Name: Tushar Panchal

En.No: 21162101014

Sub: EADC (Enterprise Application Development for Cloud)

Branch: CBA

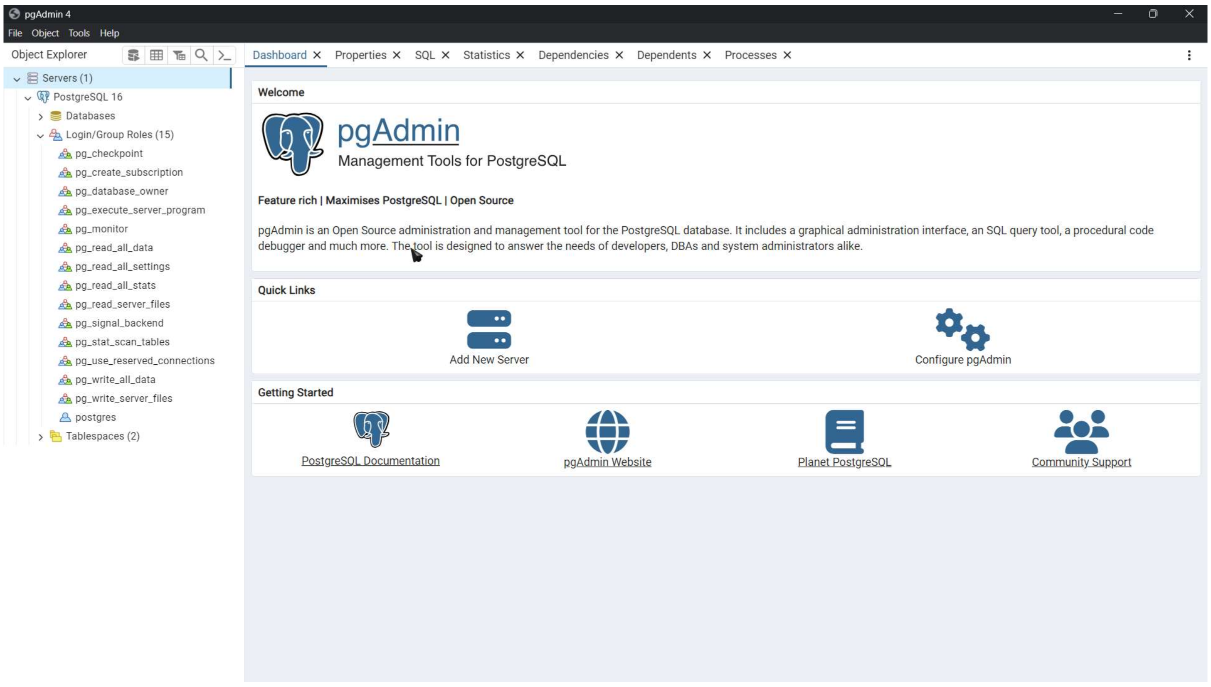
Batch:61

---------------------------PRACTICAL 19----------------------------

Implement an application with CRUD operation using Postgres with NodeJS.
A social media website xyz.com wants to manage its users records and this task is given to you. They are seeking functionalities:

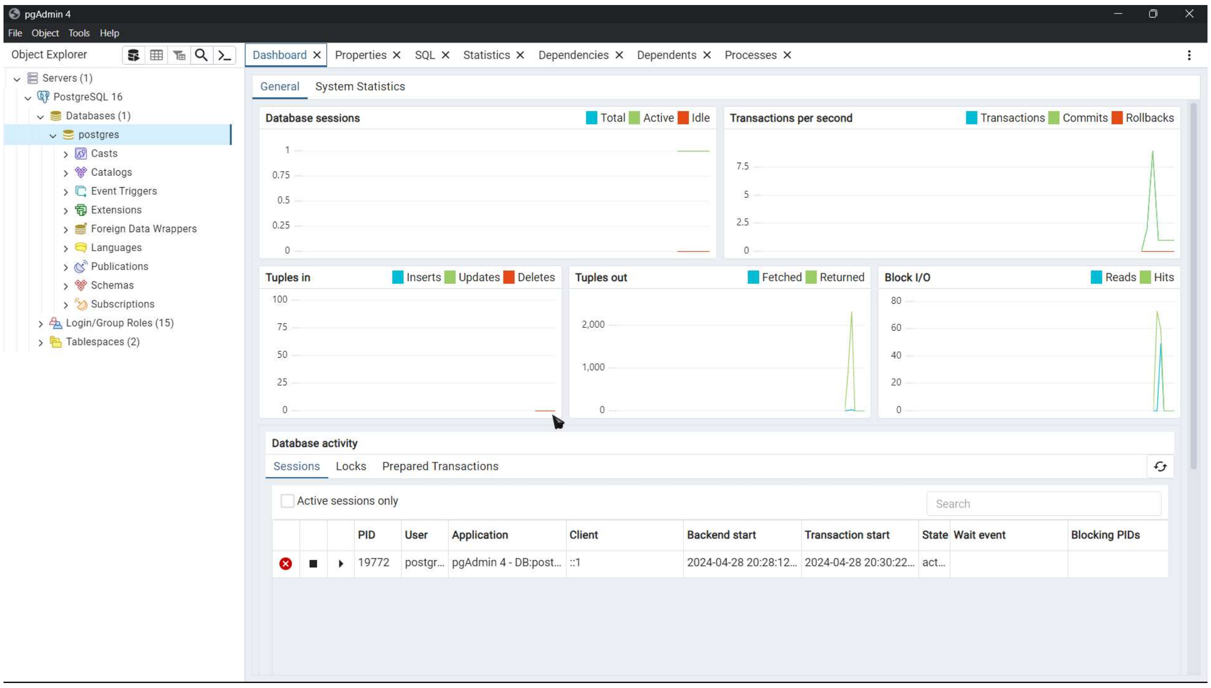
Practical 19.1: New joiner data should be stored in DB.
Practical 19.2: The admin can see the whole DB,
Practical 19.3: The admin can filter out the DB.
Practical 19.4: The user can update their information, which should be updated in the DB.
Practical 19.5: The admin can delete the record from DB.
Practical 19.6: Integrate your Nodejs application with the IBM cloud service- Databases for Postgresql and check for the database status using pg admin.

First login in to pgAdmin



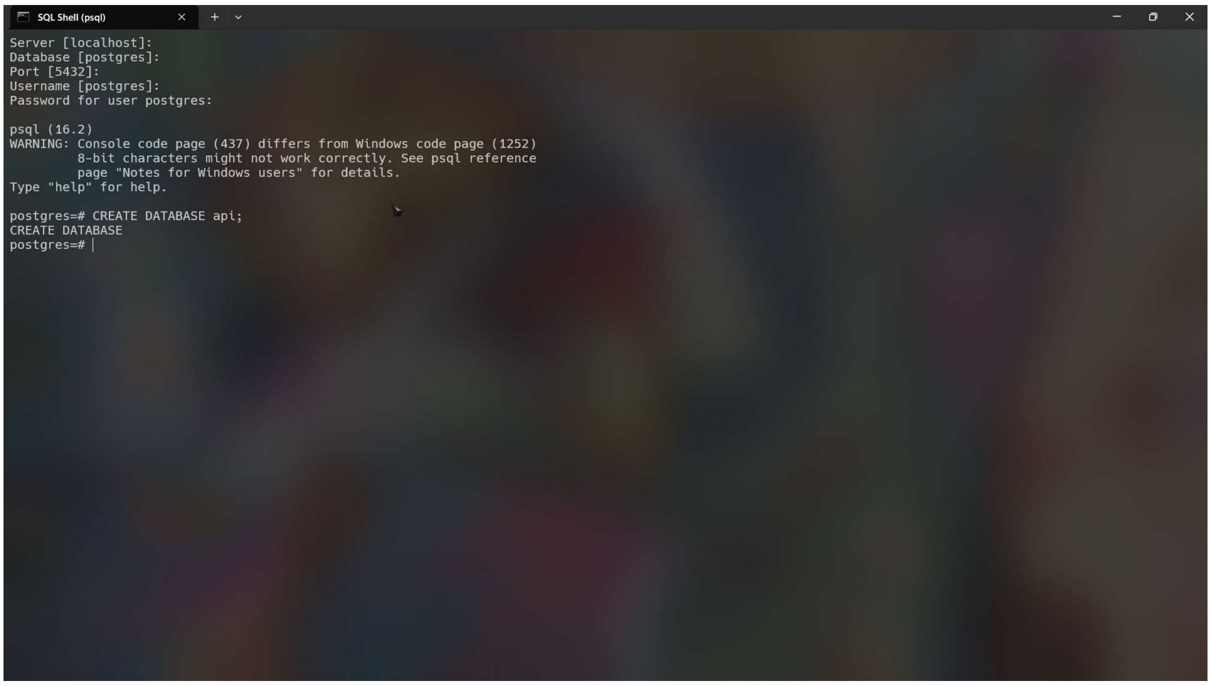
Feature rich | Maximises PostgreSQL | Open Source

pgAdmin is an Open Source administration and management tool for the PostgreSQL database. It includes a graphical administration interface, an SQL query tool, a procedural code debugger and much more. The tool is designed to answer the needs of developers, DBAs and system administrators alike.



			PID	User	Application	Client	Backend start	Transaction start	State	Wait event	Blocking PIDs
⊗	■	▸	19772	postgr...	pgAdmin 4 - DB:post...	::1	2024-04-28 20:28:12...	2024-04-28 20:30:22...	act...		

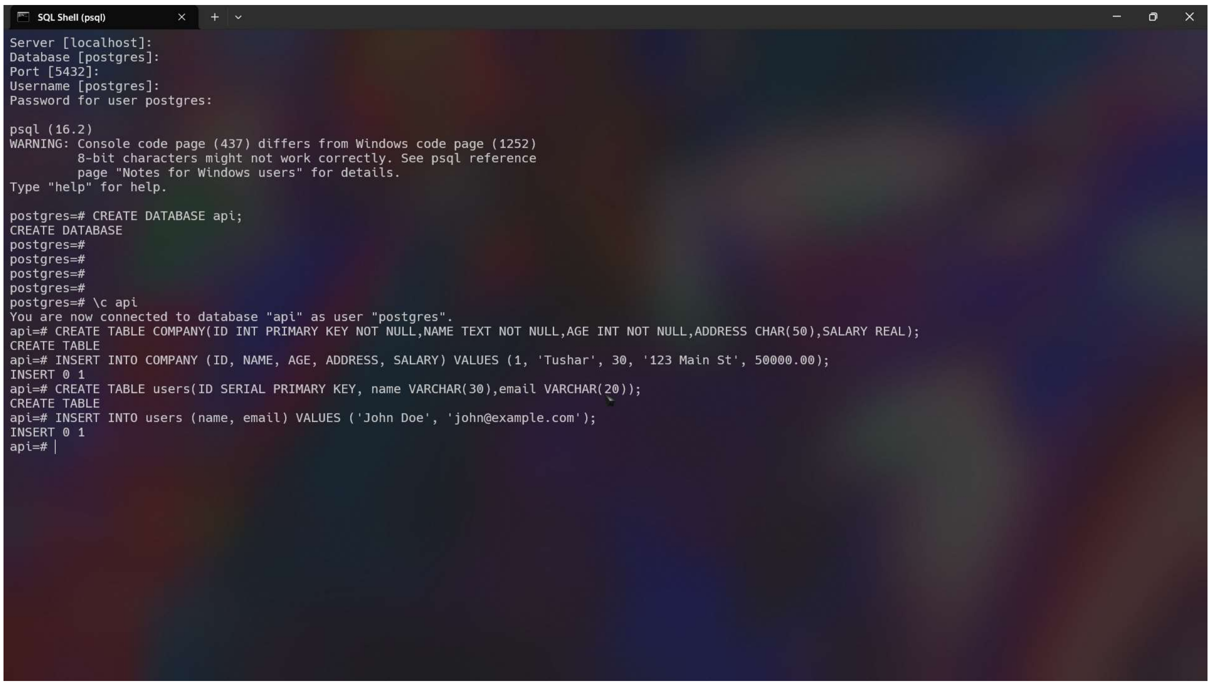
Now open SQL shell then click enter set all default than give password of postgres create database name api



```
Server [localhost]:
Database [postgres]:
Port [5432]:
Username [postgres]:
Password for user postgres:

psql (16.2)
WARNING: Console code page (437) differs from Windows code page (1252)
         8-bit characters might not work correctly. See psql reference
         page "Notes for Windows users" for details.
Type "help" for help.

postgres=# CREATE DATABASE api;
CREATE DATABASE
postgres=# |
```

