WEATHER FORECAST APP

HTML

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Weather cast</title>
    <link rel="stylesheet" href="style.css">
    <link rel="stylesheet"
href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">

</head>

<body>

  <div class="wrapper">
    <div class="sidebar">
      <div>
        <form class="search" id="search">
          <input type="text" id="query" placeholder="Search..." />
          <button type="submit"><i class="fa fa-search"></i></button>
        </form>
        <div class="weather-icon">
          <img id="icon" src="icons/sun/4.png" alt="" />
        </div>
        <div class="temperature">
          <h1 id="temp">0</h1>
          <span class="temp-unit">°C</span>
        </div>
        <div class="date-time">
          <p id="date-time">Monday, 12:00</p>
        </div>
        <div class="divider"></div>
        <div class="condition-rain">
          <div class="condition">
            <i class="fas fa-cloud"></i>
            <p id="condition">condition</p>
          </div>
          <div class="rain">
            <i class="fas fa-tint"></i>
            <p id="rain">perc - 0%</p>
          </div>
        </div>
      </div>
```

```
  <div class="location">
    <div class="location-icon">
      <i class="fas fa-map-marker-alt"></i>
    </div>
    <div class="location-text">
      <p id="location">location</p>
    </div>
  </div>
</div>
<div class="main">
  <nav>
    <ul class="options">

      <button class="week active">week</button>
    </ul>
    <ul class="options units">
      <button class="celcius active">°C</button>
      <button class="fahrenheit">°F</button>
    </ul>
  </nav>
  <div class="cards" id="weather-cards"></div>
  <div class="highlights">
    <h2 class="heading">today's highlights</h2>
    <div class="cards">


      <div class="card2">
        <h4 class="card-heading">Wind Status</h4>
        <div class="content">
          <p class="wind-speed">0</p>
          <p>km/h</p>
        </div>
      </div>


      <div class="card2">
        <h4 class="card-heading">Humidity</h4>
        <div class="content">
          <p class="humidity">0</p>
          <p class="humidity-status">Normal</p>
        </div>
      </div>


    </div>
  </div>

</div>
```

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

  </body>
  </html>
```

