Peer Review Concept wdb: MC-1 FS23

Fernando Benites, 01.03.2023 version $0.\overline{9}$

Motivation

- Students learn very limited with my inputs/learn material. I can only show what I would do, and why, but seeing other code and having to evaluate puts a totally different light on their own written code and coding process.
- More interaction between students to learn more (Vygotski, zone of proximal development), here different aspects play a specific role: own coding vs evaluation of other code; project vs general knowledge vs exam

Procedure, students submit their code, then, the next day the repos are opened to all others. Each student receive a repo/project to evaluate from the FE. The student forks the code and create issues/comments/pull-requests evaluating/improving/commenting the code. The student submits then the repo to the lecturer (fernando.benites@fhnw.ch). After 1-2 weeks the lecturer gives the note on the project and says if the student's evaluation was good (0.25 plus to the project note), neutral (project note does not change) or bad/not submitted (-0.25 to the project note). The goal is not to penalize, but to assure, the knowledge obtained in the projects is distributed through the groups. A good evaluation follows the rules described in this document, and provides a service to the group of the project and to the evaluating student himself/herself. A bad evaluation is not done properly (e.g., missing/skipping/not addressing many points, not comments on many points, being destructive in the tone, giving wrong suggestions, or not done at all). The goal is to get everyone engaged and that everyone learns from each other. It is also a goal, that between MC-1 a jump in quality is seen (by applying the suggestion given by the evaluators).

Organisation

Hours budget:
MC-1 20h (2-3 Persons: 40-50h total)

10h Exam preparation
1h Exam
Peerreview 5h
Mediathek 19h

MC-1 Submission: 15.05.2023
MC-1 Review: 26.05.2023

Woche	1	2	3	4	5	6	7	8	9	10	11	12	13	14
MC-1										Submission Monday				
Review												Review Friday		

Evaluation

Schema Mini Challenge 1:

1. Objective is to acquire data in an efficient way despite security checks/cookies management
    1.1. 5 pt what happens if there is a blockage, how to circumvent?

2. You need 1000-10000 of items saved (so I am convinced you didn't click yourself through the websites)
    2.1. 10 pt for 10k, (8 pt for 1000, 5 points for 500, 2 pt for 100)
3. Your code follows basic clean code rules: https://docs.python-guide.org/writing/style/ . This weights 10% of the mini-challenge note.
    3.1. 5pt Functions are no longer than 100 lines
    3.2. 5pt There are many functions
    3.3. 5pt There is a differentiation (in form of different functions) between frontend (webcode/selenium) and back end (storage/saving the information)
    3.4. 2pt sensible naming of variables and functions
    3.5. 3pt use of global variables is minimized
4. You have documented with comments and possibly unit tests (making it reproducible and easy for others to understand): https://realpython.com/python-testing/ (unittest and selenium: https://www.lambdatest.com/support/docs/running-unit-testing-in-python-on-lambdatest-selenium-grid/). You have checked many edge cases in the tests/try/excepts (also for RegExp). This makes 50% of the mini-challenge note, criteria is how good are the tests (coverage and quantity), how much documentation is there, are the results reproducible (I might ask you if there is some problem).
    4.1. 20 pt code has comments and they are helpful (5 pt: did very little, 15 pt everything is useful and there, 18pt very good, 20pt obsessive disorder)
    4.2. 5pt there is a readme that is useful (guide how to install, why is a good project, any further useful information)
    4.3. 20pt code is reproducible (you can clone the repo and install and run; 5 pt you can clone it and but you need to figure out which libraries, there is errors when running which need fixing, maybe the branching is also confusing, etc;

15pt there is a requirements.txt, a bash script to run or and IDE (vs-code/jetbrains), which also has a run configuration, 18pt, you don't have any constructive advices, 20pt there is a dockerfile which does all the installing and lets it run)

