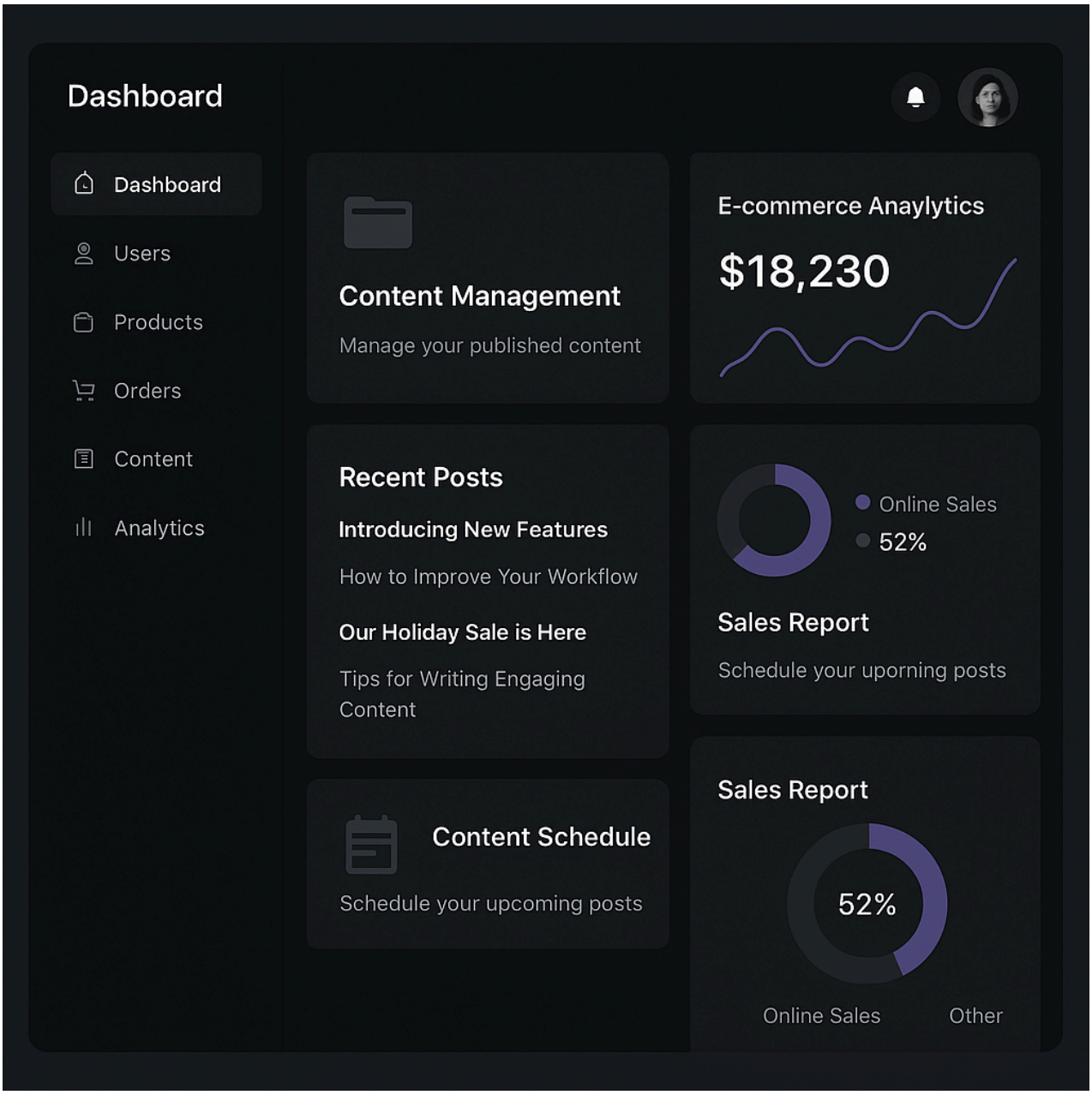
Creative Publishing Platform Admin Dashboard - Complete Implementation Guide

Based on your extensive conversation with Copilot, I'll design and create a comprehensive admin dashboard for your creative publishing platform that seamlessly integrates e-commerce, content management, and worldbuilding tools.

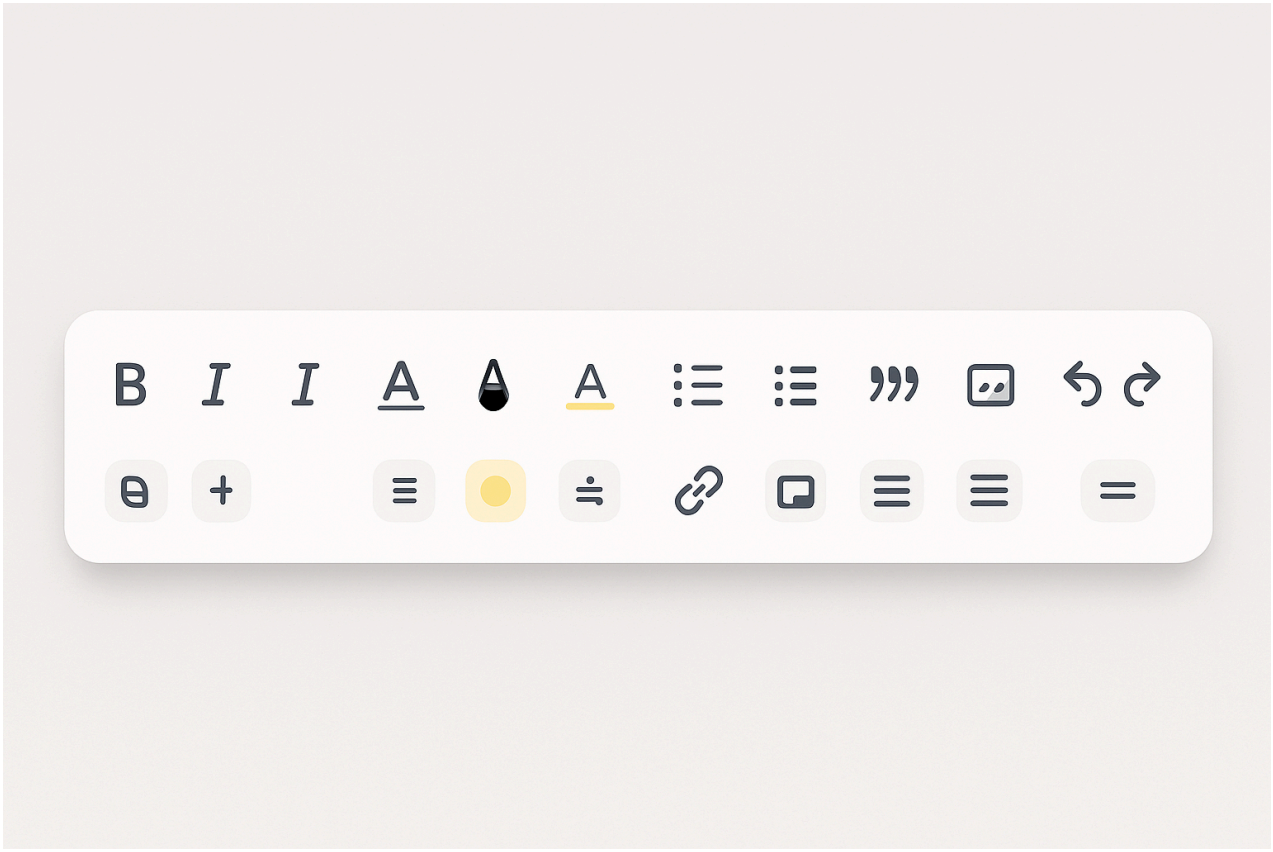
Visual Design Concepts



Admin Dashboard Design Concept

This is the admin dashboard concept I've designed for your platform. It features:

- **Dark sidebar navigation** with organized sections for all your modules
- **Analytics overview** combining business and creative metrics
- **Modern card-based layout** for easy content management
- **Unified interface** for both commerce and creative tools



Advanced Rich Text Editor Interface

Here's the advanced rich text editor interface that will power your blog posts, wiki entries, and other content creation. It includes:

- **Two-row toolbar** with comprehensive formatting options
- **Custom blocks** for info boxes, footnotes, and media
- **Column layout support** and advanced typography controls
- **Reusable component** architecture for different content types

Let me provide you with the complete implementation following your requirements:

