“Plone: Guillotina API Evolution” GSoC 2019

Name and Contact Information

Name : Karan Pratap Singh

University : Indian Institute of Technology Kharagpur, India

Email : karanpratapsingh43@gmail.com karan.pratapsingh@iitkgp.ac.in

Github : karannaoh LinkedIn : karannaoh

WebSite : karannaoh.github.io

Open Source involvements

Here is the list of contributions to Plone -

- Created one click Heroku deploy for Guillotina guillotina-heroku , 1fde547
- Issue #377 resolve PR #470

Other contribution in xwiki-

- XWIKI-16054 : Fix #1076: CSS bug fixed
- XWIKI-16219 : Improve #1084: Refractored javascript.

Abstract

Following six things will be implemented during the project

- Improving Guillotina API by making it a more compliant to JSON-LD.
    - Research all specific features of JSON-LD and make Guillotina follow all the standard practices of JSON-LD
- All the API functionalities will be supported by a websocket endpoint also.
    - Will be Implementing websocket endpoint for the API so that user can access the API through websocket client.

- JSON validation for all the payloads by well-defined JSON schemas.
    - JSON schema validation is already supported by Guillotina but not on all the JSONs are validated, validation of all the JSON objects will be ensured by this task
- Updating swagger documentation with OpenAPI 3.
    - Swagger documentation needs to be updated to work with OpenAPI3 /swagger3.
- API versioning
    - It's generally necessary to version API after every major change, which is change in response format or response type.
    - A mechanism to version will be implemented to version every major change on the API
- Helm Charts for Guillotina to make Guillotina more scalable.
    - Aim of this is to create an easy deployable on any Kubernetes cluster
    - Final result will be a repository with helm charts of Guillotina and a PR to helm/charts repository with helm charts
- Coordinating with plone.restapi to ensure similar implementation of APIs.
    - As it is a major API evolvement it must be confirmed that plone.restapi is coherent with the changes made to maintain integrity.

Implementation

This project mostly requires peer review from experienced Plone developers. So I'll create several feature branches in the repository and push my codes so that the community can see my progress as well as suggest code changes. This will eliminate last moment reviews and give more time to experienced Plone developers for review. And also the community can track my progress if interested.

Firstly, I'll start by making API coherent to JSON-LD best practices. This will need a bit of research about what parts of Guillotina need improvements so I can start this during the community bonding period so that I can analyse Guillotina and finish the first task simultaneously.

First Task

- It will start with analysing JSON-LD standard and where Guillotina API fails to follow them.

- The second thing will be to make changes where API fails to follow JSON-LD specifications.
- It's going to be iterative way, will be analysing and updating API iteratively.

Then, I'll start to work on the second task to add support of websockets for all the API functionalities.

- For this, I will be using websocket provides `aiohttp`.
- It will start from analysing the already existing websocket endpoint `ws.py`, making a list of API functionalities it doesn't supports.
- Afterward adding all the missing functionalities to the websocket endpoint.
- Adding unit tests for all the functionalities .

The third task will be to validate all the JSON

- For this, I will be using `jsonschema`.
- Guillotina already supports validation json schema for some of the JSONs, this task will start from making list of all the JSON bodies which are not being validated.
- The second thing will be validating all of those, will be using `jsonschema` which is already used in Guillotina.

Fourth to updating swagger documentation.

- Will be updating swagger documentation of the API

Fifth implementing versioning to API endpoints.

- There are three different versioning mechanism for APIs as follow:-
    1. URI versioning.
    2. Versioning using custom Request Headers.
    3. Versioning using Accept Headers.

  Following https://restfulapi.net/versioning/

- The first is adding version to URIs
- Second is to add Add version in the in a custom header of the request

```
Accept-Version:v1
```

- The third is to add version in the Accept header of the request

  ```
  GET /api/abc HTTP/1.1

  Accept: application/abc.myapp-v2+json

  HTTP/1.1 200 OK

  Content-Type: application/abc.myapp-v2+json
  ```

- We will be following the 3 mechanism to add version in the Accept Headers of the API.

The last thing will be to implement helm chart for Guillotina.

- Start from writing kubernetes configuration files for templates.
- Adding support of addons like ingress.
- Creating helm charts with postgress helm chart as a dependency
- Testing Helm charts on minikube, EKS, and GKE.

In between, it will be ensured that all change made in API are in sync with plone.restapi.

Timeline

I have already started getting familiar with the code base, so I'll be looking to start early so that things go smoothly during coding period. I often set pre-deadlines which help in improvising in the last minutes too.

The whole project can be broadly categorized into milestones and labels :-

Milestone 1: Making API more JSON-LD Compliant.

