EPEPEP : Extensive Project Evaluation Platform for Education Purposes

Introduction

EPEPEP is a platform that a Web developer teacher can use to evaluate a large number of projects through their git repository.

How it works:

1. A list of GitHub repositories is provided to the platform.
2. The platform clones each repository and runs a set of tests.
3. The platform generates a report with the results of the tests.
4. The teacher can access the report and see the results of the tests for each project.

Report

The platform needs to report the following information:

Github activity

Commits

- Total number of commits
- Number of commits per TM (team member)
- Chronological display of commits per TM (goal: to see if the team is working consistently)

Branches activity

- Number of branches
- Number of commits per branch
- Number of commits per TM per branch
- Average number of commits per branch

Other (optional)

- Number of issues opened and closed (if possible)
- Number of pull requests opened and closed (if possible)

Contributions

Total number of LOC (lines of code)

- Total number of LOC (added, deleted, modified)
- Total number of LOC per TM (added, deleted, modified)
- Total percentage of LOC per TM (added, deleted, modified) according to the total number of LOC

Total here means the sum of the LOC added, deleted and modified for each commit of the whole project.

Final number of LOC

- Final commit number of LOC per TM
- Final percentage number of LOC per TM

Final here means the LOC of the project at its final state. By looking at the final state, we can see what lines of code have been modified by each team member (git blame).

For example, if TM A has add 100 lines of code and TM B has modified 50 of the 100 lines, then, by counting each commits, the total number of LOC will be:

TM	LOC
A	100
B	50

But the final number of LOC, by looking only at the final state of the project, will be:

TM	LOC
A	50
B	50

Because TM B has modified 50 lines of code that TM A has added (check the To help you section). If TM C modify the 50 lines of code that TM A has, then TM A will have 0 final LOC (still have 100 Total).

Git good practices

- List the names of the branches (max 100?)
- List the message of the commits (max 100?)
- Run Cspell on the branches names, display the branches names that have spelling mistakes
- Run Cspell on the commit messages, display the commit messages that have spelling mistakes

Code quality

- Run Cspell on the code, display the lines of code that have spelling mistakes and which TM has written them
- Display the project tree structure

output

One ore more markdown files per project would do the job. A web interface would be a plus, but not mandatory.

Feel free to add more information to the report if you think it is relevant.

Configuration

Sometime, multiple git accounts represent the same person. The platform should be able to be specified which accounts are the same person.

For example, check the DebiAI PM contribution percentage:

TM	LOC	%
tom.mansion	4619	71.9%
Nicolas Leroux	1396	21.73%
FadyCoding	355	5.53%
MacaireM	46	0.72%
Fady BEKKAR	8	0.12%

`FadyCoding` and `Fady BEKKAR` are the same person, so the platform should be able to merge these two accounts (according to an handwritten user configuration).

Constraints

- Should be ready at the latest by January 5, 2025
- No spelling mistakes in the report

Deliverables

- The platform and all the necessary documentation to install and use it
- In a GitHub repository (open or private, as you wish)

Remuneration

- Your hourly rate

To help you

State of the art

Spend a little bit of time to check if this does not already exist. Maybe some libraries or tools already do most of the work.

Last commit number of LOC per TM

It is possible to know, at the current commit, who has contributed what to the project. This git `count-lines` alias allows to count the number of lines added, removed and modified by a specific author:

```
git config --global alias.count-lines "! git log --author=\"\$1\" --
pretty=tformat: --numstat | awk '{ add += \$1; subs += \$2; loc += \$1 -
\$2 } END { printf \"added lines: %s, removed lines: %s, total lines:
%s\n\", add, subs, loc }' #"
```

Also, the get_line_contributors.py script allows to get the number of lines modified by each contributor at the current repository commit.

List all the branches

```
git branch -a
```