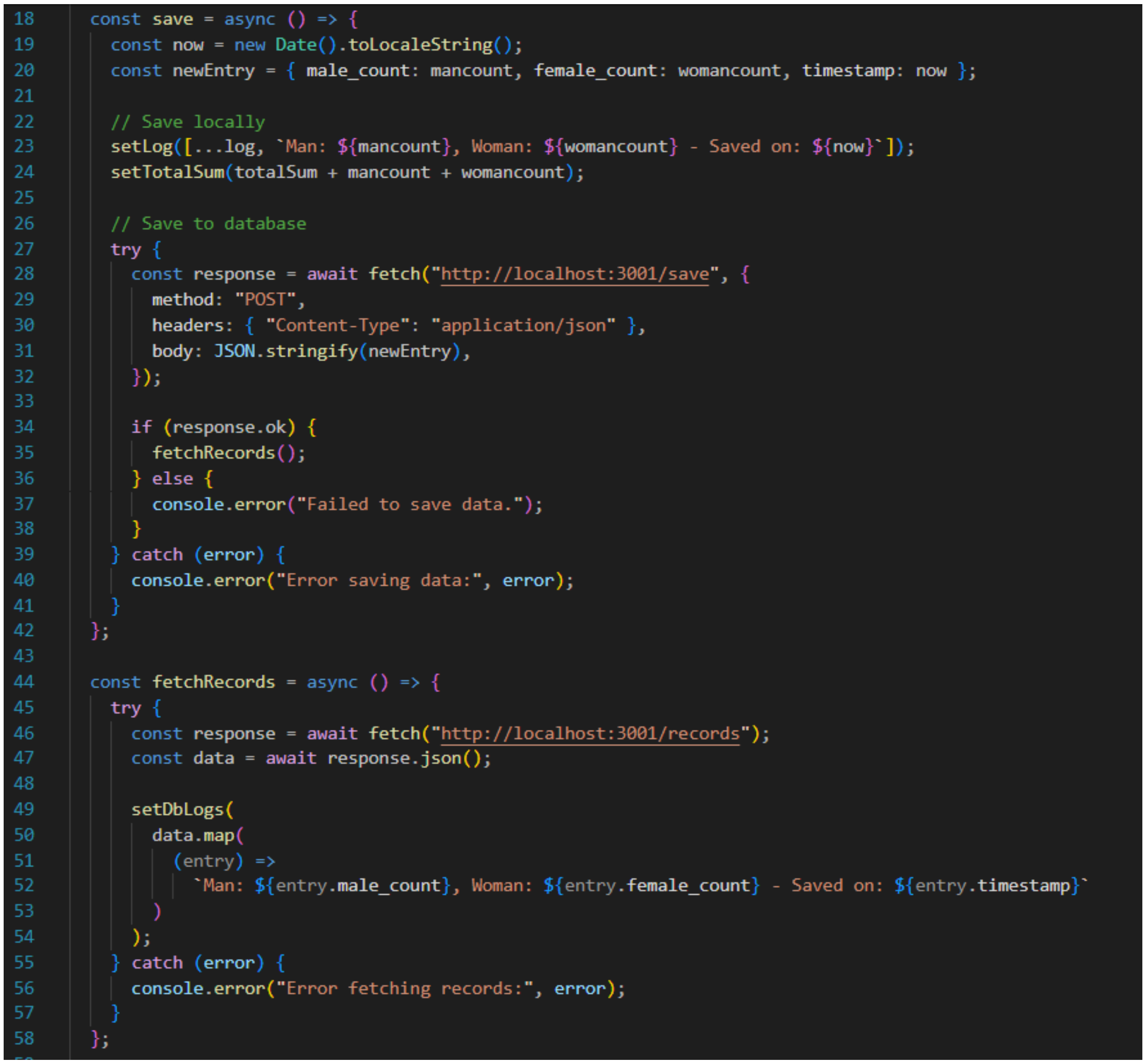
Fun 05 : MySQL

App.jsx

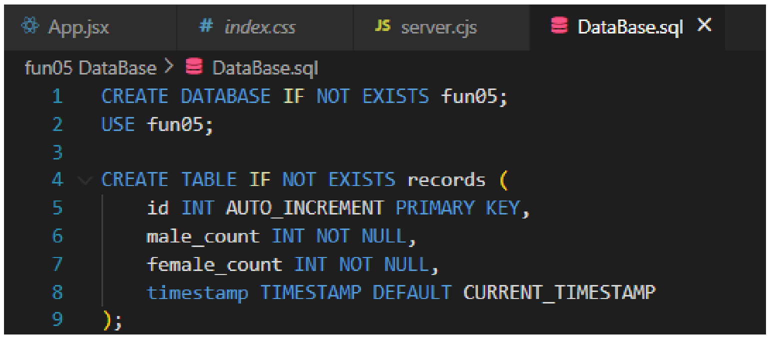
- Mostly remain the same as fun04 but I edited the function “save” to save on web (react) as local and save on database(sql) as always.



```
const save = async () => {
  const now = new Date().toLocaleString();
  const newEntry = { male_count: mancount, female_count: womancount, timestamp: now };

  // Save locally
  setLog([...log, `Man: ${mancount}, Woman: ${womancount} - Saved on: ${now}`]);
  setTotalSum(totalSum + mancount + womancount);

  // Save to database
  try {
    const response = await fetch("http://localhost:3001/save", {
      method: "POST",
      headers: { "Content-Type": "application/json" },
