**DOKUMENTACJA**

**PROJEKTU**

**Technologia: REST (Http)**

**Język programowania: Python**

**Skład osobowy grupy:**

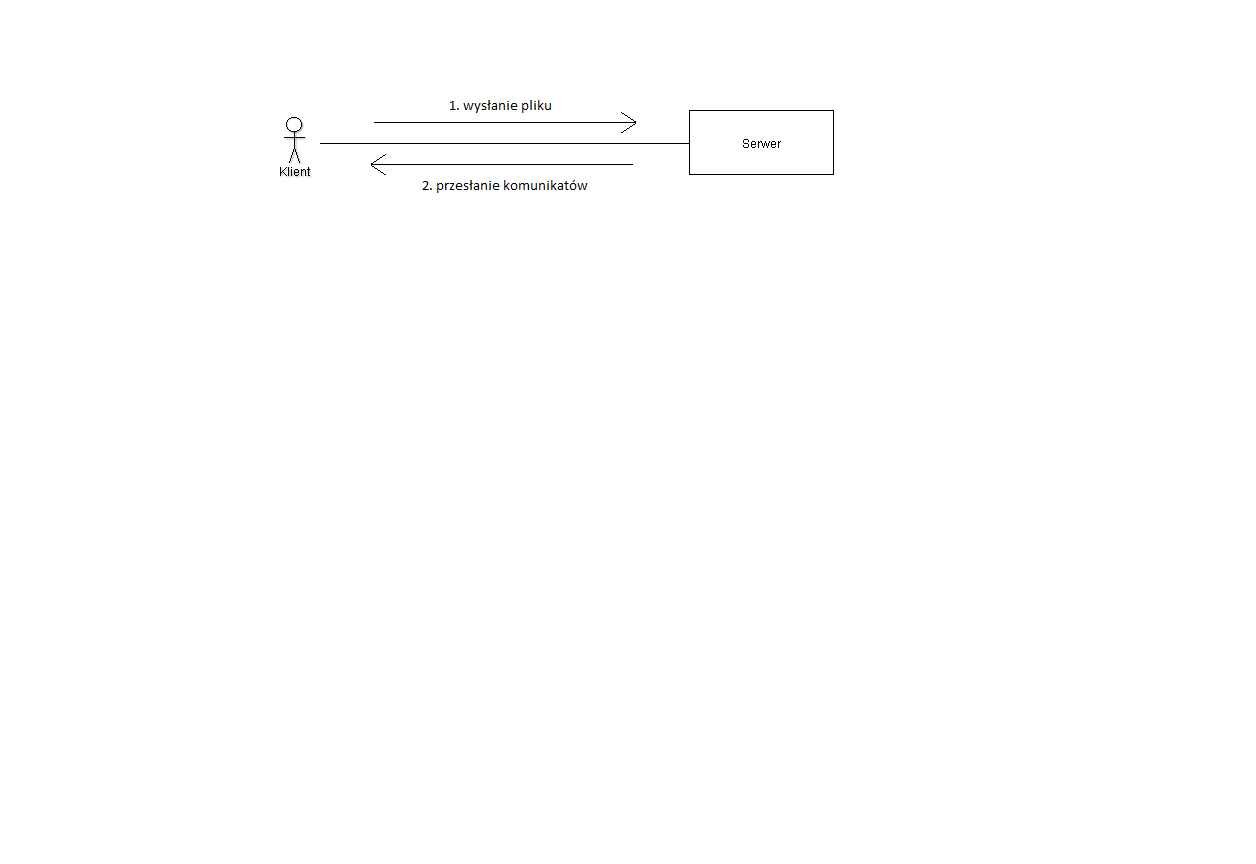
* **Kiełbania Piotr**
* **Kowalska Paulina**
* **Kurkowski Wojciech**
* **Machalska Anna**
* **Mazurek Paweł**
* **Pasierbiewicz Karolina**

