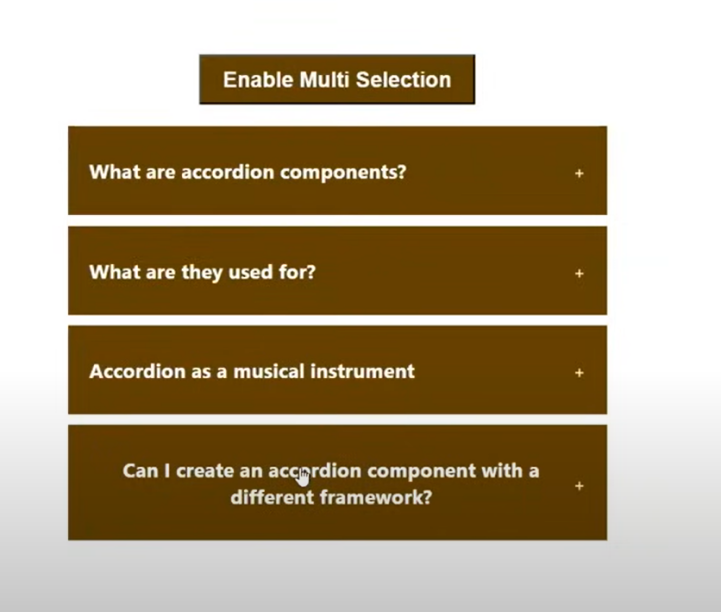
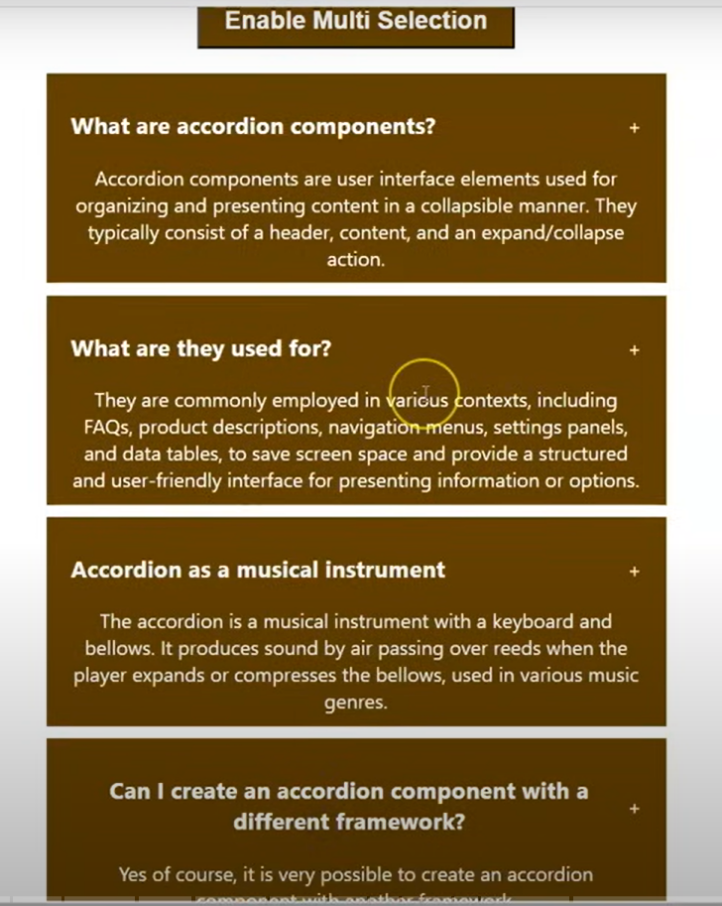
**REACT PROJECTS**

