Homework 1 – Advanced Software Engineering (2021/22)

Deadline: Oct. 14th, 2021, at 15:59 AM.

Solution code **MUST** be submitted to Moodle as a single `surname_name_hw1.zip` file.

All submissions will be checked for plagiarism. Plagiarised solutions will be awarded an F(0) grade.

Exercise – Wilma and Betty need your help to implement the new *Bedrock-a-Party* RESTful service, exposing the API reported on the next page (→).

The service manages the list of food and drinks, which users bring to a party. Wilma and Betty have already coded a simple `PartySkeleton` based on the microservice skeleton that we have seen in class.

Particularly, Wilma and Betty provide you with:

- a `myservice/classes/party.py` module, which implements the *Bedrock-a-Party* basic functionalities as plain Python code,
- a `myservice/views/parties.py` blueprint, which you must complete to offer all required functionalities of *Bedrock-a-Party* as a RESTful service by using the functions above,
- a `myservice/tests/tests.py` file, which you can run against your solution code by issuing the command `pytest` in the main project folder (after running `pip install pytest`).

Download the `PartySkeleton.zip` available from the Moodle and prototype *Bedrock-a-Party*, relying on the *Flask* micro-framework and working on the `myservice/views/parties.py` file **only**[1].

The solution must pass all provided tests and must be uploaded to your GitHub (please make it public only after the deadline).

Write a short report (300 words at most[2]) containing:

1. the link to the GitHub repository of the project (please make it public only after the deadline), and
2. the screenshot of the successful execution of `myservice/tests/tests.py`,
3. the screenshots of the tests provided in `tests.py`, performed with PostMan for **all** operations[3].

Upload to the Moodle **both** the report and your new `parties.py` file.

Learning Outcomes

- ✓ Revise programming concepts with Python.
- ✓ Revise command-line usage.
- ✓ Get familiar with the Flask microframework and Postman.
- ✓ Get familiar with GitHub.

[1] The API must not be changed nor adapted.

[2] Submitted solutions which exceed the words limit for the report will incur in grading penalties.

[3] Use PostMan to run at least all tests that you find in `tests.py`.

URI	ReqType	Description
`/parties`	`POST`	*Creates a new party and gets the party identifier back.*
	`GET`	*Retrieves all scheduled parties.*
`/parties/loaded`	`GET`	*Returns the number of parties currently loaded in the system*
`/party/<id>`	`GET`	*Retrieves the party identified by* `<id>`*.*
	`DELETE`	*Deletes the party identified by* `<id>` *from the system.*
`/party/<id>/foodlist`	`GET`	*Retrieves the current foodlist of the party identified by* `<id>.`
`/party/<id>/foodlist/<user>/<item>`	`POST`	*Adds* the* `<item>` *brought by* `<user>` *to the food-list of the party* `<id>.`
	`DELETE`	*Removes* the given* `<item>` *brought by* `<user>` *from the food-list of the party* `<id>`

*** Only people invited to the party can add and remove food items from the party list.**