

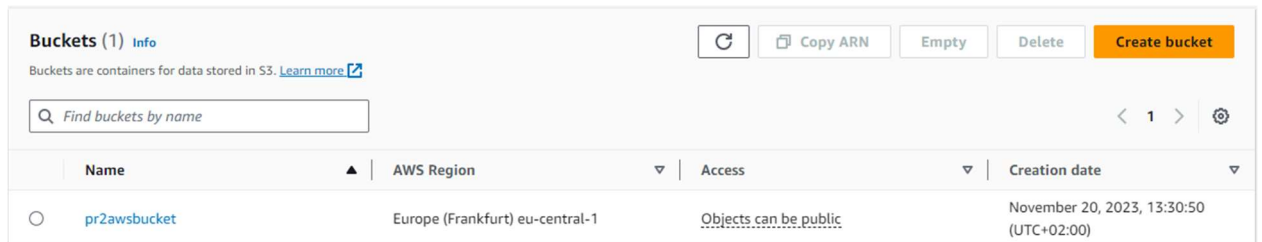
Звіт
З практичної роботи №2
Студента групи МІТ-31
Добровольського Арсенія Михайловича

Тема роботи: Розгортання додатку на хмарній платформі AWS

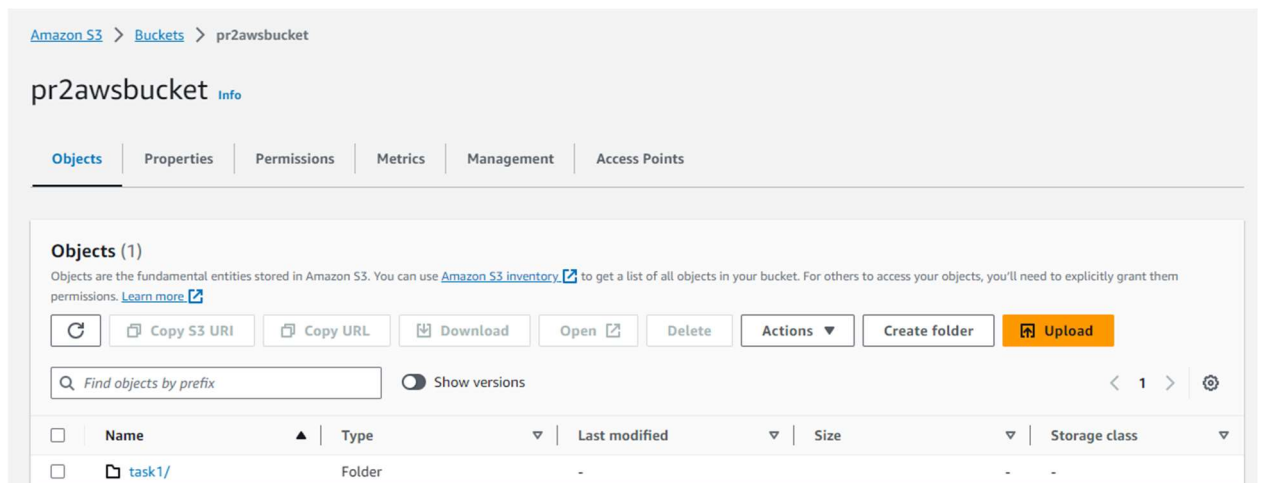
Мета роботи: ознайомитися із базовими сервісами AWS; навчитися розгортати додаток на хмарній платформі AWS.

Завдання 1 (AWS S3)

Створення і перегляд кошика:



Створимо у цьому кошику папку task1, в яку завантажимо декілька файлів для статичного вебсайту (файли взято з напрацювань минулорічної дисципліни):



Amazon S3 > Buckets > pr2awsbucket > task1/

task1/ Copy S3 URI

Objects Properties

Objects (4)

Objects are the fundamental entities stored in Amazon S3. You can use [Amazon S3 inventory](#) to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. [Learn more](#)

Refresh Copy S3 URI Copy URL Download Open Delete Actions Create folder Upload

Show versions < 1 >

<input type="checkbox"/>	Name	Type	Last modified	Size	Storage class
<input type="checkbox"/>	css/	Folder	-	-	-
<input type="checkbox"/>	img/	Folder	-	-	-
<input type="checkbox"/>	index.html	html	November 20, 2023, 14:14:55 (UTC+02:00)	7.9 KB	Glacier Instant Retrieval
<input type="checkbox"/>	WD_online_course.pdf	pdf	November 20, 2023, 14:14:57 (UTC+02:00)	328.0 KB	Glacier Instant Retrieval

Переглянемо властивості одного з файлів (наприклад, index.html):

Amazon S3 > Buckets > pr2awsbucket > task1/ > index.html

index.html Info Copy S3 URI Download Open Object actions

Properties Permissions Versions

Object overview

Owner c9df7e1805584f8290901172457dbbeb89571a2423882305bc55852eda91591e	S3 URI s3://pr2awsbucket/task1/index.html
AWS Region Europe (Frankfurt) eu-central-1	Amazon Resource Name (ARN) arn:aws:s3::pr2awsbucket/task1/index.html
Last modified November 20, 2023, 14:14:55 (UTC+02:00)	Entity tag (Etag) 869971f0223b94feca944ed673b22b97
Size 7.9 KB	Object URL https://pr2awsbucket.s3.eu-central-1.amazonaws.com/task1/index.html
Type html	
Key task1/index.html	

При спробі відкрити цей файл у браузері отримуємо повідомлення:

← → ↻ pr2awsbucket.s3.eu-central-1.amazonaws.com/task1/index.html

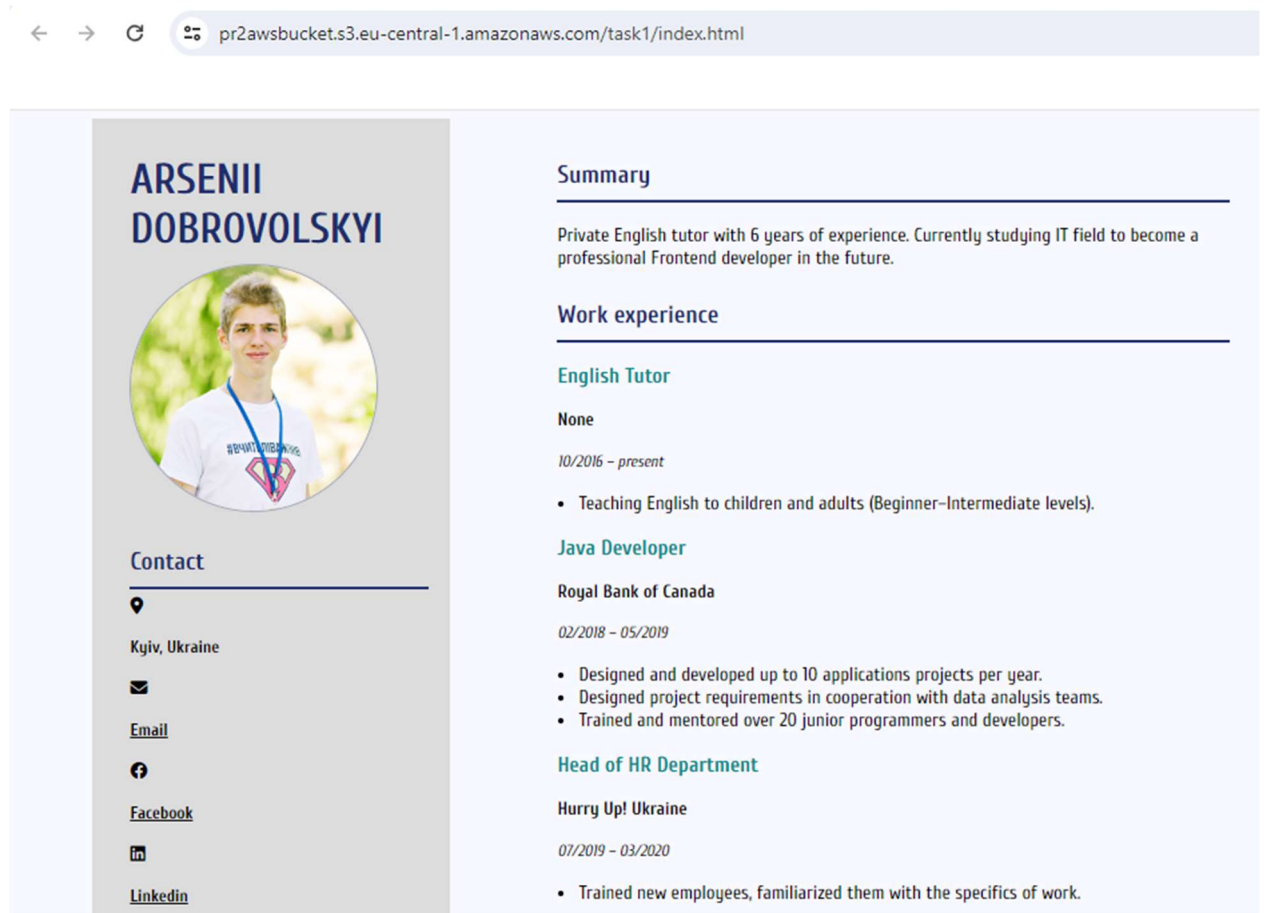
This XML file does not appear to have any style information associated with it. The document tree is shown below.

```
<?xml version="1.0" encoding="UTF-8" ?>
<Error>
  <Code>AccessDenied</Code>
  <Message>Access Denied</Message>
  <RequestId>13737QNR732WR9P2</RequestId>
  <HostId>Q6G5yT5yUUeZQ/2Pr7WYeJIA9YPvYXGtXWLM5V4EkAmbm/shgKNyPaHVdyQ+vZcytEhy6ZNb1cusZEqYwNHVDQ==</HostId>
</Error>
```

Для виправлення цієї помилки потрібно виконати 2 кроки:

1. У вікні кошика на вкладці **permissions** у секції **object ownership** дозволити використання ACL (ACLs enabled)
2. У вікні файлу в правому верхньому куті обрати **Object actions** -> **Make public using ACL** (якщо файлів декілька, це можна також зробити у вікні кошика, вибравши всі файли, потім **Actions** -> **Make public using ACL**)

Виконавши ці кроки, вебсторінка справно завантажується за [посиланням](#):



Дослідимо версіювання та міжрегіональну реплікацію.

Версіювання дозволяє зберігати різні версії одного і того ж об'єкта в тому самому бакеті. Якщо версіювання увімкнено для бакету, кожен раз, коли відбуваються зміни в об'єкті, система зберігає нову версію цього об'єкта. Це неабияк стає в пригоді, коли виникає потреба відновити попередню версію файлу або відновитися після помилок чи неправильних змін.

Внесемо невеликі зміни у файл `index.html` і повторно вивантажимо його у наш бакет. Далі ввімкнемо повзунок **Show versions** для того, щоб побачити версії нашого файлу (підкреслені червоним):

Objects (5)
Objects are the fundamental entities stored in Amazon S3. You can use [Amazon S3 inventory](#) to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. [Learn more](#)

[Refresh](#) [Copy S3 URI](#) [Copy URL](#) [Download](#) [Open](#) [Delete](#) [Actions](#) [Create folder](#) [Upload](#)

☒ Show versions < 1 > [Settings](#)

<input type="checkbox"/>	Name	Type	Version ID	Last modified	Size	Storage class
<input type="checkbox"/>	img/	Folder	-	-	-	-
<input type="checkbox"/>	index.html	html	C9PyjeY9spelvf_vC_R_FrtuFZxDKvwJ	November 21, 2023, 14:48:34 (UTC+02:00)	7.9 KB	Glacier Instant Retrieval
<input type="checkbox"/>	index.html	html	QE_vd_Nwe5Hym.Ri45Ft8AduiYdW0XCR	November 20, 2023, 14:14:55 (UTC+02:00)	7.9 KB	Glacier Instant Retrieval

Кожна версія об'єкта отримує унікальний ідентифікатор, який можна використовувати для відновлення конкретної версії за необхідності:

[Amazon S3](#) > [Buckets](#) > [pr2awsbucket](#) > [task1/](#) > [index.html](#)

index.html [Info](#) [Copy S3 URI](#) [Download](#) [Open](#) [Object actions](#)

Version ID:
[QE_vd_Nwe5Hym.Ri45Ft8AduiYdW0XCR](#)

[Properties](#) [Permissions](#) [Versions](#)

Versions (2) [Download](#) [Open](#) [Delete](#) [Actions](#) < 1 >

<input type="checkbox"/>	Version ID	Type	Last modified	Size	Storage class
<input type="checkbox"/>	C9PyjeY9spelvf_vC_R_FrtuFZxDKvwJ (Current version)	html	November 21, 2023, 14:48:34 (UTC+02:00)	7.9 KB	Glacier Instant Retrieval
<input type="checkbox"/>	QE_vd_Nwe5Hym.Ri45Ft8AduiYdW0XCR	html	November 20, 2023, 14:14:55 (UTC+02:00)	7.9 KB	Glacier Instant Retrieval

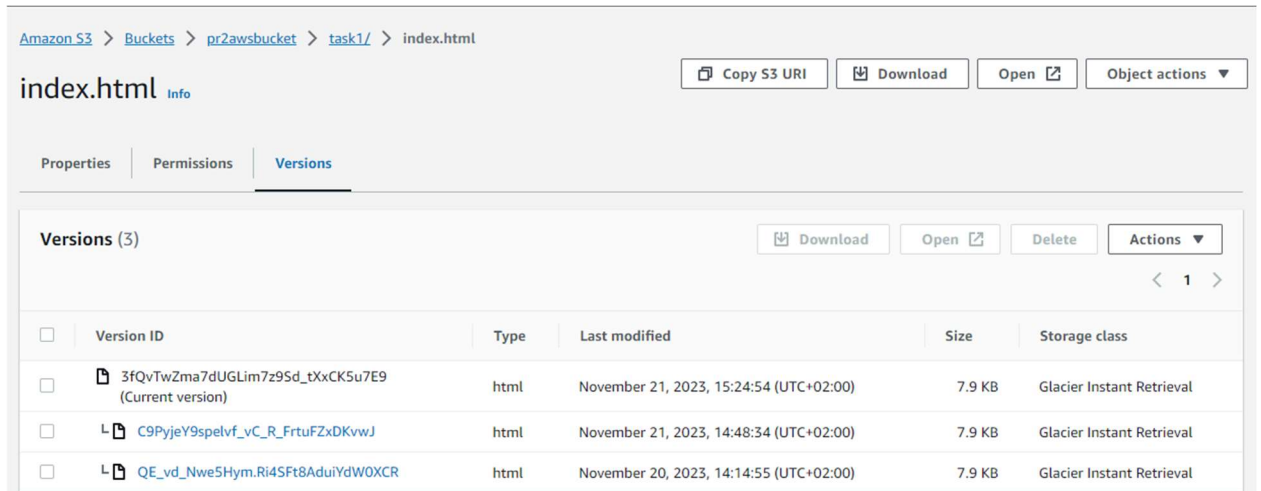
Для того, щоб «відкотитися» до попередньої версії файлу, AWS пропонує два підходи:

