```
import React, { useState, useEffect } from 'react';
import axios from 'axios';
import {
  Box,
  Button,
  Card,
  Container,
  Flex,
  Grid,
  GridItem,
  Heading,
  Image,
  Link,
  SimpleGrid,
  Spinner,
  Stat,
  StatLabel,
  StatNumber,
  StatHelpText,
  StatArrow,
  Stack,
  Tab,
  TabList,
  TabPanel,
  TabPanels,
  Tabs,
  Text,
  useColorModeValue,
  useToast,
} from '@chakra-ui/react';
import {
  BarChart,
  Bar,
  LineChart,
  Line,
  XAxis,
  YAxis,
  CartesianGrid,
  Tooltip,
  Legend,
  ResponsiveContainer,
  RadarChart,
  PolarGrid,
  PolarAngleAxis,
```

```
  PolarRadiusAxis,
  Radar,
} from 'recharts';

const API_BASE_URL = process.env.REACT_APP_API_BASE_URL ||
'http://localhost:3001/api';

/**
 * DashboardComponent - Main business dashboard
 * Displays KPIs, assessment results, and AI recommendations
 */
const DashboardComponent = () => {
  const [isLoading, setIsLoading] = useState(true);
  const [dashboardData, setDashboardData] = useState(null);
  const [assessmentHistory, setAssessmentHistory] = useState([]);
  const [recommendations, setRecommendations] = useState([]);
  const toast = useToast();

  const bgCard = useColorModeValue('white', 'gray.700');
  const borderColor = useColorModeValue('gray.200', 'gray.600');

  // Fetch dashboard data on component mount
  useEffect(() => {
    const fetchDashboardData = async () => {
      try {
        setIsLoading(true);

        // Fetch all required data in parallel
        const [dashboardResponse, assessmentsResponse, recommendationsResponse] = await
Promise.all([
          axios.get(`${API_BASE_URL}/dashboard/summary`),
          axios.get(`${API_BASE_URL}/assessments/history`),
          axios.get(`${API_BASE_URL}/suggestions/latest`)
        ]);

        setDashboardData(dashboardResponse.data);
        setAssessmentHistory(assessmentsResponse.data);
        setRecommendations(recommendationsResponse.data);

        setIsLoading(false);
      } catch (error) {
        console.error('Error fetching dashboard data:', error);
        toast({
          title: 'Error loading dashboard',
```

```
        description: 'We could not load your dashboard data. Please try again later.',
        status: 'error',
        duration: 5000,
        isClosable: true,
      });
      setIsLoading(false);
    }
  };

  fetchDashboardData();
 }, [toast]);

 // Generate mock data for development if API fails
 const generateMockData = () => {
  // Mock dashboard summary data
  const mockDashboard = {
   businessName: "Acme Corporation",
   overallScore: 72,
   previousScore: 65,
   completedAssessments: 5,
   implementedSuggestions: 12,
   improvementRate: 10.8,
   departmentScores: {
    sales: 76,
    marketing: 68,
    operations: 81,
    management: 72,
    legal: 65,
    hr: 70,
    ict: 74,
    finance: 69
   }
  };

  // Mock assessment history
  const mockAssessmentHistory = [
   { date: '2025-01-15', overallScore: 65, salesScore: 62, marketingScore: 59, operationsScore:
70 },
   { date: '2025-02-20', overallScore: 67, salesScore: 65, marketingScore: 62, operationsScore:
73 },
   { date: '2025-03-25', overallScore: 69, salesScore: 68, marketingScore: 64, operationsScore:
75 },
   { date: '2025-04-30', overallScore: 70, salesScore: 72, marketingScore: 66, operationsScore:
76 },
```

```
    { date: '2025-05-15', overallScore: 72, salesScore: 76, marketingScore: 68, operationsScore: 81 },
  ];

  // Mock recommendations
  const mockRecommendations = [
    {
      title: "Standardize Sales Process Documentation",
      description: "Create detailed, step-by-step documentation for your sales process that can be consistently followed across all team members and locations.",
      impact: "Increased conversion rates and more predictable revenue forecasting",
      implementation: "Map your current sales process and identify the top-performing approaches to standardize"
    },
    {
      title: "Implement CRM System Integration",
      description: "Fully integrate your CRM system with marketing automation and customer service platforms to create a seamless customer data ecosystem.",
      impact: "360-degree customer view and improved cross-department collaboration",
      implementation: "Audit current data silos and select integration middleware that connects your existing systems"
    },
    {
      title: "Develop Operations Playbook",
      description: "Create a comprehensive operations playbook that documents all standard procedures, quality control measures, and troubleshooting protocols.",
      impact: "Consistent service delivery and faster onboarding of new staff",
      implementation: "Start with your most critical operational process and document it in detail as a template"
    },
    {
      title: "Establish KPI Dashboard System",
      description: "Implement a real-time KPI dashboard system that tracks performance metrics across all departments and makes them visible to leadership.",
      impact: "Data-driven decision making and increased accountability",
      implementation: "Define your critical KPIs for each department and set up automated data collection"