1. Project Setup & Architecture

1.1 Directory Structure

```
project-root/
├── frontend/ (Next.js + React)
│   ├── src/
│   │   ├── app/
│   │   │   ├── account/
│   │   │   │   ├── admin/
│   │   │   │   │   ├── analytics/
│   │   │   │   │   ├── content/
│   │   │   │   │   ├── world/
│   │   │   │   │   ├── commerce/
│   │   │   │   │   └── beta/
│   │   │   │   └── page.tsx
│   │   │   ├── login/
│   │   │   └── layout.tsx
│   │   ├── components/
│   │   │   ├── admin/
│   │   │   ├── editor/
│   │   │   └── ui/
│   │   ├── context/
│   │   ├── lib/
│   │   └── hooks/
└── backend/ (Express.js API - optional)
```

1.2 Install Dependencies

```
# Frontend (Next.js)
cd frontend
npm create next-app@latest . --typescript --tailwind --app --src-dir --import-alias "@/*"

# Core dependencies
npm install @supabase/supabase-js @supabase/auth-helpers-nextjs

# Rich text editor
npm install @tiptap/react @tiptap/starter-kit @tiptap/extension-image @tiptap/extension-l

# UI and charts
npm install recharts lucide-react clsx tailwind-merge class-variance-authority
npm install @headlessui/react @heroicons/react

# Utilities
npm install date-fns jwt-decode axios
```

2. Supabase Configuration

2.1 Environment Setup

```
# .env.local
NEXT_PUBLIC_SUPABASE_URL=https://your-project.supabase.co
NEXT_PUBLIC_SUPABASE_ANON_KEY=your-anon-key
SUPABASE_SERVICE_ROLE_KEY=your-service-role-key
JWT_SECRET=your-jwt-secret
```

2.2 Database Schema

```
-- profiles table for user roles
CREATE TABLE profiles (
  id UUID REFERENCES auth.users(id) PRIMARY KEY,
  email TEXT,
  role TEXT DEFAULT 'user' CHECK (role IN ('user', 'editor', 'manager', 'admin')),
  privileges TEXT[] DEFAULT '{}',
  created_at TIMESTAMPTZ DEFAULT NOW(),
  updated_at TIMESTAMPTZ DEFAULT NOW()
);

-- Enable RLS
ALTER TABLE profiles ENABLE ROW LEVEL SECURITY;

-- Policies
CREATE POLICY "Users can view own profile" ON profiles
  FOR SELECT USING (auth.uid() = id);

CREATE POLICY "Users can update own profile" ON profiles
  FOR UPDATE USING (auth.uid() = id);

-- Content tables
CREATE TABLE blog_posts (
  id UUID DEFAULT gen_random_uuid() PRIMARY KEY,
  title TEXT NOT NULL,
  slug TEXT UNIQUE NOT NULL,
  content JSONB,
  excerpt TEXT,
  cover_url TEXT,
  status TEXT DEFAULT 'draft',
  published_at TIMESTAMPTZ,
  created_at TIMESTAMPTZ DEFAULT NOW(),
  updated_at TIMESTAMPTZ DEFAULT NOW(),
  author_id UUID REFERENCES profiles(id)
);

CREATE TABLE wiki_entries (
  id UUID DEFAULT gen_random_uuid() PRIMARY KEY,
  title TEXT NOT NULL,
  slug TEXT UNIQUE NOT NULL,
  content JSONB,
  tags TEXT[],
```

```
  created_at TIMESTAMPTZ DEFAULT NOW(),
  updated_at TIMESTAMPTZ DEFAULT NOW()
);

CREATE TABLE characters (
  id UUID DEFAULT gen_random_uuid() PRIMARY KEY,
  name TEXT NOT NULL,
  slug TEXT UNIQUE NOT NULL,
  bio JSONB,
  avatar_url TEXT,
  tags TEXT[],
  created_at TIMESTAMPTZ DEFAULT NOW(),
  updated_at TIMESTAMPTZ DEFAULT NOW()
);

CREATE TABLE timeline_events (
  id UUID DEFAULT gen_random_uuid() PRIMARY KEY,
  title TEXT NOT NULL,
  occurred_at TIMESTAMPTZ,
  description TEXT,
  links JSONB,
  created_at TIMESTAMPTZ DEFAULT NOW()
);

CREATE TABLE artist_collaborations (
  id UUID DEFAULT gen_random_uuid() PRIMARY KEY,
  title TEXT NOT NULL,
  brief JSONB,
  status TEXT DEFAULT 'draft',
  due_at TIMESTAMPTZ,
  files JSONB,
  payout_cents INTEGER,
  created_at TIMESTAMPTZ DEFAULT NOW(),
  updated_at TIMESTAMPTZ DEFAULT NOW()
);

CREATE TABLE beta_applications (
  id UUID DEFAULT gen_random_uuid() PRIMARY KEY,
  applicant_email TEXT NOT NULL,
  answers JSONB,
  status TEXT DEFAULT 'received',
  created_at TIMESTAMPTZ DEFAULT NOW(),
  updated_at TIMESTAMPTZ DEFAULT NOW()
);

CREATE TABLE ebooks (
  id UUID DEFAULT gen_random_uuid() PRIMARY KEY,
  title TEXT NOT NULL,
  slug TEXT UNIQUE NOT NULL,
  price_cents INTEGER,
  cover_url TEXT,
  description TEXT,
  status TEXT DEFAULT 'draft',
  created_at TIMESTAMPTZ DEFAULT NOW(),
  updated_at TIMESTAMPTZ DEFAULT NOW()
);
```

```
CREATE TABLE orders (
  id UUID DEFAULT gen_random_uuid() PRIMARY KEY,
  customer_id UUID REFERENCES profiles(id),
  total_cents INTEGER,
  status TEXT DEFAULT 'pending',
  items JSONB,
  created_at TIMESTAMPTZ DEFAULT NOW()
);

-- Analytics events
CREATE TABLE analytics_events (
  id UUID DEFAULT gen_random_uuid() PRIMARY KEY,
  user_id UUID REFERENCES profiles(id),
  session_id TEXT,
  event_type TEXT,
  target_id TEXT,
  metadata JSONB,
  occurred_at TIMESTAMPTZ DEFAULT NOW()
);
```

3. Core Authentication & Context

3.1 Supabase Client

```
// src/lib/supabase.ts
import { createClientComponentClient, createServerComponentClient } from '@supabase/auth-
import { createClient } from '@supabase/supabase-js'
import { cookies } from 'next/headers'

export const createSupabaseClient = () =>
  createClientComponentClient()

export const createSupabaseServerClient = () =>
  createServerComponentClient({ cookies })

export const supabaseAdmin = createClient(
  process.env.NEXT_PUBLIC_SUPABASE_URL!,
  process.env.SUPABASE_SERVICE_ROLE_KEY!,
  {
    auth: {
      autoRefreshToken: false,
      persistSession: false
    }
  }
)
```

3.2 Auth Context

```
// src/context/AuthContext.tsx
'use client'
import { createContext, useContext, useEffect, useState, ReactNode } from 'react'
import { User } from '@supabase/auth-helpers-nextjs'
import { createSupabaseClient } from '@/lib/supabase'

interface Profile {
  id: string
  email: string
  role: 'user' | 'editor' | 'manager' | 'admin'
  privileges: string[]
}

interface AuthContextType {
  user: User | null
  profile: Profile | null
  loading: boolean
  signOut: () => Promise<void>
}

const AuthContext = createContext<AuthContextType>({
  user: null,
  profile: null,
  loading: true,
  signOut: async () => {}
})

export function AuthProvider({ children }: { children: ReactNode }) {
  const [user, setUser] = useState<User | null>(null)
  const [profile, setProfile] = useState<Profile | null>(null)
  const [loading, setLoading] = useState(true)
  const supabase = createSupabaseClient()

  useEffect(() => {
    const getInitialSession = async () => {
      const { data: { session } } = await supabase.auth.getSession()
      setUser(session?.user ?? null)

      if (session?.user) {
        const { data: profileData } = await supabase
          .from('profiles')
          .select('*')
          .eq('id', session.user.id)
          .single()

        setProfile(profileData)
      }
      setLoading(false)
    }

    getInitialSession()

    const { data: { subscription } } = supabase.auth.onAuthStateChange(
      async (event, session) => {
```

```
        setUser(session?.user ?? null)

        if (session?.user) {
          const { data: profileData } = await supabase
            .from('profiles')
            .select('*')
            .eq('id', session.user.id)
            .single()

          setProfile(profileData)
        } else {
          setProfile(null)
        }
        setLoading(false)
      }
    )

    return () => subscription.unsubscribe()
  }, [])

  const signOut = async () => {
    await supabase.auth.signOut()
  }

  return (
    <AuthContext.Provider value={{ user, profile, loading, signOut }}>
      {children}
    </AuthContext.Provider>
  )
}

export const useAuth = () => {
  const context = useContext(AuthContext)
  if (!context) {
    throw new Error('useAuth must be used within AuthProvider')
  }
  return context
}
```