- Скопіювати попередню версію об'єкта в той самий бакет
Скопійований об'єкт стає поточною версією цього об'єкта, при цьому всі версії об'єкта зберігаються.
- Назавжди видалити поточну версію об'єкта

При видаленні поточної версії об'єкта, попередня версія цього об'єкта фактично стає поточною.

Розглянемо обидва способи.

Для реалізації першого завантажимо попередню версію файлу і знову вивантажимо її у бакет:



Amazon S3 > Buckets > pr2awsbucket > task1/ > index.html

index.html Info

Copy S3 URI Download Open Object actions

Properties Permissions Versions

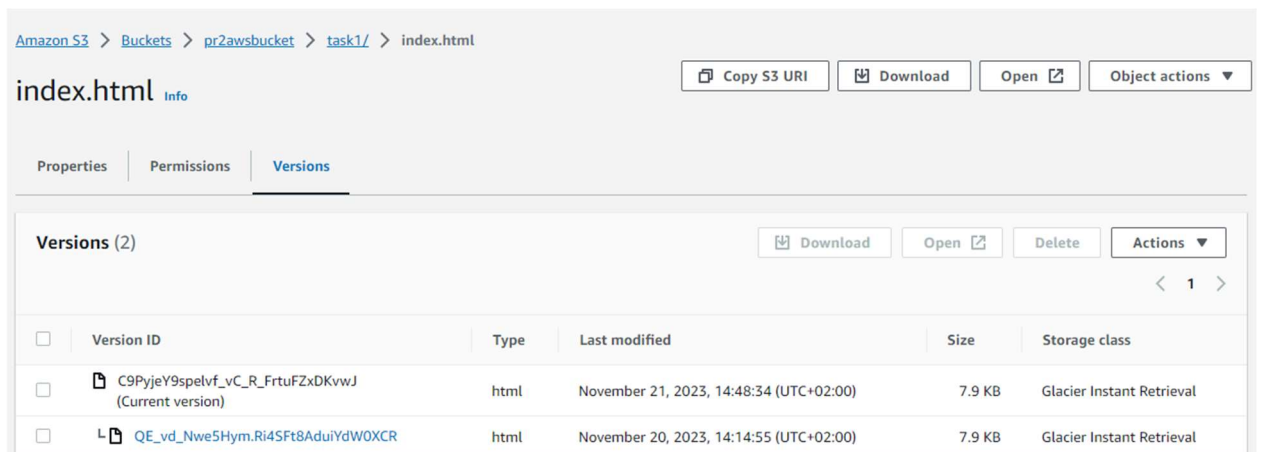
Versions (3)

Download Open Delete Actions

<input type="checkbox"/>	Version ID	Type	Last modified	Size	Storage class
<input type="checkbox"/>	3fQvTwZma7dUGLim7z9Sd_tXxCK5u7E9 (Current version)	html	November 21, 2023, 15:24:54 (UTC+02:00)	7.9 KB	Glacier Instant Retrieval
<input type="checkbox"/>	C9PyjeY9spelvf_vC_R_FrtuFZxDKvwJ	html	November 21, 2023, 14:48:34 (UTC+02:00)	7.9 KB	Glacier Instant Retrieval
<input type="checkbox"/>	QE_vd_Nwe5Hym.Ri45Ft8AduiYdW0XCR	html	November 20, 2023, 14:14:55 (UTC+02:00)	7.9 KB	Glacier Instant Retrieval

AWS присвоїв новий ідентифікатор вивантаженій версії, і вона стала поточною.

Видалити версію можливо, вибравши checkbox навпроти потрібної версії і натиснувши кнопку **Delete** праворуч вгорі.



Amazon S3 > Buckets > pr2awsbucket > task1/ > index.html

index.html Info

Copy S3 URI Download Open Object actions

Properties Permissions Versions

Versions (2)

Download Open Delete Actions

<input type="checkbox"/>	Version ID	Type	Last modified	Size	Storage class
<input type="checkbox"/>	C9PyjeY9spelvf_vC_R_FrtuFZxDKvwJ (Current version)	html	November 21, 2023, 14:48:34 (UTC+02:00)	7.9 KB	Glacier Instant Retrieval
<input type="checkbox"/>	QE_vd_Nwe5Hym.Ri45Ft8AduiYdW0XCR	html	November 20, 2023, 14:14:55 (UTC+02:00)	7.9 KB	Glacier Instant Retrieval

Видаливши останню версію, попередня знову стала поточною.

Міжрегіональна реплікація дозволяє автоматично копіювати об'єкти з одного бакету в одному регіоні в інший бакет в іншому регіоні. Ця функція створена для забезпечення резервного копіювання у випадку втрати даних або регіону, відновлення даних та забезпечення вищого рівня доступності та відмовостійкості.

Створимо новий бакет у регіоні, відмінному від регіону першого бакета:

[Amazon S3](#) > [Buckets](#) > Create bucket

Create bucket [Info](#)

Buckets are containers for data stored in S3. [Learn more](#)

General configuration

Bucket name

pr2crrbucket

Bucket name must be unique within the global namespace and follow the bucket naming rules. [See rules for bucket naming.](#)

AWS Region

Canada (Central) ca-central-1

Copy settings from existing bucket - *optional*

Only the bucket settings in the following configuration are copied.

Choose bucket

✓

 pr2awsbucket

Creation date: November 20, 2023, 13:30:50 (UTC+02:00)

Для того, щоб файли могли копіюватися з першого бакета в другий, потрібно у першому створити *правило реплікації*. Для цього переходимо у перший бакет, обираємо вкладку **Management**, в ній розділ **Replication rules** -> **Create replication rule**:

[Amazon S3](#) > [Buckets](#) > [pr2awsbucket](#) > [Replication rules](#) > Create replication rule

Create replication rule Info

Replication rule configuration

Replication rule name

Up to 255 characters. In order to be able to use CloudWatch metrics to monitor the progress of your replication rule, the replication rule name must only contain English characters.

Status

Choose whether the rule will be enabled or disabled when created.

☒ Enabled

☐ Disabled

Priority

The priority value resolves conflicts that occur when an object is eligible for replication under multiple rules to the same destination. The rule is added to the configuration at the highest priority and the priority can be changed on the replication rules table.

