

Київський національний університет імені Тараса Шевченка
Факультет інформаційних технологій

Кафедра програмних систем і технологій

Лабораторна робота № 3

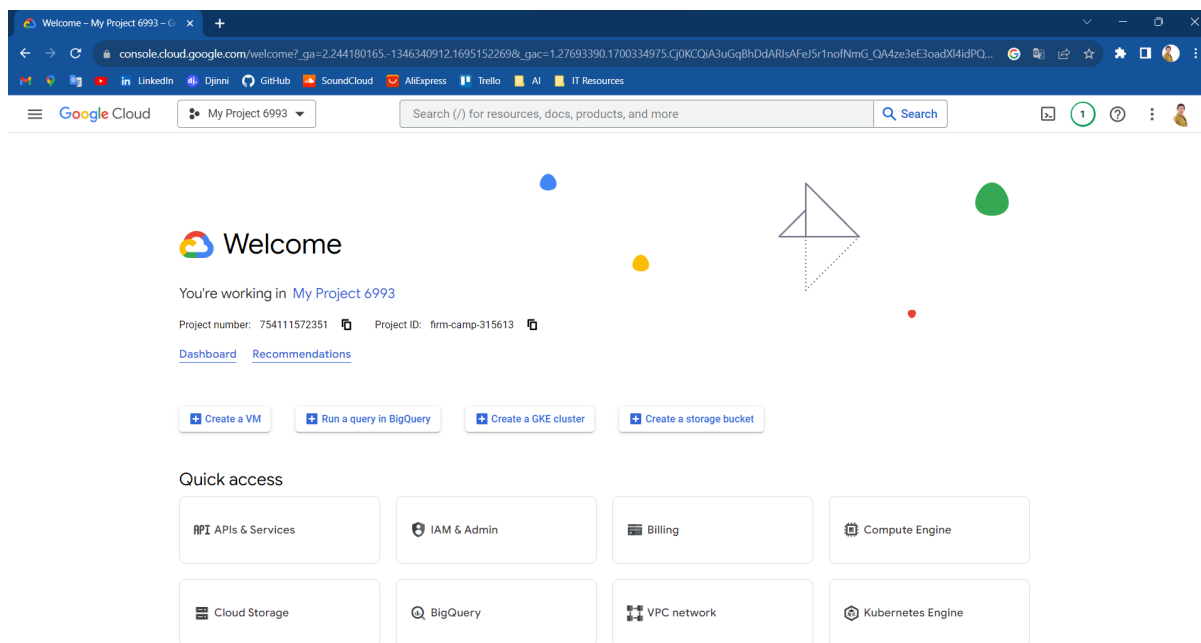
Тема: «Реєстрація та налаштування сервісу хмарних обчислень»

Дисципліна «Хмарні технології»

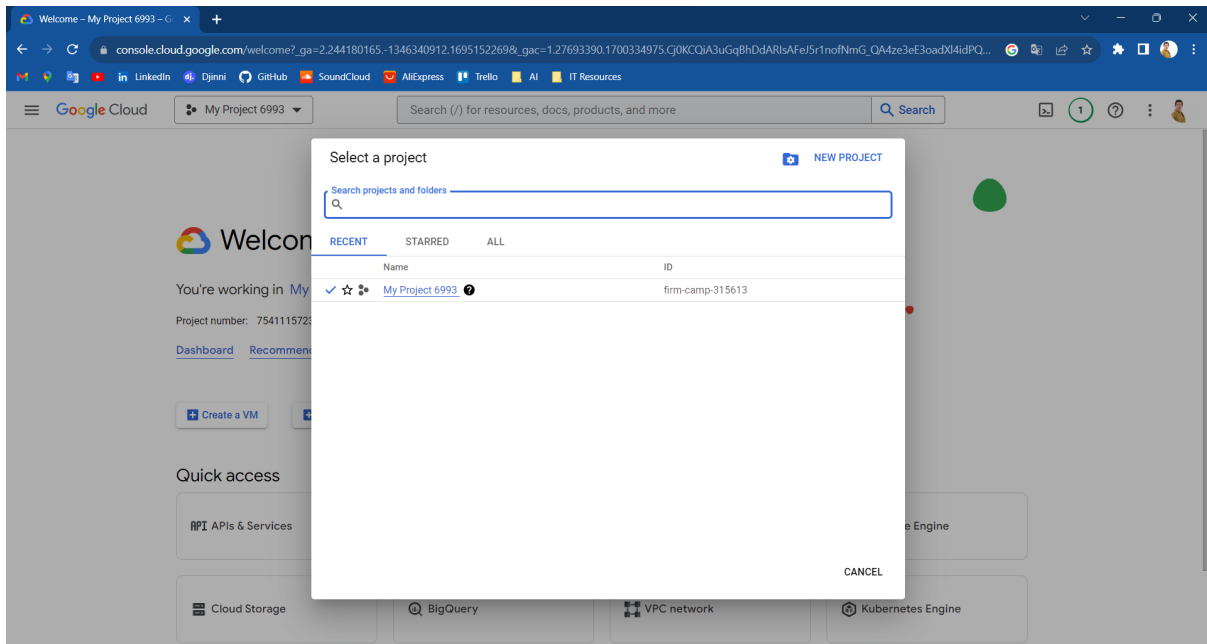
Підготував:
студент гр. ПІЗ-43(1)
Мішак Максим

1. Створення елементу “App Engine”