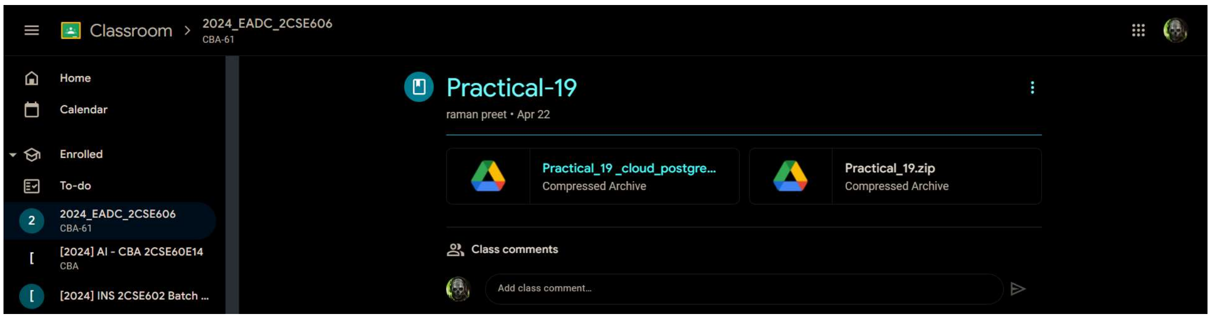
Now use \c api command to go in to api database

And create some tables and insert values too



```
Server [localhost]:
Database [postgres]:
Port [5432]:
Username [postgres]:
Password for user postgres:

psql (16.2)
WARNING: Console code page (437) differs from Windows code page (1252)
         8-bit characters might not work correctly. See psql reference
         page "Notes for Windows users" for details.
Type "help" for help.

postgres=# CREATE DATABASE api;
CREATE DATABASE
postgres=#
postgres=#
postgres=#
postgres=#
postgres=# \c api
You are now connected to database "api" as user "postgres".
api=# CREATE TABLE COMPANY(ID INT PRIMARY KEY NOT NULL,NAME TEXT NOT NULL,AGE INT NOT NULL,ADDRESS CHAR(50),SALARY REAL);
CREATE TABLE
api=# INSERT INTO COMPANY (ID, NAME, AGE, ADDRESS, SALARY) VALUES (1, 'Tushar', 30, '123 Main St', 50000.00);
INSERT 0 1
api=# CREATE TABLE users(ID SERIAL PRIMARY KEY, name VARCHAR(30),email VARCHAR(20));
CREATE TABLE
api=# INSERT INTO users (name, email) VALUES ('John Doe', 'john@example.com');
INSERT 0 1
api=# |
```

Now download this code and open in VS code



2024_EADC_2CSE606
CBA-61

Practical-19
raman preet • Apr 22

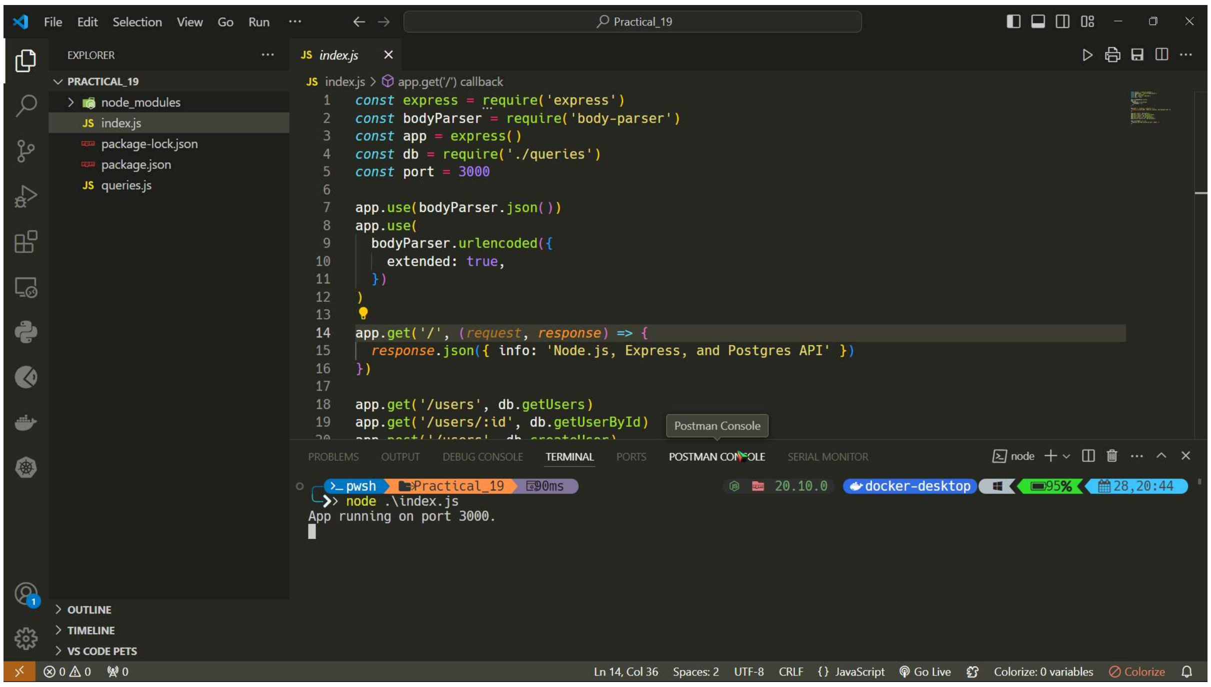
Practical_19_cloud_postgre...
Compressed Archive

Practical_19.zip
Compressed Archive

Class comments

Add class comment...

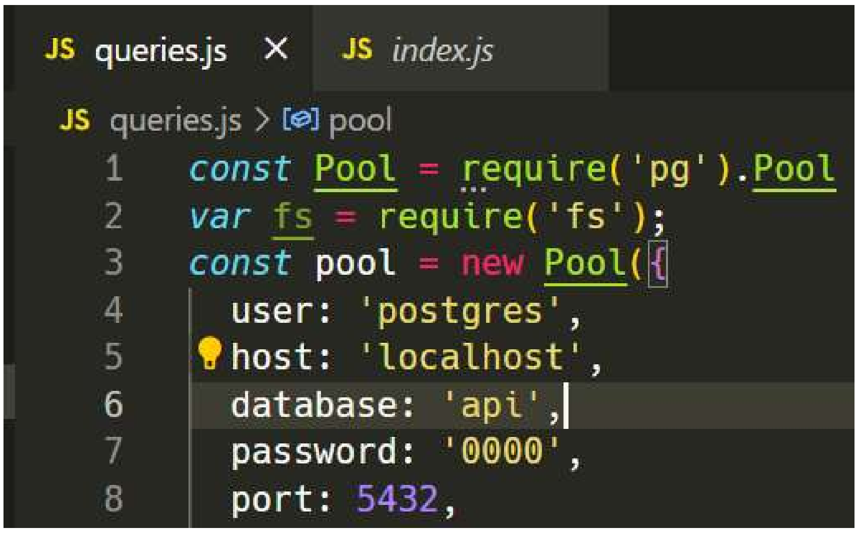
Now no need to change anything and run the code Command:-
Node index.js



```
const express = require('express')
const bodyParser = require('body-parser')
const app = express()
const db = require('./queries')
const port = 3000

app.use(bodyParser.json())
app.use(
  bodyParser.urlencoded({
    extended: true,
  })
)

app.get('/', (request, response) => {
  response.json({ info: 'Node.js, Express, and Postgres API' })
})

app.get('/users', db.getUsers)
app.get('/users/:id', db.getUserById)
```

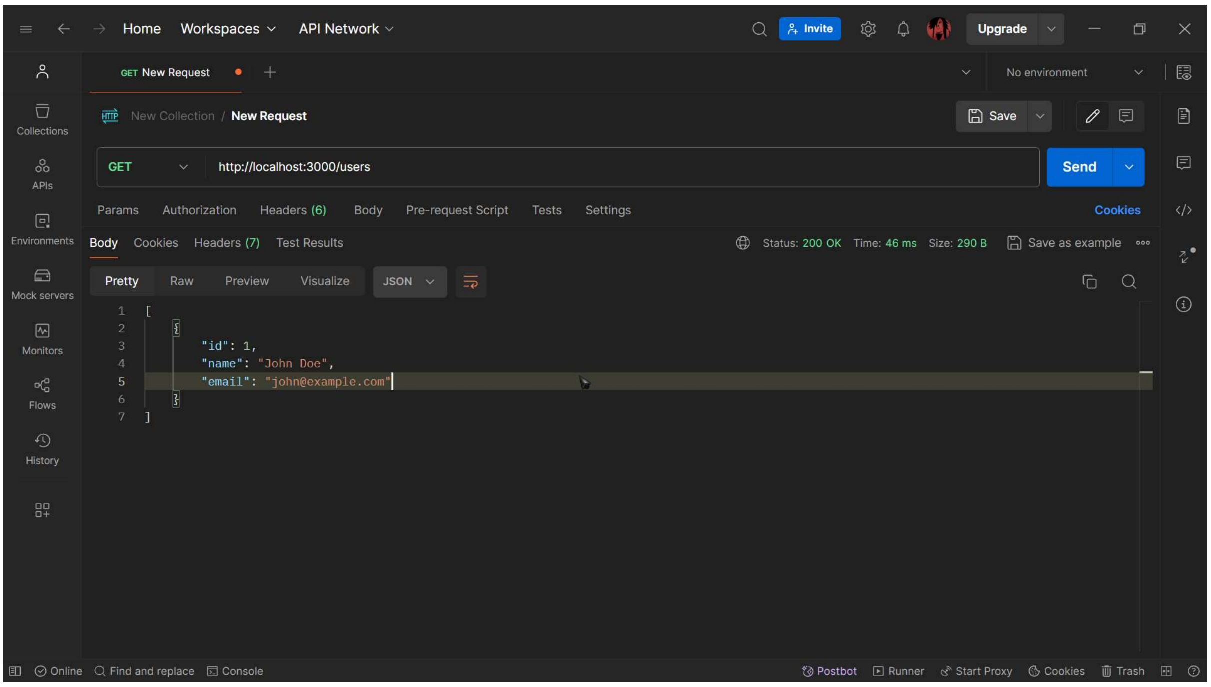
```
> node .\index.js
App running on port 3000.
```

Just change port number and password of yours



```
const Pool = require('pg').Pool
var fs = require('fs');
const pool = new Pool({
  user: 'postgres',
  host: 'localhost',
  database: 'api',
  password: '0000',
  port: 5432,
```

