- **/user/login**
  - Type - POST
  - Takes in JSON

```
{
    "email" : "users email",
    "password" : "users password"
}
```

  - Used to log a user in
  - Returns json data with users account information

- **/user/create**
  - Type - POST
  - Takes in JSON

```
{
    "firstName" : "first name",
    "lastName" : "last name",
    "phoneNumber" : "phone number",
    "email" : "email",
    "address" : "users address",
    "userName" : "user name",
    "password" : "users password"
}
```

  - Used to create a new user account
  - Will return a string
    - Account Exists - email used already has an account
    - Account Created - account was created successfully
  - Once account is created it will send an email with a verification code to the user, there will need to be a pop up or separate page after they create their account so they can enter the code

- **/user/update**
  - Type - PUT
  - Takes in JSON
    - Even if the field has not changed, still send the old info with the request. All data fields need to be full

```json
{
    "email" : "update email",
    "firstName" : "update first name",
    "lastName" : "updated last name",
    "phoneNumber" : "updated phone number",
    "address" : "updated address",
    "userName" : "Updated user name"
}
```

  - Used to update character account information
    - First name
    - Last name
    - Phone number
    - Address
    - Username
  - Returns a boolean
    - True it was updated successfully
    - False there was an error
- **/user/updatePassword**
  - Type - PUT
  - Takes in JSON
    - Make them type in their old password as well as the new password, it will be used to verify they type in the correct current password for the account before it is updated to the new one

```json
{
    "email" : "user email",
    "oldPassword" : "users old password",
    "newPassword" : "users new password"
}
```

  - Allows user to update password from their dashboard
  - Returns a boolean
    - True for successful update
    - False if they old password didn't match correctly

- **/user/verify**
  - Type-Put
  - Takes in JSON
    - Send the users email along with the request for the purpose of finding their account in the database

```
{
    "email" : "users email",
    "verificationCode" : "typed in code"
}
```

  - Used after a new account is made and they are verifying their email
  - Returns a boolean

- **/user/resendVerification**
  - Type-PUT
  - Takes in plain text
    - Just send the users email as plain text to the endpoint, JSON format will not work
  - Used to resend a new verification code to the users email
  - Doesn't return anything

- **/user/delete**
  - Type - DELETE
  - Takes in plain text
    - Send the users email as plain text to the endpoint, JSON format will not work
  - Used to let a user delete their account from our database
  - Returns a boolean
    - True deleted successfully
    - False there was an error deleting the account
- **/trip/createTrip**
  - Type - PUT
  - Takes in JSON data
    - The usersID that the trip is being made for
    - The destinationID the user is selecting
    - The 3 generic tags for the excursions selected

```
{
    "usersID" : usersID,
    "destinationID" : selected destination id,
    "excursion1Tag" : "generic tag for excursion 1",
    "excursion2Tag" : "generic tag for excursion 2",
    "excursion3Tag" : "generic tag for excursion 3"
}
```

- ○ Used to add a trip to the database based on the users selected location and excursions.
- ○ Returns the the trip information to be displayed to the user after it is created

```
{
    "destination": {
        "destinationID": 1,
        "destinationState": "Alabama",
        "destinationCity": "Huntsville",
        "destinationClimateTag": "Warm",
        "destinationPicture": null
    },
    "excursion1": {
        "excursionID": 4,
        "excursionTitle": "camping in somewhere else",
        "excursionDescription": "go camping",
        "typeOfExcursionTag": "outdoors",
        "excursionPicture": "asfdsf",
        "destinationID": 1
    },
    "excursion2": {
        "excursionID": 5,
        "excursionTitle": "theme park somewhere else",
        "excursionDescription": "go to a theme park",
        "typeOfExcursionTag": "adventure",
        "excursionPicture": "asfa",
        "destinationID": 1
    },
    "excursion3": {
        "excursionID": 6,
        "excursionTitle": "pizza somewhere else",
        "excursionDescription": "go eat pizza",
        "typeOfExcursionTag": "food",
        "excursionPicture": "afaf",
        "destinationID": 1
    }
}
```

- ● /trip/like
  - ○ Type- PUT
  - ○ Send just the tripID
  - ○ Used when a user wants to like a trip
  - ○ Returns nothing

- **/trip/comment**
  - Type - POST
  - Takes in JSON data
    - The ID of the user posting the comment
    - The ID of the trip they are posting the comment on
    - The comment itself
      ```json
      {
          "usersID" : 2,
          "tripID" : 3,
          "comment" : "This is a really good trip you have here!"
      }
      ```
  - Used when a user wants to comment on a trip that has shown up on their page
  - Currently returns nothing
- **/user/userTrips**
  - Type - POST
  - Send just the users ID
  - Used when a user goes to their dashboard to get all trips
  - Returns a list of trips that contains all the trip details
- **/user/getComments**
  - Type - POST
  - Send just the trip ID front end is getting the comments for
  - Used when a user wants to view the comments on a post
  - Returns a list of comments for the trip with the username of the person who made the comment