Virtual-Vehicles REST API Documentation

General Notes

Client Registration

All resource URIs require an Authorization request header with a JSON Web Tokens (JWT)
To obtain a JWT, you need to register as an API client, using the Authentication service
You must supply your application's name and a secret as part of the HTTP request
In return, you will receive your client-specific API key

Obtaining JWTs

Use your API key and secret to obtain a JSON Web Tokens (JWT) using the Authentication service
Include the JWT in an Authorization request header, for all resources
Use standard format of: Header: "Authorization", Value: "Bearer {jwt}"
JWTs have an expiration date and time, set by the Authentication service (1 day default)
If JWT fails authentication, STATUS is returned as '401 Unauthorized'

API Key and Secret

Your API key and secret should be kept **confidential** and never exposed to your end-users
The apiKey contains (24) alphanumeric characters, mixed case (defined by API)
Your secret should contain a minimum of (15) alphanumeric characters, mixed case

Date and Time Format

All date and time values should use the ISO 8601 standard, which also matches MongoDB
All date and time values should be in UTC
Format: YYYY-MM-DDThh:mm:ss.sZ
For example, "serviceDateTime": "2015-05-07T08:22:03.400Z"

Media Types

Media type is optionally indicated by using'.{format}' in all RESTful resource URIs
Media types available include '.json' or' .xml'
JSON is default if not implicitly included in resource URI

Use of PUT

When using PUT to update data, the "updatedAt" and "id" not required in the raw JSON-format HTTP request body
The "updatedAt" will be handled by MongoDB
The "id" is used as part of the resource URI

PATCH is Unsupported

RestExpress has no built in support for PATCH, and hacking a POST as a partial update also does not work. Must use PUT.

Bug with Find using Query Filter

There seems to be a bug using the find operation with int based property. Strings work fine.
For example, GET 'http://localhost:8585/valets/utils/find?filter=parkingSpot::24'
Give error:

May 10, 2015 6:34:06 PM org.mongodb.morphia.query.QueryValidator validateQuery
WARNING: Validation warnings:
[Validation failed: 'When type is one of the integer types the value should be an Integer.
Type was int and value '24' was a class java.util.regex.Pattern']

Virtual-Vehicles REST API Resources		
Microservice	**Purpose (Business Capability)**	**Functions**
Authentication Service	Manage API clients and JWT authentication	• Create a new API client (public) • Read, filter, sort, count, paginate API clients (admin) • Read a single API client (public) • Update an existing API client (public) • Delete an existing API client (admin) • Create new JWT (public) • Validate a JWT (admin/internal) • Service health ping (admin)
Vehicle Service	Manage virtual vehicles	• Create a new vehicle (public) • Read, filter, sort, count, paginate vehicles (admin) • Read a single vehicle (public) • Update an existing vehicle (public) • Delete an existing vehicle (admin) • Service health ping (admin)
Maintenance Service	Manage maintenance on vehicles	• Create a new maintenance record (public) • Read, filter, sort, count, paginate maintenance records (admin) • Read a single maintenance record (public) • Update an existing maintenance record (public) • Delete an existing maintenance record (admin) • Service health ping (admin)
Valet Parking Service	Manage a valet service to park for vehicles	• Create a new valet parking transaction (public) • Read, filter, sort, count, paginate valet parking transactions (admin) • Read a single valet parking transaction (public) • Update an existing valet parking transaction (public) • Delete an existing valet parking transaction (admin) • Service health ping (admin)

Microservice	**Purpose (Business Capability)**	**Functions**
Sales Service	*Manage the buying and selling of vehicles*	*Not implemented...*
Registration Service	*Manage the registration of vehicles to owners*	*Not implemented...*
Auction Service	*Manage a virtual car auction*	*Not implemented...*
Car Show Service	*Manage a virtual car show*	*Not implemented...*
Interaction Service	*Manage the interaction of users with vehicles (ie. drive)*	*Not implemented...*

Authentication Service	
Client Management	
Function	Create a new API client
Method	POST
URI	/clients.{format}

Request Body (raw)	{
	"application": "Virtual Car Collector",
	"secret": "pbZCmrFSBqkYtMh"
	}

