Task 8 Advanced Server-Side Where Data Meets Intelligence Functionality

Objective: Implement advanced server-side features for a robust application.

Code :

```
import React, { useState, useEffect } from 'react';
import {
  Server,
  Clock,
  Database,
  Settings,
  List,
  AlertCircle,
  Activity,
  Monitor,
  HardDrive,
  BarChart
} from 'lucide-react';
import { Alert, AlertDescription, AlertTitle } from "@/components/ui/alert";
import { Card, CardContent, CardHeader, CardTitle } from '@/components/ui/card';
import { LineChart, Line, XAxis, YAxis, CartesianGrid, Tooltip, ResponsiveContainer } from 'recharts';

// Mock data for the performance chart
const performanceData = [
```

```
  { time: '00:00', cpu: 45, memory: 60, requests: 100 },
  { time: '04:00', cpu: 55, memory: 65, requests: 120 },
  { time: '08:00', cpu: 75, memory: 85, requests: 200 },
  { time: '12:00', cpu: 65, memory: 75, requests: 180 },
  { time: '16:00', cpu: 80, memory: 90, requests: 220 },
  { time: '20:00', cpu: 70, memory: 80, requests: 190 },
];

const ServerDashboard = () => {
  const [activeServers, setActiveServers] = useState(4);
  const [cpuUsage, setCpuUsage] = useState(75);
  const [memoryUsage, setMemoryUsage] = useState(60);

  // Simulate real-time updates
  useEffect(() => {
    const interval = setInterval(() => {
      setCpuUsage(prev => Math.min(100, Math.max(0, prev + Math.random() * 10 - 5)));
      setMemoryUsage(prev => Math.min(100, Math.max(0, prev + Math.random() * 10 - 5)));
    }, 3000);
    return () => clearInterval(interval);
  }, []);

  return (
    <div className="min-h-screen bg-gray-50 p-6">
      {/* Header Section */}
```

```
    <div className="mb-8">
      <h1 className="text-3xl font-bold text-gray-800 mb-2">Server Management Dashboard</h1>
      <p className="text-gray-600">Real-time monitoring and management of server-side operations</p>
    </div>

    {/* Alert Section */}
    <Alert className="mb-6 bg-blue-50 border-blue-200">
      <AlertCircle className="h-4 w-4 text-blue-600" />
      <AlertTitle className="text-blue-800">System Status</AlertTitle>
      <AlertDescription className="text-blue-700">
        All systems operational. Last updated: {new Date().toLocaleTimeString()}
      </AlertDescription>
    </Alert>

    {/* Stats Grid */}
    <div className="grid grid-cols-1 md:grid-cols-2 lg:grid-cols-4 gap-6 mb-8">
      <StatsCard
        icon={<Server className="text-purple-600" />}
        title="Active Servers"
        value={activeServers}
        trend="+1 from last hour"
      />
      <StatsCard
        icon={<Monitor className="text-blue-600" />}
        title="CPU Usage"
```

```
          value={${Math.round(cpuUsage)}%}
          trend="Normal load"
        />
        <StatsCard
          icon={<Database className="text-green-600" />}
          title="Memory Usage"
          value={${Math.round(memoryUsage)}%}
          trend="2GB available"
        />
        <StatsCard
          icon={<Activity className="text-red-600" />}
          title="Request Rate"
          value="1.2k/sec"
          trend="+15% from average"
        />
      </div>

      {/* Main Content Grid */}
      <div className="grid grid-cols-1 lg:grid-cols-2 gap-6 mb-8">
        {/* Performance Chart */}
        <Card className="col-span-2">
          <CardHeader>
            <CardTitle className="text-xl font-semibold">System
Performance</CardTitle>
          </CardHeader>
          <CardContent>
            <div className="h-80">
```

```
          <ResponsiveContainer width="100%" height="100%">
            <LineChart data={performanceData}>
              <CartesianGrid strokeDasharray="3 3" />
              <XAxis dataKey="time" />
              <YAxis />
              <Tooltip />
              <Line type="monotone" dataKey="cpu" stroke="#8b5cf6" name="CPU Usage" />
              <Line type="monotone" dataKey="memory" stroke="#10b981" name="Memory Usage" />
              <Line type="monotone" dataKey="requests" stroke="#ef4444" name="Requests/sec" />
            </LineChart>
          </ResponsiveContainer>
        </div>
      </CardContent>
    </Card>

    {/* Server Processes */}
    <Card>
      <CardHeader>
        <CardTitle className="text-xl font-semibold">Active Processes</CardTitle>
      </CardHeader>
      <CardContent>
        <div className="space-y-4">
          <ProcessItem
```

