

# Cloud computing Lab3 – sprawozdanie

## Rozwiązanie Etap I – cz. I

Treść

### Etap I – cz. I

- Utworzyć bazę danych PostgreSQL lub MySQL
  - Wybrać opcję Free tier
    - Konfiguracja: db.t3.micro
- Umożliwić połączenie z bazą z zewnątrz (psycopg2, MySQL-python)
- Dodanie i pobranie danych z bazy, co najmniej 1 tabela

# Tworzenie bazy danych

database-1-user15



Modify

Actions ▾

## Summary


**DB identifier**  
database-1-user15

**CPU**  
 24.61%

**Status**  
 Backing-up

**Class**  
db.t3.micro

**Role**  
Instance

**Current activity**  
 0.00 sessions

**Engine**  
PostgreSQL

**Region & AZ**  
us-west-2a

**Recommendations**

Connectivity & security

Monitoring

Logs & events

Configuration

Maintenance & backups


Data migrations - *new*

Tags

Recommendations

## Connectivity & security

### Endpoint & port

**Endpoint**  
 database-1-user15.cj888w4mq49y.us-west-2.rds.amazonaws.com

**Port**  
5432

### Networking

**Availability Zone**  
us-west-2a


**VPC**  
[vpc-084a24e5745595310](#)

**Subnet group**  
[default-vpc-084a24e5745595310](#)

**Subnets**  
[subnet-0a30120a7970b9165](#)  
[subnet-0a6db4ca95e6bdebd](#)  
[subnet-0719a47ea848dc5b2](#)  
[subnet-0834f2592551da9dc](#)

**Network type**  
IPv4

### Security

**VPC security groups**  
[default \(sg-0243a190684e478c7\)](#)  
 Active

**Publicly accessible**  
Yes

**Certificate authority** [Info](#)  
rds-ca-rsa2048-g1

**Certificate authority date**  
May 25, 2061, 00:59 (UTC+02:00)

**DB instance certificate expiration date**  
January 19, 2026, 22:32 (UTC+01:00)

# Połączenie z bazą danych

## Kod

```
import psycopg2
```

```
def connect_to_postgres(host, port, database, user, password):
```

```
    try:
```

```
        connection = psycopg2.connect(
            host=host,
            port=port,
            database=database,
            user=user,
            password=password
        )
```

```
        print("Connected")
        return connection
```

```
    except Exception as e:
```

```
        print("Error connecting")
        print(e)
        return None
```

```

def main():
    host = "database-1-user15.cj888w4mq49y.us-west-2.rds.amazonaws.com"
    port = 5432
    database = "postgres"
    user = "postgres"
    password = "radziu13"

    conn = connect_to_postgres(host, port, database, user, password)

    if conn:
        try:
            cursor = conn.cursor()

            cursor.execute("SELECT version();")
            version = cursor.fetchone()
            print(f"{version[0]}")

        finally:
            conn.close()

if __name__ == "__main__":
    main()

```

## Wykonanie

*Connected*

*PostgreSQL 16.3 on x86\_64-pc-linux-gnu, compiled by gcc (GCC) 7.3.1 20180712 (Red Hat 7.3.1-17), 64-bit*

Process finished with exit code 0

## Dodawanie i pobieranie danych

### Kod

```

import psycopg2

def connect_to_postgres(host, port, database, user, password):

    try:
        connection = psycopg2.connect(

```

```

        host=host,
        port=port,
        database=database,
        user=user,
        password=password
    )
    print("Connected")
    return connection
except Exception as e:
    print("Error connecting")
    print(e)
    return None

```

```

def create_table_query():
    return """CREATE TABLE IF NOT EXISTS person (
        first_name VARCHAR(255) NOT NULL,
        last_name VARCHAR(255) NOT NULL,
        age INT NOT NULL,
        email VARCHAR(255) NOT NULL UNIQUE
    );"""