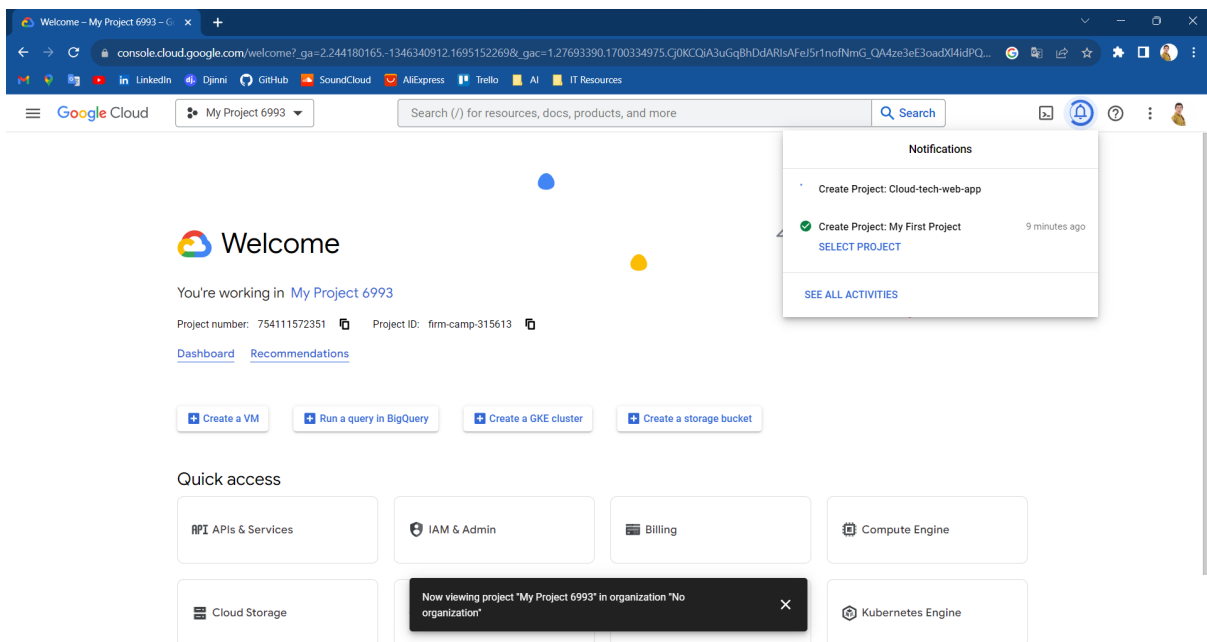
Для створення та розгортання проекту потрібно обрати, який саме елемент використовувати для отримання відповідного результату. В моєму випадку для мінімізації налаштувань, оптимізації витрат та отримання можливості розширення інфраструктури в подальшому, також для отримання потрібних елементів “з коробки” я буду використовувати App Engine. Для створення елементу потрібно зареєструватись на порталі Google Cloud.



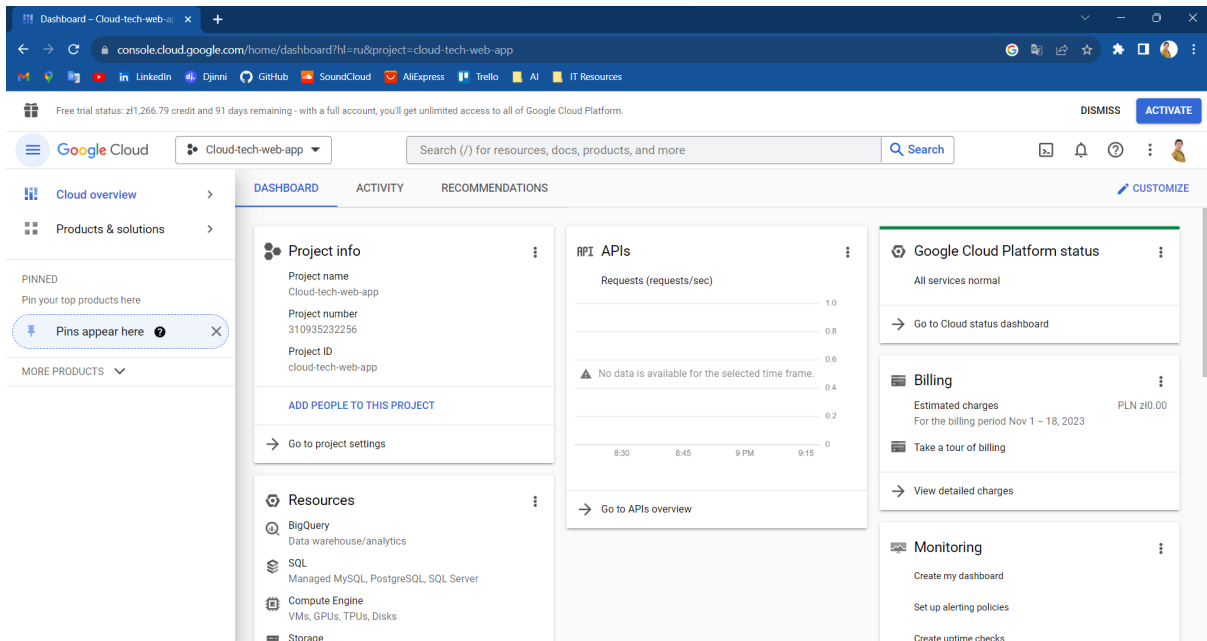
Після реєстрації потрібно створити новий проект, в рамках якого буде відбуватись робота. Для цього натискаємо на створений стандартно проект та написуємо “NEW PROJECT”.



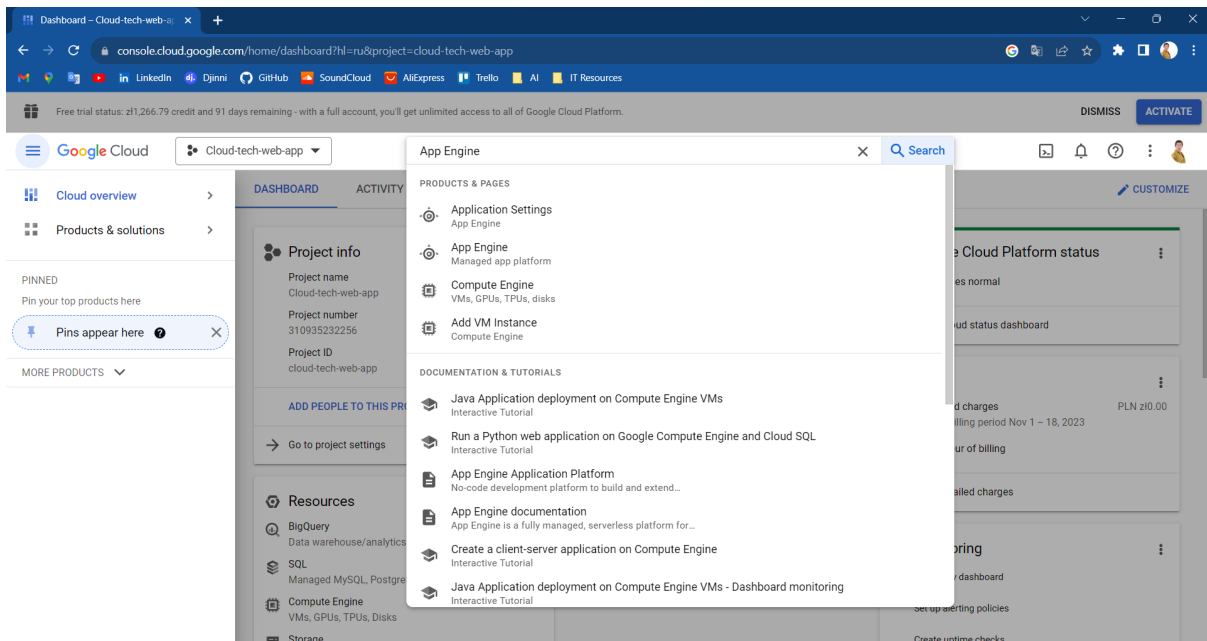
Після натискання заповнюємо форму для створення проекту. Для проекту необхідно заповнити декілька полів: назва проекту, локація серверів. Після цього натискаємо кнопку “Create”, та чекаємо розгортання проекту.