0

При створенні правила з'явиться вікно із запитанням, чи потрібно скопіювати вже наявні об'єкти у першому бакеті в другий. Обираємо варіант «Так»:

Replicate existing objects? ×

You can enable a one-time Batch Operations job from this replication configuration to replicate objects that already exist in the bucket and to synchronize the source and destination buckets. [Learn more](#) [or see pricing](#)

Existing objects

☐ No, do not replicate existing objects.

☒ Yes, replicate existing objects.

Cancel Submit

Копіювання файлів займає деякий час, це нормально:

Amazon S3 > Batch Operations

Batch Operations

Info

A job is used to execute batch operations on a list of S3 objects. The list of S3 objects is contained in a manifest object, which can be an S3 inventory report or a list of objects that you generate. After the total number of objects listed in the manifest has been confirmed, the job status will update to *Awaiting your confirmation to run*, and you must **Run job** within 30 days. Job events are published to [CloudWatch Events](#). Jobs are deleted 90 days after they finish or fail. [Learn more](#)

Jobs (1)

Search by job ID or description

All status types

< 1 >

Europe (Frankfurt) eu-central-1

	Job ID	Status	Description	Operation	Date created	Total objects	% Complete	Total failed (rate)	Priority
	c1155ae2-fc2a-4113-aca4-327d16536441	Completing	2023-11-21 - Replicate	Replicate	November 21, 2023, 17:21:36 (UTC+02:00)	7	100%	0 (0%)	10

Amazon S3 > Buckets > pr2crrbucket

pr2crrbucket

Info

Publicly accessible

Objects

Properties

Permissions

Metrics

Management

Access Points

Objects (1)

Objects are the fundamental entities stored in Amazon S3. You can use [Amazon S3 inventory](#) to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. [Learn more](#)

Copy S3 URI

Copy URL

Download

Open

Delete

Actions

Create folder

Upload

Find objects by prefix

Show versions

< 1 >

	Name	Type	Last modified	Size	Storage class
	task1/	Folder	-	-	-

Додамо новий файл у перший бакет, з якого відбувається реплікація об'єктів (підкреслений червоним):

Amazon S3 > Buckets > pr2awsbucket

pr2awsbucket

Info

Publicly accessible

Objects

Properties

Permissions

Metrics

Management

Access Points

Objects (2)

Objects are the fundamental entities stored in Amazon S3. You can use [Amazon S3 inventory](#) to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. [Learn more](#)

Copy S3 URI

Copy URL

Download

Open

Delete

Actions

Create folder

Upload

Find objects by prefix

Show versions

< 1 >

	Name	Type	Last modified	Size	Storage class
	task1/	Folder	-	-	-
	<u>Smilyvi_vidnovliuvaty.jpg</u>	jpg	November 21, 2023, 17:35:12 (UTC+02:00)	184.0 KB	Standard

Переходимо у новостворений бакет і бачимо, що там файл також з'явився:

pr2crrbucket [Info](#)

Publicly accessible

[Objects](#) | [Properties](#) | [Permissions](#) | [Metrics](#) | [Management](#) | [Access Points](#)

Objects (2)

Objects are the fundamental entities stored in Amazon S3. You can use [Amazon S3 inventory](#) to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. [Learn more](#)

[Refresh](#) [Copy S3 URI](#) [Copy URL](#) [Download](#) [Open](#) [Delete](#) [Actions](#) [Create folder](#) [Upload](#)

☐ Show versions < 1 > ⚙

<input type="checkbox"/>	Name	Type	Last modified	Size	Storage class
<input type="checkbox"/>	Smilyvi_vidnovliuvaty.jpg	jpg	November 21, 2023, 17:35:12 (UTC+02:00)	184.0 KB	Standard
<input type="checkbox"/>	task1/	Folder	-	-	-

Завдання 2 (EC2)

Для початку створимо EC2 Instance:

[EC2](#) > [Instances](#) > Launch an instance

Launch an instance [Info](#)

Amazon EC2 allows you to create virtual machines, or instances, that run on the AWS Cloud. Quickly get started by following the simple steps below.

Name and tags [Info](#)

Name

[Add additional tags](#)

Під'єднаємося до створеного EC2 Instance через SSH. Для цього відкриваємо командний рядок і вводимо там таку команду:

```

C:\ ec2-user@ip-172-31-28-55:~
Microsoft Windows [Version 10.0.19045.3693]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Fijitsu E546>ssh -i "ars13KP.pem" ec2-user@ec2-3-121-228-131.eu-central-1.compute.amazonaws.com

#_
#####      Amazon Linux 2023
nnn\#####\
nnn\###|
nnn\#/      https://aws.amazon.com/linux/amazon-linux-2023
nnnVn'-'->

#_
#####      Amazon Linux 2023
nnn\#####\
nnn\###|
nnn\#/      https://aws.amazon.com/linux/amazon-linux-2023
nnnVn'-'->

nnnn
nnn_-.-
    _/_m/'-/->

Last login: Wed Nov 22 12:53:20 2023 from 88.155.152.112

```

Далі встановлюємо віртуальне середовище для Python:

```

Select ec2-user@ip-172-31-28-55:~
[ec2-user@ip-172-31-28-55 ~]$ sudo yum install python3-virtualenv
Last metadata expiration check: 1:18:37 ago on Wed Nov 22 11:38:22 2023.
Dependencies resolved.
=====
Package                        Architecture      Version                               Repository      Size
=====
Installing:
python3-virtualenv            noarch            20.4.0-3.amzn2023.0.3               amazonlinux     233 k
Installing dependencies:
python3-appdirs               noarch            1.4.4-2.amzn2023.0.2               amazonlinux     23 k
python3-distlib               noarch            0.3.1-4.amzn2023.0.2               amazonlinux     188 k
python3-filelock              noarch            3.0.12-9.amzn2023.0.2              amazonlinux     22 k
python3-wheel-wheel           noarch            1:0.37.1-1.amzn2023.0.3            amazonlinux     43 k
=====
Transaction Summary
-----
Install 5 Packages

Total download size: 508 k
Installed size: 2.0 M
Is this ok [y/N]: y
Downloading Packages:
(1/5): python3-wheel-wheel-0.37.1-1.amzn2023.0.3.noarch.rpm           521 kB/s | 43 kB    00:00
(2/5): python3-filelock-3.0.12-9.amzn2023.0.2.noarch.rpm             203 kB/s | 22 kB    00:00
(3/5): python3-appdirs-1.4.4-2.amzn2023.0.2.noarch.rpm               523 kB/s | 23 kB    00:00
(4/5): python3-distlib-0.3.1-4.amzn2023.0.2.noarch.rpm               1.4 MB/s | 188 kB   00:00
(5/5): python3-virtualenv-20.4.0-3.amzn2023.0.3.noarch.rpm           2.5 MB/s | 233 kB   00:00
-----
Total                                1.9 MB/s | 508 kB   00:00