```

```

def pupulate_table_query():
    return """INSERT INTO person (first_name, last_name, age, email)
VALUES
    ('John', 'Doe', 30, 'john.doe@example.com'),
    ('Jane', 'Smith', 25, 'jane.smith@example.com'),
    ('Michael', 'Johnson', 40, 'michael.johnson@example.com'),
    ('Emily', 'Davis', 22, 'emily.davis@example.com'),
    ('William', 'Brown', 35, 'william.brown@example.com')
ON CONFLICT DO NOTHING;"""

```

```

def get_data_from_table_query():
    return """SELECT * FROM person;"""

```

```

def main():
    host = "database-1-user15.cj888w4mq49y.us-west-2.rds.amazonaws.com"
    port = 5432
    database = "postgres"
    user = "postgres"
    password = "radziu13"

```

```

conn = connect_to_postgres(host, port, database, user, password)

```

```

if conn:
    try:
        cursor = conn.cursor()

        cursor.execute("SELECT version();")
        version = cursor.fetchone()
        print(f"{version[0]}")

        cursor.execute(create_table_query())
        conn.commit()

        cursor.execute(pupulate_table_query())
        conn.commit()

        cursor.execute(get_data_from_table_query())
        rows = cursor.fetchall()
        print("Person data:")
        for row in rows:
            print(row)

    finally:
        conn.close()

if __name__ == "__main__":
    main()

```

## Wykonanie

*Connected*

*PostgreSQL 16.3 on x86\_64-pc-linux-gnu, compiled by gcc (GCC) 7.3.1 20180712 (Red Hat 7.3.1-17), 64-bit*

*Person data:*

*('John', 'Doe', 30, 'john.doe@example.com')*

*('Jane', 'Smith', 25, 'jane.smith@example.com')*

*('Michael', 'Johnson', 40, 'michael.johnson@example.com')*

*('Emily', 'Davis', 22, 'emily.davis@example.com')*

*('William', 'Brown', 35, 'william.brown@example.com')*

*Process finished with exit code 0*

## Rozwiązanie Etap I – cz. II

### Etap I – cz. II

- Utworzyć tabelę w DynamoDB
- Utworzyć funkcje lambda:
  - Dodanie nowego rekordu do bazy na podstawie przekazanych parametrów + prosta walidacja
  - Zwrócenie rekordów wybranych na podstawie parametrów

## Tworzenie tabeli

**dynamoDB-user15** ☆



Actions ▼

### Explore table items

<

## Overview

## Indexes

## Monitor

## Global tables

## Backups

## Exports and streams

## Permissions



## Protect your DynamoDB table from accidental writes and deletes

[Edit PITR](#)


When you turn on point-in-time recovery (PITR), DynamoDB backs up your table data automatically so that you can restore to any given second in the preceding 35 days. Additional charges apply. [Learn more](#) 

### General information [Info](#)


**Partition key**  
dynamo-user15-partition-key  
(String)

Sort key

Capacity mode  
On-demand

**Table status**  
 Active

Alarms  
✔ No active alarms

**Point-in-time recovery (PITR)** [Info](#)  
 Off

Resource-based policy [Info](#)

► **Additional info**

DynamoDB updates the following information approximately every six hours.

