Contents

RADIANT v4.18.0 - Admin Dashboard Export

Component: Next.js Admin Web Application **Framework**: Next.js 14, React 18, TypeScript **Styling**: Tailwind CSS, shadcn/ui **Files**: 120+ TSX pages/components

---

Architecture Narrative

The Admin Dashboard is a **Next.js 14 web application** providing administrators with full control over the RADIANT platform. It uses the App Router, Server Components, and modern React patterns for optimal performance and developer experience.

Technology Stack

- **Framework**: Next.js 14 (App Router)
- **UI Components**: shadcn/ui (Radix primitives)
- **Styling**: Tailwind CSS
- **State Management**: React Query (TanStack Query)

- **Forms**: React Hook Form + Zod
- **Icons**: Lucide React
- **Charts**: Recharts

Directory Structure

```
apps/admin-dashboard/
   app/                         # Next.js App Router
      (dashboard)/             # Dashboard layout group
          page.tsx             # Main dashboard
          layout.tsx           # Dashboard layout
          brain/               # AGI Brain admin (8 pages)
          cato/                # Cato Safety admin (6 pages)
          consciousness/       # Consciousness admin (7 pages)
          thinktank/           # Think Tank admin (18 pages)
          models/              # Model management (5 pages)
          security/            # Security admin (8 pages)
          orchestration/       # Orchestration (10 pages)
          ...                  # 50+ more sections
      api/                     # API routes
      layout.tsx               # Root layout
   components/                  # Reusable components
      ui/                      # shadcn/ui components (31)
      dashboard/               # Dashboard components (4)
      thinktank/               # Think Tank components (18)
      ...                      # Domain components
   lib/                         # Utilities and hooks
      hooks/                   # Custom React hooks
      api/                     # API client functions
      utils/                   # Helper utilities
   public/                      # Static assets
```

Key Pages

Main Dashboard (page.tsx)

Purpose: Platform overview with key metrics, system health, and quick actions.

```
'use client';

import { MetricCard } from '@/components/dashboard/metric-card';
import { SystemHealth } from '@/components/dashboard/system-health';
import { ActivityFeed } from '@/components/dashboard/activity-feed';
import { QuickActions } from '@/components/dashboard/quick-actions';
import { useDashboard } from '@/lib/hooks/use-dashboard';
import {
  Activity,
  DollarSign,
  Cpu,
  Users,
  Zap,
  TrendingUp
} from 'lucide-react';

export default function DashboardPage() {
  const { data, isLoading } = useDashboard();

  return (
    <div className="space-y-6">
      <div>
        <h1 className="text-3xl font-bold tracking-tight">Dashboard</h1>
        <p className="text-muted-foreground">
          RADIANT platform overview and metrics
        </p>
      </div>

      <div className="grid gap-4 md:grid-cols-2 lg:grid-cols-4">
        <MetricCard
          title="Total Requests"
          value={data?.totalRequests?.value ?? 0}
          change={data?.totalRequests?.change ?? 0}
          icon={Activity}
          loading={isLoading}
        />
        <MetricCard
          title="Active Models"
          value={data?.activeModels?.value ?? 0}
          change={0}
          icon={Cpu}
          loading={isLoading}
          format="number"
        />
        <MetricCard
          title="Revenue (MTD)"
          value={data?.revenue?.value ?? 0}
          change={data?.revenue?.change ?? 0}
          icon={DollarSign}
          loading={isLoading}
          format="currency"
        />
        <MetricCard
          title="Error Rate"
          value={data?.errorRate?.value ?? 0}
          change={data?.errorRate?.change ?? 0}
          icon={Zap}
          loading={isLoading}
        />
      </div>

      <div className="grid gap-6 md:grid-cols-2 lg:grid-cols-3">
        <div className="lg:col-span-2">
          <SystemHealth />
        </div>
        <div>
          <QuickActions />
        </div>
      </div>

      <div className="grid gap-6 md:grid-cols-2">
        <ActivityFeed />
        <div className="rounded-lg border bg-card p-6">
          <h3 className="font-semibold mb-4 flex items-center gap-2">
            <TrendingUp className="h-5 w-5" />
            Usage Trends
          </h3>
          <div className="h-[300px] flex items-center justify-center text-muted-foreground">
            Chart placeholder
          </div>
        </div>
      </div>
    </div>
  );
}
```

Page Inventory

Brain Administration (8 pages)

Page	Purpose
`brain/page.tsx`	Brain dashboard overview
`brain/config/page.tsx`	SOFAI configuration
`brain/ghost/page.tsx`	Ghost vector management
`brain/flash-facts/page.tsx`	Flash fact administration
`brain/dreams/page.tsx`	Dream cycle monitoring
`brain/routing/page.tsx`	Routing analytics
`brain/oversight/page.tsx`	Human oversight queue
`brain/calibration/page.tsx`	Trust calibration

