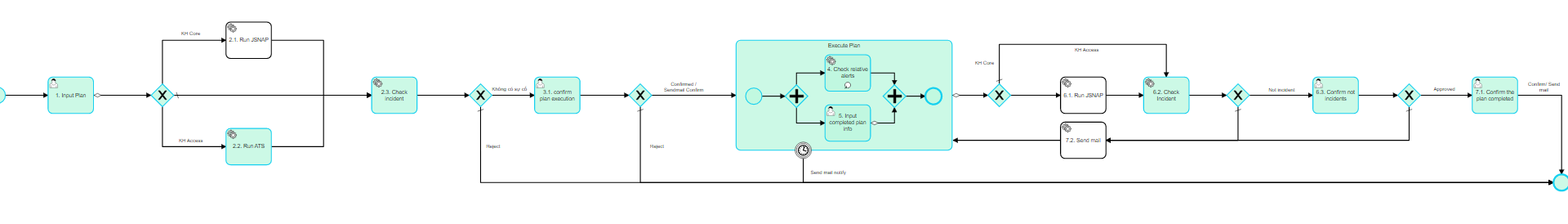
**BÁO CÁO 13: TÌM HIỂU VỀ HỆ THỐNG INCIDENT MANAGEMENT**

1. **Giới thiệu**
2. **Giới thiệu nghiệp vụ**

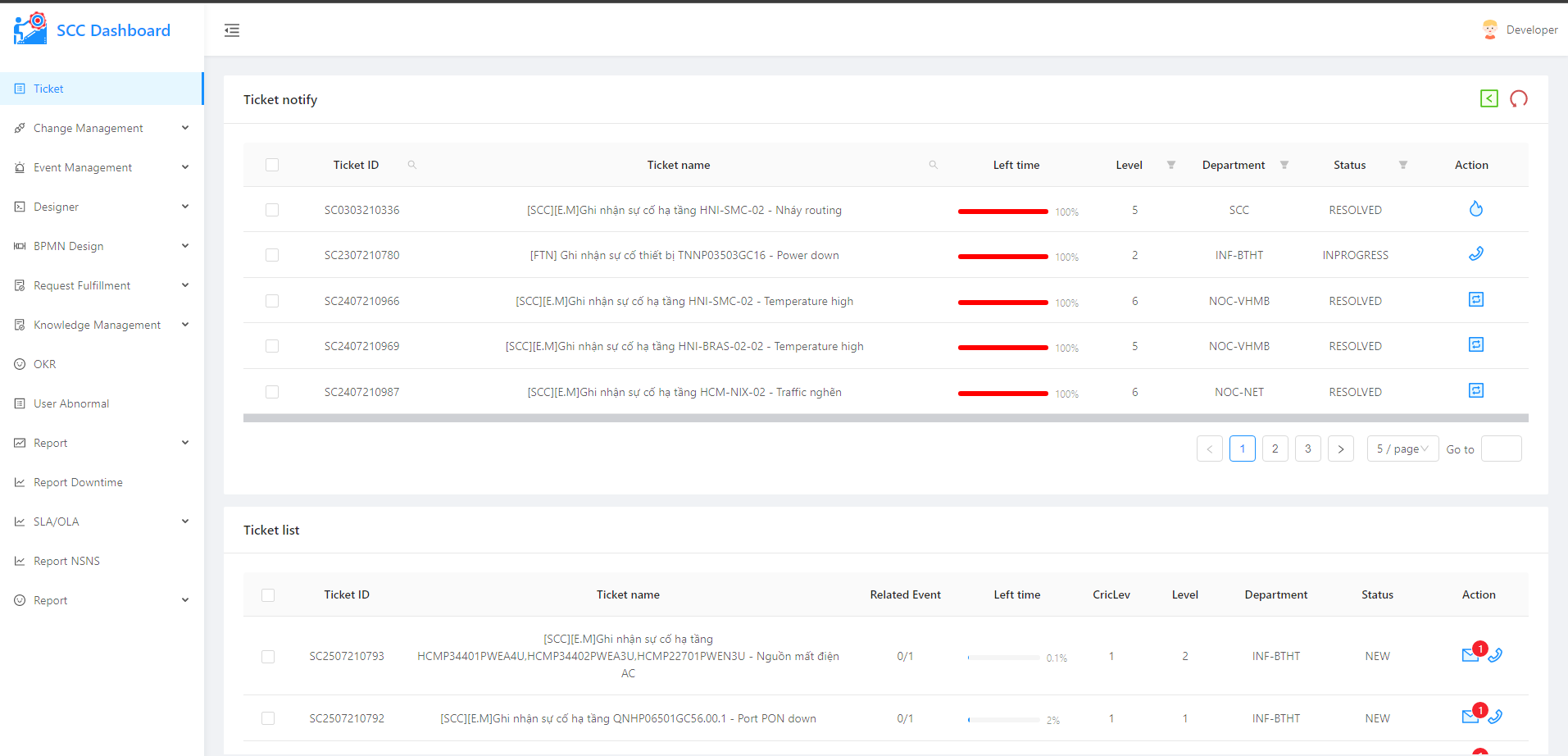
* Hằng đêm (từ 23h đến 5h sáng) đội ngũ sẽ kiểm tra các thiết bị hạ tầng và thực hiện các kế hoạch như nâng cấp, bảo trì, …
* Diagram này sẽ giúp đội ngũ có thể thao tác dễ dàng hơn trong việc kiểm tra, quản lý và thực hiện các kế hoạch.