      body: JSON.stringify(newEntry),
    });

    if (response.ok) {
      fetchRecords();
    } else {
      console.error("Failed to save data.");
    }
  } catch (error) {
    console.error("Error saving data:", error);
  }
};

const fetchRecords = async () => {
  try {
    const response = await fetch("http://localhost:3001/records");
    const data = await response.json();

    setDbLogs(
      data.map(
        (entry) =>
          `Man: ${entry.male_count}, Woman: ${entry.female_count} - Saved on: ${entry.timestamp}`
      )
    );
  } catch (error) {
    console.error("Error fetching records:", error);
  }
};
```

DataBase.sql

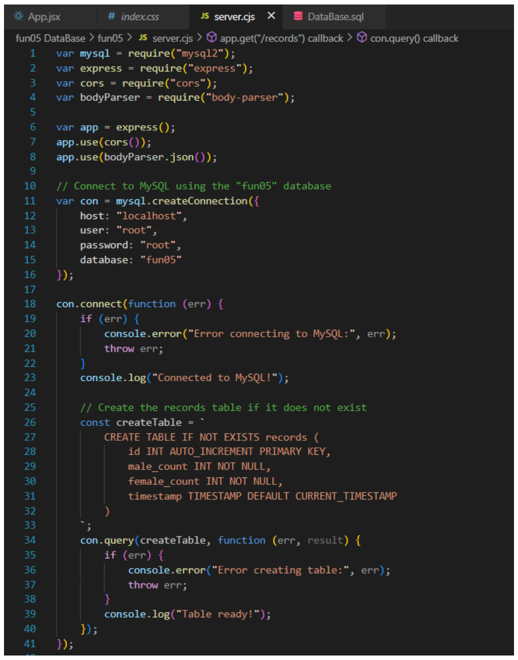
- Created the table that I want to save.



