Entity APIs

The following entity APIs are provided: Asset, Authority, Lookup, Repair Order, User, and Work Order. These entities are described in more detail in the following sections.

Asset

Use the Asset API entity to work with assets in the AgileAssets system as well as inventories associated with the assets. This entity allows both GET and POST calls.

The following GET calls may be made to request data from the AgileAssets system via the API:

{baseURL}/asset

This call retrieves all asset types in the system. An example of this type is shown below:

http://vega:8080/ams_ky/rest/v1/asset?q={"projection":["ASSET_TYPE_ID", "ASSET_TYPE_NAME"],"page":{"number":4,"size":2}}


```
 {
 "SETUP_ASSET_TYPE"
: [
 {
"ASSET_TYPE_ID": 3,

"ASSET_TYPE_NAME":
"Equipment"
},
 {
"ASSET_TYPE_ID": 5,

"ASSET_TYPE_NAME":
"Employee"
}
]
}
```


{baseURL}/asset/{id}

This call retrieves asset type information for a particular asset ID. An example of this type is shown below:

http://vega:8080/ams_ky/rest/v1/asset/639


```
{
ASSET_TYPE_ID: 639,
ASSET_TYPE_NAME: "Sig
nals",
PARENT_ID: 2,
LEVEL_ID: 3,
INVENTORY_TABLE_NAME:
"SIGNAL_INVENTORY",
NUM_CHILD: 0,
USER_UPDATE: "EUGENE",

DATE_UPDATE: "2011053
1173140",
ORDER_ID: 50,
PM_APPL: 1,
DEFECT_SURV_APPL: 0,
LOCATION_APPL: 1,
MODULE_ID: 3,
INVENTORY_MEASURE_EXP:
 "1"
}
```


{baseURL}/asset/{name}

This call retrieves the header inventory table for the asset with name {name}. This is not case sensitive. If the name contains spaces or slashes, omit them entirely in the URL. An example of this type is shown below:

http://vega:8080/ams_ky/rest/v1/asset/sign


```
{
ASSET_TYPE_ID: 6,
ASSET_TYPE_NAME: "Si
gn",
PARENT_ID: 1,
LEVEL_ID: 2,
INVENTORY_TABLE_NAME:
 "SIGN_INVENTORY",
NUM_CHILD: 0,
USER_UPDATE: "BABAK",

DATE_UPDATE: "201009
09121727",
MODULE_ID: 3,
MOBILE_LAYER_NAME: "
ASSET_LOC_POINT"
}
```


{baseURL}/asset/{name}/parent

This call retrieves the parent asset type for an asset with name {name} Note: This is not case sensitive. If the name contains spaces or slashes, omit them entirely in the URL. An example of this type is shown below:

http://vega:8080/ams_ky/rest/v1/asset/walls/parent


```
{
ASSET_TYPE_ID: 2,
ASSET_TYPE_NAME: "Roa
d Sections",
PARENT_ID: 1,
LEVEL_ID: 2,
INVENTORY_TABLE_NAME:
"SECTION_INVENTORY",
NUM_CHILD: 0,
USER_UPDATE: "EUGENE",

DATE_UPDATE: "2011053
1173148",
ASSET_TYPE_CAT: 2,
ORDER_ID: 1,
PM_APPL: 1,
DEFECT_SURV_APPL: 1,
LOCATION_APPL: 1,
MODULE_ID: 3,
INVENTORY_MEASURE_EXP:
 "1"
}
```


{baseURL}/asset/{id}/parent

This call retrieves the parent asset type for asset with ID {id}. An example of this type is shown below:

http://vega:8080/ams_ky/rest/v1/asset/963/parent

```
{
ASSET_TYPE_ID: 2,
ASSET_TYPE_NAME: "Roa
d Sections",
PARENT_ID: 1,
LEVEL_ID: 2,
INVENTORY_TABLE_NAME:
"SECTION_INVENTORY",
NUM_CHILD: 0,
USER_UPDATE: "EUGENE",

DATE_UPDATE: "2011053
1173148",
ASSET_TYPE_CAT: 2,
ORDER_ID: 1,
PM_APPL: 1,
DEFECT_SURV_APPL: 1,
LOCATION_APPL: 1,
MODULE_ID: 3,
INVENTORY_MEASURE_EXP:
 "1"
}
```


{baseURL}/asset/{id}/inventory

This call retrieves the inventory table for an asset with this numeric ID. An example of this call is shown below:

http://vega:8080/ams_ky/rest/v1/asset/639/inventory

{baseURL}/asset/{name}/inventory

This call retrieves the header inventory table for the asset with name {name}. Note: This is not case sensitive. If the name contains spaces, tabs, line breaks, or slashes, omit them entirely in the URL. An example of this call for the Road Sections inventory is shown below (note that since "Road Sections" contains spaces, these are omitted in the call):

http://vega:8080/ams_ky/rest/v1/asset/roadsections/inventory

{baseURL}/asset/{id}/group

This call retrieves inventory class codes for the asset identified by the {id} asset ID. An example of this type is shown below:

http://vega:8080/ams_ky/rest/v1/asset/18/group


```
[
-{
LAKE_CLASS_CODE_ID: 1,
LAKE_CLASS_CODE_NAME: "Full"
},
-{
LAKE_CLASS_CODE_ID: 2,
LAKE_CLASS_CODE_NAME: "Dry"
}
]
```


{baseURL}/asset/{name}/group
This call retrieves all inventory class codes for the asset identified by the {name} asset name. An example of this type is shown below:

http://vega:8080/ams_ky/rest/v1/asset/sign/group


```
[
-{
DATE_UPDATE: "20100909122723",
LEVEL_ID: 2,
NUM_CHILD: 0,
PARENT_ID: 11821,
PERIODIC_MAINT_ID: -1,
SIGN_CLASS_CODE_ID: 11820,
SIGN_CLASS_CODE_NAME: "Gantry",
USER_UPDATE: "BABAK"
},
-{
DATE_UPDATE: "20100909122700",
LEVEL_ID: 1,
NUM_CHILD: 1,
ORDER_ID: 1,
PERIODIC_MAINT_ID: -1,
SIGN_CLASS_CODE_ID: 11821,
SIGN_CLASS_CODE_NAME: "All",
USER_UPDATE: "BABAK"
},
-{
DATE_UPDATE: "20100909122739",
LEVEL_ID: 2,
NUM_CHILD: 0,
PARENT_ID: 11821,
PERIODIC_MAINT_ID: -1,
SIGN_CLASS_CODE_ID: 11822,
SIGN_CLASS_CODE_NAME: "Post",
USER_UPDATE: "BABAK"
}
]
```


{baseURL}/asset/{id}/key
This call retrieves the inventory table key column name, where {id} is the numeric ID of the asset. An example of this type is shown below:

http://vega:8080/ams_ky/rest/v1/asset/638/key


```
{
key: "SIGN_ID"
}
```


{baseURL}/asset/{name}/key
This call retrieves the inventory table key column name, where {name} is the name of the asset. An example of this type is shown below:

http://vega:8080/ams_ky/rest/v1/asset/signs/key


```
{
key: "SIGN_ID"
}
```


{baseURL}/asset/{id}/inventory/{invenId}
This call retrieves a particular inventory element {invenId} from an inventory table for the asset with the {id} id. An example of this type is shown below:

http://vega:8080/ams_ky/rest/v1/asset/18/inventory/3228834


```
{
LAKE_CLASS_CODE_ID: 1,
LAKE_NAME: "CouterClock",
LAKE_ID: 3228834,
GEOM: "MULTIPOINT EMPTY"
}
```