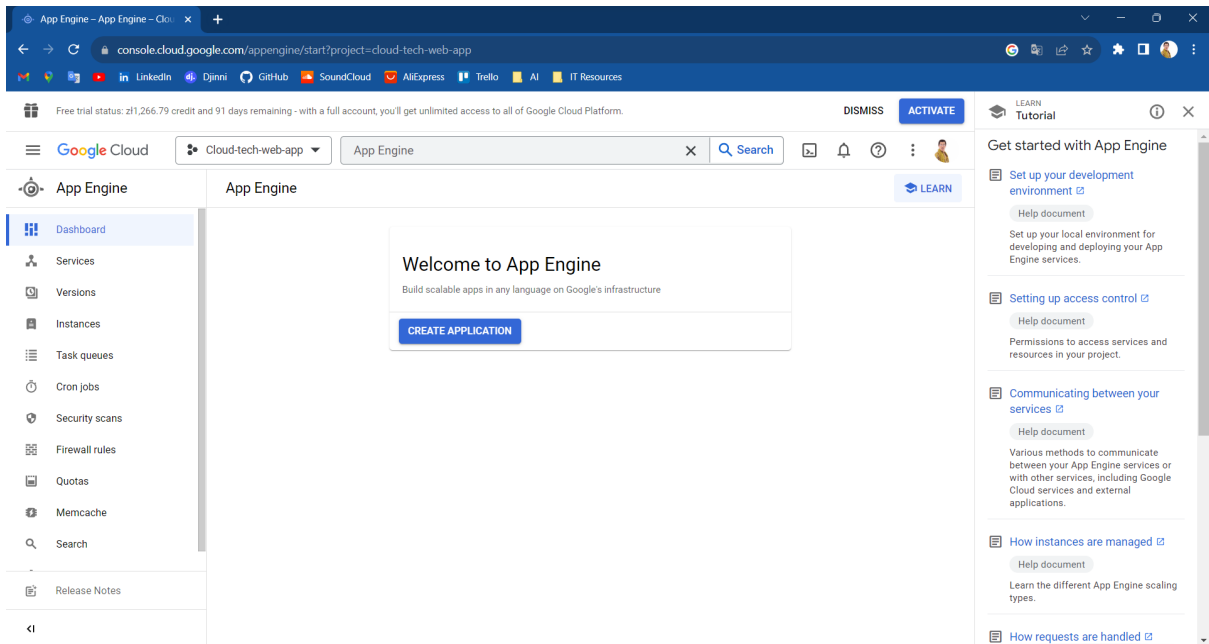
Після успішного розгортання з’явиться повідомлення, про те, що проект готово. Процес може зайняти декілька хвилин. Після розгортання проекту обираємо його, та переходим на його адміністративну панель “Dashboard”.



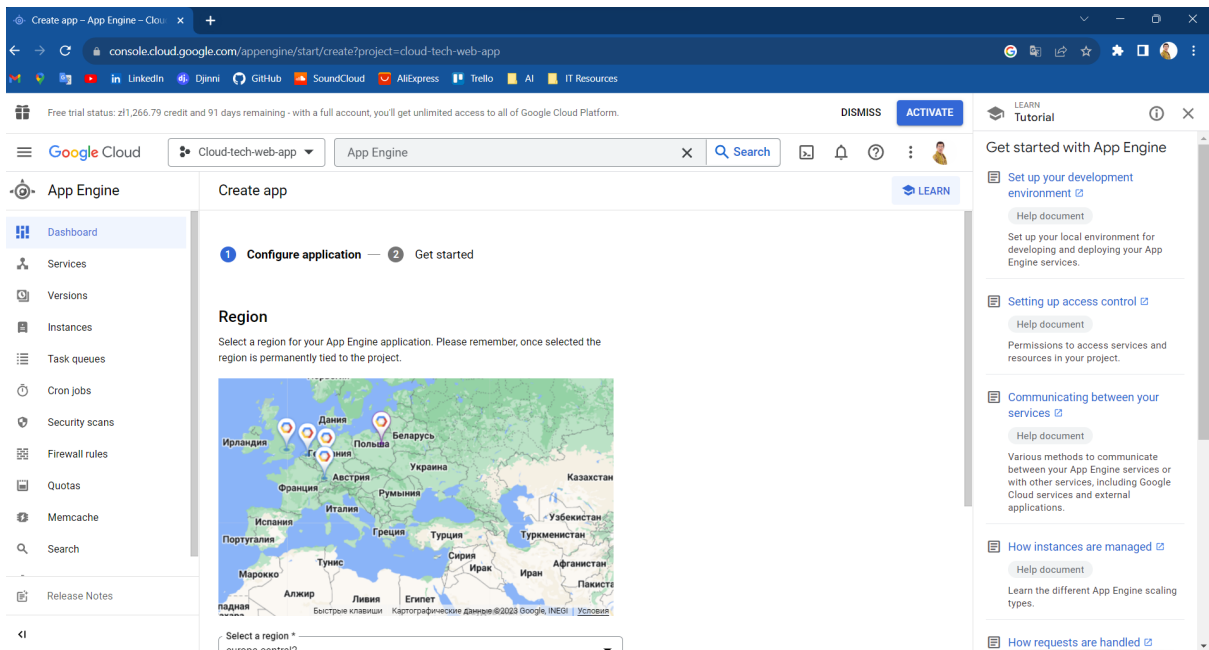
Після того, як ми перейшли в адмін. панель потрібно розгорнути екземпляр відповідного елемента, в моєму випадку це “App Engine”. Для цього в рядку пошуку пишемо назву елемента, та вибираємо перший в списку, який відповідає потребам

