Boost Space Swagger UI

https://apidoc.boost.space/

Servers

Computed URL: `https://test.boost.space/api`

Server variables

system	

Auth Authentication

1. Using /auth request
2. Using Private Token in header

Request /auth

User sends POST request with supplying username and password inside request data. Authentication is then saved into session data and kept valid for 20 minutes (sliding authentication - every request resets timeout to another 20 minutes)

Private token in header

User firstly generates Private token, using request /auth/:userId/token. Token can then be sent in every request header Authorization or X-Authorization, with Bearer header syntax: `Authorization: Bearer <TOKEN>` or `X-Authorization: Bearer <TOKEN>` .Every Private token can be specifically configured with Create / Read / Update / Delete permissions for each (or all) modules. Bearer Token authentication ain't saved into Sessions and does not have sliding (expires immediately after request is finished)
For better understanding of Private token permissions, please see Token API

Automatization Automatization module Handles Automatization tasks. Use can specify bunch of triggers, and bunch of actions. Actions can be triggered by one or more triggers. Triggers can be (so far) of type CREATE / UPDATE / DELETE / FIELD_CHANGE and can trigger certain actions, when some record has been created / updated / deleted or any of its fields changed to expected value (used for system status changes actions) Actions are specifications about what is about to happen, when trigger is fired. So fat, it can send e-mail, send notification or visit certain url (using curl request) Triggers are meant to be configured using urls automatization/trigger, actions using automatization/action Use field action.triggers to specify which trigger should fire that action (can be empty)

BusinessOfferModule for handling business offers, one of the submodules in whole CRM system. When connecting to contact and parent business case has specified contact or contact group, connecting contact must be part of theseBusiness offer is having field Number, which is its unique identification. This number is created automatically upon creation and cannot be changed. Number is created using mask in its parent space.System automatically finds latest number in database, according to correct mapping and increments it.Business Offer can be connected to Labels. Only labels specified in parent space with the type business_offer are allowed to be connected here.

BusinessOrderModule for handling business orders, one of the submodules in whole CRM system. When connecting to contact and parent business case has specified subjcet or contact group, connecting contact must be part of theseBusiness order is having field Number, which is its unique identification. This number is created automatically upon creation and cannot be changed. Number is created using mask in its parent business case → business process.System automatically finds latest number in database, according to correct mapping and increments it.Business Order can be connected to Labels. Only labels specified in Business process with the type business_order are allowed to be connected here.

ContactModule to handle contact request. Contacts are end-points for bussiness operations, it can be companies (mostly) or persons, etc. Contacts must be always connected to its space.Permissions to all CRUD operations are then taken from its space. Contact can be connected to addresses, spaces and custom informations.Spaces are connected by field spaces in the POST / PUT request.Addresses and Custom Informations are connected by subrequests /contact/:idS/address/:id or /contact/:idS/custom-info/:idSub-request for address can contain specific fields type, category and primary. Category can hold the ID of wanted address custom type. Valid custom types are the one having custom_type.category = type_of_address.When primary is sent and set to TRUE, all other addresses related to this record with same type, are set to FALSE (only one primary address having this type can exist). It is important to know that the mark of primary, category and type are on the connection level - therefore the same address can be to one record primary and invoicing, but the same address can serve as workshop address for another record.

ExportExport moduleUniversal exporting module, able to export records from any module, filtered by IDS (rows) and columns

Export stock requestExport stock request into PDF

Invoice**Module for handling invoices, tightly connected to ordersInvoice is having field Number, which is its unique identification. This number is created automatically upon creation and cannot be changed. Number is created using mask in its parent space.System automatically finds latest number in database, according to correct mapping and increments it.Invoice is using SystemStatus to determine whether invoice is created/payed/overdue . When Invoice system status is changed to Payed, system automatically fills the PayedDate field**

Resource**Resources Resources are a set of user-defined resources (i.e. machines), which are connectible to Categories and Teams. Resources represents a limited user-like usability and they are used for work plan scheduling and reporting. Use resources property on teams and categories to connect them to specific category / team and classic CRUD operations to work with resources themselves. Use resource/:id/schedule api endpoints to manage and plan resource schedule Connect resources to work-reports using standard module/recordId properties on work-report - just instead of user/userId set resource/resourceId**

string
string
string
array<any>
object
array<any>
array<integer>
object
array<any>
object
array<any>
object
array<any>
object
array<any>
object
array<any>
object
array<any>
object
array<any>
object
array<any>
object
array<any>

object
array<any>
object
array<any>
object
array<any>
object
object
object
array<any>
object
object
object
object
object
array<any>
object
object
object
object
object
object
array<any>
array<any>
object
array<any>
array<integer>
object
array<any>
object
array<any>
array<integer>
object
array<any>
array<integer>
object
array<any>
object
array<any>
object
array<any>
object
array<any>
object
object

array<any>
array<integer>
object
object
array<any>
object
array<any>
object
array<any>
object
array<any>
object
array<any>
object
array<any>
object
object
array<any>
object
array<any>
object
array<any>
object
array<any>
object
array<any>
object
array<any>
object
array<any>
object
array<any>
object
array<any>
object
array<any>
object
object
object
object
object
array<any>
stringbinary
array<any>
array<any>
array<any>
object

object
object
array<any>
object
array<any>
array<integer>
object
array<any>
object
array<any>
object
object
object
object
array<any>
array<any>
array<integer>
object
array<any>
object
array<any>
object
array<any>
object
object
array<any>
array<integer>
object
array<any>
object
object
array<any>
object
array<any>
object
array<any>
array<any>
object
object
object
array<any>
array<integer>
object
object
array<any>

array<any>
object
object
array<any>
object
array<any>
object
array<any>
object
array<any>
object
array<any>
object
array<any>
object
array<any>
object
array<any>
object
array<any>
object
array<any>
object
object
object
array<any>
object
array<any>
object
array<any>
object
array<any>
object
array<any>
object
array<any>
object
array<any>
object
array<any>
object
array<any>
object
object
object

array<any>
object
array<any>
array<integer>
object
array<any>
object
array<any>
object
array<any>
object
object
array<any>
object
array<any>
object
object
array<any>
object
array<integer>
array<any>
object
array<any>
object
array<object>
array<any>
object
array<any>
object
array<any>
object
array<any>
object
object
array<any>
object
array<any>
object