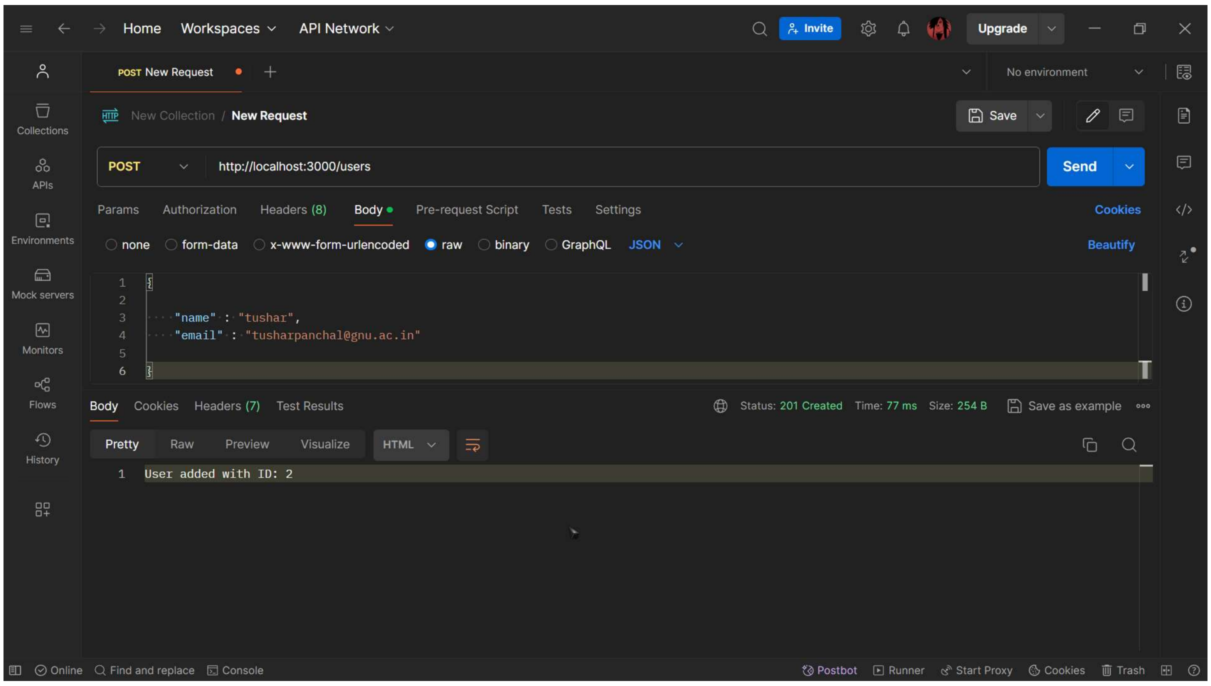
Now open postman and the GET request command : **http://localhost:3000/users** and we can get the users data


GET http://localhost:3000/users

Status: 200 OK Time: 46 ms Size: 290 B

```
[
    {
        "id": 1,
        "name": "John Doe",
        "email": "john@example.com"
    }
]
```

Now i posted my data in users by POST request

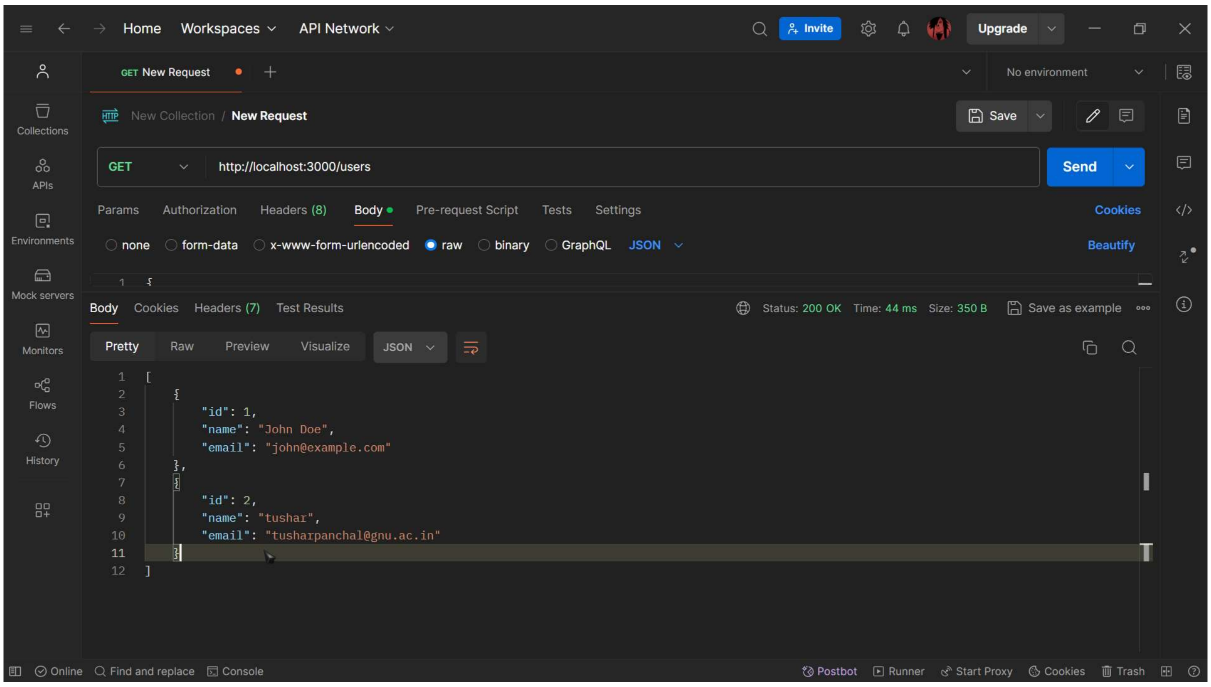

POST http://localhost:3000/users

```
{

    "name" : "tushar",
    "email" : "tusharpanchal@gnu.ac.in"

}
```

Status: 201 Created Time: 77 ms Size: 254 B

```
User added with ID: 2
```

We can see my data showing on no.2



GET New Request

New Collection / New Request

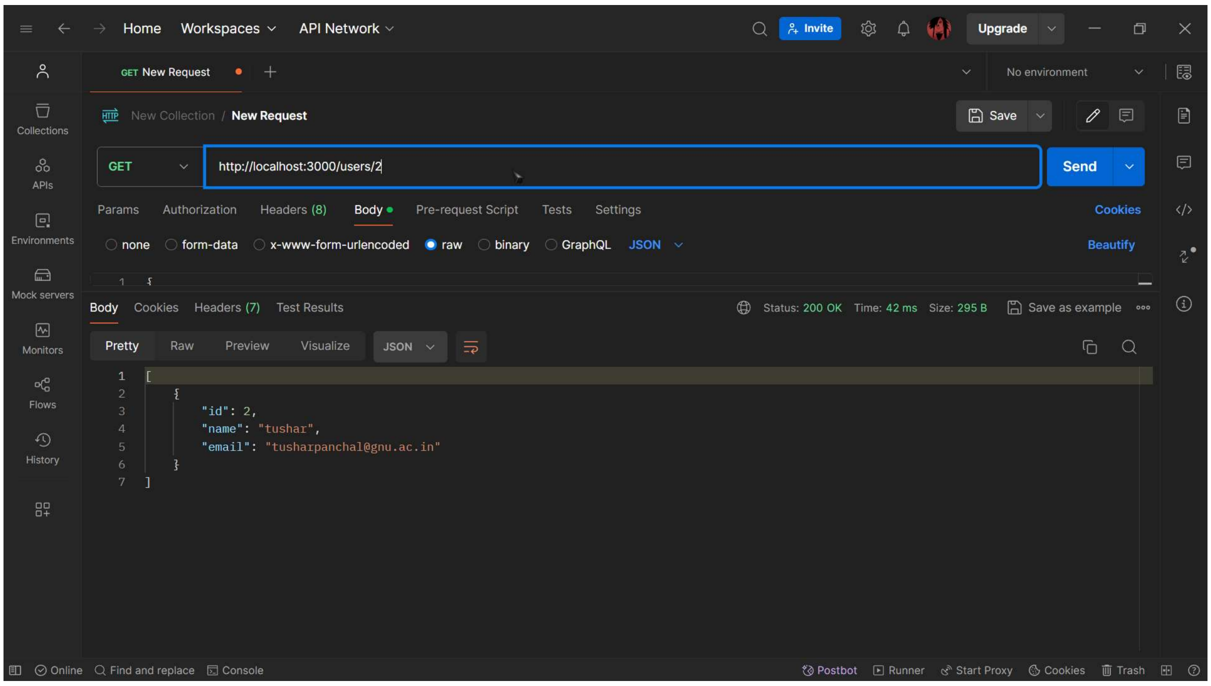
GET http://localhost:3000/users

Body — raw — JSON

Status: 200 OK Time: 44 ms Size: 350 B

```
[
    {
        "id": 1,
        "name": "John Doe",
        "email": "john@example.com"
    },
    {
        "id": 2,
        "name": "tushar",
        "email": "tusharpanchal@gnu.ac.in"
    }
]
```

To see only my data use this command
http://localhost:3000/users/2



GET New Request

New Collection / New Request

GET http://localhost:3000/users/2

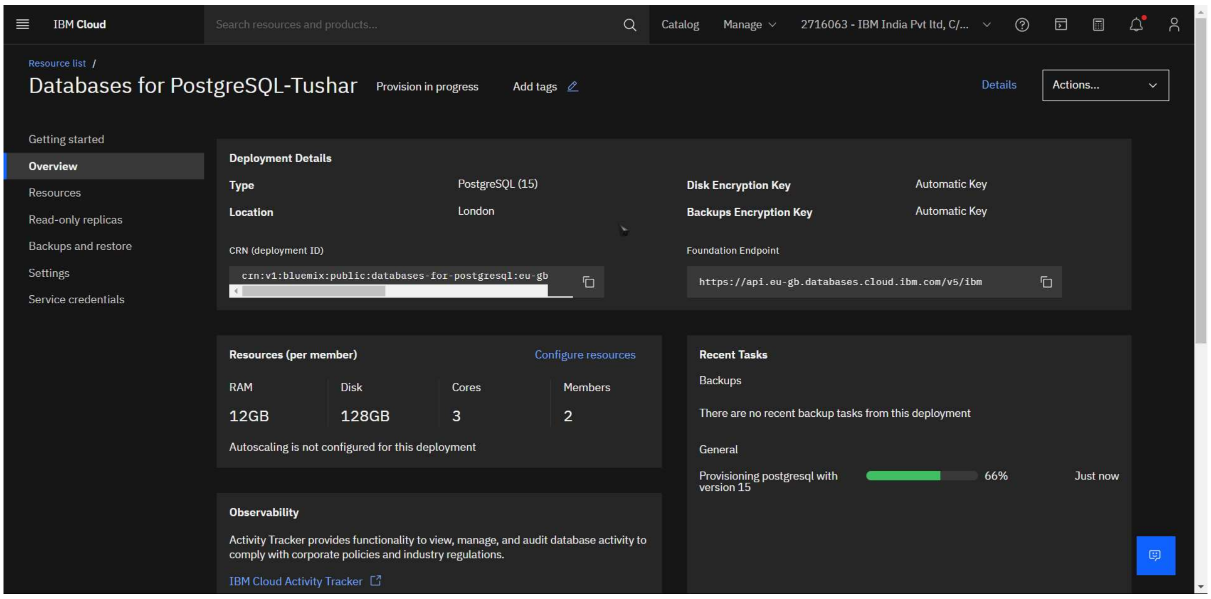
Body — raw — JSON

Status: 200 OK Time: 42 ms Size: 295 B

```
[
    {
        "id": 2,
        "name": "tushar",
        "email": "tusharpanchal@gnu.ac.in"
    }
]
```

Now open IBM cloud and open Database for postgreSQL-ts

Create if not exists, the PostgreSQL service on IBM Cloud and create if not exist its service credentials


Resource list /

Databases for PostgreSQL-Tushar   Provision in progress   Add tags

Deployment Details

Type	PostgreSQL (15)	**Disk Encryption Key**	Automatic Key
Location	London	**Backups Encryption Key**	Automatic Key