Item count  
0

Table size  
0 bytes

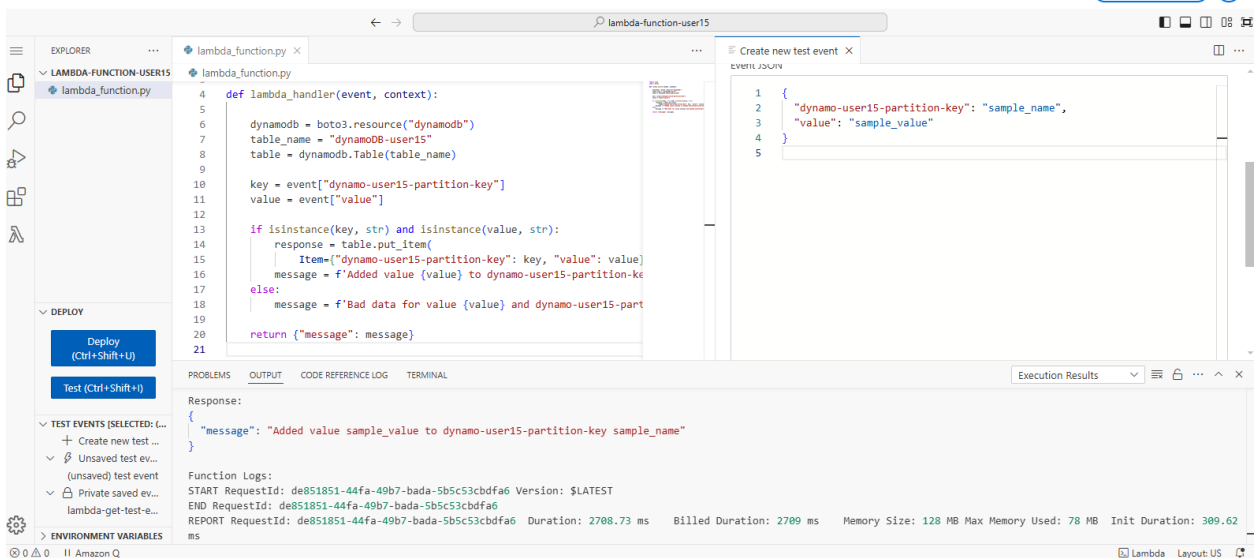
Average item size  
0 bytes

[Get live item count](#)

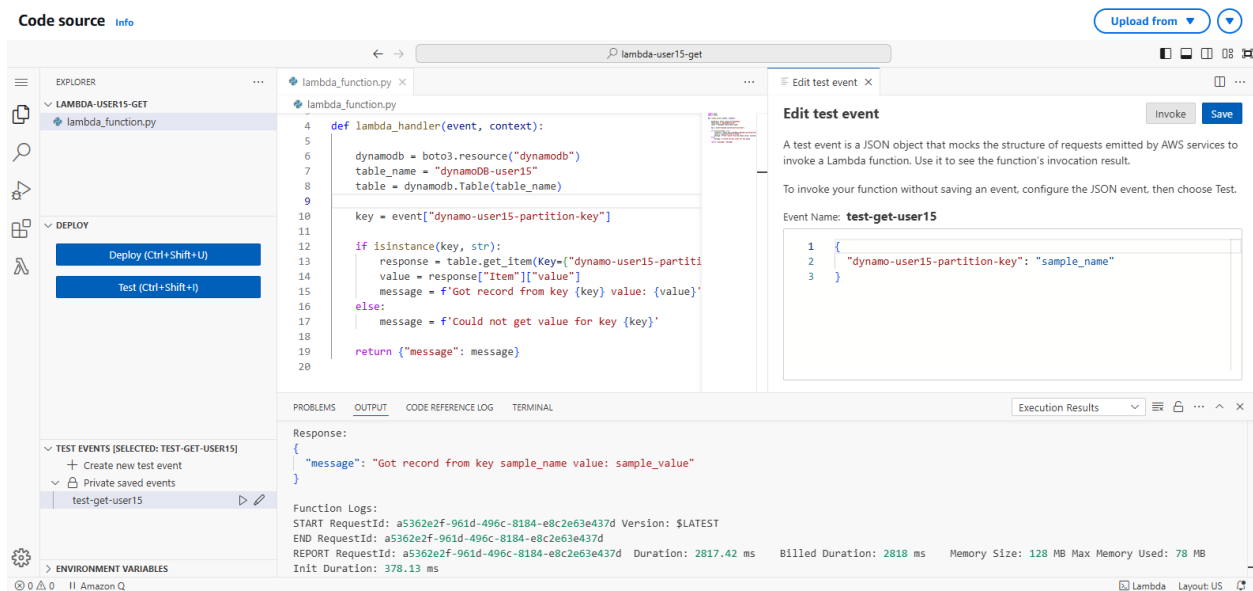
## Tworzenie funkcji lambda do dodawania rekordu z walidacją + test

[Code source](#) [Info](#)