Status	201 Created

Response Body	New client object
	
	{
	"_links": {
	"self": {
	"href": "http://virtual-vehicles.com:8587/clients/55482c1bc830337da8c6bd81"
	},
	"up": {
	"href": "http://virtual-vehicles.com:8587/clients"
	}
	},
	"application": "Virtual Car Collector",
	"secret": "pbZCmrFSBqkYtMh",
	"apiKey": "PT2IOIdZG6hSEDddushzSN1G",
	"id": "55482c1bc830337da8c6bd81",
	"createdAt": "2015-05-05T02:34:03.400Z",
	"updatedAt": "2015-05-05T02:34:03.400Z"
	}

MongoDB Document View (BSON)	{
	"_id" : ObjectId("55482c1bc830337da8c6bd81"),
	"className" : "com.example.authentication.objectid.Client",
	"application" : "Virtual Car Collector",
	"secret" : "pbZCmrFSBqkYtMh",
	"apiKey" : "PT2IOIdZG6hSEDddushzSN1G",
	"createdAt" : ISODate("2015-05-05T02:34:03.400Z"),
	"updatedAt" : ISODate("2015-05-05T02:34:03.400Z")
	}

Function	Read, filter, sort, count, paginate API clients filter = org.restexpress.common.query QueryFilter instance (uses contains method, no wildcard option) sort = org.restexpress.common.query QueryOrder instance (use - for descending order) limit and offset = org.restexpress.common.query QueryRange instance (offset is optional - default is 0) countOnly=true returns document count with empty clients object in response body All query params may be combined RestExpress bug/limitation with using non-strings for filtering! Only appears to works for string matching patterns… ([Validation failed: 'Patterns can only be used as query values for Strings']
Method	GET
URI	/clients.{format} /clients?filter={key::value|key::value|...}.{format} /clients?filter=apiKey::PT2IOIdZG6hSEDddushzSN1G /clients?filter=application::Garage|secret::9DH /clients?sort={key|-key|...}&limit={int}&offset={int} /clients?sort=application&limit=20 /clients?sort=application|-createdAt&limit=10&offset=20 /clients?countOnly=true /clients?countOnly=true&filter=apiKey::PT2IOIdZG6hSEDddushzSN1G
Request	(5) optional URL params above: filter, sort, limit, offset, countOnly
Status	200 OK

Response Body	Potentially one or more API client objects
	
	{
	"_links": {
	"self": {
	"href": "http://virtual-vehicles.com:8587/clients/55482c1bc830337da8c6bd81"
	},
	"up": {
	"href": "http://virtual-vehicles.com:8587/clients"
	}
	},
	"application": "Virtual Car Collector",
	"secret": "pbZCmrFSBqkYtMh",
	"apiKey": "PT2IOIdZG6hSEDddushzSN1G",
	"id": "55482c1bc830337da8c6bd81",
	"createdAt": "2015-05-05T02:34:03.400Z",
	"updatedAt": "2015-05-05T02:34:03.400Z"
	},
	...

Function	Read a single API client
Method	GET
URI	/clients/{id}.{format} /clients/55482c1bc830337da8c6bd81
Request	none
Status	200 OK

Response Body	Single client object
	
	{
	"_links": {
	"self": {
	"href": "http://virtual-vehicles.com:8587/clients/55482c1bc830337da8c6bd81"
	},
	"up": {
	"href": "http://virtual-vehicles.com:8587/clients"
	}
	},
	"application": "Virtual Car Collector",
	"secret": "pbZCmrFSBqkYtMh",
	"apiKey": "PT2IOIdZG6hSEDddushzSN1G",
	"id": "55482c1bc830337da8c6bd81",
	"createdAt": "2015-05-05T02:34:03.400Z",
	"updatedAt": "2015-05-05T02:34:03.400Z"
	}

Function	Update an existing API client (commonly used to change client secret or apiKey)
Method	PUT
URI	/clients/{id}.{format} /clients/55482c1bc830337da8c6bd81

Request Body (raw)	{
	"application": "Virtual Car Collector",
	"secret": "TYRmHF7nNq2fum8",
	"apiKey": "ABKJCeVDzU9L5MGNPU6pQsZy",
	"createdAt": "2015-05-05T02:34:03.400Z"
	}
	