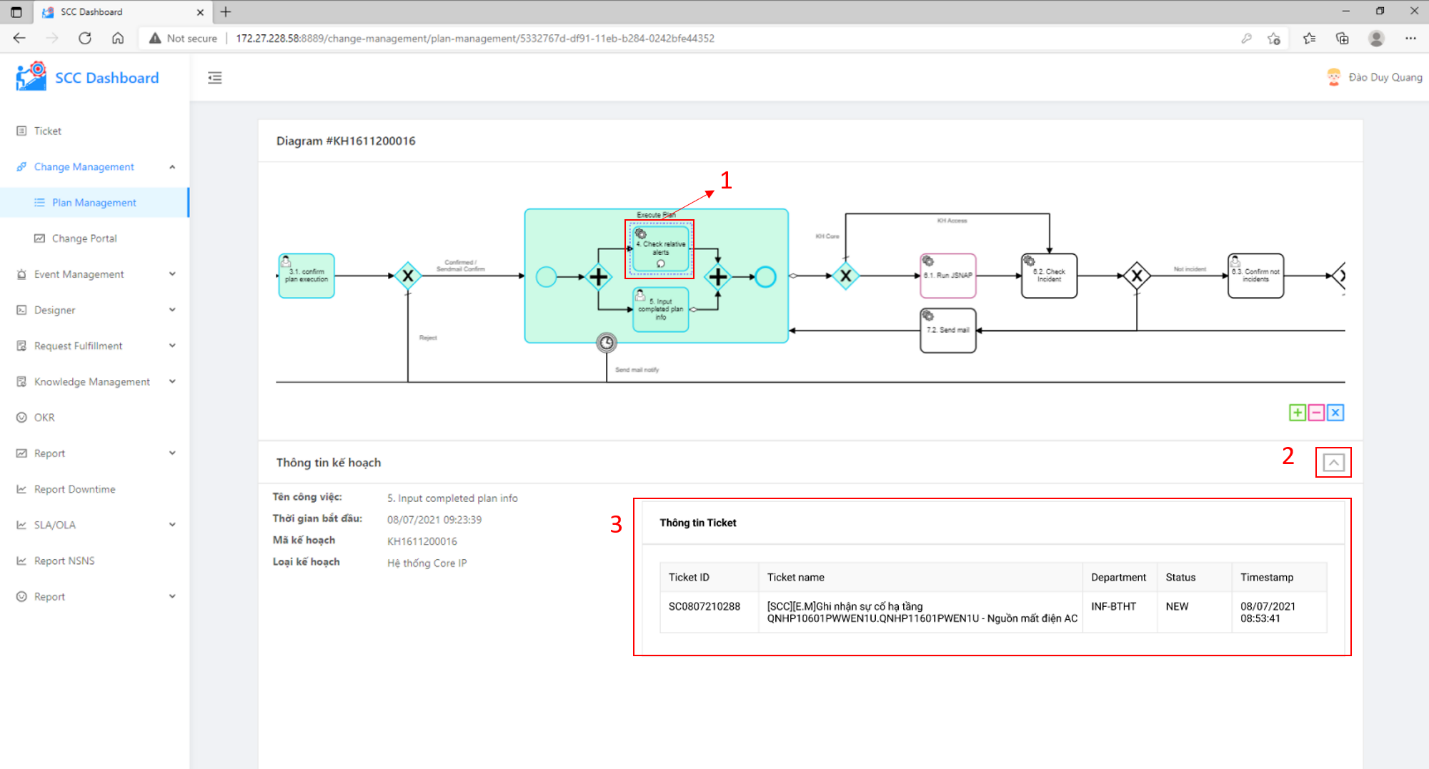
* Bước *1. Input plan*: Đây là bước nhập kế hoạch
* Sau khi đã nhập kế hoạch xong qua bước *2.1. Run JSNAP*: đây là bước chạy các ip của các thiết bị trong hạ tầng, sau đó qua bước kiểm tra incident
* Bước *2.2. Check incident:* đây là bước kiểm tra có sự cố phát sinh trong thiết bị hay không nếu có thì reject còn nếu không qua bước *3.1. Confirm plan* execution để xác nhận kế hoạch
* *3.1. Confirm plan execution:* đây là bước xác nhận kế hoạch, sau khi đã xác nhận, thì chuyển qua bước thực hiện kế hoạch, nếu không xác nhận thì reject
* Ở bước thực hiện kế hoạch, sẽ có 2 nhiệm vụ:
  + Bước *4. Check relative alerts*: kiểm tra các cảnh báo liên quan đến sự cố
  + Bước *5. Input completed plan info*: Nhập đầy đủ thông tin kế hoạch
* Trong bước thực hiện kế hoạch sẽ gửi mail thông báo
* Sau khi thực hiện kế hoạch, phải thực hiện lại bước *6.1 Run JSNAP* và kiểm tra incident 1 lần nữa tại bước *6.2. Check incident.*
* Nếu có lỗi thì sẽ gửi mail tại bước *7.2. Send mail* và quay lại bước thực hiện kế hoạch. Nếu như không có lỗi thì qua bước xác nhận bước *6.3. Confirm not incidents,* sau đó sẽ được phê duyệt là đã hoàn thành xong kế hoạch và gửi mail thông báo.