Upload from ▼



## Tworzenie funkcji lambda do zwrócenia rekordów na podstawie parametrów + test



## Rozwiązanie Etap II

### Etap II

- Utworzyć API Gateway wywołujące funkcje lambda
- Wywołać endpoint API Gateway z lokalnej aplikacji
- Utworzyć bucket w S3
  - Dodać 1 plik do 100 kb
  - Dodać funkcje lambda:
    - Do wyświetlenia listy plików
    - Do wyszukania pliku po nazwie



## Tworzenie API Gateway wywołujący funkcję lambda

Create resource

/

/resource-get-user15

POST

/resource-put-user15

POST

arn:aws:execute-api:us-west-2:864981746296:9ya1ocqd37/\*

POST/resource-put-user15

Resource ID

6vc5rw

Client

Method request

Integration request

Method response

Integration response

Lambda integration

Method request

Integration request

Integration response

Method response

Test

Test method

Make a test call to your method. When you make a test call, API Gateway skips authorization and directly invokes your method.

Query strings

param1=value1&param2=value2

Headers

Enter a header name and value separated by a colon (:). Use a new line for each header.

header1:value1  
header2:value2

Client certificate

No client certificates have been generated.

Request body

```

1 {
2   "dynamo-user15-partition-key": "sample_key",
3   "value": "sample_value"
4 }
```

## Resources

[Create resource](#)

/

/resource-get-user15

POST

/resource-put-user15

POST

**/resource-get-user15 - POST - Method execution**

[API actions](#)

Deploy API

[Update documentation](#)[Delete](#)

ARN

arn:aws:execute-api:us-west-2:864981746296:9ya1ocqd37/\*/\*/POST/resource-get-user15

Resource ID

529xfb

Client

→

Method request

→

Integration request

→

Lambda integration

←

Method response

←

Integration response

←

Method request

Integration request

Integration response

Method response

**Test**

**Test method**

Make a test call to your method. When you make a test call, API Gateway skips authorization and directly invokes your method.

**Query strings**

param1=value1&param2=value2

**Headers**

Enter a header name and value separated by a colon (:). Use a new line for each header.

header1:value1  
header2:value2

**Client certificate**

No client certificates have been generated.

**Request body**

1 {

2 "dynamo-user15-partition-key": "sample\_name"

3 }

## Wywołanie endpointa API Gateway z lokalnej aplikacji

### Kod

```
import json
```

```
import requests
```

```
URL = 'https://9ya1ocqd37.execute-api.us-west-2.amazonaws.com/stage-user15'
```

```
def add_record(key, value):
```

```
    url = URL + '/resouce-put-user15'
```

```
    payload = {  
        "dynamo-user15-partition-key": key,  
        "value": "value"  
    }
```

```
    headers = {'content-type': 'application/json'}
```

```

try:
    response = requests.post(url, data=json.dumps(payload), headers=headers)
    response.raise_for_status()
    return response.json()
except requests.exceptions.HTTPError as e:
    print('Http Error')
    print(e)

return None

```

```
def get_record(key):
```

```

    url = URL + '/resouce-get-user15'

```

```

    payload = {"dynamo-user15-partition-key": key}

```

```

    headers = {'content-type': 'application/json'}

```

```

try:
    response = requests.post(url, data=json.dumps(payload), headers=headers)
    response.raise_for_status()
    return response.json()
except requests.exceptions.HTTPError as e:
    print('Http Error')
    print(e)

return None

```

```

if __name__ == '__main__':
    key = 'local_sample_key'
    value = 'local_sample_value'

```

```

    print(add_record(key, value))
    print(get_record(key))

```

## Wykonanie

```
{'message': 'Added value value to dynamo-user15-partition-key local_sample_key'}
```

```
{'message': 'Got record from key local_sample_key value: value'}
```

*Process finished with exit code 0*

# Tworzenie bucketu w S3

## Tworzenie bucketu

bucket-user15 [Info](#)

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

**Objects (0)** [Info](#)

🔄 Copy S3 URI | 📄 Copy URL | ⬇️ Download | 🔗 Open | 🗑️ Delete | ⚙️ Actions | 📁 Create folder | 📤 Upload

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](#)

< 1 > ⚙️

| Name                                       | Type | Last modified | Size | Storage class |
|--------------------------------------------|------|---------------|------|---------------|
| No objects                                 |      |               |      |               |
| You don't have any objects in this bucket. |      |               |      |               |

## Wgrywanie pliku

**Upload succeeded**  
For more information, see the [Files and folders](#) table.

**Upload: status** [Close](#)

ⓘ After you navigate away from this page, the following information is no longer available.