	"updatedAt" and "id" not required

Status	201 Created

Response Body	{
	"_links": {
	"self": {
	"href": "http://virtual-vehicles.com:8587/clients/55482c1bc830337da8c6bd81"
	},
	"up": {
	"href": "http://virtual-vehicles.com:8587/clients"
	}
	},
	"application": "Virtual Car Collector",
	"secret": "TYRmHF7nNq2fum8",
	"apiKey": "ABKJCeVDzU9L5MGNPU6pQsZy",
	"id": "55482c1bc830337da8c6bd81",
	"createdAt": "2015-05-05T02:34:03.400Z",
	"updatedAt": "2015-05-05T02:34:03.400Z"
	}

Function	Delete an existing API client
Method	DELETE
URI	/clients/{id}.{format} /clients/55482c1bc830337da8c6bd81
Request	none
Status	204 No Content
Response Body	none

JWT Management	
Function	Create a new JWT
Method	GET
URI	/jwts.{format}?apiKey={apiKey}&secret={secret} /jwts?apiKey=PT2IOIdZG6hSEDddushzSN1G&secret=pbZCmrFSBqkYtMh
Request	(2) URL params above
Status	200 OK
Response Body	New jwt as String eyJ0...NiJ9.eyJp...0fQ.ZTzC...8RdY (jwt abridged for length)

Function	Validate a JWT
Method	GET
URI	/jwts/{jwt}.{format} /jwts/eyJ0...NiJ9.eyJp...0fQ.ZTzC...8RdY (jwt abridged for length)
Request	(1) URL param above
Status	200 OK
Response Body	Boolean: true or false

Function	Service health ping
Method	GET
URI	/clients/utils/ping.{format}
Request	none
Status	200 OK
Response Body	Boolean: true or false

TOP

Vehicle Service	
Function	Create a new vehicle
Method	POST
URI	/vehicles.{format}
Request Header	Header: Authorization Value: Bearer {jwt}

Request Body (raw)	{
	"year": 2015,
	"make": "Chevrolet",
	"model": "Corvette Stingray",
	"color": "Blue",
	"type": "Coupe",
	"mileage": 200
	}

Status	201 Created

Response Body	New vehicle object
	{
	"_links": {
	"self": {
	"href": "http://virtual-vehicles.com:8581/vehicles/554e8bd4c830093007d8b949"
	},
	"up": {
	"href": "http://virtual-vehicles.com:8581/vehicles"
	}
	},
	"year": 2015,
	"make": "Chevrolet",
	"model": "Corvette Stingray",
	"color": "Blue",
	"type": "Coupe",
	"mileage": 200,
	"id": "554e8bd4c830093007d8b949",
	"createdAt": "2015-05-09T22:36:04.808Z",
	"updatedAt": "2015-05-09T22:36:04.808Z"
	}

MongoDB Document View (BSON)	{
	"_id" : ObjectId("554e8bd4c830093007d8b949"),
	"className" : "com.example.vehicle.objectid.Vehicle",
	"year" : 2015,
	"make" : "Chevrolet",
	"model" : "Corvette Stingray",
	"color" : "Blue",
	"type" : "Coupe",
	"mileage" : 200,
	"createdAt" : ISODate("2015-05-09T22:36:04.808Z"),
	"updatedAt" : ISODate("2015-05-09T22:36:04.808Z")
	}

Function	Read, filter, sort, count, paginate vehicles filter = org.restexpress.common.query QueryFilter instance (uses contains method, no wildcard option) sort = org.restexpress.common.query QueryOrder instance (use - for descending order) limit and offset = org.restexpress.common.query QueryRange instance (offset is optional - default is 0) countOnly=true returns document count with empty vehicles object in response body All query params may be combined
Method	GET
URI	/vehicles.{format} /vehicles?filter={key::value|key::value|...}.{format} /vehicles?filter=make::Chevrolet /vehicles?filter=type::Coupe|color::Blue /vehicles?sort={key|-key|...}&limit={int}&offset={int} /vehicles?sort=make&limit=20 /vehicles?sort=make|-model&limit=10&offset=20 /vehicles?countOnly=true /vehicles?countOnly=true&filter=make::Chevrolet
Request Header	Header: Authorization Value: Bearer {jwt}
Request	(5) optional URL params above: filter, sort, limit, offset, countOnly
Status	200 OK