CRN (deployment ID): `crn:v1:bluemix:public:databases-for-postgresql:eu-gb`

Foundation Endpoint: `https://api.eu-gb.databases.cloud.ibm.com/v5/ibm`

Resources (per member)   Configure resources

RAM	Disk	Cores	Members
12GB	128GB	3	2

Autoscaling is not configured for this deployment

Recent Tasks

Backups

There are no recent backup tasks from this deployment

General

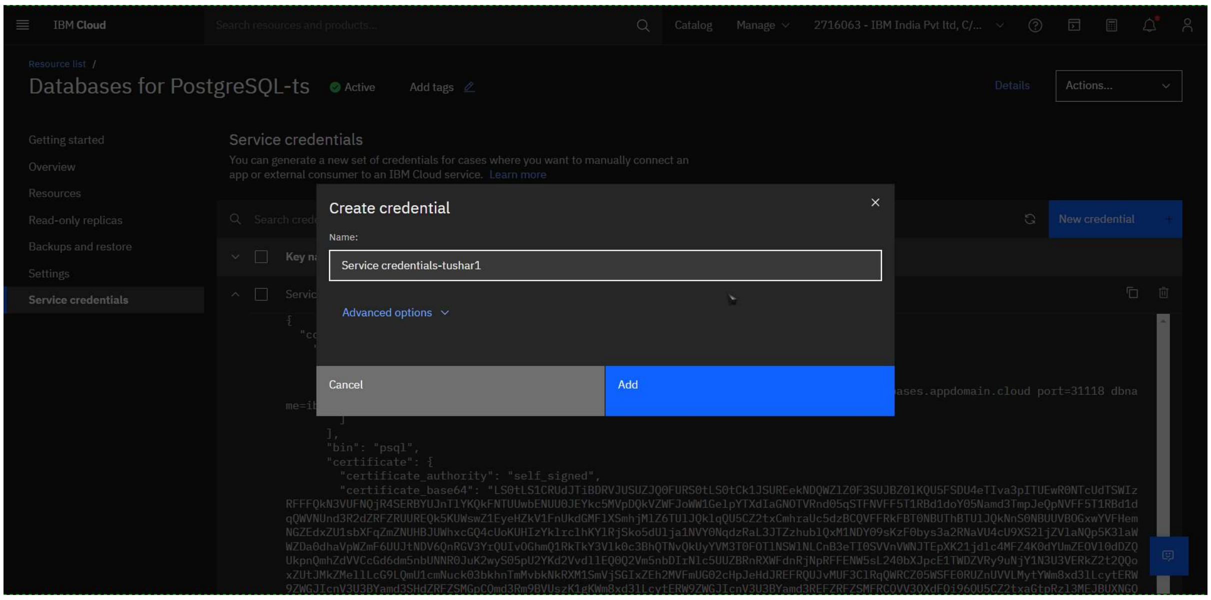
Provisioning postgresql with version 15   66%   Just now

Observability

Activity Tracker provides functionality to view, manage, and audit database activity to comply with corporate policies and industry regulations.

IBM Cloud Activity Tracker

Create new credential


Resource list /

Databases for PostgreSQL-ts   Active   Add tags

Service credentials

You can generate a new set of credentials for cases where you want to manually connect an app or external consumer to an IBM Cloud service. Learn more

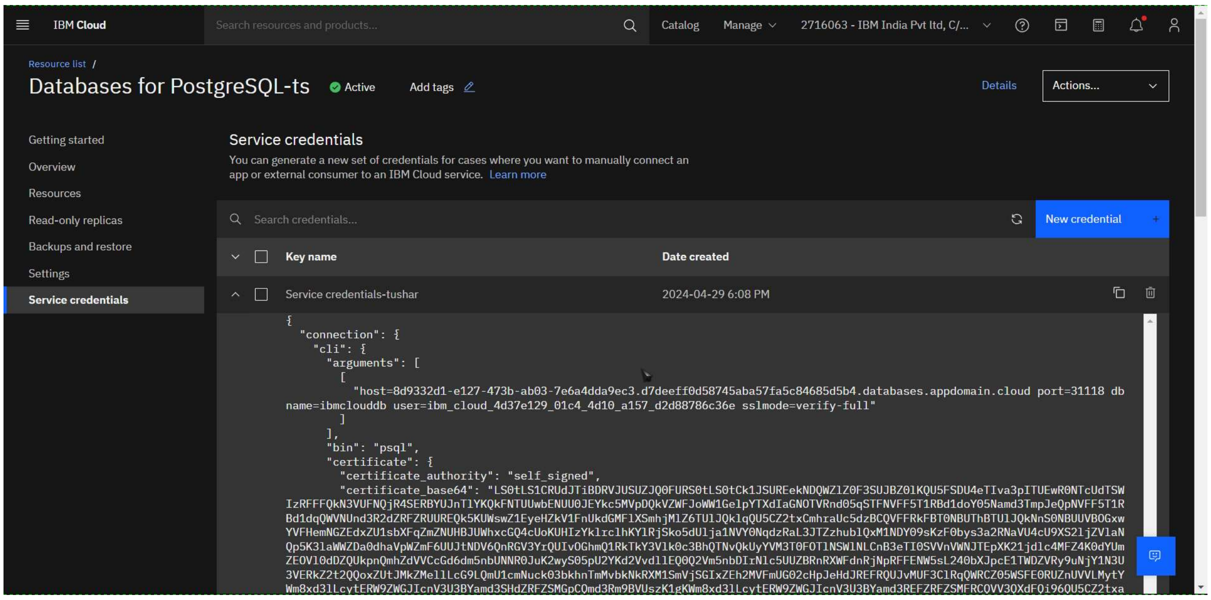
Create credential

Name: Service credentials-tushar1

Advanced options

```
{
  "co
...ases.appdomain.cloud port=31118 dbna
me=ib
      ]
    ],
    "bin": "psql",
    "certificate": {
      "certificate_authority": "self_signed",
      "certificate_base64": "LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1JSUREekNDQWZlZ0F3SUJBZ0lKQU5FSDU4eTIva3pITUEwR0NTcUdTSIz
RFFFQkN3VUFNQjR4SERBYUJnTlYKQkFNTUUwbENUU0JEYkc5MVpDQkVZWFJoWW1GelpYTXdIaGNOTVRnd05qSTFNVFF5T1RBd1doY05Namd3TmpJeQpNVFF5T1RBd1d
qQWVNUnd3R2dZRFZRUURFQk5KUWswZ1EyeHZkV1FnUkdGMFlXSmhjMlZ6TUlJQklqQU5CZ2txCmhraUc5dzBCQVFFRkFBT0NBUThBTUlJQkNnS0NBUUVBOGxwYVFHem
NGZEdxZU1sbXFqZmZNUHBJUWhxcGQ4cUoKUHIzYklrclhKYlRjSko5dUlja1NVY0NqdzRaL3JTZzhublQxM1NDY09sKzF0bys3a2RNaVU4cU9XS2ljZVlaNQp5K3laW
WZDa0dhaVpWZmF6UUJtNDV6QnRGV3YrQUIvOGhmQ1RkTkY3Vlk0c3BhQTNvQkUyYVM3T0FOTlNSWlNLCnB3eTI0SVVnVWNJTEpXK21jdlc4MFZ4K0dYUmZEOVl0dDZQ
UkpnQmhZdVVCcGd6dm5nbUNNR0JuK2wyS05pU2YKd2VvdllEQ0Q2Vm5nbDIrNlc5UUZBRnRXWFdnRjNpRFFENW5sL240bXJpcE1TWDZVRy9uNjY1N3U3VERkZ2t2QQo
xZUtJMkZMellLcG9LQmU1cmNuck03bkhnTmMvbkNkRXM1SmVjSGIxZEh2MVFmUG02cHpJeHdJREFRQUJvMUF3ClRqQWRCZ05WSFE0RUZnUVVLMytYWm8xd3lLcytERW
9ZWGJIcnV3U3BYamd3SHdZRFZSMGpCQmd3Rm9BVUszK1gKWm8xd3lLcytERW9ZWGJIcnV3U3BYamd3REFZRFZSMFRCQVV3QXdFQi96QU5CZ2txaGtpRzl3MEJBUXNGQ
```

As we can see new credential created



Databases for PostgreSQL-ts Active Add tags

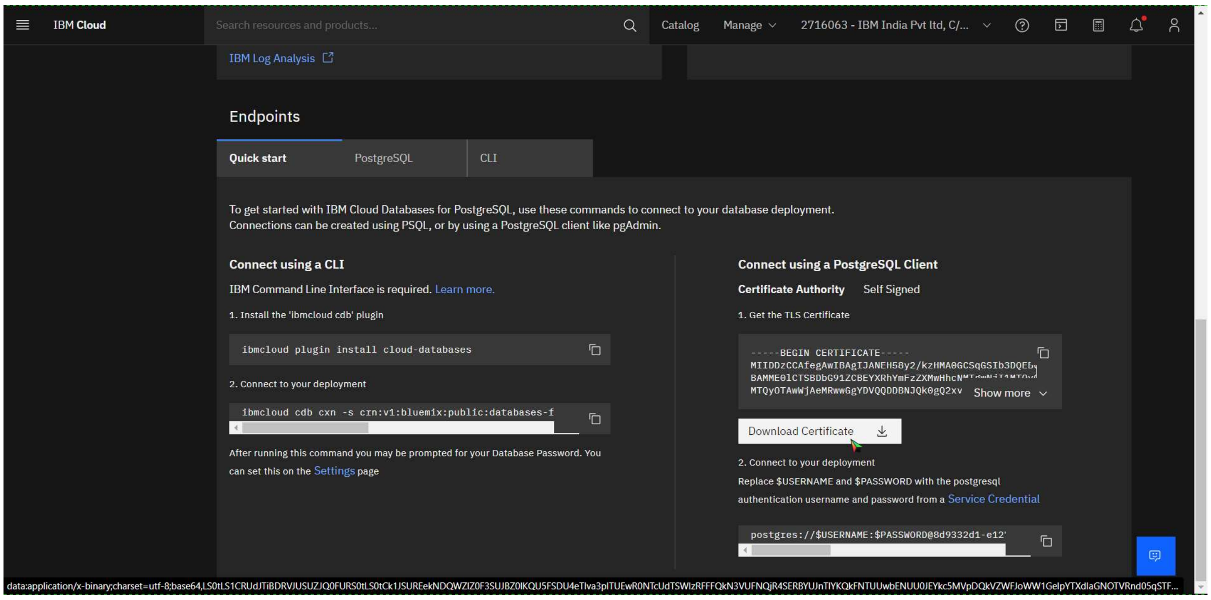
Service credentials

You can generate a new set of credentials for cases where you want to manually connect an app or external consumer to an IBM Cloud service. Learn more

Key name	Date created
Service credentials-tushar	2024-04-29 6:08 PM

```
{
  "connection": {
    "cli": {
      "arguments": [
        [
          "host=8d9332d1-e127-473b-ab03-7e6a4dda9ec3.d7deeff0d58745aba57fa5c84685d5b4.databases.appdomain.cloud port=31118 db
name=ibmclouddb user=ibm_cloud_4d37e129_01c4_4d10_a157_d2d88786c36e sslmode=verify-full"
        ]
      ],
      "bin": "psql",
      "certificate": {
        "certificate_authority": "self_signed",
        "certificate_base64": "LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1JSUREekNDQWZlZ0F3SUJBZ0lKQU5FSDU4eTIva3pITUEwR0NTcUdTSW
IzRFFFQkN3VUFNQjR4SERBYUJnTlYKQkFNTUUwbENUU0JEYkc5MVpDQkVZWFJoWW1GelpYTXdIaGNOTVRnd05qSTFNVFF5T1RBd1doY05Namd3TmpJeQpNVFF5T1R
Bd1dqQWVNUnd3R2dZRFZRUURQk5KUWswZ1EyeHZkV1FnUkdGMFlXSmhjMlZ6TUlJQklqQU5CZ2txCmhraUc5dzBCQVFFRkFBT0NBUThBTUlJQkNnS0NBUUVBOGxw
YVFHemNGZEdxZU1sbXFqZmZNUHBJUWhxcGQ4cUoKUHIzYklrclhKY1RjSko5dUlja1NVY0NqdzRaL3JTZzhublQxM1NDY09sKzF0bys3a2RNaVU4cU9XS2ljZVlaN
Qp5K3laWWZDa0dhaVpWZmF6UUJtNDV6QnRGV3YrQUIvOGhmQ1RkTkY3Vlk0c3BhQTNvQkUyYVM3T0FOTlNSWlNLCnB3eTI0SVVnVWNJTEpXK21jdlc4MFZ4K0dYUm
ZEOVl0dDZQUkpnQmhZdVVCcGd6dm5nbUNNR0JuK2wyS05pU2YKd2VvdllEQ0Q2Vm5nbDIrNlc5UUZBRnRXWFdnRjNpRFFENW5sL240bXJpcE1TWDZVRy9uNjY1N3U
3VERkZ2t2QQoxZUtJMkZMellLcG9LQmU1cmNuck03bkhnTmMvbkNkRXM1SmVjSGIxZEh2MVFmUG02cHpJeHdJREFRQUJvMUF3ClRqQWRCZ05WSFE0RUZnUVVLMytY
Wm8xd3lLcytERW9ZWGJIcnV3U3BYamd3SHdZRFZSMGpCQmd3Rm9BVUszK1gKWm8xd3lLcytERW9ZWGJIcnV3U3BYamd3REFZRFZSMFRCQVV3QXdFQi96QU5CZ2txa
```

Now scroll down and download certificate



Endpoints

To get started with IBM Cloud Databases for PostgreSQL, use these commands to connect to your database deployment. Connections can be created using PSQL, or by using a PostgreSQL client like pgAdmin.

