ROHTO BUGS - Bug #1028

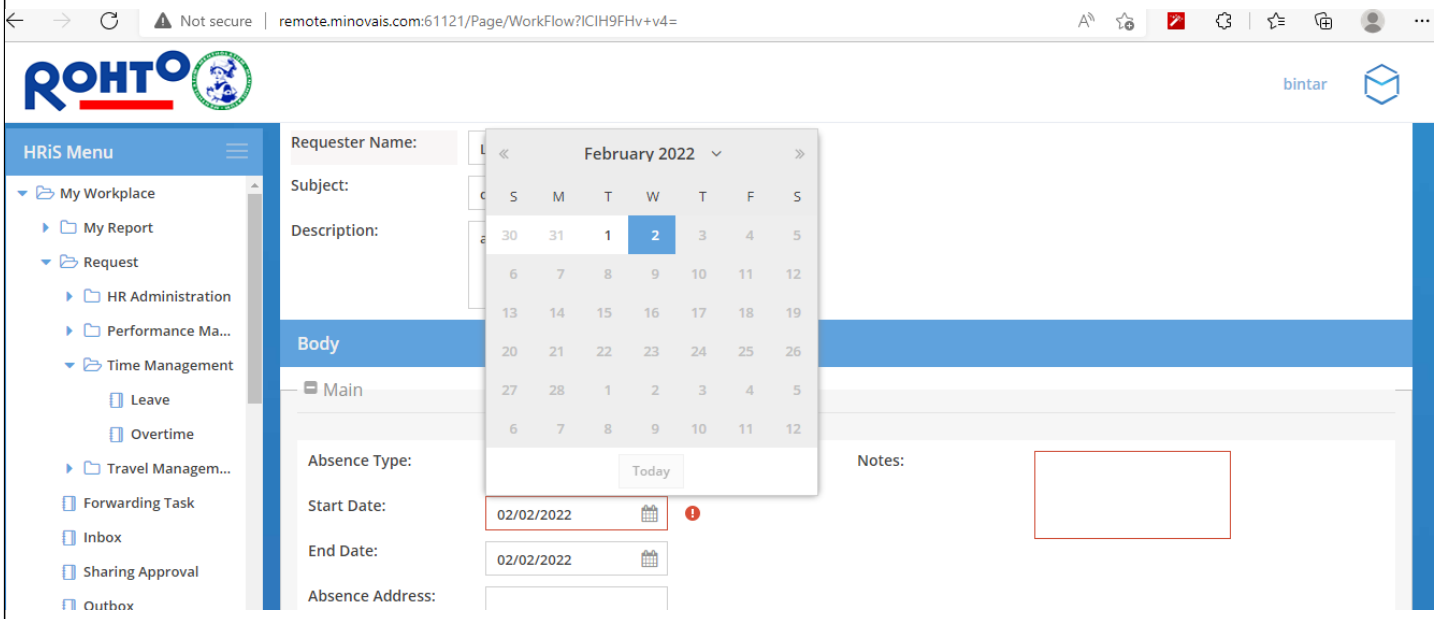
[ESS-LEAVE] validasi untuk backdate maksimal 14 hari tetapi terkadang masih bisa memilih lebih di backdate 15,16 dst

04/07/2022 03:10 PM - Muhammad Bintar

Status:	Closed	**Start date:**	04/07/2022
Priority:	Normal	**Due date:**	04/11/2022
Assignee:	M Azid Wahyudi	**% Done:**	0%
Category:		**Estimated time:**	0.00 hour
Target version:		**Spent time:**	1.00 hour

Description

dear tim developer mohon support nya untuk fixing pada pemilihan tanggal startdate dimana jika memilih backdate di hari ke 15 dst maka bisa terpilih padahal seharusnya tidak bisa.