CSS

```
@import
url("https://fonts.googleapis.com/css2?family=Poppins:wght@400;500;600&display=swap");

:root {
  --primary-color: #e9ecf0;
}

* {
  margin: 0;
  padding: 0;
  box-sizing: border-box;
  font-family: "Poppins", sans-serif;
}
body {
  display: flex;
  justify-content: center;
  min-height: 100vh;
  min-width: 1000px;
  padding: 50px;
  background: var(--primary-color);
  background-image: linear-gradient(rgba(0, 0, 0, 0.5), rgba(0, 0, 0, 0.5)),
    url(https://i.pinimg.com/originals/26/be/b0/26beb09153b8df233d82e66bef3edfbb.jpg);
  background-size: cover;
  background-position: center;
  transition: background-image 0.3s ease;
}
img {
  width: 100%;
}
.wrapper {
  display: flex;
  width: 1200px;
  min-width: 900px;
  border-radius: 20px;
  overflow: hidden;
}
.sidebar {
```

```
  width: 30%;
  min-width: 250px;
  padding: 20px;
  background: rgba(197, 192, 192, 0.788);
  display: flex;
  flex-direction: column;
  justify-content: space-between;
}

.search {
  display: flex;
  align-items: center;
  justify-content: space-between;
  margin-bottom: 30px;
  margin-top: 20px;
  position: relative;
}
.search input {
  width: 100%;
  height: 40px;
  border: 1px solid #ced4da;
  border-top-left-radius: 5px;
  border-bottom-left-radius: 5px;
  padding: 0 15px;
  font-size: 14px;
  color: #495057;
}
.search input:focus {
  outline: none;
  border: 1px solid var(--primary-color);
}
.search button {
  min-width: 40px;
  height: 40px;
  border: none;
  border-top-right-radius: 5px;
  border-bottom-right-radius: 5px;
  background:rgba(5, 240, 138, 0.785);
  color: #fff;
  font-size: 14px;
  cursor: pointer;
}
.search ul {
  max-height: 300px;
  overflow-y: auto;
  position: absolute;
  width: 100%;
  top: 40px;
```

```
  border-radius: 5px;
  transition: all 0.3s ease;
  box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
  background-color: #fff;
}
.search ul li {
  padding: 10px 15px;
  border-bottom: 1px solid #f1f1f1;
  cursor: pointer;
  text-transform: capitalize;
}
.search ul li:last-child {
  border-bottom: none;
}
.search ul li:hover {
  background-color: #f1f1f1;
}
.search ul li.active {
  background-color: #f1f1f1;
}
.weather-icon {
  width: 100%;
  height: 150px;
  text-align: center;
  margin-top: 20px;
  margin-bottom: 100px;
}
.weather-icon #icon {
  width: 80%;
  object-fit: cover;
}
.temperature {
  display: flex;
}
.temperature #temp {
  font-size: 70px;
  font-weight: 100;
  line-height: 1;
}
.temperature span {
  font-size: 40px;
  margin-top: -10px;
  display: block;
}
.divider {
  width: 100%;
  height: 1px;
  background: #e9ecef;
```

```
  margin: 20px 0;
}
.condition-rain {
  font-size: 12px;
  text-transform: capitalize;
}
.condition-rain div {
  display: flex;
  align-items: center;
  gap: 10px;
  margin-bottom: 10px;
}
.condition-rain div i {
  width: 20px;
}
.location {
  display: flex;
  align-items: center;
  font-size: 14px;
  gap: 10px;
  margin-top: 10px;
}
.main {
  width: 100%;
  min-width: 400px;
  padding: 20px 40px;
  background-color: #f6f6f8;
  position: relative;
  padding-bottom: 90px;
}

.main nav {
  display: flex;
  align-items: center;
  justify-content: space-between;
}
.main nav .options {
  display: flex;
  gap: 20px;
  align-items: center;
}
.main nav .options button {
  border: none;
  background: none;
  font-size: 16px;
  font-weight: 600;
  color: #495057;
  cursor: pointer;
```

```
  text-transform: capitalize;
}
.main nav .options button.active {
  color: rgb(241, 6, 6);
}

.main nav .units button {
  width: 40px;
  height: 40px;
  border-radius: 50%;
  color: #1a1a1a;
  background-color: #fff;
}
.main nav .units button.active {
  color: #fff;
  background-color: #1a1a1a;
}
.main .cards {
  display: flex;
  flex-wrap: wrap;
  gap: 20px;
  margin-top: 50px;
}

.cards .card {
  width: 100px;
  height: 130px;
  border-radius: 20px;
  color: #1a1a1a;
  background-color: #fff;
  text-align: center;
  padding: 10px 0;
  display: flex;
  flex-direction: column;
  justify-content: space-between;
}
.card h2 {
  font-size: 15px;
  font-weight: 600;
}
.card .card-icon {
  width: 50%;
  margin: 0 auto;
}
.card .day-temp {
  font-size: 12px;
  display: flex;
  justify-content: center;
```

```
  display: flex;
}
.highlights {
  display: flex;
  flex-wrap: wrap;
  gap: 20px;
  margin-top: 50px;
}
.highlights .heading {
  width: 100%;
  font-size: 20px;
  font-weight: 600;
  text-transform: capitalize;
}

.card2 {
  width: 250px;
  height: 150px;
  border-radius: 20px;
  color: #1a1a1a;
  background-color: #fff;
  padding: 10px 20px;
  display: flex;
  flex-direction: column;
}
.card2 .card-heading {
  color: #c2c2c2;
}

.card2 .content {
  margin-top: 20px;
}
.card2 .content p:first-child {
  text-align: center;
  font-size: 30px;
}
.card2 .content p:nth-child(2) {
  font-size: 12px;
  margin-top: 20px;
  text-align: left;
}
```

