ENTIRE PROJECT SETUP

Part 1: SQL Challenge

Creating Tables:

Open MySQL Workbench.

Create a new database named "authors".

Inside the "authors" database, execute the following SQL commands to create three tables: "authors", "books", and "sale_items".

```
1.CREATE TABLE authors (

id serial PRIMARY KEY,

name text,

email text,

date_of_birth timestamp

);

2.CREATE TABLE books (

id serial PRIMARY KEY,

author_id integer REFERENCES authors (id),

isbn text,

);

3.CREATE TABLE sale_items (

id serial PRIMARY KEY,

book_id integer REFERENCES books (id),

customer_name text,

item_price money,

quantity integer

);
```

Inserting Data:

Once tables are created, insert random data into each table using SQL INSERT INTO statements.

```
INSERT INTO authors (id, name, email, date_of_birth) VALUES
(1, 'Lorelai Gilmore', 'lorelai@example.com', '1970-04-05'),
(2, 'Rory Gilmore', 'rory@example.com', '1984-10-08'),
(3, 'Luke Danes', 'luke@example.com', '1966-11-21'),
(4, 'Emily Gilmore', 'emily@example.com', '1944-02-12'),
(5, 'Richard Gilmore', 'richard@example.com', '1937-09-02'),
(6, 'Lane Kim', 'lane@example.com', '1981-12-29'),
(7, 'Paris Geller', 'paris@example.com', '1983-06-19'),
(8, 'Jess Mariano', 'jess@example.com', '1981-07-15'),
(9, 'Dean Forester', 'dean@example.com', '1980-03-14'),
(10, 'Sookie St. James', 'sookie@example.com', '1973-05-27'),
(11, 'Christopher Hayden', 'chris@example.com', '1968-08-17'),
(12, 'Liz Danes', 'liz@example.com', '1968-05-03'),
(13, 'Michel Gerard', 'michel@example.com', '1971-11-15'),
(14, 'Taylor Doose', 'taylor@example.com', '1950-01-30'),
(15, 'Kirk Gleason', 'kirk@example.com', '1974-09-27');


INSERT INTO books (id, author_id, isbn) VALUES
(1, 1, '978-1234567890'),
(2, 2, '978-2345678901'),
(3, 3, '978-3456789012'),
(4, 4, '978-4567890123'),
(5, 5, '978-5678901234'),
```

(6, 6, '978-6789012345'),

(7, 7, '978-7890123456'),

(8, 8, '978-8901234567'),

(9, 9, '978-9012345678'),

(10, 10, '978-0123456789'),

(11, 11, '978-1122334455'),

(12, 12, '978-2233445566'),

(13, 13, '978-3344556677'),

(14, 14, '978-4455667788'),

(15, 15, '978-5566778899'),

(16, 1, '978-1234567800'),

(17, 2, '978-2345678911'),

(18, 3, '978-3456789022'),

(19, 4, '978-4567890133'),

(20, 5, '978-5678901244'),

(21, 6, '978-6789012355'),

(22, 7, '978-7890123466'),

(23, 8, '978-8901234577'),

(24, 9, '978-9012345688'),

(25, 10, '978-0123456799'),

(26, 11, '978-1122334466'),

(27, 12, '978-2233445577'),

(28, 13, '978-3344556688'),

(29, 14, '978-4455667799'),

(30, 15, '978-5566778800'),

(31, 1, '978-1234567811'),

(32, 2, '978-2345678922'),

(33, 3, '978-3456789033'),

(34, 4, '978-4567890144'),

(35, 5, '978-5678901255');

