

API Challenge link - <https://apichallenges.herokuapp.com/gui/challenges>

API documentation link - <https://apichallenges.herokuapp.com/docs>

The solved challenge has a green color.

Challenge solutions:

1. Create a new challenger session

Getting Started

If you want to track your challenge progress, in multi-user mode then you need to solve the challenges in this section to generate a unique ID that we can associate your progress with.

ID	Challenge	Done	Description
01	POST /challenger (201)	false	<p>Issue a POST request on the '/challenger' end point, with no body, to create a new challenger session. Use the generated X-CHALLENGER header in future requests to track challenge completion.</p> <p>► Hints ► Solution</p>

Solution:

The screenshot shows the API Challenge tool interface. A POST request is being made to <https://apichallenges.herokuapp.com/challenger>. The response details show a 201 Created status with a response time of 1068 ms and 864 B. The X-Challenger header is highlighted with a red box and its value is dbdad721-f2b9-4855-9632-41a310f3de06.

01	POST /challenger (201)	true	<p>Issue a POST request on the '/challenger' end point, with no body, to create a new challenger session. Use the generated X-CHALLENGER header in future requests to track challenge completion.</p> <p>► Hints ► Solution</p>
----	------------------------	------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

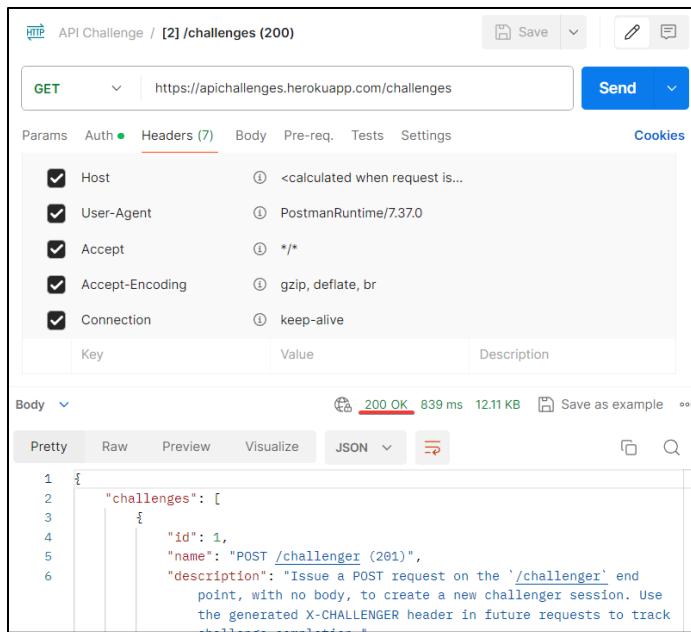
2. Retrieve a list of challenges

First Real Challenge

For your first challenge, get the list of challenges. You'll be able to use this to see your progress in your API Client, as well as using the GUI.

ID	Challenge	Done	Description
02	GET /challenges (200)	false	<p>Issue a GET request on the '/challenges' end point</p> <p>► Solution</p>

Solution:



The screenshot shows the Postman interface with a successful GET request to `https://apichallenges.herokuapp.com/challenges`. The response body is a JSON object with a single key `challenges` containing two items. The first item has an `id` of 1, a name of "POST /challenger (201)", and a description about creating a new challenger session. The second item has an `id` of 2, a title of "file paperwork", a done status of `false`, and an empty description.

02	GET /challenges (200)	true	Issue a GET request on the `/challenges` end point ► Solution
----	-----------------------	------	------------------------------------------------------------------

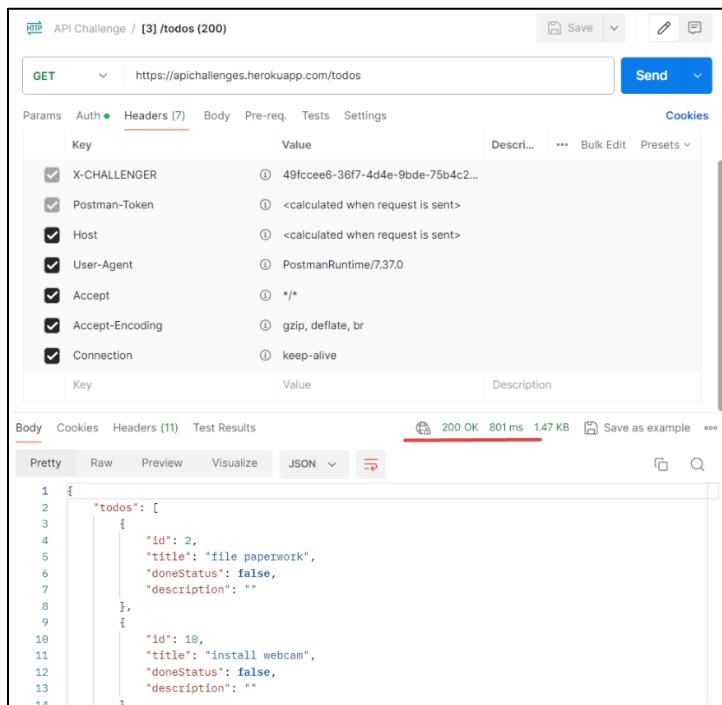
3. Retrieve a list of 'todos'

GET Challenges

To retrieve, or read information from an API we issue GET requests. This section has a bunch of GET request challenges to try out.

ID	Challenge	Done	Description
03	GET /todos (200)	false	Issue a GET request on the `/todos` end point ► Solution

Solution:



The screenshot shows the Postman interface with a successful GET request to `https://apichallenges.herokuapp.com/todos`. The response body is a JSON object with a single key `todos` containing two items. The first item has an `id` of 2, a title of "file paperwork", a done status of `false`, and an empty description. The second item has an `id` of 10, a title of "install webcam", a done status of `false`, and an empty description.

03	GET /todos (200)	true	Issue a GET request on the `/todos` end point ► Solution
----	------------------	------	-------------------------------------------------------------

4. Retrieve the 404 status-code

04	GET /todo (404) not plural	false	Issue a GET request on the `/todo` end point should 404 because nouns should be plural ► Solution
----	----------------------------	-------	------------------------------------------------------------------------------------------------------

Solution:

The screenshot shows a Postman interface with a failed GET request to `https://apichallenges.herokuapp.com/todo`. The response status is `404 Not Found`. The request headers include `Postman-Token`, `Host`, `User-Agent`, `Accept`, `Accept-Encoding`, `Connection`, and `Content-Type`.

04	GET /todo (404) not plural	true	Issue a GET request on the `/todo` end point should 404 because nouns should be plural ► Solution
----	----------------------------	------	------------------------------------------------------------------------------------------------------

5. Retrieve specific 'todo' with status code 200

05	GET /todos/{id} (200)	false	Issue a GET request on the `/todos/{id}` end point to return a specific todo ► Hints ► Solution
----	-----------------------	-------	-------------------------------------------------------------------------------------------------------

Solution:

The screenshot shows a Postman interface with a successful GET request to `https://apichallenges.herokuapp.com/todos/1`. The response status is `200 OK`. The request headers include `X-CHALLENGER`, `Postman-Token`, `Host`, `User-Agent`, `Accept`, and `Accept-Encoding`. The response body is a JSON object containing a single todo item.

05	GET /todos/{id} (200)	true	Issue a GET request on the `/todos/{id}` end point to return a specific todo ► Hints ► Solution
----	-----------------------	------	-------------------------------------------------------------------------------------------------------

6. Retrieve 404 status code when {todo id} doesn't exist

06	GET /todos/{id} (404)	false	Issue a GET request on the `/todos/{id}` end point for a todo that does not exist ► Hints ► Solution
----	-----------------------	-------	------------------------------------------------------------------------------------------------------------

Solution:

API Challenge / [6] /todos/{id} (404)

GET https://apichallenges.herokuapp.com/todos/10000

Send

Params Auth Headers (7) Body Pre-req. Tests Settings

Headers Hide auto-generated headers

Key	Value
X-CHALLENGER	49fccee6-36f7-4d4e-9bde-75b4...
Postman-Token	<calculated when request is sent>
Host	<calculated when request is sent>
User-Agent	PostmanRuntime/7.37.0
Accept	*/*

Body

404 Not Found 617 ms 853 B Save as example

Pretty Raw Preview Visualize JSON

```

1 {
2   "errorMessages": [
3     "Could not find an instance with todos/10000"
4   ]
5 }
```

06	GET /todos/{id} (404)	true	Issue a GET request on the `/todos/{id}` end point for a todo that does not exist ▶ Hints ▶ Solution
----	-----------------------	------	------------------------------------------------------------------------------------------------------------

7. Get a filtered list of todos

07	GET /todos (200) ?filter	false	Issue a GET request on the `/todos` end point with a query filter to get only todos which are 'done'. There must exist both 'done' and 'not done' todos, to pass this challenge. ▶ Hints ▶ Solution
----	--------------------------	-------	-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Solution:

API Challenge / [7] /todos (200) ?filter

GET https://apichallenges.herokuapp.com/todos?doneStatus=true

Send

Params Auth Headers (7) Body Pre-req. Tests Settings

Key	Value	Description
Host	<calculated when request is...>	
User-Agent	PostmanRuntime/7.37.0	
Accept	*/*	
Accept-Encoding	gzip, deflate, br	
Connection	keep-alive	

Body

200 OK 618 ms 797 B Save as example

Pretty Raw Preview Visualize JSON

```

1 {
2   "todos": []
3 }
```

07	GET /todos (200) ?filter	true	Issue a GET request on the `/todos` end point with a query filter to get only todos which are 'done'. There must exist both 'done' and 'not done' todos, to pass this challenge. ▶ Hints ▶ Solution
----	--------------------------	------	-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

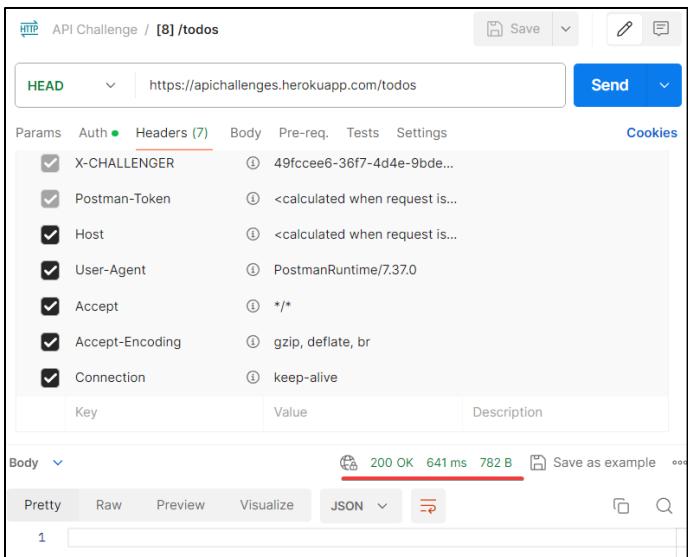
8. Create a HEAD request for todos

HEAD Challenges

A HEAD request, is like a GET request, but only returns the headers and status code.

