```
import React, { useRef, useEffect } from "react";
import PropTypes from "prop-types";
import { Card } from "antd";
import ApexChart from "react-apexcharts";
import {
  apexLineChartDefaultOption,
  apexBarChartDefaultOption,
  apexAreaChartDefaultOption,
} from "constants/ChartConstant";
import { useResizeDetector } from "react-resize-detector";
import { DIR_RTL } from "constants/ThemeConstant";

const titleStyle = {
  position: "absolute",
  zIndex: "1",
};

const extraStyle = {
  position: "absolute",
  zIndex: "1",
  right: "0",
  top: "-2px",
};

const getChartTypeDefaultOption = (type) => {
  switch (type) {
    case "line":
      return apexLineChartDefaultOption;
    case "bar":
      return apexBarChartDefaultOption;
    case "area":
      return apexAreaChartDefaultOption;
    default:
      return apexLineChartDefaultOption;
  }
};

const ChartWidget = ({
  title,
  series,
  width,
  height,
  xAxis,
  customOptions,
  card,
  type,
  extra,
  direction,
  bodyClass,
}) => {
  let options = JSON.parse(JSON.stringify(getChartTypeDefaultOption(type)));
  const isMobile = window.innerWidth < 768;
  const setLegendOffset = () => {
    if (chartRef.current) {
      const lengend = chartRef.current.querySelectorAll(
        "div.apexcharts-legend"
      )[0];
      lengend.style.marginRight = `${
        isMobile ? 0 : extraRef.current?.offsetWidth
      }px`;
      if (direction === DIR_RTL) {
        lengend.style.right = "auto";
        lengend.style.left = "0";
      }
      if (isMobile) {
        lengend.style.position = "relative";
        lengend.style.top = 0;
        lengend.style.justifyContent = "start";
        lengend.style.padding = 0;
      }
    }
  };

  useEffect(() => {
    setLegendOffset();
    // eslint-disable-next-line react-hooks/exhaustive-deps
  }, []);

  const extraRef = useRef(null);
  const chartRef = useRef();

  options.xaxis.categories = xAxis;
  if (customOptions) {
    options = { ...options, ...customOptions };
  }

  const onResize = () => {
    setTimeout(() => {
      setLegendOffset();
    }, 600);
  };

  const { ref: resizeRef } = useResizeDetector({
    onResize: () => {
      onResize();
    },
  });

  const renderChart = () => (
    <div ref={resizeRef}>
      <div
        style={direction === DIR_RTL ? { direction: "ltr" } : {}}
        className="chartRef"
        ref={chartRef}
      >
        <ApexChart
          options={options}
          type={type}
          series={series}
          width={width}
          height={height}
        />
      </div>
    </div>
  );

  return (
    <>
      {card ? (
        <Card>
          <div className={`position-relative ${bodyClass}`}>
            {<div style={!isMobile ? { titleStyle } : {}}>{title}</div> && (
              <h4
                className="font-weight-bold"
                style={!isMobile ? { titleStyle } : {}}
              >
                {title}
              </h4>
            )}
            {extra && (
              <div ref={extraRef} style={!isMobile ? { extraStyle } : {}}>
                {extra}
              </div>
            )}
            {renderChart()}
          </div>
        </Card>
      ) : (
        renderChart()
      )}
    </>
  );
};

ChartWidget.propTypes = {
  title: PropTypes.oneOfType([PropTypes.string, PropTypes.element]),
  series: PropTypes.array.isRequired,
  xAxis: PropTypes.array,
  customOptions: PropTypes.object,
  width: PropTypes.oneOfType([PropTypes.string, PropTypes.number]),
  height: PropTypes.oneOfType([PropTypes.string, PropTypes.number]),
  card: PropTypes.bool,
  type: PropTypes.string,
  extra: PropTypes.element,
  bodyClass: PropTypes.string,
};

ChartWidget.defaultProps = {
  series: [],
  height: 300,
  width: "100%",
  card: true,
  type: "line",
};

export default ChartWidget;
```