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

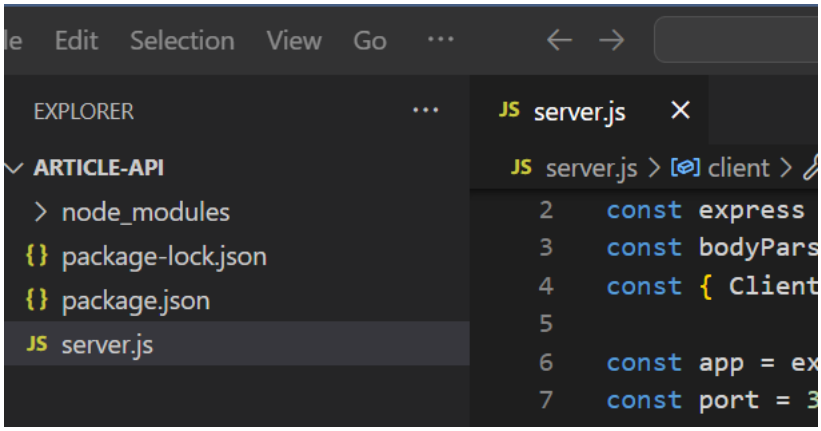
```
// app.js
const express = require('express');
const bodyParser = require('body-parser');
const { Client } = require('pg');

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) {
```

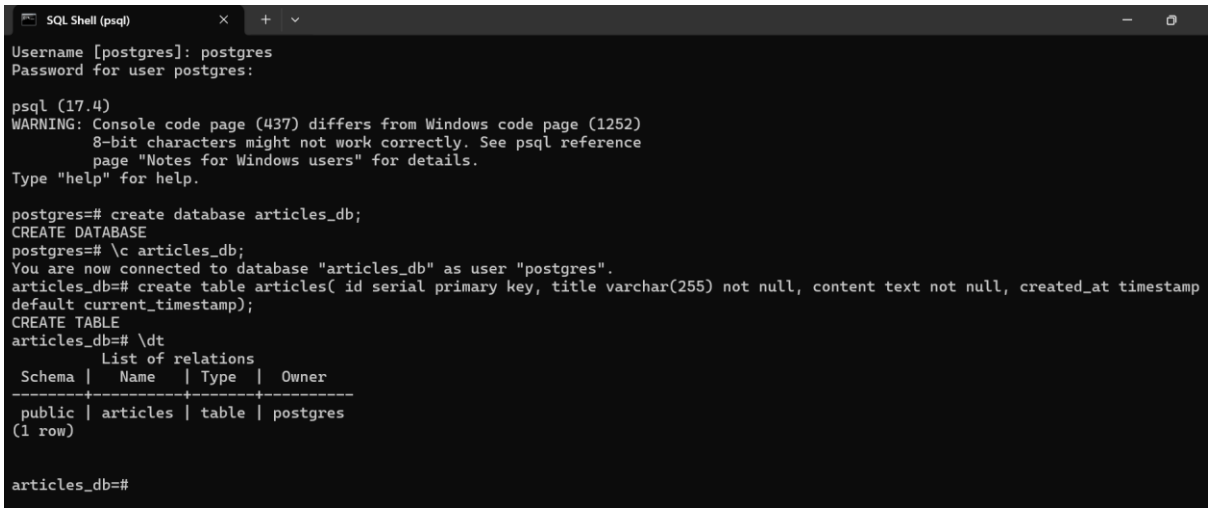
```
    console.error('Error fetching articles:', err);
    res.status(500).json({ error: 'An error occurred while fetching articles' });
  }
});