1. **Cài đặt cơ bản**

* Em sẽ clone theo repo sau: [Files · sangtt9/dev · SangTT9 / ant\_template · GitLab](http://172.27.228.161/sangtt9/ant_template/-/tree/sangtt9/dev) và đăng nhập. Như vậy em đã có thể sử dụng được hệ thống

****

1. **Cài đặt hiển thị thông tin sự cố**
2. **Vấn đề**

* Hiện tại, ở bước *4. Check relative alerts* không hiển thị ra thông tin công việc chính vì thế, em sẽ thêm thông tin vào bước này như sau:
* Chú thích:

• Số 1: Thao tác click chọn để xem được thông tin Ticket

• Số 2: Icon thả xuống để mở rộng danh sách thông tin Ticket

• Số 3: Danh sách thông tin Ticket sự cố liên quan đến kế hoạch

1. **Xử lý vấn đề**

* Đầu tiên em sẽ viết hàm gọi api để lấy thông tin, trong file ***services.js***:

export async function getTicketById(data) {

  return request(`${URL\_API\_SCCD\_PRODUCTION}/get-ticket-by-ip`, {

    method: 'POST',

    params: {

      api\_key: API\_KEY\_GET\_TICKET,

    },

    headers: {

      'Content-Type': 'application/json',

    },

    data: {

      ...data,

    },

  });

}

* Cấu trúc gọi api có body như sau :

{

    "deviceIp": "HNIB22802ME35,BGGM001-IPMS,BDHP03802GC57",

    "startDate": "2020-12-28 00:03:00",

    "endDate": "2020-12-31 00:50:00"

}

* Trong đó :
  + deviceIp : tên các thiết bị
  + startDate : ngày bắt đầu
  + endDate : ngày kết thúc
* Để lấy được tên các thiết bị, em sẽ phải truy cập đến state detailPlan/processInfo/listDevice trong Redux.
* Với ngày bắt đầu em sẽ lấy *endTime* trong bước *2.3. check incident* đã được lưu trong Redux.
* Với ngày kết thúc em sẽ lấy thời gian hiện tại.
* Như vậy logic để xử lý cấu trúc trên như sau, trong file ***model.js***:

devices = yield select(state => state.detailPlan.processInfo.listDevice);

listDeviceIp = yield devices

.map(function f(device) {

   return device.deviceName;

  })

  .join(',');

listActivityInstance = yield select(state => state.diagram.historyActivityInstance);

const cmTask23 = yield listActivityInstance.find(

item => item.activityId === 'cm\_task\_2\_3',

);

data = {

deviceIp: listDeviceIp,

  startDate: moment(cmTask23.endTime).format('YYYY-MM-DD HH:mm:ss'),

  endDate: moment().format('YYYY-MM-DD HH:mm:ss'),

};

* Sau đó, em sẽ gọi API vừa tạo mỗi khi nhấn vào bước 4, trong file ***model.js*** :

taskInfo = yield call(getTicketById, data);

const taskInfoCmTask4 = listActivityInstance.find(

item => item.activityId === 'cm\_task\_4',

);

const fakeTaskInfo = [

[

{

    activityId: taskInfoCmTask4.activityId,

     activityName: taskInfoCmTask4.activityName,

     activityType: taskInfoCmTask4.activityType,

     id: taskInfoCmTask4.id,

     startTime: taskInfoCmTask4.startTime,

   },

 ],

 { result: { info: { listTicket: taskInfo.data } } },

];

yield put({

type: 'save',

  payload: {

   taskInfo: fakeTaskInfo,

   taskId: payload.taskId,

  },

});

* Cuối cùng, em sẽ sửa lại các component sao cho khớp với data trả về khi gọi api.
* Trong file ***infoPlanCommon.jsx***em sẽ thêm vào trường hợp sau:

case 'cm\_task\_4':

return <ListTicket data={task[1]?.result} />;

* Và trong file ***ListTicket.jsx*** em sẽ sửa lại các column trong table sao cho khớp với data.

1. **Cài đặt hiển thị thông tin khi check incident**
2. **Vấn đề**

* Hiện tại khi nhấn vào bước *6.2. Check incident* vẫn chưa có data cho Thông tin ticket. Vì vậy em sẽ lấy dữ liệu từ bước *4. Check relative alerts* để hiển thị thông tin ticket

1. **Xử lý vấn đề**

* Tương tự như ở mục **II.**em sẽ gọi lại API getTicketById tuy nhiên trong body, startDate sẽ là endTime của cm\_start\_event.
* Như vậy trong file ***model.js***, em sẽ sử dụng switch case để với trường hợp người dùng nhấn vào bước 4 hay 6.2 hay các bước khác sẽ hiển thị tương ứng dữ liệu:

case 'cm\_task\_6\_2': {

devices = yield select(state => state.detailPlan.processInfo.listDevice);

listDeviceIp = devices

.map(function f(device) {

return device.deviceName;