aplikasi rohto production
remote.minovais.com:61121

database sql server 2019
remote.minovais.com, 1445\MSSQLSERVER2019

History

#1 - 04/08/2022 11:06 AM - M Azid Wahyudi

- Status changed from New to QA Test

- Assignee changed from M Azid Wahyudi to Muhammad Bintar

dear masbintar , udah bisa di cek ya

perubahan ada di bizproc -> id = 'leave_sj' di method saja perubahannya seperti dibawah ini

```
function getDateDifference(dateFrom, dateTo, differentType) {
var year = dateFrom.substring(0, 4);
var month = dateFrom.substring(4, 6);
var day = dateFrom.substring(6, 8);
var df = new Date(year, month - 1, day);
year = dateTo.substring(0, 4);
month = dateTo.substring(4, 6);
day = dateTo.substring(6, 8);
var dt = new Date(year, month - 1, day);
var diff = 0;
if (differentType === 'day') {
var diffDays = (dt.getTime() - df.getTime()) / (1000 * 60 * 60 * 24);
diff = diffDays;
```

```
} else if (differentType === 'month') {
var months;
months = (dt.getFullYear() - df.getFullYear()) * 12;
months -= df.getMonth();
months += dt.getMonth();
diff = months;
} else {
var diffYears = (dt.getFullYear() - df.getFullYear());
diff = diffYears;
}
return diff;
}
function prorateMNC() {
var Body = MinovaUtil.WORKFLOW.getViewBodyWorkflow();
var qstartdate = Body.down('[name=QuotaStartDate]').getValue();
var qstartdate_2 = Body.down('[name=QuotaStartDate2]').getValue();
var remaining_quota = Body.down('[name=RemainQuota]').getValue();
var remaining_quota_show = Body.down('[name=RemainQuota]');
var remaining_quota_2 = Body.down('[name=RemainQuota2]').getValue();
var remaining_quota_2_show = Body.down('[name=RemainQuota2]');
var leave_quota = Body.down('[name=OriginalQuota]');
var quota_taken = Body.down('[name=QuotaTaken]');
var total_remaining_quota = Body.down('[name=TotalRemainingQuota]');
var leave_quota_2 = Body.down('[name=OriginalQuota2]');
var total_remaining_quota_2 = Body.down('[name=TotalRemainingQuota2]');
var quota_taken_2 = Body.down('[name=QuotaTaken2]');
var start_cuti = Body.down('[name=StartDate]').getValue();
var now = Ext.Date.format(new Date(), 'Ymd');
var qprorate = getDateDifference(qstartdate, start_cuti, 'month') + 1;
if (qprorate >= 12) {
qprorate = leave_quota.getValue();
}
var qprorate_2 = 0;
if (!Ext.isEmpty(qstartdate_2)) {
qprorate_2 = getDateDifference(qstartdate_2, start_cuti, 'month') + 1;
}
if (qprorate_2 >= 12) {
qprorate_2 = leave_quota_2.getValue();
}
var quotapernahdiambil = parseInt(leave_quota.getValue()) - parseInt(remaining_quota);
var quotapernahdiambil_2 = parseInt(leave_quota_2.getValue()) - parseInt(remaining_quota_2);
var rs = 0;
var trs = 0;
var qtaken = 0;
var sisapengambilancuti1 = 0;
if (qprorate < leave_quota.getValue()) {
rs = qprorate - quotapernahdiambil;
trs = rs - quota_taken.getValue();
if (trs < 0 && qprorate_2 > 0) {
trs = 0;
sisapengambilancuti1 = parseInt(quota_taken.getValue()) - rs;
quota_taken.setValue(rs);
total_remaining_quota.setValue(remaining_quota - parseInt(quota_taken.getValue()));
}
remaining_quota_show.setValue(rs);
total_remaining_quota.setValue(trs);
} else {
rs = remaining_quota;
trs = total_remaining_quota.getValue();
remaining_quota_show.setValue(rs);
total_remaining_quota.setValue(trs);
}
if (!Ext.isEmpty(leave_quota_2.getValue()) && qprorate_2 < leave_quota_2.getValue()) {
rs = qprorate_2 - quotapernahdiambil_2;
qtaken = parseInt(quota_taken_2.getValue()) + sisapengambilancuti1;
quota_taken_2.setValue(qtaken);