```

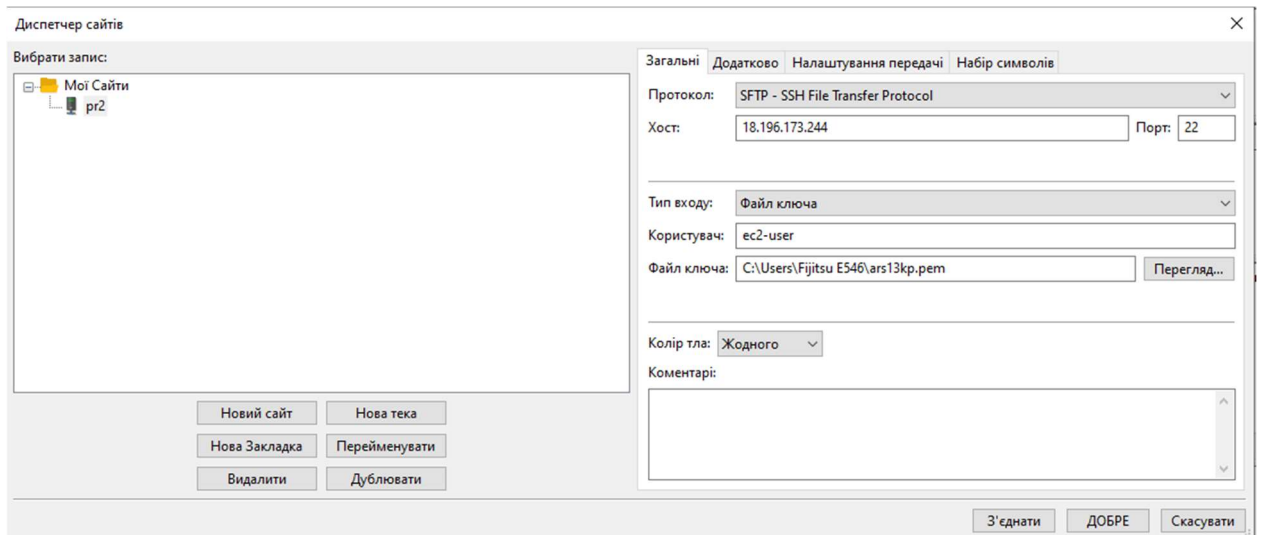
Створюємо папку **task2**, в якій розгортаємо віртуальне середовище **flaskApp** і активуємо його:

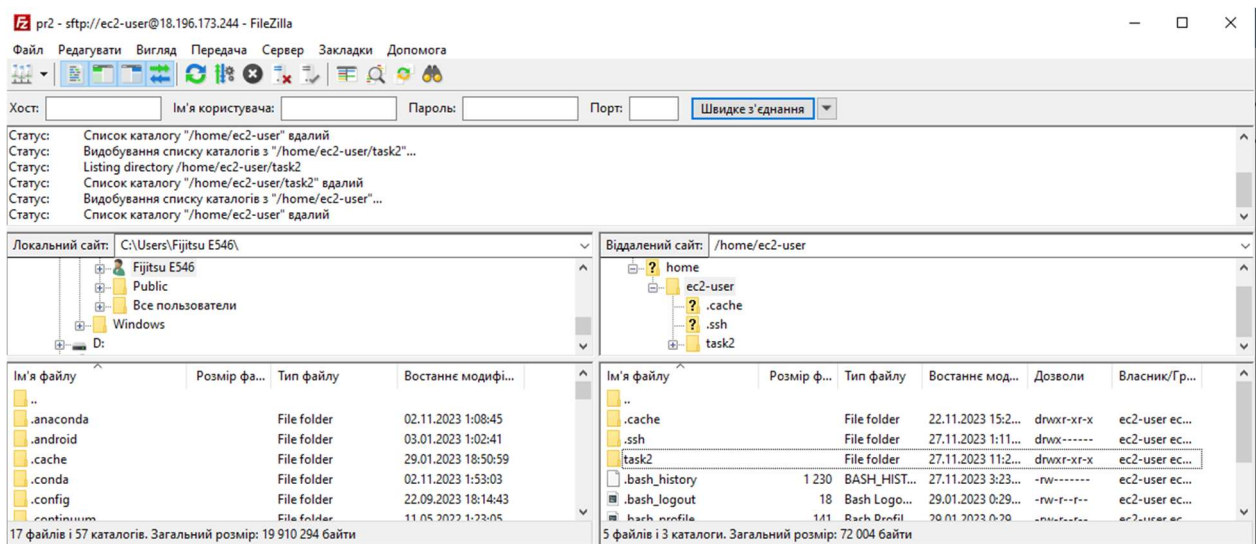
```
[ec2-user@ip-172-31-28-55 ~]$ mkdir task2
[ec2-user@ip-172-31-28-55 ~]$ cd task2
[ec2-user@ip-172-31-28-55 task2]$ python3 -m venv flaskApp
[ec2-user@ip-172-31-28-55 task2]$ source flaskApp/bin/activate
(flaskApp) [ec2-user@ip-172-31-28-55 task2]$
```

Встановлюємо Flask:

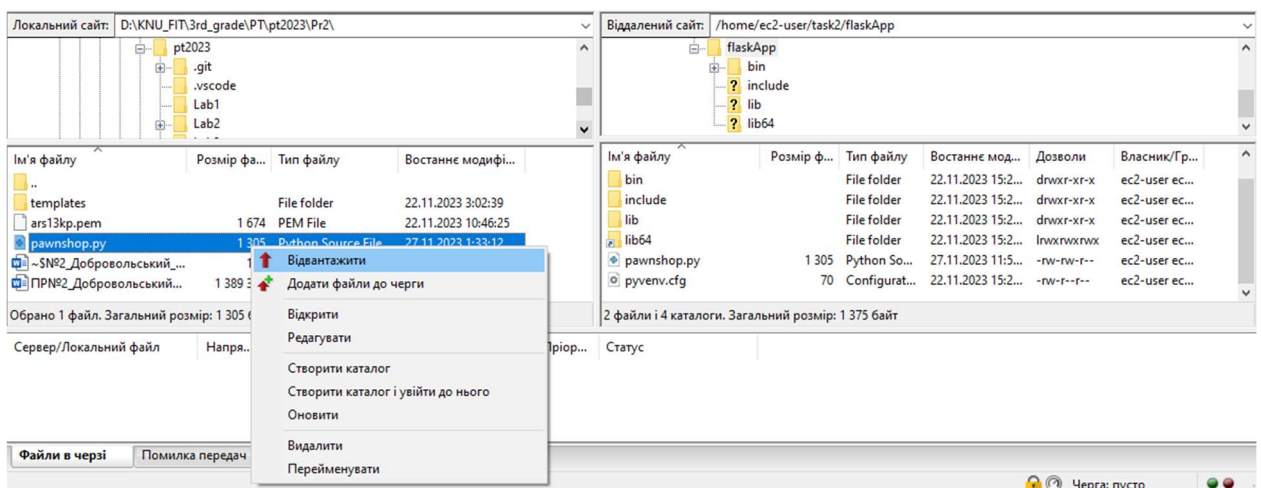
```
(flaskApp) [ec2-user@ip-172-31-28-55 task2]$ pip install Flask
Collecting Flask
  Downloading flask-3.0.0-py3-none-any.whl (99 kB)
    |#####| 99 kB 4.1 MB/s
Collecting Werkzeug>=3.0.0
  Downloading werkzeug-3.0.1-py3-none-any.whl (226 kB)
    |#####| 226 kB 30.7 MB/s
Collecting click>=8.1.3
  Downloading click-8.1.7-py3-none-any.whl (97 kB)
    |#####| 97 kB 15.8 MB/s
Collecting blinker>=1.6.2
  Downloading blinker-1.7.0-py3-none-any.whl (13 kB)
Collecting Jinja2>=3.1.2
  Downloading Jinja2-3.1.2-py3-none-any.whl (133 kB)
    |#####| 133 kB 53.5 MB/s
Collecting itsdangerous>=2.1.2
  Downloading itsdangerous-2.1.2-py3-none-any.whl (15 kB)
Collecting importlib-metadata>=3.6.0
  Downloading importlib_metadata-6.8.0-py3-none-any.whl (22 kB)
Collecting zipp>=0.5
  Downloading zipp-3.17.0-py3-none-any.whl (7.4 kB)
Collecting MarkupSafe>=2.0
  Downloading MarkupSafe-2.1.3-cp39-cp39-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (25 kB)
Installing collected packages: zipp, MarkupSafe, Werkzeug, Jinja2, itsdangerous, importlib-metadata, click, blinker, Flask
Successfully installed Flask-3.0.0 Jinja2-3.1.2 MarkupSafe-2.1.3 Werkzeug-3.0.1 blinker-1.7.0 click-8.1.7 importlib-metadata-6.8.0 itsdangerous-2.1.2 zipp-3.17.0
```