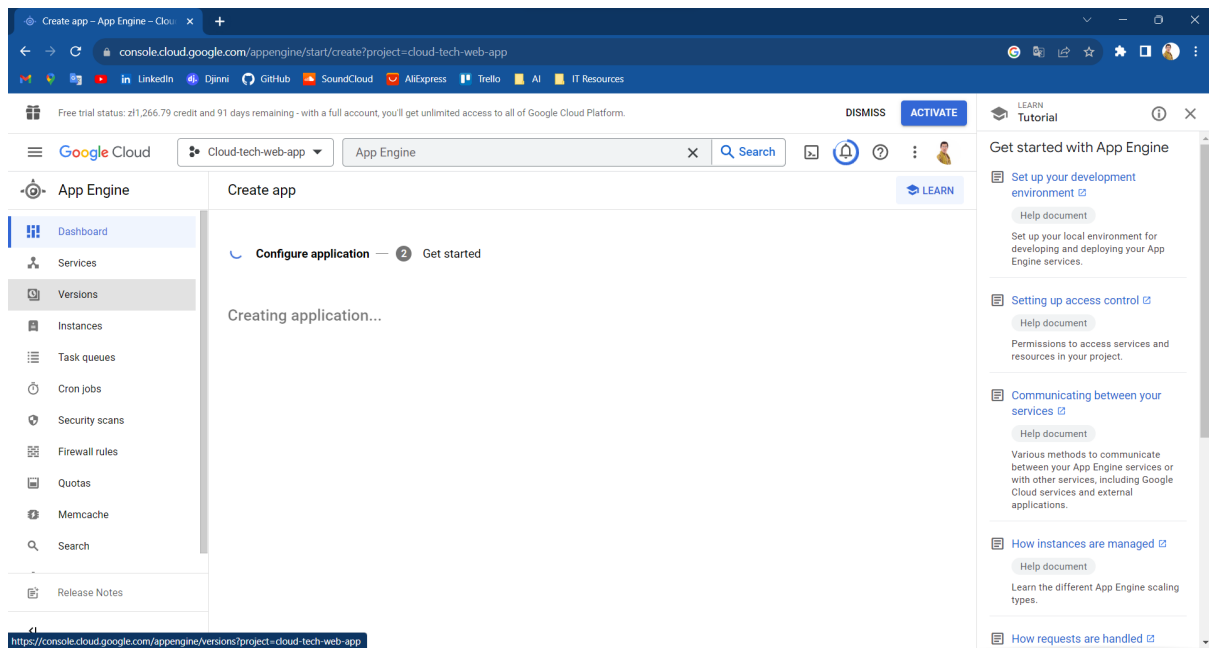


Натискаємо “App Engine” , після чого натискаємо “Create application”

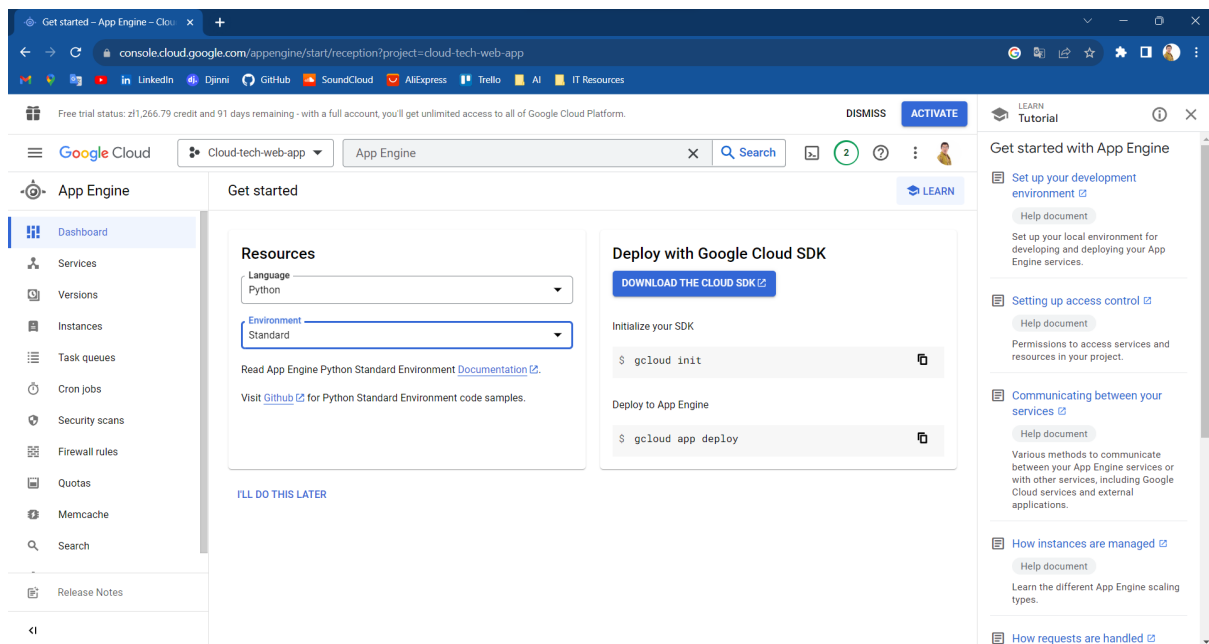


Після натискання, буде розгорнута форма, для заповнення параметрів елемента. У випадку “App Engine” це поля: локація, мова програмування та тип середовища. Я обрав центральну Європу, мову програмування Python, та стандартне середовище.

