Testausdokumentti_Testing document

Data Structures Lab Course (start of summer 2023)

What has been tested, how was this done:

In order to perform comprehensive unit testing for the entire program, firstly individual functions and components were identified to test. Below identified units will be tested:

def home():
def geocode():
def find_shortest_path():
def plot_shortest_path():

Test cases for each function will be developed, to cover various scenarios. For example, the geocode and home function with valid addresses and invalid addresses should be tested to ensure it returns the expected coordinates or error messages appropriately. For the find_shortest_path and plot_shortest_path function, test cases with different start and end coordinates should be created to verify that the shortest path is calculated correctly. Separate test py file was created to implement those testing. Test classes and methods were used for each function that need to be tested.

Because there may be something wrong with the algorithm implementation, so in the beginning ready-made solutions were used to validate and test if the program frame can function well. The ready-made networkx.shortest_path function provided by the NetworkX library seems to a good solution, as it implements the Dijkstra's algorithm. It returns a list representing the shortest path between the start and end nodes.

How each test case works for each function. For example, test case for home function covers the behavior of the POST request, error handling, redirect and template rendering. Geocoding, path finding and the content of the html file are not covered, as they will be tested for the separate corresponding function. Therefore, significant portion of the code is covered by tests.

Unit testing coverage report:

If you prefer to review or generate coverage reports to your local folder, please first install the coverage package by “pip install coverage”.

Ensure that the **Shortest_path.py** module file is located in the same directory as the **test_shortest_path.py** test file.

Run the test file by using the testing framework: **python -m unittest test_shortest_path.py**. Generate the coverage report by running: **coverage html**, which generates a HTML report in the current directory (wrote HTML report to htmlcov\index.html). Please open the generated HTML report (htmlcov/index.html) in a web browser to view the coverage report. Or run command **coverage run -m unittest discover**, which tells the coverage tool to run the tests and collect coverage data, then unittest will automatically discover and run all test cases it finds in the project.

Current test coverage:

Coverage report: 92%

coverage.py v7.2.7,

Module	*statements*	*missing*	*excluded*	*coverage*
Shortest_path.py	74	7	0	91%
test_shortest_path.py	81	6	0	93%
Total	**155**	**13**	**0**	**92%**

coverage.py v7.2.7,

How can the tests be repeated:

If set up a testing environment to execute all those tests, place all test files in one folder, write command to run them, whenever need to repeat the tests, just run the command and review the test results.