Response Body	Potentially one or more vehicle objects
	
	{
	"_links": {
	"self": {
	"href": "http://virtual-vehicles.com:8581/vehicles"
	}
	},
	"_embedded": {
	"vehicles": [
	{
	"_links": {
	"self": {
	"href": "http://virtual-vehicles.com:8581/vehicles/554e8bd4c830093007d8b949"
	},
	"up": {
	"href": "http://virtual-vehicles.com:8581/vehicles"
	}
	},
	"year": 2015,
	"make": "Chevrolet",
	"model": "Corvette Stingray",
	"color": "Blue",
	"type": "Coupe",

	"mileage": 200, "id": "554e8bd4c830093007d8b949", "createdAt": "2015-05-09T22:36:04.808Z", "updatedAt": "2015-05-09T22:36:04.808Z" }, ...] } }

Function	Read a single vehicle
Method	GET
URI	/vehicles/{id}.{format} /vehicles/554e8bd4c830093007d8b949
Request Header	Header: Authorization Value: Bearer {jwt}
Request	none
Status	200 OK

Response Body	Single vehicle object
	
	{
	"_links": {
	"self": {
	"href": "http://virtual-vehicles.com:8581/vehicles/554e8bd4c830093007d8b949"
	},
	"up": {
	"href": "http://virtual-vehicles.com:8581/vehicles"
	}
	},
	"year": 2015,
	"make": "Chevrolet",
	"model": "Corvette Stingray",
	"color": "Blue",
	"type": "Coupe",
	"mileage": 200,
	"id": "554e8bd4c830093007d8b949",
	"createdAt": "2015-05-09T22:36:04.808Z",
	"updatedAt": "2015-05-09T22:36:04.808Z"
	}

Function	Update an existing vehicle (commonly used to update milage or change vehicle color)
Method	PUT
URI	/vehicles/{id}.{format} /vehicles/554e8bd4c830093007d8b949
Request Header	Header: Authorization Value: Bearer {jwt}

Request Body (raw)	{
	"year": 2015,
	"make": "Chevrolet",
	"model": "Corvette Stingray",
	"color": "White",
	"type": "Coupe",
	"mileage": 902,
	"createdAt": "2015-05-09T22:36:04.808Z"
	}
	
	"updatedAt" and "id" not required

Status	201 Created

Response Body	{
	"_links": {
	"self": {
	"href": "http://virtual-vehicles.com:8581/vehicles/554e8bd4c830093007d8b949"
	},
	"up": {
	"href": "http://virtual-vehicles.com:8581/vehicles"
	}
	},
	"year": 2015,
	"make": "Chevrolet",
	"model": "Corvette Stingray",
	"color": "White",
	"type": "Coupe",
	"mileage": 902,
	"id": "554e8bd4c830093007d8b949",
	"createdAt": "2015-05-09T22:36:04.808Z",
	"updatedAt": "2015-05-23T12:07:31.829Z"
	}

Function	Delete an existing vehicle
Method	DELETE
URI	/vehicles/{id}.{format} /vehicles/554e8bd4c830093007d8b949
Request Header	Header: Authorization Value: Bearer {jwt}
Request	none
Status	204 No Content
Response Body	none

Function	Service health ping
Method	GET
URI	/vehicles/utils/ping.{format}
Request	none
Status	200 OK
Response Body	Boolean: true or false

TOP

Maintenance Service	
Function	Create a new maintenance record
Method	POST
URI	/maintenances.{format}
Request Header	Header: Authorization Value: Bearer {jwt}

Request Body (raw)	{
	"vehicleId": "554e8bd4c830093007d8b949",
	"serviceDateTime": "2015-05-07T08:22:03.400Z",
	"mileage": 4025,
	"type": "Oil change",
	"notes": "Use same grade oil as last service."
	}