{baseURL}/asset/{name}/inventory/{invenId}
This call retrieves a particular inventory element {invenId} from an inventory table for the asset named {name}. An example of this type is shown below:

http://vega:8080/ams_ky/rest/v1/asset/sign/inventory/200310


```
{
COMMENT_STR: "TEST",
DATE_UPDATE: "20101223132745",
SIGN_CLASS_CODE_ID: 11822,
SIGN_ID: 200310,
SIGN_NAME: "Alex",
SIGN_STATUS_ID: 106,
USER_UPDATE: "JG",
LOC_IDENT: 793926,
OWNER_ID: 50,
GEOM: "MULTIPOINT ((5005996.75895799 3537933.89637713))"
}
```


{baseURL}/asset/{id}/inventory/{invenId}/detail
This call retrieves details for a particular inventory element {invenId} from an inventory table for the asset with {id} id. An example of this type is shown below:

http://vega:8080/ams_la/rest/v1/asset/1022/inventory/2416640/detail


```
{
USER_UPDATE: "SHANKER",
ITS_SITE_ID: 2416640,
DATE_UPDATE: "20120321000000",
LOC_IDENT: 20750660
}
```


{baseUrl}/asset/{name}/inventory/{invenId}/detail
This call retrieves details for a particular inventory element {invenId} from an inventory table for the asset named {name}. An example of this type is shown below:

http://vega:8080/ams_la/rest/v1/asset/itssites/inventory/2416640/detail


```
{
USER_UPDATE: "SHANKER",
ITS_SITE_ID: 2416640,
DATE_UPDATE: "20120321000000",
LOC_IDENT: 20750660
}
```


POST calls are made to create data in the AgileAssets system through the API. When a new item is created, the correct ID is returned to the caller (providing the inventory element ID via a POST call is useless because POST is assumed to only create data). The following POST calls are available:

- **{baseUrl}/asset/{id}/inventory**
  This call creates new inventory element. The call should include:
- - Header: Content-Type: application/json
  - Data: JSON array that represents one or more records of the inventory table. The required fields are the minimum set of fields that may appear in the array (with the exception of ID fields that are assigned by the system).

For example, the following POST call will insert two sign assemblies (in this example, the URL provides the asset type ID and the system uses this to identify the correct inventory table name):

```
       curl -i -X POST -H 'Content-Type: application/json' http://localhost:8080/ams_dev/rest/v1/asset/638
/inventory/ -d '[{"SIGN_NAME":"Alex Protyagov1"},{"SIGN_NAME":"Protyagov"}]'
```

The system returns the following: [2416662,2416661]

Authority

Use the Authority API entity to retrieve authorities from an AgileAssets system. This entity allows only GET calls.

The following GET calls may be made to request data from the AgileAssets system via the API:

{baseURL}/authority
This call retrieves records for authorities in the system. Use the projection of a Q object to limit the number of returned columns. An example of this call is shown below:
http://vega:8080/ams_la/rest/v1/authority

{baseURL}/authority/{id}
This call retrieves authorities for the administrative unit identified by the ID {id}. Use the projection of a Q object to limit the number of returned columns. An example of this type is shown below:

http://vega:8080/ams_la/rest/v1/authority/1317


```
{
OWNER_ID: 1317,
OWNER_NAME: "LaDOTD",
LEVEL_ID: 1,
USER_UPDATE: "LEN",
DATE_UPDATE: "20100920074240",
SETUP_OWNER_S_1: "1",
OWNER_CAT: 1317,
NUM_CHILD: 0,
ORDER_ID: 1,
EXT_OWNER_ID: "00000000",
PLAN_ID: 283,
UT_APPROP_CODE: "0010",
PARISH_ID: 1
}
```


{baseURL}/authority/{id}/name
This call retrieves the name of a particular authority unit identified by the ID {id}. An example of this type is shown below:

http://vega:8080/ams_la/rest/v1/authority/1317


```
{
OWNER_ID: 1317,
OWNER_NAME: "LaDOTD",
LEVEL_ID: 1,
USER_UPDATE: "LEN",
DATE_UPDATE: "20100920074240",
SETUP_OWNER_S_1: "1",
OWNER_CAT: 1317,
NUM_CHILD: 0,
ORDER_ID: 1,
EXT_OWNER_ID: "00000000",
PLAN_ID: 283,
UT_APPROP_CODE: "0010",
PARISH_ID: 1
}
```


{baseURL}/authority/{id}/labor
This call retrieves retrieves the labor inventory of a particular authority unit identified by the ID {id}. Use the projection of a Q object to limit the number of returned columns. An example of this type is shown below:

http://localhost:8080/ams_dev/rest/v1/authority/1949/labor?q={projection:[LABOR_ID,LABOR_NAME]}


```
[
-{
LABOR_ID: 81944,
LABOR_NAME: "BALLARD, BILLY"
},
-{
LABOR_ID: 81945,
LABOR_NAME: "HAGAN, SAM"
},
-{
LABOR_ID: 81946,
LABOR_NAME: "PITTMAN, KEITH"
},...
```


{baseURL}/authority/{id}/child
This call retrieves the child units of a particular authority unit identified by the ID {id}. An example of this type is shown below:

http://vega:8080/ams_la/rest/v1/authority/56/child


```
[
-{
OWNER_ID: 595,
OWNER_NAME: "S56/G001 - INTELLIGENT TRANSPORTATION SY",
PARENT_ID: 56,
LEVEL_ID: 3,
DATE_BIRTH: "20100305122253",
USER_UPDATE: "EUGENE",
DATE_UPDATE: "20100305122253",
OWNER_CAT: 56,
NUM_CHILD: 0,
EXT_OWNER_ID: "50356761"
},
-{
OWNER_ID: 596,
OWNER_NAME: "S56/G001 - ITS INFRASTRUCTURE MANAGEMENT",
PARENT_ID: 56,
LEVEL_ID: 3,
DATE_BIRTH: "20100305122253",
USER_UPDATE: "EUGENE",
DATE_UPDATE: "20100305122253",
OWNER_CAT: 56,
NUM_CHILD: 0,
EXT_OWNER_ID: "50381313"
},...
```


{baseURL}/authority/{id}/parent
This call retrieves the parent unit of a particular authority unit identified by the ID {id}. An example of this type is shown below:

http://localhost:8080/ams_dev/rest/v1/authority/61/parent


```
{
OWNER_ID: 1317,
OWNER_NAME: "LaDOTD",
LEVEL_ID: 1,
USER_UPDATE: "LEN",
DATE_UPDATE: "20100920074240",
SETUP_OWNER_S_1: "1",
OWNER_CAT: 1317,
NUM_CHILD: 0,
ORDER_ID: 1,
EXT_OWNER_ID: "00000000",
PLAN_ID: 283,
UT_APPROP_CODE: "0010",
PARISH_ID: 1
}
```

{baseURL}/authority/{id}/user
This call retrieves the users assigned to a particular authority unit identified by the ID {id}. An example of this type is shown below:

http://vega:8080/ams_la/rest/v1/authority/56/user


```
[
"ALEX",
"BOYD",
"BRANDON",
"DCKIBBE1",
"LEN",
"LOS",
"MA",
"MARK",
"SHANKER",
"SYS_API",
"TESTER"
]
```


