

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ

імені ІГОРЯ СІКОРСЬКОГО”

Факультет прикладної математики

Кафедра програмного забезпечення комп’ютерних систем

**Курсова робота**

з дисципліни: «Компоненти програмної інженерії»

на тему: «Програмний додаток системи файлів, що міститься у Docker-контейнері, розроблення інтерфейсу командного рядка та тести написані з використанням Robot фреймворку»

Виконала

студентка IІІ курсу

групи КП-02

Гаценко Марія Андріївна

Перевірив

Погорєлов Володимир Володимирович

Київ 2023

**Код**

|  |
| --- |
| **app\_connector.py** |
| **import requests**  **from resources.status\_codes import \***  **from resources.params import \***  **from resources.routes import BASE\_ROUTE**    **BASE = "http://127.0.0.1:5000/"**      **def post(json\_data, file\_type: str, file\_name: str):**  **if file\_type == "directory":**  **response = requests.post(BASE + file\_type + "/" + file\_name, {"parent": json\_data["parent"]})**  **else:**  **response = requests.post(BASE + file\_type + "/" + file\_name, {"content": json\_data["content"], "parent": json\_data["parent"]})**    **if response.status\_code == STATUS\_NOT\_FOUND:**  **raise Exception(ERROR\_NOT\_FOUND)**    **if response.status\_code < STATUS\_SUCCESSFULL or response.status\_code > 300:**  **response\_body = response.json()**  **err = response\_body[DATA\_PARAM][ERROR\_PARAM]**  **raise Exception(f'error: {err}')**    **print(response.json())**      **def delete(file\_type: str, file\_name: str):**  **response = requests.delete(BASE + file\_type + "/" + file\_name)**    **if response.status\_code == STATUS\_NOT\_FOUND:**  **raise Exception(ERROR\_NOT\_FOUND)**    **if response.status\_code < STATUS\_SUCCESSFULL or response.status\_code > 300:**  **response\_body = response.json()**  **err = response\_body[DATA\_PARAM][ERROR\_PARAM]**  **raise Exception(f'error: {err}')**  **print(response.json())**      **def put(json\_data, file\_type: str, file\_name: str):**  **response = requests.put(BASE + file\_type + "/" + file\_name, {"name": file\_name, "content": json\_data["content"], "parent": json\_data["parent"]})**    **if response.status\_code == STATUS\_NOT\_FOUND:**  **raise Exception(ERROR\_NOT\_FOUND)**    **if response.status\_code < STATUS\_SUCCESSFULL or response.status\_code > 300:**  **response\_body = response.json()**  **err = response\_body[DATA\_PARAM][ERROR\_PARAM]**  **raise Exception(f'error: {err}')**  **print(response.json())**      **def get(file\_type, file\_name):**  **print(BASE + file\_type + "/" + file\_name)**  **response = requests.get(BASE + file\_type + "/" + file\_name)**    **if response.status\_code == STATUS\_NOT\_FOUND:**  **raise Exception(ERROR\_NOT\_FOUND)**    **if response.status\_code < STATUS\_SUCCESSFULL or response.status\_code > 300:**  **response\_body = response.json()**  **err = response\_body[DATA\_PARAM][ERROR\_PARAM]**  **raise Exception(f'error: {err}')**    **print(response.json())** |
| **command\_processor.py** |
| **import click**  **import internal.app\_connector as app\_connector**  **from resources.routes import \***  **from resources.params import \***      **# Main route**  **@click.group()**  **def process\_command():**  **pass**      **@process\_command.group()**  **def create():**  **pass**      **# Create subgroup**  **@click.argument('parent')**  **@click.argument('NODE\_NAME')**  **@create.command()**  **def directory(node\_name, parent):**  **json\_data = {"parent": parent}**    **try:**  **app\_connector.post(json\_data, DIR\_ROUTE, node\_name)**  **except Exception as e:**  **print(e)**  **return**    **print(SUCCESS\_OPERATION)**      **@click.argument('PARENT')**  **@click.argument('CONTENT')**  **@click.argument('NODE\_NAME')**  **@create.command()**  **def binary\_file(node\_name, content, parent):**  **json\_data = {**  **"content": content,**  **"parent": parent**  **}**    **try:**  **app\_connector.post(json\_data, BIN\_FILE\_ROUTE, node\_name)**  **except Exception as e:**  **print(e)**  **return**    **print(SUCCESS\_OPERATION)**      **@click.argument('PARENT')**  **@click.argument('CONTENT')**  **@click.argument('NODE\_NAME')**  **@create.command()**  **def log\_file(node\_name, content, parent):**  **json\_data = {**  **"content": content,**  **"parent": parent**  **}**    **try:**  **app\_connector.post(json\_data, LOG\_FILE\_ROUTE, node\_name)**  **except Exception as e:**  **print(e)**  **return**    **print(SUCCESS\_OPERATION)**      **@click.argument('PARENT')**  **@click.argument('CONTENT')**  **@click.argument('NODE\_NAME')**  **@create.command()**  **def buffer\_file(node\_name, content, parent):**  **json\_data = {**  **"content": content,**  **"parent": parent**  **}**    **try:**  **app\_connector.post(json\_data, BUF\_FILE\_ROUTE, node\_name)**  **except Exception as e:**  **print(e)**  **return**    **print(SUCCESS\_OPERATION)**      **# Other methods**  **@click.argument('NAME\_NODE')**  **@click.argument('FILE\_TYPE')**  **@process\_command.command()**  **def delete(name\_node, file\_type):**  **try:**  **app\_connector.delete(file\_type, name\_node)**  **except Exception as e:**  **print(e)**  **return**    **print(SUCCESS\_OPERATION)**      **@click.argument('FILE\_TYPE')**  **@click.argument('NAME\_NODE')**  **@process\_command.command()**  **def read(name\_node, file\_type):**  **try:**  **app\_connector.get(file\_type, name\_node)**  **except Exception as e:**  **print(e)**  **return**      **@click.argument('CONTENT')**  **@click.argument('PARENT')**  **@click.argument('NAME\_NODE')**  **@click.argument('FILE\_TYPE')**  **@process\_command.command()**  **def write(file\_type, name\_node, parent, content):**  **if file\_type == "directory":**  **print("Content of a directory should be changed by modifying files.")**  **return**    **json\_data = {**  **"parent": parent,**  **"content": content,**  **}**    **try:**  **app\_connector.put(json\_data, file\_type, name\_node)**  **except Exception as e:**  **print(e)**  **return**    **print(SUCCESS\_OPERATION)**      **def process():**  **process\_command()** |

**Висновки**

Метою курсової роботи було розроблення системи файлів, розташованої у Docker-контейнері, розроблення інтерфейсу командного рядка та його тестування за допомогою Robot framework.