**Ejercicios en React para estudiar a fondo los conceptos**

**<https://www.youtube.com/watch?v=5ZdHfJVAY-s&t=10s>**

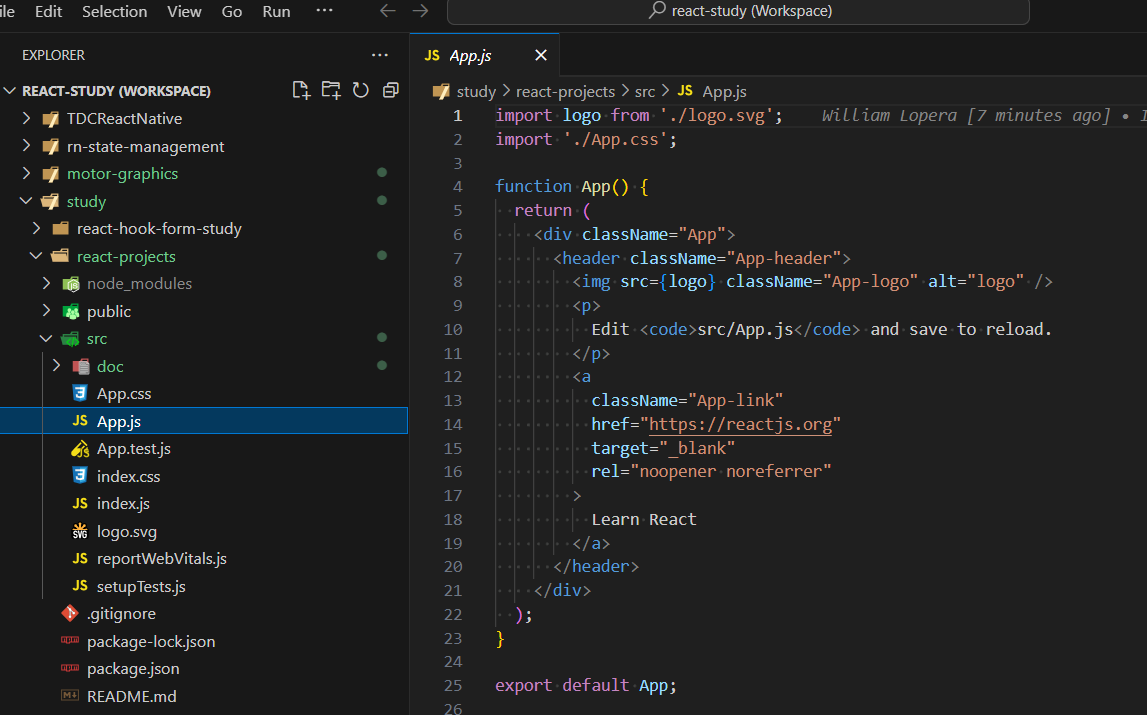
1. **Crear un acordeón de apertura simple o multi apertura a la vez**

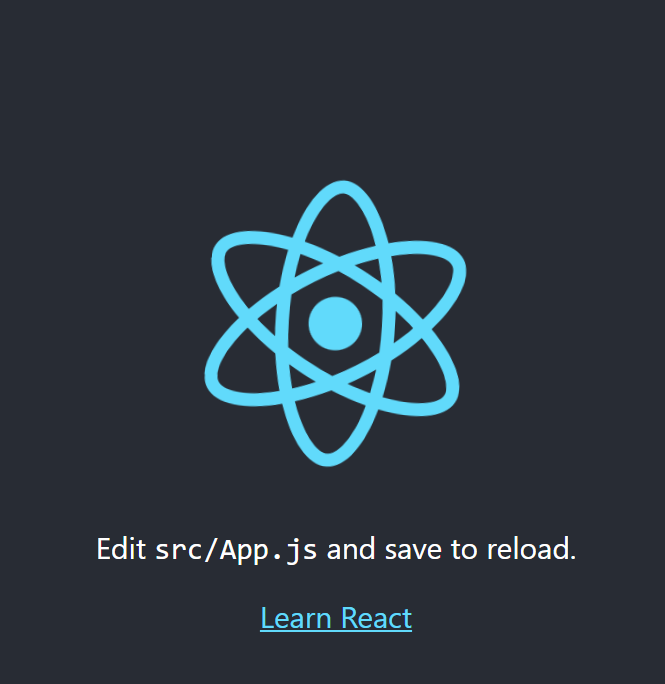




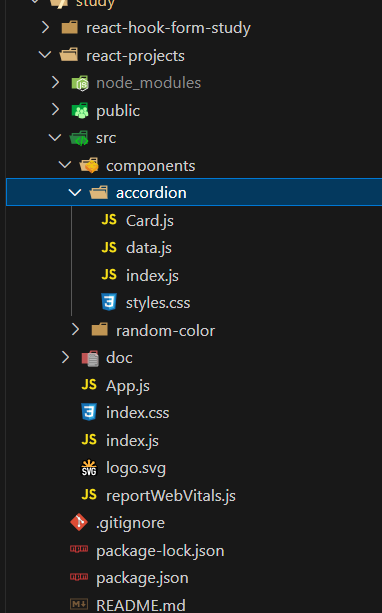
* Crear un App de react

\_ > npx create-react-app react-projects





* Código para crear Acordeón



* + Index.js

import React, { useState } from "react";