ID	Challenge	Done	Description
08	HEAD /todos (200)	false	Issue a HEAD request on the `/todos` end point ▶ Solution

Solution:



The screenshot shows the Postman interface with a HEAD request to `https://apichallenges.herokuapp.com/todos`. The Headers tab is selected, showing the following configuration:

Key	Value
X-CHALLENGER	49fccee6-36f7-4d4e-9bde...
Postman-Token	<calculated when request is...
Host	<calculated when request is...
User-Agent	PostmanRuntime/7.37.0
Accept	*/*
Accept-Encoding	gzip, deflate, br
Connection	keep-alive

The response status is 200 OK, 641 ms, 782 B. A note on the right says: "Issue a HEAD request on the `/todos` endpoint" with a "► Solution" link.

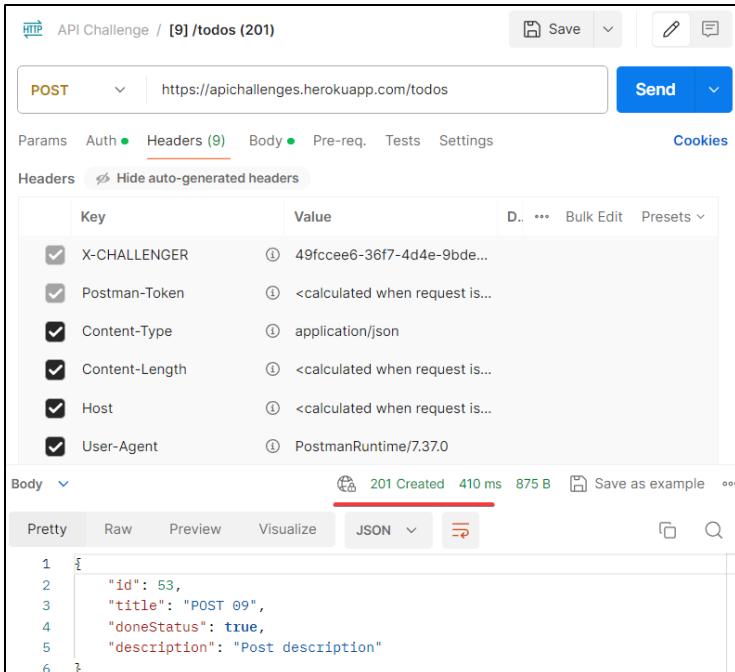
9. Create a new todo record

Creation Challenges with POST

A POST request can be used to create and update data, these challenges are to 'create' data. As a Hint, if you are not sure what the message body should be, try copying in the response from the associated GET request, and amending it.

ID	Challenge	Done	Description
09	POST /todos (201)	false	Issue a POST request to successfully create a todo ► Solution

Solution:



The screenshot shows the Postman interface with a POST request to `https://apichallenges.herokuapp.com/todos`. The Headers tab is selected, showing the following configuration:

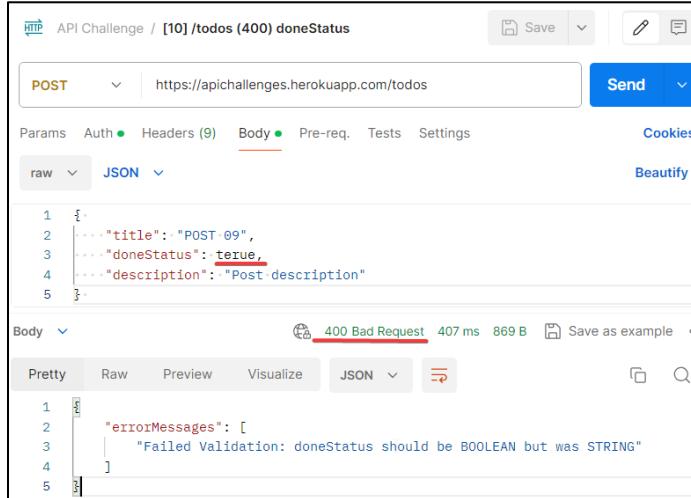
Key	Value
X-CHALLENGER	49fccee6-36f7-4d4e-9bde...
Postman-Token	<calculated when request is...
Content-Type	application/json
Content-Length	<calculated when request is...
Host	<calculated when request is...
User-Agent	PostmanRuntime/7.37.0

The response status is 201 Created, 410 ms, 875 B. A note on the right says: "Issue a POST request to successfully create a todo" with a "► Solution" link.

10. Fail validation on the 'doneStatus' field

10	POST /todos (400) doneStatus	false	Issue a POST request to create a todo but fail validation on the 'doneStatus' field ► Solution
----	---------------------------------	-------	-------------------------------------------------------------------------------------------------------

Solution:



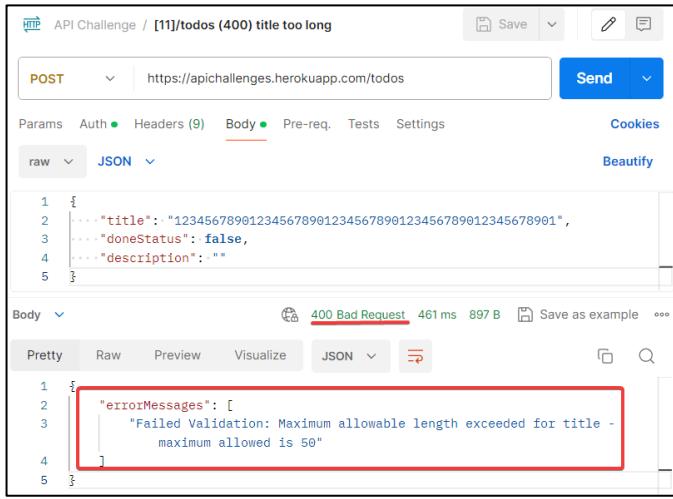
The screenshot shows the Postman interface with a POST request to `https://apichallenges.herokuapp.com/todos`. The request body is JSON with a typo in the 'doneStatus' field: `{"title": "POST-09", "doneStatus": true, "description": "Post description"}`. The response is a 400 Bad Request with the error message: `"errorMessages": ["Failed Validation: doneStatus should be BOOLEAN but was STRING"]`.

10	POST /todos (400) doneStatus	true	Issue a POST request to create a todo but fail validation on the 'doneStatus' field ► Solution
----	---------------------------------	------	-------------------------------------------------------------------------------------------------------

11. Create a todo record with fail validation on the 'title' field:

11	POST /todos (400) title too long	false	Issue a POST request to create a todo but fail length validation on the 'title' field because your title exceeds maximum allowable characters.
----	-------------------------------------	-------	------------------------------------------------------------------------------------------------------------------------------------------------

Solution:



The screenshot shows the Postman interface with a POST request to `https://apichallenges.herokuapp.com/todos`. The request body is JSON with a very long title: `{"title": "123456789012345678901234567890123456789012345678901", "doneStatus": false, "description": ""}`. The response is a 400 Bad Request with the error message: `"errorMessages": ["Failed Validation: Maximum allowable length exceeded for title - maximum allowed is 50"]`.

11	POST /todos (400) title too long	true	Issue a POST request to create a todo but fail length validation on the 'title' field because your title exceeds maximum allowable characters.
----	----------------------------------	------	------------------------------------------------------------------------------------------------------------------------------------------------

12. Create a todo record with fail validation on the 'title' field:

12	POST /todos (400) description too long	false	Issue a POST request to create a todo but fail length validation on the `description` because your description exceeds maximum allowable characters.
----	-------------------------------------------	-------	------------------------------------------------------------------------------------------------------------------------------------------------------

Solution:

The screenshot shows a Postman interface with a POST request to <https://apichallenges.herokuapp.com/todos>. The JSON body contains a todo item with a very long description. The response is a 400 Bad Request with an error message: "Failed Validation: Maximum allowable length exceeded for description - maximum allowed is 200".

12	POST /todos (400) description too long	true	Issue a POST request to create a todo but fail length validation on the `description` because your description exceeds maximum allowable characters.
----	-------------------------------------------	------	------------------------------------------------------------------------------------------------------------------------------------------------------

13. Create a todo record with max length of title and description fields

13	POST /todos (201) max out content	false	Issue a POST request to create a todo with maximum length title and description fields. ▶ Hints
----	--------------------------------------	-------	----------------------------------------------------------------------------------------------------

Solution:

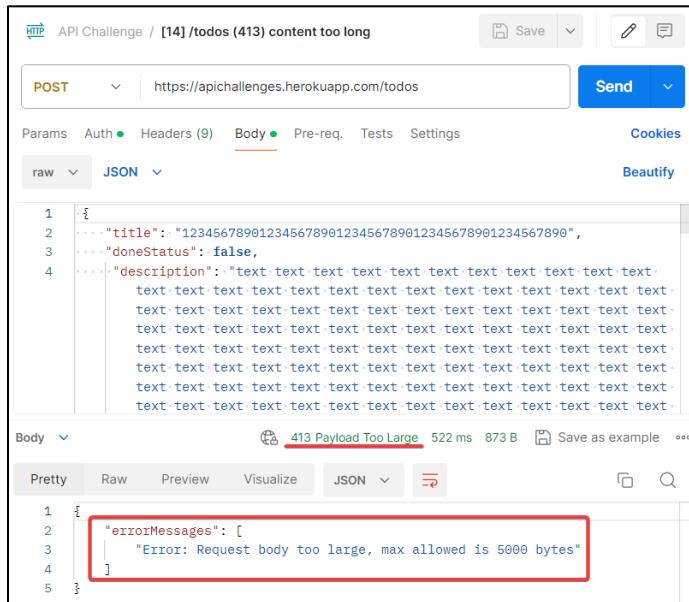
The screenshot shows a Postman interface with a POST request to <https://apichallenges.herokuapp.com/todos>. The JSON body contains a todo item with both a long title and a long description. The response is a 201 Created with a new todo object containing the same long strings.