{baseURL}/authority/{id}/labor/shortlist
This call retrieves the labor shortlist for a particular authority unit identified by the ID {id}. (Note: If {id} is omitted, the shortlists of all authority units is returned.) An example of this type is shown below:

http://vega:8080/ams_la/rest/v1/authority/482/labor/shortlist


```
{
LABOR_ID: 2413982,
OWNER_ID: 482,
LAST_NAME: "CHRISTOPHER",
FIRST_NAME: "WILLIAM",
MID_INITIAL: "R",
LABOR_CLASS_CODE_ID: 231997,
LABOR_STATUS_ID: 25,
RATE: 12.73,
ADDRESS: "77100 IBERVILLE DR",
PHONE: " - ",
USER_UPDATE: "INTERFACE7",
DATE_UPDATE: "20110808103002",
START_DATE: "20090423000000",
CITY: "MARINGOUIN",
STATE: "LA",
POSTAL: "70757",
LABOR_NAME: "WILLIAM R CHRISTOPHER",
SAP_EXT_EMPLOYEE_ID: "00230790"
}
```


{baseURL}/authority/{id}/equipment
This call retrieves the equipment shortlist for a particular authority unit identified by the ID {id}. (Note: If {id} is omitted, the shortlists of all authority units is returned.) An example of this type is shown below:

http://vega:8080/ams_la/rest/v1/authority/239/equipment


```
[
-{
EQUIPMENT_ID: 2213309,
EQUIPMENT_CLASS_CODE_ID: 231605,
EQUIPMENT_NAME: "500033504",
OWNER_LOCATED: 3,
MAKE: 322,
MODEL: 1557,
USER_UPDATE: "INTERFACES",
DATE_UPDATE: "20100831121519",
ACCUMULATED_ODOMETER: 0,
DESCRIPTION: "2007 JOHN DEERE (Skid Steer)",
OWNER_ID: 239,
OWNER_ASSIGNED: 239,
GARAGE_ASSIGNED: 1317,
EQUIPMENT_STATUS_ID: 893,
EXT_EQUIPMENT_ID: "000500033504"
},
-{
EQUIPMENT_ID: 2213154,
EQUIPMENT_CLASS_CODE_ID: 231428,
EQUIPMENT_NAME: "500033307",
OWNER_LOCATED: 3,
MAKE: 363,
MODEL: 101,
USER_UPDATE: "INTERFACES",
DATE_UPDATE: "20100831121519",
ACCUMULATED_ODOMETER: 0,
DESCRIPTION: "1967 LE ROI (Air Compressor)",
OWNER_ID: 239,
OWNER_ASSIGNED: 239,
GARAGE_ASSIGNED: 1317,
EQUIPMENT_STATUS_ID: 893,
EXT_EQUIPMENT_ID: "000500033307"
},
.........
```


{baseURL}/authority/{id}/material/shortlist

This call retrieves the material shortlist for a particular authority unit identified by the ID {id}. (Note: If {id} is omitted, the shortlists of all authority units is returned.) An example of this type is shown below:

http://vega:8080/ams_la/rest/v1/authority/518/material/shortlist


```
[
-{
STOCK_ID: 1883132,
STOCK_NAME: "10523 - HERBICIDE,ROUNDUP PRO,2.5 GAL/CONT (GAL - Gallon (US)) @ 276R - Kentwood MU",
OWNER_ID: 518,
MASTER_CODE_ID: 10560
},
-{
STOCK_ID: 1884276,
STOCK_NAME: "11595 - BITUMINOUS MIX,F/HOT APPL,RECLAIMED (YD3 - Cubic Yard) @ 276R - Greensburg MU",
OWNER_ID: 518,
MASTER_CODE_ID: 11170
},
-{
STOCK_ID: 1884153,
STOCK_NAME: "11586 - ASPHALT,EMULSIFIED,CATIONIC,CRS2,RAPID (GAL - Gallon (US)) @ 276R - Greensburg MU",
OWNER_ID: 518,
MASTER_CODE_ID: 11852
},
-{
STOCK_ID: 1884852,
STOCK_NAME: "12234 - TAPE,PAVEMENT,TYPE II,YELLOW,4 INX150FT (LF - Linear Foot) @ 276R - Greensburg MU",
OWNER_ID: 518,
MASTER_CODE_ID: 12202
},
-{
STOCK_ID: 1887306,
STOCK_NAME: "49902 - ON ROAD DIESEL (GAL - Gallon (US)) @ 276R - Greensburg MU",
OWNER_ID: 518,
MASTER_CODE_ID: 12669
},
-{
STOCK_ID: 1887563,
STOCK_NAME: "13001 - TIRE,P195/75R14,LR B,TL,RADIAL (EA - Each) @ 276R - Greensburg MU",
OWNER_ID: 518,
MASTER_CODE_ID: 12880
}
]
```


{baseURL}/authority/{id}/project

This call retrieves projects that belong to particular owner identified by {id}. (If {id} is omitted, all projects of all authority units are returned.) An example of this type is shown below:

http://vega:8080/ams_la/rest/v1/authority/62/project


```
[
-{
PROJECT_ID: 1256,
PROJECT_NAME: "Signals Maintenance",
MODULE_ID: 3,
OWNER_ID: 62,
STATUS: 4,
USER_UPDATE: "SHANKER",
DATE_UPDATE: "20100219114324",
START_DATE: "20100219114105",
PROJECT_TYPE: 0,
CONTRACT_PLAN_TYPE: 0
},
-{
PROJECT_ID: 4494,
PROJECT_NAME: "01 - Surface Maintenance",
MODULE_ID: 3,
OWNER_ID: 62,
STATUS: 4,
USER_UPDATE: "SHANKER",
DATE_UPDATE: "20100804183542",
START_DATE: "20090901084350",
PLAN_BUDGET: 11,
PROJECT_TYPE: 0,
CONTRACT_PLAN_TYPE: 0,
PARENT_PROJECT_ID: 1213
}
]
```


{baseURL}/authority/{oid}/project/{pid}
This call retrieves a project identified by {pid} that belongs to particular owner identified by {oid}. (If {pid} is omitted, all projects for the identified owner are returned. If {oid} is omitted, all projects identified by {pid} are returned.) An example of this type is shown below:

http://vega:8080/ams_la/rest/v1/authority/62/project/1256


```
[
-{
PROJECT_ID: 1256,
PROJECT_NAME: "Signals Maintenance",
MODULE_ID: 3,
OWNER_ID: 62,
STATUS: 4,
USER_UPDATE: "SHANKER",
DATE_UPDATE: "20100219114324",
START_DATE: "20100219114105",
PROJECT_TYPE: 0,
CONTRACT_PLAN_TYPE: 0
}
]
```


{baseURL}/authority/{oid}/project/{pid}/asset/{iid}
This call retrieves the asset identified by {iid} associated with the project identified by {pid} that belongs to particular owner identified by {oid}. (The {iid}, {pid}, and {oid} terms are optional. If the {iid} term is omitted, all assets associated with project {pid} are returned. If {pid} is omitted, either asset (s) for all projects for the identified owner are returned. If {oid} is omitted, all assets for all projects identified are returned.) An example of this type is shown below:

http://vega:8080/ams_la/rest/v1/authority/1317/project/1213/asset/35


```
[
-{
ASSET_TYPE_ID: 35,
ASSET_TYPE_NAME: "Fences",
PARENT_ID: 2,
LEVEL_ID: 3,
INVENTORY_TABLE_NAME: "FENCE_INVENTORY",
NUM_CHILD: 0,
USER_UPDATE: "EUGENE",
DATE_UPDATE: "20070619164044",
ASSET_TYPE_CAT: 2,
ORDER_ID: 13,
PM_APPL: 1,
DEFECT_SURV_APPL: 0,
LOCATION_APPL: 1,
MODULE_ID: 3,
INVENTORY_MEASURE_EXP: "1"
}
]
```