Jscript

```
const temp = document.getElementById("temp"),
  date = document.getElementById("date-time"),
```

```
  condition = document.getElementById("condition"),
  rain = document.getElementById("rain"),
  mainIcon = document.getElementById("icon"),
  currentLocation = document.getElementById("location"),

  windSpeed = document.querySelector(".wind-speed"),

  humidity = document.querySelector(".humidity"),

  humidityStatus = document.querySelector(".humidity-status"),

  searchForm = document.querySelector("#search"),
  search = document.querySelector("#query"),
  celciusBtn = document.querySelector(".celcius"),
  fahrenheitBtn = document.querySelector(".fahrenheit"),
  tempUnit = document.querySelectorAll(".temp-unit"),
  hourlyBtn = document.querySelector(".hourly"),
  weekBtn = document.querySelector(".week"),
  weatherCards = document.querySelector("#weather-cards");

let currentCity = "";
let currentUnit = "c";
let hourlyorWeek = "week";

function getDateTime() {
  let now = new Date();
  let hour = now.getHours();
  let minute = now.getMinutes();

  let days = [
    "Sunday",
    "Monday",
    "Tuesday",
    "Wednesday",
    "Thursday",
    "Friday",
    "Saturday",
  ];
  hour = hour % 12;
  if (hour < 10) {
    hour = "0" + hour;
  }
  if (minute < 10) {
    minute = "0" + minute;
  }
  let dayString = days[now.getDay()];
  return `${dayString}, ${hour}:${minute}`;
}
```

```
date.innerText = getDateTime();
setInterval(() => {
  date.innerText = getDateTime();
}, 1000);

function getPublicIp() {
  fetch("https://geolocation-db.com/json/", {
    method: "GET",
    headers: {},
  })
    .then((response) => response.json())
    .then((data) => {
      currentCity = data.city;
      getWeatherData(data.city, currentUnit, hourlyorWeek);
    })
    .catch((err) => {
      console.error(err);
    });
}

getPublicIp();

function getWeatherData(city, unit, hourlyorWeek) {
  fetch(

`https://weather.visualcrossing.com/VisualCrossingWebServices/rest/services/timeline/${city}
?unitGroup=metric&key=EJ6UBL2JEQGYB3AA4ENASN62J&contentType=json`,
    {
      method: "GET",
      headers: {},
    }
  )
    .then((response) => response.json())
    .then((data) => {
      let today = data.currentConditions;
      if (unit === "c") {
        temp.innerText = today.temp;
      } else {
        temp.innerText = celciusToFahrenheit(today.temp);
      }
      currentLocation.innerText = data.resolvedAddress;
      condition.innerText = today.conditions;
      rain.innerText = "Perc - " + today.precip + "%";
      windSpeed.innerText = today.windspeed;
      mainIcon.src = getIcon(today.icon);
      changeBackground(today.icon);
      humidity.innerText = today.humidity + "%";
```

```
      updateHumidityStatus(today.humidity);
      if (hourlyorWeek === "hourly") {
        updateForecast(data.days[0].hours, unit, "day");
      } else {
        updateForecast(data.days, unit, "week");
      }
    })
    .catch((err) => {
      console.error(err);
    });
}




//function to update Forecast
function updateForecast(data, unit, type) {
  weatherCards.innerHTML = "";
  let day = 0;
  let numCards = 0;
  if (type === "day") {
    numCards = 24;
  } else {
    numCards = 7;
  }
  for (let i = 0; i < numCards; i++) {
    let card = document.createElement("div");
    card.classList.add("card");
    let dayName = getHour(data[day].datetime);
    if (type === "week") {
      dayName = getDayName(data[day].datetime);
    }
    let dayTemp = data[day].temp;
    if (unit === "f") {
      dayTemp = celciusToFahrenheit(data[day].temp);
    }
    let iconCondition = data[day].icon;
    let iconSrc = getIcon(iconCondition);
    let tempUnit = "°C";
    if (unit === "f") {
      tempUnit = "°F";
    }
    card.innerHTML = `
              <h2 class="day-name">${dayName}</h2>
            <div class="card-icon">
              <img src="${iconSrc}" class="day-icon" alt="" />