Тепер необхідно скопіювати файл з Flask-застосунком до EC2 Instance. Для цього скористаємося програмою FileZilla. У верхньому лівому куті обираємо **Файл** -> **Диспетчер сайтів** і створюємо новий сайт, вказавши необхідні параметри (наводяться на скриншоті):





Вдало під'єднавшись до EC2 Instance (праворуч), скопіюємо наш файл до його сховища. Обираємо потрібний файл і натискаємо кнопку «Відвантажити»:



Встановлюємо Gunicorn:

```

^C(flaskApp) [ec2-user@ip-172-31-28-55 task2]$ pip install gunicorn
Collecting gunicorn
  Downloading gunicorn-21.2.0-py3-none-any.whl.metadata (4.1 kB)
Collecting packaging (from gunicorn)
  Downloading packaging-23.2-py3-none-any.whl.metadata (3.2 kB)
Downloading gunicorn-21.2.0-py3-none-any.whl (80 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 80.2/80.2 kB 5.1 MB/s eta 0:00:00
Downloading packaging-23.2-py3-none-any.whl (53 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 53.0/53.0 kB 9.2 MB/s eta 0:00:00
Installing collected packages: packaging, gunicorn
Successfully installed gunicorn-21.2.0 packaging-23.2

```

Gunicorn – це популярний HTTP-сервер WSGI для програм Python. Він слугуватиме мостом між додатком Flask та Інтернетом.

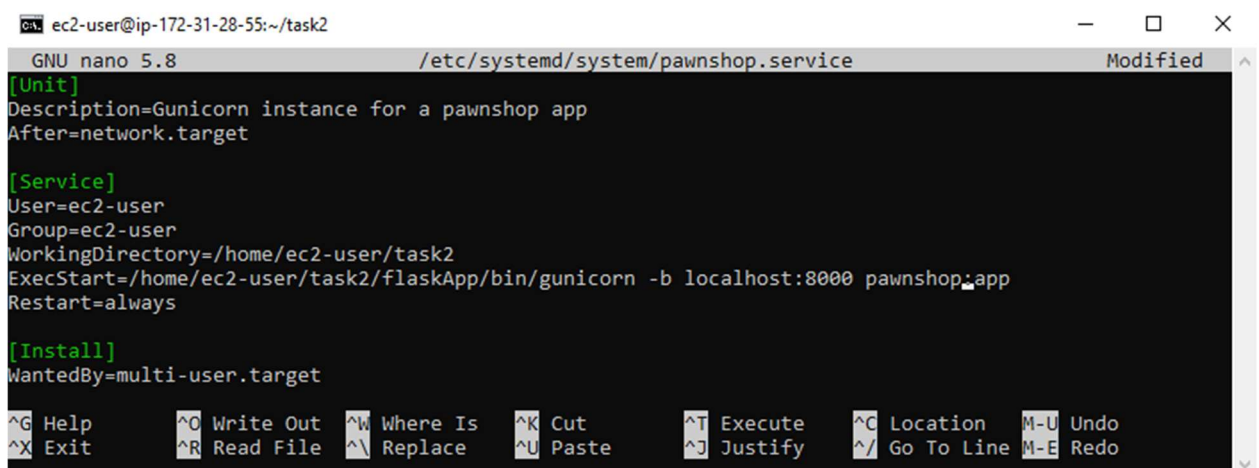
Перевірка роботи програми локально з Gunicorn:

```
(flaskApp) [ec2-user@ip-172-31-28-55 task2]$ gunicorn -b 0.0.0.0:8000 pawnshop:app
[2023-11-27 10:10:12 +0000] [22438] [INFO] Starting gunicorn 21.2.0
[2023-11-27 10:10:12 +0000] [22438] [INFO] Listening at: http://0.0.0.0:8000 (22438)
[2023-11-27 10:10:12 +0000] [22438] [INFO] Using worker: sync
[2023-11-27 10:10:12 +0000] [22439] [INFO] Booting worker with pid: 22439
```

Для синхронної роботи Gunicorn та EC2 Instance скористаємося менеджером ініціалізації та управління процесами Systemd. У папці **/etc/systemd/system** необхідно створити файл з розширенням **.service**, в якому вказуємо, як має поводитися gunicorn у разі перезавантаження системи. Цей файл міститиме 3 частини:

- Unit – для опису проєкту та основних залежностей служби
- Service – для визначення користувача/групи, від імені якого/якої відбуватиметься запуск служби. Сюди також входять команда для запуску, змінні середовища, права доступу тощо
- Install – за допомогою спеціальних інструкцій повідомляє systemd, у який момент під час процесу завантаження служба повинна запускатися

Текст файлу такий:



```
ec2-user@ip-172-31-28-55:~/task2
GNU nano 5.8 /etc/systemd/system/pawnshop.service Modified
[Unit]
Description=Gunicorn instance for a pawnshop app
After=network.target

[Service]
User=ec2-user
Group=ec2-user
WorkingDirectory=/home/ec2-user/task2
ExecStart=/home/ec2-user/task2/flaskApp/bin/gunicorn -b localhost:8000 pawnshop:app
Restart=always

[Install]
WantedBy=multi-user.target

^G Help      ^O Write Out ^W Where Is  ^K Cut       ^T Execute  ^C Location  M-U Undo
^X Exit      ^R Read File ^\ Replace   ^U Paste     ^J Justify  ^_ Go To Line M-E Redo
```

