

7. Build a REST API to Create (POST) and Fetch All Articles (GET) using Express • Create a POST endpoint/article to insert new article data into PostgreSQL. • Create a GET endpoint /articles to fetch all articles from PostgreSQL. • Test both APIs using Postman. • Output: Successfully inserted article should return a JSON object, and GET should return all articles.

Server.js

```
// app.js

const express = require('express');
const bodyParser = require('body-parser');
const { Client } = require('pg');

const app = express();
const port = 3000;

// Middleware to parse JSON bodies
app.use(bodyParser.json());

// Set up PostgreSQL client
const client = new Client({
  user: 'postgres', // Replace with your PostgreSQL username
  host: 'localhost',
  database: 'articles_db', // Replace with your PostgreSQL database name
  password: 'postgres', // Replace with your PostgreSQL password
  port: 5432,
});

// Connect to PostgreSQL
client.connect();
```

```
// POST endpoint to insert a new article
app.post('/article', async (req, res) => {
  const { title, content } = req.body;

  if (!title || !content) {
    return res.status(400).json({ error: 'Title and content are required' });
  }

  try {
    const result = await client.query(
      'INSERT INTO articles (title, content) VALUES ($1, $2) RETURNING *',
      [title, content]
    );
    const newArticle = result.rows[0];
    res.status(201).json(newArticle); // Return the inserted article as JSON
  } catch (err) {
    console.error('Error inserting article:', err);
    res.status(500).json({ error: 'An error occurred while inserting the article' });
  }
});
```

```
// GET endpoint to fetch all articles
app.get('/articles', async (req, res) => {
  try {
    const result = await client.query('SELECT * FROM articles ORDER BY created_at DESC');
    const articles = result.rows;
    res.status(200).json(articles); // Return the articles as JSON
  } catch (err) {
```

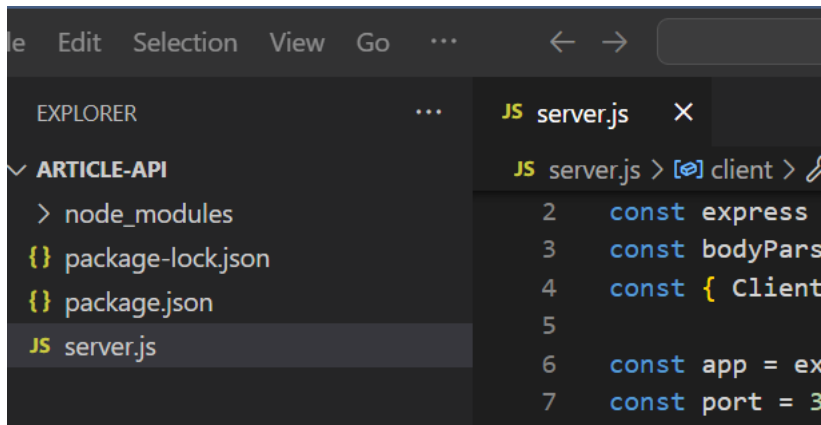
```
    console.error('Error fetching articles:', err);

    res.status(500).json({ error: 'An error occurred while fetching articles' });
  }
});

// Start the server
app.listen(port, () => {
  console.log(`Server is running on http://localhost:${port}`);
});
```

```
\article-api>npm init -y
```

```
o\article-api>npm install express pg body-parser
```



Psq shell:

```
SQL Shell (psql)
Username [postgres]: postgres
Password for user postgres:

psql (17.4)
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 articles_db;
CREATE DATABASE
postgres=# \c articles_db;
You are now connected to database "articles_db" as user "postgres".
articles_db=# create table articles( id serial primary key, title varchar(255) not null, content text not null, created_at timestamp
default current_timestamp);
CREATE TABLE
articles_db=# \dt
      List of relations
Schema | Name   | Type  | Owner
-----+-----+-----+-----
public | articles | table | postgres
(1 row)

articles_db=#
```

OUTPUT:

```
(C) Microsoft Corporation. All rights reserved.

C:\Users\gayathri rao>cd article-api

C:\Users\gayathri rao\article-api>node server.js
Server is running on http://localhost:3000
```

Postman:

HTTP

http://localhost:3000/article

Save

POST

http://localhost:3000/article

Send

Params

Authorization

Headers (8)

Body

Pre-request Script

Tests

Settings

Cookies

none

form-data

x-www-form-urlencoded

raw

binary

JSON

Beautify

```
1 {
2   "title": "My First Article",
3   "content": "This is the content of the first article."
4 }
5
```

Body

Cookies

Headers (7)

Test Results

201 Created

62 ms

370 B

Save Response

Pretty

Raw

Preview

Visualize

JSON

```
1 {
2   "id": 1,
3   "title": "My First Article",
4   "content": "This is the content of the first article.",
5   "created_at": "2025-03-30T06:59:46.191Z"
6 }
```

POST http://localhost:3000/article

POST http://localhost:3000/article

GET http://localhost:3000/articles

+

...

HTTP

http://localhost:3000/articles

Save

GET

http://localhost:3000/articles

Send

Params

Authorization

Headers (6)

Body

Pre-request Script

Tests

Settings

Cookies

Query Params

	Key	Value	Bulk Edit
	Key	Value	

Body

Cookies

Headers (7)

Test Results

200 OK

13 ms

367 B

Save Response

Pretty

Raw

Preview

Visualize

JSON

```
1 [
2   {
3     "id": 1,
4     "title": "My First Article",
5     "content": "This is the content of the first article.",
6     "created_at": "2025-03-30T06:59:46.191Z"
7   }
8 ]
```