13	POST /todos (201) max out content	true	Issue a POST request to create a todo with maximum length title and description fields. ▶ Hints
----	-----------------------------------	------	----------------------------------------------------------------------------------------------------

14. Create todo with long content

14	POST /todos (413) content too long	false	Issue a POST request to create a todo but fail payload length validation on the `description` because your whole payload exceeds maximum allowable 5000 characters. ► Hints
----	------------------------------------	-------	------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Solution:



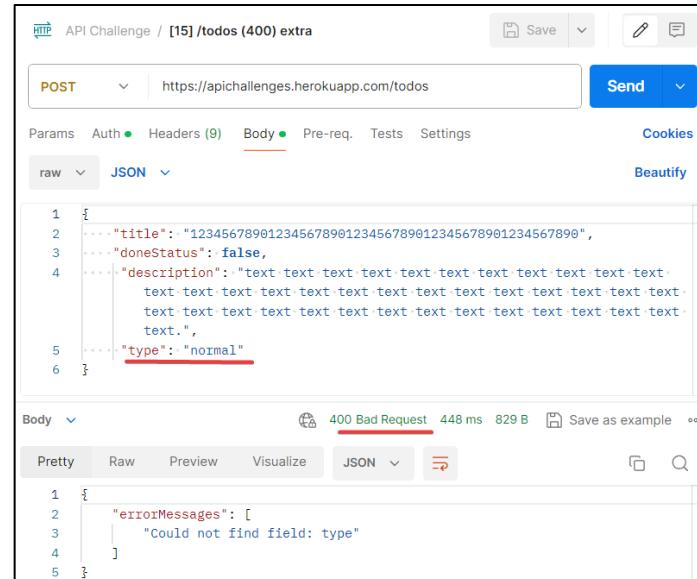
The screenshot shows the Postman interface with a POST request to `https://apichallenges.herokuapp.com/todos`. The JSON body contains a long string of text for the 'description' field. The response status is `413 Payload Too Large`, and the error message is `"Error: Request body too large, max allowed is 5000 bytes"`.

14	POST /todos (413) content too long	true	Issue a POST request to create a todo but fail payload length validation on the `description` because your whole payload exceeds maximum allowable 5000 characters. ► Hints
----	------------------------------------	------	------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

15. Create a todo record using an unrecognized field in the payload

15	POST /todos (400) extra	false	Issue a POST request to create a todo but fail validation because your payload contains an unrecognised field. ► Hints
----	-------------------------	-------	-------------------------------------------------------------------------------------------------------------------------------

Solution:



The screenshot shows the Postman interface with a POST request to `https://apichallenges.herokuapp.com/todos`. The JSON body contains a long string of text for the 'description' field and an additional field `"type": "normal"`. The response status is `400 Bad Request`, and the error message is `"Could not find field: type"`.

15	POST /todos (400) extra	true	Issue a POST request to create a todo but fail validation because your payload contains an unrecognised field. ► Hints
----	-------------------------	------	-------------------------------------------------------------------------------------------------------------------------------

16. Use a PUT request for unsuccessfully creating a todo record

Creation Challenges with PUT

A PUT request can often be used to create and update data. The todo application we are using has automatically generated IDs, so you cannot use PUT to create. As a Hint, if you are not sure what the message body should be, try copying in the response from the associated GET request, and amending it.

ID	Challenge	Done	Description
16	PUT /todos/{id} (400)	false	Issue a PUT request to unsuccessfully create a todo

Solution:

The screenshot shows the API Challenge tool interface. The URL is https://apichallenges.herokuapp.com/todos/156. The request method is PUT. The response status is 400 Bad Request, with the message "Cannot create todo with PUT due to Auto fields id".

ID	Challenge	Done	Description
16	PUT /todos/{id} (400)	true	Issue a PUT request to unsuccessfully create a todo

17. Update todo record

Update Challenges with POST

Use a POST request to amend something that already exists. These are 'partial' content updates so you usually don't need to have all details of the entity in the request, e.g. you could just update a title, or a description, or a status

ID	Challenge	Done	Description
17	POST /todos/{id} (200)	false	Issue a POST request to successfully update a todo ► Hints ► Solution

Solution:

The screenshot shows the Postman interface with the following details:

- Method: POST
- URL: https://apichallenges.herokuapp.com/todos/1
- Body tab selected, showing JSON content:

```
1 {  
2   ... "title": "16",  
3   ... "doneStatus": false,  
4   ... "description": "Lorem ipsum del amore."  
5 }
```
- Response status: 200 OK
- Response body:

```
1 {  
2   "id": 1,  
3   "title": "16",  
4   "doneStatus": false,  
5   "description": "Lorem ipsum del amore."  
6 }
```
- Panel on the right:

17	POST /todos/{id} (200)	true	Issue a POST request to successfully update a todo ▶ Hints ▶ Solution
----	------------------------	------	-----------------------------------------------------------------------------

18. Update a todo record which doesn't exist

18	POST /todos/{id} (404)	false	Issue a POST request for a todo which does not exist. Expect to receive a 404 response. ▶ Hints
----	------------------------	-------	----------------------------------------------------------------------------------------------------

Solution:

The screenshot shows the Postman interface with the following details:

- Method: POST
- URL: https://apichallenges.herokuapp.com/todos/156
- Body tab selected, showing JSON content:

```
1 {  
2   ... "title": "1",  
3   ... "doneStatus": false,  
4   ... "description": "Lorem ipsum del amore."  
5 }
```
- Response status: 404 Not Found
- Response body:

```
1 {  
2   "errorMessages": [  
3     "No such todo entity instance with id == 156 found"  
4   ]  
5 }
```
- Panel on the right:

18	POST /todos/{id} (404)	true	Issue a POST request for a todo which does not exist. Expect to receive a 404 response. ▶ Hints
----	------------------------	------	----------------------------------------------------------------------------------------------------

19. Update an existing todo record with a complete payload

Update Challenges with PUT

A PUT request can be used to amend data. REST Put requests are idempotent, they provide the same result each time.

ID	Challenge	Done	Description
19	PUT /todos/{id} full (200)	false	Issue a PUT request to update an existing todo with a complete payload i.e. title, description and donestatus.

Solution:

The screenshot shows the Postman interface with the following details:

- Method:** PUT
- URL:** https://apichallenges.herokuapp.com/todos/9
- Body (JSON):**

```
1 {
2   ... "title": "file paperworkgggggggg",
3   ... "doneStatus": true,
4   ... "description": "Lorem ipsum"
5 }
```
- Response Status:** 200 OK
- ResponseBody:**

```
1 {
2   "id": 9,
3   "title": "file paperworkgggggggg",
4   "doneStatus": true,
5   "description": "Lorem ipsum"
6 }
```

A callout box on the right side of the interface provides the following instruction:

Issue a PUT request to update an existing todo with a complete payload i.e. title, description and donestatus.

20. Update mandatory items for an existing todo record

20	PUT /todos/{id} partial (200)	false	Issue a PUT request to update an existing todo with just mandatory items in payload i.e. title.
----	-------------------------------	-------	-------------------------------------------------------------------------------------------------

Solution:

The screenshot shows the Postman interface with the following details:

- Method:** PUT
- URL:** https://apichallenges.herokuapp.com/todos/1
- Body (JSON):**

```
1 {
2   ...
3   ... "title": "file paperwork (UPDATED!)"
4 }
```
- Response Status:** 200 OK
- ResponseBody:**

```
1 {
2   "id": 1,
3   "title": "file paperwork (UPDATED!)",
4   "doneStatus": false,
5   "description": ""
6 }
```

A callout box on the right side of the interface provides the following instruction:

Issue a PUT request to update an existing todo with just mandatory items in payload i.e. title.

21. Fail to update an existing todo record

21	PUT /todos/{id} no title (400)	false	Issue a PUT request to fail to update an existing todo because title is missing in payload. ► Hints
----	--------------------------------	-------	--------------------------------------------------------------------------------------------------------

Solution:

The screenshot shows the Postman interface with a PUT request to `https://apichallenges.herokuapp.com/todos/5`. The body is set to JSON and contains the following payload:

```

1 {
2   ... "doneStatus": false,
3   ... "description": ""
4 }

```

The response status is `400 Bad Request` with a duration of `445 ms` and a size of `833 B`. The response body is:

```

1 {
2   "errorMessages": [
3     "title : field is mandatory"
4   ]
5 }

```

21	PUT /todos/{id} no title (400)	true	Issue a PUT request to fail to update an existing todo because title is missing in payload. ▶ Hints
----	--------------------------------	------	--------------------------------------------------------------------------------------------------------

22. Fail to update an existing todo

22	PUT /todos/{id} no amend id (400)	false	Issue a PUT request to fail to update an existing todo because id different in payload. ▶ Hints
----	-----------------------------------	-------	----------------------------------------------------------------------------------------------------

Solution:

The screenshot shows the Postman interface with a PUT request to `https://apichallenges.herokuapp.com/todos/5`. The body is set to JSON and contains the following payload:

```

1 {
2   ... "id": "22",
3   ... "title": "new one",
4   ... "doneStatus": false,
5   ... "description": ""
6 }

```

The response status is `400 Bad Request` with a duration of `452 ms` and a size of `836 B`. The response body is:

```

1 {
2   "errorMessages": [
3     "Can not amend id from 5 to 22"
4   ]
5 }

```

22	PUT /todos/{id} no amend id (400)	true	Issue a PUT request to fail to update an existing todo because id different in payload. ▶ Hints
----	-----------------------------------	------	----------------------------------------------------------------------------------------------------

23. Delete the todo record

DELETE Challenges

Use a DELETE request to delete an entity. Since this is an extreme request, normally you have to be logged in or authenticated, but we wanted to make life easier for you so we cover authentication later. Anyone can delete To Do items without authentication in this system.