Status	201 Created

Response Body	New maintenance record object
	
	{
	"_links": {
	"self": {
	"href": "http://localhost:8583/maintenances/554eca58c83012f8863d910f"
	},
	"up": {
	"href": "http://localhost:8583/maintenances"
	}
	},
	"vehicleId": "554e8bd4c830093007d8b949",
	"serviceDateTime": "2015-05-07T08:22:03.400Z",
	"mileage": 4025,
	"type": "Oil change",
	"notes": "Use same grade oil as last service.",
	"id": "554eca58c83012f8863d910f",
	"createdAt": "2015-05-10T03:02:48.703Z",
	"updatedAt": "2015-05-10T03:02:48.703Z"
	}

MongoDB Document View (BSON)	{
	"_id" : ObjectId("554eca58c83012f8863d910f"),
	"className" : "com.example.maintenance.objectid.Record",
	"vehicleId" : "554e8bd4c830093007d8b949",
	"serviceDateTime" : "2015-05-05T02:39:03.400Z",
	"mileage" : 4025,
	"type" : "Oil change",
	"notes" : "Use same grade oil as last service.",
	"createdAt" : ISODate("2015-05-10T03:02:48.703Z"),
	"updatedAt" : ISODate("2015-05-10T03:02:48.703Z")
	}

Function	Read, filter, sort, count, paginate maintenance records filter = org.restexpress.common.query QueryFilter instance (uses contains method, no wildcard option) sort = org.restexpress.common.query QueryOrder instance (use - for descending order) limit and offset = org.restexpress.common.query QueryRange instance (offset is optional - default is 0) countOnly=true returns document count with empty maintenance records object in response body All query params may be combined
Method	GET
URI	/maintenances.{format} /maintenances?filter={key::value|key::value|...}.{format} /maintenances?filter=vehicleId::554e8bd4c830093007d8b949 /maintenances?filter=type::Oil|notes::change /maintenances?sort={key|-key|...}&limit={int}&offset={int} /maintenances?sort=type&limit=20 /maintenances?sort=type|-notes&limit=10&offset=20 /maintenances?countOnly=true /maintenances?countOnly=true&filter=type::Oil
Request Header	Header: Authorization Value: Bearer {jwt}
Request	(5) optional URL params above: filter, sort, limit, offset, countOnly
Status	200 OK

Response Body	Potentially one or more maintenance record objects
	
	{
	"_links": {
	"self": {
	"href": "http://localhost:8583/maintenances"
	}
	},
	"_embedded": {
	"records": [
	{
	"_links": {
	"self": {
	"href": "http://localhost:8583/maintenances/554eca58c83012f8863d910f"
	},
	"up": {
	"href": "http://localhost:8583/maintenances"
	}
	},
	"vehicleId": "554e8bd4c830093007d8b949",
	"serviceDateTime": "2015-05-05T02:39:03.400Z",
	"mileage": 4025,
	"type": "Oil change",
	"notes": "Use same grade oil as last service.",

	"id": "554eca58c83012f8863d910f", "createdAt": "2015-05-10T03:02:48.703Z", "updatedAt": "2015-05-10T03:02:48.703Z" }, ...] } }

Function	Read a single maintenance record
Method	GET
URI	/maintenances/{id}.{format} /maintenances/554eca58c83012f8863d910f
Request Header	Header: Authorization Value: Bearer {jwt}
Request	none
Status	200 OK

Response Body	Single maintenance record object
	
	{
	"_links": {
	"self": {
	"href": "http://localhost:8583/maintenances/554eca58c83012f8863d910f"
	},
	"up": {
	"href": "http://localhost:8583/maintenances"
	}
	},
	"vehicleId": "554e8bd4c830093007d8b949",
	"serviceDateTime": "2015-05-07T08:22:03.400Z",
	"mileage": 4025,
	"type": "Oil change",
	"notes": "Use same grade oil as last service.",
	"id": "554eca58c83012f8863d910f",
	"createdAt": "2015-05-10T03:02:48.703Z",
	"updatedAt": "2015-05-10T03:02:48.703Z"
	}

Function	Update an existing maintenance record (commonly used to change service date and time or change type)
Method	PUT
URI	/maintenances/{id}.{format} /maintenances/554eca58c83012f8863d910f
Request Header	Header: Authorization Value: Bearer {jwt}