Connect using a CLI

IBM Command Line Interface is required. Learn more.

1. Install the 'ibmcloud cdb' plugin

```
ibmcloud plugin install cloud-databases
```

2. Connect to your deployment

```
ibmcloud cdb cxn -s crn:v1:bluemix:public:databases-f
```

After running this command you may be prompted for your Database Password. You can set this on the Settings page

Connect using a PostgreSQL Client

Certificate Authority Self Signed

1. Get the TLS Certificate

```
-----BEGIN CERTIFICATE-----
MIIDDzCCAfegAwIBAgIJANEH58y2/kzHMA0GCSqGSIb3DQEB
BAMME0lCTSBDbG91ZCBEYXRhYmFzZXMwHhcN
MTQyOTAwWjAeMRwwGgYDVQQDDBNJQk0gQ2xv
```

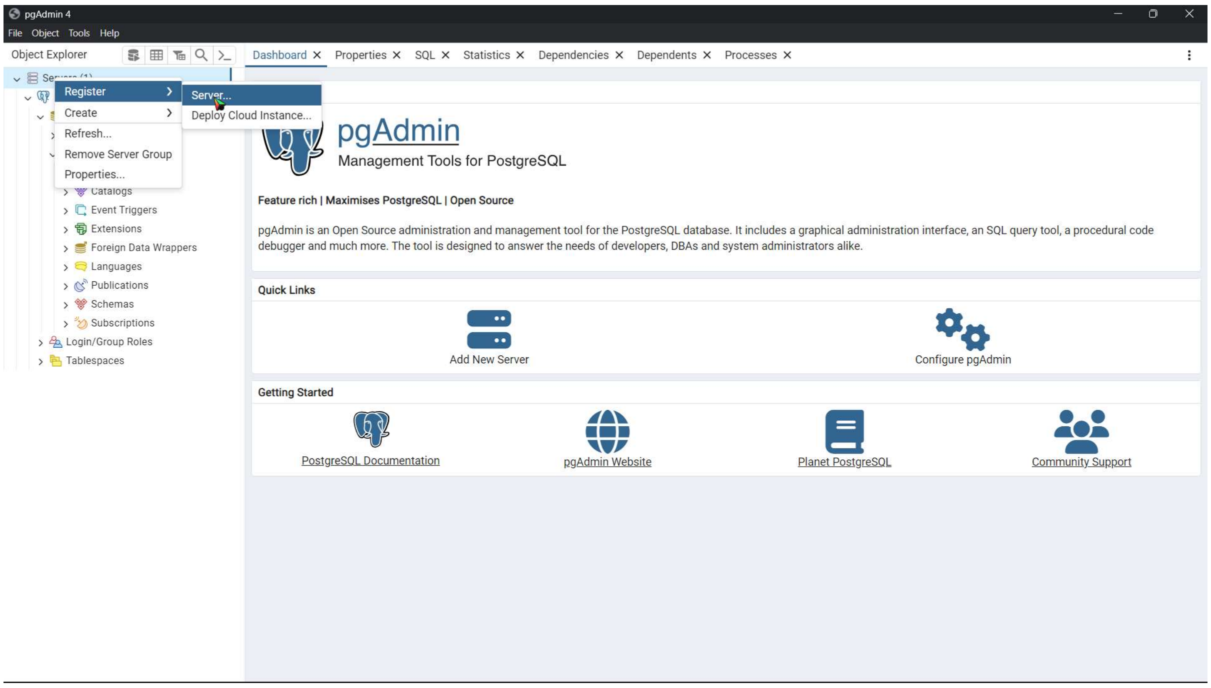
2. Connect to your deployment

Replace $USERNAME and $PASSWORD with the postgresql authentication username and password from a Service Credential

```
postgres://$USERNAME:$PASSWORD@8d9332d1-e12
```

data:application/x-binary;charset=utf-8;base64,LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1JSUREekNDQWZlZ0F3SUJBZ0lKQU5FSDU4eTIva3pITUEwR0NTcUdTSWIzRFFFQkN3VUFNQjR4SERBYUJnTlYKQkFNTUUwbENUU0JEYkc5MVpDQkVZWFJoWW1GelpYTXdIaGNOTVRnd05qSTF...

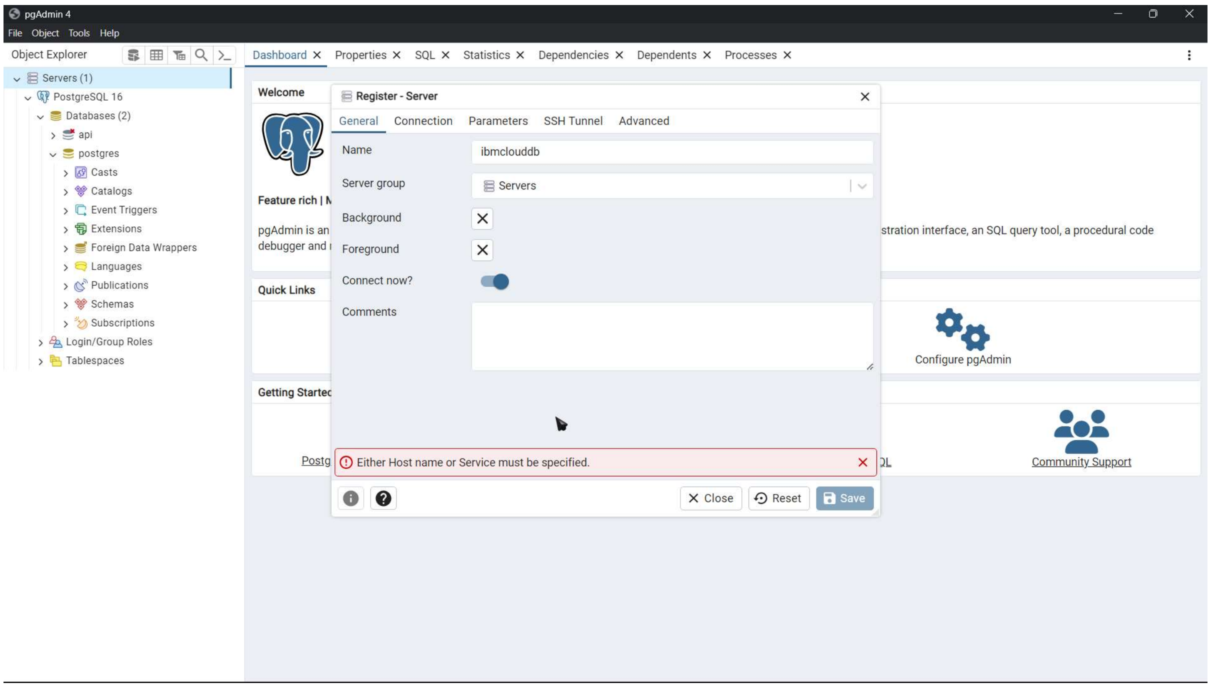
Open pgAdmin and create new server



Feature rich | Maximises PostgreSQL | Open Source

pgAdmin is an Open Source administration and management tool for the PostgreSQL database. It includes a graphical administration interface, an SQL query tool, a procedural code debugger and much more. The tool is designed to answer the needs of developers, DBAs and system administrators alike.

Give name ibmclouddb than go to connection

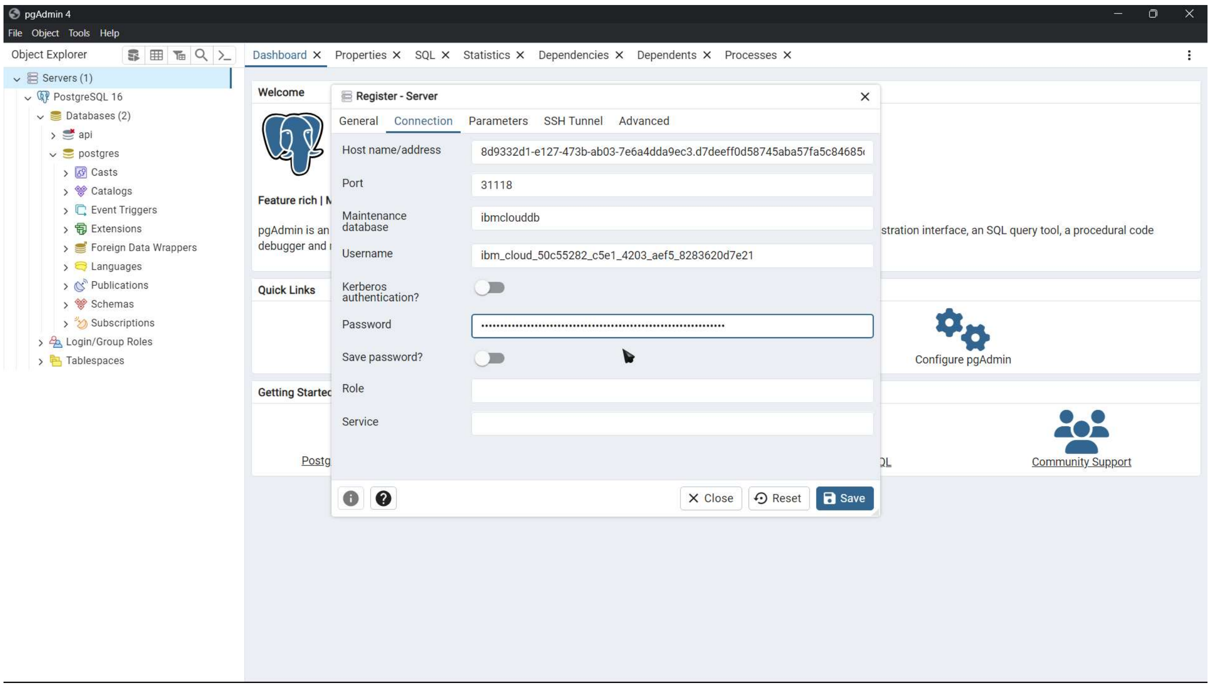


Name	ibmclouddb
Server group	Servers
Background	
Foreground	
Connect now?	on
Comments	

Either Host name or Service must be specified.