ID	Challenge	Done	Description
23	DELETE /todos/{id} (200)	false	Issue a DELETE request to successfully delete a todo ▶ Hints ▶ Solution

Solution:

The screenshot shows the Postman interface with the following details:

- HTTP:** API Challenge / [23] /todos/{id} (200)
- Method:** DELETE
- URL:** https://apichallenges.herokuapp.com/todos/10
- Headers:** (7)
X-CHALLENGER: 49fccee6-36f7-4d4e-9bde-...
Postman-Token: <calculated when request is...
Host: <calculated when request is...
User-Agent: PostmanRuntime/7.37.0
Accept: */*
Accept-Encoding: gzip, deflate, br
- Body:** (200 OK, 398 ms, 780 B, Save as example)
- Response Headers:** (1)
Allow: OPTIONS, GET, HEAD, POST

Table (Challenge 23):

23	DELETE /todos/{id} (200)	true	Issue a DELETE request to successfully delete a todo ► Hints ► Solution
----	--------------------------	------	-------------------------------------------------------------------------------

24. Get an 'Allow' header in the OPTION request

OPTIONS Challenges

Use an OPTIONS verb and check the `Allow` header, this will show you what verbs are allowed to be used on an endpoint. When you test APIs it is worth checking to see if all the verbs listed are allowed or not.

ID	Challenge	Done	Description
24	OPTIONS / todos (200)	false	Issue an OPTIONS request on the `/todos` end point. You might want to manually check the 'Allow' header in the response is as expected. ► Solution

Solution:

The screenshot shows the Postman interface with the following details:

- HTTP:** API Challenge / [24] /todos (200)
- Method:** OPTIONS
- URL:** https://apichallenges.herokuapp.com/todos
- Headers:** (7)
X-CHALLENGER: 49fccee6-36f7-4d4e-9bde-...
Postman-Token: <calculated when request is...
Host: <calculated when request is...
User-Agent: PostmanRuntime/7.37.0
Accept: */*
- Body:** (200 OK, 642 ms, 812 B, Save as example)
- Response Headers:** (1)
Report-To: {"group":"heroku-nei","max_age":3600,"endpoint...
Reporting-Endpoints: heroku-nei=https://nei.herokuapp.com/reports?ts=1...
Nel: {"report_to":"heroku-nei","max_age":3600,"succes...
Connection: close
Date: Wed, 20 Mar 2024 13:05:17 GMT
Allow: OPTIONS, GET, HEAD, POST
Access-Control-Allow-Origin: *

Table (Challenge 24):

24	OPTIONS /todos (200)	true	Issue an OPTIONS request on the `/todos` end point. You might want to manually check the 'Allow' header in the response is as expected. ► Solution
----	----------------------	------	-------------------------------------------------------------------------------------------------------------------------------------------------------

25. Receive a result in XML format

Accept Challenges

The `Accept` header, tells the server what format you want the response to be in. By changing the `Accept` header you can specify JSON or XML.

ID	Challenge	Done	Description
25	GET /todos (200) XML	false	<p>Issue a GET request on the `/todos` end point with an `Accept` header of `application/xml` to receive results in XML format</p> <p>► Solution</p>

Solution:

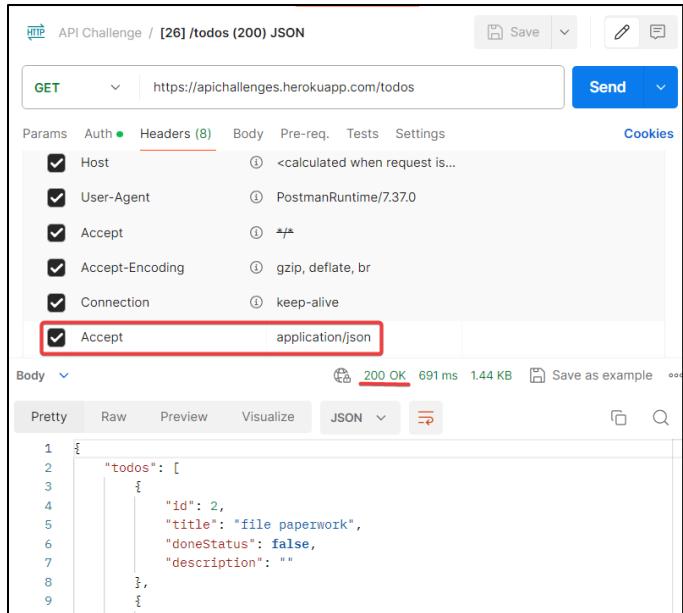
The screenshot shows a Postman request configuration for challenge 25. The method is set to GET, and the URL is https://apichallenges.herokuapp.com/todos. In the Headers tab, the 'Accept' header is explicitly defined as 'application/xml'. The response status is 200 OK, and the body is displayed in XML format, showing a single todo item with id 2 and title 'file paperwork'.

25	GET /todos (200) XML	true	<p>Issue a GET request on the `/todos` end point with an `Accept` header of `application/xml` to receive results in XML format</p> <p>► Solution</p>
----	-------------------------	------	------------------------------------------------------------------------------------------------------------------------------------------------------

26. Receive a result in JSON format

26	GET /todos (200) JSON	false	<p>Issue a GET request on the `/todos` end point with an `Accept` header of `application/json` to receive results in JSON format</p> <p>► Solution</p>
----	--------------------------	-------	--------------------------------------------------------------------------------------------------------------------------------------------------------

Solution:



The screenshot shows the Postman interface with a GET request to `https://apichallenges.herokuapp.com/todos`. The 'Headers' tab is selected, showing the following configuration:

- Host
- User-Agent
- Accept: `application/json` (highlighted with a red box)
- Accept-Encoding
- Connection

The response status is 200 OK, and the JSON body is displayed in the preview tab:

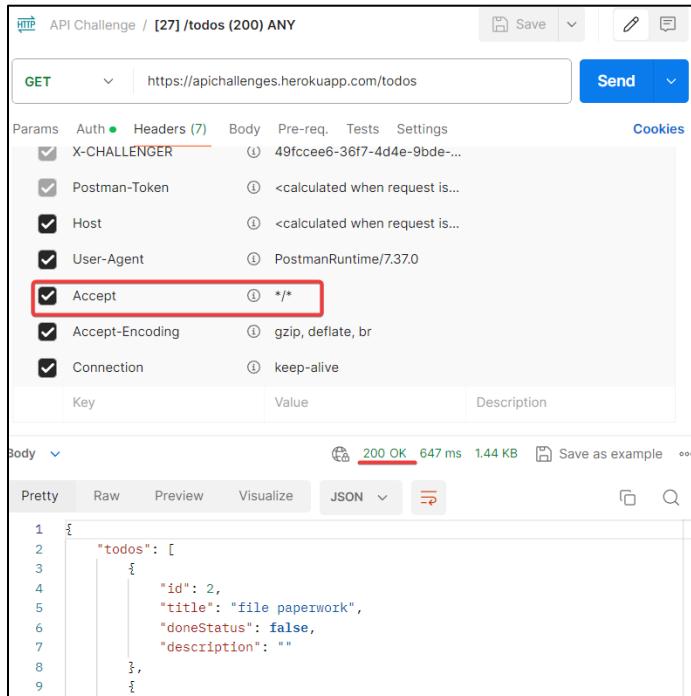
```
1 {
2   "todos": [
3     {
4       "id": 2,
5       "title": "file paperwork",
6       "doneStatus": false,
7       "description": ""
8     }
9   ]
}
```

26	GET /todos (200) JSON	true	Issue a GET request on the '/todos' end point with an 'Accept' header of 'application/json' to receive results in JSON format ► Solution
----	-----------------------	------	---------------------------------------------------------------------------------------------------------------------------------------------

27. Receive a result in default JSON format

27	GET /todos (200) ANY	false	Issue a GET request on the '/todos' end point with an 'Accept' header of '*'/* to receive results in default JSON format ► Solution
----	----------------------	-------	----------------------------------------------------------------------------------------------------------------------------------------

Solution:



The screenshot shows the Postman interface with a GET request to `https://apichallenges.herokuapp.com/todos`. The 'Headers' tab is selected, showing the following configuration:

- X-CHALLENGER
- Postman-Token
- Host
- User-Agent
- Accept: `*/*` (highlighted with a red box)
- Accept-Encoding
- Connection

The response status is 200 OK, and the JSON body is displayed in the preview tab:

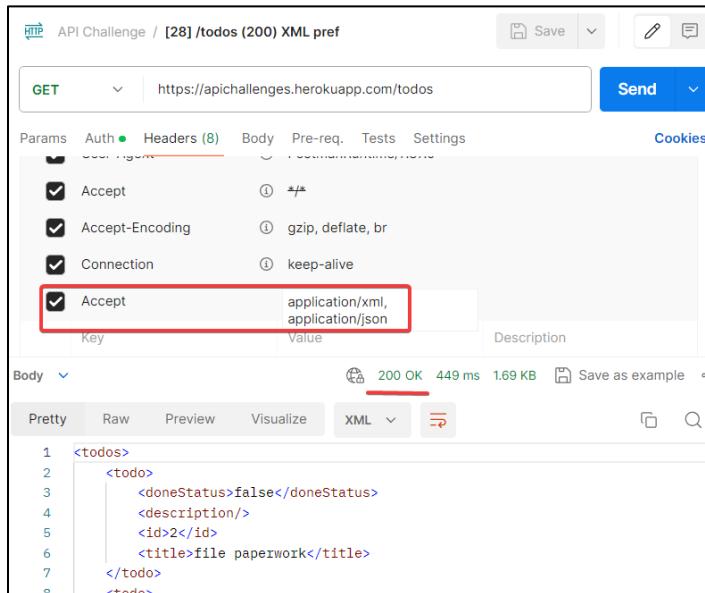
```
1 {
2   "todos": [
3     {
4       "id": 2,
5       "title": "file paperwork",
6       "doneStatus": false,
7       "description": ""
8     }
9   ]
}
```

27	GET /todos (200) ANY	true	Issue a GET request on the '/todos' end point with an 'Accept' header of '*'/* to receive results in default JSON format ► Solution
----	----------------------	------	----------------------------------------------------------------------------------------------------------------------------------------

28. Receive a result in the XML format

28	GET /todos (200) XML pref	false	Issue a GET request on the `/todos` end point with an `Accept` header of `application/xml, application/json` to receive results in the preferred XML format ► Solution
----	------------------------------	-------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Solution:



The screenshot shows the Postman interface for challenge 28. In the 'Headers' tab, the 'Accept' header is explicitly defined as 'application/xml, application/json'. The 'Body' tab displays the XML response received from the API.