{baseURL}/authority/{oid}/project/{pid}/asset/{iid}/activity/{aid}
This call retrieves the asset identified by {iid} associated with the activity identified by {aid} performed in the project identified by {pid} that belongs to particular owner identified by {oid}. (The {aid}, {iid}, {pid}, and {oid} terms are optional.) An example of this type where all optional terms are included is shown below:

http://vega:8080/ams_la/rest/v1/authority/1317/project/1213/asset/2/activity/260


```
[
-{
ACTIVITY_ID: 260,
ACTIVITY_NAME: "400-05 LEVELING - HAND METHOD (TON - Ton (US))",
PARENT_ID: 255,
ORDER_ID: 6,
LEVEL_ID: 4,
NUM_CHILD: 0,
UNIT_ID: 245
}
]
```


Another example where all optional terms other than asset ID are omitted is shown below:

http://vega:8080/ams_la/rest/v1/authority/project/asset/638/activity/


```
[
-{
ACTIVITY_ID: 408,
ACTIVITY_NAME: "530-00 STANDARD SIGN REPAIR (EA - Each)",
PARENT_ID: 409,
ORDER_ID: 1,
LEVEL_ID: 4,
NUM_CHILD: 0,
UNIT_ID: 249
},
-{
ACTIVITY_ID: 410,
ACTIVITY_NAME: "530-01 STANDARD SIGN REPLACEMENT (EA - Each)",
PARENT_ID: 409,
ORDER_ID: 2,
LEVEL_ID: 4,
NUM_CHILD: 0,
UNIT_ID: 249
},
-{
ACTIVITY_ID: 411,
ACTIVITY_NAME: "530-02 LARGE SIGN REPAIR (EA - Each)",
PARENT_ID: 409,
ORDER_ID: 3,
LEVEL_ID: 4,
NUM_CHILD: 0,
UNIT_ID: 249
},
-{
ACTIVITY_ID: 412,
ACTIVITY_NAME: "530-03 LARGE SIGN REPLACEMENT (EA - Each)",
PARENT_ID: 409,
ORDER_ID: 4,
LEVEL_ID: 4,
NUM_CHILD: 0,
UNIT_ID: 249
},
```


Lookup

Use the Lookup API entity to extract metadata. For example, mapping column IDs to the actual labels that a user wants to see on his or her computer or obtaining error messages by their IDs. This entity allows only GET calls.

The following GET calls may be made to request data from the AgileAssets system via the API:

{baseURL}/lookup/column

This call retrieves all column metadata from the system as an object. Use the projection of a Q object to prevent the amount of returned data from being excessive. An example of this call is shown below:

http://localhost:8080/ams_dev/rest/v1/lookup/column?q={"page":{"size":2,"number":7}}


```
[
-{
COLUMN_ID: "END_DATE",
COLUMN_LABEL: "Finish Date",
VIEW_TYPE: "D",
RIGHT_TO_SEE: 2,
RIGHT_TO_EDIT: 3,
FORMAT_STR: "[shortdate]",
USE_FOR_WEIGHTED_AVERAGE: 0,
FINEST_PARTITION_DT_RULE_ID: 1,
HAS_NUM_CHILD: 0
},
-{
COLUMN_ID: "APPROVED",
COLUMN_LABEL: "Approved",
VIEW_TYPE: "C",
RIGHT_TO_SEE: 2,
RIGHT_TO_EDIT: 4,
USE_FOR_WEIGHTED_AVERAGE: 0,
FINEST_PARTITION_DT_RULE_ID: 1,
HAS_NUM_CHILD: 0
}]
```


{baseURL}/lookup/column/{id}

This call retrieves column metadata for the column identified by {id} as a collection. Use the projection of a Q object to prevent the amount of returned data from being excessive. An example of this call is shown below:

http://localhost:8080/ams_dev/rest/v1/lookup/column/SIGN_ID


```
{
COLUMN_ID: "SIGN_ID",
COLUMN_LABEL: "Sign",
VIEW_TYPE: "T",
TABLE_NAME: "SIGN_INVENTORY",
ITEM_COL_NAME: "SIGN_ID",
DESCR_COL_NAME: "SIGN_NAME",
RIGHT_TO_SEE: 3,
RIGHT_TO_EDIT: 3,
USE_FOR_WEIGHTED_AVERAGE: 0,
HAS_NUM_CHILD: 0,
DATA_TYPE: 7
}
```


{baseURL}/lookup/column/{id}/map

This call retrieves the key-value-pair object that represents all possible values for the particular column identified by {id}. Use the projection of a Q object to prevent the amount of returned data from being excessive. An example of this call is shown below:

http://localhost:8080/ams_dev/rest/v1/lookup/column/ROUTE_ID/map


```
{
30445: "047-KY-0313 -000",
30446: "047-KY-0313 -010",
30447: "047-KY-0313 -111",
30448: "047-KY-0313 -141",
30449: "047-KY-0347 -000",
30450: "047-KY-0391 -000",
30451: "047-KY-0434 -000",
30452: "047-KY-0434 -020",...
```


{baseURL}/lookup/column/{id}/map/{idd}
This call retrieves the key-value-pair object that includes the value {idd} for the particular column identified by {id}. This call is useful for performing quick-search functions such as for type-ahead controls. Use the projection of a Q object to prevent the amount of returned data from being excessive. An example of this call is shown below:

http://localhost:8080/ams_dev/rest/v1/lookup/column/ROUTE_ID/map/0347


```
{
30347: "047-US-0062 -010",
30449: "047-KY-0347 -000"
}
```


{baseURL}/lookup/column/{id}/value
For columns of view type T (List) or H (Huge List), this call retrieves the values to populate the list. Use the projection of a Q object to prevent the amount of returned data from being excessive. An example of this call is shown below:

http://localhost:8080/ams_dev/rest/v1/lookup/column/SIGN_CLASS_CODE_ID/value


```
[
-{
SIGN_CLASS_CODE_ID: 11820,
ITEM_NAME: "Gantry",
IN_USE_: 1
},
-{
SIGN_CLASS_CODE_ID: 11822,
ITEM_NAME: "Post",
IN_USE_: 1
}
]
```


{baseURL}/lookup/column/{id}/barcode/{bc}
For columns of view type T (List) or H (Huge List) whose look-up tables contain bar codes, this call retrieves the values-key/value map that matches the bar code. The search for the specified bar code is performed in both keys and values, with all matches being returned. An example of this call is shown below:

http://localhost:8080/ams_trunk/rest/v1/lookup/column/EQUIPMENT_ID/barcode/4444


```
{
4444: "0KA8647",
14444: "1KW0924"
}
```


Repair Order

Note: The Repair Order API entity will only function if the Fleet Management module is part of the AgileAssets system being queried and only if that module is assigned a module ID of 5. (This is because repair orders are only used in the Fleet Management module.)