Запускаємо службу та перевіряємо її статус:

```
(flaskApp) [ec2-user@ip-172-31-28-55 task2]$ sudo systemctl daemon-reload
o systemctl restart pawnshop
sudo systemctl status pawnshop
(flaskApp) [ec2-user@ip-172-31-28-55 task2]$ sudo systemctl restart pawnshop
(flaskApp) [ec2-user@ip-172-31-28-55 task2]$ sudo systemctl status pawnshop
• pawnshop.service - Gunicorn instance for a pawnshop app
  Loaded: loaded (/etc/systemd/system/pawnshop.service; enabled; preset: disabled)
  Active: active (running) since Mon 2023-11-27 13:24:02 UTC; 125ms ago
    Main PID: 30348 (gunicorn)
      Tasks: 1 (limit: 1114)
     Memory: 7.8M
        CPU: 58ms
    CGroup: /system.slice/pawnshop.service
            └─30348 /home/ec2-user/task2/flaskApp/bin/python3 /home/ec2-user/task2/flaskApp/bin/gunicorn -b localhost:
8000 pawnshop:app

Nov 27 13:24:02 ip-172-31-28-55.eu-central-1.compute.internal systemd[1]: Started pawnshop.service - Gunicorn instance
for a pawnshop app.
```

Перевірка працездатності застосунку:

```
ec2-user@ip-172-31-28-55:~/task2

(flaskApp) [ec2-user@ip-172-31-28-55 task2]$ curl localhost:8000
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>Pawnshop</title>
  </head>
  <body>
    <h1>Available Goods</h1>
    <ul>

      <li>Radio: 200</li>

      <li>Headphones: 300</li>

      <li>Watch: 499.55555</li>

      <li>Ring: 749.47755</li>

    </ul>
    <h2>Add Good</h2>
    <form action="/add_good" method="post">
      <label for="name">Name:</label>
      <input type="text" name="name" id="name" required />
      <label for="price">Price:</label>
      <input type="number" name="price" id="price" required />
      <input type="submit" value="Add" />
    </form>
    <h2>Remove Good</h2>
    <form action="/remove_good" method="post">
      <label for="name">Name:</label>
      <input type="text" id="name" name="name" required />
      <button type="submit">Remove</button>
    </form>
```

Зокрема, необхідно використати Nginx Webserver, який прийматиме запити від користувача та направлятиме їх до Gunicorn.


Спершу встановлюємо Nginx, який діятиме як зворотний проксі для Flask-застосунку:

```
ec2-user@ip-172-31-28-55: ~/task2
(flaskApp) [ec2-user@ip-172-31-28-55 task2]$ sudo yum install nginx
Last metadata expiration check: 4:23:58 ago on Mon Nov 27 09:09:13 2023.
Dependencies resolved.
=====
Package                                Architecture      Version           Repository        Size
=====
Installing:
nginx                                  x86_64            1:1.24.0-1.amzn2023.0.2  amazonlinux      32 k
Installing dependencies:
generic-logos-httpd                  noarch            18.0.0-12.amzn2023.0.3  amazonlinux      19 k
gperftools-libs                      x86_64            2.9.1-1.amzn2023.0.3    amazonlinux      308 k
libunwind                            x86_64            1.4.0-5.amzn2023.0.2    amazonlinux      66 k
nginx-core                           x86_64            1:1.24.0-1.amzn2023.0.2  amazonlinux      586 k
nginx-filesystem                     noarch            1:1.24.0-1.amzn2023.0.2  amazonlinux      9.1 k
nginx-mimetypes                      noarch            2.1.49-3.amzn2023.0.3    amazonlinux      21 k
=====
Transaction Summary
=====
Install 7 Packages

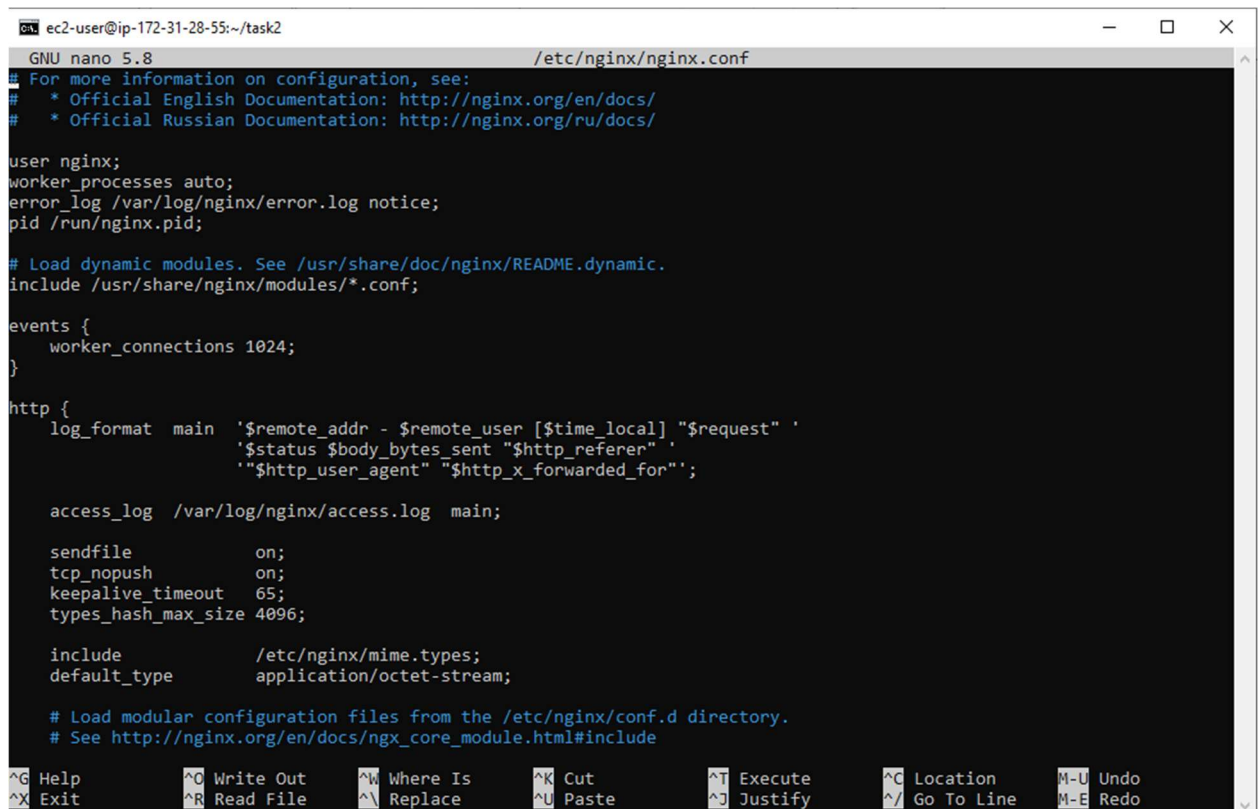
Total download size: 1.0 M
Installed size: 3.4 M
Is this ok [y/N]: y
Downloading Packages:
(1/7): libunwind-1.4.0-5.amzn2023.0.2.x86_64.rpm 883 kB/s | 66 kB 00:00
(2/7): nginx-1.24.0-1.amzn2023.0.2.x86_64.rpm 1.4 MB/s | 32 kB 00:00
(3/7): gperftools-libs-2.9.1-1.amzn2023.0.3.x86_64.rpm 2.5 MB/s | 308 kB 00:00
(4/7): nginx-core-1.24.0-1.amzn2023.0.2.x86_64.rpm 4.2 MB/s | 586 kB 00:00
(5/7): nginx-mimetypes-2.1.49-3.amzn2023.0.3.noarch.rpm 1.3 MB/s | 21 kB 00:00
(6/7): nginx-filesystem-1.24.0-1.amzn2023.0.2.noarch.rpm 642 kB/s | 9.1 kB 00:00
(7/7): generic-logos-httpd-18.0.0-12.amzn2023.0.3.noarch.rpm 290 kB/s | 19 kB 00:00
-----
Total 4.6 MB/s | 1.0 MB 00:00
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
```