**Summary**

|                                          |                                                 |                                      |
|------------------------------------------|-------------------------------------------------|--------------------------------------|
| <b>Destination</b><br>s3://bucket-user15 | <b>Succeeded</b><br>🟢 1 file, 89.2 KB (100.00%) | <b>Failed</b><br>🔴 0 files, 0 B (0%) |
|------------------------------------------|-------------------------------------------------|--------------------------------------|

[Files and folders](#) | [Configuration](#)

**Files and folders** (1 total, 89.2 KB)

| Name                             | Folder | Type                         | Size    | Status      | Error |
|----------------------------------|--------|------------------------------|---------|-------------|-------|
| <a href="#">wine+quality.zip</a> | -      | application/x-zip-compressed | 89.2 KB | 🟢 Succeeded | -     |

## Funkcje do wyświetlania listy plików i wyszukania pliku po nazwie

**Code source** [Info](#) [Upload from](#)

**EXPLORER**

- LAMBDA-S3-LIST-USER15
  - lambda\_function.py

**DEPLOY**

Deploy (Ctrl+Shift+U)

Test (Ctrl+Shift+I)

**TEST EVENTS [SELECTED: (UNSAVED) TEST EVENT]**

+ Create new test event

🔍 Unsaved test events (unsaved) test event

**ENVIRONMENT VARIABLES**

0 0 0 Amazon Q

**lambda\_function.py**

```
1 import boto3
2
3 def lambda_handler(event, context):
4
5     s3 = boto3.client('s3')
6
7     response = s3.list_objects_v2(Bucket="bucket-user15")
8     res = response['Contents']
9
10    if res:
11        files = [item['Key'] for item in res]
12    else:
13        files = []
14
15    message = f'List of files: {files}'
16
17    return {"message": message}
18
```

**Create new test event**

☒ Private  
This event is only available in the Lambda Console and to the event creator. You can configure a total of ten. [Learn more](#)

☐ Shareable  
This event is available to IAM users within the same account who have permissions to access and use shareable events. [Learn more](#)