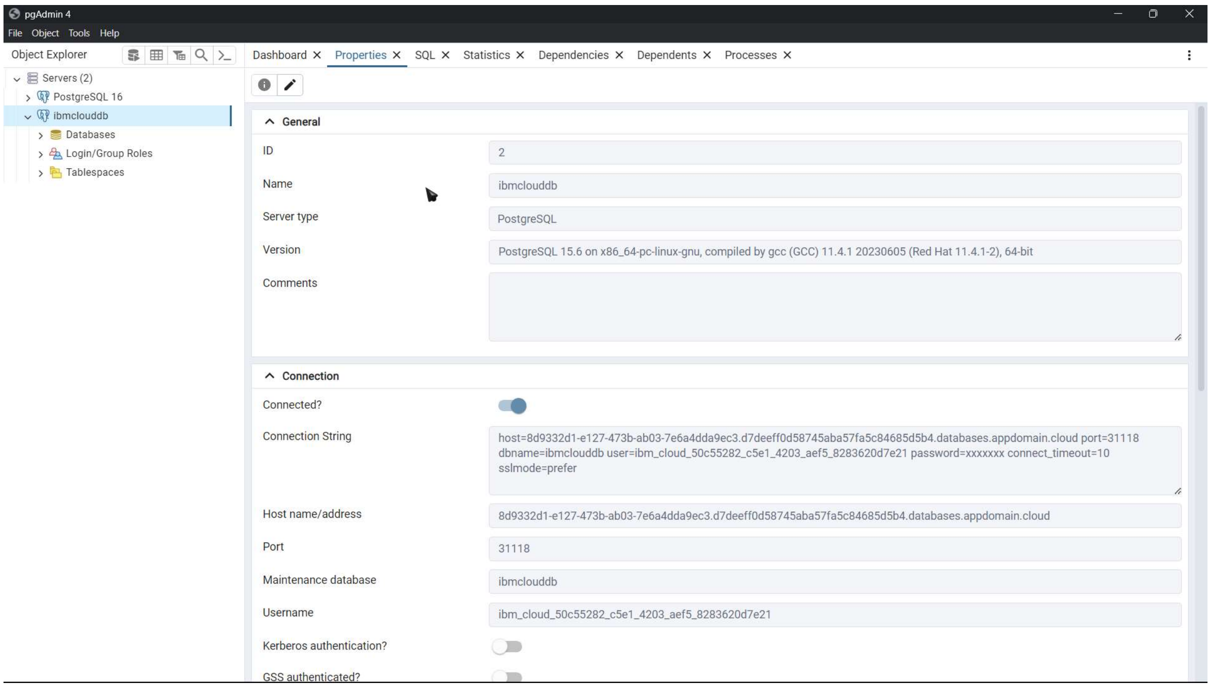
Add this on connection part than click on save

```
user: 'ibm_cloud_50c55282_c5e1_4203_aef5_8283620d7e21',
  host:'8d9332d1-e127-473b-ab03-
7e6a4dda9ec3.d7deeff0d58745aba57fa5c84685d5b4.databases.appdomain.cloud
',
  database: 'ibmclouddb',
  password:'9b88569fe0efc13c2833745be5183986f3b2cf18d73090c7e8e91620750
f80d5',
  port: 31118,
```



Host name/address	8d9332d1-e127-473b-ab03-7e6a4dda9ec3.d7deeff0d58745aba57fa5c84685
Port	31118
Maintenance database	ibmclouddb
Username	ibm_cloud_50c55282_c5e1_4203_aef5_8283620d7e21
Kerberos authentication?	
Password	••
Save password?	
Role	
Service	

As we can see ibmclouddb is created



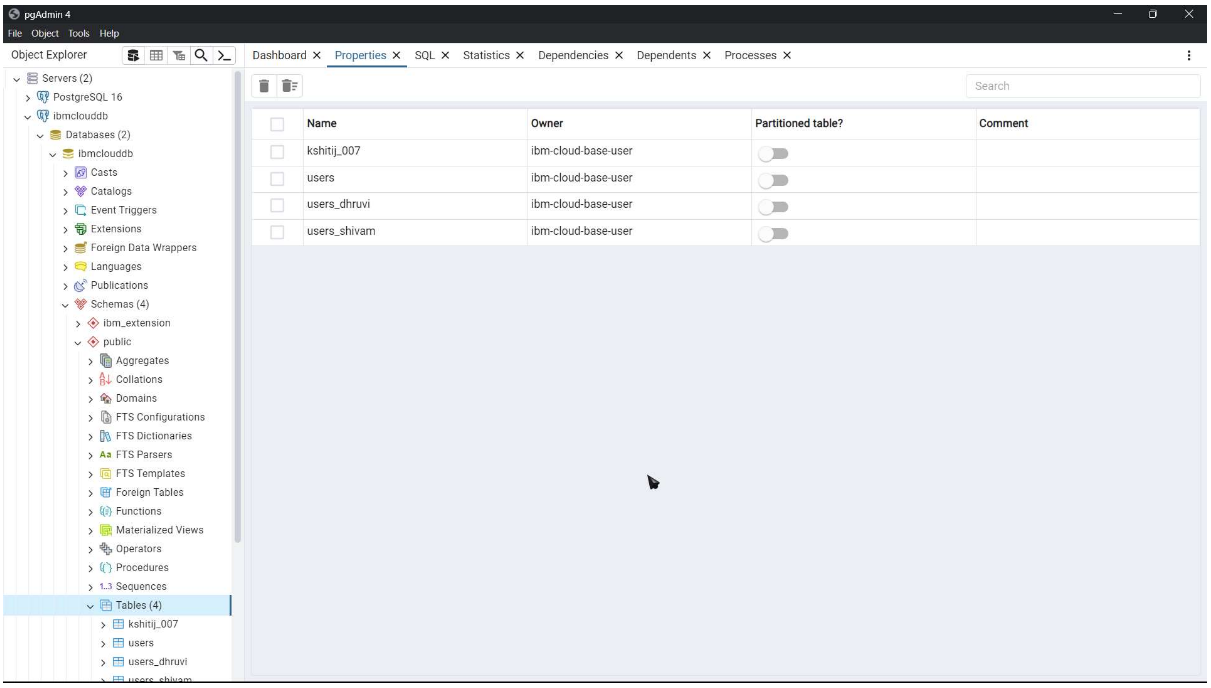
General

ID	2
Name	ibmclouddb
Server type	PostgreSQL
Version	PostgreSQL 15.6 on x86_64-pc-linux-gnu, compiled by gcc (GCC) 11.4.1 20230605 (Red Hat 11.4.1-2), 64-bit
Comments	

Connection

Connected?	●
Connection String	host=8d9332d1-e127-473b-ab03-7e6a4dda9ec3.d7deeff0d58745aba57fa5c84685d5b4.databases.appdomain.cloud port=31118 dbname=ibmclouddb user=ibm_cloud_50c55282_c5e1_4203_aef5_8283620d7e21 password=xxxxxxx connect_timeout=10 sslmode=prefer
Host name/address	8d9332d1-e127-473b-ab03-7e6a4dda9ec3.d7deeff0d58745aba57fa5c84685d5b4.databases.appdomain.cloud
Port	31118
Maintenance database	ibmclouddb
Username	ibm_cloud_50c55282_c5e1_4203_aef5_8283620d7e21
Kerberos authentication?	
GSS authenticated?	

In ibmclouddb we can see users Tables

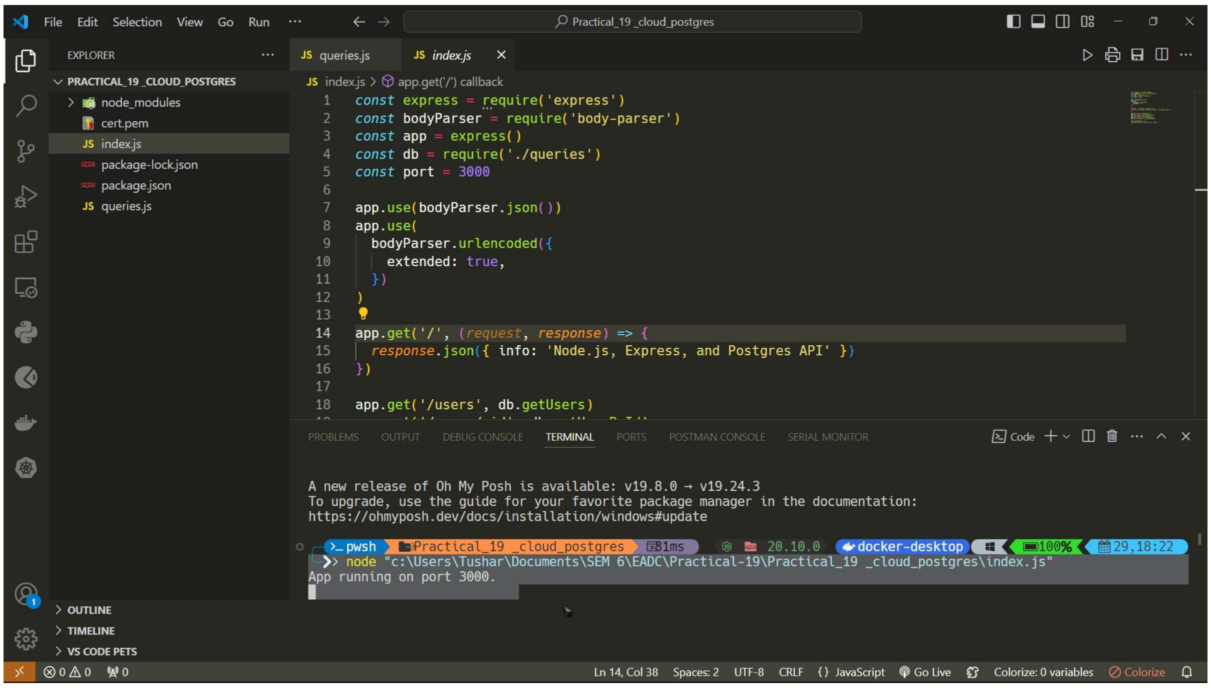


Name	Owner	Partitioned table?	Comment
kshitij_007	ibm-cloud-base-user		
users	ibm-cloud-base-user		
users_dhruvi	ibm-cloud-base-user		
users_shivam	ibm-cloud-base-user		

Now change all data from Database for postgreSQL-ts

```
const Pool = require('pg').Pool
var fs = require('fs');
const pool = new Pool({
  user: 'ibm_cloud_50c55282_c5e1_4203_aef5_8283620d7e21',
  host:'8d9332d1-e127-473b-ab03-
7e6a4dda9ec3.d7deeff0d58745aba57fa5c84685d5b4.databases.appdomain.cloud
',
  database: 'ibmclouddb',
  password:'9b88569fe0efc13c2833745be5183986f3b2cf18d73090c7e8e91620750
f80d5',
  port: 31118,

  ssl: {
    rejectUnauthorized: false,
    cert: fs.readFileSync('./cert.pem').toString(),

}
})
```