Cato Safety (6 pages)

Page	Purpose
`cato/page.tsx`	Cato dashboard
`cato/personas/page.tsx`	Mood/persona management
`cato/cbf/page.tsx`	Control Barrier Functions
`cato/audit/page.tsx`	Safety audit logs
`cato/recovery/page.tsx`	Epistemic recovery
`cato/config/page.tsx`	Safety configuration

Consciousness Engine (7 pages)

Page	Purpose
`consciousness/page.tsx`	Consciousness dashboard
`consciousness/mcp/page.tsx`	MCP server status
`consciousness/sleep/page.tsx`	Sleep cycle configuration
`consciousness/memory/page.tsx`	Memory management
`consciousness/goals/page.tsx`	Goal tracking
`consciousness/budget/page.tsx`	Budget monitoring
`consciousness/research/page.tsx`	Deep research queue

Think Tank (18 pages)

Page	Purpose
`thinktank/artifacts/page.tsx`	Artifact management
`thinktank/collaborate/page.tsx`	Collaboration settings
`thinktank/compliance/page.tsx`	Think Tank compliance
`thinktank/conversations/page.tsx`	Conversation history
`thinktank/delight/page.tsx`	Delight engine
`thinktank/domain-modes/page.tsx`	Domain mode config
`thinktank/ego/page.tsx`	Ego system config
`thinktank/governor/page.tsx`	Economic Governor
`thinktank/grimoire/page.tsx`	Procedural memory
`thinktank/model-categories/page.tsx`	Model categories
`thinktank/my-rules/page.tsx`	User rules
`thinktank/settings/page.tsx`	Think Tank settings
`thinktank/shadow-testing/page.tsx`	Shadow testing
`thinktank/users/page.tsx`	Think Tank users

Models (5 pages)

Page	Purpose
`models/page.tsx`	Model registry
`models/providers/page.tsx`	Provider config
`models/self-hosted/page.tsx`	Self-hosted models

Page	Purpose
models/routing/page.tsx	Model routing
models/costs/page.tsx	Model costs

Security (8 pages)

Page	Purpose
security/page.tsx	Security dashboard
security/rbac/page.tsx	Role management
security/audit/page.tsx	Security audit
security/compliance/page.tsx	Compliance status
security/encryption/page.tsx	Encryption settings
security/secrets/page.tsx	Secret management
security/network/page.tsx	Network security
security/alerts/page.tsx	Security alerts

Orchestration (10 pages)

Page	Purpose
orchestration/page.tsx	Orchestration dashboard
orchestration/workflows/page.tsx	Workflow management
orchestration/patterns/page.tsx	Pattern library
orchestration/swarms/page.tsx	Swarm configuration
orchestration/consensus/page.tsx	Consensus settings
orchestration/queues/page.tsx	Queue monitoring
orchestration/timeouts/page.tsx	Timeout config
orchestration/retries/page.tsx	Retry policies
orchestration/events/page.tsx	Event history
orchestration/metrics/page.tsx	Orchestration metrics

Additional Sections (40+ pages)

Section	Pages	Purpose
administrators/	2	Admin user management
analytics/	3	Usage analytics
audit-logs/	2	System audit logs
aws-logs/	3	AWS CloudWatch integration
billing/	2	Billing management
compliance/	5	Compliance controls
configuration/	3	System configuration
costs/	1	Cost management
deployments/	2	Deployment history
experiments/	2	A/B testing

Section	Pages	Purpose
`geographic/`	2	Geographic distribution
`health/`	2	System health
`localization/`	5	i18n management
`migrations/`	2	Database migrations
`multi-region/`	2	Multi-region config
`notifications/`	2	Notification settings
`providers/`	2	Provider management
`qa/`	2	Quality assurance
`revenue/`	2	Revenue tracking
`settings/`	6	System settings
`storage/`	2	Storage management
`tenants/`	1	Tenant management
`time-machine/`	2	Time Machine admin

Component Library

shadcn/ui Components (31 files)

Component	Purpose
`button.tsx`	Button variants
`card.tsx`	Card container
`dialog.tsx`	Modal dialogs
`dropdown-menu.tsx`	Dropdown menus
`form.tsx`	Form wrapper
`input.tsx`	Input fields
`label.tsx`	Form labels
`select.tsx`	Select dropdowns
`table.tsx`	Data tables
`tabs.tsx`	Tab navigation
`toast.tsx`	Toast notifications
`tooltip.tsx`	Tooltips
`avatar.tsx`	User avatars
`badge.tsx`	Status badges
`checkbox.tsx`	Checkboxes
`command.tsx`	Command palette
`popover.tsx`	Popovers
`progress.tsx`	Progress bars
`separator.tsx`	Visual separators
`sheet.tsx`	Slide-out panels
`skeleton.tsx`	Loading skeletons
`slider.tsx`	Range sliders
`switch.tsx`	Toggle switches
`textarea.tsx`	Text areas

