```
import {
  Button,
  Col,
  Form,
  Input,
  Row,
  Select,
  Spin,
  notification,
} from "antd";
import { isEmpty } from "lodash";
import React, { useEffect, useRef, useState } from "react";
import { useDispatch, useSelector } from "react-redux";

import SlipModalComponent from "components/SlipModalComponent";
import TextArea from "antd/lib/input/TextArea";
import {
  GetDoanKhamSucKhoe,
  UpsertDoanKhamSucKhoe,
} from "store/slices/khamSucKhoeDoanSlice";
import ModalEmployeeGetAll from "views/app-views/hrm/employee/components/ModalEmployeeGetAll";

const formater = new Intl.NumberFormat("vi-VN");

// formatter and parser input number
export const formatterNumber = (val) => {
  if (!val) return 0;
  return formater.format(val);
};

export const parserNumber = (val) => {
  if (!val) return 0;
  return Number.parseFloat(
    // eslint-disable-next-line no-useless-escape
    val.replace(/\$\s?|(\.*)/g, "").replace(/(\,{1})/g, ".")
  ).toFixed(2);
};
export default function ModalDoanKham({
  visibleModal,
  setVisibleModal,
  loading,
  reloadData,
  doanKhamId,
  setDoanKhamId,
  permitList = [],
}) {
  const dispatch = useDispatch();
  const [form] = Form.useForm();

  const { DoanKhamSucKhoeDetail, CongTyDtoList } = useSelector(
    (state) => state.ksk
  );
  const { branchId } = useSelector((state) => state.auth);
  const { employeeList } = useSelector((state) => state.category);
  const [visibleNvKinhDoanhId, setVisibleNvKinhDoanhId] = useState(false);
  const [selectedRowKeys, setSelectedRowKeys] = useState([]);
  const childRef = useRef(null);

  useEffect(() => {
    if (isEmpty(doanKhamId)) return;
    dispatch(GetDoanKhamSucKhoe(doanKhamId));
  }, [dispatch, doanKhamId]);

  useEffect(() => {
    if (isEmpty(doanKhamId)) return;
    form.setFieldsValue(DoanKhamSucKhoeDetail);
  }, [dispatch, doanKhamId, DoanKhamSucKhoeDetail, form]);

  const onFinish = (values) => {
    const payload = {
      id: values.id,
```

```
    tenDoan: values.tenDoan,
    nguoiLienHe: values.nguoiLienHe,
    diaChi: values.diaChi,
    dienThoai: values.dienThoai,
    ghiChu: values.ghiChu,
    vietTat: values.vietTat,
    chiNhanhId: branchId,
    congTyId: values.congTyId,
    nvKinhDoanhId: values.nvKinhDoanhId,
    trangThai: isEmpty(doanKhamId) ? 0 : DoanKhamSucKhoeDetail?.trangThai,

    onSuccess: ({ data }) => {
      // Refresh data after add/update table
      if (isEmpty(doanKhamId)) {
        setDoanKhamId(data.data.Id);
      }
      if (reloadData) reloadData();

      notification.info({
        message: "Lưu đoàn khám sức khỏe",
        description: "Lưu thành công thông tin đoàn khám sức khỏe",
      });
    },
  };

  payload.id = doanKhamId;
  dispatch(UpsertDoanKhamSucKhoe(payload));
};

return (
  <SlipModalComponent
    ref={childRef}
    maskClosable={false}
    bodyStyle={{ overflowY: "auto", maxHeight: "calc(100vh - 300px)" }}
    width={1400}
    title={
      !isEmpty(doanKhamId)
        ? "Sửa đoàn khám sức khỏe"
        : "Thêm đoàn khám sức khỏe"
    }
    objectId={doanKhamId}
    objectType={"DoanKhamSucKhoe"}
    //documentTypeId={DOCUMENT_TYPE_ID.DanhGiaUngVien}
    widthInfo={550}
    //titleTLink="Đoàn khám sức khỏe"
    //descriptionTLink={DoanKhamSucKhoeDetail?.maUngVien}
    isVisibleModal={visibleModal}
    onCancelModal={() => {
      setVisibleModal(!visibleModal);
      form.resetFields();
      if (!isEmpty(doanKhamId)) setDoanKhamId(null);
    }}
    onRenderInfo={() => {
      return (
        <Spin tip="Đang tải..." spinning={loading}>
          <Form
            form={form}
            layout="vertical"
            onFinish={onFinish}
            className="p-3"
          >
            <Row gutter={24}>
              <Col sm={16}>
                <Form.Item
                  name="tenDoan"
                  label="Tên đoàn"
                  rules={[
                    {
