Software Evolution - Practical Session:
Building a GitHub Bot

Guest Lecturer:
Mairieli Wessel, TU Delft

Academic year 2021-2022

The goal of this practical session is to familiarize yourself with the development of bots for managing GitHub repositories, and to explore the benefits and challenges of developing and using GitHub bots.

GitHub bots are applications that automate workflow activities on GitHub, using webhooks and the GitHub API. Bots can do many different things, such as automatically responding to users, applying labels, closing issues, creating new issues, and even merging pull requests.

1. Bots for GitHub repository management

1.1 Preparation

Before starting, please do the following:

1. Make sure you have **Python 3.9.X** installed on your machine.
2. Create a **GitHub account**, if you do not already have one yet.
3. You need to create two GitHub repositories to complete the assignment

   The first repository is needed to hold the codebase for the bot. This is where you will actually push the source code. Fork the following repository which contains the source code of the bot presented in the practical session:
   https://github.com/mairieli/umons-bot-tutorial

   The second repository will be the repository where the bot will be installed. Your bot will interact with this repository. In the real world, this will be the project that you're managing.

There are other **important resources** that we are going to use throughout the assignment:

1. **GitHub API** v3 documentation
   a. Events and payloads:
      https://docs.github.com/en/developers/webhooks-and-events/webhooks/webhook-events-and-payloads
2. **Flask** as our Python webserver
   a. Installation: pip install -U Flask

    b. Flask documentation: https://flask.palletsprojects.com

3. **PyGitHub** to communicate with GitHub
    a. Installation: pip install PyGithub
    b. PyGitHub documentation: https://pygithub.readthedocs.io

4. **cryptography** for decrypt our Github App certificates
    a. Installation: pip install cryptography
    b. cryptography documentation: https://cryptography.io

5. **smee.io** to receive payloads from GitHub
    a. Installation: npm install --global smee-client

1.2 Implemented feature: Greeting first-time developers

Fork the bot repository

1. Fork the following repository which contains the source code of the bot presented in the practical session: https://github.com/mairieli/umons-bot-tutorial

Creating a Webhook Payload Delivery

2. Go to smee.io click on “Start new channel”
3. Copy the WebHook Proxy URL and save it for later use
4. Install smee client on your machine:

    npm install --global smee-client

5. After installing, validate the installation by running:

    smee --version

    If the smee version appears, you are good to go!

Creating a GitHub App

6. Log in to your GitHub account, click on ***your account icon*** → ***Settings*** → ***Developer settings***.
7. Go to ***GitHub Apps***, and click on the ***New GitHub App*** button to the top right.
8. Give your app a name and a homepage

9. Go to the ***Webhook*** section, in the ***Webhook URL*** copy and paste the URL that you got from smee.io (**Step 4**)

10. Go to the ***Repository permissions*** section, here we will define what permission our bot will need when someone installs it on its repository. Our bot needs (so far!) permission to read & write a comment on a pull request, so you need to add ***Access: Read & Write*** under the ***Pull requests*** section and the ***Issues*** section.

11. Go to ***Subscribe to events*** and check the ***Pull request*** checkbox and the ***Issues*** checkbox, it will tell the bot to send us an event payload when a pull request and issue events are created.

12. Click on the ***Create GitHub App***

13. After you create the ***GitHub App*** please save the ***App ID*** for later use

14. We need to create a private key to authenticate with GitHub. Go to the bottom of the bot app page, you will find the ***Private keys*** section, click on the ***Generate a private key***, it will create and download a ***.pem file*** with the private key inside.

15. This .pem file is very important, ***save it*** in a folder in your bot repository, we will need it later.

16. Go to your bot app page and on the top, you will find the ***App Public URL***

17. Copy and paste it into your browser, it will open the bot installation page.

18. Install it on your second repository

Fork and clone the bot project

19. Clone to your machine the repository you have forked (**Step 1**)

20. Run *pip install -r requirements.txt*

21. Change line 7 with your App ID (as Integer)

22. In line 10 we read the certificate for the bot, please make sure to change the location to your .pem file

23. Redirect the payload from smee.io to your bot by running the following command:

    smee -u https://smee.io/<Your Channel ID> --port 5000

24. Run *python3 app.py*

2. Assignment

2.1 Exercises

Using the source code presented in the practical session as a baseline, implement the following features for the bot:

1. **Say thanks when a pull request has been merged.**
   Make the bot post a comment to say thanks, whenever a pull request has been merged.

   For this case, you will want to subscribe to the **pull_request** event, specifically when the action to the event is **closed**.

   For reference, the relevant GitHub API documentation for the pull request event is here: https://developer.github.com/v3/activity/events/types/#pullrequestevent

   Note: A pull request can be closed without it getting merged. You will need to find how to determine whether the pull request was merged, or simply closed.

2. **Automatically delete a merged branch.**
   Automate even more tedious tasks whenever a pull request has been merged. Make the bot automatically delete a merged branch. The branch name can be found in the pull request webhook event.

   As for the previous exercise, you will want to subscribe to the **pull_request** event, specifically when the action to the event is **closed**.

   For reference, the relevant GitHub documentation for references is here:
   https://docs.github.com/en/rest/reference/git#references
   Relevant PyGitHub documentation:
   https://pygithub.readthedocs.io/en/latest/github_objects/GitRef.html#github.GitRef.GitRef

3. **Prevent merging of pull requests with "WIP" in the title.**
   Make the bot set a pull request status to **pending** if it finds one of the following terms in the pull request titles: “wip”, “work in progress”, or “do not merge”. Developers might update the pull request title at any time, so take it into account and update the pull request status accordingly. For example, once the pull request is updated and term removed, set back the pull request status to **success**.

   For this case, you will want to subscribe to the **pull_request** event, specifically when the action to the event is **edited**.

   For reference, the relevant GitHub documentation for the commit statuses is here:
   https://docs.github.com/en/rest/reference/commits#commit-statuses

Examples using PyGitHub:
https://pygithub.readthedocs.io/en/latest/examples/Commit.html#create-commit-status-check

Note: Commit statuses can determine the overall status of a pull request. You will need to find how to change the pull request status based on a commit status. Read the documentation for reference.

2.2 Assignment report

In a short report document, provide:

1. The URL of the public GitHub repository that contains the source code of all implemented features of the bot (outcomes of exercises 1, 2, and 3).

2. For each exercise, also include the respective screenshots showing the successful bot interaction (i.e., bot outcome) on a pull request.

2.3 Bonus exercise

Automatically restore a deleted merged branch based on a developer request.
Implement a new feature for the bot to make it interact with a developer. Have the bot react to a developer commenting “@mons-bot ready for review” in a pending pull request. The bot, then, overwrites the pending status regardless of the pull request title content.

Note: Replace “@mons-bot”, in the command that calls the bot, with the name of the bot you registered at GitHub.com.