```
import { Checkbox, Table } from "antd";
import React, { forwardRef, memo, useEffect } from "react";

const MediTable = forwardRef(function MediTable(
  {
    dataSource,
    tableColumns,
    onRow,
    onRenderFooter,
    enableSelectAll,
    rowKeyField = "id",
    // Callback area
    onRowSelectChange,
    bordered = false,
    components,
    getCheckboxProps,
    rowSelectionType = "checkbox",
    rowKeyValue = [],
  },
  ref
) {
  const [selectedRowKeys, setSelectedRowKeys] = React.useState([]);
  const [expandedRowKeys, setExpandedRowKeys] = React.useState([]);
  useEffect(() => {
    if (!rowKeyValue) return;
    setSelectedRowKeys(rowKeyValue);
  }, [rowKeyValue]);

  // useEffect(() => {
  //   if (onRowSelectChange) onRowSelectChange(selectedRowKeys);
  // }, [onRowSelectChange, selectedRowKeys]);

  const handleSelect = (record, selected) => {
    let rowKeys = [];
    if (selected) {
      rowKeys = [...rowKeyValue, record[rowKeyField]];
      setSelectedRowKeys((keys) => [...keys, record[rowKeyField]]);
    } else {
      const index = rowKeyValue.indexOf(record[rowKeyField]);
      rowKeys = [
        ...rowKeyValue.slice(0, index),
        ...rowKeyValue.slice(index + 1),
      ];
      setSelectedRowKeys((keys) => [
        ...keys.slice(0, index),
        ...keys.slice(index + 1),
      ]);
    }
    if (onRowSelectChange) onRowSelectChange(rowKeys);
  };

  const toggleSelectAll = () => {
    const selectedKeys =
      selectedRowKeys.length === dataSource.length
        ? []
        : dataSource.map((r) => r[rowKeyField]);
    setSelectedRowKeys([...selectedKeys]);
    if (onRowSelectChange) onRowSelectChange([...selectedKeys]);
  };

  const headerCheckbox = (
    <Checkbox
      checked={selectedRowKeys.length}
      indeterminate={
        selectedRowKeys.length > 0 && selectedRowKeys.length < dataSource.length
      }
      onChange={toggleSelectAll}
    />
  );

  const rowSelection = {
```

```
    selectedRowKeys,
    type: rowSelectionType,
    fixed: true,
    onSelect: rowSelectionType === "checkbox" && handleSelect,
    columnTitle: rowSelectionType === "checkbox" && headerCheckbox,
    onChange: (keys) => {
      if (rowSelectionType === "radio") {
        setSelectedRowKeys(keys);
        if (onRowSelectChange) onRowSelectChange(keys);
      }
    },
    getCheckboxProps: getCheckboxProps,
  };
  const handleExpand = (expanded, record) => {
    if (expanded) {
      setExpandedRowKeys((prev) => [...prev, record[rowKeyField]]);
    } else {
      setExpandedRowKeys((prev) =>
        prev.filter((key) => key !== record[rowKeyField])
      );
    }
  };
  return (
    <Table
      expandable={{
        expandedRowKeys,
        onExpand: handleExpand,
      }}
      components={components}
      rowSelection={enableSelectAll ? rowSelection : null}
      columns={tableColumns}
      dataSource={dataSource}
      scroll={{ x: "max-content" }}
      rowKey={(item) => item[rowKeyField]}
      footer={onRenderFooter}
      bordered={bordered}
      onRow={onRow}
      pagination={{
        showSizeChanger: true,
        showTotal: (total, range) => `Hiển thị: ${range[1]} / ${total}`,
      }}
    />
  );
});

export default memo(MediTable);
```