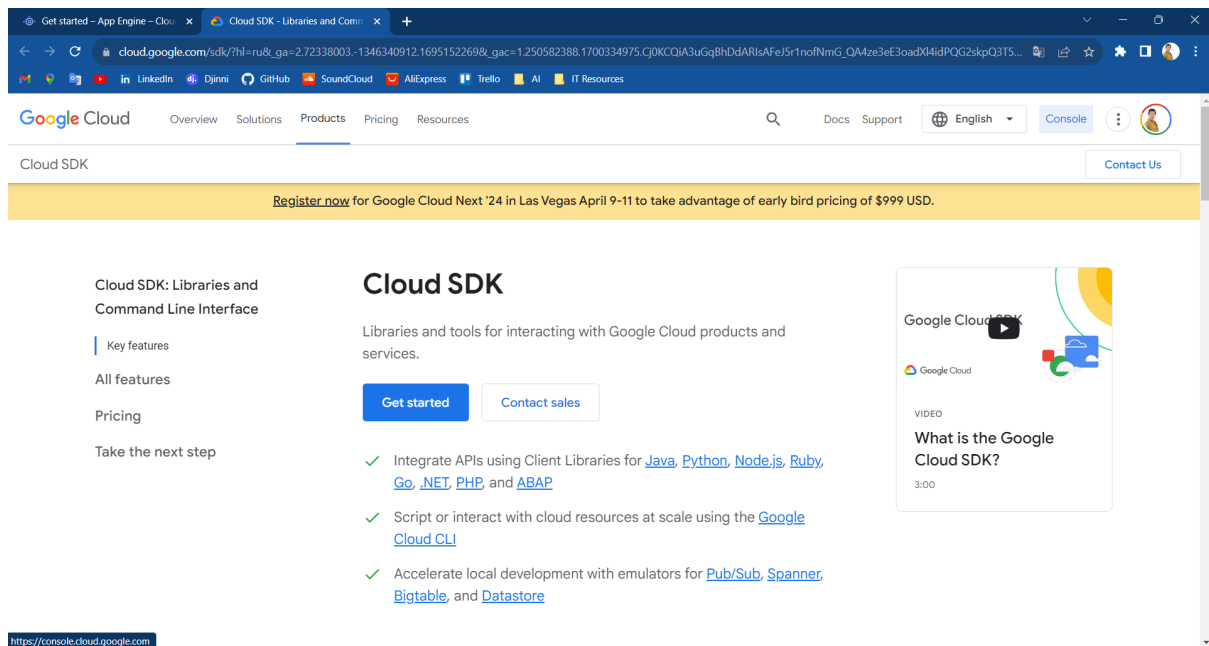




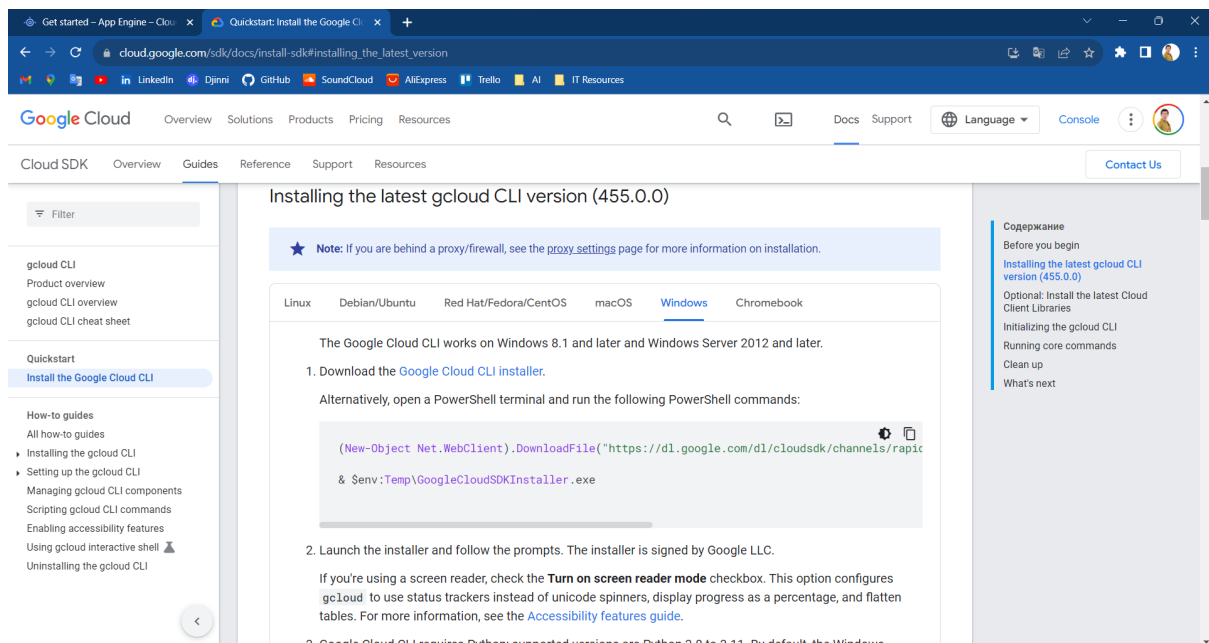
Після вибору локації потрібно зачекати встановлення параметра, після чого автоматично відбувається перехід на наступний екран.



Для зручнішого доступу до проекту з терміналу потрібно встановити SDK для роботи з Google Cloud. Для цього переходимо на “Download the Google SDK”.



Натискаємо “Get started”



Обираємо потрібну нам операційну систему, в моєму випадку Windows та переходимо по посиланню “Google Cloud CLI interface”, або обираємо метод встановлення через термінал. Я скористався першим варіантом. Після завершення завантаження файлу, відкриваємо майстер встановлення. Після чого встановлюємо SDK.



Під час встановлення програмного забезпечення, одним з етапів буде авторизація, яку потрібно пройти від користувача, для якого попередньо створювали App Engine.