4.4. 5pt the directories are sound, there is a test (maybe in src), a docs, and a src directory.
4.5. 5 pt there is a branch system helping the features, testing and other tasks clear
4.6. 10 pt there are issues which are linked to the
4.7. 5 pt extensive logging
4.8. 25 pt there are try excepts (10 pt if obvious catches are there, 20 pt for many exceptions, 25 pt, there is extensive error handling of the exceptions and even pdb infos)
4.9. *Bonus: 5 pt there is an integration test (wget or selenium, probing the different features of the webservice probed)

5. Extract valuable information
   5.1. 10pt use regexp/xpath/json path to extract important infos of each page.
   5.2. 5pt do EDA on this data (also on the websites you crawled).
   5.3. 5pt create at least 2 interesting insights about the Data.
   5.4. 5pt visualize your insights
   5.5. 5pt what are use cases for your insights?
   5.6. 10pt you use another (cloud) API to enrich your data and gain even more insights (e.g. you azure/google cloud - Vision or Text (also custom model) to get some insights on the pictures (how many people) or text (which sentiment))
6. For each mini-challenge you submit, you include a report about the main ideas of your mini-challenge implementation (1-3 pages). This makes 30% of the note. Here the main aspects I will look for are (in order of importance): description of the idea, data science aspects (any further analysis of the results), relevancy to real world, originality, usefulness.
   6.1. 10 pt language is sound and the report has a good structure
   6.2. 10 pt the motivation and introduction is well written
   6.3. 10 pt you know what the script is targeting and how it access (html, xpath, regex, etc)

Schema Feedback-Mini Challenges:

- Konstruktiv
- Sorgfältig
- Quellen
- Wohlwollend
- Beschreib das Problem genau und schlage konkrete Verbesserungen
- Summary

Guidelines

Method:
Preferred: Pull request /comments summary file
Accepted: summary file with code/comment links (blame -> commit -> comment)
Tolerated (small minus): just issues
Penalized (bigger minus): just comments on file without any source
Full 0.5 Reduction: no effort made

Summary file:
Use schema above, give a general feedback, with at least 3 positive Aspects, tell where points were taken, and link there the comments (exemplary, more examples better, these can be gathered in an issue though).
Plus: where and how they can improve submission.

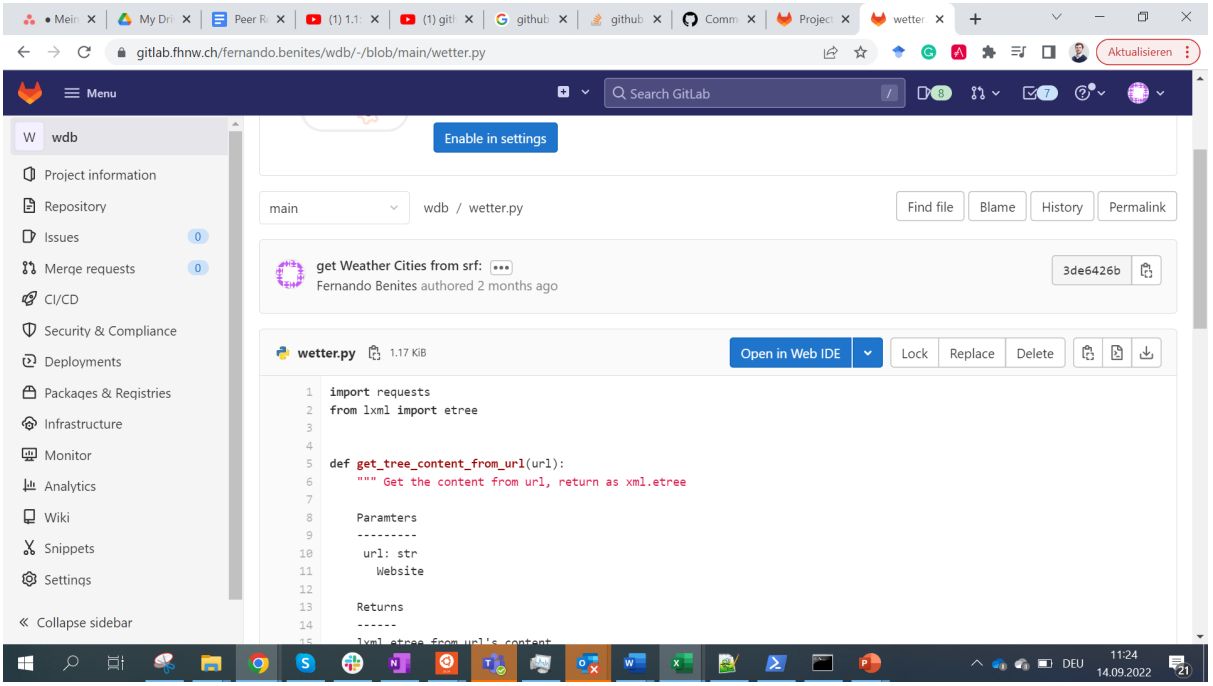
How to grade:
Generally, we want to motivate and round towards better grades. Effort made is commendable, however, looking for better solutions and doing research, experiments, trying new approaches is preferred.