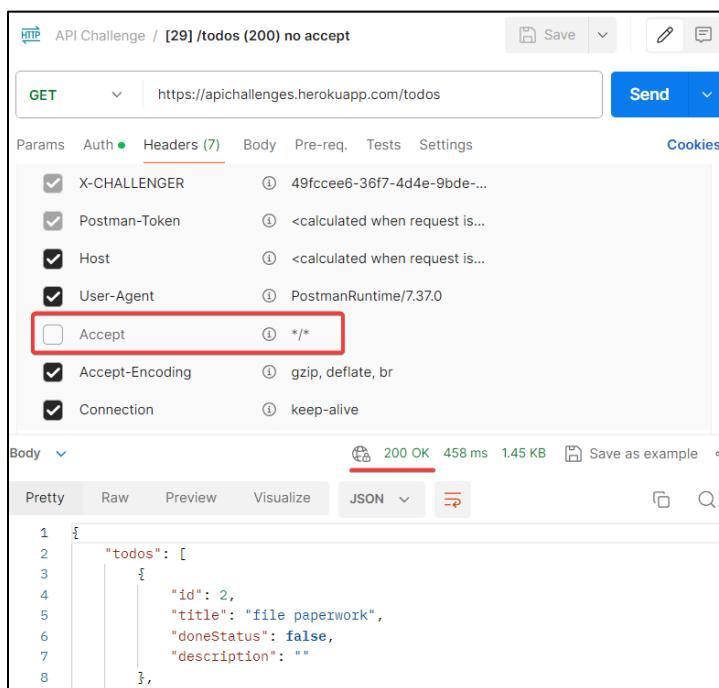
```
<todos>
<todo>
<doneStatus>false</doneStatus>
<description/>
<id>2</id>
<title>file paperwork</title>
</todo>
</todos>
```

28	GET /todos (200) XML pref	true	Issue a GET request on the `/todos` end point with an `Accept` header of `application/xml, application/json` to receive results in the preferred XML format ► Solution
----	------------------------------	------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

29. Receive results in default JSON format

29	GET /todos (200) no accept	false	Issue a GET request on the `/todos` end point with no `Accept` header present in the message to receive results in default JSON format ► Solution
----	-------------------------------	-------	----------------------------------------------------------------------------------------------------------------------------------------------------------

Solution:



The screenshot shows the Postman interface for challenge 29. In the 'Headers' tab, the 'Accept' header is removed from the list of headers. The 'Body' tab displays the JSON response received from the API.

```
{
  "todos": [
    {
      "id": 2,
      "title": "file paperwork",
      "doneStatus": false,
      "description": ""
    }
  ]
}
```

29	GET /todos (200) no accept	true	Issue a GET request on the `/todos` end point with no `Accept` header present in the message to receive results in default JSON format ► Solution
----	-------------------------------	------	----------------------------------------------------------------------------------------------------------------------------------------------------------

30. Create a GET request with an ‘Accept’ header ‘application/gzip’

30	GET /todos (406)	false	Issue a GET request on the `/todos` end point with an `Accept` header `application/gzip` to receive 406 'NOT ACCEPTABLE' status code ► Solution
----	---------------------	-------	--------------------------------------------------------------------------------------------------------------------------------------------------------

Solution:

The screenshot shows the Postman interface with a GET request to the URL `https://apichallenges.herokuapp.com/todos`. In the Headers tab, the 'Accept' header is explicitly set to `application/gzip`. The response status is 406 Not Acceptable, and the error message in the body is `"errorMessages": ["Unrecognised Accept Type"]`.

30	GET /todos (406)	true	Issue a GET request on the `/todos` end point with an `Accept` header `application/gzip` to receive 406 'NOT ACCEPTABLE' status code ► Solution
----	---------------------	------	--------------------------------------------------------------------------------------------------------------------------------------------------------

31. Create a todo record using Content-Type ‘application/xml’

Content-Type Challenges

The ‘Content-Type’ header, tells the server what format type your ‘body’ content is, e.g. are you sending XML or JSON.

ID	Challenge	Done	Description
31	POST /todos XML	false	Issue a POST request on the `/todos` end point to create a todo using Content-Type ‘application/xml’, and Accepting only XML ie. Accept header of ‘application/xml’ ► Solution

Solution:

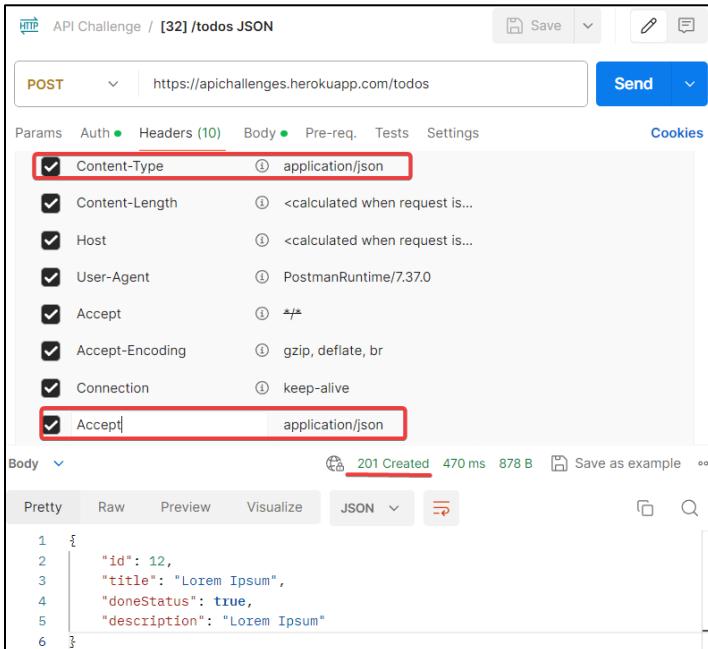
The screenshot shows the Postman interface with a POST request to the URL `https://apichallenges.herokuapp.com/todos`. In the Headers tab, both the 'Content-Type' and 'Accept' headers are set to `application/xml`. The response status is 201 Created, and the body contains XML data representing a new todo item.

31	POST /todos XML	true	Issue a POST request on the `/todos` end point to create a todo using Content-Type ‘application/xml’, and Accepting only XML ie. Accept header of ‘application/xml’ ► Solution
----	--------------------	------	---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

32. Create POST request using Content-Type ‘application/json’

32	POST /todos JSON	false	Issue a POST request on the '/todos' end point to create a todo using Content-Type 'application/json', and Accepting only JSON ie. Accept header of 'application/json' ► Solution
----	---------------------	-------	------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Solution:



The screenshot shows the Postman interface with the following details:

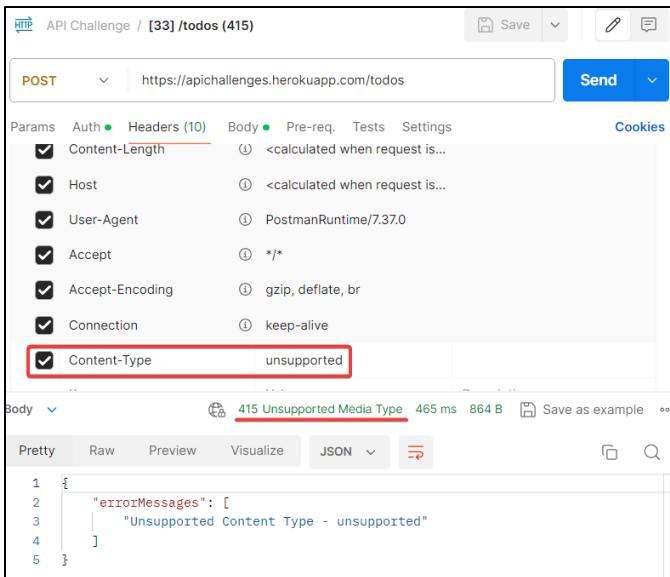
- Method: POST
- URL: https://apichallenges.herokuapp.com/todos
- Headers:
 - Content-Type: application/json
 - Accept: application/json
- Body:
 - Pretty: {
1 "id": 12,
2 "title": "Lorem Ipsum",
3 "doneStatus": true,
4 "description": "Lorem Ipsum"
5 }
 - Raw: {"id": 12, "title": "Lorem Ipsum", "doneStatus": true, "description": "Lorem Ipsum"}
- Response status: 201 Created

32	POST /todos JSON	true	Issue a POST request on the '/todos' end point to create a todo using Content-Type 'application/json', and Accepting only JSON ie. Accept header of 'application/json' ► Solution
----	---------------------	------	------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

33. Create a request with an unsupported content type

33	POST /todos (415)	false	Issue a POST request on the '/todos' end point with an unsupported content type to generate a 415 status code ► Solution
----	----------------------	-------	---------------------------------------------------------------------------------------------------------------------------------

Solution:



The screenshot shows the Postman interface with the following details:

- Method: POST
- URL: https://apichallenges.herokuapp.com/todos
- Headers:
 - Content-Length
 - Host
 - User-Agent: PostmanRuntime/7.37.0
 - Accept: */*
 - Accept-Encoding: gzip, deflate, br
 - Connection: keep-alive
 - Content-Type: unsupported
- Body:
 - Pretty: {
1 "errorMessages": [
2 | "Unsupported Content Type - unsupported"
3]
4 }
Raw: {"errorMessages": ["Unsupported Content Type - unsupported"]}
 - Raw: {"errorMessages": ["Unsupported Content Type - unsupported"]}
- Response status: 415 Unsupported Media Type

33	POST /todos (415)	true	Issue a POST request on the '/todos' end point with an unsupported content type to generate a 415 status code ► Solution
----	----------------------	------	---------------------------------------------------------------------------------------------------------------------------------

34. Return the progress data payload for current challenger GUID

Fancy a Break? Restore your session

Your challenge progress can be saved, and as long as you remember you challenger ID you can restore it. Leaving a challenger idle in the system for more than 10 minutes will remove the challenger from memory. Challenger status and the todos database can be saved to, and restored from, the browser localStorage.

