ID: 2020091

Name: Jackson Ferreira Dos Santos

Heroku: [https://ancient-tundra-70389.herokuapp.com/](https://ancient-tundra-70389.herokuapp.com/" \t "_blank)

Github: [https://github.com/ferreirajackson/CBWA-API-CA1](https://github.com/ferreirajackson/CBWA-API-CA1" \t "_blank)

The goal of this document is explaining the changes made on this project. I decided to implement the following four functionalities:

* Front-end;
* Add in due date;
* Add in email notifications;
* Add in watchers;

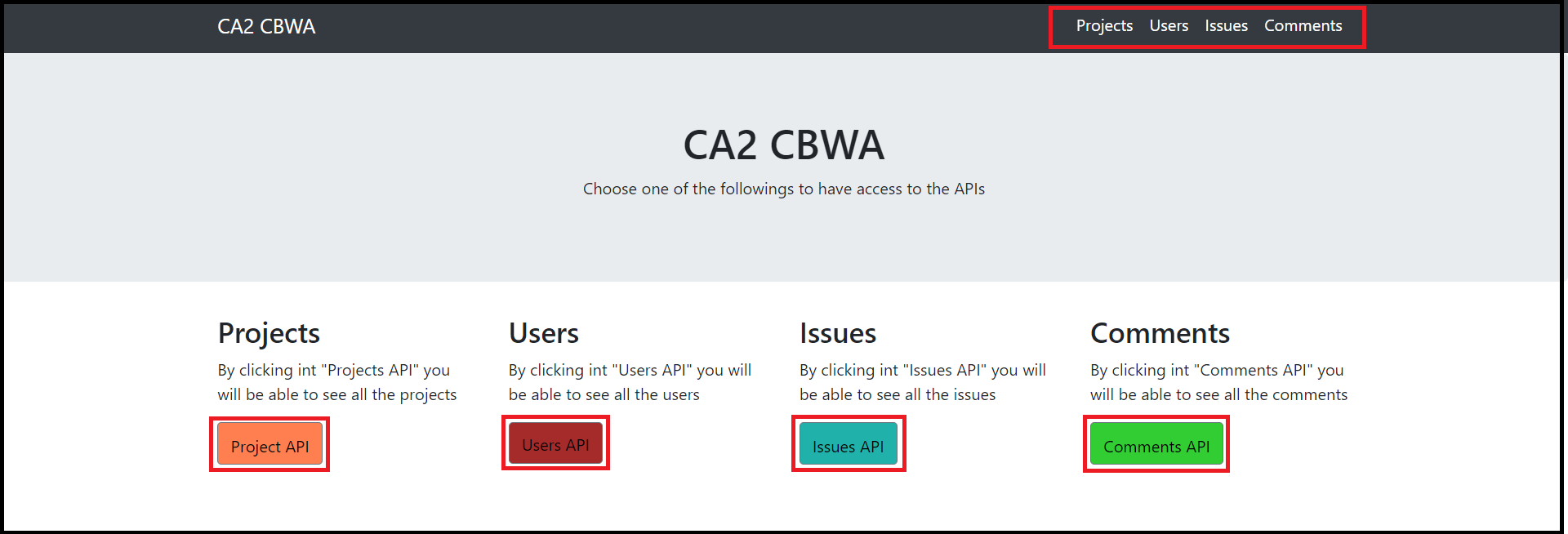
Throughout this document I’ll be also showing the procedure to test these functionalities.