- Label 1.1: Analyzing Guillotina API, where it need changes according to JSON-LD.
- Label 1.2: Making Guillotina completely follow JSON-LD.

Milestone 2: API versioning.

- Label 2.1: Adding versioning to all the API endpoints.
- Label 2.2 Updating documentation and unit test accordingly and Coordinate with plone.restapi to make relevant changes.

Milestone 3: Adding all the API's functionalities to the websocket endpoint.

- Label 3.1: It will start from analysing the existing websocket route(`ws.py`), listing all the funtionanlities it doesn't has.
- Label 3.2: Adding all those functionalities to the websocket enpoint.
- Label 3.3: Unittest for socket route testing all the functionalities of API.

Milestone 4: JSON schema validation, and updating swagger documentation.

- Label 4.1: Write schemas to validate all the JSON payloads.
- Label 4.2: Update swagger documentation to support OpenAPI3.

Will ensure all the changes made to any endpoint is accordingly changed in plone.restapi.

Wishlist (W): Work in free time / after Milestones.

- Adding kubernetes configuration files.
- Writing helm charts to make deployment easy on kubernetes.

Community Bonding Period

- Get more familiar with Guillotina and other coding conventions.
- Get more familiar with the code structure, and community by actively participating in discussions (community.plone.org /gitter) and solving bugs/ issues.
- Start to work on milestone 1.
- Research all change Guillotina needs to become fully JSON-LD compliant.
- set up blogs for weekly/ fortnight update. Keep my work documented.

Week I (May 27 - June 3) - Week II (June 3 - June 10)

- Make all the necessary changes needed to make Guillotina follow JSON-LD
- Update unittests accordingly.

- Get feedback from mentors.

Week III (June 10 - June 17)

- API versioning: Adding a version number to all the API endpoints.
- Updating and adding unittest for the changes in endpoints

Week IV (June 17 - June 24)

- Refactor code and push initial work for reviews.
- Ensuring integrity of application with plone.restapi.

Mid Term Evaluation I (June 24 - July 1)

- Document everything and write a blog for milestone 1 and 2
- Accomplishment: Milestone 1 and 2.

Week VI(July 1 - July 8) - Week VII (July 8 - July 15)

- Work on adding websocket endpoint for the API.
- Ensure all features work on websocket, write unittests.

Week VIII (July 15 - July 22)

- JSON schema validation, Making List of all the JSON which needs to be not validated.
- Writing validation for all the JSON

Mid Term Evaluation II (July 22 - July 26)

- Document everything and write a blog for milestone 3 and 4

- Accomplishment: Milestone 3 and 4.

Week X (July 26 - Aug 2) - Week XI (Aug 2 - Aug 9)

- OpenAPI3 Support: Updating Swagger documentation.
- Creating kubernetes configurations and helm charts for Guillotina

Week XII (Aug 9 - Aug 19)

- Ensuring Integrity of Application with plone.restapi.
- Ensure documentation of all the code I have written.
- Work on reviews.

Final Evaluation (Aug 19 - Aug 26)

- Finalize codes and documents for final submission
- Work on feedback/review, if any.
- Wrap up everything.
- Accomplishment: All 4 Milestones

Why ME?

I'm from Non-CS background, I was introduced to C programming language in my First year. Started to work as a Developer in a Startup from the very first year later led the entire tech team of that startup.

In the process, I interned at two companies. Worked on web development, scraping and deployment of applications on AWS and Heroku. Worked intensively on Python during the first intern for scraping more than 1 million pages of Wikipedia, written unittests and followed TDD. I have been doing web development for last 3 year and have good experience with web server, APIs and architecture.

I started contributing to the Open source community so that I can build some valuable skills. I have been contributing to small Open source repositories of my college from last 2 years.

I think my passion speaks louder than my words.

Previous experience with Plone

Python has been my favorite language and nearly all of my projects are written in python. I'm familiar with pythonic way of writing codes and writing tests cases. I learned these from internships and while working on some other projects. I have good experience with backend development(doing for almost last 3 years), worked on Django, Flask, NodeJS and PHP and deployment created "easy deploy on Heroku" for Guillotina. And I've also fixed some issues, improved documentation and made some feature improvements. All the above claim is mentioned in Open source involvements section.

Proficiency

Programming Languages : Python, Javascript, Julia, PHP, C/C++ (in order of proficiency).

Web Tech: Django, Nodejs, Laravel.

Utilities: Git, Docker, Kubernetes, AWS, Heroku, helm.

Other commitments

My exams will end before the starting of GSoC coding period. So this makes me favorable to start early and pre-initialize milestone 1. This will be like my full-time job so, I can easily give 40+ hours/week.

Acknowledgment

- Nathan Van Gheem @vangheem
- Kumar Akshay @kakshay21

Thanks for your cooperation.