```
import React, { useMemo } from "react";
import { Link } from "react-router-dom";
import { Menu, Grid, Badge, Tooltip } from "antd";
import IntlMessage from "../util-components/IntlMessage";
import Icon from "../util-components/Icon";
import navigationConfig from "configs/NavigationConfig";
import { useSelector, useDispatch } from "react-redux";
import { SIDE_NAV_LIGHT, NAV_TYPE_SIDE } from "constants/ThemeConstant";
import utils from "utils";
import { onMobileNavToggle } from "store/slices/themeSlice";
import { PERMIT } from "constants/AuthConstant";
import Utils from "utils";
import { useLocation } from "react-router-dom";
import { SYSTEM_SCOPE } from "configs/AppConfig";

const { useBreakpoint } = Grid;

const setLocale = (localeKey, isLocaleOn = true) =>
  isLocaleOn ? <IntlMessage id={localeKey} /> : localeKey.toString();

const setDefaultOpen = (key) => {
  let keyList = [];
  let keyString = "";
  if (key) {
    const arr = key.split("-");
    for (let index = 0; index < arr.length; index++) {
      const elm = arr[index];
      index === 0 ? (keyString = elm) : (keyString = `${keyString}-${elm}`);
      keyList.push(keyString);
    }
  }
  return keyList;
};

const MenuItem = ({ title, icon, path, bagetnumber = 0 }) => {
  const dispatch = useDispatch();
  const isMobile = !utils.getBreakPoint(useBreakpoint()).includes("lg");

  const closeMobileNav = () => {
    if (isMobile) {
      dispatch(onMobileNavToggle(false));
    }
  };

  return (
    <>
      {icon && <Icon type={icon} />}
      <span>{setLocale(title)}</span>
      {bagetnumber > 0 && (
        <span
          style={{
            marginLeft: 3,
          }}
        >
          <Tooltip placement="topLeft" title="Số lượt đặt lịch chờ xác nhận">
            <Badge
              style={{ marginLeft: "10px" }}
              count={bagetnumber >= 10 ? "+9" : bagetnumber}
            />
          </Tooltip>
        </span>
      )}
      {path && <Link onClick={closeMobileNav} to={path} />}
    </>
  );
};

const getSideNavMenuItem = (navItem, data, scopeId) => {
  const newNav = [];
  // eslint-disable-next-line array-callback-return
  navItem
```

```
    .filter((item) => !scopeId || !item.scopeId || scopeId === item.scopeId)
    .map((item) => {
      if (item.isPermit) {
        item.badgeNumber = Utils.CheckNumerByPermit(data, item.permit);

        const permits = JSON.parse(localStorage.getItem(PERMIT));
        // kiem tra neu có submenu,thi check hien thi theo submenu
        if (Array.isArray(item.submenu) && item.submenu.length > 0) {
          newNav.push({ ...item, viewBySubMenu: true });
        } else if (Utils.checkPermitValue(item.action, permits, item.permit))
          newNav.push(item);
        else return newNav;
      } else return newNav.push(item);
    });
  const finalMenu = newNav.map((nav) => {
    return {
      key: nav.key,
      label: (
        <MenuItem
          title={nav.title}
          bagetnumber={nav.badgeNumber}
          {...(nav.isGroupTitle ? {} : { path: nav.path, icon: nav.icon })}
        />
      ),
      ...(nav.isGroupTitle ? { type: "group" } : {}),
      ...(nav.submenu.length > 0
        ? { children: getSideNavMenuItem(nav.submenu, data) }
        : {}),
      viewBySubMenu: nav?.viewBySubMenu,
    };
  });
  return finalMenu.filter(
    (item) =>
      !item.viewBySubMenu ||
      (item.viewBySubMenu &&
        Array.isArray(item.children) &&
        item.children.length > 0)
  );
};

const getTopNavMenuItem = (navItem) =>
  navItem.map((nav) => {
    return {
      key: nav.key,
      label: (
        <MenuItem
          title={nav.title}
          icon={nav.icon}
          {...(nav.isGroupTitle ? {} : { path: nav.path })}
        />
      ),
      ...(nav.submenu.length > 0
        ? { children: getTopNavMenuItem(nav.submenu) }
        : {}),
    };
  });

const SideNavContent = (props) => {
  const { routeInfo, hideGroupTitle, scopeId } = props;
  const { bagetList } = useSelector((state) => state.category);

  const sideNavTheme = useSelector((state) => state.theme.sideNavTheme);

  // eslint-disable-next-line react-hooks/exhaustive-deps
  const menuItems = useMemo(() =>
    getSideNavMenuItem(navigationConfig, bagetList, scopeId)
  );

  return (
    <Menu
      mode="inline"
      theme={sideNavTheme === SIDE_NAV_LIGHT ? "light" : "dark"}
      style={{ height: "100%", borderRight: 0 }}
      defaultSelectedKeys={[routeInfo?.key]}
```

```
      defaultOpenKeys={setDefaultOpen(routeInfo?.key)}
      className={hideGroupTitle ? "hide-group-title" : ""}
      items={menuItems}
    />
  );
};

const TopNavContent = () => {
  const topNavColor = useSelector((state) => state.theme.topNavColor);

  const menuItems = useMemo(() => getTopNavMenuItem(navigationConfig), []);

  return (
    <Menu
      mode="horizontal"
      style={{ backgroundColor: topNavColor }}
      items={menuItems}
    />
  );
};

const MenuContent = (props) => {
  const location = useLocation();
  const scopeId = SYSTEM_SCOPE.find((item) =>
    location.pathname.includes(item.path)
  )?.value;
  return props.type === NAV_TYPE_SIDE ? (
    <>
      <SideNavContent scopeId={scopeId} {...props} />
    </>
  ) : (
    <>
      <TopNavContent scopeId={scopeId} {...props} />
    </>
  );
};

export default MenuContent;
```