            </div>
```

```
          <div class="day-temp">
            <h2 class="temp">${dayTemp}</h2>
            <span class="temp-unit">${tempUnit}</span>
          </div>
  `;
    weatherCards.appendChild(card);
    day++;
  }
}

// function to change weather icons
function getIcon(condition) {
  if (condition === "partly-cloudy-day") {
    return "https://i.ibb.co/PZQXH8V/27.png";
  } else if (condition === "partly-cloudy-night") {
    return "https://i.ibb.co/Kzkk59k/15.png";
  } else if (condition === "rain") {
    return "https://i.ibb.co/kBd2NTS/39.png";
  } else if (condition === "clear-day") {
    return "https://i.ibb.co/rb4rrJL/26.png";
  } else if (condition === "clear-night") {
    return "https://i.ibb.co/1nxNGHL/10.png";
  } else {
    return "https://i.ibb.co/rb4rrJL/26.png";
  }
}

// function to change background depending on weather conditions
function changeBackground(condition) {
  const body = document.querySelector("body");
  let bg = "";
  if (condition === "partly-cloudy-day") {
    bg =
"https://img.freepik.com/premium-vector/sky-background-video-conferencing_23-214864167
6.jpg?w=1060";
  } else if (condition === "partly-cloudy-night") {
    bg =
"https://img.freepik.com/free-vector/night-sea-landscape-with-coastline-moon-sky_107791-1
664.jpg?w=1060&t=st=1685346007~exp=1685346607~hmac=51cff96ca49eef9bdb786e45a
69a921c84d4a1ebd5d39b4206fa15058f0a70e2";
  } else if (condition === "rain") {
    bg = "https://i.ytimg.com/vi/P-7Lwh93vDI/maxresdefault.jpg";
  } else if (condition === "clear-day") {
    bg =
"https://img.freepik.com/free-vector/ocean-sea-beach-nature-tranquil-landscape_33099-224
8.jpg?w=1380&t=st=1685346222~exp=1685346822~hmac=97e6d4f325908552d6b75c1fc64
9a677a7f237012f58d9a827cf925c8bcbbe1e";
  } else if (condition === "clear-night") {
```

```
    bg =
"https://img.freepik.com/free-vector/cartoon-safari-landscape-night_52683-81606.jpg?w=138
0&t=st=1685346273~exp=1685346873~hmac=546c5098c0cd0a4a098ed0e48170cbdb1dc5
dc719eaddf37945524aace1aa9f7";
  } else {
    bg = "https://wallpaperaccess.com/full/3430.jpg";
  }
  body.style.backgroundImage = `linear-gradient( rgba(0, 0, 0, 0.5), rgba(0, 0, 0, 0.5)
),url(${bg})`;
}

//get hours from hh:mm:ss
function getHour(time) {
  let hour = time.split(":")[0];
  let min = time.split(":")[1];
  if (hour > 12) {
    hour = hour - 12;
    return `${hour}:${min} PM`;
  } else {
    return `${hour}:${min} AM`;
  }
}

// convert time to 12 hour format
function covertTimeTo12HourFormat(time) {
  let hour = time.split(":")[0];
  let minute = time.split(":")[1];
  let ampm = hour >= 12 ? "pm" : "am";
  hour = hour % 12;
  hour = hour ? hour : 12; // the hour '0' should be '12'
  hour = hour < 10 ? "0" + hour : hour;
  minute = minute < 10 ? "0" + minute : minute;
  let strTime = hour + ":" + minute + " " + ampm;
  return strTime;
}

// function to get day name from date
function getDayName(date) {
  let day = new Date(date);
  let days = [
    "Sunday",
    "Monday",
    "Tuesday",
    "Wednesday",
    "Thursday",
    "Friday",
    "Saturday",
  ];
```

```
  return days[day.getDay()];
}

// function to get humidity status
function updateHumidityStatus(humidity) {
  if (humidity <= 30) {
    humidityStatus.innerText = "Low";
  } else if (humidity <= 60) {
    humidityStatus.innerText = "Moderate";
  } else {
    humidityStatus.innerText = "High";
  }
}


// function to handle search form
searchForm.addEventListener("submit", (e) => {
  e.preventDefault();
  let location = search.value;
  if (location) {
    currentCity = location;
    getWeatherData(location, currentUnit, hourlyorWeek);
  }
});

// function to conver celcius to fahrenheit
function celciusToFahrenheit(temp) {
  return ((temp * 9) / 5 + 32).toFixed(1);
}


var currentFocus;
search.addEventListener("input", function (e) {
  removeSuggestions();
  var a,
    b,