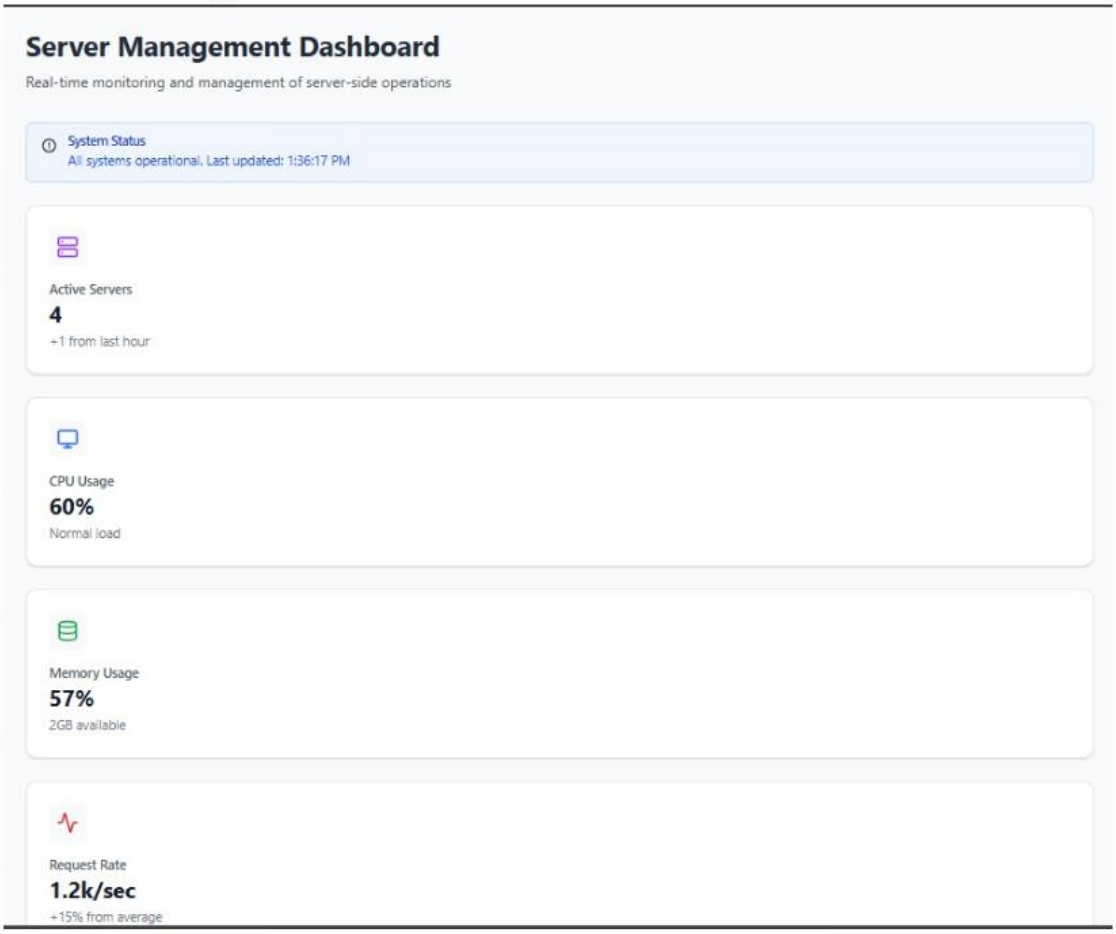
```
            name="Redis Queue Worker"
            status="Running"
            icon={<Clock />}
            uptime="5d 12h"
          />
          <ProcessItem
            name="Cache Manager"
            status="Running"
            icon={<Database />}
            uptime="12d 5h"
          />
          <ProcessItem
            name="Request Logger"
            status="Running"
            icon={<List />}
            uptime="3d 8h"
          />
        </div>
      </CardContent>
    </Card>

    {/* Storage Status */}
    <Card>
      <CardHeader>
        <CardTitle className="text-xl font-semibold">Storage
Status</CardTitle>
      </CardHeader>
```

```
        <CardContent>
          <div className="space-y-4">
            <StorageItem
              name="Primary SSD"
              used={75}
              total="500GB"
              icon={<HardDrive />}
            />
            <StorageItem
              name="Backup Drive"
              used={45}
              total="1TB"
              icon={<HardDrive />}
            />
            <StorageItem
              name="Cache Storage"
              used={30}
              total="100GB"
              icon={<BarChart />}
            />
          </div>
        </CardContent>
      </Card>
    </div>
  </div>
);
```