total_remaining_quota_2.setValue(remaining_quota_2 - qtaken);
trs = rs - qtaken;
remaining_quota_2_show.setValue(rs);
total_remaining_quota_2.setValue(trs);
} else if (!Ext.isEmpty(leave_quota_2.getValue())) {
rs = remaining_quota_2;
qtaken = parseInt(quota_taken_2.getValue()) + sisapengambilancuti1;
quota_taken_2.setValue(qtaken);
total_remaining_quota_2.setValue(remaining_quota_2 - qtaken);
```

```
trs = total_remaining_quota_2.getValue();
remaining_quota_2_show.setValue(rs);
total_remaining_quota_2.setValue(trs);
}
}
function calculateCutiKhusus(formLeave, langid) {
var formValue = formLeave.getValues();
var body = MinovaUtil.WORKFLOW.getViewBodyWorkflow();
var reg = MinovaUtil.WORKFLOW.getRegId();
var regId = 0;
if (reg !== "") {
regId = parseInt(reg);
}
MinovaUtil.ExecuteParamQuery({
ID: "LeaveCutiKhususTUGU",
absenceType: formValue.AbsenceType,
stLeave: formValue.StartDate,
endLeave: formValue.EndDate,
empId: MinovaUtil.WORKFLOW.getEmpId(),
regId: regId,
langid: MinovaUtil.GetLangID()
}, function (s) {
var result = Ext.decode(s.responseText);
var dt = Ext.decode(Ext.decode(result.data));
formLeave.setValues(dt[0]);
}, function (f) {});
}
function leaveDetailandBlock(formLeave, langid) {
var formValue = formLeave.getValues();
var body = MinovaUtil.WORKFLOW.getViewBodyWorkflow();
var reg = MinovaUtil.WORKFLOW.getRegId();
var regId = 0;
if (reg !== "") {
regId = parseInt(reg);
}
MinovaUtil.ExecuteParamQuery({
ID: "LeaveDetailBlock",
absenceType: formValue.AbsenceType,
stLeave: formValue.StartDate,
endLeave: formValue.EndDate,
empId: MinovaUtil.WORKFLOW.getEmpId(),
regId: regId,
langid: MinovaUtil.GetLangID(),
leaveBlockList: 'AL1|AL2|AL3|AL4'
}, function (s) {
var result = Ext.decode(s.responseText);
var dt = Ext.decode(Ext.decode(result.data));
if (dt[0].WarningType === '1') {
MinovaMessageError("Error", dt[0].WarningText, "");
body.down('[name=StartDate]').setValue('');
} else {
formLeave.setValues(dt[0]);
}
}, function (f) {});
}
function setQuotaForm(field) {
try {
var body = MinovaUtil.WORKFLOW.getViewBodyWorkflow();
var forms = body.down('form').getForm();
var absenceType = body.down('[name=AbsenceType]');
var quotaDeduction = absenceType.displayTplData[0].QuotaDeduction;
var formLeave = body.down('form[name=PDSWFMDLEAVE]').getForm();
var formValue = formLeave.getValues();
if (quotaDeduction  'Y') {
if ((formValue.AbsenceType ! '') && (formValue.StartDate !== '') && (formValue.EndDate !== '')) {
leaveDetailandBlock(forms);
}
} else {
calculateCutiKhusus(forms);
}
} catch (scombo) {}
}
function leaveCheckDate(field, empid, sdt, edt, reg_id, langid) {
var body = MinovaUtil.WORKFLOW.getViewBodyWorkflow();
var absenceType = body.down('[name=AbsenceType]').getValue();
```

```
var stLeave = body.down('[name=StartDate]').getValue();
MinovaUtil.ExecuteParamQuery({
ID: 'LeaveCheckDate',
emp_id: empid,
start_date: sdt,
end_date: edt,
reg_id: reg_id,
langid: MinovaUtil.GetLangID()
}, function (s) {
var obj = Ext.decode(s.responseText);
var dt = Ext.decode(Ext.decode(obj.data));
if (dt[0].res == '1') {
body.down('[name=StartDate]').setValue('');
MinovaMessageError("Error", dt[0].code, "");
} else {
setQuotaForm();
}
}, function (f) {});
}
function calculate(field) {
var reg = MinovaUtil.WORKFLOW.getRegId();
var body = MinovaUtil.WORKFLOW.getViewBodyWorkflow();
var taskid = MinovaUtil.WORKFLOW.getTaskId();
var sDate = body.down('field[name=StartDate]');