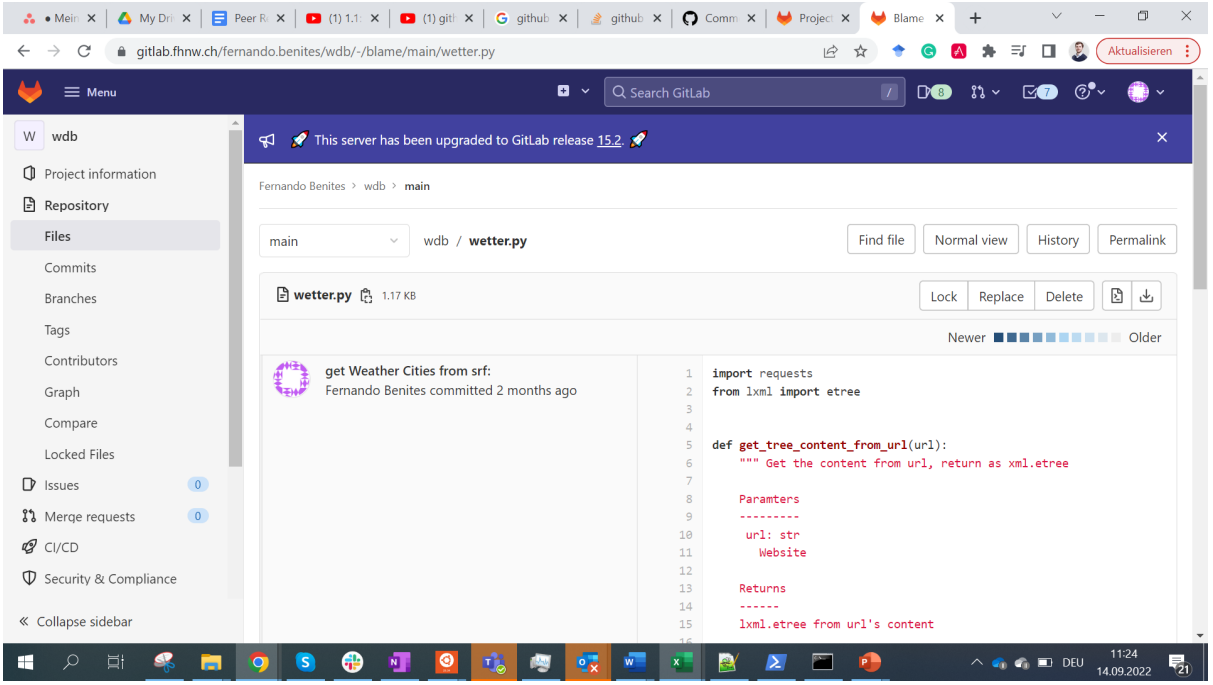
- Full points: wow, or complete without problems
- 80-90%: complete however with small problems
- 60-70%: has problems
- 30-50%: has big Problems
- 10-20%: tried something
- 0: nothing to grade