Запускаємо службу Nginx і у браузері переходимо за публічною IP-адресою нашого EC2 Instance для того, щоб побачити сторінку Nginx за замовчуванням:

```
(flaskApp) [ec2-user@ip-172-31-28-55 task2]$ sudo systemctl start nginx
(flaskApp) [ec2-user@ip-172-31-28-55 task2]$ sudo systemctl enable nginx
```



Створимо файл конфігурації Nginx. Для цього вводимо команду **sudo nano /etc/nginx/nginx.conf**, після чого потрапляємо у таке вікно:



```
ec2-user@ip-172-31-28-55:~/task2
GNU nano 5.8 /etc/nginx/nginx.conf
# For more information on configuration, see:
# * Official English Documentation: http://nginx.org/en/docs/
# * Official Russian Documentation: http://nginx.org/ru/docs/

user nginx;
worker_processes auto;
error_log /var/log/nginx/error.log notice;
pid /run/nginx.pid;

# Load dynamic modules. See /usr/share/doc/nginx/README.dynamic.
include /usr/share/nginx/modules/*.conf;

events {
    worker_connections 1024;
}

http {
    log_format main '$remote_addr - $remote_user [$time_local] "$request" '
        '$status $body_bytes_sent "$http_referer" '
        '"$http_user_agent" "$http_x_forwarded_for"';

    access_log /var/log/nginx/access.log main;

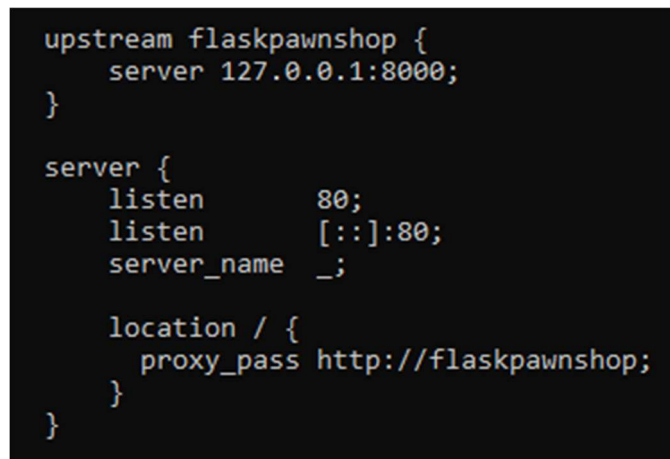
    sendfile        on;
    tcp_nopush      on;
    keepalive_timeout 65;
    types_hash_max_size 4096;

    include          /etc/nginx/mime.types;
    default_type     application/octet-stream;

    # Load modular configuration files from the /etc/nginx/conf.d directory.
    # See http://nginx.org/en/docs/nginx_core_module.html#include

^G Help      ^O Write Out  ^W Where Is   ^K Cut        ^T Execute    ^C Location   M-U Undo
^X Exit      ^R Read File  ^_ Replace    ^U Paste      ^J Justify    ^_/ Go To Line M-E Redo
```

Вносимо необхідні зміни і зберігаємо файл:

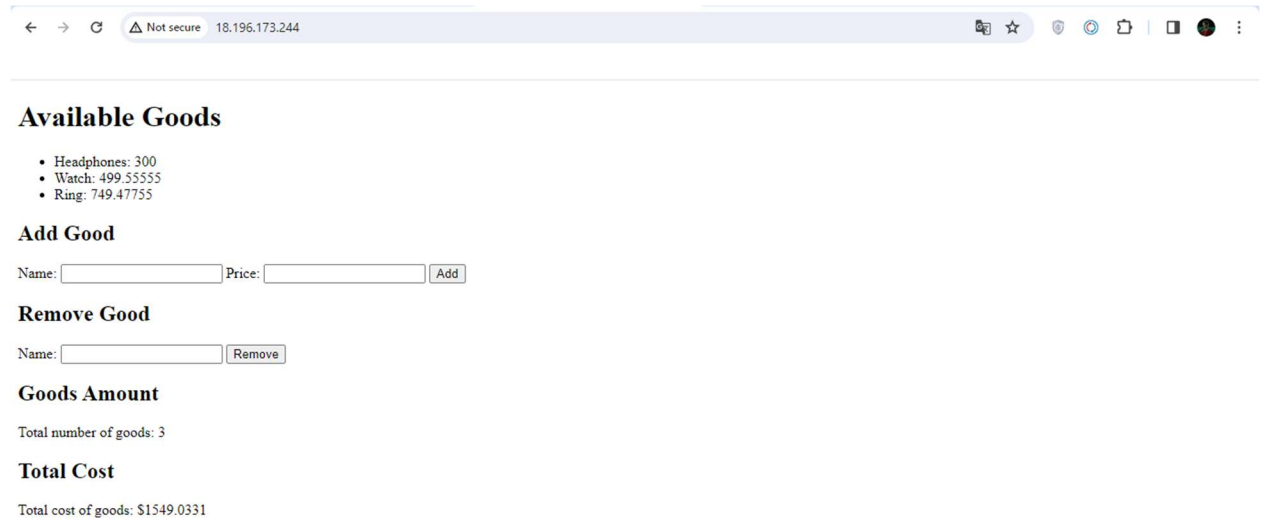


```
upstream flaskpawncshop {
    server 127.0.0.1:8000;
}

server {
    listen      80;
    listen      [::]:80;
    server_name _;

    location / {
        proxy_pass http://flaskpawncshop;
    }
}
```

Наостанок у вікні браузера переходимо за [публічною IP-адресою](#) EC2 Instance і бачимо успішно розгорнутий застосунок:



The screenshot shows a web browser window with the address bar displaying '18.196.173.244'. The page content includes a section titled 'Available Goods' with a list of items: Headphones (300), Watch (499.55555), and Ring (749.47755). Below this is an 'Add Good' form with input fields for 'Name' and 'Price', and an 'Add' button. There is also a 'Remove Good' section with a 'Name' input field and a 'Remove' button. At the bottom, the 'Goods Amount' section shows 'Total number of goods: 3', and the 'Total Cost' section shows 'Total cost of goods: \$1549.0331'.

Available Goods

- Headphones: 300
- Watch: 499.55555
- Ring: 749.47755

Add Good

Name: Price:

Remove Good

Name:

Goods Amount

Total number of goods: 3

Total Cost

Total cost of goods: \$1549.0331

Висновок: в ході виконання практичної роботи було ретельно опрацьовано два базових сервіси AWS, а саме S3 та EC2, і розглянуто такі поняття, як версіювання, міжрегіональна реплікація, EC2 Instance та ін. Насамкінець, було реалізовано розгортання Flask-застосунку на хмарній платформі AWS. Для виконання практичної роботи було залучено додаткове програмне забезпечення FileZilla, за допомогою якого значно спростився процес обміну файлами між локальним (комп'ютер) та віддаленим (EC2 Instance) сховищами даних.