/direct POST

Parameter:

motor_l	integer -10 … 10	required
motor_r	integer -10 … 10	required
user	email	required
remote_key	gets returned when remote is enabled	required

Info: For direct motor control, enable remote control has to be called first

/toggle-remote-control POST

Parameter:

status	enabled / disabled	required
user	email	required

Info: Enable / Disable remote control

status -> enabled:

additional Parameter:

verify_key	generated verification hash	required

Info: returns session unique ‘remote_key’

status -> disabled:

additional Parameter:

remote_key	gets returned when remote is enabled	required

/add-order POST

Parameter:

location	ex. Morty	required
user	email	required
priority	True / False	not required
verify_key	generated verification hash	required

Info: Add Order to que

/cancel-order POST

Parameter:

user	email	required
verify_key	generated verification hash	required

Info: Cancel open order

/confirm-delivery POST
Parameter:

user	email	required
verify_key	generated verification hash	required

Info:Mark your order as delivered

/get-orders GET
Parameter: None
Info: Get all open orders as JSON

Status Codes:

CODE	TYPE	DEFINITION
100	OK	Call Successful
101	ERROR	no open order
102	ERROR	no open remote session
103	ERROR	already remote controlled returns current remote-user as data['user']
104	ERROR	not your remote session returns current remote-user as data['user']
105	ERROR	speed for left motor not defined
106	ERROR	speed for right motor not defined
107	ERROR	Unknown left speed Identifier
108	ERROR	Unknown right speed Identifier
109	ERROR	Invalid left speed Identifier - speed Identifier has to be float
110	ERROR	Invalid right speed Identifier - speed Identifier has to be float
111	ERROR	remote control not enabled
115	ERROR	user not defined

116	ERROR	authorisation failed - key invalid
117	ERROR	timestamp missing
199	ERROR	General Error, see returned response data

Verification Hash:

```
def generate_verification_hash(identifier, secret_key):
    import hashlib
    from datetime import datetime
    hash = hashlib.sha224((identifier + secret_key +
str(datetime.utcnow().timestamp())).encode('UTF-8')).hexdigest()
    return hash
```

For Interface-to-Robot-Calls the identifier is the user, so in most cases the email address.

```
def verify_call(hash, timestamp, identifier, secret_key):
    import hashlib
    from datetime import datetime
    try:
        timestamp = float(timestamp)
    except:
        return False
    if datetime.utcnow().timestamp() - timestamp < 30:
        compare_hash = hashlib.sha224((identifier + secret_key +
str(timestamp)).encode('UTF-8')).hexdigest()
        if hash == compare_hash:
            return True
    return False
```

For Robot-to-Interface-Calls the identifier is an empty string.

Keys:

```
secret_keys = {'add_order': '45Aa*+=H5Nc_NdLm',
               'cancel_order': '^SF%NqDZB8av_KbB',
               'confirm_order': 'EdHD&k5X$y4UTv%z',
               'enable_remote': '@Qzc?UwqkVbh5z@W',
               'update_remote_control_status': 'A5_8umdWZd84RLar',
               'update_order_list': 'yYRJ6=P*H9e7x^nA',
               'update_last_barcode': '9qm*YAG#rQJK+fY^'}
```