var eDate = body.down('field[name=EndDate]');
var empid = MinovaUtil.WORKFLOW.getEmpId();
var reg_id = 0;
var sdt = sDate.getValue();
var edt = eDate.getValue();
var langid = MinovaUtil.SESSIONS.LangID;
if (reg != "" && reg != undefined) {
reg_id = parseInt(reg);
}
if (sdt != "" && edt != "" && sdt != undefined && edt != undefined) {
leaveCheckDate(field, empid, sdt, edt, reg_id, langid);
}
}
function selectDate(field, combo) {
var me = this;
var now = Ext.Date.format(new Date(), 'Ymd');
var body = MinovaUtil.WORKFLOW.getViewBodyWorkflow();
var absenceType = body.down('[name=AbsenceType]').getValue();
var stLeave = body.down('[name=StartDate]').getValue();
var taskid = MinovaUtil.WORKFLOW.getTaskId();
if (taskid === 'create' || taskid === 'start') {
MinovaUtil.ExecuteParamQuery({
ID: "LeaveCheckBackdate",
startDate: stLeave,
absenceType: absenceType,
langId: MinovaUtil.GetLangID()
}, function (s) {
var result = Ext.decode(s.responseText);
var dt = Ext.decode(Ext.decode(result.data));
if (dt[0].res === '1') {                    // penambahan validasi
MinovaMessageError("Error", dt[0].code, "");
body.down('[name=StartDate]').setValue('');
} else { // end penambahan validasi
changeDateRange(combo);
}
}, function (f) {})
} else {
changeDateRange(combo);
}
}
function fillEndDate(field, combo) {
var body = MinovaUtil.WORKFLOW.getViewBodyWorkflow();
var absenceType = body.down('[name=AbsenceType]').getValue();
var stLeave = body.down('[name=StartDate]').getValue();
var taskid = MinovaUtil.WORKFLOW.getTaskId();
if (taskid === 'create' || taskid === 'start') {
MinovaUtil.ExecuteParamQuery({
ID: "LeaveAutofillEndDateTUGU",
EmployeeID: MinovaUtil.WORKFLOW.getEmpId(),
AbsenceType: absenceType,
StartDate: stLeave
```

```
}, function (s) {
var result = Ext.decode(s.responseText);
var dt = Ext.decode(Ext.decode(result.data));
body.down('[name=EndDate]').setValue(dt[0].EndDate);
}, function (f) {});
}
}
function changeDateRange(field, combo) {
var me = this;
var now = Ext.Date.format(new Date(), 'Ymd');
var body = MinovaUtil.WORKFLOW.getViewBodyWorkflow();
var absenceType = body.down('[name=AbsenceType]').getValue();
var stLeave = body.down('[name=StartDate]').getValue();
calculate(field);
}
function changeDelegate(field) {
var body = MinovaUtil.WORKFLOW.getViewBodyWorkflow();
var DelegateTo = body.down('[name=DelegateTo]').getValue();
MinovaUtil.ExecuteParamQuery({
ID: "LeaveAutofillDelegateTUGU",
EmployeeID: MinovaUtil.WORKFLOW.getEmpId(),
Now: MinovaUtil.GetNowDate()
}, function (s) {
var result = Ext.decode(s.responseText);
var dt = Ext.decode(Ext.decode(result.data));
if (dt.length > 0) {
body.down('[name=DelegateName]').setValue(dt[0].DelegateName);
} else {
body.down('[name=DelegateName]').setValue('');
}
}, function (f) {});
}
function getFormHeader() {
console.log('>>getFormHeader');
var formHeader = Ext.ComponentQuery.query('wf-request-formHeader')[0];
if (formHeader !== undefined) {
formHeader = Ext.ComponentQuery.query('wf-request-formHeader')[0].getForm().getValues();
} else {
formHeader = Ext.ComponentQuery.query('wf-run-formHeader')[0].getForm().getValues();
}
console.log('<<getFormHeader');
return formHeader;
}
function selectAbsenceType(combo, field) {
var me = this;
var empId = MinovaUtil.WORKFLOW.getEmpId();
var body = MinovaUtil.WORKFLOW.getViewBodyWorkflow();
var now = Ext.Date.format(new Date(), 'Ymd');
var absenceType = body.down('[name=AbsenceType]').getValue();