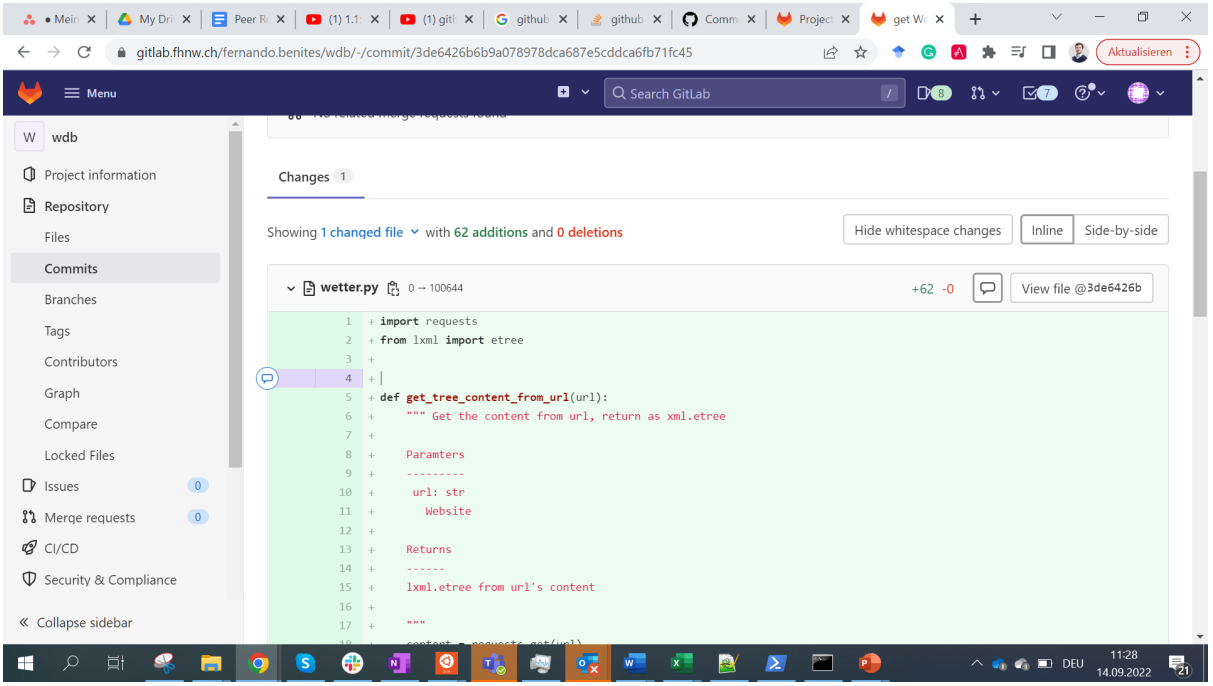
Submission/Distribution:
You allow me access to the repo, I allow all the other students access to the repo, and distribute the assignment (each person grades one project). If you do not grant me enough access rights, you need to invite the other students. We grant read rights to the other students, so that when you are evaluating a project (task), you can check how the other projects did it.

