**HTTP Request Parameters**

Request 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.

**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 JSON body is included in the response. Otherwise, the HTTP status code (401 Unauthorized, 500 Internal Server Error, etc.) indicates what went wrong. There may be a JSON body.

**Response Body Format (JSON)**

{

“value”: object,

“another\_value”: object,

...

}

Many responses give only a “data” string which may contain a relatively user-friendly message. A (NULL) next to a key-value pair in the documentation indicates the pair may be null.

**HTTP Status Codes**

* **200 OK** - API route completed successfully
* **400 Bad Request** - API route determined input is invalid
* **401 Unauthorized** - Invalid or incorrect credentials (user + pass or session token)
* **403 Forbidden** - Authenticated but not allowed into this route, either because the wrong form of authentication was used or because the type of user is not allowed
* **500 Internal Server Error** - Usually means an exception was thrown and not caught; this may happen for invalid inputs, so many such cases should be caught and replaced with an explicit 400 Bad Request

## 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”)
* If client
  + String first\_name
  + String last\_name
  + String cell\_phone
  + Bool paying
* If business
  + String name
  + String work\_phone
  + [String instructions]

Body on OK:

“data”: string

Body on Bad Request:

“data”: string

**Login**

POST api/user/login

Authorize: UserPass

Body on OK:

“session\_token”: string

**Logout**

POST api/user/logout

Authorize: Session

Body on OK:

“data”: string

**Get User Info**

POST api/user/getinfo

Authorize: Session

Body on OK:

“username”: string

“email”: string

“address”: string

“zip”: string

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

“cid” : int (client only)

“first\_name”: string (client only)

“last\_name”: string (client only)

“cell\_phone”: string (client only)

“paying”: bool (client only) [FIXME: not implemented]

“name”: string (business only)

“work\_phone”: string (business only)

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

Body on Bad Request:

“data”: string

**Get User Type**

POST api/user/getusertype

Authorize: Session

Body on OK:

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

Body on Bad Request:

“data”: string

**Logout All Sessions**

POST api/user/logoutall

Authorize: UserPass

Body on OK:

“data”: string

**Set User Info**

POST api/user/setinfo

Authorize: UserPass

IN

* String new\_username
* String email
* String address
* String zip
* If client
  + String first\_name
  + String last\_name
  + String cell\_phone
  + Bool paying [FIXME: not implemented]
* If business
  + String name
  + String work\_phone
  + [String instructions]

Body on OK:

“data”: string

Body on Bad Request:

“data”: string

**Set Password**

POST api/user/setpassword

Authorize: UserPass

IN

* String new\_password

Body on OK:

“data”: string

**Delete User (client only)**

POST api/user/delete

Authorize: UserPass

Body on OK:

“data”: string

**FIXME: Need routes to Request Reset Password (sends an email) and Reset Password (reset password with non-UserPass authorization, related to that email)**

**FIXME: Need something similar for when user forgets username (or just replace usernames with email addresses)**

**Get Business (for public view)**

POST api/business/getbusiness

Authorize: None

IN

* Int bid

Body on OK:

“email”: string

“address”: string

“zip”: string

“name”: string

“work\_phone”: string

“instructions”: string (NULL)

Body on Bad Request:

“data”: string

**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

Body 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)

},

...

]

Body on Bad Request:

“data”: string

**Create Package (business only)**

POST api/package/createpackage

Authorize: Session

IN

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

Body on OK:

“data”: string

**Delete Package (business only)**

POST api/package/deletepackage

Authorize: Session

IN

* Int pid

Body on OK:

“data”: string

Body on Bad Request:

“data”: string

**Set Package’s Claim**

POST api/package/claim

Authorize: Session

IN

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

Body on OK:

“data”: string

Body on Bad Request:

“data”: string

**Mark Package Received (business only)**

POST api/package/markreceived

Authorize: Session

IN

* Int pid

Body on OK:

“data”: string

Body on Bad Request:

“data”: string