import { data } from "./data";

import { Card } from "./Card";

import "./styles.css";

export const Accordion = () => {

  const [listDisplay, setListDisplay] = useState(["1"]);

  const [enableMultiple, setEnabledMultiple] = useState(false);

  const records = data;

  const handleSetListDisplay = (id) => {

    if (enableMultiple) {

      let list = [...listDisplay];

      if (listDisplay.includes(id)) {

        list = list.filter((item) => item !== id);

      } else {

        list.push(id);

      }

      setListDisplay(list);

    } else {

      setListDisplay([id]);

    }

  };

  const handleSetEnabledMultiple = () => {

    if (enableMultiple) {

      setListDisplay(["1"]);

    }

    setEnabledMultiple(!enableMultiple);

  };

  return (

    <div className="container">

      <button onClick={handleSetEnabledMultiple} className="button">

        {enableMultiple ? "Selección Multiple" : "Selección Simple"}

      </button>

      {records &&

        records.length > 0 &&

        records.map((record) => (

          <Card

            key={record.id}

            id={record.id}

            question={record.question}

            answer={record.answer}

            showAnswer={listDisplay.includes(record.id)}

            onShowAnswer={handleSetListDisplay}

          />

        ))}

    </div>

  );

};

* card.js

import React from "react";

import "./styles.css";

export const Card = ({

  id,

  question,

  answer,

  showAnswer = false,

  onShowAnswer,

}) => {

  return (

    <div className="card-container">

      <div className={showAnswer ? "card-question-selected" : "card-question"}>

        <h5>{question}</h5>

        <div style={{ cursor: "pointer" }} onClick={() => onShowAnswer(id)}>

          {showAnswer ? "-" : "+"}

        </div>

      </div>

      {showAnswer && <div>{answer}</div>}

    </div>

  );

};

* data.js

export const data = [

  {

    id: "1",

    question: "¿Qué son los componentes del acordeón?",

    answer:

      "Los componentes de acordeón son elementos de la interfaz de usuario que se utilizan para organizar y presentar contenido de forma plegable. Por lo general, constan de un encabezado, contenido y una acción de expandir/contraer.",

  },

  {

    id: "2",

    question: "¿Para qué se utilizan?",

    answer:

      "Se emplean comúnmente en diversos contextos, incluidas preguntas frecuentes, descripciones de productos, menús de navegación, paneles de configuración y tablas de datos, para ahorrar espacio en la pantalla y proporcionar una interfaz estructurada y fácil de usar para presentar información u opciones.",

  },

  {

    id: "3",

    question: "El acordeón como instrumento musical",

    answer:

      "El acordeón es un instrumento musical con teclado y fuelle. Produce sonido mediante el paso del aire sobre las lengüetas cuando el intérprete expande o comprime el fuelle, utilizado en diversos géneros musicales.",

  },

  {

    id: "4",

    question: "¿Puedo crear un componente de acordeón con un marco diferente?",

    answer:

      "Sí, por supuesto, es muy posible crear un componente de acordeón con otro marco.",

  },

];

* + Styles.css

.container {

    display: flex;

    flex-direction: column;

    justify-content: center;

    align-items: center;

}

.button {

    margin: 10px;

}

.card-container {

    border: 1px solid lightgray;

    width: 30%;

}

.card-question {

    display: flex;

    flex-direction: row;

    justify-content: space-between;

}

.card-question-selected {

    display: flex;