Use the Repair Order API entity to work with repair orders in the AgileAssets Fleet Management module. This entity allows GET, POST, PUT, and DELETE calls.

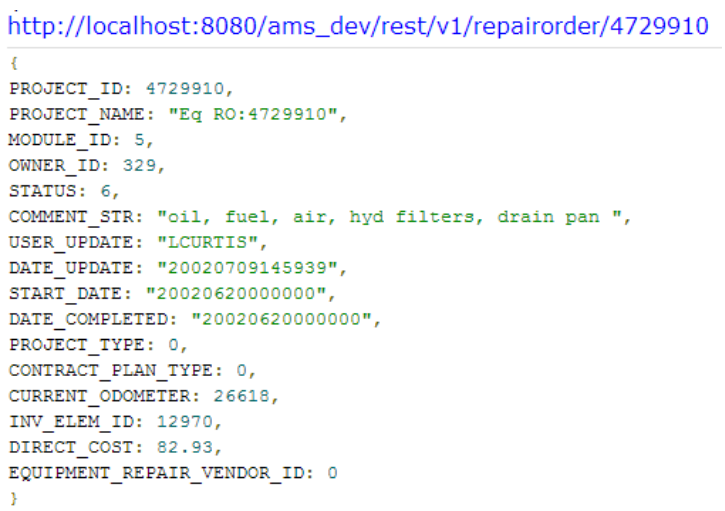
The following GET calls may be made to request repair order data from the AgileAssets system via the API:

{baseURL}/repairOrder

This call retrieves all repair orders in the Fleet Management module. Use the projection of a Q object to prevent the amount of returned data from being excessive. (If input Q does not specify pagination, a default one will be used.) An example of this type is shown below:

{baseURL}/repairOrder/{id}

This call retrieves the repair order identified by {id}. Use the projection of a Q object to prevent the amount of returned data from being excessive. An example of this type is shown below:



{baseURL}/repairOrder/search/{word}

This call retrieves all repair orders that have {word} in a vehicle's name, plate, or VIN. Use the projection of a Q object to prevent the amount of returned data from being excessive. An example of this type is shown below:

http://localhost:8080/ams_trunk/rest/v1/repairorder/search/ra149?q={"page":{"number":1,"size":20},"projection":["PROJECT_ID","PROJECT_NAME","PLATE","EQUIPMENT_NAME","VIN","EQUIPMENT_REPAIR_VENDOR_ID","EQUIPMENT_ID"]}

```
[
-{
PROJECT_ID: null,
PROJECT_NAME: "NEW Repair Order",
PLATE: "1498KW",
EQUIPMENT_NAME: "TRA1498",
VIN: "417608",
EQUIPMENT_REPAIR_VENDOR_ID: null,
EQUIPMENT_ID: 296
},
-{
PROJECT_ID: null,
PROJECT_NAME: "NEW Repair Order",
PLATE: "1497KW",
EQUIPMENT_NAME: "TRA1497",
VIN: "1ZE10N12SMLP14066",
EQUIPMENT_REPAIR_VENDOR_ID: null,
EQUIPMENT_ID: 295
},
-{
PROJECT_ID: null,
PROJECT_NAME: "NEW Repair Order",
PLATE: "1495KW",
EQUIPMENT_NAME: "TRA1495",
VIN: "1WC200G25P1060311",
EQUIPMENT_REPAIR_VENDOR_ID: null,
EQUIPMENT_ID: 294
},…
```


{baseURL}/repairOrder/barcodesearch/{word}
This call retrieves the repair orders that contain the bar code {word}. Use the projection of a Q object to prevent the amount of returned data from being excessive.

{baseURL}/repairOrder/{id}/DirectCost
This call retrieves the direct costs for the repair order identified by {id}. Use the projection of a Q object to prevent the amount of returned data from being excessive. An example of this type is shown below:

http://localhost:8080/ams_dev/rest/v1/repairorder/4729910/DirectCost

```
[
{
RECORD_ID: 636169,
PROJECT_ID: 4729910,
DATE_WORK: "20020620000000",
APPROVED: 0,
MATERIAL_COST: 6.37,
COMMENT_STR: "fuel filter from Napa PC#4729910",
ACTIVITY_ID: 2,
DESCRIBE_STR: "fuel filter",
QUANTITY: 1,
UNIT_COST: 6.37,
DI_NUM: 4
},
-{
RECORD_ID: 636170,
PROJECT_ID: 4729910,
DATE_WORK: "20020620000000",
APPROVED: 0,
MATERIAL_COST: 6.77,
COMMENT_STR: "hyd filter from Napa PC#4729910",
ACTIVITY_ID: 2,
DESCRIBE_STR: "hyd filter",
QUANTITY: 1,
UNIT_COST: 6.77,
DI_NUM: 5
},
-{
RECORD_ID: 636171,
PROJECT_ID: 4729910,
DATE_WORK: "20020620000000",
APPROVED: 0,
MATERIAL_COST: 29.08,
COMMENT_STR: "drain pan from Napa PC#4729910",
ACTIVITY_ID: 2,
DESCRIBE_STR: "drain pan",
QUANTITY: 1,
UNIT_COST: 29.08,
DI_NUM: 6
}
]
```


{baseURL}/repairOrder/{id}/DirectCost/Sum
This call retrieves the sum of labor and equipment costs for all direct costs associated with the repair order identified by {id}. Use the projection of a Q object to prevent the amount of returned data from being excessive. An example of this type is shown below:

http://192.168.242.174:8080/ams_trunk/rest/v1/repairorder/6501161/DirectCost/Sum

```
{
sum: 62.36
}
```


{baseURL}/repairOrder/DirectCost/{id}
This call retrieves the direct cost identified by particular direct cost record ID number {id} as a JSON object. Use the projection of a Q object to prevent the amount of returned data from being excessive. An example of this type is shown below:

http://192.168.242.174:8080/ams_trunk/rest/v1/repairorder/directcost/1541354


```
{
RECORD_ID: 1541354,
PROJECT_ID: 6501161,
EQUIPMENT_REPAIR_VENDOR_ID: null,
DATE_WORK: "20060928090647",
APPROVED: 0,
LABOR_COST: null,
MATERIAL_COST: 28.27,
COMMENT_STR: null,
COMMENT_ID: null,
USER_UPDATE: null,
DATE_UPDATE: null,
ACTIVITY_ID: 2,
ADJ_COST: null,
DESCRIBE_STR: "AIR FILTER ",
QUANTITY: 1,
UNIT_COST: 28.27,
DI_NUM: 2
}
```


{baseURL}/repairOrder/{id}/vendor
This call retrieves the vendor information for the repair order identified by {id}. Use the projection of a Q object to prevent the amount of returned data from being excessive.

{baseURL}/repairOrder/vendor
This call retrieves all vendors in the Fleet Management module. Use the projection of a Q object to prevent the amount of returned data from being excessive.