Template - optional  
Hello World

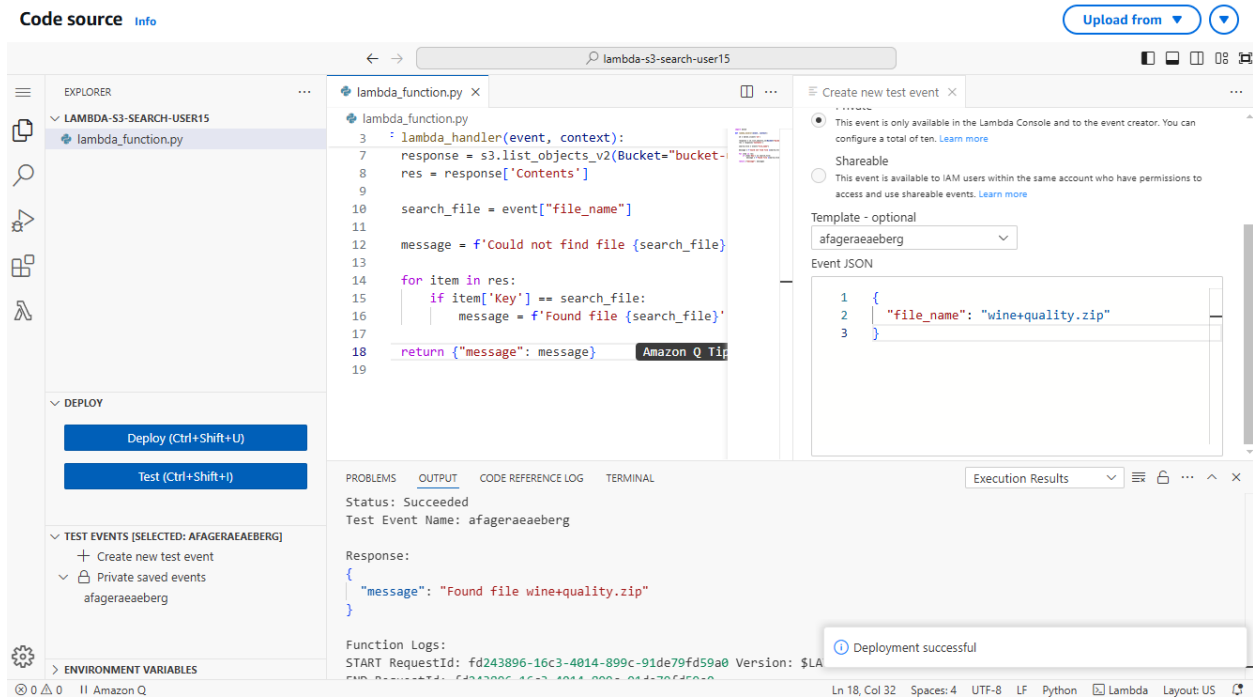
Event JSON  
1

**PROBLEMS** **OUTPUT** **CODE REFERENCE LOG** **TERMINAL**

Status: Succeeded  
Test Event Name: (unsaved) test event

Response:  
{  
 "message": "List of files: ['wine+quality.zip']"  
}

Function Logs:  
START RequestId: c7c902b2-4f66-460e-a9d1-9688d6ccac04 Version: \$LATEST



## Rozwiązanie Etap III


### Etap III

- Uruchomić instancję maszyny wirtualnej o następujących parametrach:
  - System operacyjny – Amazon Linux 2023 AMI
  - Typ instancji – t3.micro/t2.micro
- Nawiązać połączenie z maszyną wirtualną:
  - Poprzez EC2 Instance Connect
  - Nawiązać połączenie z relacyjną bazą danych i pobranie/dodanie rekordu