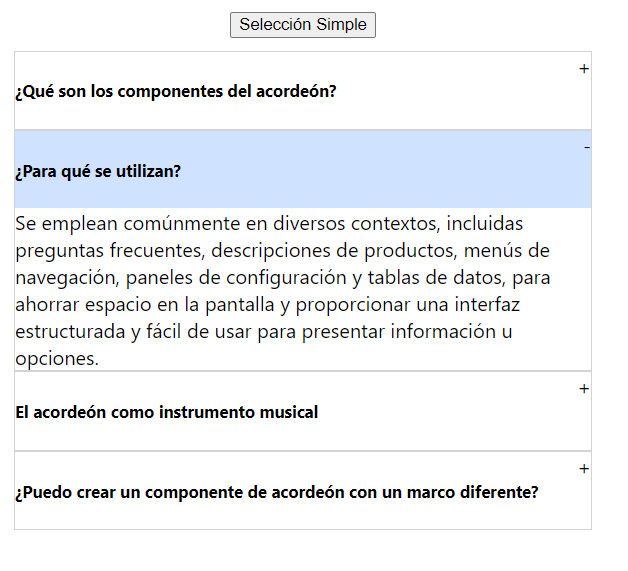
    flex-direction: row;

    background-color: #CFE2FF;

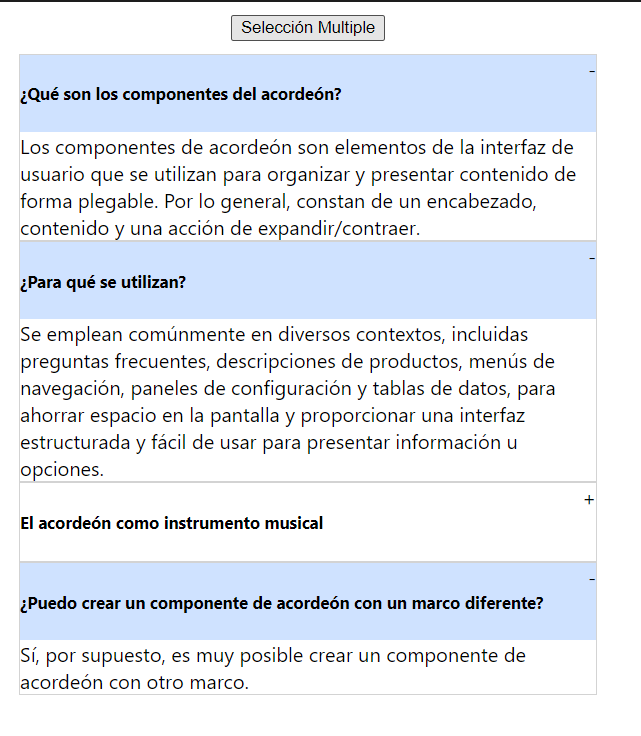
    justify-content: space-between;

}

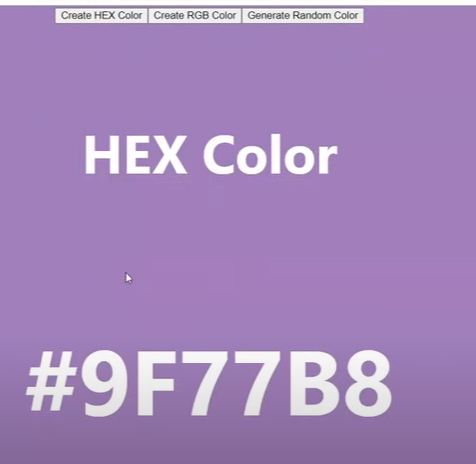
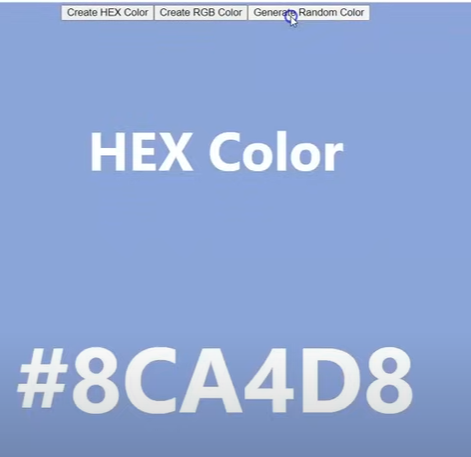
* Salida Selección Simple:

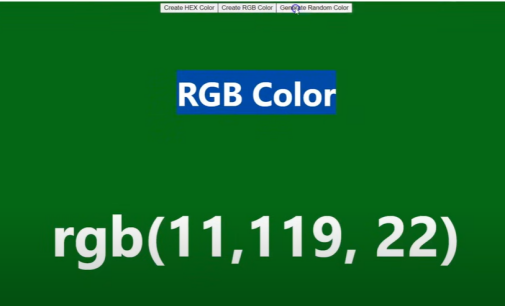
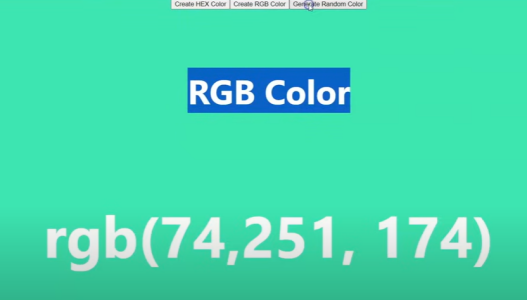


* Salida Selección Multiple:



1. **Crear un App con tres botones para mostrar colores aleatoriamente en RGB o Hexadecimal**

* Código para crear colores
* index.js

import React, { useState } from "react";

import "./styles.css";

export const RandomColor = () => {

  const [color, setColor] = useState("#FF0000");

  const [typeColor, setTypeColor] = useState("HEX");

  const [typeButton, setTypeButton] = useState("HEX");

  const getRandom15 = () => {

    return Math.floor(Math.random() \* 16)

      .toString(16)

      .toUpperCase();

  };

  const getRandom255 = () => {

    return Math.floor(Math.random() \* 256);

  };

  const getColorHEX = () => {

    let hex = "#";

    for (let index = 0; index < 6; index++) {

      hex += getRandom15();

    }

    return hex;

  };

  const getColorRGB = () => {

    return `rgb(${getRandom255()},${getRandom255()},${getRandom255()})`;

  };

  const handleColorHex = () => {

    setColor(getColorHEX());

    setTypeColor("HEX");

    setTypeButton("HEX");

  };

  const handleColorRGB = () => {

    setColor(getColorRGB());

    setTypeColor("RGB");

    setTypeButton("RGB");

  };

  const getColorRandom = () => {

    setTypeButton("RAN");

    if (typeColor === "RGB") {

      setColor(getColorRGB());

    } else {

      setColor(getColorHEX());

    }

  };

  const divColor = {

    backgroundColor: color,

  };

  return (

    <div className="color-container" style={divColor}>

      <div className="color-button-container">

        <button

          onClick={handleColorHex}

          style={

            typeButton === "HEX"

              ? { backgroundColor: "#6d99f1" }

              : { backgroundColor: "" }

          }

        >

          Color HEX

        </button>

        <button

          onClick={handleColorRGB}

          style={

            typeButton === "RGB"

              ? { backgroundColor: "#6d99f1" }

              : { backgroundColor: "" }

          }

        >

          Color RGB

        </button>

        <button

          onClick={getColorRandom}

          style={

            typeButton === "RAN"

              ? { backgroundColor: "#6d99f1" }

              : { backgroundColor: "" }

          }

        >

          Generar Color

        </button>

      </div>

      <div className="color-txt">{`Color ${typeColor}`}</div>

      <div className="color-txt-color">{color}</div>

    </div>

  );

};

* styles.css

.color-container {

    display: flex;

    flex-direction: column;

    justify-content:flex-start;

    align-items: center;

    width: 100vw;

    height: 100vh;

}

.color-button-container{

    margin: 20px;

    display: flex;

    gap: 15px;

}

.color-txt{

    font-size: 60px;

    font-weight: bold;

    color: white;

    margin: 60px;

}

.color-txt-color {

    font-size: 80px;

