VIEW data  
API responses op een *data* call hebben een vaste structuur. Hieronder een voorbeeld voor de Api call '/translations':

{

data:[{colname:value},...] //data te vertonen in kolom

,config:{ // configuratie van view

row\_editable: true // moet 'Edit' button bij rij worden vertoond

,columns:[{// configuratie van kolommen

columnName: 'id' //welke data wordt in deze kolom vertoond

,visible:true //wordt de kolom vertoond

,header\_visible:true //wordt de kolom header vertoond

,header\_text:'Key' //kolom header tekst

,cell\_editable: false //is de individuele kolom cel te wijzigen

},{

columnName: 'DisplayName'

,visible:true

,header\_visible:true

,header\_text:'Vertaling'

,cell\_editable: true

}]

,pagination: { //configuratie voor de pagineringsbuttons

enable:true //worden pagineringsbuttons vertoond

,maxSize: 10 //hoeveel pagina buttons tonen

,itemsPerPage:15 //aantal regels per pagina

}

}

}

API endpoint data  
De API calls die de client maakt worden niet hard gecodeerd in de frontend maar opgevraagd aan de backend (bijv via '/APIconfig'. De data die de client terugverwacht ziet er als volgt uit - hier voor /translations en /joblist endpoints:

{

translations : { //configuratie van translation endpoint

api\_url: 'http://api-development.hrmatches.com' + '/translation' // url

,api\_method: 'POST' //http method

,api\_params: [{ // of en welke parameters moeten worden meegestuurd

key: 'language'

,value: 'nl\_NL'

},{

key:'languageKey'

,value: ''

}]

}

,joblist: {

api\_url: 'http://api-development.hrmatches.com' + '/joblist'

,api\_method: 'GET'

,api\_params: []

}

}