# Tworzenie maszyny wirtualnej

## Instance summary for i-0d429d6745c349155 (vm-user15) Info

Updated less than a minute ago



Connect

Instance state ▼

Actions ▼

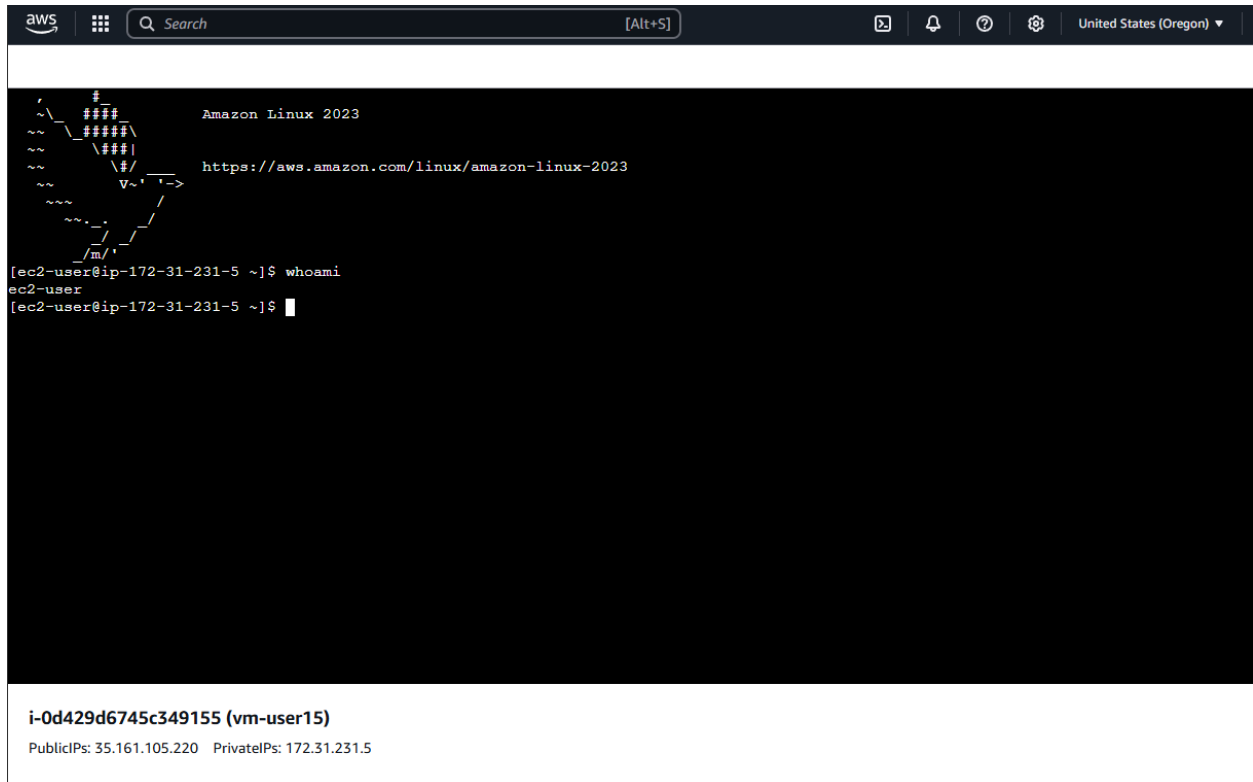
|                                                                             |                                                                                        |                                                                                                                                                                                                                                                                                                        |
|-----------------------------------------------------------------------------|----------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Instance ID</b><br>i-0d429d6745c349155                                   | <b>Public IPv4 address</b><br>35.161.105.220   <a href="#">open address</a>            | <b>Private IPv4 addresses</b><br>172.31.231.5                                                                                                                                                                                                                                                          |
| <b>IPv6 address</b><br>-                                                    | <b>Instance state</b><br>Running                                                       | <b>Public IPv4 DNS</b><br>ec2-35-161-105-220.us-west-2.compute.amazonaws.com   <a href="#">open address</a>                                                                                                                                                                                            |
| <b>Hostname type</b><br>IP name: ip-172-31-231-5.us-west-2.compute.internal | <b>Private IP DNS name (IPv4 only)</b><br>ip-172-31-231-5.us-west-2.compute.internal   |                                                                                                                                                                                                                                                                                                        |
| <b>Answer private resource DNS name</b><br>-                                | <b>Instance type</b><br>t3.micro                                                       | <b>Elastic IP addresses</b><br>-                                                                                                                                                                                                                                                                       |
| <b>Auto-assigned IP address</b><br>35.161.105.220 [Public IP]               | <b>VPC ID</b><br>vpc-00b02f6161149dfe5                                                 | <b>AWS Compute Optimizer finding</b><br><div>⊗</div> User: arn:aws:iam::864981746296:user/user_15 is not authorized to perform: compute-optimizer:GetEnrollmentStatus on resource: * because no identity-based policy allows the compute-optimizer:GetEnrollmentStatus action<br><a href="#">Retry</a> |
| <b>IAM Role</b><br>-                                                        | <b>Subnet ID</b><br>subnet-0e9a6b28d6169debb (subnet-us-west-2a)                       | <b>Auto Scaling Group name</b><br>-                                                                                                                                                                                                                                                                    |
| <b>IMDSv2</b><br>Required                                                   | <b>Instance ARN</b><br>arn:aws:ec2:us-west-2:864981746296:instance/i-0d429d6745c349155 | <b>Managed</b><br>false                                                                                                                                                                                                                                                                                |
| <b>Operator</b><br>-                                                        |                                                                                        |                                                                                                                                                                                                                                                                                                        |

- Details
- Status and alarms
- Monitoring
- Security
- Networking
- Storage
- Tags