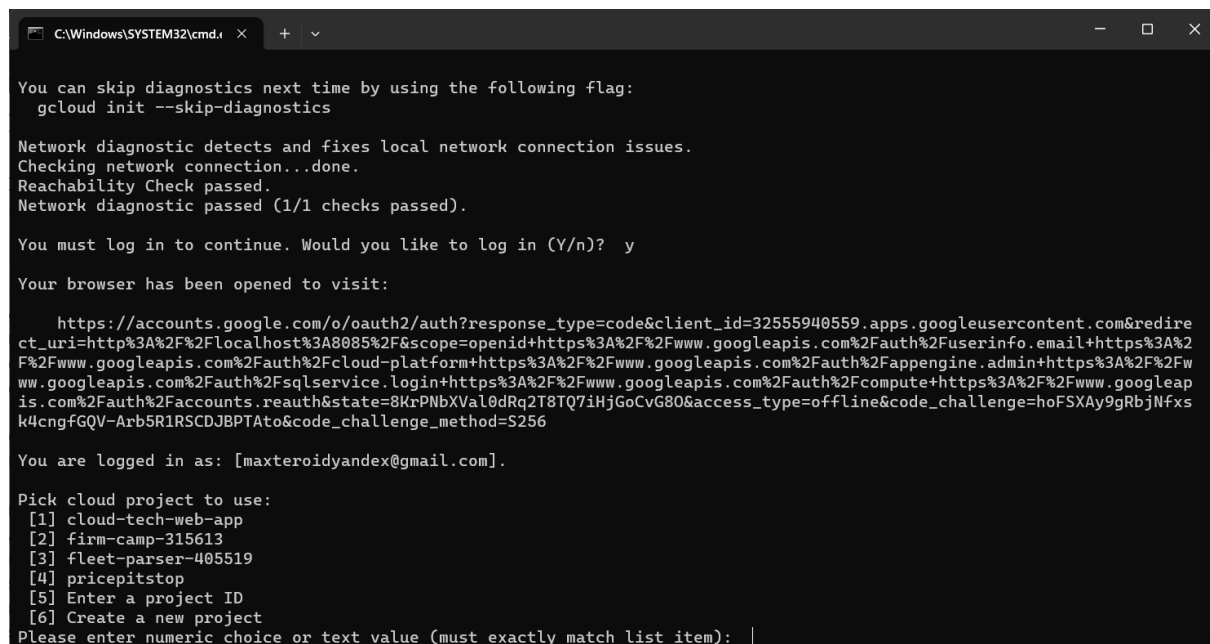
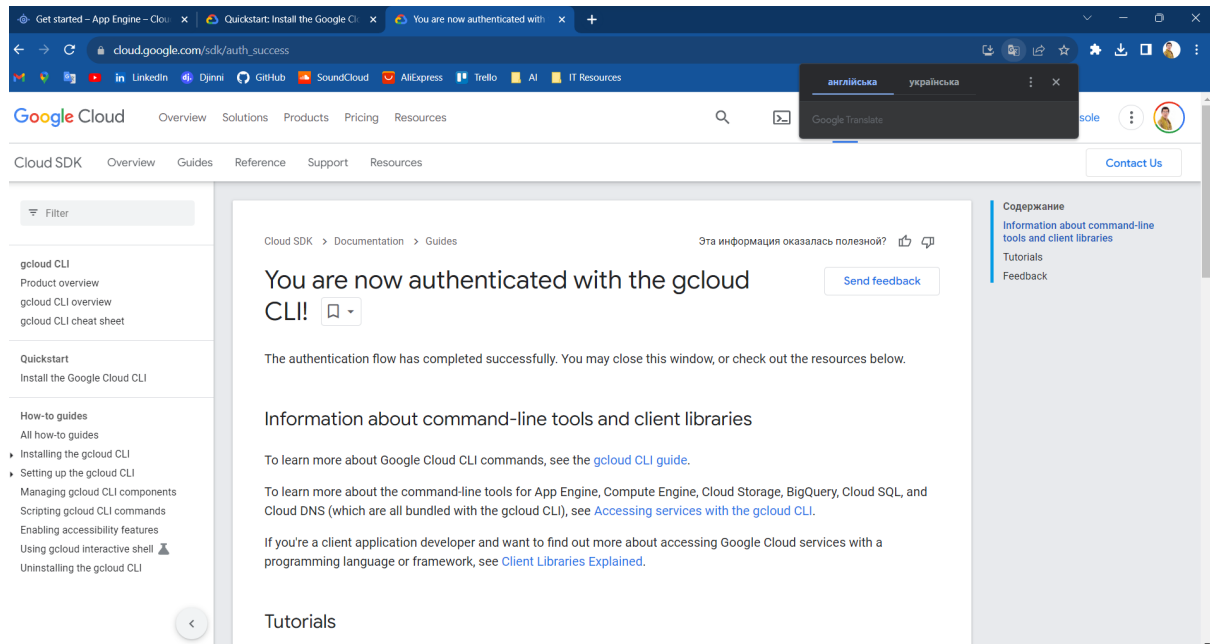
```
C:\Windows\SYSTEM32\cmd.exe
Welcome to the Google Cloud CLI! Run "gcloud -h" to get the list of available commands.
---
Welcome! This command will take you through the configuration of gcloud.

Your current configuration has been set to: [default]

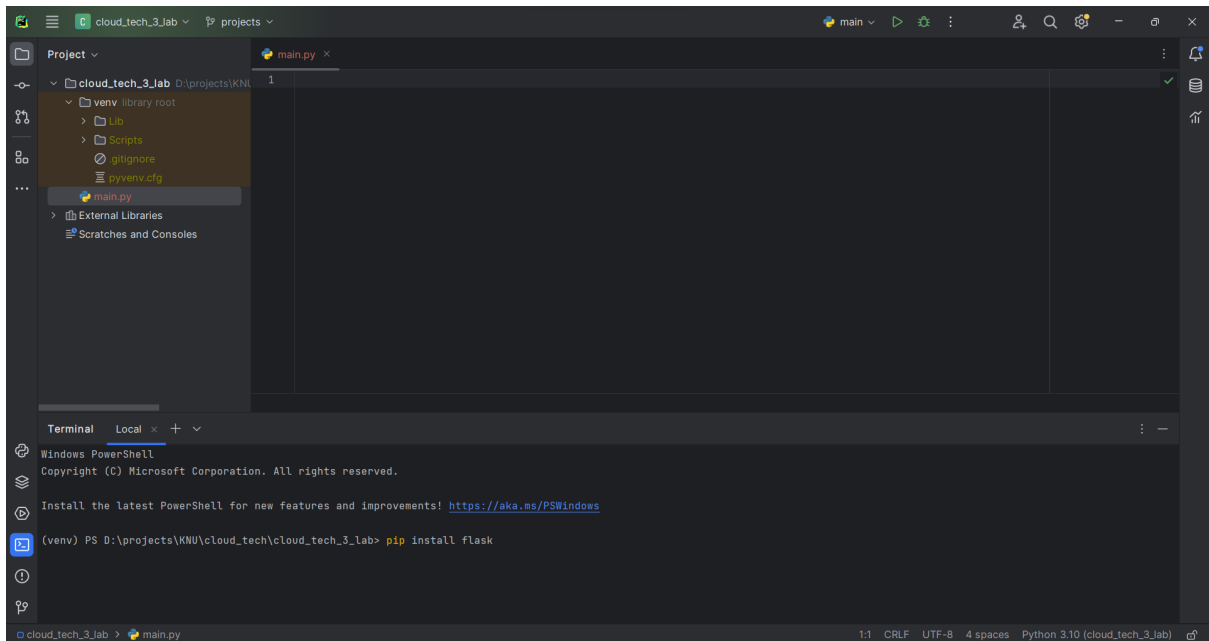
You can skip diagnostics next time by using the following flag:
  gcloud init --skip-diagnostics

Network diagnostic detects and fixes local network connection issues.
Checking network connection...|
```