Component	Purpose
…	7 more

Dashboard Components (4 files)

Component	Purpose
`metric-card.tsx`	KPI metric display
`system-health.tsx`	Health status grid
`activity-feed.tsx`	Recent activity list
`quick-actions.tsx`	Common action buttons

Think Tank Components (18 files)

Component	Purpose
`brain-plan-viewer.tsx`	Brain plan visualization
`artifact-viewer.tsx`	Artifact display
`conversation-panel.tsx`	Conversation UI
`domain-mode-card.tsx`	Domain mode display
`ego-dashboard.tsx`	Ego system UI
`governor-config.tsx`	Governor settings
`grimoire-table.tsx`	Heuristics table
`model-category-card.tsx`	Model category display
`rating-panel.tsx`	Rating interface
`rule-editor.tsx`	User rule editor
`swarm-visualizer.tsx`	Swarm visualization
`user-context-panel.tsx`	User context display
…	6 more

API Integration

Custom Hooks

```
// lib/hooks/use-dashboard.ts
import { useQuery } from '@tanstack/react-query';
import { fetchDashboardData } from '@/lib/api/dashboard';

export function useDashboard() {
  return useQuery({
    queryKey: ['dashboard'],
    queryFn: fetchDashboardData,
    refetchInterval: 30000, // 30 seconds
  });
}
```

API Client

```
// lib/api/dashboard.ts
import { apiClient } from './client';

export async function fetchDashboardData() {
  const response = await apiClient.get('/api/admin/dashboard');
  return response.data;
}

export async function fetchModels() {
  const response = await apiClient.get('/api/admin/models');
  return response.data;
}

export async function updateModel(id: string, data: ModelUpdate) {
  const response = await apiClient.patch(`/api/admin/models/${id}`, data);
  return response.data;
}
```

Layout Structure

Dashboard Layout

```
// app/(dashboard)/layout.tsx
import { Sidebar } from '@/components/layout/sidebar';
import { Header } from '@/components/layout/header';
import { Breadcrumbs } from '@/components/layout/breadcrumbs';

export default function DashboardLayout({
  children,
}: {
  children: React.ReactNode;
}) {
  return (
    <div className="flex h-screen">
      <Sidebar />
      <div className="flex-1 flex flex-col overflow-hidden">
        <Header />
        <main className="flex-1 overflow-y-auto p-6">
          <Breadcrumbs />
          {children}
        </main>
      </div>
    </div>
  );
}
```

UI Patterns

Data Tables

```
import {
  Table,
  TableBody,
  TableCell,
  TableHead,
  TableHeader,
  TableRow,
} from '@/components/ui/table';

function ModelTable({ models }: { models: Model[] }) {
  return (
    <Table>
      <TableHeader>
        <TableRow>
          <TableHead>Name</TableHead>
          <TableHead>Provider</TableHead>
          <TableHead>Status</TableHead>
          <TableHead>Actions</TableHead>
        </TableRow>
      </TableHeader>
      <TableBody>
        {models.map((model) => (
          <TableRow key={model.id}>
            <TableCell>{model.name}</TableCell>
            <TableCell>{model.provider}</TableCell>
            <TableCell>
              <Badge variant={model.active ? 'default' : 'secondary'}>
                {model.active ? 'Active' : 'Inactive'}
              </Badge>
            </TableCell>
            <TableCell>
              <DropdownMenu>...</DropdownMenu>
            </TableCell>
          </TableRow>
        ))}
      </TableBody>
    </Table>
  );
}
```

Form Patterns

```
import { useForm } from 'react-hook-form';
import { zodResolver } from '@hookform/resolvers/zod';
import * as z from 'zod';

const schema = z.object({
  name: z.string().min(1, 'Name is required'),
  enabled: z.boolean(),
});

function SettingsForm() {
  const form = useForm({
    resolver: zodResolver(schema),
    defaultValues: { name: '', enabled: true },
  });

  return (
    <Form {...form}>
      <form onSubmit={form.handleSubmit(onSubmit)}>
        <FormField
          control={form.control}
          name="name"
          render={({ field }) => (
            <FormItem>
              <FormLabel>Name</FormLabel>
              <FormControl>
                <Input {...field} />
              </FormControl>
              <FormMessage />
            </FormItem>
          )}
        />
        <Button type="submit">Save</Button>
      </form>
    </Form>
  );
}
```

This concludes the Admin Dashboard export. See other export files for additional components.