4. Advanced Rich Text Editor Components

4.1 Custom Tiptap Extensions

```
// src/components/editor/extensions/InfoBox.ts
import { Node, mergeAttributes } from '@tiptap/core'

export const InfoBox = Node.create({
  name: 'infoBox',
  group: 'block',
  content: 'inline*',

  parseHTML() {
    return [{ tag: 'div[data-type="info-box"]' }]
  },
```

```
  renderHTML({ HTMLAttributes }) {
    return [
      'div',
      mergeAttributes(HTMLAttributes, {
        'data-type': 'info-box',
        class: 'bg-blue-50 border-l-4 border-blue-400 p-4 my-4 rounded-r-md'
      }),
      0
    ]
  },

  addCommands() {
    return {
      setInfoBox: () => ({ commands }) => {
        return commands.insertContent({
          type: this.name,
          content: [{ type: 'paragraph', content: [{ type: 'text', text: 'Info box conten
        })
      }
    }
  }
})
```

```
// src/components/editor/extensions/Footnote.ts
import { Mark } from '@tiptap/core'

export const Footnote = Mark.create({
  name: 'footnote',

  addAttributes() {
    return {
      number: { default: 1 },
      content: { default: '' }
    }
  },

  parseHTML() {
    return [{ tag: 'sup[data-footnote]' }]
  },

  renderHTML({ HTMLAttributes }) {
    return [
      'sup',
      mergeAttributes(HTMLAttributes, {
        'data-footnote': HTMLAttributes.number,
        class: 'text-blue-600 cursor-pointer hover:underline'
      }),
      HTMLAttributes.number
    ]
  },

  addCommands() {
    return {
      setFootnote: (attributes) => ({ chain }) => {
```

```
        return chain()
          .setMark(this.name, attributes)
          .run()
      }
    }
  }
})
```

```
// src/components/editor/extensions/Columns.ts
import { Node, mergeAttributes } from '@tiptap/core'

export const Columns = Node.create({
  name: 'columns',
  group: 'block',
  content: 'column+',

  parseHTML() {
    return [{ tag: 'div[data-type="columns"]' }]
  },

  renderHTML({ HTMLAttributes }) {
    return [
      'div',
      mergeAttributes(HTMLAttributes, {
        'data-type': 'columns',
        class: 'grid grid-cols-2 gap-4 my-4'
      }),
      0
    ]
  },

  addCommands() {
    return {
      setColumns: () => ({ commands }) => {
        return commands.insertContent({
          type: this.name,
          content: [
            { type: 'column', content: [{ type: 'paragraph' }] },
            { type: 'column', content: [{ type: 'paragraph' }] }
          ]
        })
      }
    }
  }
})

export const Column = Node.create({
  name: 'column',
  content: 'block+',

  parseHTML() {
    return [{ tag: 'div[data-type="column"]' }]
  },

  renderHTML({ HTMLAttributes }) {
```

```
    return [
      'div',
      mergeAttributes(HTMLAttributes, {
        'data-type': 'column',
        class: 'column'
      }),
      0
    ]
  }
})
```

4.2 Advanced Editor Component

```
// src/components/editor/AdvancedEditor.tsx
'use client'
import { useEditor, EditorContent } from '@tiptap/react'
import StarterKit from '@tiptap/starter-kit'
import Image from '@tiptap/extension-image'
import Link from '@tiptap/extension-link'
import TextAlign from '@tiptap/extension-text-align'
import TextStyle from '@tiptap/extension-text-style'
import Color from '@tiptap/extension-color'
import Table from '@tiptap/extension-table'
import TableRow from '@tiptap/extension-table-row'
import TableCell from '@tiptap/extension-table-cell'
import TableHeader from '@tiptap/extension-table-header'
import { InfoBox } from './extensions/InfoBox'
import { Footnote } from './extensions/Footnote'
import { Columns, Column } from './extensions/Columns'
import { EditorToolbar } from './EditorToolbar'

interface AdvancedEditorProps {
  content?: string
  onChange?: (content: string) => void
  placeholder?: string
  className?: string
}

export function AdvancedEditor({
  content = '',
  onChange,
  placeholder = 'Start writing...',
  className = ''
}: AdvancedEditorProps) {
  const editor = useEditor({
    extensions: [
      StarterKit.configure({
        heading: { levels: [1, 2, 3, 4, 5, 6] }
      }),
      Image.configure({
        inline: false,
        allowBase64: true,
        HTMLAttributes: {
          class: 'max-w-full h-auto rounded-lg'
        }
```

```
      }),
      Link.configure({
        openOnClick: false,
        HTMLAttributes: {
          class: 'text-blue-600 underline hover:text-blue-800'
        }
      }),
      TextAlign.configure({
        types: ['heading', 'paragraph'],
        alignments: ['left', 'center', 'right', 'justify']
      }),
      TextStyle,
      Color,
      Table.configure({ resizable: true }),
      TableRow,
      TableCell,
      TableHeader,
      InfoBox,
      Footnote,
      Columns,
      Column
    ],
    content,
    onUpdate: ({ editor }) => {
      onChange?.(editor.getHTML())
    },
    editorProps: {
      attributes: {
        class: `prose prose-lg max-w-none focus:outline-none p-4 min-h-[400px] ${classNam
      }
    }
  })

  if (!editor) return null

  return (
    <div className="border rounded-lg bg-white">
      <EditorToolbar editor={editor} />
      <div className="border-t">
        <EditorContent editor={editor} />
      </div>
    </div>
  )
}
```