**Frontend**

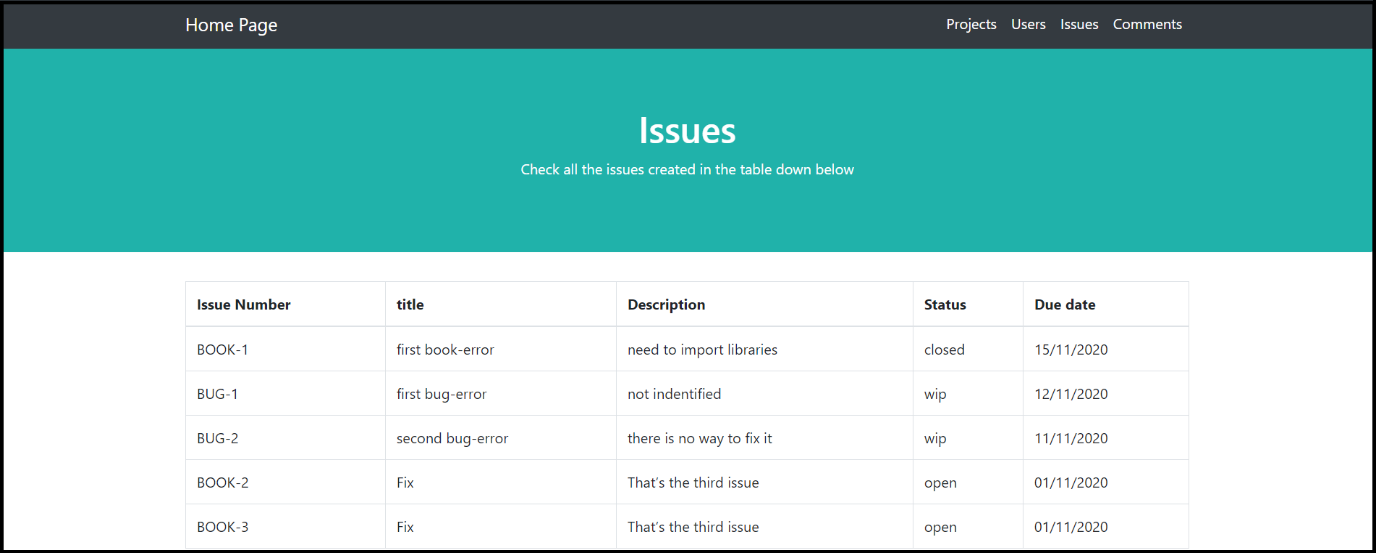
The frontend part has been created with a few pages so the user can access the data in two formats, in a **table** and **json**. Not much was developed for this point as not being the main focus for the “Your choice” topic.

You can access the website through the link <https://ancient-tundra-70389.herokuapp.com/>.

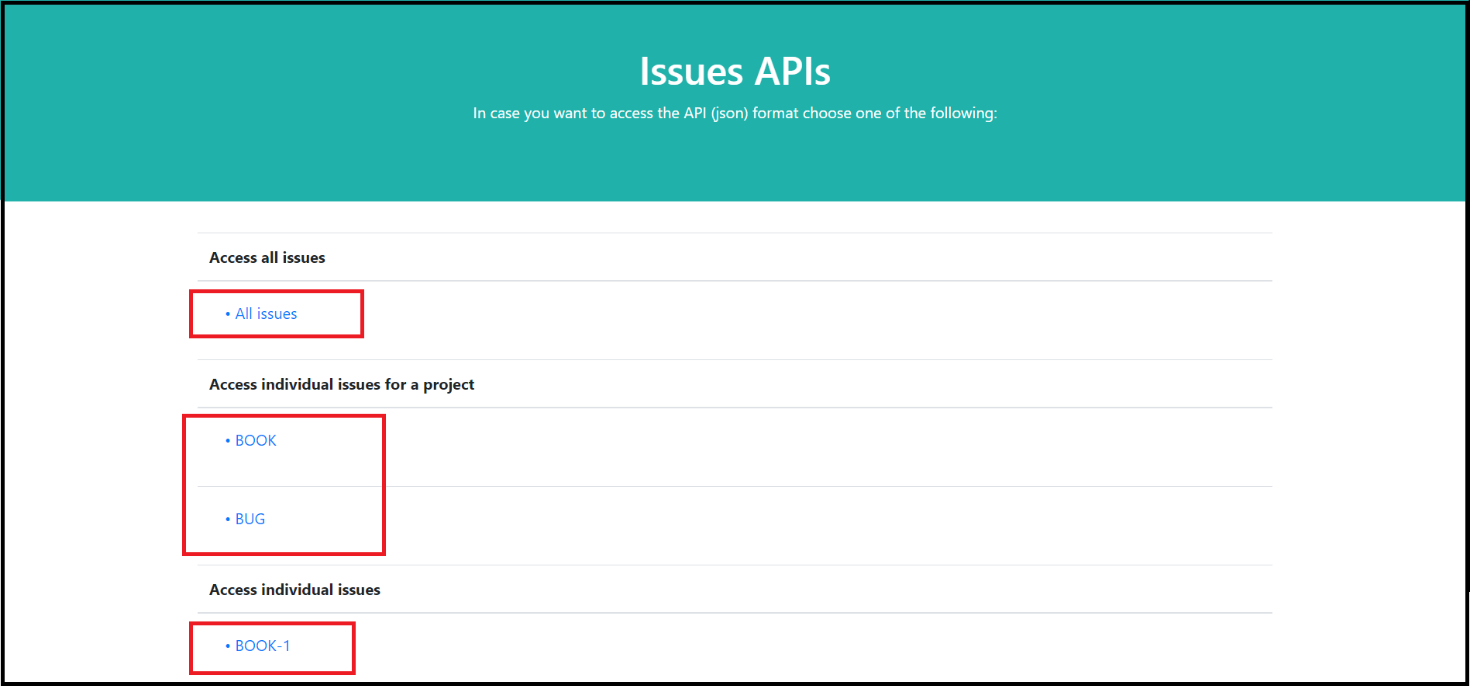
For the **table,** you can access all for main searches (GET) by clicking in the highlighted buttons as the screenshot shows:



And you’ll get something like:



If you want to the **json,** click in one of the links:



And you will get the **json** version of it:



**Due date**

A due date field was added to the issues collections. No comments can be added to an issue neither the status can be updated if an issue has its due date expired.

IMPORTANT: As a new field (dueDate) was added, its necessary to use now the following json format when posting a new issue:

The date is formed following this format: MONTH/DAY/YEAR – 11/30/2020

Add new issues to a project individually

[https://ancient-tundra-70389.herokuapp.com/projects/](https://still-journey-06848.herokuapp.com/projects/)BOOK/issues

json:

{

    "title": "Fix",

    "description": "That’s the third issue",

"dueDate": "11/20/2020"

}

OK, now that we have added a new issue that expired five days ago, let’s try to add a comment to this expired issue.

<https://ancient-tundra-70389.herokuapp.com/issues/BOOK-3/comments/>

json:

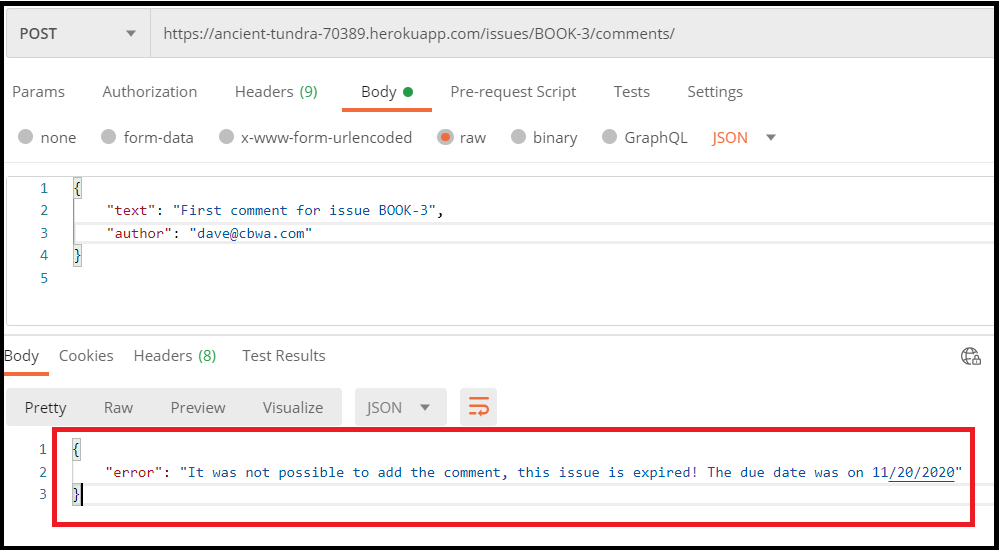
{

    "text": "First comment for issue BOOK-3",

    "author": "[dave@cbwa.com](mailto:dave@cbwa.com)"