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

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

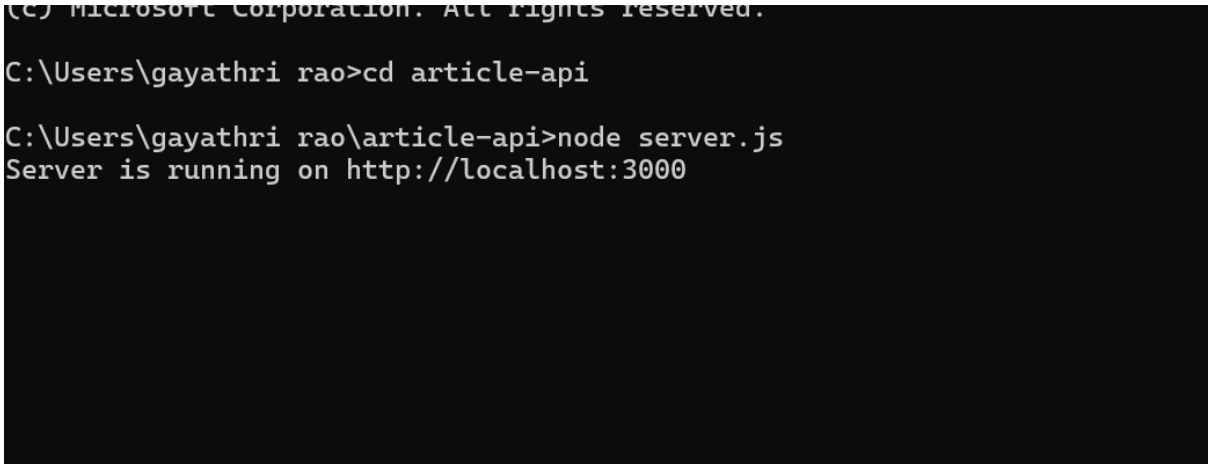
```
o\article-api>npm install express pg body-parser
```



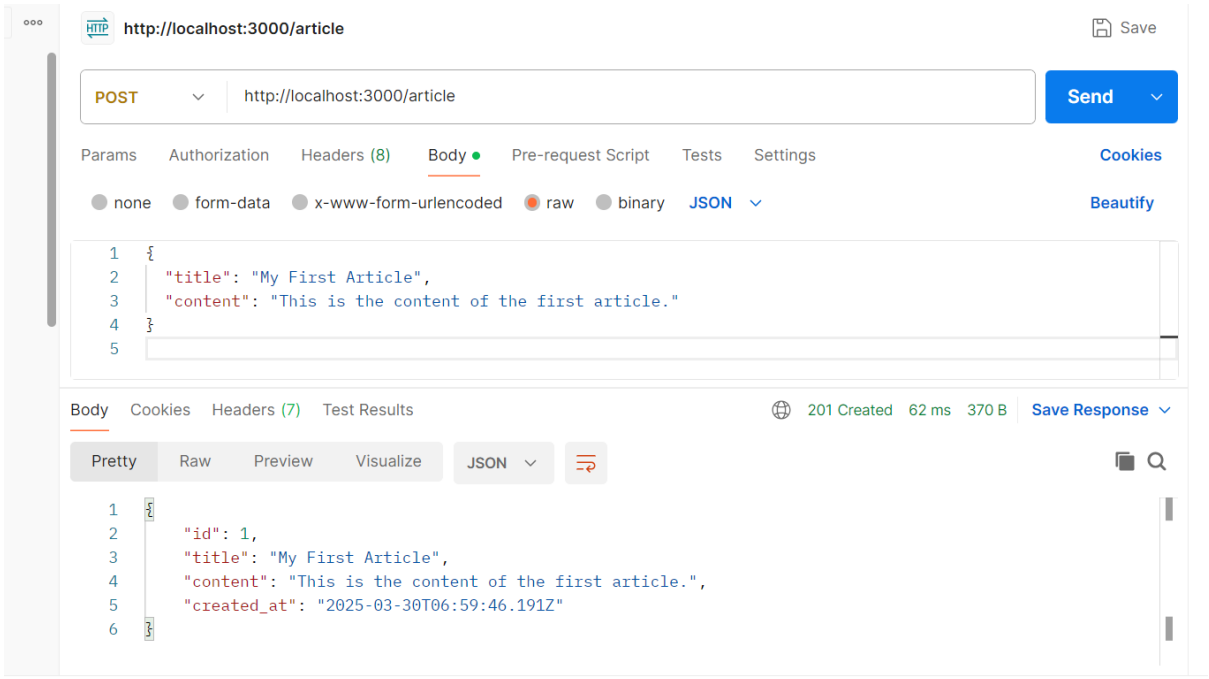
Psql shell:

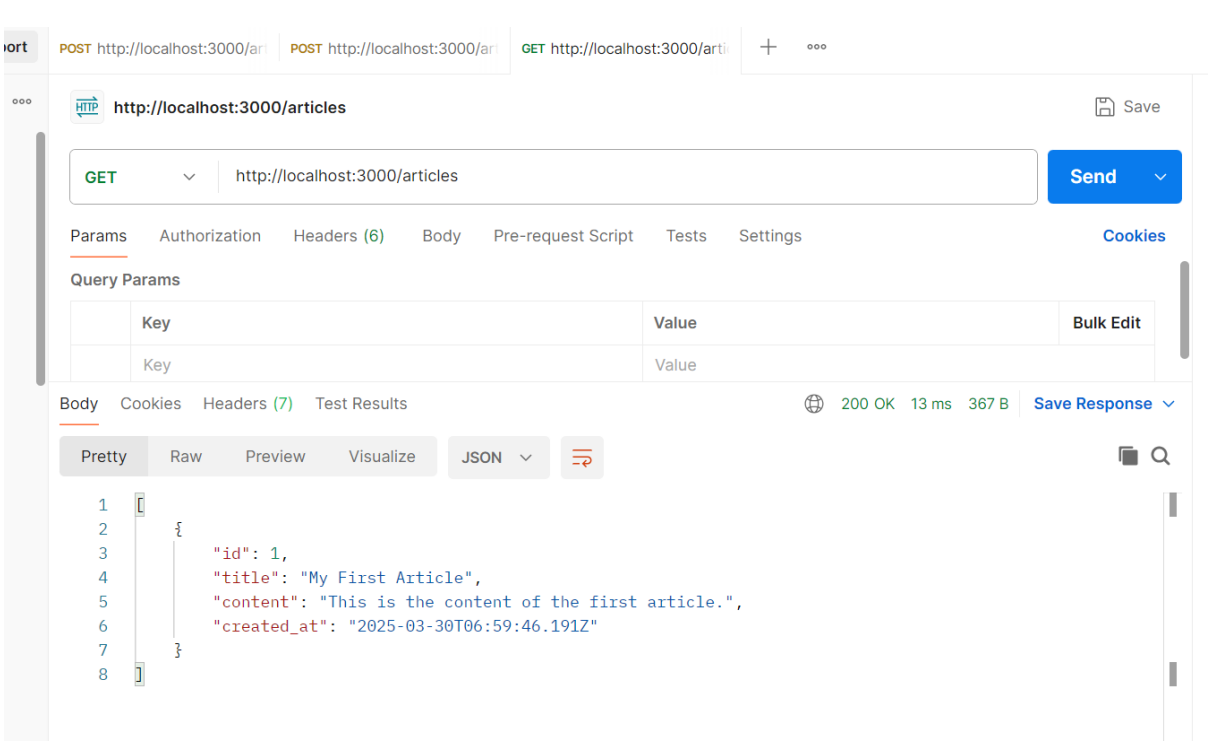


OUTPUT:



Postman:


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
{
"title": "My First Article",
"content": "This is the content of the first article."
}
Body
Cookies
Headers (7)
Test Results
201 Created
62 ms
370 B
Save Response
Pretty
Raw
Preview
Visualize
JSON
{
"id": 1,
"title": "My First Article",
"content": "This is the content of the first article.",
"created_at": "2025-03-30T06:59:46.191Z"
}


POST http://localhost:3000/ar
POST http://localhost:3000/ar
GET http://localhost:3000/arti
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
[
{
"id": 1,
"title": "My First Article",
"content": "This is the content of the first article.",
"created_at": "2025-03-30T06:59:46.191Z"
}
]