4.3 Editor Toolbar

```
// src/components/editor/EditorToolbar.tsx
'use client'
import { Editor } from '@tiptap/react'
import {
  Bold,
  Italic,
  Underline,
  Strikethrough,
```

```
  AlignLeft,
  AlignCenter,
  AlignRight,
  AlignJustify,
  List,
  ListOrdered,
  Quote,
  Code,
  Link,
  Image,
  Table,
  Info,
  Columns,
  Type,
  Palette
} from 'lucide-react'

interface EditorToolbarProps {
  editor: Editor
}

export function EditorToolbar({ editor }: EditorToolbarProps) {
  const addImage = () => {
    const url = window.prompt('Image URL')
    if (url) {
      editor.chain().focus().setImage({ src: url }).run()
    }
  }

  const addLink = () => {
    const url = window.prompt('URL')
    if (url) {
      editor.chain().focus().setLink({ href: url }).run()
    }
  }

  const addFootnote = () => {
    const number = window.prompt('Footnote number')
    const content = window.prompt('Footnote content')
    if (number && content) {
      editor.chain().focus().setFootnote({ number, content }).run()
    }
  }

  return (
    <div className="flex flex-wrap gap-1 p-2 border-b bg-gray-50">
      {/* Row 1: Text Formatting */}
      <div className="flex items-center gap-1 pr-2 border-r">
        <select
          className="text-sm border rounded px-2 py-1"
          onChange={(e) => {
            const level = parseInt(e.target.value)
            if (level === 0) {
              editor.chain().focus().setParagraph().run()
            } else {
              editor.chain().focus().toggleHeading({ level: level as 1 | 2 | 3 | 4 | 5 |
```

```
            }
          }}
        >
          <option value="0">Paragraph</option>
          <option value="1">Heading 1</option>
          <option value="2">Heading 2</option>
          <option value="3">Heading 3</option>
          <option value="4">Heading 4</option>
          <option value="5">Heading 5</option>
          <option value="6">Heading 6</option>
        </select>

        <select className="text-sm border rounded px-2 py-1 ml-2">
          <option>Inter</option>
          <option>Georgia</option>
          <option>Times</option>
          <option>Arial</option>
        </select>

        <select className="text-sm border rounded px-2 py-1 ml-2">
          <option>14px</option>
          <option>12px</option>
          <option>16px</option>
          <option>18px</option>
          <option>20px</option>
        </select>
      </div>

      <div className="flex items-center gap-1 pr-2 border-r">
        <button
          onClick={() => editor.chain().focus().toggleBold().run()}
          className={`p-1 rounded ${editor.isActive('bold') ? 'bg-blue-200' : 'hover:bg-g
        >
          <Bold size={16} />
        </button>
        <button
          onClick={() => editor.chain().focus().toggleItalic().run()}
          className={`p-1 rounded ${editor.isActive('italic') ? 'bg-blue-200' : 'hover:bg
        >
          <Italic size={16} />
        </button>
        <button
          onClick={() => editor.chain().focus().toggleUnderline().run()}
          className={`p-1 rounded ${editor.isActive('underline') ? 'bg-blue-200' : 'hover
        >
          <Underline size={16} />
        </button>
        <button
          onClick={() => editor.chain().focus().toggleStrike().run()}
          className={`p-1 rounded ${editor.isActive('strike') ? 'bg-blue-200' : 'hover:bg
        >
          <Strikethrough size={16} />
        </button>

        <input
          type="color"
```

```
          onInput={(e) => editor.chain().focus().setColor((e.target as HTMLInputElement).
          className="w-8 h-8 border rounded"
          title="Text Color"
        />
      </div>

      {/* Row 2: Layout & Content */}
      <div className="flex items-center gap-1 pr-2 border-r">
        <button
          onClick={() => editor.chain().focus().setTextAlign('left').run()}
          className={`p-1 rounded ${editor.isActive({ textAlign: 'left' }) ? 'bg-blue-200
        >
          <AlignLeft size={16} />
        </button>
        <button
          onClick={() => editor.chain().focus().setTextAlign('center').run()}
          className={`p-1 rounded ${editor.isActive({ textAlign: 'center' }) ? 'bg-blue-2
        >
          <AlignCenter size={16} />
        </button>
        <button
          onClick={() => editor.chain().focus().setTextAlign('right').run()}
          className={`p-1 rounded ${editor.isActive({ textAlign: 'right' }) ? 'bg-blue-20
        >
          <AlignRight size={16} />
        </button>
        <button
          onClick={() => editor.chain().focus().setTextAlign('justify').run()}
          className={`p-1 rounded ${editor.isActive({ textAlign: 'justify' }) ? 'bg-blue-
        >
          <AlignJustify size={16} />
        </button>
      </div>

      <div className="flex items-center gap-1 pr-2 border-r">
        <button
          onClick={addImage}
          className="p-1 rounded hover:bg-gray-200"
          title="Insert Image"
        >
          <Image size={16} />
        </button>
        <button
          onClick={() => editor.chain().focus().setInfoBox().run()}
          className="p-1 rounded hover:bg-gray-200"
          title="Info Box"
        >
          <Info size={16} />
        </button>
        <button
          onClick={() => editor.chain().focus().insertTable({ rows: 3, cols: 3, withHeade
          className="p-1 rounded hover:bg-gray-200"
          title="Insert Table"
        >
          <Table size={16} />
        </button>
```

```
        <button
          onClick={() => editor.chain().focus().setColumns().run()}
          className="p-1 rounded hover:bg-gray-200"
          title="2 Columns"
        >
          <Columns size={16} />
        </button>
        <button
          onClick={addFootnote}
          className="p-1 rounded hover:bg-gray-200"
          title="Footnote"
        >
          <Type size={16} />
        </button>
        <button
          onClick={addLink}
          className="p-1 rounded hover:bg-gray-200"
          title="Add Link"
        >
          <Link size={16} />
        </button>
      </div>

      <div className="flex items-center gap-1">
        <button
          onClick={() => editor.chain().focus().toggleBulletList().run()}
          className={`p-1 rounded ${editor.isActive('bulletList') ? 'bg-blue-200' : 'hove
        >
          <List size={16} />
        </button>
        <button
          onClick={() => editor.chain().focus().toggleOrderedList().run()}
          className={`p-1 rounded ${editor.isActive('orderedList') ? 'bg-blue-200' : 'hov
        >
          <ListOrdered size={16} />
        </button>
        <button
          onClick={() => editor.chain().focus().toggleBlockquote().run()}
          className={`p-1 rounded ${editor.isActive('blockquote') ? 'bg-blue-200' : 'hove
        >
          <Quote size={16} />
        </button>
        <button
          onClick={() => editor.chain().focus().toggleCode().run()}
          className={`p-1 rounded ${editor.isActive('code') ? 'bg-blue-200' : 'hover:bg-g
        >
          <Code size={16} />
        </button>
      </div>
    </div>
  )
}
```

5. Admin Dashboard Layout & Components

5.1 Protected Route Component

```
// src/components/ProtectedRoute.tsx
'use client'
import { useAuth } from '@/context/AuthContext'
import { useRouter } from 'next/navigation'
import { useEffect } from 'react'

interface ProtectedRouteProps {
  children: React.ReactNode
  requiredRole?: 'user' | 'editor' | 'manager' | 'admin'
  requiredPrivileges?: string[]
}

export function ProtectedRoute({
  children,
  requiredRole,
  requiredPrivileges = []
}: ProtectedRouteProps) {
  const { user, profile, loading } = useAuth()
  const router = useRouter()

  useEffect(() => {
    if (!loading) {
      if (!user) {
        router.push('/login')
        return
      }

      if (requiredRole && profile?.role !== requiredRole) {
        router.push('/unauthorized')
        return
      }

      if (requiredPrivileges.length > 0) {
        const hasAllPrivileges = requiredPrivileges.every(privilege =>
          profile?.privileges.includes(privilege)
        )

        if (!hasAllPrivileges) {
          router.push('/unauthorized')
          return
        }
      }
    }
  }, [loading, user, profile, router, requiredRole, requiredPrivileges])

  if (loading) {
    return <div className="flex justify-center items-center h-screen">Loading...</div>
  }

  if (!user) return null
  if (requiredRole && profile?.role !== requiredRole) return null
```

```
  return <>{children}</>
}
```

5.2 Admin Layout with Sidebar

```
// src/components/admin/AdminLayout.tsx
'use client'
import { useState } from 'react'
import { useAuth } from '@/context/AuthContext'
import {
  BarChart3,
  FileText,
  Globe,
  ShoppingCart,
  Users,
  Settings,
  BookOpen,
  Clock,
  UserCircle,
  Palette,
  TestTube,
  Menu,
  X
} from 'lucide-react'
import Link from 'next/link'
import { usePathname } from 'next/navigation'

const sidebarItems = [
  { label: 'Dashboard', href: '/account/admin', icon: BarChart3 },
  { label: 'Analytics', href: '/account/admin/analytics', icon: BarChart3 },
  {
    label: 'Content',
    icon: FileText,
    children: [
      { label: 'Pages', href: '/account/admin/content/pages' },
      { label: 'Blog', href: '/account/admin/content/blog' },
      { label: 'Files', href: '/account/admin/content/files' },
      { label: 'Chapters', href: '/account/admin/content/chapters' },
      { label: 'Homepage', href: '/account/admin/content/homepage' },
      { label: 'About', href: '/account/admin/content/about' }
    ]
  },
  {
    label: 'World',
    icon: Globe,
    children: [
      { label: 'Wiki', href: '/account/admin/world/wiki' },
      { label: 'Timelines', href: '/account/admin/world/timelines' },
      { label: 'Characters', href: '/account/admin/world/characters' },
      { label: 'Artist Collab', href: '/account/admin/world/artists' }
    ]
  },
  {
    label: 'Commerce',
```

```
    icon: ShoppingCart,
    children: [
      { label: 'Shop', href: '/account/admin/commerce/shop' },
      { label: 'Orders', href: '/account/admin/commerce/orders' },
      { label: 'Customers', href: '/account/admin/commerce/customers' }
    ]
  },
  {
    label: 'Beta',
    icon: TestTube,
    children: [
      { label: 'Applications', href: '/account/admin/beta/applications' },
      { label: 'Managers', href: '/account/admin/beta/managers' },
      { label: 'Activity', href: '/account/admin/beta/activity' }
    ]
  },
  {
    label: 'Settings',
    icon: Settings,
    children: [
      { label: 'Users', href: '/account/admin/settings/users' },
      { label: 'Roles', href: '/account/admin/settings/roles' },
      { label: 'Integrations', href: '/account/admin/settings/integrations' }
    ]
  }
]

export function AdminLayout({ children }: { children: React.ReactNode }) {
  const [sidebarOpen, setSidebarOpen] = useState(false)
  const { profile, signOut } = useAuth()
  const pathname = usePathname()

  return (
    <div className="flex h-screen bg-gray-100">
      {/* Sidebar */}
      <div className={`fixed inset-y-0 left-0 z-50 w-64 bg-gray-900 transform ${sidebarOp
        <div className="flex items-center justify-between h-16 px-4 bg-gray-800">
          <h1 className="text-white font-semibold">Creative Admin</h1>
          <button
            onClick={() => setSidebarOpen(false)}
            className="lg:hidden text-gray-300 hover:text-white"
          >
            <X size={24} />
          </button>
        </div>

        <nav className="mt-5 px-2 space-y-1">
          {sidebarItems.map((item) => (
            <div key={item.label}>
              {item.children ? (
                <div>
                  <div className="flex items-center px-2 py-2 text-sm font-medium text-gr
                    <item.icon className="mr-3 h-5 w-5" />
                    {item.label}
                  </div>
                  <div className="ml-8 space-y-1">
```

```
                {item.children.map((child) => (
                  <Link
                    key={child.href}
                    href={child.href}
                    className={`block px-2 py-1 text-sm rounded-md ${
                      pathname === child.href
                        ? 'bg-gray-800 text-white'
                        : 'text-gray-300 hover:bg-gray-700 hover:text-white'
                    }`}
                  >
                    {child.label}
                  </Link>
                ))}
              </div>
            </div>
          ) : (
            <Link
              href={item.href}
              className={`flex items-center px-2 py-2 text-sm font-medium rounded-md
                pathname === item.href
                  ? 'bg-gray-800 text-white'
                  : 'text-gray-300 hover:bg-gray-700 hover:text-white'
              }`}
            >
              <item.icon className="mr-3 h-5 w-5" />
              {item.label}
            </Link>
          )}
        </div>
      ))}
    </nav>
  </div>

  {/* Main Content */}
  <div className="flex flex-col flex-1 lg:ml-0">
    {/* Top Bar */}
    <header className="bg-white shadow-sm border-b">
      <div className="flex items-center justify-between h-16 px-4">
        <button
          onClick={() => setSidebarOpen(true)}
          className="lg:hidden text-gray-600 hover:text-gray-900"
        >
          <Menu size={24} />
        </button>

        <div className="flex items-center space-x-4">
          <input
            type="search"
            placeholder="Quick search..."
            className="border border-gray-300 rounded-md px-3 py-2 text-sm w-64 focus
          />

          <div className="flex items-center space-x-2">
            <span className="text-sm text-gray-700">
              {profile?.email} ({profile?.role})
            </span>
```

```
                <button
                  onClick={signOut}
                  className="text-sm text-gray-600 hover:text-gray-900"
                >
                  Sign out
                </button>
              </div>
            </div>
          </div>
        </header>

        {/* Main Content Area */}
        <main className="flex-1 overflow-y-auto p-6">
          {children}
        </main>
      </div>

      {/* Overlay for mobile */}
      {sidebarOpen && (
        <div
          className="fixed inset-0 z-40 bg-gray-600 bg-opacity-75 lg:hidden"
          onClick={() => setSidebarOpen(false)}
        />
      )}
    </div>
  )
}
```