    i,
    val = this.value;
  if (!val) {
    return false;
  }
  currentFocus = -1;

  a = document.createElement("ul");
  a.setAttribute("id", "suggestions");

  this.parentNode.appendChild(a);
```

```
  for (i = 0; i < cities.length; i++) {
    /*check if the item starts with the same letters as the text field value:*/
    if (
      cities[i].name.substr(0, val.length).toUpperCase() == val.toUpperCase()
    ) {
      /*create a li element for each matching element:*/
      b = document.createElement("li");
      /*make the matching letters bold:*/
      b.innerHTML =
        "<strong>" + cities[i].name.substr(0, val.length) + "</strong>";
      b.innerHTML += cities[i].name.substr(val.length);
      /*insert a input field that will hold the current array item's value:*/
      b.innerHTML += "<input type='hidden' value='" + cities[i].name + "'>";
      /*execute a function when someone clicks on the item value (DIV element):*/
      b.addEventListener("click", function (e) {
        /*insert the value for the autocomplete text field:*/
        search.value = this.getElementsByTagName("input")[0].value;
        removeSuggestions();
      });

      a.appendChild(b);
    }
  }
});
/*execute a function presses a key on the keyboard:*/
search.addEventListener("keydown", function (e) {
  var x = document.getElementById("suggestions");
  if (x) x = x.getElementsByTagName("li");
  if (e.keyCode == 40) {
    /*If the arrow DOWN key
      is pressed,
      increase the currentFocus variable:*/
    currentFocus++;
    /*and and make the current item more visible:*/
    addActive(x);
  } else if (e.keyCode == 38) {
    /*If the arrow UP key
      is pressed,
      decrease the currentFocus variable:*/
    currentFocus--;
    /*and and make the current item more visible:*/
    addActive(x);
  }
  if (e.keyCode == 13) {
    /*If the ENTER key is pressed, prevent the form from being submitted,*/
    e.preventDefault();
    if (currentFocus > -1) {
      /*and simulate a click on the "active" item:*/
```

```
      if (x) x[currentFocus].click();
    }
  }
});
function addActive(x) {
  /*a function to classify an item as "active":*/
  if (!x) return false;
  /*start by removing the "active" class on all items:*/
  removeActive(x);
  if (currentFocus >= x.length) currentFocus = 0;
  if (currentFocus < 0) currentFocus = x.length - 1;
  /*add class "autocomplete-active":*/
  x[currentFocus].classList.add("active");
}
function removeActive(x) {
  /*a function to remove the "active" class from all autocomplete items:*/
  for (var i = 0; i < x.length; i++) {
    x[i].classList.remove("active");
  }
}

function removeSuggestions() {
  var x = document.getElementById("suggestions");
  if (x) x.parentNode.removeChild(x);
}

fahrenheitBtn.addEventListener("click", () => {
  changeUnit("f");
});
celciusBtn.addEventListener("click", () => {
  changeUnit("c");
});


function changeUnit(unit) {
  if (currentUnit !== unit) {
    currentUnit = unit;
    tempUnit.forEach((elem) => {
      elem.innerText = `°${unit.toUpperCase()}`;
    });
    if (unit === "c") {
      celciusBtn.classList.add("active");
      fahrenheitBtn.classList.remove("active");
    } else {
      celciusBtn.classList.remove("active");
      fahrenheitBtn.classList.add("active");
    }
    getWeatherData(currentCity, currentUnit, hourlyorWeek);
```

```
  }
}

hourlyBtn.addEventListener("click", () => {
  changeTimeSpan("hourly");
});
weekBtn.addEventListener("click", () => {
  changeTimeSpan("week");
});


function changeTimeSpan(unit) {
  if (hourlyorWeek !== unit) {
    hourlyorWeek = unit;
    if (unit === "hourly") {
      hourlyBtn.classList.add("active");
      weekBtn.classList.remove("active");
    } else {
      hourlyBtn.classList.remove("active");
      weekBtn.classList.add("active");
    }
    getWeatherData(currentCity, currentUnit, hourlyorWeek);
  }
}
```