// Monitoring Dashboard Components

interface SystemMetrics {

api: {

health: number;

responseTime: number;

errorRate: number;

requestVolume: number;

};

users: {

active: number;

newSignups: number;

postVolume: number;

exerciseCompletions: number;

};

content: {

queueSize: number;

successRate: number;

failureRate: number;

retryCount: number;

};

}

class MonitoringDashboard extends React.Component {

state: SystemMetrics;

componentDidMount() {

this.initializeMetrics();

this.startMetricsPolling();

}

async initializeMetrics() {

const metrics = await this.fetchSystemMetrics();

this.setState(metrics);

}

render() {

return (

<DashboardLayout>

<MetricsGrid>

<SystemHealthCard metrics={this.state.api} />

<UserActivityCard metrics={this.state.users} />

<ContentStatsCard metrics={this.state.content} />

<AlertsPanel alerts={this.state.alerts} />

</MetricsGrid>

<AdminTools />

</DashboardLayout>

);

}

}

// User Notification Preferences

interface NotificationPreferences {

channels: {

email: boolean;

inApp: boolean;

push: boolean;

};

types: {

performance: boolean;

system: boolean;

exercises: boolean;

suggestions: boolean;

};

frequency: {

type: 'immediate' | 'daily' | 'weekly' | 'custom';

customSchedule?: {

days: number[];

times: string[];

};

};

smart: {

aiTiming: boolean;

priorityLevel: 1 | 2 | 3;

contextAware: boolean;

};

}

const NotificationPreferencesForm = ({ user, preferences, onUpdate }) => {

const [settings, setSettings] = useState<NotificationPreferences>(preferences);

const handleUpdate = async (updates: Partial<NotificationPreferences>) => {

const newSettings = { ...settings, ...updates };

await onUpdate(newSettings);

setSettings(newSettings);

};

return (

<Form>

<ChannelSettings

settings={settings.channels}

onUpdate={handleUpdate}

/>

<NotificationTypes

settings={settings.types}

onUpdate={handleUpdate}

/>

<FrequencyControls

settings={settings.frequency}

onUpdate={handleUpdate}

/>

<SmartSettings

settings={settings.smart}

onUpdate={handleUpdate}

/>

</Form>

);

};