    },
    {
      title: "Create Training Certification Program",
      description: "Develop an internal certification program for key roles that ensures all team members meet the same skill standards regardless of location.",
      impact: "Consistent skill levels and reduced performance variation",
```

```
      implementation: "Identify your most critical role and develop a skills assessment and training module"
    }
  ];

  return { mockDashboard, mockAssessmentHistory, mockRecommendations };
};

// If loading fails, use mock data for development purposes
if (!dashboardData && !isLoading) {
  const { mockDashboard, mockAssessmentHistory, mockRecommendations } = generateMockData();
  setDashboardData(mockDashboard);
  setAssessmentHistory(mockAssessmentHistory);
  setRecommendations(mockRecommendations);
}

// Format department scores for radar chart
const formatDepartmentScoresForRadar = () => {
  if (!dashboardData || !dashboardData.departmentScores) return [];

  return Object.entries(dashboardData.departmentScores).map(([department, score]) => ({
    department: department.charAt(0).toUpperCase() + department.slice(1),
    score
  }));
};

// Render loading state
if (isLoading) {
  return (
    <Container maxW="container.xl" centerContent py={10}>
      <Flex direction="column" align="center" justify="center" minH="60vh">
        <Spinner size="xl" thickness="4px" speed="0.65s" color="blue.500" />
        <Text mt={6} fontSize="xl">Loading your dashboard...</Text>
      </Flex>
    </Container>
  );
}

return (
  <Container maxW="container.xl" py={6}>
    {/* Header */}
    <Flex mb={8} justify="space-between" align="center" wrap="wrap">
      <Box>
```

```
        <Heading size="lg" mb={2}>Business Dashboard</Heading>
        <Text color="gray.500">
          Welcome back to {dashboardData?.businessName || 'your business'} dashboard
        </Text>
      </Box>
      <Button colorScheme="blue" size="md" as={Link} href="/assessment/new">
        Start New Assessment
      </Button>
    </Flex>

    {/* Key Metrics */}
    <SimpleGrid columns={{ base: 1, md: 2, lg: 4 }} spacing={6} mb={8}>
      <Card p={6} bg={bgCard} borderRadius="lg" borderWidth="1px"
borderColor={borderColor} boxShadow="sm">
        <Stat>
          <StatLabel fontSize="md">Overall Business Score</StatLabel>
          <StatNumber fontSize="3xl">{dashboardData?.overallScore || 0}/100</StatNumber>
          <StatHelpText>
            <StatArrow type={dashboardData?.overallScore > dashboardData?.previousScore ?
'increase' : 'decrease'} />
            {Math.abs((dashboardData?.overallScore || 0) - (dashboardData?.previousScore || 0))}
points since last assessment
          </StatHelpText>
        </Stat>
      </Card>

      <Card p={6} bg={bgCard} borderRadius="lg" borderWidth="1px"
borderColor={borderColor} boxShadow="sm">
        <Stat>
          <StatLabel fontSize="md">Completed Assessments</StatLabel>
          <StatNumber fontSize="3xl">{dashboardData?.completedAssessments ||
0}</StatNumber>
          <StatHelpText>
            Latest: {assessmentHistory.length > 0 ? assessmentHistory[assessmentHistory.length -
1].date : 'N/A'}
          </StatHelpText>
        </Stat>
      </Card>

      <Card p={6} bg={bgCard} borderRadius="lg" borderWidth="1px"
borderColor={borderColor} boxShadow="sm">
        <Stat>
          <StatLabel fontSize="md">Implemented Suggestions</StatLabel>
```

```
          <StatNumber fontSize="3xl">{dashboardData?.implementedSuggestions ||
0}</StatNumber>
          <StatHelpText>
            From {recommendations.length || 0} total recommendations
          </StatHelpText>
        </Stat>
      </Card>

      <Card p={6} bg={bgCard} borderRadius="lg" borderWidth="1px"
borderColor={borderColor} boxShadow="sm">
        <Stat>
          <StatLabel fontSize="md">Improvement Rate</StatLabel>
          <StatNumber fontSize="3xl">{dashboardData?.improvementRate || 0}%</StatNumber>
          <StatHelpText>
            <StatArrow type="increase" />
            Month over month
          </StatHelpText>
        </Stat>
      </Card>
    </SimpleGrid>

    {/* Charts and Data Tabs */}
    <Tabs variant="enclosed" colorScheme="blue" mb={8}>
      <TabList>
        <Tab>Performance Trends</Tab>
        <Tab>Department Scores</Tab>
        <Tab>AI Recommendations</Tab>
      </TabList>

      <TabPanels>
        {/* Performance Trends Tab */}
        <TabPanel>
          <Card p={6} bg={bgCard} borderRadius="lg" borderWidth="1px"
borderColor={borderColor} boxShadow="sm">
            <Heading size="md" mb={6}>Business Performance Over Time</Heading>
            <Box height="400px">
              <ResponsiveContainer width="100%" height="100%">
                <LineChart
                  data={assessmentHistory}
                  margin={{ top: 5, right: 30, left: 20, bottom: 5 }}
                >
                  <CartesianGrid strokeDasharray="3 3" />
                  <XAxis dataKey="date" />
                  <YAxis domain={[0, 100]} />
```

```