### ▼ Instance details Info

|                                                                   |                                                                                                                    |                                                                              |
|-------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------|
| <b>AMI ID</b><br>ami-093a4ad9a8cc370f4                            | <b>Monitoring</b><br>disabled                                                                                      | <b>Platform details</b><br>Linux/UNIX                                        |
| <b>AMI name</b><br>al2023-ami-2023.6.20250115.0-kernel-6.1-x86_64 | <b>Allowed image</b><br>-                                                                                          | <b>Termination protection</b><br>Disabled                                    |
| <b>Stop protection</b><br>Disabled                                | <b>Launch time</b><br>Mon Jan 20 2025 00:02:38 GMT+0100 (czas środkowoeuropejski standardowy) (less than a minute) | <b>AMI location</b><br>amazon/al2023-ami-2023.6.20250115.0-kernel-6.1-x86_64 |

## Nawiązanie połączenia z maszyną wirtualną



## Nawiązanie połączenia z bazą danych, dodanie/pobranie rekordu

```
ec2-user@ip-172-31-231-5 ~]$
ec2-user@ip-172-31-231-5 ~]$ psql --host=database-1-user15.cj888w4mq49y.us-west-2-rds.amazonaws.com
Password for user ec2-user:
psql: error: connection to server at "database-1-user15.cj888w4mq49y.us-west-2-rds.amazonaws.com" (52.42.33.177), port 5432 failed: FATAL: password authentication failed for user "ec2-user"
connection to server at "database-1-user15.cj888w4mq49y.us-west-2-rds.amazonaws.com" (52.42.33.177), port 5432 failed: FATAL: no pg_hba.conf entry for host "35.161.105.220", user "ec2-user", database "ec2-user", no encryption
ec2-user@ip-172-31-231-5 ~]$ psql --host=database-1-user15.cj888w4mq49y.us-west-2-rds.amazonaws.com --username=postgres
Password for user postgres:
psql (15.9, server 16.3)
WARNING: psql major version 15, server major version 16.
         Some psql features might not work.
SSL connection (protocol: TLSv1.3, cipher: TLS_AES_256_GCM_SHA384, compression: off)
Type "help" for help.

postgres=> \l
          List of databases
  Name | Owner | Encoding | Collate | Ctype | ICU Locale | Locale Provider | Access privileges
-----+-----+-----+-----+-----+-----+-----+-----
postgres | postgres | UTF8 | en_US.UTF-8 | en_US.UTF-8 | | libc | |
rdsadmin | rdsadmin | UTF8 | en_US.UTF-8 | en_US.UTF-8 | | libc | | rdsadmin=Ctc/rdsadmin
templatel | rdsadmin | UTF8 | en_US.UTF-8 | en_US.UTF-8 | | libc | | =c/rdsadmin
templatel | postgres | UTF8 | en_US.UTF-8 | en_US.UTF-8 | | libc | | rdsadmin=Ctc/rdsadmin
templatel | postgres | UTF8 | en_US.UTF-8 | en_US.UTF-8 | | libc | | =c/postgres
(4 rows)

postgres=> \dt
          List of relations
 Schema | Name | Type | Owner
-----+-----+-----+-----
public | person | table | postgres
(1 row)

postgres=> select * from person
postgres-> ;
 first_name | last_name | age | email
-----+-----+-----+-----
 John      | Doe       | 30  | john.doe@example.com
 Jane      | Smith     | 25  | jane.smith@example.com
 Michael   | Johnson   | 40  | michael.johnson@example.com
 Emily     | Davis     | 22  | emily.davis@example.com
 William   | Brown     | 35  | william.brown@example.com
(5 rows)

postgres=> insert into person values ('aaa', 'AAA', 33, 'aaa.AAA@gmail.com');
INSERT 0 1
postgres=> select * from person
 first_name | last_name | age | email
-----+-----+-----+-----
 John      | Doe       | 30  | john.doe@example.com
 Jane      | Smith     | 25  | jane.smith@example.com
 Michael   | Johnson   | 40  | michael.johnson@example.com
 Emily     | Davis     | 22  | emily.davis@example.com
 William   | Brown     | 35  | william.brown@example.com
 aaa       | AAA       | 33  | aaa.AAA@gmail.com
(6 rows)

postgres=>
```