                      required: true,
                      message: "Vui lòng nhập tên đoàn",
                    },
                  ]}
                >
                  <Input />
```

```
    </Form.Item>
  </Col>
  <Col sm={8}>
    <Form.Item
      name="vietTat"
      label="Tên viết tắt"
      rules={[
        {
          required: true,
          message: "Vui lòng nhập viết tắt",
        },
      ]}
    >
      <Input />
    </Form.Item>
  </Col>
</Row>

<Row gutter={24}>
  <Col sm={24}>
    <Form.Item
      name="congTyId"
      label="Công ty"
      rules={[
        {
          required: true,
          message: "Vui lòng nhập công ty",
        },
      ]}
    >
      <Select
        showSearch
        filterOption={(input, option) =>
          (option?.label ?? "")
            .toLowerCase()
            .includes(input.toLowerCase())
        }
        options={CongTyDtoList?.map((i) => ({
          label: i.tenCongTy,
          value: i.id,
        }))}
        onChange={(e) => {
          form.setFieldValue(
            "dienThoai",
            CongTyDtoList?.filter((f) => f.id === e)[0]?.dienThoai
          );
          form.setFieldValue(
            "nguoiLienHe",
            CongTyDtoList?.filter((f) => f.id === e)[0]
              ?.nguoiLienHe
          );
          form.setFieldValue(
            "diaChi",
            CongTyDtoList?.filter((f) => f.id === e)[0]?.diaChi
          );
        }}
      />
    </Form.Item>
  </Col>
</Row>

<Row gutter={24}>
  <Col sm={24}>
    <Form.Item name="diaChi" label="Địa chỉ">
      <Input />
    </Form.Item>
  </Col>
</Row>

<Row gutter={24}>
  <Col sm={24}>
    <Form.Item
      name="nvKinhDoanhId"
      label="Nhân viên kinh doanh"
```

```
          rules={[
            {
              required: true,
              message: "Vui lòng nhập nhân viên kinh doanh",
            },
          ]}
        >
          <Select
            showSearch
            filterOption={(input, option) =>
              (option?.label ?? "")
                .toLowerCase()
                .includes(input.toLowerCase())
            }
            options={employeeList}
            onClick={() => {
              setVisibleNvKinhDoanhId((prev) => !prev);
            }}
            open={false}
          />
        </Form.Item>
      </Col>
    </Row>

    <Row gutter={24}>
      <Col sm={12} md={12} lg={12}>
        <Form.Item
          name="nguoiLienHe"
          label="Người liên hệ"
          rules={[
            {
              required: true,
              message: "Vui lòng nhập người liên hệ",
            },
          ]}
        >
          <Input />
        </Form.Item>
      </Col>
      <Col sm={12} md={12} lg={12}>
        <Form.Item name="dienThoai" label="Điện thoại liên hệ">
          <Input />
        </Form.Item>
      </Col>
    </Row>

    <Row gutter={24}>
      <Col sm={24}>
        <Form.Item
          name="ghiChu"
          label="Ghi chú"
          style={{ width: "100%" }}
        >
          <TextArea rows={4} />
        </Form.Item>
      </Col>
    </Row>

    <Row>
      <Col sm={24}>
        <Button
          type="primary"
          onClick={() => {
            form.submit();
          }}
        >
          Lưu
        </Button>
      </Col>
    </Row>
  </Form>

  <ModalEmployeeGetAll
    visibleModal={visibleNvKinhDoanhId}
```

```
            selectedRowKeys={selectedRowKeys}
            setSelectedRowKeys={setSelectedRowKeys}
            onCancel={() => {
              setVisibleNvKinhDoanhId((prev) => !prev);
              form.setFieldValue("nvKinhDoanhId", selectedRowKeys);
            }}
            onOk={() => {
              setVisibleNvKinhDoanhId((prev) => !prev);
              form.setFieldValue("nvKinhDoanhId", selectedRowKeys);
            }}
            type={"radio"}
          />
        </Spin>
      );
    }}
  ></SlipModalComponent>
);
}
```