5.3 Dashboard Analytics Components

```
// src/components/admin/KpiCard.tsx
interface KpiCardProps {
  title: string
  value: string | number
  trend?: string
  color?: 'blue' | 'green' | 'purple' | 'red'
}

export function KpiCard({ title, value, trend, color = 'blue' }: KpiCardProps) {
  const colorClasses = {
    blue: 'bg-blue-50 border-blue-200',
    green: 'bg-green-50 border-green-200',
    purple: 'bg-purple-50 border-purple-200',
    red: 'bg-red-50 border-red-200'
  }

  return (
    <div className={`p-6 rounded-lg border ${colorClasses[color]}`}>
      <h3 className="text-sm font-medium text-gray-600">{title}</h3>
      <div className="mt-2 flex items-baseline">
        <p className="text-2xl font-semibold text-gray-900">{value}</p>
        {trend && (
          <p className="ml-2 text-sm font-medium text-green-600">{trend}</p>
        )}
      </div>
```

```
    </div>
  )
}
```

```
// src/components/admin/ChartsSection.tsx
'use client'
import { LineChart, Line, XAxis, YAxis, CartesianGrid, Tooltip, ResponsiveContainer, BarC

const blogViewsData = [
  { date: 'Mon', views: 400 },
  { date: 'Tue', views: 520 },
  { date: 'Wed', views: 800 },
  { date: 'Thu', views: 620 },
  { date: 'Fri', views: 700 },
  { date: 'Sat', views: 900 },
  { date: 'Sun', views: 860 }
]

const chaptersReadData = [
  { date: 'Mon', reads: 300 },
  { date: 'Tue', reads: 360 },
  { date: 'Wed', reads: 540 },
  { date: 'Thu', reads: 820 },
  { date: 'Fri', reads: 610 },
  { date: 'Sat', reads: 700 },
  { date: 'Sun', reads: 410 }
]

export function ChartsSection() {
  return (
    <div className="grid grid-cols-1 lg:grid-cols-2 gap-6">
      {/* Blog Views Chart */}
      <div className="bg-white p-6 rounded-lg border">
        <h3 className="text-lg font-semibold text-gray-900 mb-4">Blog Views (7 days)</h3>
        <ResponsiveContainer width="100%" height={300}>
          <LineChart data={blogViewsData}>
            <CartesianGrid strokeDasharray="3 3" />
            <XAxis dataKey="date" />
            <YAxis />
            <Tooltip />
            <Line
              type="monotone"
              dataKey="views"
              stroke="#2563eb"
              strokeWidth={2}
              dot={{ fill: '#2563eb' }}
            />
          </LineChart>
        </ResponsiveContainer>
      </div>

      {/* Chapters Read Chart */}
      <div className="bg-white p-6 rounded-lg border">
        <h3 className="text-lg font-semibold text-gray-900 mb-4">Chapters Read (7 days)</
        <ResponsiveContainer width="100%" height={300}>
```

```
          <BarChart data={chaptersReadData}>
            <CartesianGrid strokeDasharray="3 3" />
            <XAxis dataKey="date" />
            <YAxis />
            <Tooltip />
            <Bar dataKey="reads" fill="#7c3aed" />
          </BarChart>
        </ResponsiveContainer>
      </div>
    </div>
  )
}
```

6. Main Dashboard & Routes

6.1 Profile Page (Entry Point)

```
// src/app/account/page.tsx
'use client'
import { useAuth } from '@/context/AuthContext'
import Link from 'next/link'

export default function ProfilePage() {
  const { user, profile } = useAuth()

  return (
    <div className="max-w-4xl mx-auto p-6">
      <h1 className="text-2xl font-bold text-gray-900 mb-6">My Profile</h1>

      <div className="bg-white rounded-lg border p-6 mb-6">
        <div className="grid grid-cols-1 md:grid-cols-2 gap-6">
          <div>
            <label className="block text-sm font-medium text-gray-700">Email</label>
            <p className="mt-1 text-gray-900">{user?.email}</p>
          </div>

          <div>
            <label className="block text-sm font-medium text-gray-700">Role</label>
            <p className="mt-1">
              <span className={`inline-flex px-2 py-1 text-xs font-medium rounded-full ${
                profile?.role === 'admin'
                  ? 'bg-red-100 text-red-800'
                  : profile?.role === 'manager'
                  ? 'bg-yellow-100 text-yellow-800'
                  : 'bg-blue-100 text-blue-800'
              }`}>
                {profile?.role}
              </span>
            </p>
          </div>
        </div>

        {profile?.privileges && profile.privileges.length > 0 && (
          <div className="mt-6">
```

```
            <label className="block text-sm font-medium text-gray-700">Privileges</label>
            <div className="mt-1 flex flex-wrap gap-2">
              {profile.privileges.map((privilege) => (
                <span
                  key={privilege}
                  className="inline-flex px-2 py-1 text-xs font-medium bg-green-100 text-
                >
                  {privilege}
                </span>
              ))}
            </div>
          </div>
        )}
      </div>

      {profile?.role === 'admin' && (
        <div className="bg-gradient-to-r from-blue-500 to-purple-600 rounded-lg p-6">
          <h2 className="text-xl font-semibold text-white mb-2">
            Admin Dashboard Access
          </h2>
          <p className="text-blue-100 mb-4">
            Manage your creative publishing platform, e-commerce, and content from the ad
          </p>
          <Link
            href="/account/admin"
            className="inline-flex items-center px-4 py-2 bg-white text-blue-600 font-med
          >
            Go to Admin Dashboard →
          </Link>
        </div>
      )}
    </div>
  )
}
```