INSERT INTO sale_items (id, book_id, customer_name, item_price, quantity) VALUES

(1, 1, 'Alice', 20, 1),

(2, 2, 'Bob', 25, 2),

(3, 3, 'Charlie', 30, 1),

(4, 4, 'David', 15, 3),

(5, 5, 'Emma', 40, 1),

(6, 6, 'Frank', 35, 2),

(7, 7, 'Grace', 20, 1),

(8, 8, 'Hannah', 25, 3),

(9, 9, 'Ian', 30, 1),

(10, 10, 'Julia', 15, 2),

(11, 11, 'Kevin', 40, 1),

(12, 12, 'Linda', 35, 2),

(13, 13, 'Mike', 20, 1),

(14, 14, 'Nancy', 25, 3),

(15, 15, 'Oscar', 30, 1),

(16, 16, 'Paula', 15, 2),

(17, 17, 'Quincy', 40, 1),

(18, 18, 'Rachel', 35, 2),

(19, 19, 'Steve', 20, 1),

(20, 20, 'Tina', 25, 3),

(21, 21, 'Ursula', 30, 1),

(22, 22, 'Victor', 15, 2),

(23, 23, 'Wendy', 40, 1),

(24, 24, 'Xavier', 35, 2),

(25, 25, 'Yara', 20, 1),

(26, 26, 'Zane', 25, 3),

(27, 27, 'Amy', 30, 1),

(28, 28, 'Ben', 15, 2),

(29, 29, 'Cara', 40, 1),

(30, 30, 'Dylan', 35, 2),

(31, 31, 'Ella', 20, 1),

(32, 32, 'Felix', 25, 3),

(33, 33, 'Gina', 30, 1),

(34, 34, 'Henry', 15, 2),

(35, 35, 'Ivy', 40, 1),

(36, 1, 'Jack', 35, 2),

(37, 2, 'Karen', 20, 1),

(38, 3, 'Leo', 25, 3),

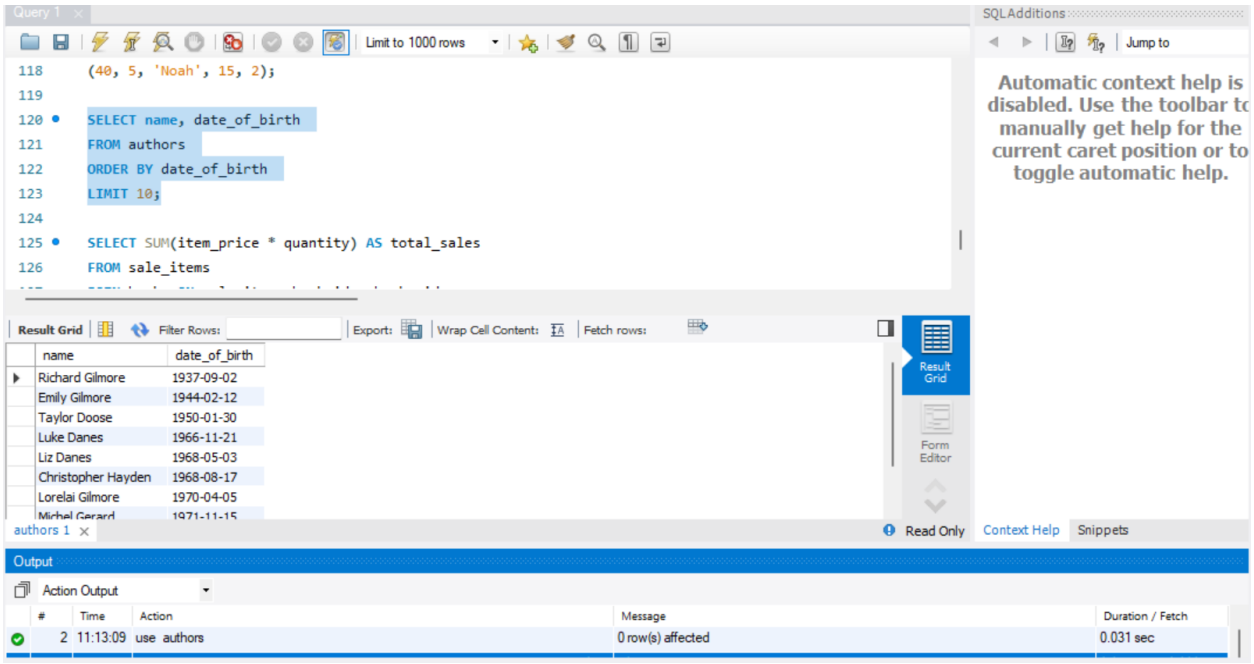
(39, 4, 'Mia', 30, 1),

(40, 5, 'Noah', 15, 2);

Running Queries:

1.Who are the first 10 authors ordered by date_of_birth?

```
SELECT name, date_of_birth
FROM authors
ORDER BY date_of_birth
LIMIT 10;
```