  })

  .join(',');

listActivityInstance = yield select(state => state.diagram.historyActivityInstance);

const cmStartEvent = listActivityInstance.find(

item => item.activityId === 'cm\_start\_event',

);

data = {

deviceIp: listDeviceIp,

  startDate: moment(cmStartEvent.endTime).format('YYYY-MM-DD HH:mm:ss'),

  endDate: moment().format('YYYY-MM-DD HH:mm:ss'),

};

const serviceTask = yield call(requestServiceTask, body);

const ticket = yield call(getTicketById, data);

taskInfo = [serviceTask[0], { result: { info: { listTicket: ticket.data } } }];

yield put({

type: 'save',

  payload: {

   taskInfo,

    taskId: payload.taskId,

  },

});

break;

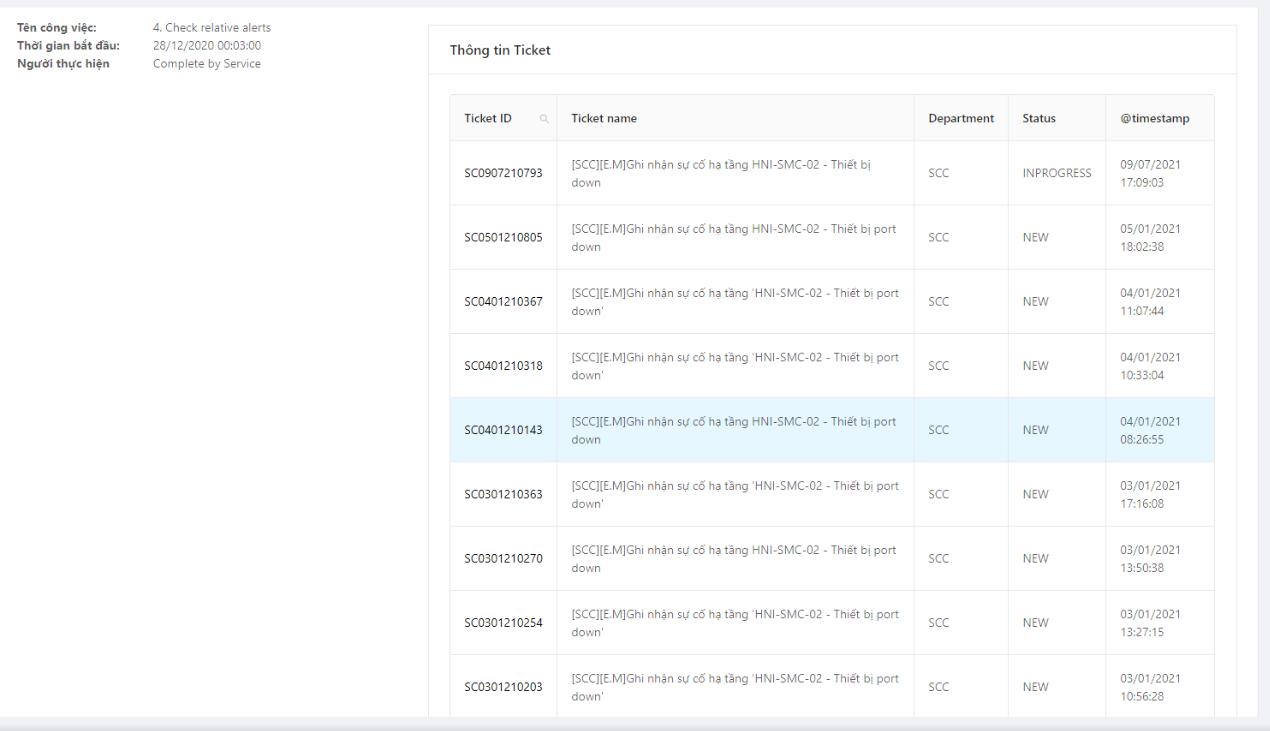
}

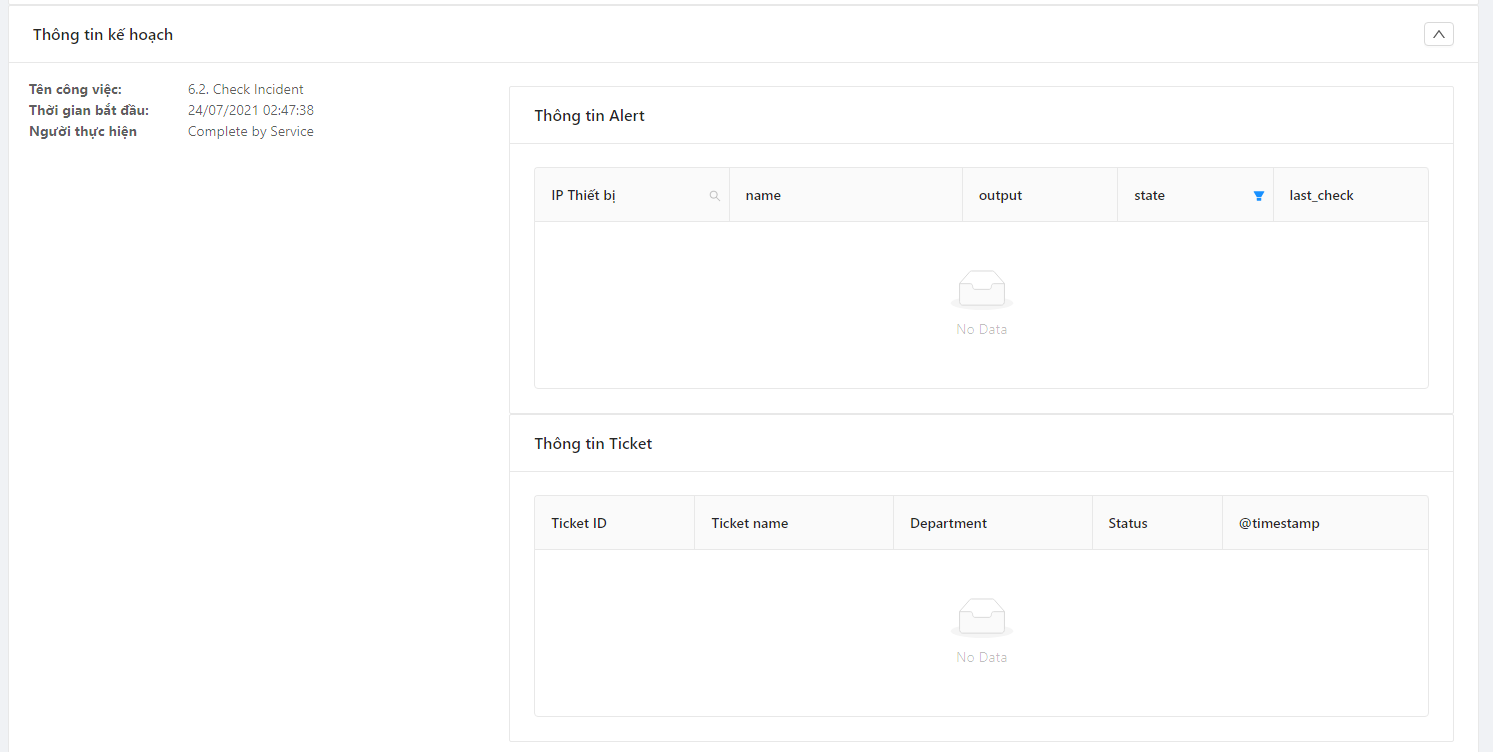
1. **Source**

* Tất cả source code nằm trong link github sau:

[Files · uy/dev · SangTT9 / ant\_template · GitLab](http://172.27.228.161/sangtt9/ant_template/-/tree/uy/dev)

1. **Demo**

****

****