6.2 Main Admin Dashboard

```
// src/app/account/admin/page.tsx
import { KpiCard } from '@/components/admin/KpiCard'
import { ChartsSection } from '@/components/admin/ChartsSection'

export default function AdminDashboard() {
  return (
    <div className="space-y-6">
      <div className="flex items-center justify-between">
        <h1 className="text-2xl font-bold text-gray-900">
          Creative + Commerce Dashboard
        </h1>
        <div className="text-sm text-gray-500">
          Last updated: {new Date().toLocaleString()}
        </div>
      </div>

      {/* KPI Cards */}
      <div className="grid grid-cols-1 md:grid-cols-2 lg:grid-cols-4 gap-6">
```

```
        <KpiCard
          title="Total Sales"
          value="€12,530"
          trend="+15.2%"
          color="green"
        />
        <KpiCard
          title="Active Orders"
          value="284"
          trend="+8.7%"
          color="blue"
        />
        <KpiCard
          title="Blog Views (7d)"
          value="4,210"
          trend="+6.3%"
          color="purple"
        />
        <KpiCard
          title="Beta Applications"
          value="42"
          color="red"
        />
      </div>

      {/* Creative Metrics */}
      <div className="grid grid-cols-1 md:grid-cols-2 lg:grid-cols-4 gap-6">
        <KpiCard
          title="Wiki Entries"
          value="127"
          color="blue"
        />
        <KpiCard
          title="Characters"
          value="45"
          color="purple"
        />
        <KpiCard
          title="Timeline Events"
          value="238"
          color="green"
        />
        <KpiCard
          title="Artist Projects"
          value="7"
          trend="3 active"
          color="red"
        />
      </div>

      {/* Charts */}
      <ChartsSection />

      {/* Recent Activity */}
      <div className="grid grid-cols-1 lg:grid-cols-2 gap-6">
        <div className="bg-white rounded-lg border">
```

```
          <div className="p-6 border-b">
            <h3 className="text-lg font-semibold">Recent Orders</h3>
          </div>
          <div className="p-6">
            <div className="space-y-4">
              {/* Mock recent orders */}
              <div className="flex items-center justify-between">
                <div>
                  <p className="font-medium">Persian Mythology eBook</p>
                  <p className="text-sm text-gray-500">Customer: john@example.com</p>
                </div>
                <span className="text-green-600 font-semibold">€24.99</span>
              </div>
              <div className="flex items-center justify-between">
                <div>
                  <p className="font-medium">Character Guide Bundle</p>
                  <p className="text-sm text-gray-500">Customer: sarah@example.com</p>
                </div>
                <span className="text-green-600 font-semibold">€39.99</span>
              </div>
            </div>
          </div>
        </div>

        <div className="bg-white rounded-lg border">
          <div className="p-6 border-b">
            <h3 className="text-lg font-semibold">Content Activity</h3>
          </div>
          <div className="p-6">
            <div className="space-y-4">
              <div className="flex items-center space-x-3">
                <div className="w-2 h-2 bg-blue-500 rounded-full"></div>
                <div>
                  <p className="font-medium">New blog post published</p>
                  <p className="text-sm text-gray-500">2 hours ago</p>
                </div>
              </div>
              <div className="flex items-center space-x-3">
                <div className="w-2 h-2 bg-green-500 rounded-full"></div>
                <div>
                  <p className="font-medium">Wiki entry updated: "Fereydun"</p>
                  <p className="text-sm text-gray-500">5 hours ago</p>
                </div>
              </div>
              <div className="flex items-center space-x-3">
                <div className="w-2 h-2 bg-purple-500 rounded-full"></div>
                <div>
                  <p className="font-medium">New character added</p>
                  <p className="text-sm text-gray-500">1 day ago</p>
                </div>
              </div>
            </div>
          </div>
        </div>
      </div>
    </div>
```

```
  )
}
```

6.3 Blog Management Example

```
// src/app/account/admin/content/blog/page.tsx
'use client'
import { useState, useEffect } from 'react'
import Link from 'next/link'
import { createSupabaseClient } from '@/lib/supabase'
import { Plus, Edit, Eye, Trash2 } from 'lucide-react'

interface BlogPost {
  id: string
  title: string
  slug: string
  status: string
  published_at: string | null
  created_at: string
}

export default function BlogManagement() {
  const [posts, setPosts] = useState<BlogPost[]>([])
  const [loading, setLoading] = useState(true)
  const supabase = createSupabaseClient()

  useEffect(() => {
    fetchPosts()
  }, [])

  const fetchPosts = async () => {
    try {
      const { data, error } = await supabase
        .from('blog_posts')
        .select('id, title, slug, status, published_at, created_at')
        .order('created_at', { ascending: false })

      if (error) throw error
      setPosts(data || [])
    } catch (error) {
      console.error('Error fetching posts:', error)
    } finally {
      setLoading(false)
    }
  }

  const deleteBlogPost = async (id: string) => {
    if (!confirm('Are you sure you want to delete this post?')) return

    try {
      const { error } = await supabase
        .from('blog_posts')
        .delete()
        .eq('id', id)
```

```
      if (error) throw error

      setPosts(posts.filter(post => post.id !== id))
    } catch (error) {
      console.error('Error deleting post:', error)
      alert('Failed to delete post')
    }
  }

  if (loading) {
    return <div className="flex justify-center items-center h-64">Loading...</div>
  }

  return (
    <div className="space-y-6">
      <div className="flex items-center justify-between">
        <h1 className="text-2xl font-bold text-gray-900">Blog Posts</h1>
        <Link
          href="/account/admin/content/blog/new"
          className="inline-flex items-center px-4 py-2 bg-blue-600 text-white rounded-md
        >
          <Plus className="w-4 h-4 mr-2" />
          New Post
        </Link>
      </div>

      <div className="bg-white rounded-lg border overflow-hidden">
        <table className="min-w-full divide-y divide-gray-200">
          <thead className="bg-gray-50">
            <tr>
              <th className="px-6 py-3 text-left text-xs font-medium text-gray-500 upperc
                Title
              </th>
              <th className="px-6 py-3 text-left text-xs font-medium text-gray-500 upperc
                Status
              </th>
              <th className="px-6 py-3 text-left text-xs font-medium text-gray-500 upperc
                Published
              </th>
              <th className="px-6 py-3 text-left text-xs font-medium text-gray-500 upperc
                Actions
              </th>
            </tr>
          </thead>
          <tbody className="bg-white divide-y divide-gray-200">
            {posts.map((post) => (
              <tr key={post.id} className="hover:bg-gray-50">
                <td className="px-6 py-4">
                  <div>
                    <div className="text-sm font-medium text-gray-900">
                      {post.title}
                    </div>
                    <div className="text-sm text-gray-500">/{post.slug}</div>
                  </div>
                </td>
                <td className="px-6 py-4 whitespace-nowrap">
```

```
                  <span className={`inline-flex px-2 py-1 text-xs font-semibold rounded-f
                    post.status === 'published'
                      ? 'bg-green-100 text-green-800'
                      : post.status === 'draft'
                      ? 'bg-yellow-100 text-yellow-800'
                      : 'bg-gray-100 text-gray-800'
                  }`}>
                    {post.status}
                  </span>
                </td>
                <td className="px-6 py-4 whitespace-nowrap text-sm text-gray-500">
                  {post.published_at
                    ? new Date(post.published_at).toLocaleDateString()
                    : 'Not published'
                  }
                </td>
                <td className="px-6 py-4 whitespace-nowrap text-sm font-medium">
                  <div className="flex space-x-2">
                    <Link
                      href={`/account/admin/content/blog/${post.id}/edit`}
                      className="text-blue-600 hover:text-blue-900"
                    >
                      <Edit className="w-4 h-4" />
                    </Link>
                    <Link
                      href={`/blog/${post.slug}`}
                      className="text-green-600 hover:text-green-900"
                      target="_blank"
                    >
                      <Eye className="w-4 h-4" />
                    </Link>
                    <button
                      onClick={() => deleteBlogPost(post.id)}
                      className="text-red-600 hover:text-red-900"
                    >
                      <Trash2 className="w-4 h-4" />
                    </button>
                  </div>
                </td>
              </tr>
            ))}
          </tbody>
        </table>
      </div>
    </div>
  )
}
```