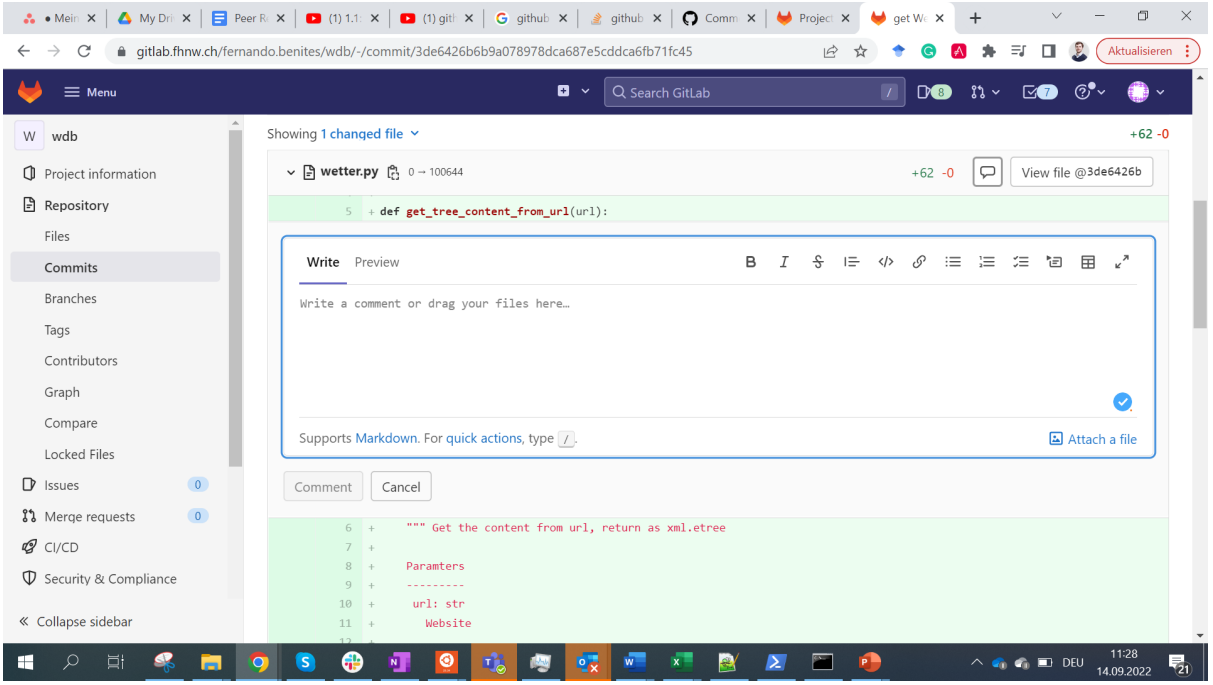
HAVE FUN! (Advice: if you can't think of a good suggestion take a break, have a kitkat or snickers!)

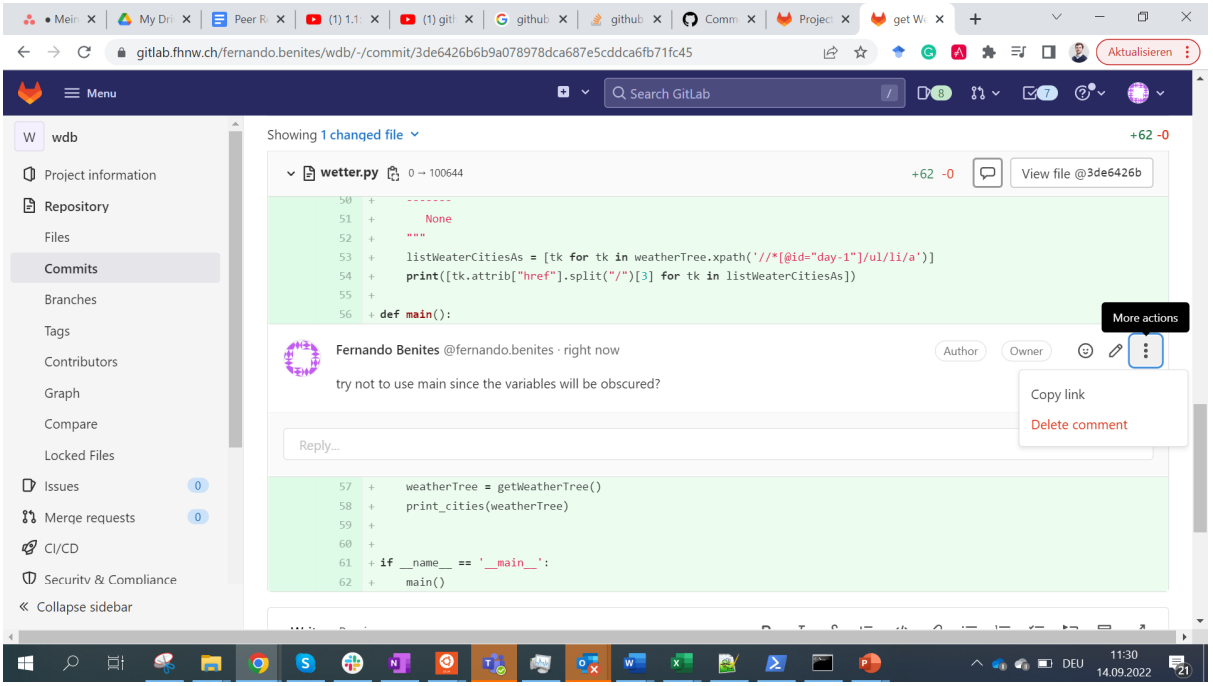
Links:
https://www.youtube.com/watch?v=oK8EvVeVltE
https://gitlab.fhnw.ch/webteam/webservices-api/-/commit/2e3a60118013f004b04f8d1bb9e2bb663e63b4cc