**Analiza dostępności**

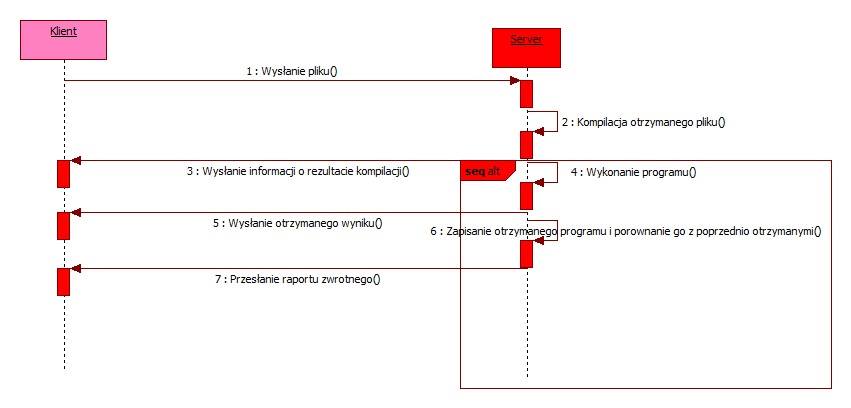
**REST** (ang. **Re**presentational **S**tate **T**ransfer) jest wzorcem narzucającym dobre praktyki tworzenia architektury aplikacji rozproszonych. Jest jednym z najpopularniejszych technologii wykorzystanych do implementacji wspomnianych aplikacji rozproszonych. Dostępność technologii REST dla wielu języków programowania w tym języka Python daje możliwość szybkiego i łatwego zaimplementowania prostej aplikacji rozproszonej wykonującej założone zadania przez projektanta.