{baseURL}/repairOrder/vendor/map
This call retrieves all vendors in the Fleet Management module where each element is a JSON object key-value-pair or vendor ID to vendor name. Use the projection of a Q object to prevent the amount of returned data from being excessive. An example of this type is shown below:

http://192.168.242.174:8080/ams_trunk/rest/v1/repairorder/vendor/map


```
{
283: "PURCHASE FORD LINCOLN MERCURY",
288: "WAL MART",
290: "FRAINES BODY SHOP",
292: "JEFF SACHS CHRYSLER",
293: "B - (DO NOT USE)",
294: "MORGAN TIRE CENTER INC",
295: "MUFFLER EXPRESS INC",
297: "BOB ALLEN MOTOR MALL-(DO NOT USE)",
516: "LAGRANGE SERVICE CTR"
}
```


{baseURL}/repairOrder/vendor/map/{id}
This call retrieves all vendors in the Fleet Management module where each element is a JSON object key-value-pair or vendor ID to vendor name and where the vendor name contains the string given in {id}. Use the projection of a Q object to prevent the amount of returned data from being excessive. An example of this type is shown below:

http://192.168.242.174:8080/ams_trunk/rest/v1/repairorder/vendor/map/zo


```
{
51: "LORENZO JOHNSON BODY SHOP",
985: "HORIZON TRANSMISSION",
4328: "AUTO ZONE 2442",
4718: "RAZORS EDGE BODY SHOP",
7930: "PENNZOIL 10 MINUTE LUBE #9 -RHL ENTERPRISES"
}
```


{baseURL}/repairOrder/{id}/activity
This call retrieves all activities performed in the repair order identified by {id}. At most, two fields are returned: ID and name of activity. Use the projection of a Q object to prevent the amount of returned data from being excessive. An example of this type is shown below:

http://192.168.242.195:8080/ams_dev/rest/v1/repairorder/5559182/activity


```
[
-{
ACTIVITY_ID: 87833,
ACTIVITY_NAME: "102 Brakes"
},
-{
ACTIVITY_ID: 87845,
ACTIVITY_NAME: "114 Suspension/Frame"
},
-{
ACTIVITY_ID: 87979,
ACTIVITY_NAME: "128 Oil Change"
}
]
```


{baseURL}/repairOrder/{id}/warranty
This call retrieves all active warranties for the repair order identified by {id}. Use the projection of a Q object to prevent the amount of returned data from being excessive. An example of this type is shown below:

http://192.168.242.237:8080/ams_dev/rest/v1/repairorder/6596957/Warranty


```
[
-{
EQUIPMENT_WARRANTY_ID: 196,
EQUIPMENT_WARRANTY_NAME: "Transaxle",
EQUIPMENT_ID: 635,
EFF_DATE: "20110527000000",
START_ODOMETER: 50000,
WARRANTY_TRIGGER_ID: 1,
WARRANTY_TRIGGER_ARG: "100000",
USER_UPDATE: "JG",
DATE_UPDATE: "20110526153228"
},
-{
EQUIPMENT_WARRANTY_ID: 195,
EQUIPMENT_WARRANTY_NAME: "Oil Filter",
EQUIPMENT_ID: 635,
EFF_DATE: "20110526000000",
START_ODOMETER: 55000,
WARRANTY_TRIGGER_ID: 1,
WARRANTY_TRIGGER_ARG: "2000",
USER_UPDATE: "JG",
DATE_UPDATE: "20110526144918"
}
]
```


When a new item is created, the correct ID is returned to the caller. Hence, providing the inventory element ID via a POST call is useless because POST is assumed to only create data.
The following POST calls may be made to the Fleet Management module via the API:

{baseURL}/repairOrder
This call creates a new repair order. The inputs should include:

- Header: Content-Type: application/json

- Data: JSON object that represents a new repair order. The system automatically sets MODULE_ID to be 5 and the STATUS to be 4. The PROJECT_ID is assigned as the next one in sequence. The PROJECT_NAME is required, along with any other required fields.

The output is a JSON array of new PROJECT_ID values. It also automatically creates inventory PM activities for today.
An example of this type is shown below:

```
curl -i -X POST -H "Content-Type: application/json" http://192.168.242.221:8080/ams_trunk/rest/v1/repairorder/  -d
'{"PROJECT_NAME":"RO 318", "INV_ELEM_ID" : 318,"OWNER_ID":1949}'
```

```
HTTP/1.1 200 OK
Server: Apache-Coyote/1.1
Content-Type: application/json;charset=utf-8
Content-Length: 10
Date: Thu, 13 Feb 2014 21:35:58 GMT

[6947650]
```

{baseURL}/repairOrder/{id}/warranty
This call creates warranty information for the already-existing repair order {id}. The inputs should include:

- Header: Content-Type: application/json
- Data: JSON object that represents a new warranty record. It must contain EQUIPMENT_WARRANTY_NAME; EQUIPMENT_ID; WARRANTY_LENGTH_DAY or/and WARRANTY_LENGTH_DAY; EFF_DATE; and START_ODOMETER.

The output is a JSON array of new WARRANTY_ID values.

{baseURL}/repairOrder/{id}/directcost
This call creates a direct cost record(s) for the already-existing repair order {id}. The inputs should include:

- Header: Content-Type: application/json
- Data: JSON object or JSON array of objects that represents the direct costs. The minimum set of fields is the required fields (with the exception of the ID fields automatically created by the system).

The output is a JSON array of new RECORD_ID values for newly created direct costs.

The following PUT calls may be made to modify records in the Fleet Management module via the API:

{baseURL}/repairOrder/{id}
The call updates the particular repair order that is identified by {id}. The input should include:

- Header: Content-Type: application/json
- Data: JSON object that represent record fields that are to be updated. Only those fields in the input JSON object should be updated in database.

The output is an HTTP status code unless there is an error.

{baseURL}/repairOrder/{id}/directcost
The call updates the direct costs associated with the particular repair order that is identified by {id}. The input should include:

- Header: Content-Type: application/json
- Data: JSON object or array of objects that represent record fields that are to be updated. Only those fields in the input JSON object should be updated in database. It must contain the RECORD_ID field and it cannot be null.

The output is an HTTP status code unless there is an error.

The following DELETE calls may be made to delete data in the Fleet Management module via the API:

{baseURL}/repairOrder/{id}
The call deletes the repair order that is identified by {id}. The operation will fail if the repair order is approved or if any of the day cards associated with the repair order are approved.
The output is an HTTP status code unless there is an error.

{baseURL}/repairOrder/directcost/{id}
The call deletes the direct cost records associated with the repair order that is identified by {id}. The operation will fail if the records are approved.
The output is an HTTP status code unless there is an error.

{baseURL}/repairOrder/{id}/activity/{idd}
The call deletes the activity records identified by {idd} associated with repair order {id}. The operation will fail if the records are approved.
The output is an HTTP status code unless there is an error.

User

Use the User API entity to retrieve user information from an AgileAssets system. User IDs are stored in upper case in the AgileAssets database, but the URL is not case sensitive and therefore the User ID may be written in any case. This entity allows only GET calls.

