**RESTlet 2.0**

**Q1: Difference between Consumer Key / Secret and token Key / Secret ?**

***Consumer Key***

* Consumer key identifies the integrated application.
* It authenticate the application.
* Its used for application level access.
* It generated during creating an integration.

***Consumer Secret***

* Consumer secret identifies a specific user in NS.
* It authenticate the user in netsuite.
* Its used for user level access.
* It generate via access token creation for an user.

**Q2: What is the difference between OAuth 1.0 and 2.0?**

***OAuth 1.0***

* OAuth 1.0 uses cryptographic signature to secure a request.
* More complex to implement.
* Security is not that strong as compare to OAuth 2.0
* It does not have token expiration.
* It is used for older APIs.

***OAuth 2.0***

* OAuth 2.0 uses access token for authorization.
* It is designed to be easier to implement and use.
* It provides more robust security features.
* Here the token expires after certain seconds and we have to refresh it again.
* It is used for modern APIs.

**Q3: Fetch a Purchase Order Record in NetSuite through postman and using OAuth2.0.**

I created a variable in params of and assigned the internal id of purchase order to it. I used **GET** request to do perform this action.

// GET method to retrieve an existing record

const get = (requestParams) => {

try {

var so\_id = requestParams.so\_id\_postman;

var so\_record = record.load({

type: record.Type.SALES\_ORDER,

id: so\_id,

isDynamic: true,

});

log.debug('Record retrieved: ', so\_record);

return so\_record;

} catch (e) {

return {

success: false,

message: e.message

};

}

};

**Q4: Create a Purchase Order Record in NetSuite through postman (don't hardcode data in script) using OAuth2.0 method.**

I created a purchase order using the POST method by giving entity as default value and mandatory fields as item, taxcode and location.

// POST method to create a new record

const post = (requestBody) => {

var new\_po\_record = record.create({

type: record.Type.PURCHASE\_ORDER,

isDynamic: true,

defaultValues: {

entity: requestBody.entity

}

})

var po\_location = new\_po\_record.setValue({

fieldId: 'location',

value: requestBody.location

})

log.debug('location is ', po\_location)

requestBody.items.forEach(function (item) {

new\_po\_record.selectNewLine({

sublistId: 'item'

})

log.debug('new line selected')

new\_po\_record.setCurrentSublistValue({

sublistId: 'item',

fieldId: 'item',

value: item.itemid

})

new\_po\_record.setCurrentSublistValue({

sublistId: 'item',

fieldId: 'taxcode',

value: item.taxcode

})

log.debug('item is ', item.itemid)

log.debug('set sublist value')

new\_po\_record.commitLine({

sublistId: 'item'

})

log.debug('line commited')

})

var purchase\_order\_id = new\_po\_record.save()

log.debug('PO created with id ', purchase\_order\_id)

try {

log.debug('PO created with id ', purchase\_order\_id)

return {

success: true,

purchase\_order\_id: purchase\_order\_id

}

} catch (e) {

return {

success: false,

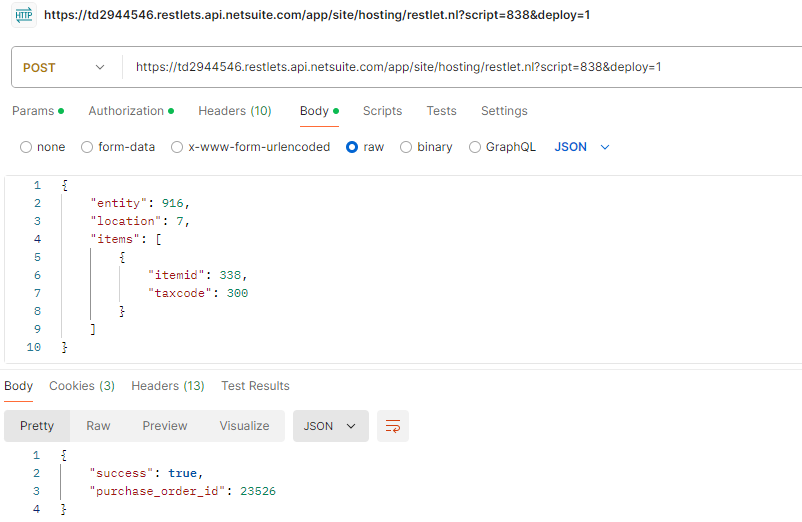
message: e.message

};