                <Tooltip />
                <Legend />
                <Line type="monotone" dataKey="overallScore" stroke="#3182CE" name="Overall
Score" strokeWidth={2} />
                <Line type="monotone" dataKey="salesScore" stroke="#38A169" name="Sales" />
                <Line type="monotone" dataKey="marketingScore" stroke="#DD6B20"
name="Marketing" />
                <Line type="monotone" dataKey="operationsScore" stroke="#805AD5"
name="Operations" />
              </LineChart>
            </ResponsiveContainer>
          </Box>
        </Card>
      </TabPanel>

      {/* Department Scores Tab */}
      <TabPanel>
        <Card p={6} bg={bgCard} borderRadius="lg" borderWidth="1px"
borderColor={borderColor} boxShadow="sm">
          <Heading size="md" mb={6}>Department Performance Breakdown</Heading>
          <Grid templateColumns={{ base: "1fr", lg: "1fr 1fr" }} gap={8}>
            <GridItem>
              <Box height="400px">
                <ResponsiveContainer width="100%" height="100%">
                  <RadarChart outerRadius={150} data={formatDepartmentScoresForRadar()}>
                    <PolarGrid />
                    <PolarAngleAxis dataKey="department" />
                    <PolarRadiusAxis domain={[0, 100]} />
                    <Radar
                      name="Department Score"
                      dataKey="score"
                      stroke="#3182CE"
                      fill="#3182CE"
                      fillOpacity={0.6}
                    />
                  </RadarChart>
                </ResponsiveContainer>
              </Box>
            </GridItem>
            <GridItem>
              <Box height="400px">
                <ResponsiveContainer width="100%" height="100%">
                  <BarChart
                    data={formatDepartmentScoresForRadar()}
```

```
                margin={{ top: 5, right: 30, left: 20, bottom: 5 }}
              >
                <CartesianGrid strokeDasharray="3 3" />
                <XAxis dataKey="department" />
                <YAxis domain={[0, 100]} />
                <Tooltip />
                <Legend />
                <Bar dataKey="score" name="Score" fill="#3182CE" />
              </BarChart>
            </ResponsiveContainer>
          </Box>
        </GridItem>
      </Grid>
    </Card>
  </TabPanel>

  {/* AI Recommendations Tab */}
  <TabPanel>
    <Card p={6} bg={bgCard} borderRadius="lg" borderWidth="1px"
borderColor={borderColor} boxShadow="sm">
      <Heading size="md" mb={6}>AI-Powered Business Recommendations</Heading>
      <Stack spacing={4}>
        {recommendations.map((recommendation, index) => (
          <Card key={index} p={4} borderWidth="1px" borderColor={borderColor}>
            <Heading size="sm" mb={2}>{recommendation.title}</Heading>
            <Text mb={3}>{recommendation.description}</Text>
            <Grid templateColumns={{ base: "1fr", md: "1fr 1fr" }} gap={4} mt={2}>
              <Box>
                <Text fontWeight="bold" fontSize="sm">Expected Impact:</Text>
                <Text fontSize="sm">{recommendation.impact}</Text>
              </Box>
              <Box>
                <Text fontWeight="bold" fontSize="sm">First Step:</Text>
                <Text fontSize="sm">{recommendation.implementation}</Text>
              </Box>
            </Grid>
            <Flex mt={4} justify="flex-end">
              <Button size="sm" colorScheme="blue" variant="outline" mr={2}>
                Mark as Implemented
              </Button>
              <Button size="sm" colorScheme="green">
                View Related Processes
              </Button>
            </Flex>
```

```
                </Card>
              ))}
            </Stack>
          </Card>
        </TabPanel>
      </TabPanels>
    </Tabs>

    {/* Quick Actions */}
    <Card p={6} bg={bgCard} borderRadius="lg" borderWidth="1px" borderColor={borderColor}
boxShadow="sm" mb={8}>
      <Heading size="md" mb={6}>Quick Actions</Heading>
      <SimpleGrid columns={{ base: 1, md: 2, lg: 4 }} spacing={6}>
        <Button colorScheme="blue" size="lg" height="100px" as={Link} href="/processes">
          Explore Processes
        </Button>
        <Button colorScheme="green" size="lg" height="100px" as={Link}
href="/assessment/history">
          View Assessment History
        </Button>
        <Button colorScheme="purple" size="lg" height="100px" as={Link} href="/suggestions">
          All Recommendations
        </Button>
        <Button colorScheme="orange" size="lg" height="100px" as={Link} href="/export">
          Export Reports
        </Button>
      </SimpleGrid>
    </Card>

    {/* Footer */}
    <Flex justify="center" mt={10} mb={4}>
      <Image
        src="/logo.png"
        alt="FranchiseAI Logo"
        height="40px"
        fallbackSrc="https://via.placeholder.com/160x40?text=FranchiseAI"
      />
    </Flex>
    <Text textAlign="center" color="gray.500" fontSize="sm">
      © 2025 FranchiseAI. All rights reserved.
    </Text>
  </Container>
 );
};
```

```
export default DashboardComponent;
```