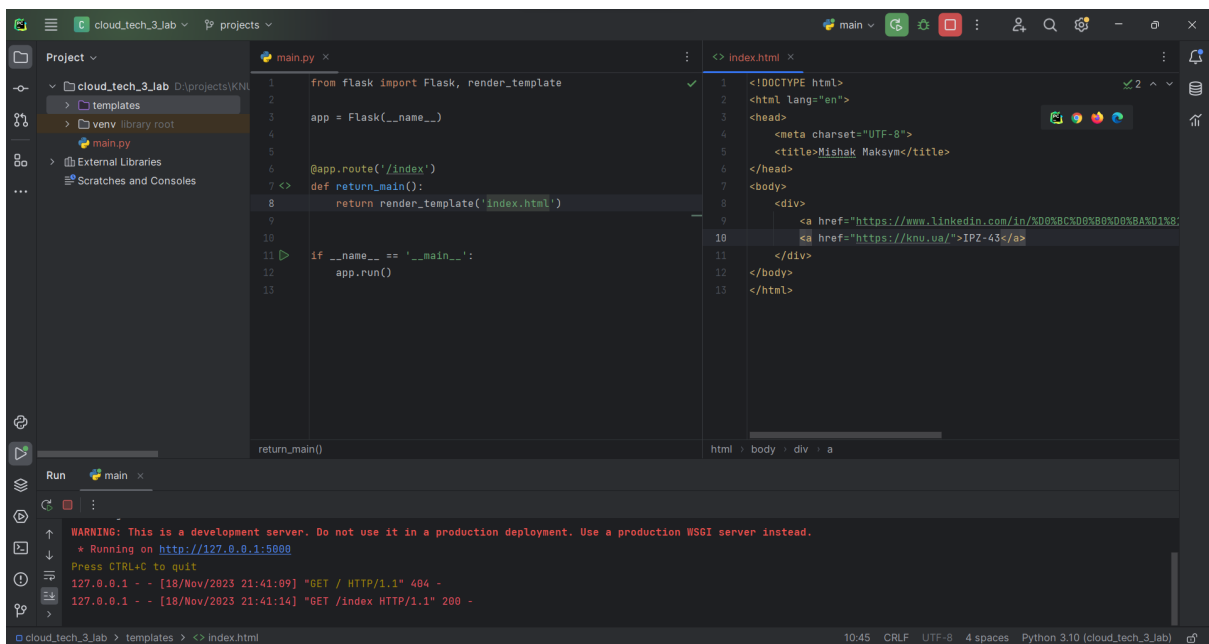

Після успішної авторизації, буде відкрита вітальна сторінка, та нам буде запропоновано декілька проектів на вибір



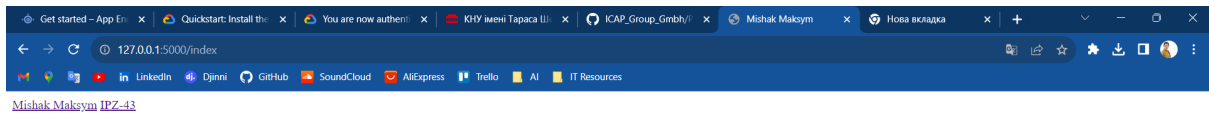
На цьому етапі ми можемо переходити до розробки проекту та розгортання його на сервісі. Для цього, як було зазначено вище - буде використаний фреймворк Flask для мови програмування Python.



Створюємо новий проект, для якого було створене нове віртуальне середовище. Встановлюємо Flask за допомогою менеджера пакетів.



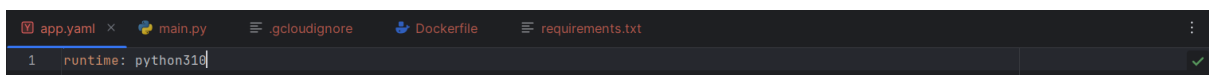
Після чого створюємо простий додаток, який обробляє один ендпоінт, і при переході на нього виводить два посилання. Для цього використано одну функцію обробник, доповнення “render_template” та проста html розмітка. Запускаємо проект локально для перевірки



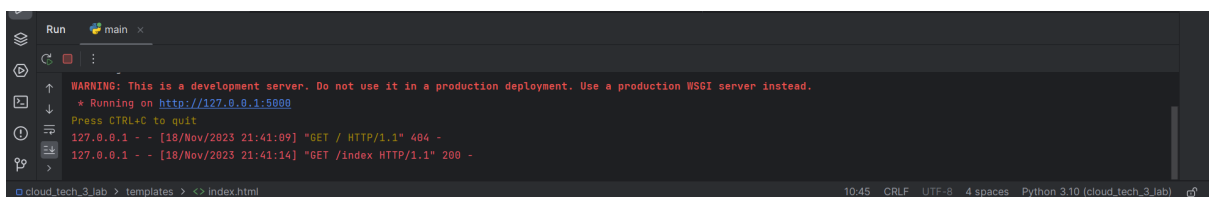
Бачимо, що локально проект працює, ендпоінт обробляється. Після цього можна приступати для написання інфраструктури. Для цього треба дізнатись декілька деталей. В моєму випадку - версію Python.



Перш за все треба створити файл конфігурації `yaml`, в якому буде вказано версію Python.



Також потрібно створити `Dockerfile`, який потрібний для використання `gunicorn`. Проекти написані на Flask мають таку специфіку, що фреймворк використовує `nginx` для локального хостингу, але його не можна використовувати для розміщення проекту в реліз, про що виводиться попередження в консоль, при запуску проекту локально.