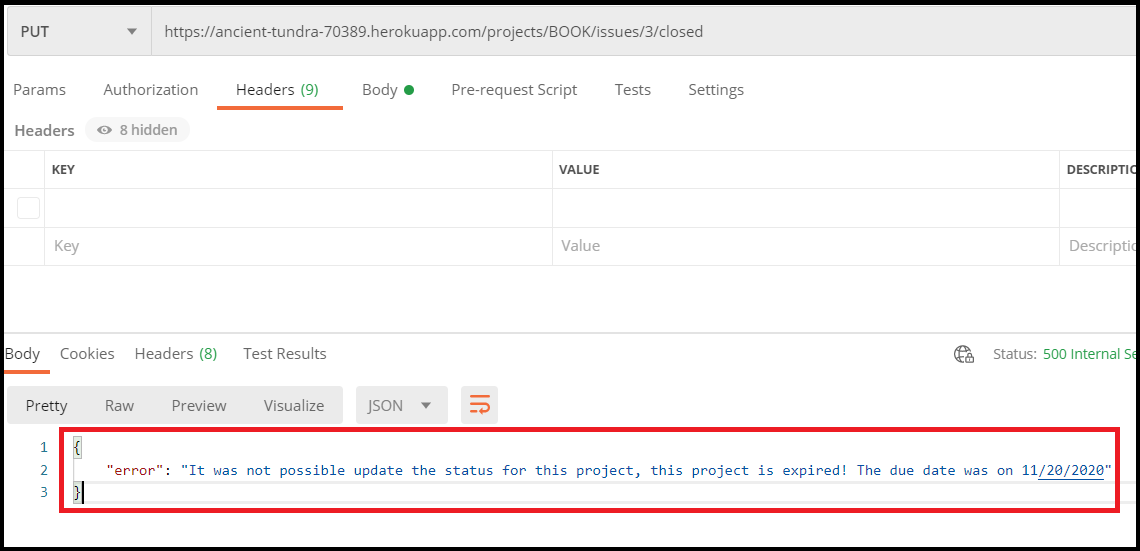
}

Trying to add a comment to an expired issue. (Today being the 11/25/2020)



Trying to update the status of an expired issue. (Today being the 11/25/2020)

<https://ancient-tundra-70389.herokuapp.com/projects/BOOK/issues/3/closed>



**Email notifications and watchers**

IMPORTANT SETUP: For most of the logic to work, a user with a valid and existent email must be created and set as admin as this system works with the email functionality.

Add new users individually (POSTMAN)

[https://ancient-tundra-70389.herokuapp.com/users](http://0.0.0.0:3000/users)

json:

For example

{

    "name": "tester",

    "email": "ca2mailer2020@gmail.com", -> (CHANGE TO YOUR PERSONAL EMAIL)

    "usertype": "admin",

"key": "new password"

}

I made use of the nodemailer library to send emails to the admins every time a comment to an issue is added and the status of a project is updated, By doing this the watchers functionality was at the same time implemented.

Add new comments to an issue

<https://ancient-tundra-70389.herokuapp.com/issues/BUG-1/comments/>

json:

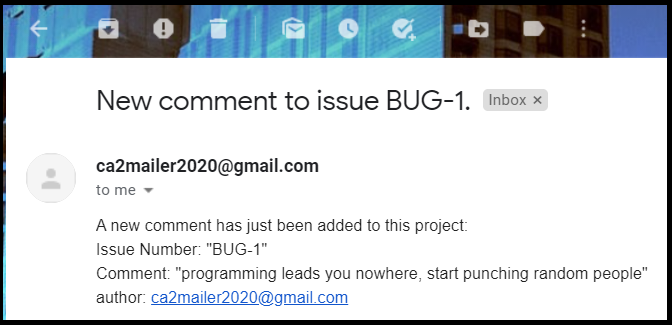
{

    "text": " programming leads you nowhere, start punching random people",",

    "author": "[ca2mailer2020@gmail.com" >>> (CHANGE](mailto:ca2mailer2020@gmail.com%22%20%20%3e%3e%3e%20(CHANGE) TO YOUR PERSONAL EMAIL)

}

And all the users with usertype defined as “admin” will receive an email



In case a status is updated:

BONUS: Updated the status of an issue – PUT METHOD

[https://ancient-tundra-70389.herokuapp.com/projects/BOOK/issues/2/closed](http://0.0.0.0:3000/projects/BOOK/issues/2/closed)

As number 2 being the position (1 – first comment | 2 – second comment …)