Request Body (raw)	{
	"vehicleId": "554e8bd4c830093007d8b949",
	"serviceDateTime": "2015-05-08T11:26:03.400Z",
	"mileage": 4025,
	"type": "Oil change and new air filter",
	"notes": "Use same grade oil as last service.",
	"createdAt": "2015-05-10T03:02:48.703Z"
	}
	
	"updatedAt" and "id" not required

Status	201 Created

Response Body	{
	"_links": {
	"self": {
	"href": "http://localhost:8583/maintenances/554eca58c83012f8863d910f"
	},
	"up": {
	"href": "http://localhost:8583/maintenances"
	}
	},
	"vehicleId": "554e8bd4c830093007d8b949",
	"serviceDateTime": "2015-05-08T11:26:03.400Z",
	"mileage": 4025,
	"type": "Oil change and new air filter",
	"notes": "Use same grade oil as last service.",
	"id": "554eca58c83012f8863d910f",
	"createdAt": "2015-05-10T03:02:48.703Z"
	"updatedAt": "2015-05-10T03:02:48.703Z"
	}

Function	Delete an existing existing maintenance record
Method	DELETE
URI	/maintenances/{id}.{format} /maintenances/554eca58c83012f8863d910f
Request Header	Header: Authorization Value: Bearer {jwt}
Request	none
Status	204 No Content
Response Body	none

Function	Service health ping
Method	GET
URI	/maintenances/utils/ping.{format}
Request	none
Status	200 OK
Response Body	Boolean: true or false

TOP

Valet Parking Service	
Function	Create a new valet parking transaction
Method	POST
URI	/valets.{format}
Request Header	Header: Authorization Value: Bearer {jwt}

Request Body (raw)	{
	"vehicleId": "554e8bd4c830093007d8b949",
	"dateTimeIn": "2015-05-07T08:22:03.400Z",
	"parkingLot": "Lot A: Main and Broad Streets",
	"parkingSpot": 24,
	"notes": "Existing dent on drivers side front fender"
	}

Status	201 Created

Response Body	New valet parking transaction object
	
	{
	"_links": {
	"self": {
	"href": "http://localhost:8585/valets/554ecdf7c8306ec3631472cd"
	},
	"up": {
	"href": "http://localhost:8585/valets"
	}
	},
	"vehicleId": "554e8bd4c830093007d8b949",
	"dateTimeIn": "2015-05-07T08:22:03.400Z",
	"parkingLot": "Lot A: Main and Broad Streets",
	"parkingSpot": 24,
	"notes": "Existing dent on drivers side front fender",
	"id": "554ecdf7c8306ec3631472cd",
	"createdAt": "2015-05-10T03:18:15.097Z",
	"updatedAt": "2015-05-10T03:18:15.097Z"
	}

MongoDB Document View (BSON)	{
	"_id" : ObjectId("554ecdf7c8306ec3631472cd"),
	"className" : "com.example.valet.objectid.Transaction",
	"vehicleId" : "554e8bd4c830093007d8b949",
	"dateTimeIn" : "2015-05-07T08:22:03.400Z",
	"parkingLot" : "Lot A: Main and Broad Streets",
	"parkingSpot" : 24,
	"notes" : "Existing dent on drivers side front fender",
	"createdAt" : ISODate("2015-05-10T03:18:15.097Z"),
	"updatedAt" : ISODate("2015-05-10T03:18:15.097Z")
	}

Function	Read, filter, sort, count, paginate valet parking transactions filter = org.restexpress.common.query QueryFilter instance (uses contains method, no wildcard option) sort = org.restexpress.common.query QueryOrder instance (use - for descending order) limit and offset = org.restexpress.common.query QueryRange instance (offset is optional - default is 0) countOnly=true returns document count with empty valet parking transactions object in response body All query params may be combined
Method	GET
URI	/valets.{format} /valets?filter={key::value|key::value|...}.{format} /valets?filter=vehicleId::554e8bd4c830093007d8b949 /valets?filter=parkingLot::Lot A|notes::dent /valets?sort={key|-key|...}&limit={int}&offset={int} /valets?sort=parkingLot&limit=20 /valets?sort=parkingLot|-notes&limit=10&offset=20 /valets?countOnly=true /maintenances?countOnly=true&filter=parkingLot::Lot A
Request Header	Header: Authorization Value: Bearer {jwt}
Request	(5) optional URL params above: filter, sort, limit, offset, countOnly
Status	200 OK