2. What is the sales total for the author named “Lorelai Gilmore”?

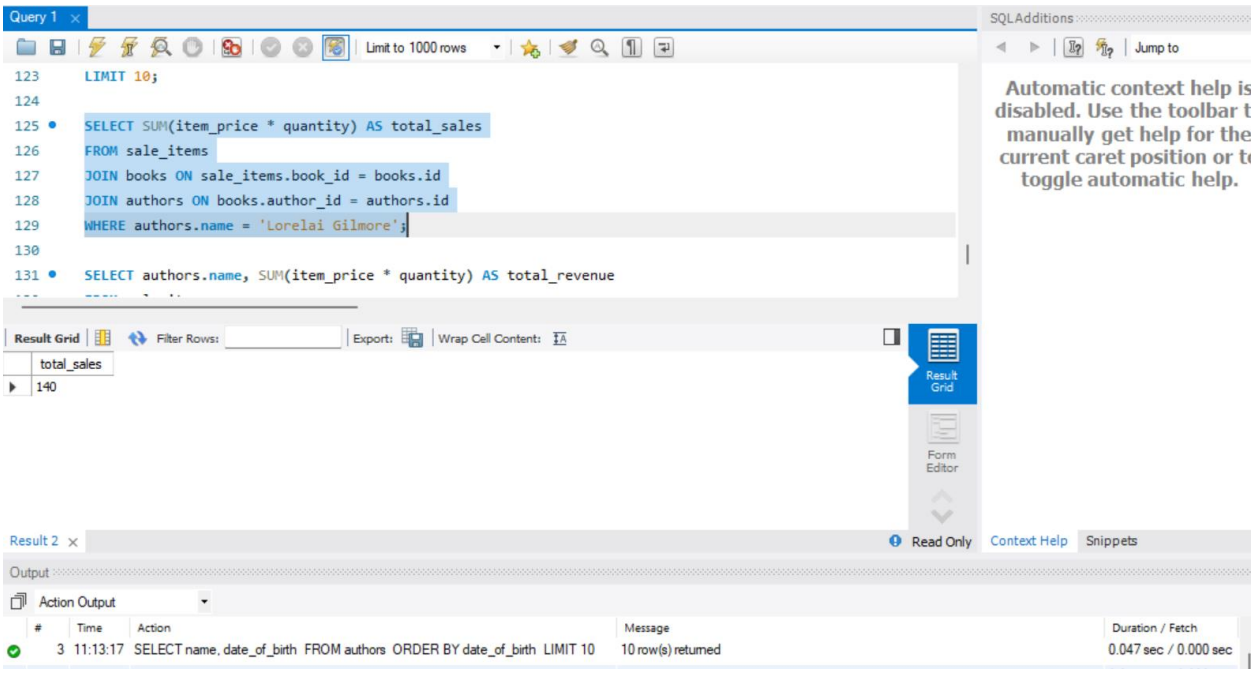
SELECT SUM(item_price * quantity) AS total_sales

FROM sale_items

JOIN books ON sale_items.book_id = books.id

JOIN authors ON books.author_id = authors.id

WHERE authors.name = 'Lorelai Gilmore';



3. What are the top 10 performing authors, ranked by sales revenue?

SELECT authors.name, SUM(item_price * quantity) AS total_revenue

FROM sale_items

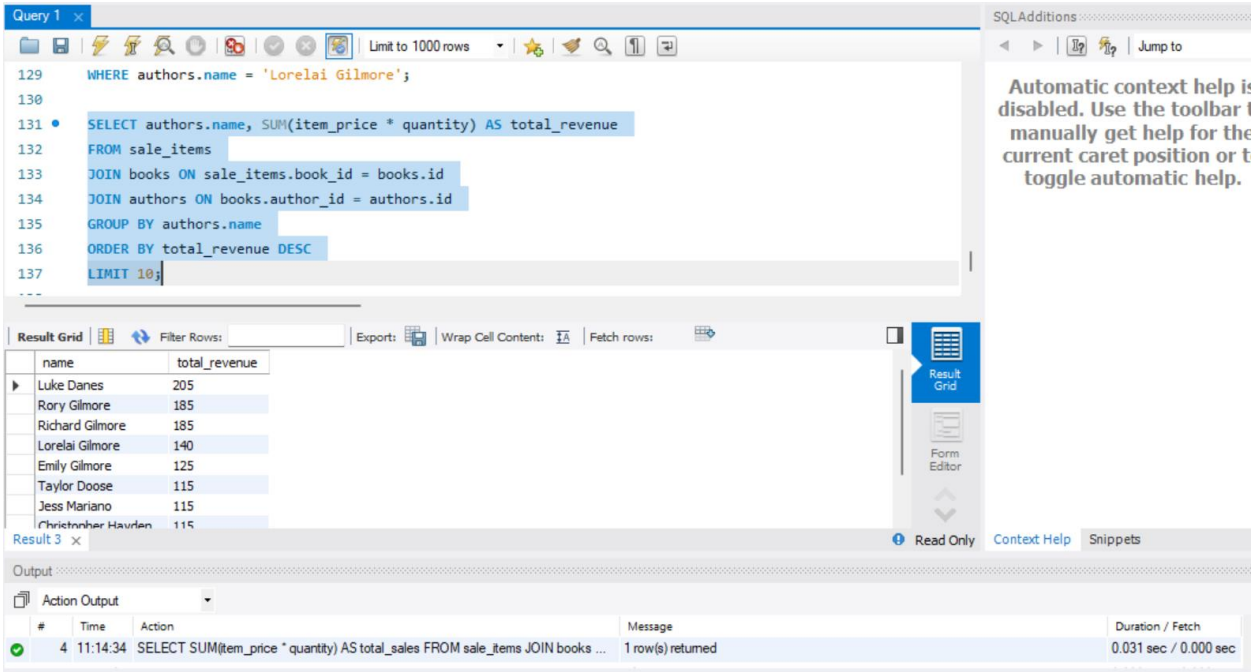
JOIN books ON sale_items.book_id = books.id