After the changes run the code Code is running on port 3000

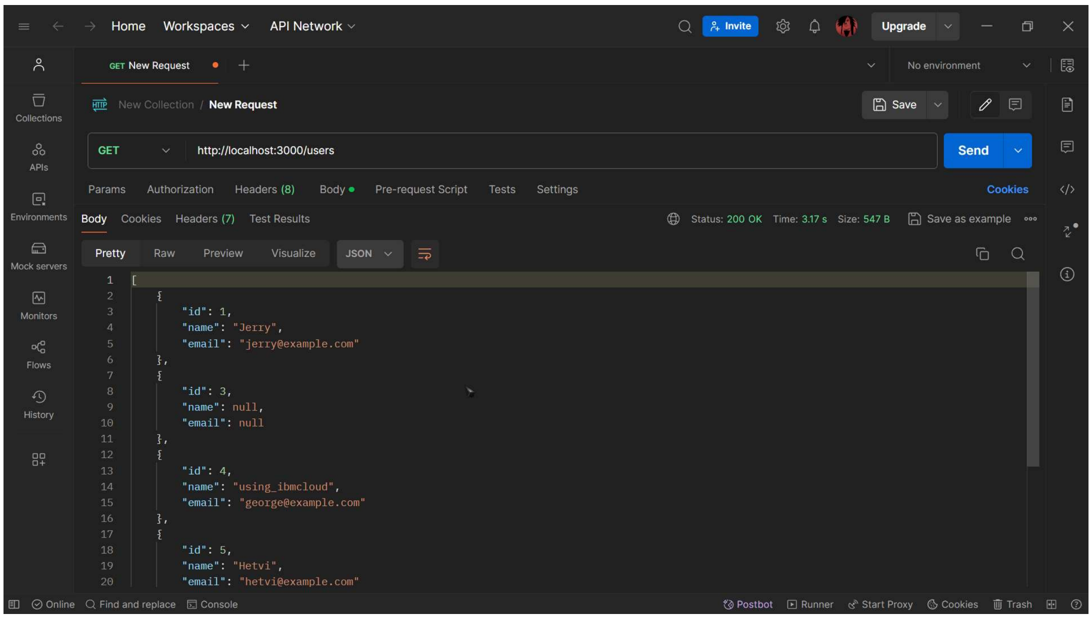


```
const express = require('express')
const bodyParser = require('body-parser')
const app = express()
const db = require('./queries')
const port = 3000

app.use(bodyParser.json())
app.use(
  bodyParser.urlencoded({
    extended: true,
  })
)

app.get('/', (request, response) => {
  response.json({ info: 'Node.js, Express, and Postgres API' })
})

app.get('/users', db.getUsers)
```

```
A new release of Oh My Posh is available: v19.8.0 → v19.24.3
To upgrade, use the guide for your favorite package manager in the documentation:
https://ohmyposh.dev/docs/installation/windows#update

pwsh  Practical_19 _cloud_postgres  81ms   20.10.0  docker-desktop  100%  29,18:22
>> node "c:\Users\Tushar\Documents\SEM 6\EADC\Practical-19\Practical_19 _cloud_postgres\index.js"
App running on port 3000.
```

a) GET all users

Endpoint:- http://localhost:3000/users



GET http://localhost:3000/users

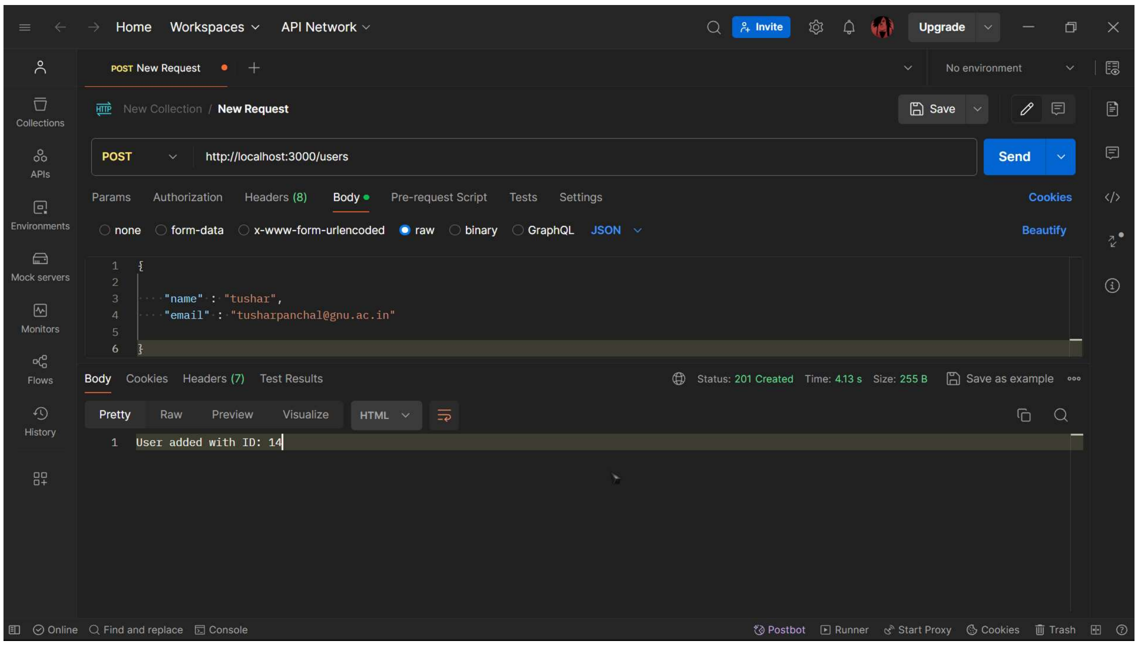
Status: 200 OK Time: 3.17 s Size: 547 B

```
[
    {
        "id": 1,
        "name": "Jerry",
        "email": "jerry@example.com"
    },
    {
        "id": 3,
        "name": null,
        "email": null
    },
    {
        "id": 4,
        "name": "using_ibmcloud",
        "email": "george@example.com"
    },
    {
        "id": 5,
        "name": "Hetvi",
        "email": "hetvi@example.com"
```

b) POST / add a new user

Endpoint:- http://localhost:3000/users

```
{
    "name" : "tushar",
    "email" : "tusharpanchal@gnu.ac.in"
}
```



POST http://localhost:3000/users

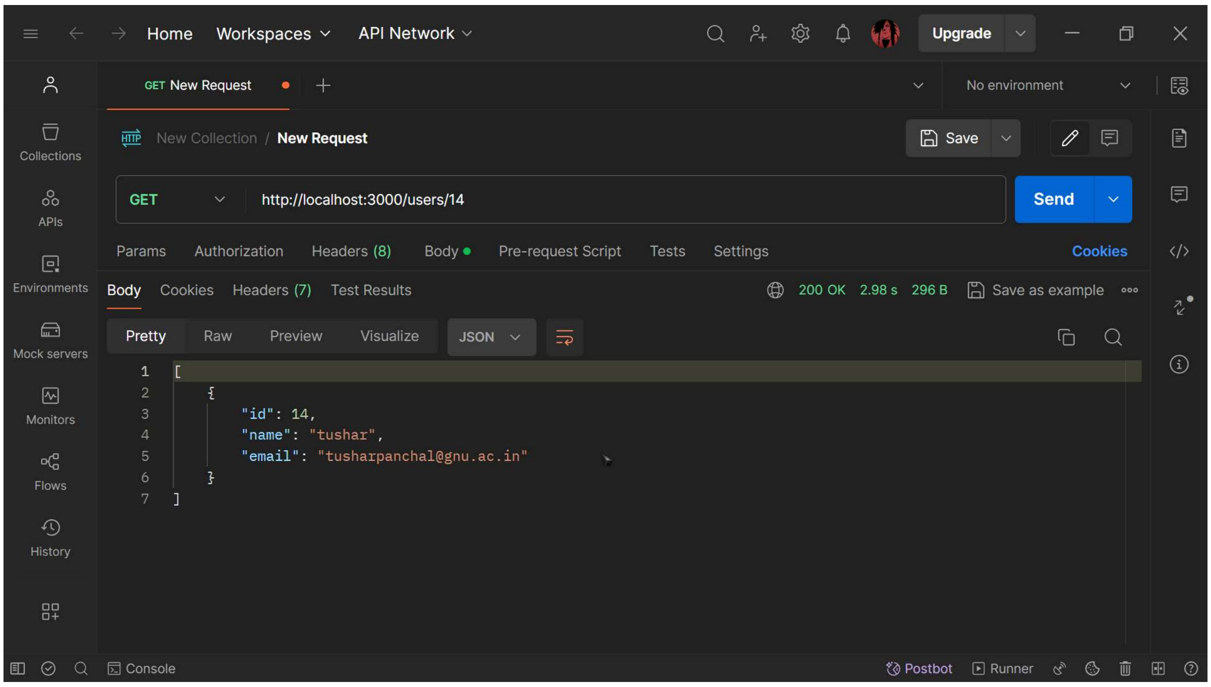
```
{

    "name" : "tushar",
    "email" : "tusharpanchal@gnu.ac.in"

}
```

Status: 201 Created Time: 4.13 s Size: 255 B

```
User added with ID: 14
```

c) Get details of user using ID

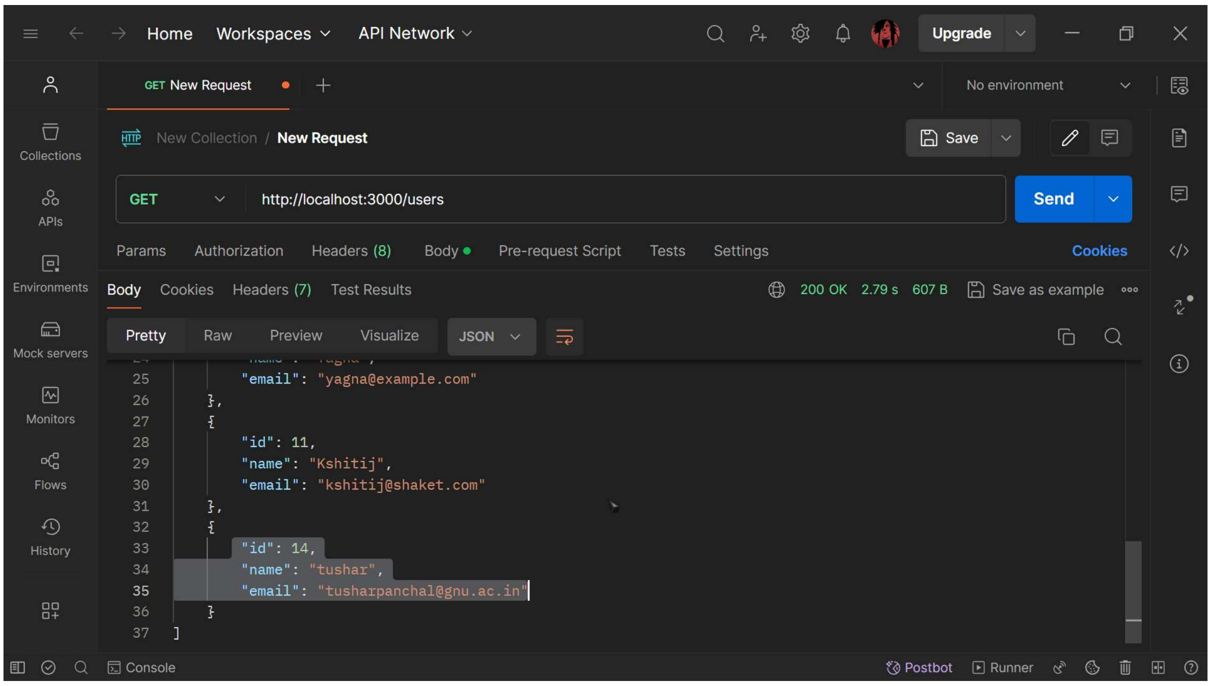
Endpoint:- http://localhost:3000/users/14



GET http://localhost:3000/users/14

```
[
    {
        "id": 14,
        "name": "tushar",
        "email": "tusharpanchal@gnu.ac.in"
    }
]
```