Тому для обробки роутингу потрібно обрати ноду. Я обрав gunicorn через те, що для цього проекту не може бути серйозних навантажень. Файл конфігурації Docker також має вказаний параметр типу середовища

```
app.yaml  main.py  .gcloudignore  Dockerfile  requirements.txt
20 FROM python:3.11-slim
21
22 # Allow statements and log messages to immediately appear in the logs
23 ENV PYTHONUNBUFFERED True
24
25 # Copy local code to the container image.
26 ENV APP_HOME /app
27 WORKDIR $APP_HOME
28 COPY . ./
29
30 # Install production dependencies.
31 RUN pip install --no-cache-dir -r requirements.txt
32
33 # Run the web service on container startup. Here we use the gunicorn
34 # webserver, with one worker process and 8 threads.
35 # For environments with multiple CPU cores, increase the number of workers
36 # to be equal to the cores available.
37 # Timeout is set to 0 to disable the timeouts of the workers to allow Cloud Run to handle instance scaling.
38 CMD exec gunicorn --bind :$PORT --workers 1 --threads 8 --timeout 0 main:app
39
40 # [END run_helloworld_dockerfile]
```

Також, для того, щоб на віддаленій машині були встановленні відповідні бібліотеки та їх розширення, потрібно написати файл, за допомогою якого Docker автоматично їх встановить при запуску проекту, для цього є відповідний стандарт, згідно якого використовується txt файл з назвою requirements. Вмістом цього файлу є назви бібліотек, фреймворків та їх версії

```
app.yaml  main.py  .gcloudignore  Dockerfile  requirements.txt
1 Flask==3.0.0
2 gunicorn==20.1.0
3 Werkzeug==3.0.1
```

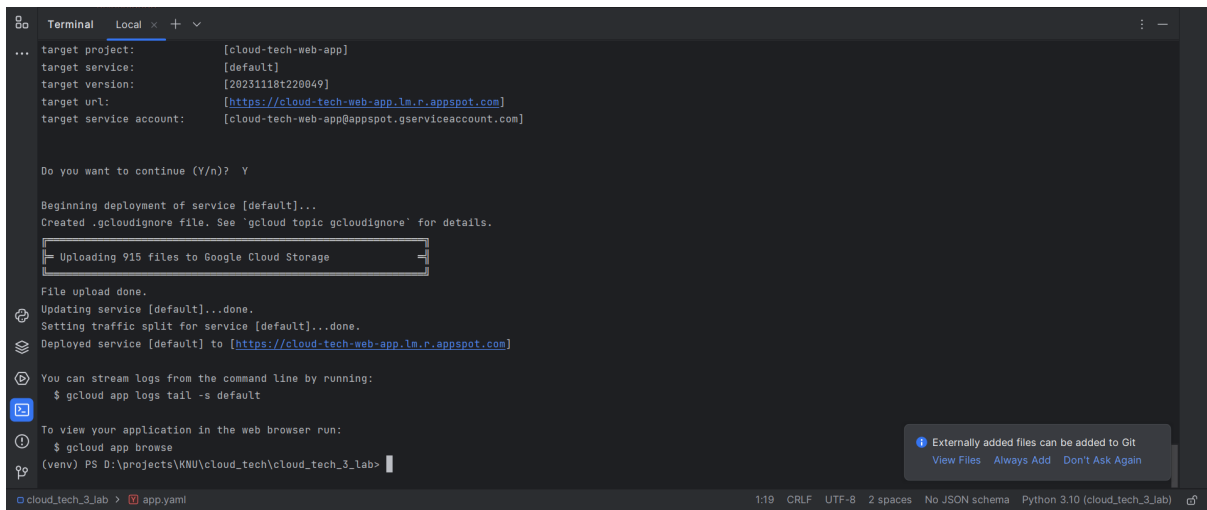
Після написання всієї архітектури можна приступити до деплою додатку на хмару. Для цього, в директорії проекту потрібно виконати команду gcloud init, за допомогою якої директорію буде ініціалізовано як робочу. Для ініціалізації також знадобиться встановити project-id того проекту, де створено App Engine. Після чого запускаємо команду gcloud app deploy

```
Terminal  Local  x  +  v
... For detailed information on this command and its flags, run:
gcloud config get --help
(venv) PS D:\projects\KNU\cloud_tech\cloud_tech_3_lab> gcloud config set project cloud-tech-web-app
Updated property [core/project].
(venv) PS D:\projects\KNU\cloud_tech\cloud_tech_3_lab> gcloud app deploy
Services to deploy:

descriptor:      [D:\projects\KNU\cloud_tech\cloud_tech_3_lab\app.yaml]
source:          [D:\projects\KNU\cloud_tech\cloud_tech_3_lab]
target project:  [cloud-tech-web-app]
target service:  [default]
target version:  [20231118t220949]
target url:      [https://cloud-tech-web-app-lm.r.appspot.com]
target service account: [cloud-tech-web-app@appspot.gserviceaccount.com]

Do you want to continue (Y/n)?
```

На цьому етапі виконується верифікація параметрів проекту. Для продовження потрібно натиснути у, після чого розпочнеться завантаження проектів та налаштування залежностей.

A terminal window showing the deployment of a service to Google Cloud. The output includes target project, service, version, url, and account information. It prompts for confirmation to continue, then shows the upload of 915 files to Google Cloud Storage. Finally, it displays the deployment status and provides commands to stream logs and view the application in a web browser.

```
... target project: [cloud-tech-web-app]
target service: [default]
target version: [20231118t220049]
target url: [https://cloud-tech-web-app.lm.r.appspot.com]
target service account: [cloud-tech-web-app@appspot.gserviceaccount.com]

Do you want to continue (Y/n)? Y

Beginning deployment of service [default]...
Created .gcloudignore file. See 'gcloud topic gcloudignore' for details.

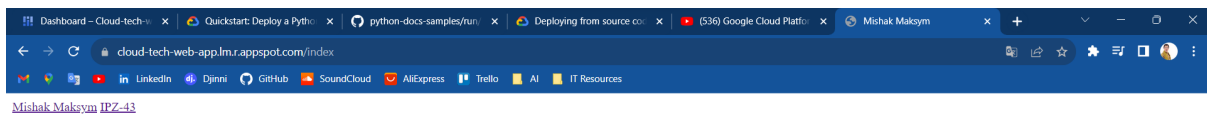
[Progress bar] Uploading 915 files to Google Cloud Storage

File upload done.
Updating service [default]...done.
Setting traffic split for service [default]...done.
Deployed service [default] to [https://cloud-tech-web-app.lm.r.appspot.com]

You can stream logs from the command line by running:
$ gcloud app logs tail -s default

To view your application in the web browser run:
$ gcloud app browse
(venv) PS D:\projects\KMU\cloud_tech\cloud_tech_3_lab>
```

По завершенню завантаження буде надіслане посилання, за допомогою якого можна перевірити роботу сервісу.



Проект розгорнуто успішно.