```
import React, { useMemo } from 'react';
import {
  BarChart,
  Bar,
  XAxis,
  YAxis,
  Tooltip,
  ResponsiveContainer,
  Legend,
  Area,
  AreaChart,
} from 'recharts';
import CustomTooltip from '../tooltip';
import { getColorForModel } from '../../helpers/get-color-model';
import { DailyCost, UserUsage } from '@/shared/utils/api/usage';

interface ModelUsageProps {
  usageData: UserUsage | null;
}

type ModalDataProps = {
  [model: string]: {
    requests: { date: string; value: number }[];
    tokens: { date: string; value: number }[];
    totalRequests: number;
    totalTokens: number;
```

```
  };
};

export default function ModelActivity({ usageData }: ModelUsageProps) {
  if (!usageData || !usageData?.dailyCosts?.length) {
    return (
      <div className="flex flex-col items-center justify-center border rounded-md px-2 sm:px-4 py-4
sm:py-8 text-card-foreground mt-5 mb-8 gap-2 w-full">
        <h3 className="opacity-60">
          No activities available for the current month
        </h3>
      </div>
    );
  }

  const daysInMonth = useMemo(() => {
    const firstDate = new Date(usageData.dailyCosts[0].date);
    const year = firstDate.getFullYear();
    const month = firstDate.getMonth();
    return new Date(year, month + 1, 0).getDate();
  }, [usageData]);

  const modelData: ModalDataProps = {};

  for (let day = 1; day <= daysInMonth; day++) {
    const date = new Date(usageData.dailyCosts[0].date);
```

```
date.setDate(day);
const dateString = date.toISOString().slice(0, 10);

usageData.dailyCosts.forEach((dailyCost) => {
  if (dailyCost.date === dateString) {
    Object.entries(dailyCost.model ?? '').forEach(
      ([model, { tokens, requests }]) => {
        if (!modelData[model]) {
          modelData[model] = {
            requests: [],
            tokens: [],
            totalRequests: 0,
            totalTokens: 0,
          };
        }

        const formattedDate = date.toLocaleDateString('en-US', {
          day: '2-digit',
          month: 'short',
        });

        modelData[model].requests.push({
          date: formattedDate,
          value: requests,
        });
        modelData[model].tokens.push({
```

```
        date: formattedDate,
        value: tokens,
      });
      modelData[model].totalRequests += requests;
      modelData[model].totalTokens += tokens;
    },
  );
} else {
  Object.keys(dailyCost.model ?? '').forEach((model) => {
    if (!modelData[model]) {
      modelData[model] = {
        requests: [],
        tokens: [],
        totalRequests: 0,
        totalTokens: 0,
      };
    }
    const formattedDate = date.toLocaleDateString('en-US', {
      day: '2-digit',
      month: 'short',
    });

    modelData[model].requests.push({ date: formattedDate, value: 0 });
    modelData[model].tokens.push({ date: formattedDate, value: 0 });
  });
}
```

```
  });
}

return (
  <div className="model-usage xl:mb-10 gap-8">
    {Object.keys(modelData).map((modelName) => (
      <div key={modelName} className="model-chart mb-10">
        <h4 className="my-6 font-semibold">{modelName}</h4>
        <div className="grid md:grid-cols-2">
          {/* API requests */}
          <div className="chart-section">
            <h5 className="mb-2">
              API requests{' '}
              {modelData[modelName].totalRequests.toLocaleString()}
            </h5>
            <ResponsiveContainer width="100%" height={250}>
              <AreaChart data={modelData[modelName].requests}>
                <XAxis
                  dataKey="date"
                  tick={{ fontSize: 12 }}
                  minTickGap={50}
                />
                <YAxis tick={{ fontSize: 12 }} />
                <Tooltip
                  cursor={{ opacity: 0.1, fill: 'white' }}
                  content={
```

```
          <CustomTooltip
            active={true}
            payload={[]}
            type="activity"
            label={''}
            usageData={usageData}
          />
        }
        contentStyle={{
          backgroundColor: 'rgba(255, 255, 255, 0.8)',
          border: '1px solid #e5e7eb',
          color: '#374151',
          fontSize: '12px',
        }}
      />
      <Area
        type="monotone"
        dataKey="value"
        stroke={getColorForModel(modelName, usageData)}
        fillOpacity={0.3}
        fill={getColorForModel(modelName, usageData)}
      />
    </AreaChart>
  </ResponsiveContainer>
</div>
```

```
{/* Tokens */}
<div className="chart-section">
  <h5 className="mb-2 text-gray-300">
    Tokens {modelData[modelName].totalTokens.toLocaleString()}
  </h5>
  <ResponsiveContainer width="100%" height={250}>
    <BarChart data={modelData[modelName].tokens} barSize={15}>
      <XAxis
        dataKey="date"
        tick={{ fontSize: 12 }}
        minTickGap={50}
      />
      <YAxis tick={{ fontSize: 12 }} />
      <Tooltip
        cursor={{ opacity: 0.1, fill: 'white' }}
        content={
          <CustomTooltip
            active={true}
            payload={[]}
            type="activity"
            label={''}
            usageData={usageData}
          />
        }
        contentStyle={{
          backgroundColor: 'rgba(255, 255, 255, 0.8)',
```

```
                  border: '1px solid #e5e7eb',
                  color: '#374151',
                  fontSize: '12px',
                }}
              />
              <Bar
                dataKey="value"
                fill={getColorForModel(modelName, usageData)}
                radius={[2, 2, 0, 0]}
              />
            </BarChart>
          </ResponsiveContainer>
        </div>
      </div>
    </div>
  ))}
</div>
);
}
```