```
import React, { Suspense } from "react";
import { connect } from "react-redux";
import { useLocation } from "react-router-dom";
import SideNav from "components/layout-components/SideNav";
import TopNav from "components/layout-components/TopNav";
import Loading from "components/shared-components/Loading";
import MobileNav from "components/layout-components/MobileNav";
import HeaderNav from "components/layout-components/HeaderNav";
import PageHeader from "components/layout-components/PageHeader";
import Footer from "components/layout-components/Footer";
import { Layout, Grid } from "antd";
import navigationConfig from "configs/NavigationConfig";
import {
  SIDE_NAV_WIDTH,
  SIDE_NAV_COLLAPSED_WIDTH,
  NAV_TYPE_SIDE,
  NAV_TYPE_TOP,
  DIR_RTL,
  DIR_LTR,
} from "constants/ThemeConstant";
import utils from "utils";

import { useThemeSwitcher } from "react-css-theme-switcher";

const { Content } = Layout;
const { useBreakpoint } = Grid;

export const AppLayout = ({ navCollapsed, navType, direction, children }) => {
  const location = useLocation();

  const currentRouteInfo = utils.getRouteInfo(
    navigationConfig,
    location.pathname
  );
  const screens = utils.getBreakPoint(useBreakpoint());
  const isMobile = screens.length === 0 ? false : !screens.includes("lg");
  const isNavSide = navType === NAV_TYPE_SIDE;
  const isNavTop = navType === NAV_TYPE_TOP;
  const getLayoutGutter = () => {
    if (isNavTop || isMobile) {
      return 0;
    }
    return navCollapsed ? SIDE_NAV_COLLAPSED_WIDTH : SIDE_NAV_WIDTH;
  };

  const { status } = useThemeSwitcher();

  if (status === "loading") {
    return <Loading cover="page" />;
  }

  const getLayoutDirectionGutter = () => {
    if (direction === DIR_LTR) {
      return { paddingLeft: getLayoutGutter() };
    }
    if (direction === DIR_RTL) {
      return { paddingRight: getLayoutGutter() };
    }
    return { paddingLeft: getLayoutGutter() };
  };

  return (
    <Layout>
      <HeaderNav isMobile={isMobile} />
      {isNavTop && !isMobile ? <TopNav routeInfo={currentRouteInfo} /> : null}
      <Layout className="app-container">
        {isNavSide && !isMobile ? (
          <SideNav routeInfo={currentRouteInfo} />
        ) : null}
        <Layout className="app-layout" style={getLayoutDirectionGutter()}>
          <div className={`app-content ${isNavTop ? "layout-top-nav" : ""}`}>
```

```
            <PageHeader
              display={currentRouteInfo?.breadcrumb}
              title={currentRouteInfo?.title}
            />
            <Content>
              <Suspense fallback={<Loading cover="content" />}>
                {children}
              </Suspense>
            </Content>
          </div>
          <Footer />
        </Layout>
      </Layout>
      {isMobile && <MobileNav />}
    </Layout>
  );
};

const mapStateToProps = ({ theme }) => {
  const { navCollapsed, navType, locale } = theme;
  return { navCollapsed, navType, locale };
};

export default connect(mapStateToProps)(React.memo(AppLayout));
```