# 

# Guidelines

* Use ES6/7 if possible. Babel is already set up.
* Express has been setup but you can use any other API frameworks as your favour.
* Add any NPM packages you need.
* Feel free to tweak the structure or create source files, etc. as you need.
* You may have questions but use your best judgement – we’re interested in understanding your decisions.
* Recommended time limit is 2 hours. If you have time left, consider the stretch requirement (bonus) you feel comfortable with.
* Once completed, you can merge to master.

# Requirements

## **1. Implement an API method to authenticate a user**

* Add a route config (API definition) in app/routes.js to handle requests.
* Take ‘clientId’ and ‘secretKey’ params from the requests.
* Return a JSON response (200) with a temporary token that expires in 1 minute and includes ‘apiUserId’ if any of ‘apiUsers’ (app/resources/apiUsers) is matched with the input parameters.
* You must handle any general errors that might happen.
* Note that you should use the token for authentication when calling the below API methods.
* Write unit tests.

## **2. Implement an API method to get groups of animals**

* Add a route config (API definition) in app/routes.js to handle requests.
* Take a token from requests and verify it.
* If the token is invalid, return a proper error response.
* If the token is valid, obtain ‘apiUserId’ from the token and return a JSON response (200) with the following requirements.
* Filter the list (app/resources/animals.json) by the ‘apiUserId’.
* Group by types and accumulate the names in an array.
* Write unit tests.
* The following snippet is an expected outcome for ‘apiUserId’, ‘test1’.

{

"apiUserId": "test1",

"animals": [

{

"type": "dog",

"names": [ "Max", "Charlie", "Molly" ]

},

{

"type": "cat",

"names": [ "Kitty", "Tiger" ]

}

]

}

# Stretch requirement (Bonus)

## **Implement an API method to print the numbers from 1 to 100**

* For multiples of three print ‘Hey’ instead of number.
* For the multiples of five print ‘Yo’.
* For number which are multiples of both three and five print ‘HeyYo’.
* Expected outcome example.

1

2

Hey

4

Yo

Hey

7

8

Hey

Yo

11

Hey

13

14

HeyYo

16

17

Hey

19

Yo

Hey

22

23

Hey

Yo

26

Hey

28

29

HeyYo

…