gitlab.fhnw.ch/fernando.benites/wdb/-/blob/main/wetter.py
Menu
Search GitLab
wdb
Project information
Repository
Issues
Merge requests
CI/CD
Security & Compliance
Deployments
Packages & Registries
Infrastructure
Monitor
Analytics
Wiki
Snippets
Settings
Collapse sidebar
Enable in settings
main
wdb / wetter.py
Find file
Blame
History
Permalink
get Weather Cities from srf:
Fernando Benites authored 2 months ago
3de6426b
wetter.py
1.17 KiB
Open in Web IDE
Lock
Replace
Delete
import requests
from lxml import etree
def get_tree_content_from_url(url):
""" Get the content from url, return as xml.etree
Paramters
url: str
Website
Returns
DEU
11:24
14.09.2022


gitlab.fhnw.ch/fernando.benites/wdb/-/blame/main/wetter.py
Menu
Search GitLab
wdb
This server has been upgraded to GitLab release 15.2
Project information
Repository
Files
Commits
Branches
Tags
Contributors
Graph
Compare
Locked Files
Issues
Merge requests
CI/CD
Security & Compliance
Collapse sidebar
Fernando Benites > wdb > main
main
wdb / wetter.py
Find file
Normal view
History
Permalink
wetter.py
1.17 KB
Lock
Replace
Delete
Newer
Older
get Weather Cities from srf:
Fernando Benites committed 2 months ago
import requests
from lxml import etree
def get_tree_content_from_url(url):
""" Get the content from url, return as xml.etree
Paramters
url: str
Website
Returns
lxml.etree from url's content
DEU
11:24
14.09.2022


gitlab.fhnw.ch/fernando.benites/wdb/-/commit/3de6426b6b9a078978dca687e5cddca6fb71fc45
Menu
Search GitLab
wdb
Project information
Repository
Files
Commits
Branches
Tags
Contributors
Graph
Compare
Locked Files
Issues
Merge requests
CI/CD
Security & Compliance
Collapse sidebar
Changes 1
Showing 1 changed file with 62 additions and 0 deletions
Hide whitespace changes
Inline
Side-by-side
wetter.py
0 → 100644
+62 -0
View file @3de6426b
+ import requests
+ from lxml import etree
+
+
+ def get_tree_content_from_url(url):
+ """ Get the content from url, return as xml.etree
+
+ Paramters
+ ---------
+ url: str
+ Website
+
+ Returns
+ ------
+ lxml.etree from url's content
+
+ """
DEU
11:28
14.09.2022


gitlab.fhnw.ch/fernando.benites/wdb/-/commit/3de6426b6b9a078978dca687e5cddca6fb71fc45
Menu
Search GitLab
wdb
Project information
Repository
Files
Commits
Branches
Tags
Contributors
Graph
Compare
Locked Files
Issues
Merge requests
CI/CD
Security & Compliance
Collapse sidebar
Showing 1 changed file
+62 -0
wetter.py
0 → 100644
+62 -0
View file @3de6426b
+ def get_tree_content_from_url(url):
Write
Preview
Write a comment or drag your files here…
Supports Markdown. For quick actions, type /.
Attach a file
Comment
Cancel
+ """ Get the content from url, return as xml.etree
+
+ Paramters
+ ---------
+ url: str
+ Website
DEU
11:28
14.09.2022



https://gitlab.fhnw.ch/fernando.benites/wdb/-/commit/3de6426b6b9a078978dca687e5cddca6fb71fc45#note_282595