|  |  |  |  |
| --- | --- | --- | --- |
| МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ імені Тараса Шевченка ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ  **Кафедра програмних систем і технологій**  Дисципліна  **«**Хмарні технології**»**  **Лабораторна робота № 4**  «Створення Application Programming Interface для хмарного додатку» | | | |
| **Виконав:** | Гоша Давід | **Перевірив**: |  |
| Група | ІПЗ-43 | Дата перевірки |  |
| Форма навчання | денна | Оцінка |  |
| Спеціальність | 121 |
| 2022 | | | |

## **Мета роботи**

Створити простий HTTP API для хмарного додатку. API має задовольняти REST вимогам побудови інтерфейсів. Продемонструвати роботу API на прикладі 3-4 методів. Кожен метод перевірити з клієнту за допомогою curl або інших аналогічних програм.

## **Завдання роботи**

1. Розробити опис методів API.
2. Підготувати опис формату даних у методах.
3. Підготувати приклади клієнтських запитів у системі curl або аналогічних.
4. Розробити код програми хмарного додатку.
5. Виконати тестування запитів API для роботи з хмарним додатком.

## **Опис методів API**

1. **GET /users** - отримати список всіх користувачів.
2. **POST /users** - створити нового користувача.
3. **GET /users/{id}** - отримати інформацію про конкретного користувача за його ID.
4. **PUT /users/{id}** - оновити інформацію про користувача.
5. **DELETE /users/{id}** - видалити користувача.

## **Формат даних у методах**

* **GET /users** - повертає список користувачів у форматі JSON.
* **POST /users** - приймає дані користувача у форматі JSON і повертає ID нового користувача.
* **GET /users/{id}** - повертає інформацію про користувача у форматі JSON.
* **PUT /users/{id}** - приймає оновлені дані користувача у форматі JSON.
* **DELETE /users/{id}** - видаляє користувача і повертає статус операції.

## **Приклади клієнтських запитів**

1. curl -X GET http://<your-api-endpoint>/users
2. curl -X POST -H "Content-Type: application/json" -d '{"name": "John", "age": 30}' http://<your-api-endpoint>/users
3. curl -X GET http://<your-api-endpoint>/users/1
4. curl -X PUT -H "Content-Type: application/json" -d '{"name": "John", "age": 31}' http://<your-api-endpoint>/users/1
5. curl -X DELETE http://<your-api-endpoint>/users/1

## **Код програми хмарного додатку**

from flask import Flask, jsonify, request

app = Flask(\_\_name\_\_)

users = [

{"id": 1, "name": "John", "age": 30},

{"id": 2, "name": "Jane", "age": 28},

]

@app.route("/users", methods=["GET"])

def get\_users():

return jsonify(users)

@app.route("/users", methods=["POST"])

def create\_user():

user = request.json

user["id"] = len(users) + 1

users.append(user)

return jsonify(user["id"])

@app.route("/users/<int:user\_id>", methods=["GET"])

def get\_user(user\_id):

user = next((user for user in users if user["id"] == user\_id), None)

if user is None:

return jsonify({"error": "User not found"}), 404

return jsonify(user)

@app.route("/users/<int:user\_id>", methods=["PUT"])

def update\_user(user\_id):

user = next((user for user in users if user["id"] == user\_id), None)

if user is None:

return jsonify({"error": "User not found"}), 404

updated\_user = request.json

user.update(updated\_user)

return jsonify(user)

@app.route("/users/<int:user\_id>", methods=["DELETE"])

def delete\_user(user\_id):

global users

users = [user for user in users if user["id"] != user\_id]

return jsonify({"status": "success"})

if \_\_name\_\_ == "\_\_main\_\_":

app.run(debug=True)

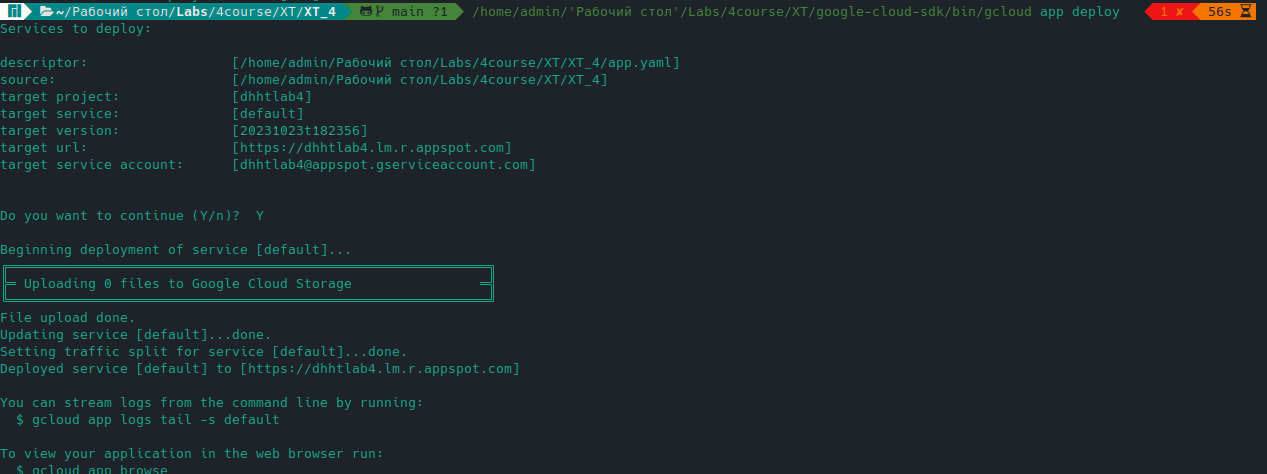
## **Інструкція для публікації на Google Cloud**

1. Створіть проект на Google Cloud Platform.
2. Встановіть і налаштуйте Google Cloud SDK на вашому комп'ютері.
3. Авторизуйтеся в Google Cloud SDK за допомогою команди **gcloud auth login**.
4. Виберіть ваш проект за допомогою команди **gcloud config set project <your-project-id>**.
5. Розгорніть ваш додаток на Google Cloud за допомогою команди **gcloud app deploy**.

Після розгортання, ваш API буде доступний за URL, який ви отримаєте в консолі Google Cloud.

## **Результати тестування API**

Після розгортання додатку на Google Cloud, ви можете виконати тестування API за допомогою запитів curl, які були наведені вище.



## **Висновки**

У ході виконання даної роботи було розроблено простий HTTP API для хмарного додатку, який задовольняє вимогам REST. Було продемонстровано роботу API на прикладі чотирьох методів, кожен з яких було перевірено з клієнту за допомогою програми curl. Також було розроблено код програми хмарного додатку та проведено тестування запитів API.

Робота над проектом дозволила набути практичних навичок використання та налаштування служби збереження даних у хмарному середовищі, а також поглибити знання в області розробки RESTful API. Важливим аспектом роботи було також вивчення можливостей Google Cloud Platform та особливостей розгортання додатків у хмарі.

Загалом, виконана робота відповідає поставленим завданням та меті, і може бути використана як основа для подальшої розробки та вдосконалення хмарного додатку.