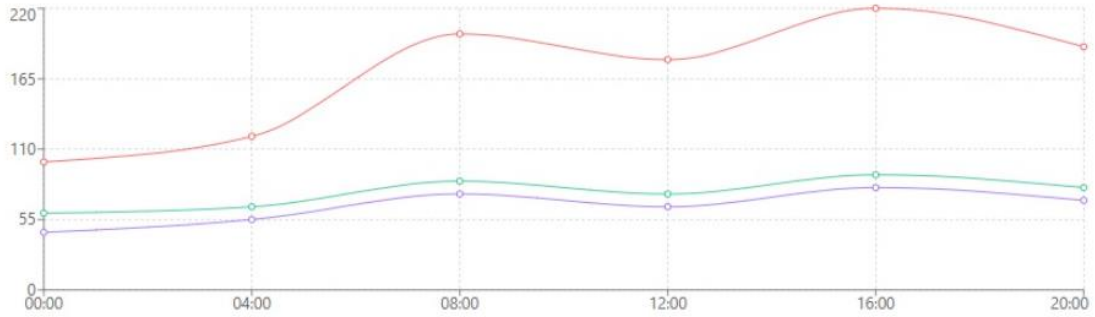
```
};

// Helper Components
const StatsCard = ({ icon, title, value, trend }) => (
  <Card>
    <CardContent className="pt-6">
      <div className="flex items-center justify-between mb-4">
        <div className="p-2 bg-gray-50 rounded-lg">{icon}</div>
      </div>
      <div>
        <p className="text-sm font-medium text-gray-600">{title}</p>
        <p className="text-2xl font-bold text-gray-900">{value}</p>
        <p className="text-sm text-gray-500 mt-1">{trend}</p>
      </div>
    </CardContent>
  </Card>
);

const ProcessItem = ({ name, status, icon, uptime }) => (
  <div className="flex items-center justify-between p-3 bg-gray-50 rounded-
lg">
    <div className="flex items-center space-x-3">
      <div className="text-gray-600">{icon}</div>
      <div>
        <p className="font-medium text-gray-900">{name}</p>
        <p className="text-sm text-gray-500">Uptime: {uptime}</p>
      </div>
```

```
    </div>
    <span className="px-3 py-1 text-sm font-medium text-green-800 bg-green-100 rounded-full">
      {status}
    </span>
  </div>
);

const StorageItem = ({ name, used, total, icon }) => (
  <div className="space-y-2">
    <div className="flex items-center justify-between">
      <div className="flex items-center space-x-2">
        <div className="text-gray-600">{icon}</div>
        <span className="font-medium text-gray-900">{name}</span>
      </div>
      <span className="text-sm text-gray-600">{used}% of {total}</span>
    </div>
    <div className="h-2 bg-gray-200 rounded-full overflow-hidden">
      <div
        className="h-full bg-blue-600 rounded-full transition-all duration-500"
        style={{ width: ${used}% }}
      />
    </div>
  </div>
);

export default ServerDashboard;
```

Output :


Server Management Dashboard
Real-time monitoring and management of server-side operations
System Status
All systems operational. Last updated: 1:36:17 PM
Active Servers
4
+1 from last hour
CPU Usage
60%
Normal load
Memory Usage
57%
2GB available
Request Rate
1.2k/sec
+15% from average

System Performance



Active Processes

Redis Queue Worker
Uptime: 5d 12h
Running

Cache Manager
Uptime: 12d 5h
Running

Request Logger
Uptime: 3d 8h
Running

Storage Status

Primary SSD 75% of 500GB

Backup Drive 45% of 1TB

Cache Storage 30% of 100GB