ID	Challenge	Done	Description
34	GET /challenger/guid (existing X-CHALLENGER)	false	<p>Issue a GET request on the `/challenger/{guid}` end point, with an existing challenger GUID. This will return the progress data payload that can be used to later restore your progress to this status.</p> <p>► Hints</p>

Solution:

The screenshot shows a Postman collection named "API Challenge". A GET request is made to the URL `https://apichallenges.herokuapp.com/challenger/49fccee6-36f7-4d4e-9bde-75b4c2790f59`. The response status is 200 OK, and the JSON payload is displayed in the body:

```

1 {
2   "xAuthToken": "2f3766c1-0b84-4264-bddc-90f63dea8ecd",
3   "xChallenger": "49fccee6-36f7-4d4e-9bde-75b4c2790f59",
4   "secretNote": "Lorem Ipsum",
5   "challengeStatus": {
6     "GET_RESTORABLE_TODOS": true,
7     "GET_ACCEPT_ANY_DEFAULT_JSON": true,
8     "CREATE_SECRET_TOKEN_201": true,
9     "PUT_RESTORABLE_CHALLENGER_PROGRESS_STATUS": true,
10    "OVERRIDE_PATCH_HEARTBEAT_500": true,
11    "POST_TODOS_TOO_LONG_PAYLOAD_SIZE": true,
12    "PATCH_HEARTBEAT_500": true,
13    "POST_TODOS_BAD_DONE_STATUS": true,
14    "GET_SECRET_NOTE_401": true,
15    ...
16  }

```

34	GET /challenger/guid (existing X-CHALLENGER)	true	<p>Issue a GET request on the `/challenger/{guid}` end point, with an existing challenger GUID. This will return the progress data payload that can be used to later restore your progress to this status.</p> <p>► Hints</p>
----	----------------------------------------------	------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

35. Restore the challenger's progress

35	PUT /challenger/guid RESTORE	false	<p>Issue a PUT request on the `/challenger/{guid}` end point, with an existing challenger GUID to restore that challenger's progress into memory.</p> <p>► Hints</p> <p>► Solution</p>
----	------------------------------	-------	----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Solution:

The screenshot shows a Postman collection named "API Challenge". A PUT request is made to the URL `https://apichallenges.herokuapp.com/challenger/49fccee6-36f7-4d4e-9bde-75b4c2790f59`. The response status is 200 OK, and the JSON payload is displayed in the body:

```

1 {
2   ...
3   "POST_TODOS_INVALID_EXTRA_FIELD": false,
4   ...
5 }

```

35	PUT /challenger/guid RESTORE	true	<p>Issue a PUT request on the `/challenger/{guid}` end point, with an existing challenger GUID to restore that challenger's progress into memory.</p> <p>► Hints</p> <p>► Solution</p>
----	------------------------------	------	----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

36. Restore challenger's progress, when challenger GUID is not currently in memory

36	PUT /challenger/guid CREATE	false	<p>Issue a PUT request on the '/challenger/{guid}' end point, with a challenger GUID not currently in memory to restore that challenger's progress into memory.</p> <ul style="list-style-type: none"> ▶ Hints ▶ Solution
----	--------------------------------	-------	---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Solution:

The screenshot shows the API Challenge tool interface. A PUT request is made to the URL <https://apichallenges.herokuapp.com/challenger/49fccee6-36f7-4d4e-9bde-75b4c2790f59>. The response is a 200 OK status with a response time of 488 ms and a size of 2.66 KB. The JSON response body is identical to the request body, indicating a successful restore operation.

```

1 {
2   "xAuthToken": "2f3766c1-0b84-4264-bddc-90f63dea8ecd",
3   "xChallenger": "49fccee6-36f7-4d4e-9bde-75b4c2790f59",
4   "secretNote": "",
5   "challengeStatus": {
6     "POST_TODOS_INVALID_EXTRA_FIELD": false,
7     "GET_HEARTBEAT_204": false,
8     "POST_SECRET_NOTE_403": false,
9     "GET_ACCEPT_XML_PREFERRED": false,
10    "POST_MAX_OUT_TITLE_DESCRIPTION_LENGTH": false,
11    "POST_TODOS_TOO_LONG_PAYLOAD_SIZE": false,
12    "GET_ACCEPT_ANY_DEFAULT_JSON": true
13  }
14}

```

36	PUT /challenger/guid CREATE	true	<p>Issue a PUT request on the '/challenger/{guid}' end point, with a challenger GUID not currently in memory to restore that challenger's progress into memory.</p> <ul style="list-style-type: none"> ▶ Hints ▶ Solution
----	--------------------------------	------	---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

37. Retrieve the current todos database for the user.

37	GET /challenger/ database/guid (200)	false	<p>Issue a GET request on the '/challenger/database/{guid}' end point, to retrieve the current todos database for the user. You can use this to restore state later.</p> <ul style="list-style-type: none"> ▶ Hints
----	-----------------------------------------	-------	------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Solution:

The screenshot shows the API Challenge tool interface. A GET request is made to the URL <https://apichallenges.herokuapp.com/challenger/database/49fccee6-36f7-4d4e-9bde-75b4c2790f59>. The response is a 200 OK status with a response time of 446 ms and a size of 1.37 KB. The JSON response body contains a list of todos, indicating a successful retrieval of the current todos database.

```

1 {
2   "todos": [
3     {
4       "id": 2,
5       "title": "file paperwork"
6     },
7     {
8       "id": 3,
9       "title": "process payments"
10    },
11    ...
12  ]
13}

```

37	GET /challenger/database/guid (200)	true	<p>Issue a GET request on the '/challenger/database/{guid}' end point, to retrieve the current todos database for the user. You can use this to restore state later.</p> <ul style="list-style-type: none"> ▶ Hints
----	-------------------------------------------	------	------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

38. Restore the todo database in memory

38	PUT /challenger/database/guid (Update)	false	Issue a PUT request on the `/challenger/database/{guid}` end point, with a payload to restore the Todos database in memory. ► Hints ► Solution
----	----------------------------------------	-------	----------------------------------------------------------------------------------------------------------------------------------------------------------

Solution:

The screenshot shows a Postman request for challenge 38. The method is PUT, the URL is <https://apichallenges.herokuapp.com/challenger/database/49fcc...>, and the body contains the following JSON payload:

```

1 {
2   "todos": [
3     {
4       "id": 1,
5       "title": "scan paperwork"
6     },
7     {
8       "id": 4,
9       "title": "escalate late payments"
10    },
11   ]
12 }
  
```

The response status is 204 No Content, and the response body is empty.

38	PUT /challenger/database/guid (Update)	true	Issue a PUT request on the `/challenger/database/{guid}` end point, with a payload to restore the Todos database in memory. ► Hints ► Solution
----	----------------------------------------	------	----------------------------------------------------------------------------------------------------------------------------------------------------------

39. Create a todo using Content-Type ‘application/xml’, but Accept ‘application/json’

Mix Accept and Content-Type Challenges

We can mix the ‘Accept’ and ‘Content-Type’ headers so that we can send JSON but receive XML. These challenges encourage you to explore some combinations.

ID	Challenge	Done	Description
39	POST /todos XML to JSON	false	Issue a POST request on the `/todos` end point to create a todo using Content-Type ‘application/xml’ but Accept ‘application/json’ ► Solution

Solution:

The screenshot shows a Postman request for challenge 39. The method is POST, the URL is <https://apichallenges.herokuapp.com/todos>, and the body contains the following XML payload:

```

1 <todo>
2   <doneStatus>false</doneStatus>
3   <description>Trust yourself</description>
4   <title>Tre</title>
5 </todo>
  
```

The response status is 201 Created, and the response body is a JSON object:

```

1 {
2   "id": 13,
3   "title": "Tre",
4   "doneStatus": false,
5   "description": "Trust yourself"
6 }
  
```

39	POST /todos XML to JSON	true	Issue a POST request on the `/todos` end point to create a todo using Content-Type ‘application/xml’ but Accept ‘application/json’ ► Solution
----	-------------------------	------	------------------------------------------------------------------------------------------------------------------------------------------------------

40. Create a todo using Content-Type 'application/json', but Accept 'application/xml'

40	POST /todos JSON to XML	false	Issue a POST request on the `/todos` end point to create a todo using Content-Type `application/json` but Accept `application/xml` ► Solution
----	----------------------------	-------	------------------------------------------------------------------------------------------------------------------------------------------------------

Solution:

The screenshot shows the Postman interface with a successful POST request to `https://apichallenges.herokuapp.com/todos`. The JSON body is set to `{\"title\": \"Titlee free\", \"doneStatus\": true, \"description\": \"Some text\"}`. The response status is `201 Created` with a time of `457 ms` and a size of `915 B`. The response body is displayed in XML format, showing the created todo item.

40	POST /todos JSON to XML	true	Issue a POST request on the `/todos` end point to create a todo using Content-Type `application/json` but Accept `application/xml` ► Solution
----	----------------------------	------	------------------------------------------------------------------------------------------------------------------------------------------------------

41. Receive 405 status code in DELETE request

Status Code Challenges

Status-codes are essential to understand, so we created some challenges that help you trigger more status codes. Remember to review httpstatuses.com to learn what the status codes mean.

ID	Challenge	Done	Description
41	DELETE /heartbeat (405)	false	Issue a DELETE request on the `/heartbeat` end point and receive 405 (Method Not Allowed) ► Solution

Solution:

The screenshot shows the Postman interface with a failed DELETE request to `https://apichallenges.herokuapp.com/heartbeat`. The response status is `405 Method Not Allowed` with a time of `724 ms` and a size of `793 B`. The response body is `405 Method Not Allowed`.

41	DELETE /heartbeat (405)	true	Issue a DELETE request on the `/heartbeat` end point and receive 405 (Method Not Allowed) ► Solution
----	-------------------------	------	-------------------------------------------------------------------------------------------------------------

42. Receive 500 status code in the PATCH request

42	PATCH /heartbeat (500)	false	Issue a PATCH request on the `/heartbeat` end point and receive 500 (internal server error) ► Solution
----	---------------------------	-------	-----------------------------------------------------------------------------------------------------------

Solution:

The screenshot shows a Postman request configuration for a PATCH operation. The URL is https://apichallenges.herokuapp.com/heartbeat. The Headers tab is selected, showing the following values:

- X-CHALLENGER: 49fccee6-36f7-4d4e-9bde-...
- Postman-Token: <calculated when request is...
- Content-Length: 0
- Host: <calculated when request is...
- User-Agent: PostmanRuntime/7.37.0
- Accept: */*
- Accept-Encoding: gzip, deflate, br
- Connection: keep-alive

The Body tab shows a response with a status of 500 Server Error, 636 ms, and 787 B. The response content is partially visible as "1".