JOIN authors ON books.author_id = authors.id

GROUP BY authors.name

ORDER BY total_revenue DESC

LIMIT 10;



Part 2A: Write an API Endpoint

Install Node.js

Navigate to desktop and open cmd or Open your command prompt and change directory to your desktop

To Create Project Directory type below commands

- mkdir authors-api
- cd authors-api
- npm init -y
- npm install express mysql

Now Inside the "authors-api" folder, create a file named index.js and paste the below code.

Note: Please change the username and password with your MySQL username and password.

```
const express = require("express");

const mysql = require("mysql");


const app = express();

const port = 3000;
```

```
const pool = mysql.createPool({
  connectionLimit: 10,
  host: "localhost",
  user: "root",
  password: "akhila",
  database: "authors",
  port: 3306,
  insecureAuth: true,
});
app.get("/authors", (req, res) => {
  const authorName = req.query.author_name;

  let query = `
    SELECT authors.name, SUM(item_price * quantity) AS total_sales
    FROM sale_items
    JOIN books ON sale_items.book_id = books.id
    JOIN authors ON books.author_id = authors.id
    GROUP BY authors.name
    ORDER BY total_sales DESC
    LIMIT 10;
  `;

  if (authorName) {
    query = `
      SELECT authors.name, SUM(item_price * quantity) AS total_sales
      FROM sale_items
      JOIN books ON sale_items.book_id = books.id
      JOIN authors ON books.author_id = authors.id
      WHERE authors.name = ?
```

```
    GROUP BY authors.name;
  `;
}

pool.getConnection((err, connection) => {
  if (err) {
    console.error("Error connecting to database:", err);
    res.status(500).json({ error: "Internal server error" });
    return;
  }

  connection.query(query, [authorName], (error, results) => {
    connection.release();

    if (error) {
      console.error("Error executing query:", error);
      res.status(500).json({ error: "Internal server error" });
      return;
    }

    if (!results.length) {
      res.status(404).json({ error: "Author not found" });
      return;
    }

    res.json(results);
  });
});
});
```

```
app.listen(port, () => {
  console.log(` Server running on port ${port}` );
});
```

Optional: Before running the server please open mysql command line client and execute following two queries one after other.

Note: Please change the username and password with your MySQL username and password

1.ALTER USER 'root'@'localhost' IDENTIFIED WITH 'mysql_native_password' BY 'akhila';

2.FLUSH PRIVILEGES;

Now to run the server Open your command prompt, ensure you're in the "authors-api" directory.

Start the server by running following command.

- node index.js.

Now server starts running on port 3000.

Open chrome and visit http://localhost:3000/top-authors **to get the top 10 authors by sales revenue.**

To get sales data for a specific author**,** visit **http://localhost:3000/top-authors?author_name=Lorelai%20Gilmore** (replace "Lorelai%20Gilmore" with the any author's name of your choice)

localhost:3000/authors?author_name=Lorelai%20Gilmore

Pretty-print


```
[
  {
    "name": "Lorelai Gilmore",
    "total_sales": 140
  }
]
```


Part 2B: API Performance

Here again we repeat all the above steps from 2A but with different folder name and optimize the index.js file to handle traffic of 1000 simultaneous users concurrently.

Navigate to desktop and open cmd or Open your command prompt and change directory to your desktop

To Create Project Directory type below commands

- mkdir optimized-authors-api
- cd authors-api
- npm init -y
- npm install express mysql

Now Inside the "optimized-authors-api" folder, create a file named index.js and paste the below code.