6.4 Blog Post Editor

```
// src/app/account/admin/content/blog/new/page.tsx
'use client'
import { useState } from 'react'
import { useRouter } from 'next/navigation'
import { createSupabaseClient } from '@/lib/supabase'
```

```
import { AdvancedEditor } from '@/components/editor/AdvancedEditor'
import { Save, Eye } from 'lucide-react'

export default function NewBlogPost() {
  const [title, setTitle] = useState('')
  const [slug, setSlug] = useState('')
  const [excerpt, setExcerpt] = useState('')
  const [content, setContent] = useState('')
  const [status, setStatus] = useState('draft')
  const [coverUrl, setCoverUrl] = useState('')
  const [saving, setSaving] = useState(false)

  const router = useRouter()
  const supabase = createSupabaseClient()

  const generateSlug = (title: string) => {
    return title
      .toLowerCase()
      .replace(/[^a-z0-9]+/g, '-')
      .replace(/(^-|-$)+/g, '')
  }

  const handleTitleChange = (newTitle: string) => {
    setTitle(newTitle)
    if (!slug || slug === generateSlug(title)) {
      setSlug(generateSlug(newTitle))
    }
  }

  const saveBlogPost = async (publishNow = false) => {
    setSaving(true)

    try {
      const { data: { user } } = await supabase.auth.getUser()
      if (!user) throw new Error('Not authenticated')

      const postData = {
        title,
        slug,
        content: JSON.stringify(content),
        excerpt,
        cover_url: coverUrl || null,
        status: publishNow ? 'published' : status,
        published_at: publishNow ? new Date().toISOString() : null,
        author_id: user.id
      }

      const { data, error } = await supabase
        .from('blog_posts')
        .insert([postData])
        .select()
        .single()

      if (error) throw error

      router.push('/account/admin/content/blog')
```

```
    } catch (error) {
      console.error('Error saving blog post:', error)
      alert('Failed to save blog post')
    } finally {
      setSaving(false)
    }
  }

  return (
    <div className="max-w-4xl mx-auto space-y-6">
      <div className="flex items-center justify-between">
        <h1 className="text-2xl font-bold text-gray-900">New Blog Post</h1>

        <div className="flex space-x-3">
          <button
            onClick={() => saveBlogPost(false)}
            disabled={saving || !title}
            className="inline-flex items-center px-4 py-2 border border-gray-300 rounded-
          >
            <Save className="w-4 h-4 mr-2" />
            Save Draft
          </button>
          <button
            onClick={() => saveBlogPost(true)}
            disabled={saving || !title || !content}
            className="inline-flex items-center px-4 py-2 bg-blue-600 text-white rounded-
          >
            <Eye className="w-4 h-4 mr-2" />
            Publish
          </button>
        </div>
      </div>

      <div className="grid grid-cols-1 lg:grid-cols-3 gap-6">
        {/* Main Content */}
        <div className="lg:col-span-2 space-y-6">
          <div className="bg-white rounded-lg border p-6">
            <div className="space-y-4">
              <div>
                <label className="block text-sm font-medium text-gray-700 mb-2">
                  Title
                </label>
                <input
                  type="text"
                  value={title}
                  onChange={(e) => handleTitleChange(e.target.value)}
                  className="w-full border border-gray-300 rounded-md px-3 py-2 focus:rin
                  placeholder="Enter blog post title..."
                />
              </div>

              <div>
                <label className="block text-sm font-medium text-gray-700 mb-2">
                  Slug
                </label>
                <input
```

```
                type="text"
                value={slug}
                onChange={(e) => setSlug(e.target.value)}
                className="w-full border border-gray-300 rounded-md px-3 py-2 focus:rin
                placeholder="url-friendly-slug"
              />
            </div>

            <div>
              <label className="block text-sm font-medium text-gray-700 mb-2">
                Excerpt
              </label>
              <textarea
                value={excerpt}
                onChange={(e) => setExcerpt(e.target.value)}
                rows={3}
                className="w-full border border-gray-300 rounded-md px-3 py-2 focus:rin
                placeholder="Brief description of the blog post..."
              />
            </div>
          </div>
        </div>

        <div>
          <label className="block text-sm font-medium text-gray-700 mb-2">
            Content
          </label>
          <AdvancedEditor
            content={content}
            onChange={setContent}
            placeholder="Start writing your blog post..."
          />
        </div>
      </div>

      {/* Sidebar */}
      <div className="space-y-6">
        <div className="bg-white rounded-lg border p-6">
          <h3 className="text-lg font-medium text-gray-900 mb-4">Settings</h3>

          <div className="space-y-4">
            <div>
              <label className="block text-sm font-medium text-gray-700 mb-2">
                Status
              </label>
              <select
                value={status}
                onChange={(e) => setStatus(e.target.value)}
                className="w-full border border-gray-300 rounded-md px-3 py-2 focus:rin
              >
                <option value="draft">Draft</option>
                <option value="published">Published</option>
                <option value="scheduled">Scheduled</option>
              </select>
            </div>
```

```
              <div>
                <label className="block text-sm font-medium text-gray-700 mb-2">
                  Cover Image URL
                </label>
                <input
                  type="url"
                  value={coverUrl}
                  onChange={(e) => setCoverUrl(e.target.value)}
                  className="w-full border border-gray-300 rounded-md px-3 py-2 focus:rin
                  placeholder="https://example.com/image.jpg"
                />
              </div>
            </div>
          </div>
        </div>
      </div>
    )
}
```

7. Route Setup & Layout

7.1 Root Layout with Auth Provider

```
// src/app/layout.tsx
import { AuthProvider } from '@/context/AuthContext'
import './globals.css'

export const metadata = {
  title: 'Creative Publishing Platform',
  description: 'Admin dashboard for creative publishing and e-commerce',
}

export default function RootLayout({
  children,
}: {
  children: React.ReactNode
}) {
  return (
    <html lang="en">
      <body>
        <AuthProvider>
          {children}
        </AuthProvider>
      </body>
    </html>
  )
}
```

7.2 Admin Layout Wrapper

```
// src/app/account/admin/layout.tsx
import { ProtectedRoute } from '@/components/ProtectedRoute'
import { AdminLayout } from '@/components/admin/AdminLayout'

export default function AdminLayoutWrapper({
  children,
}: {
  children: React.ReactNode
}) {
  return (
    <ProtectedRoute requiredRole="admin">
      <AdminLayout>
        {children}
      </AdminLayout>
    </ProtectedRoute>
  )
}
```

8. Installation & Setup Instructions

8.1 Create Supabase Project

1. Go to supabase.io
2. Create a new project
3. Run the SQL schema provided above in the SQL editor
4. Enable Row Level Security policies
5. Get your project URL and keys from Settings > API

8.2 Deploy the Frontend

```
# Clone or create your project
mkdir creative-admin-dashboard
cd creative-admin-dashboard

# Install dependencies
npm create next-app@latest . --typescript --tailwind --app --src-dir --import-alias "@/*"

# Install additional packages (all the packages listed in step 1.2)

# Set up environment variables
cp .env.example .env.local
# Add your Supabase credentials

# Run development server
npm run dev
```

8.3 Configure Authentication Flow

1. In Supabase dashboard, go to Authentication > Settings
2. Set site URL to `http://localhost:3000` (development)
3. Add redirect URLs for production
4. Enable email confirmation if desired
5. Set up OAuth providers if needed

8.4 Create Admin User

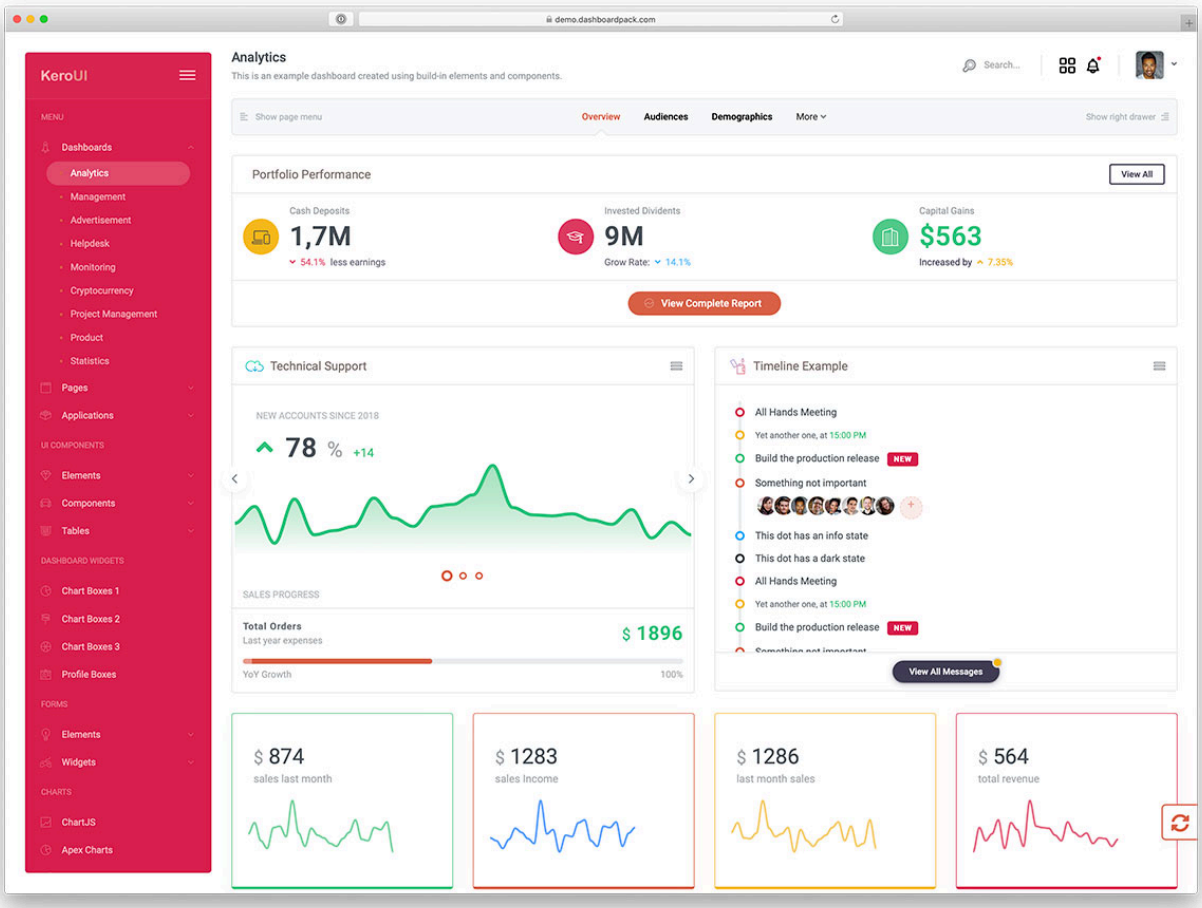
```
-- In Supabase SQL editor, after creating a user account
UPDATE profiles
SET role = 'admin', privileges = ARRAY['manage_users', 'manage_content', 'manage_commerce
WHERE email = 'your-admin-email@example.com';
```