The following GET calls may be made to request user information from the AgileAssets system via the API:

{baseUrl}/rest/v1/user/{id}

This call retrieves the user record for the User ID identified by {id}. (The USER_PASSWORD field is not returned.) Use the projection of a Q object to limit the number of returned columns. An example of this call is shown below:

http://vega:8080/ams_la/rest/v1/authority

{baseUrl}/rest/v1/user/{id}

This call retrieves the user record for the User ID identified by {id}. (The USER_PASSWORD field is not returned.) Use the projection of a Q object to limit the number of returned columns. An example of this call is shown below:

http://localhost:8080/ams_dev/rest/v1/user/LEONID


```
{
USER_ID: "LEONID",
ALLOW_LOGIN: 0,
PASSWORD_START: "20120120100720",
IS_ADMIN: 1,
UNSUCCESS_NUM: 2,
SECURITY_QUESTIONS_ANSWERED: 0,
PASSWORD_EXPIRED: 0,
AD_USER_ID: "LEONID",
USER_UPDATE: "LEONID",
DATE_UPDATE: "20120120100720"
}
```


{baseUrl}/rest/v1/user

This call retrieves all user records. (The USER_PASSWORD fields are not returned.) Use the projection of a Q object to limit the number of returned columns. An example of this call is shown below:

http://localhost:8080/ams_dev/rest/v1/user?q={"filter":[["IS_ADMIN", "!=", 1]],"projection":[USER_ID,IS_ADMIN,ALLOW_LOGIN]}


```
[
-{
USER_ID: "AARONC",
IS_ADMIN: 0,
ALLOW_LOGIN: 1
},
-{
USER_ID: "AARONT",
IS_ADMIN: 0,
ALLOW_LOGIN: 1
},
-{
USER_ID: "AARONVO",
IS_ADMIN: 0,
ALLOW_LOGIN: 1
},
-{
USER_ID: "ABHISHEK",
IS_ADMIN: 0,
ALLOW_LOGIN: 1
},...
```


{baseUrl}/rest/v1/user/{id}/authority

This call returns the mapping between OWNER_ID and OWNER_NAME for a user identified by {id}. An example of this call is shown below:

http://localhost:8080/ams_dev/rest/v1/user/alex/authority/


```
[
-{
OWNER_ID: 10,
OWNER_NAME: "District 11"
},
-{
OWNER_ID: 92,
OWNER_NAME: "GRAVES"
},
-{
OWNER_ID: 113,
OWNER_NAME: "LAUREL"
}
]
```


{baseUrl}/rest/v1/user/{id}/authority/id
This call returns the authority IDs to which the user identified by {id} is assigned. An example of this call is shown below:

http://localhost:8080/ams_dev/rest/v1/user/leonid/authority/id


```
[
2,
3,
4,
5,
7,
8,
61,
62,
1317
]
```


{baseUrl}/rest/v1/user/{id}/authority/name
This call returns the authority names to which the user identified by {id} is assigned. An example of this call is shown below:

http://localhost:8080/ams_dev/rest/v1/user/leonid/authority/name


```
[
"D02/G500-RDSIDE DEVEL COORD/BRIDGE CITY",
"District 02",
"District 03",
"District 04",
"District 05",
"District 07",
"District 08",
"District 61",
"District 62"
]
```


Work Order

Use the Work Order API entity to work with work orders in the AgileAssets system. This entity allows GET, POST, PUT, and DELETE calls.

The following GET calls may be made to request work order data from the AgileAssets system via the API:
{baseUrl}/workOrder/<workOrderId>/DayCard/<DayCardType>/<Year>/<Month>/<Day>
This call retrieves work orders, with the parameters determining which work orders are returned. (Q objects may also be used to determine what is returned.) The following parameters are available in this call:

- workOrderId – This parameter is optional. When given, it is the ID of the work order whose data is to be returned.
- DayCardType – This parameter is optional. When given, it determines what type(s) of day cards will be returned rather than all day cards. The valid values are: labor, equipment, cost, location, material, contract, and accomplishment. More than one value may be specified.
- Year – This parameter is optional. When given it is the four-digit year of the start date of the work order(s).
- Month – This parameter is optional. When given it is the two-digit month of the start date of the work order(s).
- Day – This parameter is optional. When given it is the two-digit day of the start date of the work order(s).

Note: The data is retrieved from the Day Cards table. If the Day Cards table does not exist, an error is **not** returned.

The response is a JSON array of JSONObjects. Each object is always a single pair of {key:value}, where key is the field name and value is the field value of this record.
Following are several examples of GET calls for this type:
o To retrieve data for work order 1273:
http://localhost:8080/ams_dev/rest/workOrder/1273
o To retrieve all day cards for work order 1273 that were created in 2010:
http://localhost:8080/ams_dev/rest/v1/workOrder/1273/DayCard/2010
o To retrieve all day cards created in 2010:
http://localhost:8080/ams_dev/rest/v1/workOrder/daycard/2010
o To retrieve all day cards created in September 2010:
http://localhost:8080/ams_dev/rest/workOrder/daycard/2010/09
o To retrieve all equipment day cards:
http://localhost:8080/ams_trunk/rest/v1/workorder/daycard/equipment
o To retrieve equipment day cards created on September 16, 2010:
http://localhost:8080/ams_dev/rest/workOrder/DayCard/equipment/2010/09/16
o To retrieve all day cards in the system:
http://localhost:8080/ams_dev/rest/workOrder/DayCard/
o To retrieve all contract day cards in the system:
http://localhost:8080/ams_dev/rest/workOrder/DayCard/Contract
o To retrieve all approved labor day cards in the system:
http://localhost:8080/ams_dev/rest/workOrder/DayCard/labor?q=\{filter:[[APPROVED,'=',1]]}
o To retrieve all approved cost day cards in the system created in 2010:
http://localhost:8080/ams_dev/rest/workOrder/DayCard/cost/2010?q=\{filter:[[APPROVED,'=',1]]}
{baseUrl}/workOrder
This call retrieves a collection of work orders. Rather than using parameters as in the previous call, you only use a Q object to determine what data is returned. Several examples of using Q objects are provided below.
To retrieve the fourth page of a set of work orders where two records appear on each page:

http://vega:8080/ams_la/rest/v1/workorder?q={"page":{"number":4,"size":2}}


```
[
-{
WORK_ORDER_ID: 1264,
PROJECT_ID: 1219,
ASSET_TYPE_ID: 638,
OWNER_ID: 1317,
CREW_SUB_ID: 943,
ACTIVITY_ID: 408,
START_DATE: "20100827000000",
END_DATE: "20100827000000",
DURATION: 1,
PLAN_AMOUNT: 1,
AMOUNT: 1,
STATUS: 4,
COMMENT_STR: "WOrk Date

Upload ID:966
Location Description:US 90
Location Error:'US 90' is a nonexisting route",
USER_UPDATE: "BRANDON",
DATE_UPDATE: "20130402125802",
START_HOUR: 0,
END_HOUR: 1,
CALENDAR_ID: 1,
BUDGET_CAT_ID: 167,
WORK_ORDER_SOURCE_ID: 7,
ORACLEROWNUM: 7
},
-{
WORK_ORDER_ID: 1265,
PROJECT_ID: 1219,
ASSET_TYPE_ID: 638,
OWNER_ID: 1317,
CREW_SUB_ID: 943,
ACTIVITY_ID: 408,
DURATION: 0,
PLAN_AMOUNT: 0,
AMOUNT: 0,
STATUS: 4,
COMMENT_STR: "NEW ONE",
USER_UPDATE: "ALEX",
DATE_UPDATE: "20100827110022",
START_HOUR: 0,
END_HOUR: 0,
CALENDAR_ID: 1,
BUDGET_CAT_ID: 167,
WORK_ORDER_SOURCE_ID: 7,
ORACLEROWNUM: 8
}
]
```