Note: Please change the username and password with your MySQL username and password.

```
const express = require("express");

const mysql = require("mysql");

const app = express();

const port = 3000;


const pool = mysql.createPool({

  connectionLimit: 10,

  host: "localhost",

  user: "root",

  password: "akhila",

  database: "authors",
```

```
  port: 3306,
  insecureAuth: true,
});

const cache = new Map();

app.get("/authors", (req, res) => {
  const authorName = req.query.author_name;
  const cacheKey = authorName || "top_authors";



  if (cache.has(cacheKey)) {
    return res.json(cache.get(cacheKey));
  }

  let query = `
    SELECT authors.name, SUM(item_price * quantity) AS total_sales
    FROM sale_items
    JOIN books ON sale_items.book_id = books.id
    JOIN authors ON books.author_id = authors.id
    GROUP BY authors.name
    ORDER BY total_sales DESC
    LIMIT 10;
  `;

  const queryParams = authorName ? [authorName] : [];

  if (authorName) {
    query = `
```

```
    SELECT authors.name, SUM(item_price * quantity) AS total_sales
    FROM sale_items
    JOIN books ON sale_items.book_id = books.id
    JOIN authors ON books.author_id = authors.id
    WHERE authors.name = ?
    GROUP BY authors.name;
  `;
}

pool.getConnection((err, connection) => {
  if (err) {
    console.error("Error connecting to database:", err);
    res.status(500).json({ error: "Internal server error" });
    return;
  }

  connection.query(query, queryParams, (error, results) => {
    connection.release();

    if (error) {
      console.error("Error executing query:", error);
      res.status(500).json({ error: "Internal server error" });
      return;
    }

    if (!results.length) {
      res.status(404).json({ error: "Author not found" });
      return;
    }
```

```
    // Cache the results
    cache.set(cacheKey, results);

    res.json(results);
  });
 });
});

app.listen(port, () => {
 console.log(`Server running on port ${port}`);
});
```

Optional: Before running the server please open mysql command line client and execute following two queries one after other.

Note: Please change the username and password with your MySQL username and password

1.ALTER USER 'root'@'localhost' IDENTIFIED WITH 'mysql_native_password' BY 'akhila';

2.FLUSH PRIVILEGES;

Now to run the server Open your command prompt, ensure you're in the "optimized-authors-api" directory.

Start the server by running following command.

- node index.js.

Now server starts running on port 3000.

Open chrome and visit **http://localhost:3000/top-authors** **to get the top 10 authors by sales revenue.**

We get the same output as 2A but this index.js handles simultaneous requests concurrently.

To get sales data for a specific author**,** visit **http://localhost:3000/top-authors?author_name=Lorelai%20Gilmore** (replace "Lorelai%20Gilmore" with the any author's name of your choice)

We get the same output as 2A but this index.js handles simultaneous requests concurrently.

Part 3: Build & Deploy Webpage

Please implement a small React webpage that uses the endpoint from part 2. We have provided a design for you to implement here. Please use raw css.

- **You can use the authors from the above endpoint response as your team members and**

you can use a placeholder image for the profile picture.

Optimize the API Endpoint:

In the "optimized-authors-api" folder, modify the index.js file as instructed:

Import body-parser and cors.

below these two lines:

const express = require("express");

const mysql = require("mysql");

add the following two lines of code:

const bodyParser = require("body-parser");

const cors = require("cors");

Update the port number to 3001.

modify

const port = 3000; -> const port = 3001;

Add middleware to parse JSON requests and enable CORS.

app.use(bodyParser.json());

app.use(cors());

Save the changes and keep the terminal open.

Create React App (keep the above terminal open and open another new terminal In the same folder authors-app)

Open a new terminal and navigate to the "optimized-authors-api" folder.

Now type following commands:

- npx create-react-app client
- cd client
- npm install axios

now go to that folder normally in file explorer and open the files app.js and app.css in visual studio code and paste the following codes respectively.

