";
        editBtn.classList.add("btn", "editBtn");
        li.appendChild(editBtn);

        // Creating Delete Btn
        const deleteBtn =
document.createElement("button");
        deleteBtn.innerText = "Remove";
        deleteBtn.classList.add("btn", "deleteBtn");
```

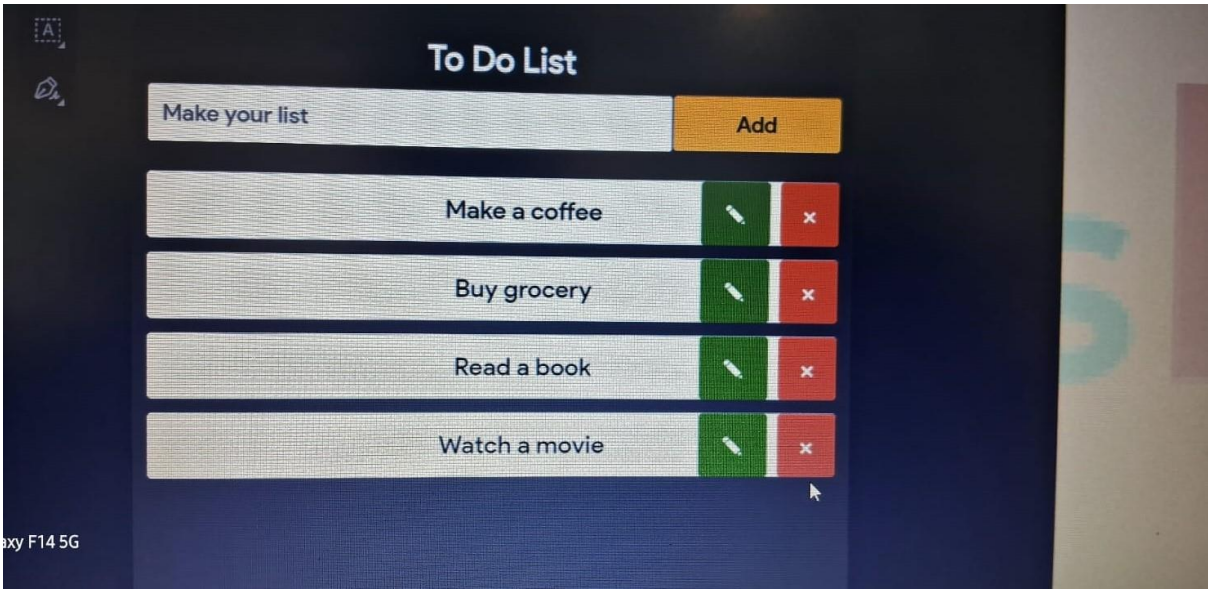
```
        li.appendChild(deleteBtn);

        todoList.appendChild(li);
        inputBox.value = "";

        saveLocalTodos(inputText);
    }
}

// Function to update : (Edit/Delete) todo
const updateTodo = (e) => {
    if (e.target.innerHTML === "Remove") {
        todoList.removeChild(e.target.parentElement);
        deleteLocalTodos(e.target.parentElement);
    }

    if (e.target.innerHTML === "Edit") {
        inputBox.value =
e.target.previousElementSibling.innerHTML;
        inputBox.focus();
        addBtn.value = "Edit";
        editTodo = e;
    }
}

// Function to save local todo
const saveLocalTodos = (todo) => {
    let todos;
    if (localStorage.getItem("todos") === null) {
```

```
        todos = [];
    }
    else {
        todos =
JSON.parse(localStorage.getItem("todos"));
    }
    todos.push(todo);
    localStorage.setItem("todos",
JSON.stringify(todos));
}

// Function to get local todo
const getLocalTodos = () => {
    let todos;
    if (localStorage.getItem("todos") === null) {
        todos = [];
    }
    else {
        todos =
JSON.parse(localStorage.getItem("todos"));
        todos.forEach(todo => {

            //Creating p tag
            const li = document.createElement("li");
            const p = document.createElement("p");
            p.innerHTML = todo;
            li.appendChild(p);
```

```
            // Creating Edit Btn
            const editBtn =
document.createElement("button");
            editBtn.innerText = "Edit";
            editBtn.classList.add("btn", "editBtn");
            li.appendChild(editBtn);

            // Creating Delete Btn
            const deleteBtn =
document.createElement("button");
            deleteBtn.innerText = "Remove";
            deleteBtn.classList.add("btn", "deleteBtn");
            li.appendChild(deleteBtn);

            todoList.appendChild(li);
        });
    }
}

// Function to delete local todo
const deleteLocalTodos = (todo) => {
    let todos;
    if (localStorage.getItem("todos") === null) {
        todos = [];
    }
else {
        todos =
JSON.parse(localStorage.getItem("todos"));
    }
```

```
    let todoText = todo.children[0].innerHTML;
    let todoIndex = todos.indexOf(todoText);
    todos.splice(todoIndex, 1);
    localStorage.setItem("todos",
JSON.stringify(todos));
    // Array functions : slice / splice
    console.log(todoIndex);
}

const editLocalTodos = (todo) => {
    let todos =
JSON.parse(localStorage.getItem("todos"));
    let todoIndex = todos.indexOf(todo);
    todos[todoIndex] = inputBox.value;
    localStorage.setItem("todos",
JSON.stringify(todos));
}

document.addEventListener('DOMContentLoaded',
getLocalTodos);
addBtn.addEventListener('click', addTodo);
todoList.addEventListener('click', updateTodo);
```

Output:


To Do List
Make your list
Add
Make a coffee
Buy grocery
Read a book
Watch a movie