combo.up().up().query('textfield[name=AbsenceQuotaType]')[0].setValue(combo.displayTplData[0].AbsenceType);
body.down('[name=QuotaDeduction]').setValue(combo.displayTplData[0].QuotaDeduction);
MinovaUtil.ExecuteParamQuery({
ID: 'LeaveBlockContinue',
EmployeeId: empId,
QuotaType: combo.displayTplData[0].AbsenceType
}, function (s) {
var obj = Ext.decode(s.responseText).data[0];
body.down('[name=BlockContain]').setValue(obj.result);
}, function (f) {});
changeDateRange(combo);
}
function expandAbsenceType(field, eOpts) {
var me = this;
var empId = MinovaUtil.WORKFLOW.getEmpId();
field.store.load({
params: {
emp_id: empId
}
});
}
function CekLvDate() {
debugger;
var body = MinovaUtil.WORKFLOW.getViewBodyWorkflow();
var emp = MinovaUtil.WORKFLOW.getEmpId();
var StartDate = body.down('field[name=StartDate]').getValue();
```

```
var Lv = '';
var lv2 = '';
MinovaUtil.ExecuteParamQuery({
ID: 'LVCKDATE',
StartDate: StartDate,
EmpID: emp
}, function (s) {
d
var result = Ext.decode(s.responseText);
var dt = Ext.decode(Ext.decode(result.data));
if (dt.length == 0) {
Lv = '';
} else {
var val = dt[0].Tglrovr;
var val2 = dt[0].Tglrun;
Lv = val;
lv2 = val2;
}
});
if (Lv == '1') {
MinovaMessageError("ERROR", "CEK19LV", "", "");
body.down('field[name=StartDate]').setValue('');
body.down('field[name=EndDate]').setValue('');
} else if (lv2 == '1') {
MinovaMessageError("ERROR", "CEKRUNLV", "", "");
body.down('field[name=StartDate]').setValue('');
body.down('field[name=EndDate]').setValue('');
}
}
function CekLvMin() {
var Body = MinovaUtil.WORKFLOW.getViewBodyWorkflow();
var ttlquo = Body.down('[name=TotalRemainingQuota]').getValue();
if (ttlquo < 0) {
MinovaMessageError("ERROR", "CEKLVQUO", "", "");
Body.down('field[name=EndDate]').setValue('');
}
}
function CutiThn() {
var body = MinovaUtil.WORKFLOW.getViewBodyWorkflow();
var abs = body.down('[name=AbsenceType]').getValue();
var StartDate = body.down('[name=StartDate]');
var StartD = body.down('[name=StartDate]').getValue();
var EndDate = body.down('[name=EndDate]');
var ss = '19000101';
var ee = '99991231';
var now = MinovaUtil.GetNowDate();
var abss = body.down('[name=TotalAbsenceTaken]');
var res = '';
MinovaUtil.ExecuteParamQuery({
ID: 'CEKTWOWEEKLEAVE'
}, function (s) {
var result = Ext.decode(s.responseText);
var dt = Ext.decode(Ext.decode(result.data));
var val = dt[0].res;
res = val;
});
if (abs == '1000') {
StartDate.setMinValue(res.substring(6, 8) + '/' + res.substring(4, 6) + '/' + res.substring(0, 4));
StartDate.setMaxValue(ee.substring(6, 8) + '/' + ee.substring(4, 6) + '/' + ee.substring(0, 4));
EndDate.setMinValue(StartD.substring(6, 8) + '/' + StartD.substring(4, 6) + '/' + StartD.substring(0, 4));
EndDate.setMaxValue(ee.substring(6, 8) + '/' + ee.substring(4, 6) + '/' + ee.substring(0, 4));
} else {
StartDate.setMinValue(ss.substring(6, 8) + '/' + ss.substring(4, 6) + '/' + ss.substring(0, 4));
EndDate.setMaxValue(ee.substring(6, 8) + '/' + ee.substring(4, 6) + '/' + ee.substring(0, 4));
}
}
```

#2 - 04/08/2022 01:57 PM - Muhammad Bintar

- Status changed from QA Test to Deploy

oke sudah deployy

#3 - 04/08/2022 01:58 PM - Muhammad Bintar

- Status changed from Deploy to Closed

- Assignee changed from Muhammad Bintar to M Azid Wahyudi

Files

clipboard-202204071501-oobqm.png	85.6 KB	04/07/2022	Muhammad Bintar
bandicam 2022-04-07 14-57-38-833.mp4	1.79 MB	04/07/2022	Muhammad Bintar