}

};

So, a PO created having id 23526



**Q5: UPDATE the Purchase Order Record in NetSuite through postman (don't hardcode data in script) also update a body field and add new line item.**

Here to update an existing record I used the **PUT** method. Here I updated the customer's name in the newly created purchase order i.e. 23526.

I changed the customer having id 916 to 618 by just passing the id of the record I want to edit and the new entity id I want to assign to the record.

// PUT method to update a record

const put = (requestBody) => {

try {

log.debug('Request body: ', requestBody);

var purchase\_order = record.load({

type: record.Type.PURCHASE\_ORDER,

id: requestBody.purchase\_order\_id,

isDynamic: true

})

log.debug('po id ', requestBody.purchase\_order\_id)

var new\_entity = requestBody.entity

log.debug('new entity is ', new\_entity)

purchase\_order.setValue({

fieldId: 'entity',

value: new\_entity

})

var updatedRecord = purchase\_order.save()

log.debug('PO updated having id ', updatedRecord)

return {

success: true,

updatedRecord: updatedRecord

};

} catch (e) {

return {

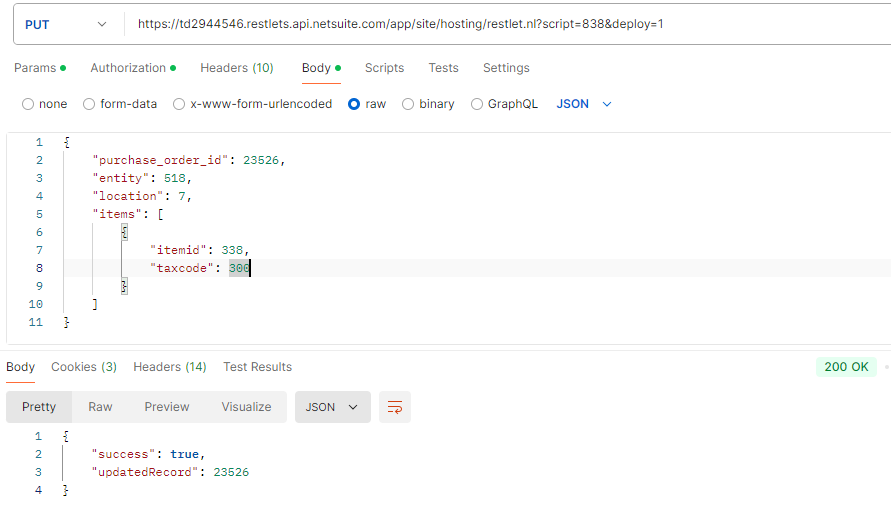
success: false,

message: e.message

};

}

};

Before updating the record:



After updating the record:

  
  
**Q6: DELETE the line items of the Purchase Order record also delete the whole record.**

I deleted the purchase order having id 23526 by using DELETE function.

// DELETE method to delete an existing record

const doDelete = (requestParams) => {

var po\_id\_dlt = requestParams.po\_id\_to\_dlt

log.debug('po id to delete ', po\_id\_dlt)

try{

var deleted\_id = record.delete({

type: record.Type.PURCHASE\_ORDER,

id: po\_id\_dlt

})

log.debug('PURCHASE order deleted with id ', deleted\_id)

return {

success: true,

deleted\_po\_id: po\_id\_dlt

}

}catch(e){

return{

success: false,

message: e.message

}

}

}