42	PATCH /heartbeat (500)	true	Issue a PATCH request on the `/heartbeat` end point and receive 500 (internal server error) ► Solution
----	---------------------------	------	-----------------------------------------------------------------------------------------------------------

43. Receive 501 status code in TRACE request

43	TRACE /heartbeat (501)	false	Issue a TRACE request on the `/heartbeat` end point and receive 501 (Not Implemented) ► Solution
----	---------------------------	-------	-----------------------------------------------------------------------------------------------------

Solution:

The screenshot shows a Postman request configuration for a TRACE operation. The URL is https://apichallenges.herokuapp.com/heartbeat. The Headers tab is selected, showing the following values:

- X-CHALLENGER: 49fccee6-36f7-4d4e-9bde-...
- Postman-Token: <calculated when request is...
- Content-Length: 0
- Host: <calculated when request is...
- User-Agent: PostmanRuntime/7.37.0
- Accept: */*
- Accept-Encoding: gzip, deflate, br
- Connection: keep-alive

The Body tab shows a response with a status of 501 Not Implemented, 568 ms, and 794 B. The response content is partially visible as "1".

43	TRACE /heartbeat (501)	true	Issue a TRACE request on the `/heartbeat` end point and receive 501 (Not Implemented) ► Solution
----	---------------------------	------	-----------------------------------------------------------------------------------------------------

44. Receive 204 status code when the server is running

44	GET /heartbeat (204)	false	Issue a GET request on the `/heartbeat` end point and receive 204 when server is running ► Solution
----	-------------------------	-------	--------------------------------------------------------------------------------------------------------

Solution:

The screenshot shows the Postman interface with a GET request to the URL `https://apichallenges.herokuapp.com/heartbeat`. The response status is 204 No Content, indicating the server is running.

44	GET /heartbeat (204)	true	Issue a GET request on the `/heartbeat` end point and receive 204 when server is running ► Solution
----	----------------------	------	--------------------------------------------------------------------------------------------------------

45. Create a POST request and receive 405 status code

HTTP Method Override Challenges

Some HTTP Clients can not send all verbs e.g. PATCH, DELETE, PUT. Use an X-HTTP-Method-Override header to simulate these with a POST request

ID	Challenge	Done	Description
45	POST /heartbeat as DELETE (405)	false	Issue a POST request on the `/heartbeat` end point and receive 405 when you override the Method Verb to a DELETE ► Hints ► Solution

Solution:

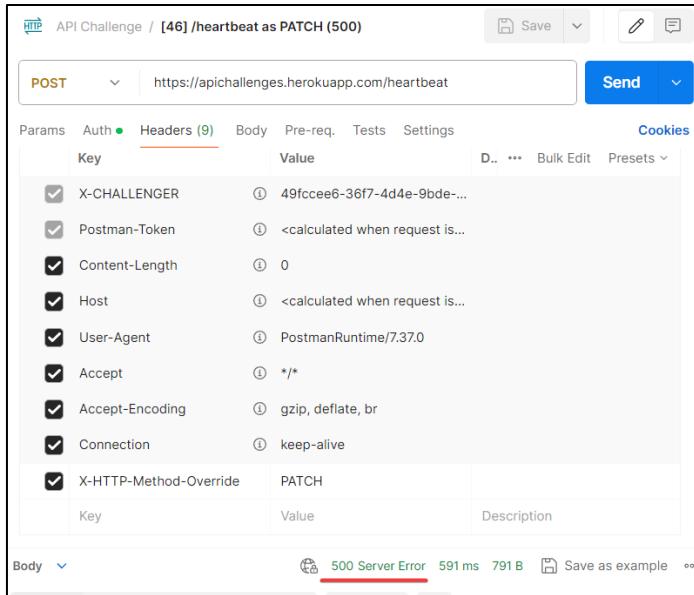
The screenshot shows the Postman interface with a POST request to the URL `https://apichallenges.herokuapp.com/heartbeat`. The X-HTTP-Method-Override header is set to DELETE. The response status is 405 Method Not Allowed.

45	POST /heartbeat as DELETE (405)	true	Issue a POST request on the `/heartbeat` end point and receive 405 when you override the Method Verb to a DELETE ► Hints ► Solution
----	---------------------------------	------	-------------------------------------------------------------------------------------------------------------------------------------------

46. Receive 500 status code when override the Method Verb to a PATCH

46	POST /heartbeat as PATCH (500)	false	Issue a POST request on the `/heartbeat` end point and receive 500 when you override the Method Verb to a PATCH ► Hints ► Solution
----	--------------------------------	-------	------------------------------------------------------------------------------------------------------------------------------------------

Solution:



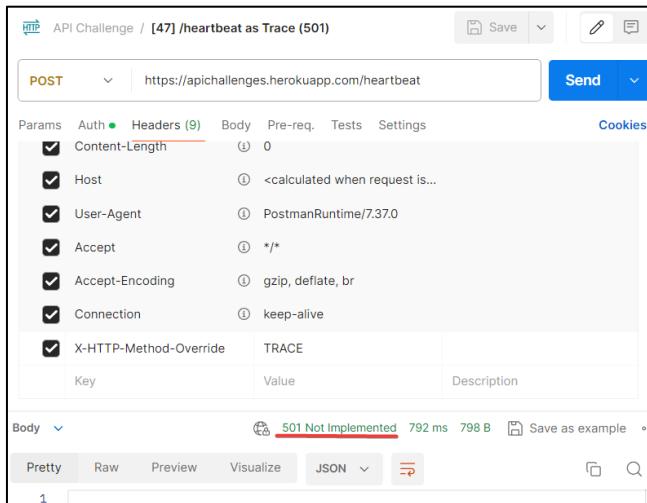
The screenshot shows the Postman interface with a POST request to `https://apichallenges.herokuapp.com/heartbeat`. The `X-HTTP-Method-Override` header is set to `PATCH`. The response status is `500 Server Error`.

46	POST /heartbeat as PATCH (500)	true	Issue a POST request on the `/heartbeat` end point and receive 500 when you override the Method Verb to a PATCH ► Hints ► Solution
----	--------------------------------	------	------------------------------------------------------------------------------------------------------------------------------------------

47. Receive 500 status code when override the Method Verb to a TRACE

47	POST /heartbeat as Trace (501)	false	Issue a POST request on the `/heartbeat` end point and receive 501 (Not Implemented) when you override the Method Verb to a TRACE ► Hints ► Solution
----	--------------------------------	-------	------------------------------------------------------------------------------------------------------------------------------------------------------------

Solution:



The screenshot shows the Postman interface with a POST request to `https://apichallenges.herokuapp.com/heartbeat`. The `X-HTTP-Method-Override` header is set to `TRACE`. The response status is `501 Not implemented`.

47	POST /heartbeat as Trace (501)	true	Issue a POST request on the `/heartbeat` end point and receive 501 (Not Implemented) when you override the Method Verb to a TRACE ► Hints ► Solution
----	--------------------------------	------	------------------------------------------------------------------------------------------------------------------------------------------------------------

48. Receive 401 status code when Basic auth username/password is not admin/password

Authentication Challenges

Authentication is telling the system who you are. In multi-user mode you are already doing that with the X-CHALLENGER header, but we have added an extra level of security on the /secret section. So first Authenticate with Basic Authentication to find out the token to use for authorisation for later challenges.

ID	Challenge	Done	Description
48	POST /secret/token (401)	false	<p>Issue a POST request on the `/secret/token` end point and receive 401 when Basic auth username/password is not admin/password</p> <ul style="list-style-type: none"> ▶ Hints ▶ Solution

Solution:

The screenshot shows the Postman interface with the following details:

- Method:** POST
- URL:** https://apichallenges.herokuapp.com/secret/token
- Type:** Basic Auth
- Headers:** (9 items)
- Body:** (Raw JSON, empty)
- Response Status:** 401 Unauthorized

48	POST /secret/token (401)	true	<p>Issue a POST request on the `/secret/token` end point and receive 401 when Basic auth username/password is not admin/password</p> <ul style="list-style-type: none"> ▶ Hints ▶ Solution
----	--------------------------------	------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

49. Receive 201 status code when Basic auth has admin/password values for username/password fields

49	POST /secret/ token (201)	false	<p>Issue a POST request on the `/secret/token` end point and receive 201 when Basic auth username/password is admin/password</p> <ul style="list-style-type: none"> ▶ Hints ▶ Solution
----	------------------------------	-------	----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Solution:

The screenshot shows the Postman interface with a successful API call. The request method is POST, and the URL is https://apichallenges.herokuapp.com/secret/token. The response status is 201 Created, and the response body is {"token": "201"}. The 'Auth' tab is selected, and the 'Basic Auth' type is chosen. The 'Username' field contains 'admin' and the 'Password' field contains 'password'.

49	POST /secret/token (201)	true	Issue a POST request on the `/secret/token` end point and receive 201 when Basic auth username/password is admin/password ► Hints ► Solution
----	--------------------------	------	----------------------------------------------------------------------------------------------------------------------------------------------------

50. Receive 403 status code when X-AUTH-TOKEN doesn't match a valid token

Authorization Challenges

Once the system knows who you are, authorization is if you have the correct level of access. In these challenges the authorization is granted using a custom API header X-AUTH-TOKEN or using a Bearer Authorization header.

ID	Challenge	Done	Description
50	GET /secret/note (403)	false	Issue a GET request on the `/secret/note` end point and receive 403 when X-AUTH-TOKEN does not match a valid token ► Hints ► Solution

Solution:

The screenshot shows the Postman interface with a failed API call. The request method is GET, and the URL is https://apichallenges.herokuapp.com/secret/note. The response status is 403 Forbidden, and the response body is {"error": "Forbidden"}. The 'Headers' tab is selected, and the 'X-AUTH-TOKEN' header is listed with the value 'key'. This header is highlighted with a red box.

