**HTTP Response**

When getting a response back, check the HTTP status code first. If it is 200 OK, that means the request was received, the authorization succeeded, and the request was processed successfully; a standard acknowledgement is included in the body. Otherwise, the HTTP status code (401 Unauthorized, 500 Internal Server Error, etc.) indicates what went wrong. The HTTP header may include a user friendly message and other key-value pairs depending on the route.

**200 OK Acknowledgement Body / Result Body**

{

“data”: {} (specified for each route, null otherwise)

}

**Input Parameters**

Parameters are formatted in JSON. Send a single dictionary {} without nesting.

Input parameters in [brackets] are optional.

An “Authorize: UserPass” route request must have these key-value pairs in its JSON body:

* String username
* String password

An “Authorize: Session” route request must have the following HTTP header:

Authorization: Bearer TOKEN

Where TOKEN is replaced with the session token returned by api/user/login. Currently, sessions expire after 20 minutes.

## 

## API

**Register User**

POST api/user/register

Authorize: None

IN

* String username
* String password
* String email
* String address
* String zip
* String user\_type (“business” / ”client”)
* Client
  + String first\_name
  + String last\_name
  + String cell\_phone
  + Bool paying
* Business
  + String name
  + String work\_phone
  + [String instructions]

OUT

* Acknowledgement
  + OK, OTHER, INVALID\_USERNAME, INVALID\_PASSWORD, INVALID\_EMAIL, INVALID\_ADDRESS, INVALID\_ZIP, INVALID\_USER\_TYPE, INVALID\_FIRST\_NAME, INVALID\_LAST\_NAME, INVALID\_CELL\_PHONE, INVALID\_NAME, INVALID\_WORK\_PHONE, INVALID\_INSTRUCTIONS

**Login**

POST api/user/login

Authorize: UserPass

OUT

* Acknowledgement
  + OK, OTHER, INVALID\_USERNAME, INVALID\_PASSWORD

Data on OK:

“session\_token”: string

**Logout**

POST api/user/logout

Authorize: Session

OUT

* Acknowledgement
  + OK, OTHER

**Get User Info**

POST api/user/getinfo

Authorize: Session

OUT

* Acknowledgement
  + OK, OTHER

Data on OK:

“username”: string

“email”: string

“address”: string

“zip”: string

“user\_type”: string (“client” OR “business”)

“first\_name”: string (NULL) (client only)

“last\_name”: string (NULL) (client only)

“cell\_phone”: string (NULL) (client only)

“name”: string (NULL) (business only)

“work\_phone”: string (NULL) (business only)

“instructions”: string (NULL) (business only)

**Get User Type**

POST api/user/getusertype

Authorize: Session

Data on OK:

“user\_type”: string (“client” OR “business”)

**Logout All Sessions**

POST api/user/logoutall

Authorize: UserPass

OUT

* Acknowledgement
  + OK, OTHER

**Set User Info**

POST api/user/setinfo

Authorize: UserPass

IN

* String new\_username
* String email
* String address
* String zip
* Client
  + String first\_name
  + String last\_name
  + String cell\_phone
* Business
  + String name
  + String work\_phone
  + [String instructions]

OUT

* Acknowledgement
  + OK, OTHER, INVALID\_USERNAME, INVALID\_PASSWORD, INVALID\_EMAIL, INVALID\_ADDRESS, INVALID\_ZIP, INVALID\_USER\_TYPE, INVALID\_FIRST\_NAME, INVALID\_LAST\_NAME, INVALID\_CELL\_PHONE, INVALID\_NAME, INVALID\_WORK\_PHONE, INVALID\_INSTRUCTIONS

**Set Password**

POST api/user/setpassword

Authorize: UserPass

IN

* String new\_password

OUT

* Acknowledgement
  + OK, OTHER, INVALID\_PASSWORD

**Delete User (client only)**

POST api/user/delete

Authorize: UserPass

OUT

* Acknowledgement
  + OK, OTHER

**FIXME: Reset Password Request and Reset Password**

**Get Business (for public view)**

POST api/business/getbusiness

Authorize: None

IN

* Int bid

OUT

* Acknowledgement
  + OK, OTHER, INVALID\_BID

Data on OK:

“email”: string

“address”: string

“zip”: string

“name”: string

“work\_phone”: string

“instructions”: string (NULL)

**Get Packages**

POST api/package/getpackages

Authorize: Session

IN

* Bool only\_eligible
  + For client:
    - True: Packages that can be claimed or received by this client
    - False: Also includes client’s received packages
  + For business:
    - True: Business’s packages that are not yet received by a client
    - False: Also includes business’s packages that have been received

OUT.

* Acknowledgement
  + OK, OTHER

Data on OK:

“packages”: [

{

“pid”: int

“owner\_bid”: int

“business\_name”: string

“business\_address”: string

“claimer\_cid”: int (NULL)

“name”: string

“description”: string (NULL)

“quantity”: string (NULL)

“price”: decimal

“created”: datetime

“expires”: datetime (NULL)

“claimed”: datetime (NULL)

“received”: datetime (NULL)

}

...

]

**Create Package (business only)**

POST api/package/createpackage

Authorize: Session

IN

* String name
* [String description]
* [String quantity]
* Decimal price
* [Datetime expires] (in UTC)

OUT

* Acknowledgement
  + OK, OTHER, INVALID\_NAME, INVALID\_DESCRIPTION, INVALID\_QUANTITY, INVALID\_PRICE, INVALID\_EXPIRES

**Delete Package (business only)**

POST api/package/deletepackage

Authorize: Session

IN

* Int pid

OUT

* Acknowledgement
  + OK, OTHER, INVALID\_PID

**Set Package’s Claim**

POST api/package/claim

Authorize: Session

IN

* Int pid
* Bool claim (ignored for businesses, always false)

OUT

* Acknowledgement
  + OK, OTHER, INVALID\_PID

**Mark Package Given (business only)**

POST api/package/markreceived

Authorize: Session

IN

* Int pid

OUT

* Acknowledgement
  + OK, OTHER, INVALID\_PID