To retrieve the seventh page of a set of work orders with four records on each page:

http://vega:8080/ams_la/rest/v1/workOrder?q={"projection":["WORK_ORDER_ID","ASSET_TYPE_ID", "ACTIVITY_ID"],"sort":{"ASSET_TYPE_ID":"DESC","ACTIVITY_ID":"DESC"},"page":{"size":4,"number":7}}


```
[
-{
WORK_ORDER_ID: 1470,
ASSET_TYPE_ID: 1013,
ACTIVITY_ID: 349
},
-{
WORK_ORDER_ID: 1062,
ASSET_TYPE_ID: 1013,
ACTIVITY_ID: 344
},
-{
WORK_ORDER_ID: 1012,
ASSET_TYPE_ID: 1013,
ACTIVITY_ID: 343
},
-{
WORK_ORDER_ID: 1009,
ASSET_TYPE_ID: 1013,
ACTIVITY_ID: 342
}
]
```


To retrieve two fields of each open work order and sort by START_DATE column:

http://vega:8080/ams_la/rest/v1/workOrder?q={"filter":[["STATUS","=",4]],"projection":["WORK_ORDER_ID","STATUS"],"sort":{"START_DATE":"DESC"},"page":{"size":4,"number":3}}


```
[
-{
WORK_ORDER_ID: 1247,
STATUS: 4
},
-{
WORK_ORDER_ID: 1246,
STATUS: 4
},
-{
WORK_ORDER_ID: 1241,
STATUS: 4
},
-{
WORK_ORDER_ID: 887,
STATUS: 4
}
]
```


{baseUrl}/workOrder/project/{id}
This call retrieves all work orders for the project identified as {id}. An example of this call is shown below:

http://localhost:8080/ams_dev/rest/v1/workorder/project/1213


```
[
-{
WORK_ORDER_ID: 1273,
PROJECT_ID: 1213,
ASSET_TYPE_ID: 2,
OWNER_ID: 1317,
ACTIVITY_ID: 253,
START_DATE: "20100914000000",
END_DATE: "20100914000000",
DURATION: 0.5,
PLAN_AMOUNT: 0,
STATUS: 4,
USER_UPDATE: "BRANDON",
DATE_UPDATE: "20130322150708",
START_HOUR: 0,
END_HOUR: 0.5,
CALENDAR_ID: 1,
BUDGET_CAT_ID: 2,
WORK_ORDER_SOURCE_ID: 1,
WO_PRIORITY_ID: 1
},
-{
WORK_ORDER_ID: 1257,
PROJECT_ID: 1213,
ASSET_TYPE_ID: 638,
OWNER_ID: 1317,
ACTIVITY_ID: 408,
START_DATE: "20100827000000",
END_DATE: "20100827000000",
DURATION: 8,
PLAN_AMOUNT: 9,
STATUS: 4,
USER_UPDATE: "ALEX",
DATE_UPDATE: "20100827100123",
START_HOUR: 0,
END_HOUR: 8,
CALENDAR_ID: 1,
BUDGET_CAT_ID: 167,
WORK_ORDER_SOURCE_ID: 1,
WO_PRIORITY_ID: 1
}...
```


POST calls create work orders and/or day cards. Creating a work order via a POST call creates a new work order and assigns it a new ID, which is returned to the client as a JSON array. Creating a day card via a POST call inserts a day card for a given work order ID. The ID is provided in the URL string or, when provided in a JSON array, in the request object. The advantage of creating a day card via a POST call rather than a PUT call is that a POST call always inserts a day card and so if a day card already exists, the operation would fail.
The following POST calls may be made to the AgileAssets system via the API:

{baseUrl}/workOrder

This call creates a new work order. The inputs should include:

- Header: Content-Type: application/json
- Data: JSON object that represents a single new work order or a JSON array of multiple new work orders. The minimum set of fields is the required fields, except for the ID fields that the system automatically sets. Depending on the client, the usual set of required values are project, asset, activity, and sub-activity.

Note: Other constraints may exist on the server, such as triggers, that must be satisfied to successfully create a record. This is client-specific functionality.

An example of this type is shown below:

curl -i -X POST -H "Content-Type: application/json" http://localhost:8080/ams_dev/rest/v1/workOrder -d '{"PROJECT_ID":1213,"ASSET_TYPE_ID":638,"ACTIVITY_ID":408}'

{baseUrl}/workOrder/{id}/DayCard

This call creates a new day card for the already existing work order identified by {id}. The inputs should include:

- Header: Content-Type: application/json
- Data: JSON array that represents the collection of JSON objects, one object per day card. Each of those objects is identified by key, day card table name and each of those objects value is a JSON array that represents the collection of day card records. The minimum set of fields is the required fields with the exception of the ID fields that are assigned by the system. Depending on the client, the usual set of required values is project, asset, activity, and sub-activity.

For example, to insert a labor day card with one employee for work order 1643, use the following POST call:
curl -i -X POST -H 'Content-Type: application/json' http://localhost:8080/ams_dev/rest/v1/workOrder/1643/DayCard/ -d '{"WORK_ORDERS_LABOR_DC":[{"LABOR_ID":2412555,"DATE_WORK":"20130314","TOTAL_HOURS":0.5,"TRC_ID":1}]}'

Note: Errors are file-driven.

The aim of PUT calls is to update already existing work orders and day cards. Updating a work order via a PUT call modifies an existing work order. Updating a day card via a PUT call modifies an existing day card for a given work order ID. The ID is provided in the URL string or, when provided in a JSON array, in the request object.
The following PUT call may be made to the AgileAssets system via the API:

{baseURL}/rest/workOrder/{workOrderId}/DayCard/
This call updates an existing work order and day card.
Two examples of this PUT call are shown below:

- To set the owner to 1317 for work order 1649:
  curl -i -X PUT -H "Content-Type: application/json" http://localhost:8080/ams_dev/rest/v1/workOrder/1649 -d '{"OWNER_ID":1317}'
  The output from this call is: html status code 204 - No Content.
- To set the work hours to 0.1 on March 14, 2103 for employee 2412555 for work order 10997795:
  curl -i -X PUT -H "Content-Type: application/json" http://vega.dellanahome.com:8080/ams_in_07_00/rest/v1/workOrder/10997795/DayCard/Location -d '"LABOR_ID":2412555,"DATE_WORK":"20130314","TRC_ID":1,"TOTAL_HOURS":0.1,"WAC_CODE_ID":null}'
  The output from this call is: html status code 204 - No Content.

Note: The JSON Object must include all columns that are part of the Primary Key. Also, since the WORK_ORDER_ID is given via the request URL, if the JSON Object also included WORK_ORDER_ID, it would be ignored.

Use the following DELETE calls to delete data in the AgileAssets system via the API:
{baseUrl}/workOrder/{id}
The call deletes the work order that is identified by {id}. Only one work order may be specified. The operation will fail if the work order is approved or if any of the day cards associated with the work order are approved.
For example, to delete work order 1620, use the following DELETE call:
curl -i -X DELETE http://localhost:8080/ams_dev/rest/workOrder/1620

{baseUrl}/workOrder/{id}/DayCard/{type}/{year}/{month}/{day}
The call deletes the day card records associated with the work order that is identified by {id}. The {id} parameter is required, but the other parameters are optional. (When provided, they specify the type of day card to be deleted and/or the date when the day cards were created.) The operation will fail if the records are approved.
See two examples of DELETE calls below:

- To delete all day cards for work order 1620:
  curl -i -X DELETE http://localhost:8080/ams_dev/rest/workOrder/1620/DayCard/
- To delete all labor day cards for work order 1620 created in 2010:
  curl -i -X DELETE http://localhost:8080/ams_dev/rest/workOrder/1620/DayCard/labor/2010