**Diagramy UML**

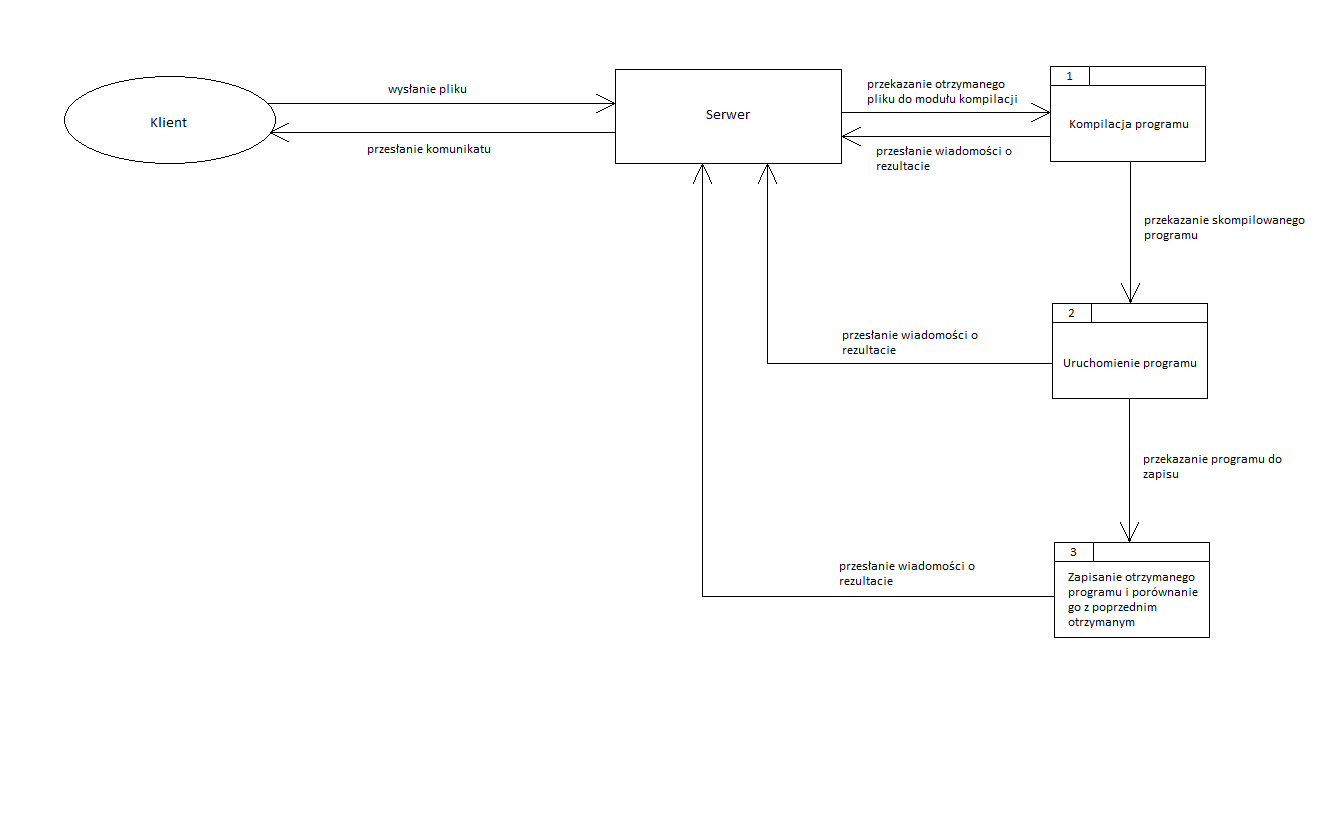
* **Diagram komunikacji**

****

* **Diagram sekwencji**

****

* **Diagram przepływu danych**

****

**Testy oprogramowania**

**Kody źródłowe oraz skrypty**

**Server.py**

**from** http.server **import** HTTPServer, SimpleHTTPRequestHandler  
**from** socketserver **import** ThreadingMixIn  
  
*#from PythonREST import executor, compiler, reporter***import** executor, compiler, reporter  
  
  
*# import executor, compiler  
# import threading***class** Server(ThreadingMixIn, HTTPServer):  
 **def** run(self, IP, PORT):  
 *"""  
 starts server at given host and port* **:param** *IP: host's IP to start server* **:param** *PORT: server's port to listen requests* **:return***: None  
 """* server = Server((IP, PORT), MyHandler)  
 print(**"server started."**)  
 server.serve\_forever()  
  
  
**class** MyHandler(SimpleHTTPRequestHandler):  
 **def** \_set\_headers(self):  
 *"""sets headers"""* self.send\_response(200)  
 self.send\_header(**'content-type'**, **'text/html'**)  
 self.end\_headers()  
  
 **def** do\_PUT(self):  
 length = int(self.headers[**'Content-Length'**])  
 content = self.rfile.read(length)  
 self.\_set\_headers()  
 self.wfile.write(**"file successfully received\n"**.encode())  
  
 **try**:  
 c = compiler.Compiler()  
  
 c.compile\_file(content)  
 self.wfile.write(**"compilation successful\n"**.encode())  
 **except** Exception **as** e:  
 self.wfile.write(**"compilation failed. Reason:"**.\_\_add\_\_(e.args).encode())  
  
 test, result = executor.execute\_program(content)  
 **if** test:  
 self.wfile.write(**"program has been executed successfully and returned: "**.\_\_add\_\_(str(result)).encode())  
 **else**:  
 self.wfile.write(**"program execution failed "**.\_\_add\_\_(result).encode())  
  
 r = reporter.Reporter()  
 name = r.save\_program(content)  
 result = r.compare(name)  
 self.wfile.write(**"\nfiles with same content actually saved on server:\n"**.\_\_add\_\_(**"\n"**.join(result)).encode())

Klasa Server pozwala na uruchomić serwer na danym komputerze podając jego adres IP oraz port. Jeśli wywołanie metody Server((IP, PORT), MyHandler)gdzie argumentami jest adres IP, port oraz klasa MyHandler.

Klasa MyHandler odpowiedzialna jest za komunikację z połączonym klientem i odebraniem żądania od niego. Metoda \_set\_headers ustawia nagłówki odpowiedzi na żądanie. Metoda do\_PUT odpowiada za:

* Przygotowanie i wysłania komunikatu zwrotnego do klienta o odebranym pliku,
* Wywołania metody compile\_file z klasy compiler odpowiedzialnej za kompilację otrzymanego programu i wysłanie komunikatu o powodzeniu lub niepowodzeniu operacji do klienta,
* Wywołania metody execute\_program z klasy executor odpowiedzialnej za uruchomienie skompilowanego programu i powiadomienie o wyniku klienta,
* Zapisanie i porównanie otrzymanych programów przez klienta za pomocą metod save\_program oraz compare i powiadomienie użytkownika o wyniku operacji.

**Analiza porównawcza implementacji**

**Bibliografia**

UML:

http://zasoby.open.agh.edu.pl/~09sbfraczek/

REST

<http://www.yarpo.pl/2012/07/29/rest-ciekawszy-sposob-na-komunikacje-client-server/>

<https://www.fullstackpython.com/api-creation.html>

<https://techietweak.wordpress.com/2015/03/30/http-restful-api-with-python-requests-library/>

<https://developers.google.com/drive/v3/web/quickstart/python>

<http://documentation.commvault.com/commvault/v10/article?p=features/rest_api/rest_api_getting_started_python.htm>