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"***

Виконав: Винничук Богдан

Група: КП-03

Перевірив: Погорєлов Володимир

Володимирович

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1 семестр 2022/2023

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«Київський політехнічний інститут імені Ігоря Сікорського»

ФАКУЛЬТЕТ ПРИКЛАДНОЇ МАТЕМАТИКИ

Кафедра програмного забезпечення комп’ютерних систем

| Узгоджено  Керівник роботи  \_\_\_\_\_\_\_\_\_\_/Погорєлов.В.В./ | ЗАХИЩЕНА "\_\_"\_\_\_\_\_\_\_\_\_2022р.  з оцінкою\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  \_\_\_\_\_\_\_\_\_\_/ Погорєлов.В.В./ |
| --- | --- |

**Програмний додаток серверу що міститься у Docker-контейнері, та тести написані з використанням Robot framework**

Виконавець роботи

Винничук Богдан

\_\_\_\_\_\_\_\_\_\_\_\_\_\_2022р.

**Код**

| api.py(сервер) |
| --- |
| from flask import Flask, request  from flask import Flask, request  import json  from flask\_restful import Resource, Api  from directory import Directory  from binaryFile import BinaryFile  from logTextFile import LogTextFile  from bufferFile import BufferFile  app = Flask(\_\_name\_\_)  api = Api(app)  fatherDirectory = Directory('fatherDir')  fileName = 'binary file'  content = 'binary content'  binary = BinaryFile(fileName, content, fatherDirectory)  name = 'buffer file'  size = 10  buffer = BufferFile(name, size)  maxElements = 10  name = 'name1'  directory = Directory(name, maxElements)  name = 'name1'  log = LogTextFile(name, fatherDirectory)  class BinaryApi(Resource):  def \_\_init\_\_(self):  self.binary = binary  def get(self):  return self.binary.\_\_read\_\_()  def post(self):  data = request.get\_json()  fatherDir = Directory(data["father"])  self.binary = BinaryFile(data["fileName"], data["content"], fatherDir)  return {'message': 'BinaryFile is successfully created'}  def put(self):  data = request.get\_json()  fatherDir = Directory(data["father"])  return self.binary.\_\_move\_\_(fatherDir)  def delete(self):  return self.binary.\_\_delete\_\_()  class BufferApi(Resource):  def \_\_init\_\_(self):  self.buffer = buffer  def get(self):  return self.buffer.\_\_consume\_\_()  def post(self):  data = request.get\_json()  fatherDir = Directory(data["father"])  self.buffer = BufferFile(data["fileName"], data["maxSize"], fatherDir)  return {'message': 'BufferFile is successfully created'}  def put(self):  data = request.get\_json()  fatherDir = Directory(data["father"])  return self.buffer.\_\_move\_\_(fatherDir)  def patch(self):  data = json.loads(request.data)  return self.buffer.\_\_push\_\_(data["element"])  def delete(self):  return self.buffer.\_\_delete\_\_()  class DirectoryApi(Resource):  def \_\_init\_\_(self):  self.directory = directory  def post(self):  data = request.get\_json()  self.directory = Directory(data["name"], data["maxElements"])  return {'message': 'Directory is successfully created'}  def put(self):  data = request.get\_json()  fatherDir = Directory(data["father"])  return self.directory.\_\_move\_\_(fatherDir)  def delete(self):  return self.directory.\_\_delete\_\_()  class logTextApi(Resource):  def \_\_init\_\_(self):  self.logText = LogTextFile  def get(self):  return self.logText.\_\_read\_\_()  def post(self):  data = request.get\_json()  fatherDir = Directory(data["father"])  self.logText = LogTextFile(data["fileName"], fatherDir)  return {'message': 'LogTextFile is successfully created'}  def put(self):  data = request.get\_json()  fatherDir = Directory(data["father"])  return self.logText.\_\_move\_\_(fatherDir)  def patch(self):  data = request.get\_json()  return self.logText.\_\_log\_\_(data["line"])  def delete(self):  return self.logText.\_\_delete\_\_()  api.add\_resource(BinaryApi, '/binaryfile')  api.add\_resource(BufferApi, '/bufferfile')  api.add\_resource(DirectoryApi, '/directory')  api.add\_resource(logTextApi, '/logtextfile')  if \_\_name\_\_ == '\_\_main\_\_':  app.run(host='0.0.0.0') |

| main.py (cli) |
| --- |
| import sys  import pip.\_vendor.requests as requests  n = len(sys.argv)  Request = "http://localhost:8888/" + sys.argv[2] + "?"  for i in range(3, n):  Request += sys.argv[i]  if i != n-1:  Request += "&"  response = ""  if sys.argv[1] == "get":  response = requests.get(Request)  if sys.argv[1] == "post":  response = requests.post(Request)  if sys.argv[1] == "patch":  response = requests.patch(Request)  if sys.argv[1] == "delete":  response = requests.delete(Request)  print("Status code:", response.status\_code)  print("Response:", response.json()) |

**Висновок**

Метою даної курсової роботи було розроблення серверу на мові python, розташування його у докер контейнері, розроблення інтерфейсу командного рядка для створення запитів до серверу та тестування за допомогою Robot framework.