We can see my data at ID = 14

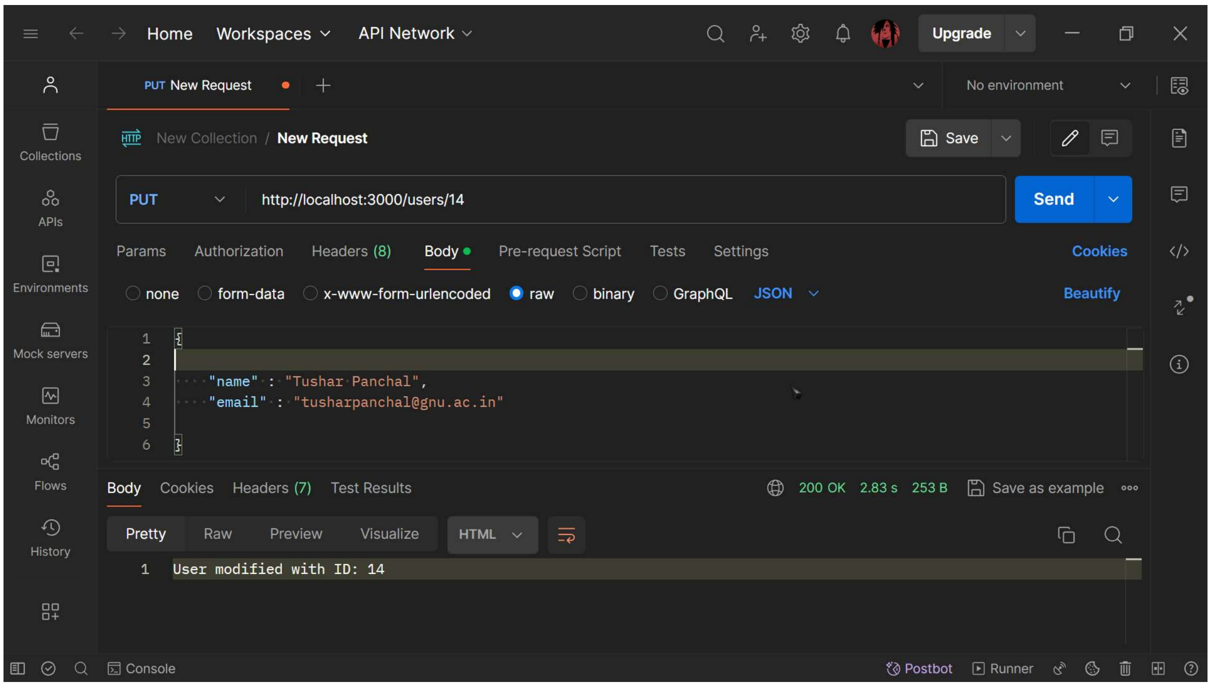


GET http://localhost:3000/users

```
        "email": "yagna@example.com"
    },
    {
        "id": 11,
        "name": "Kshitij",
        "email": "kshitij@shaket.com"
    },
    {
        "id": 14,
        "name": "tushar",
        "email": "tusharpanchal@gnu.ac.in"
    }
]
```

d) Put / Update an existing User

Endpoint:- ***http://localhost:3000/users/14***



PUT http://localhost:3000/users/14

Params · Authorization · Headers (8) · Body · Pre-request Script · Tests · Settings

none · form-data · x-www-form-urlencoded · raw · binary · GraphQL · JSON

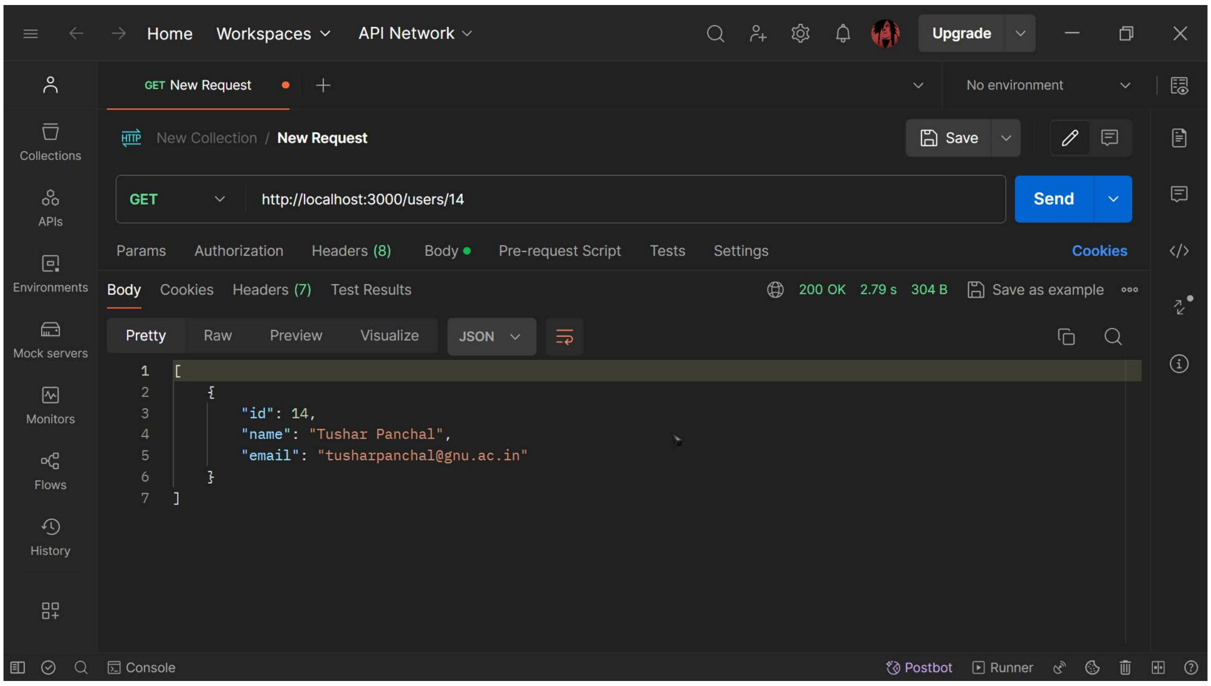
```
{

    "name" : "Tushar Panchal",
    "email" : "tusharpanchal@gnu.ac.in"

}
```

Body · Cookies · Headers (7) · Test Results — 200 OK 2.83 s 253 B · Save as example

Pretty · Raw · Preview · Visualize · HTML

```
User modified with ID: 14
```

As we can see in below screenshot My name is modified



GET http://localhost:3000/users/14

Params · Authorization · Headers (8) · Body · Pre-request Script · Tests · Settings

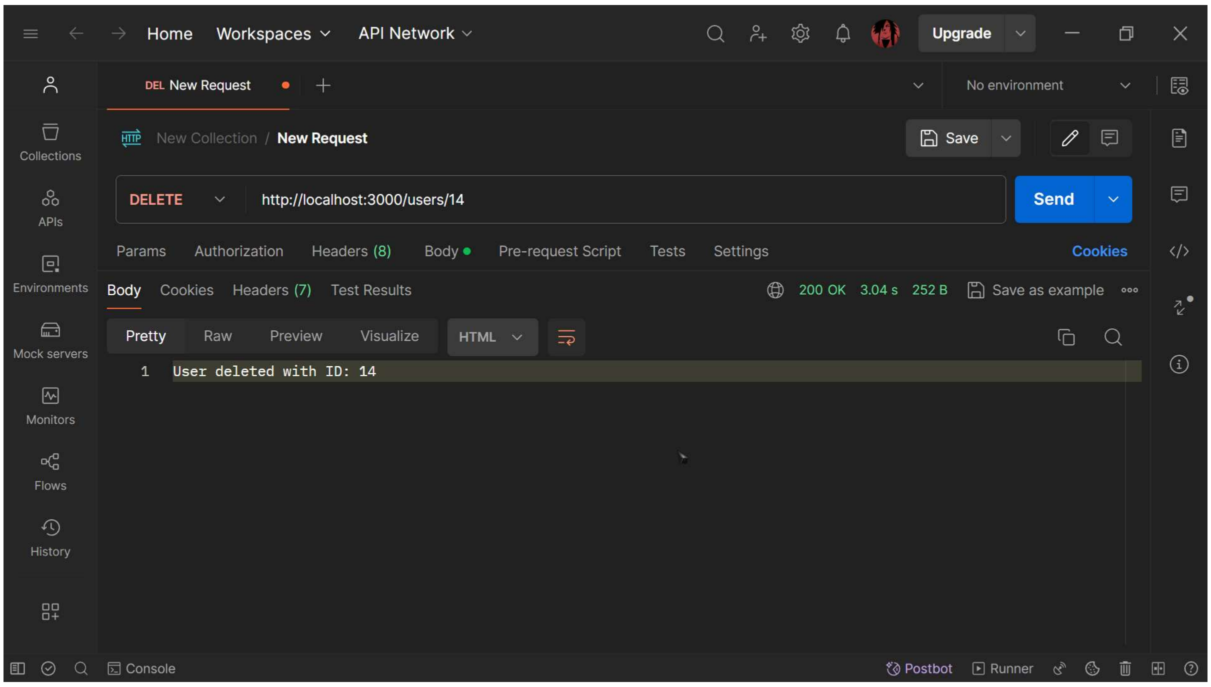
Body · Cookies · Headers (7) · Test Results — 200 OK 2.79 s 304 B · Save as example

Pretty · Raw · Preview · Visualize · JSON

```
[
    {
        "id": 14,
        "name": "Tushar Panchal",
        "email": "tusharpanchal@gnu.ac.in"
    }
]
```

e) Delete an existing User

Endpoint:- http://localhost:3000/users/14

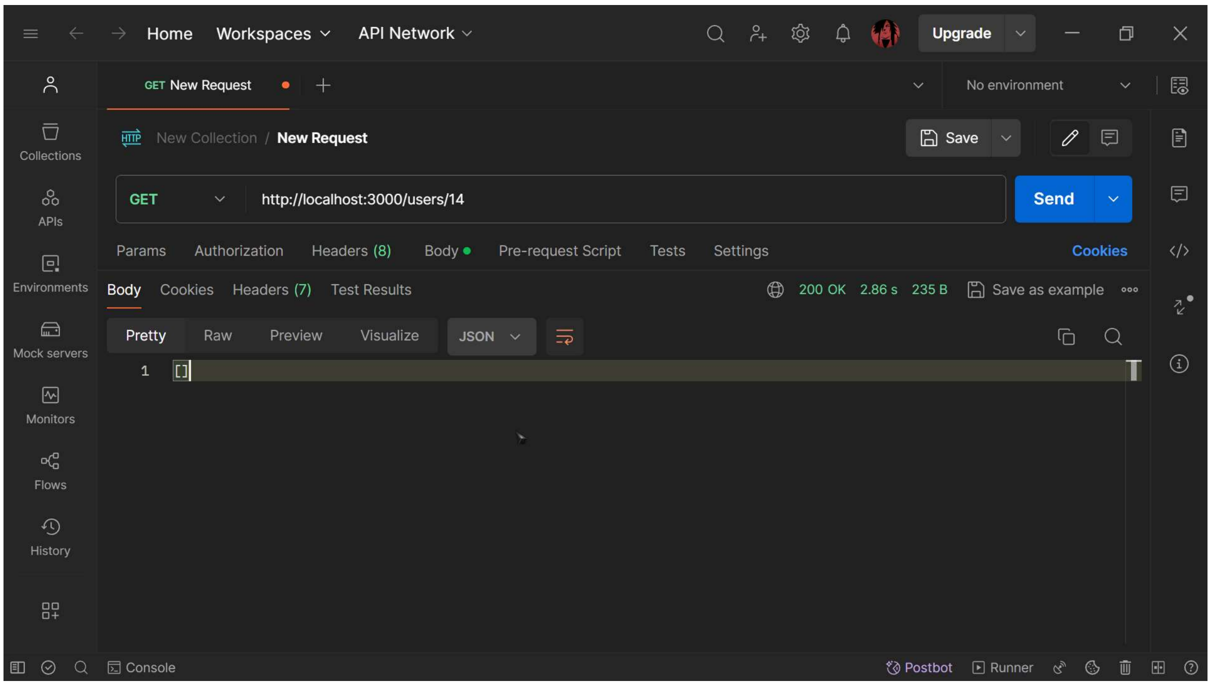


DELETE http://localhost:3000/users/14

Body Cookies Headers (7) Test Results 200 OK 3.04 s 252 B

```
User deleted with ID: 14
```

As we can see my data on no.14 ID is deleted



GET http://localhost:3000/users/14

Body Cookies Headers (7) Test Results 200 OK 2.86 s 235 B

```
[]
```