App.js

```
import React, { useState, useEffect } from "react";

import axios from "axios";

import "./App.css";

import profileImage from "./assets/profile.png"; // Import the profile image


function App() {

  const [authors, setAuthors] = useState([]);

  const [authorName, setAuthorName] = useState("");

  const [error, setError] = useState("");

  const [showTopAuthors, setShowTopAuthors] = useState(true); // State to control the display of top
authors


  useEffect(() => {

    fetchAuthors();

  }, []);


  const fetchAuthors = () => {

    axios

      .get("http://localhost:3001/authors")

      .then((response) => {

        if (response.data.error) {

          setError(response.data.error + ": " + response.data.details);

          setAuthors([]);

        } else {
```

```
      setAuthors(response.data);
      setError("");
    }
  })
  .catch((error) => {
    console.error("Error fetching data:", error);
    setError("Internal server error: " + error.message);
    setAuthors([]);
  });
};

const handleSearch = () => {
 if (authorName.trim() === "") {
  setError("Please enter an author name");
  setAuthors([]);
  return;
 }

 axios
  .get("http://localhost:3001/authors", {
    params: { author_name: authorName },
  })
  .then((response) => {
    if (response.data.error) {
      setError(response.data.error);
      setAuthors([]);
    } else {
      setAuthors(response.data);
      setError("");
```

```
      setShowTopAuthors(false); // Hide the top authors when a search is made
    }
  })
  .catch((error) => {
    console.error("Error fetching data:", error);
    setError("Internal server error: " + error.message);
    setAuthors([]);
  });
};

const handleBack = () => {
 fetchAuthors();
 setShowTopAuthors(true);
 setAuthorName("");
};

return (
 <div className="container">
  <h1 className="title">Authors</h1>
  <div className="search-container">
   <input
    type="text"
    placeholder="Search by author name"
    value={authorName}
    onChange={(e) => setAuthorName(e.target.value)}
   />
   <button onClick={handleSearch}>Search</button>
  </div>
  {error && <p className="error">{error}</p>}
```

```
{showTopAuthors && (
  <>
    <h2 className="section-title">Top 10 Authors</h2>
    <ul className="author-list">
      {authors.map((author, index) => (
        <li key={index} className="author-item">
          <img src={profileImage} alt="Profile" />
          <div>
            <h2>{author.name}</h2>
            <p>Total Sales: ${author.total_sales}</p>
          </div>
        </li>
      ))}
    </ul>
  </>
)}
{!showTopAuthors && (
  <>
    <button className="back-button" onClick={handleBack}>
      Back to Top 10 Authors
    </button>
    <ul className="author-list">
      {authors.map((author, index) => (
        <li key={index} className="author-item">
          <img src={profileImage} alt="Profile" />
          <div>
            <h2>{author.name}</h2>
            <p>Total Sales: ${author.total_sales}</p>
          </div>
```

```
            </li>
          ))}
        </ul>
      </>
    )}
  </div>
 );
}

export default App;
```

App.css

```
.container {
  max-width: 800px;
  margin: 0 auto;
  padding: 20px;
  font-family: Arial, sans-serif;
}

.title {
  text-align: center;
}

.search-container {
  display: flex;
  justify-content: space-between;
  align-items: center;
  margin-bottom: 20px;
```

```
}

.search-container input {
  flex-grow: 1;
  padding: 10px;
  font-size: 16px;
  border: 1px solid #ccc;
  border-radius: 4px;
}

.search-container button {
  padding: 10px 20px;
  font-size: 16px;
  background-color: #007bff;
  color: white;
  border: none;
  border-radius: 4px;
  cursor: pointer;
}

.error {
  color: red;
  margin-bottom: 20px;
}

.not-found {
  color: #e74c3c;
  font-size: 18px;
  text-align: center;
```

```
  margin-top: 50px;
}

.author-list-container {
  margin-top: 10px;
}

.author-list {
  list-style-type: none;
  padding: 0;
}

.author-item {
  display: flex;
  align-items: center;
  border-bottom: 1px solid #ccc;
  padding: 10px 0;
}

.author-item img {
  width: 45px;
  height: 45px;
  margin-right: 20px;
}

.author-item h2 {
  margin: 0;
  font-size: 14px;
}
```

```
.author-item p {
  margin: 3px 0 0;
  font-size: 14px;
}

.back-button {
  margin-top: 20px;
  padding: 10px 20px;
  font-size: 16px;
  background-color: #007bff;
  color: white;
  border: none;
  border-radius: 4px;
  cursor: pointer;
  transition: background-color 0.3s ease;
}

.back-button:hover {
  background-color: #0056b3;
}

.back-button:active {
  background-color: #004499;
}
```

Now open package.json file in the **optimized-authors-api** folder using visual studio code and add the following line after line number 5
"proxy": "http://localhost:3001",

Now inside client folder -> src -> create a folder named “assets” and download a image from google and save it as profile.png in that assets folder.

Now open the previous terminal and click node index.js.

And open the second terminal and click npm start.