Response Body	Potentially one or more valet parking transaction objects
	
	{
	"_links": {
	"self": {
	"href": "http://localhost:8585/valets"
	}
	},
	"_embedded": {
	"transactions": [
	{
	"_links": {
	"self": {
	"href": "http://localhost:8585/valets/554ecdf7c8306ec3631472cd"
	},
	"up": {
	"href": "http://localhost:8585/valets"
	}
	},
	"vehicleId": "554e8bd4c830093007d8b949",
	"dateTimeIn": "2015-05-07T08:22:03.400Z",
	"dateTimeOut": "2015-05-07T12:41:22.400Z",
	"parkingLot": "Lot A: Main and Broad Streets",
	"parkingSpot": 24,

	"notes": "Existing dent on drivers side front fender", "id": "554ecdf7c8306ec3631472cd", "createdAt": "2015-05-10T03:18:15.097Z", "updatedAt": "2015-05-10T14:57:02.388Z" }, ...] } }

Function	Read a single valet parking transaction
Method	GET
URI	/valets/{id}.{format} /valets/554eca58c83012f8863d910f
Request Header	Header: Authorization Value: Bearer {jwt}
Request	none
Status	200 OK

Response Body	Single valet parking transaction object
	
	{
	"_links": {
	"self": {
	"href": "http://localhost:8585/valets/554ecdf7c8306ec3631472cd"
	},
	"up": {
	"href": "http://localhost:8585/valets"
	}
	},
	"vehicleId": "554e8bd4c830093007d8b949",
	"dateTimeIn": "2015-05-07T08:22:03.400Z",
	"dateTimeOut": "2015-05-07T12:41:22.400Z",
	"parkingLot": "Lot A: Main and Broad Streets",
	"parkingSpot": 24,
	"notes": "Existing dent on drivers side front fender",
	"id": "554ecdf7c8306ec3631472cd",
	"createdAt": "2015-05-10T03:18:15.097Z",
	"updatedAt": "2015-05-10T03:18:15.097Z"
	}

Function	Update an existing valet parking transaction (commonly used to add dateTimeOut to transaction)
Method	PUT
URI	/valets/{id}.{format} /valets/554eca58c83012f8863d910f
Request Header	Header: Authorization Value: Bearer {jwt}

Request Body (raw)	{
	"vehicleId": "554e8bd4c830093007d8b949",
	"dateTimeIn": "2015-05-07T08:22:03.400Z",
	"dateTimeOut": "2015-05-07T12:41:22.400Z",
	"parkingLot": "Lot A: Main and Broad Streets",
	"parkingSpot": 24,
	"notes": "Existing dent on drivers side front fender",
	"createdAt": "2015-05-10T03:18:15.097Z"
	}
	
	"updatedAt" and "id" not required

Status	201 Created

Response Body	{
	"_links": {
	"self": {
	"href": "http://localhost:8585/valets/554ecdf7c8306ec3631472cd"
	},
	"up": {
	"href": "http://localhost:8585/valets"
	}
	},
	"vehicleId": "554e8bd4c830093007d8b949",
	"dateTimeIn": "2015-05-07T08:22:03.400Z",
	"dateTimeOut": "2015-05-07T12:41:22.400Z",
	"parkingLot": "Lot A: Main and Broad Streets",
	"parkingSpot": 24,
	"notes": "Existing dent on drivers side front fender",
	"id": "554ecdf7c8306ec3631472cd",
	"createdAt": "2015-05-10T03:18:15.097Z",
	"updatedAt": "2015-05-10T03:18:15.097Z"
	}

Function	Delete an existing valet parking transaction
Method	DELETE
URI	/valets/{id}.{format} /valets/554eca58c83012f8863d910f
Request Header	Header: Authorization Value: Bearer {jwt}
Request	none
Status	204 No Content
Response Body	none

Function	Service health ping
Method	GET
URI	/valets/utils/ping.{format}
Request	none
Status	200 OK
Response Body	Boolean: true or false

TOP