9. Key Features Summary

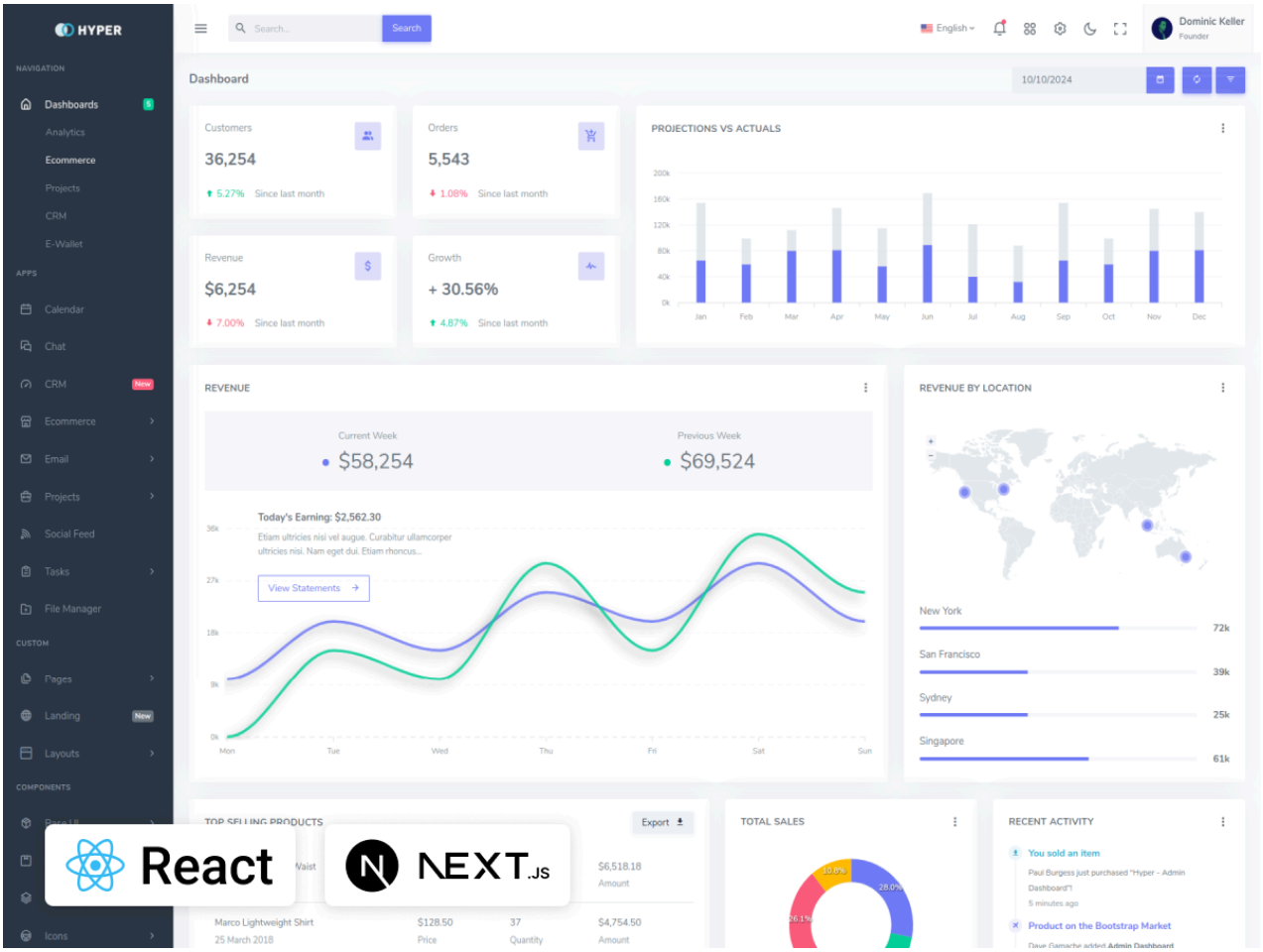
This complete implementation provides:

✅ **Unified Authentication** - Single login, role-based access via profile page
✅ **Comprehensive Dashboard** - Both creative and commerce analytics
✅ **Advanced Rich Text Editor** - Custom Tiptap editor with info boxes, footnotes, columns
✅ **Content Management** - Blog, Wiki, Characters, Timelines, Artist collaborations
✅ **E-commerce Tools** - Orders, customers, products management
✅ **Beta Reader System** - Application and manager tools
✅ **Analytics & Insights** - Real-time metrics for all platform aspects
✅ **Responsive Design** - Works on all devices with mobile-friendly sidebar
✅ **Secure & Scalable** - Built with Supabase RLS and Next.js best practices

The dashboard seamlessly blends your creative publishing tools with e-commerce functionality, exactly as discussed in your conversation with Copilot. You can now manage everything from blog posts and character development to sales analytics and beta reader applications in one unified interface.



Admin dashboard interface from KeroUI displaying portfolio metrics, technical support analytics, and a timeline of events in a clean layout.



React and Next.js based admin dashboard interface with sidebar navigation and multiple widgets for ecommerce and analytics.

⁂

1. paste.txt
2. https://nextjstemplates.com/blog/admin-dashboard-templates
3. https://themeselection.com/react-admin-panel-template/
4. https://nextadmin.co/blog/ecommerce-admin-dashboard-template
5. https://support.microsoft.com/en-us/topic/using-copilot-pages-with-microsoft-copilot-cbceeea1-d3be-4028-9ec5-ba1b55dac1a3
6. https://themefisher.com/free-responsive-html5-bootstrap-admin-templates
7. https://marmelab.com/react-admin/Demos.html
8. https://themeselection.com/ecommerce-admin-panel-template-free/
9. https://learn.microsoft.com/en-us/viva/insights/org-team-insights/copilot-dashboard
10. https://www.creative-tim.com/blog/web-design/free-dashboards-templates-laravel/
11. https://dribbble.com/tags/react-admin
12. https://muz.li/blog/dashboard-design-inspirations-in-2024/
13. https://dribbble.com/tags/ecommerce-admin-dashboard
14. https://learn.microsoft.com/en-us/copilot/overview

15. https://supabase.com/docs/guides/getting-started/tutorials/with-nextjs
16. https://www.reddit.com/r/nextjs/comments/1bxz5de/nextjs_supabase_whats_the_best_way_to_quickly_set/
17. https://www.permit.io/blog/supabase-authentication-and-authorization-in-nextjs-implementation-guide
18. https://supabase.com/docs/guides/auth/server-side/nextjs
19. https://www.zestminds.com/blog/supabase-auth-nextjs-setup-guide/
20. https://peerlist.io/vaishnavs/articles/creating-a-flexible-rich-text-editor-with-tiptap
21. https://www.figma.com/community/website-templates/sidebar
22. https://www.youtube.com/watch?v=D3HC_NyrTe8
23. https://dribbble.com/search/admin-dashboard
24. https://dev.to/abdelraman_ahmed_e83db59f/building-a-richtext-editor-with-tiptap-in-react-with-mentions-2bdp
25. https://colorlib.com/wp/bootstrap-sidebar/
26. https://dev.to/aviatorscode2/how-to-build-a-custom-medusajs-admin-dashboard-with-nextjs-supabase-and-tailwind-css-2iop
27. https://tiptap.dev/docs/editor/getting-started/install/react
28. https://dribbble.com/tags/dashboard-sidebar
29. https://github.com/w3labkr/nextjs-with-supabase-auth
30. https://reactjs-tiptap-editor.vercel.app
31. https://dribbble.com/search/sidebar-ecommerce
32. https://www.figma.com/templates/dashboard-designs/
33. https://bootstrapadmintemplate.com/the-5-best-modern-bootstrap-admin-template-for-2024/
34. https://colorlib.com/wp/free-bootstrap-admin-dashboard-templates/
35. https://marmelab.com/blog/2025/04/23/react-admin-with-shadcn.html
36. https://www.figma.com/community/file/1480498382035772022/e-commerce-dashboard-admin-ui-kits
37. https://www.youtube.com/watch?v=ZEtb9k7Lxr0