50	GET /secret/note (403)	true	Issue a GET request on the `/secret/note` end point and receive 403 when X-AUTH-TOKEN does not match a valid token ► Hints ► Solution
----	------------------------	------	---------------------------------------------------------------------------------------------------------------------------------------------

51. Receive 401 when no X-AUTH-TOKEN header present

51	GET /secret/note (401)	false	Issue a GET request on the `/secret/note` end point and receive 401 when no X-AUTH-TOKEN header present ► Hints ► Solution
----	------------------------	-------	----------------------------------------------------------------------------------------------------------------------------------

Solution:

The screenshot shows a Postman API challenge for task 51. The request method is GET, and the URL is https://apichallenges.herokuapp.com/secret/note. The Headers tab is selected, showing several headers including X-CHALLENGER, Postman-Token, Host, User-Agent, Accept, Accept-Encoding, and Connection. The response status is 401 Unauthorized, and the response body is empty.

51	GET /secret/note (401)	true	Issue a GET request on the `/secret/note` end point and receive 401 when no X-AUTH-TOKEN header present ► Hints ► Solution
----	------------------------	------	----------------------------------------------------------------------------------------------------------------------------------

52. Receive 200 status code when valid X-AUTH-TOKEN used – response body should contain the note

52	GET /secret/note (200)	false	Issue a GET request on the `/secret/note` end point receive 200 when valid X-AUTH-TOKEN used - response body should contain the note ► Hints ► Solution
----	------------------------	-------	---------------------------------------------------------------------------------------------------------------------------------------------------------------

Solution:

The screenshot shows a Postman API challenge for task 52. The request method is GET, and the URL is https://apichallenges.herokuapp.com/secret/note. The Headers tab is selected, showing Postman-Token, Host, User-Agent, Accept, Accept-Encoding, Connection, X-Challenger (with value 49fccee6-36f7-4d4e-9bde-...), and X-AUTH-TOKEN (with value 2f3766c1-0b84-4264-bddc...). The response status is 200 OK, and the response body is a JSON object with a note key containing "Lorem Ipsum".

52	GET /secret/note (200)	true	Issue a GET request on the `/secret/note` end point receive 200 when valid X-AUTH-TOKEN used - response body should contain the note ► Hints ► Solution
----	------------------------	------	---------------------------------------------------------------------------------------------------------------------------------------------------------------

53. Receive 200 status code when valid X-AUTH-TOKEN used.

53	POST /secret/note (200)	false	<p>Issue a POST request on the `/secret/note` end point with a note payload e.g. {"note":"my note"} and receive 200 when valid X-AUTH-TOKEN used. Note is maximum length 100 chars and will be truncated when stored.</p> <p>► Hints ► Solution</p>
----	-------------------------	-------	---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Solution:

The screenshot shows a Postman API challenge for task 53. The request method is POST, URL is https://apichallenges.herokuapp.com/secret/note, and the body is a JSON object with a single key "note" and value "Lorem Ipsum Lore Amore". The response status is 200 OK, time taken is 451 ms, and size is 805 B. The response body is identical to the request body.

53	POST /secret/note (200)	true	<p>Issue a POST request on the `/secret/note` end point with a note payload e.g. {"note":"my note"} and receive 200 when valid X-AUTH-TOKEN used. Note is maximum length 100 chars and will be truncated when stored.</p> <p>► Hints ► Solution</p>
----	-------------------------	------	---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

54. Receive 401 when no X-AUTH-TOKEN present

54	POST /secret/note (401)	false	<p>Issue a POST request on the `/secret/note` end point with a note payload {"note":"my note"} and receive 401 when no X-AUTH-TOKEN present</p> <p>► Hints ► Solution</p>
----	-------------------------	-------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Solution:

The screenshot shows a Postman API challenge for task 54. The request method is POST, URL is https://apichallenges.herokuapp.com/secret/note, and the headers section shows several auto-generated headers including X-CHALLENGER, Postman-Token, Content-Type, Content-Length, Host, User-Agent, Accept, Accept-Encoding, and Connection. The response status is 401 Unauthorized, time taken is 466 ms, and size is 790 B. The response body is "401 Unauthorized".

54	POST /secret/note (401)	true	<p>Issue a POST request on the `/secret/note` end point with a note payload {"note":"my note"} and receive 401 when no X-AUTH-TOKEN present</p> <p>► Hints ► Solution</p>
----	-------------------------	------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

55. Receive 403 status code when X-AUTH-TOKEN does not match a valid token

55	POST /secret/note (403)	false	Issue a POST request on the `/secret/note` end point with a note payload {"note":"my note"} and receive 403 when X-AUTH-TOKEN does not match a valid token ► Hints ► Solution
----	-------------------------------	-------	-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Solution:

The screenshot shows a Postman request configuration for a POST to https://apichallenges.herokuapp.com/secret/note. The Headers tab is selected, showing the following key-value pairs:

- X-CHALLENGER: 49fccee6-36f7-4d4e-9bde-...
- Postman-Token: <calculated when request is...>
- Content-Type: application/json
- Content-Length: <calculated when request is...>
- Host: <calculated when request is...>
- User-Agent: PostmanRuntime/7.37.0
- Accept: */*
- Accept-Encoding: gzip, deflate, br
- Connection: keep-alive
- X-AUTH-TOKEN: KEY

The Body tab shows an empty JSON object. The response status is 403 Forbidden, and the response body is empty.

55	POST /secret/note (403)	true	Issue a POST request on the `/secret/note` end point with a note payload {"note":"my note"} and receive 403 when X-AUTH-TOKEN does not match a valid token ► Hints ► Solution
----	-------------------------------	------	-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

56. Receive 200 when using the X-AUTH-TOKEN value as an Authorization Bearer token

56	GET /secret/note (Bearer)	false	Issue a GET request on the `/secret/note` end point receive 200 when using the X-AUTH-TOKEN value as an Authorization Bearer token - response body should contain the note ► Hints ► Solution
----	---------------------------	-------	---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Solution:

The screenshot shows a Postman request configuration for a GET to https://apichallenges.herokuapp.com/secret/note. The Headers tab is selected, showing the following key-value pairs:

- Authorization: Bearer 2f376c1-0b84-426...
- Postman-Token: <calculated when request is...>
- Host: <calculated when request is...>
- User-Agent: PostmanRuntime/7.37.0
- Accept: */*
- Accept-Encoding: gzip, deflate, br
- Connection: keep-alive
- X-Challenger: 49fccee6-36f7-4d4e-9bde-...

The Body tab shows an empty JSON object. The response status is 200 OK, and the response body contains the note "Lorem Ipsum Loze Amore".

56	GET /secret/note (Bearer)	true	Issue a GET request on the `/secret/note` end point receive 200 when using the X-AUTH-TOKEN value as an Authorization Bearer token - response body should contain the note ► Hints ► Solution
----	---------------------------------	------	---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

57. Receive 200 status code when a valid X-AUTH-TOKEN value used as an Authorization Bearer token.

57	POST /secret/note (Bearer)	false	<p>Issue a POST request on the `/secret/note` end point with a note payload e.g. {"note":"my note"} and receive 200 when valid X-AUTH-TOKEN value used as an Authorization Bearer token. Status code 200 received. Note is maximum length 100 chars and will be truncated when stored.</p> <p>► Hints ► Solution</p>
----	----------------------------	-------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Solution:

The screenshot shows a Postman interface with the following details:

- Method:** POST
- URL:** https://apichallenges.herokuapp.com/secret/note
- Headers:**
 - Authorization: Bearer 2f3766c1-0b84-426...
 - Content-Type: application/json
 - Content-Length: <calculated when request is...
 - Host: <calculated when request is...
 - User-Agent: PostmanRuntime/7.37.0
 - Accept: */*
 - Accept-Encoding: gzip, deflate, br
 - Connection: keep-alive
 - X-Challenger: 49fccee6-36f7-4d4e-9bde-...
- Body:** {"note": "Lorem Ipsum"}
- Response:** 200 OK, 444 ms, 802 B

57	POST /secret/note (Bearer)	true	<p>Issue a POST request on the `/secret/note` end point with a note payload e.g. {"note":"my note"} and receive 200 when valid X-AUTH-TOKEN value used as an Authorization Bearer token. Status code 200 received. Note is maximum length 100 chars and will be truncated when stored.</p> <p>► Hints ► Solution</p>
----	----------------------------	------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

58. Delete the last todo record in the system so that there are no more todos in the system

Miscellaneous Challenges

We left these challenges to the end because they seemed fun, but... different.

ID	Challenge	Done	Description
58	DELETE /todos/{id} (200) all	false	<p>Issue a DELETE request to successfully delete the last todo in system so that there are no more todos in the system</p> <p>► Hints</p>

Solution:

The screenshot shows the Postman interface with the following details:

- Method: DELETE
- URL: https://apichallenges.herokuapp.com/todos/10
- Headers (7): X-CHALLENGER, Postman-Token, Host, User-Agent, Accept, Accept-Encoding, Connection
- Body: None
- Response status: 200 OK
- Response time: 476 ms
- Response size: 780 B

58	DELETE /todos/{id} (200) all	true	Issue a DELETE request to successfully delete the last todo in system so that there are no more todos in the system
▶ Hints			

59. Create as many POST requests as possible for todos

59	POST /todos (201) all	false	Issue as many POST requests as it takes to add the maximum number of TODOS allowed for a user. The maximum number should be listed in the documentation.
----	-----------------------	-------	----------------------------------------------------------------------------------------------------------------------------------------------------------

Solution:

The screenshot shows the Postman interface with the following details:

- Method: POST
- URL: https://apichallenges.herokuapp.com/todos
- Headers (9): raw, JSON, Beautify
- Body (JSON):

```
1 {
2   ... "title": "POST_21",
3   ... "doneStatus": false,
4   ... "description": "Post description_21"
5 }
```
- Response status: 400 Bad Request
- Response time: 451 ms
- Response size: 858 B
- Response body:

```
1 {
2   "errorMessages": [
3     "ERROR: Cannot add instance, maximum limit of 20 reached"
4   ]
5 }
```

59	POST /todos (201) all	true	Issue as many POST requests as it takes to add the maximum number of TODOS allowed for a user. The maximum number should be listed in the documentation.
----	-----------------------	------	----------------------------------------------------------------------------------------------------------------------------------------------------------