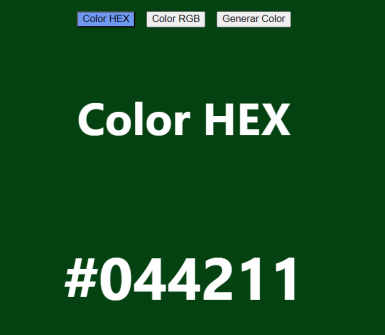
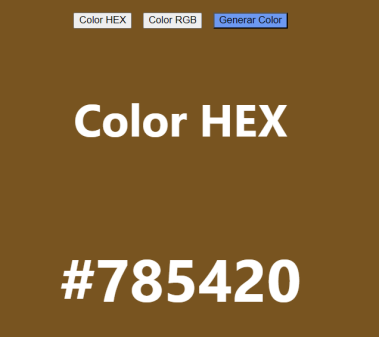
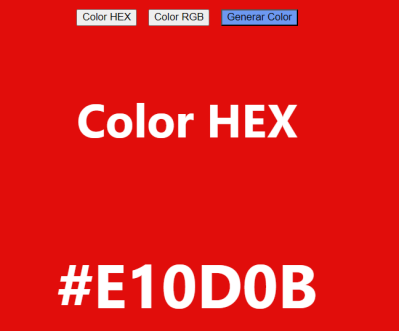
    font-weight: bold;

    color: white;

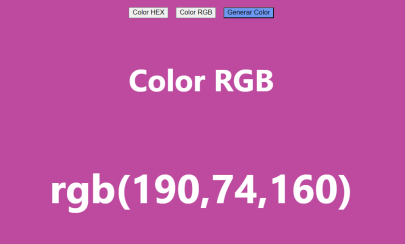
    margin: 60px;

}

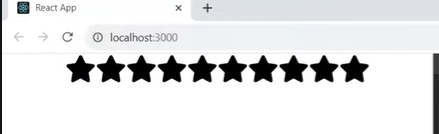
* Salida HEX:

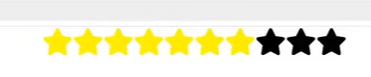
  

* Salida RGB:

1. **Seleccionar Raiting - Calificación de estrellas**





* Instalación de libraría para imágenes (icon start)

npm install react-icons --save

* Código
* index.js

import React, { useState } from "react";

import { Star } from "./star";

import "./styles.css";

const arr = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9];

export const StarRating = () => {

  const [currentSeleted, setCurrentSeleted] = useState([]);

  const [hoverSeleted, setHoverSeleted] = useState([]);

  const getArrSeleted = (id) => {

    const arr = [];

    for (let index = 0; index < id + 1; index++) {

      arr.push(index);

    }

    return arr;

  };

  const handleClick = (id) => {

    setCurrentSeleted(getArrSeleted(id));

  };

  const handleMouseEnter = (id) => {

    setHoverSeleted(getArrSeleted(id));

  };

  const handleMouseLeave = (id) => {

    setHoverSeleted(currentSeleted);

  };

  const selected = hoverSeleted.length > 0 ? hoverSeleted : currentSeleted;

  return (

    <div className="container">

      {arr &&

        arr.map((item) => (

          <Star

            key={item}

            id={item}

            isSeleted={selected.includes(item)}

            onPress={handleClick}

            mouseEnter={handleMouseEnter}

            mouseLeave={handleMouseLeave}

            size={40}          />

        ))}

      <button

        className="button"

        onClick={() => {

          setHoverSeleted([]);

          setCurrentSeleted([]);

        }}

      >

        Limpiar

      </button>

    </div>

  );

};

* star.js

import React from "react";

import { FaStar } from "react-icons/fa";

import "./styles.css";

export const Star = ({

  id,

  isSeleted,

  onPress,

  mouseEnter,

  mouseLeave,

  size,

}) => {

  return (

    <FaStar

      className={isSeleted ? "star-selected" : "star"}

      size={size}

      onClick={() => onPress(id)}

      onMouseEnter={() => mouseEnter(id)}

      onMouseLeave={() => mouseLeave(id)}

    />

  );

};

* styles.css

.container {

    display: flex;

    justify-content: center;

    align-items: center;

    margin: 20px;

}

.star {

    cursor: pointer;

}

.star-selected {

    color: yellow;

}

.star:hover {

    color: yellow;

}

.button{

    border: none;

    background-color: transparent;

    cursor: pointer;

    text-decoration: underline;

}

.button:hover {

    color: blue;

}

* Salida:











* + Luego de click en limpiar

