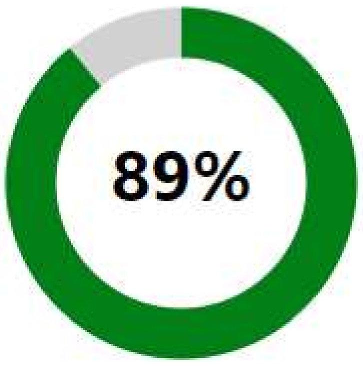

Fig :- ZProgress Bar Core Component

Code

```
import React, { useState, useEffect } from 'react';
import { useStyles } from './Zprogresscss';
import { mergeClasses } from '@fluentui/react-components';

interface CircularProgressProps {
    radius: number;
    strokeWidth: number;
    percentage: number;
    texttitleclassName: string;
    circleFactor: number;
}

const Zprogress: React.FC<CircularProgressProps> = ({ radius, strokeWidth,
percentage , texttitleclassName , circleFactor , ...Props}) => {
    const classes = useStyles();
    const [normalizedRadius, setNormalizedRadius] = useState(radius - strokeWidth
/ circleFactor);
    const [circumference, setCircumference] = useState(2 * Math.PI *
normalizedRadius);
    const [offset, setOffset] = useState(circumference);
    const [animatedOffset, setAnimatedOffset] = useState(circumference);
    const [color, setColor] = useState('#4caf50');

    useEffect(() => {
        const newCircumference = 2 * Math.PI * normalizedRadius;
        setCircumference(newCircumference);
        const newOffset = newCircumference - (percentage / 100) *
newCircumference;
        setOffset(newOffset);
        animateFill(newOffset);

        if (percentage <= 25) {
            setColor('#ff0000');
        } else if (percentage <= 50) {
            setColor('#ffa500');
        } else if (percentage <= 80) {
            setColor('#ffff00');
        } else {
            setColor('#008000');
        }
    }, [percentage, normalizedRadius]);

    const animateFill = (newOffset: number) => {
        const duration = 500;
        let start = performance.now();

        const animate = (time: number) => {
            let timeFraction = (time - start) / duration;
            if (timeFraction > 1) timeFraction = 1;

            const currentOffset = offset - (offset - newOffset) * timeFraction;
            setAnimatedOffset(currentOffset);

            if (timeFraction < 1) {
                requestAnimationFrame(animate);
            }
        };

        requestAnimationFrame(animate);
    };

    return (
        <div>
            <svg
                height={radius * 2}
                width={radius * 2}
            >
```

```
                <circle
                    stroke='#D1D1D1'
                    fill='transparent'
                    strokeWidth={strokeWidth}
                    r={normalizedRadius}
                    cx={radius}
                    cy={radius}
                />
                <circle
                    stroke={color}
                    fill="transparent"
                    strokeWidth={strokeWidth}
                    strokeDasharray={`${circumference} ${circumference}`}
                    style={{
                        strokeDashoffset: animatedOffset,
                        transition: 'stroke-dashoffset 1s ease', // Smooth
transition

                        transform: 'rotate(-90deg)',
                        transformOrigin: '50% 50%',
                    }}
                    r={normalizedRadius}
                    cx={radius}
                    cy={radius}
                />

                <text
                    x={radius}
                    y={radius}
                    textAnchor="middle"
                    dominantBaseline="middle"
                    fontSize={radius / 2}
                    fill='#000'
                    className={mergeClasses(classes.root, texttitleclassName)}
                >
                    {percentage}%
                </text>
            </svg>
        </div>
    );
}

export default Zprogress;
```

CSS Code

```
import { makeStyles } from "@fluentui/react-components";

export const useStyles = makeStyles({
    root:{
        textAlign: "center",
        fontSize: "7px",
        fontStyle:"normal",
        fontWeight:700,
        height:"14px",
        width:"14px",
    }

});



      <Zprogress radius={25} strokeWidth={5} percentage={89} circleFactor={0.5}
texttitleclassName="Title-text" />
```

Props Required for a ZProgress Bar Core Component

1) Radius
2) Strokewidth
3) Percentage
4) CircleFactor
5) texttitleclassName

How to use this component

```
Firstly render That Component Wherver it is required
then add that are already being added in the above
code where radius is used to increase the height and
width and strokewidth is used to increase the size of
the circle , we can change dynamically the Percentage
and titleclassname is used for to change the size of
text Percentage inside a Circle and CircleFactor is
used to increase or decrease the outer circle....



possible ways to improve this component

1) The Following Ways through Which we can Improve
Our ZProgressBar such that we can dynamically change
the Percentage Through Backend Part Such That
Whenever we fill up a form if it left half then it
will show 50% , 25% , 10% and Many Color....
```