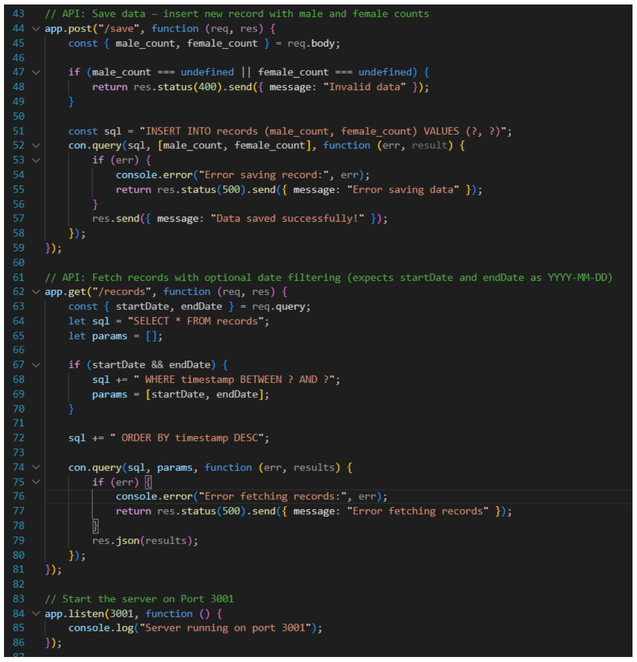
```sql
CREATE DATABASE IF NOT EXISTS fun05;
USE fun05;

CREATE TABLE IF NOT EXISTS records (
    id INT AUTO_INCREMENT PRIMARY KEY,
    male_count INT NOT NULL,
    female_count INT NOT NULL,
    timestamp TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

Server.cjs

- This part for connect the web to mysql and created the table for records the data.

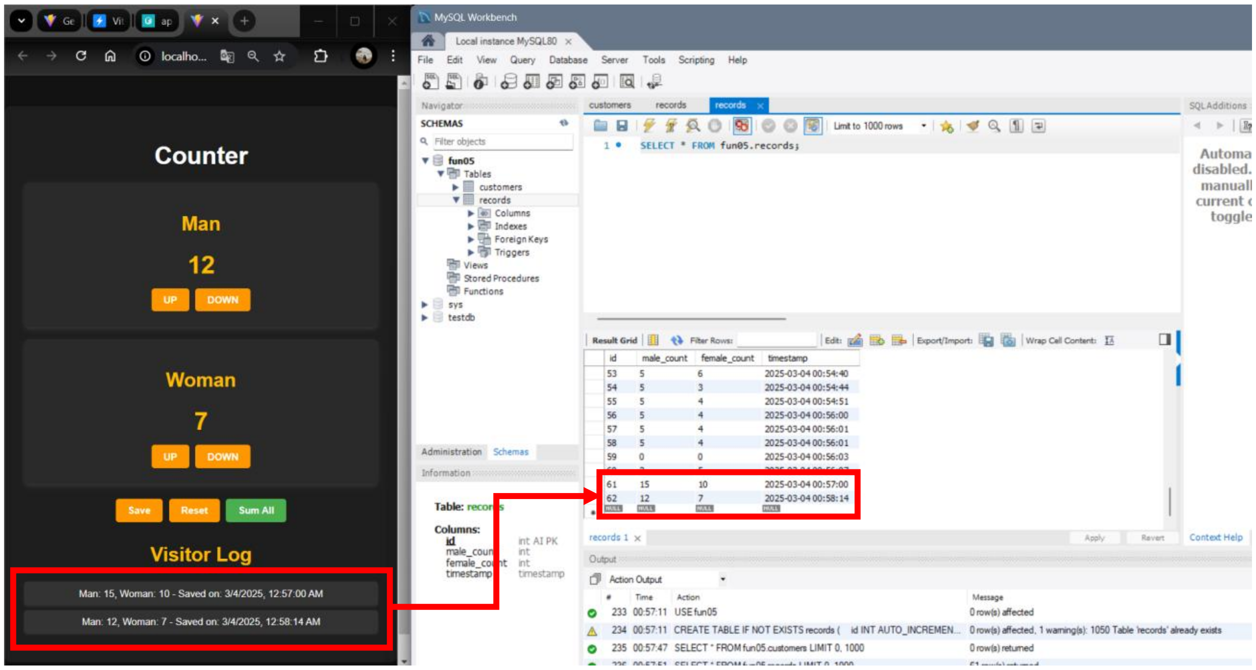

App.jsx
index.css
server.cjs
DataBase.sql
fun05 DataBase > fun05 > server.cjs > app.get("/records") callback > con.query() callback

```javascript
var mysql = require("mysql2");
var express = require("express");
var cors = require("cors");
var bodyParser = require("body-parser");

var app = express();
app.use(cors());
app.use(bodyParser.json());

// Connect to MySQL using the "fun05" database
var con = mysql.createConnection({
    host: "localhost",
    user: "root",
    password: "root",
    database: "fun05"
});

con.connect(function (err) {
    if (err) {
        console.error("Error connecting to MySQL:", err);
        throw err;
    }
    console.log("Connected to MySQL!");

    // Create the records table if it does not exist
    const createTable = `
        CREATE TABLE IF NOT EXISTS records (
            id INT AUTO_INCREMENT PRIMARY KEY,
            male_count INT NOT NULL,
            female_count INT NOT NULL,
            timestamp TIMESTAMP DEFAULT CURRENT_TIMESTAMP
        )
    `;
    con.query(createTable, function (err, result) {
        if (err) {
            console.error("Error creating table:", err);
            throw err;
        }
        console.log("Table ready!");
    });
});
```



```javascript
// API: Save data - insert new record with male and female counts
app.post("/save", function (req, res) {
    const { male_count, female_count } = req.body;

    if (male_count === undefined || female_count === undefined) {
        return res.status(400).send({ message: "Invalid data" });
    }

    const sql = "INSERT INTO records (male_count, female_count) VALUES (?, ?)";
    con.query(sql, [male_count, female_count], function (err, result) {
        if (err) {
            console.error("Error saving record:", err);
            return res.status(500).send({ message: "Error saving data" });
        }
        res.send({ message: "Data saved successfully!" });
    });
});

// API: Fetch records with optional date filtering (expects startDate and endDate as YYYY-MM-DD)
app.get("/records", function (req, res) {
    const { startDate, endDate } = req.query;
    let sql = "SELECT * FROM records";
    let params = [];

    if (startDate && endDate) {
        sql += " WHERE timestamp BETWEEN ? AND ?";
        params = [startDate, endDate];
    }

    sql += " ORDER BY timestamp DESC";

    con.query(sql, params, function (err, results) {
        if (err) {
            console.error("Error fetching records:", err);
            return res.status(500).send({ message: "Error fetching records" });
        }
        res.json(results);
    });
});

// Start the server on Port 3001
app.listen(3001, function () {
    console.log("Server running on port 3001");
});
```

Result



Counter

Man

12

Woman

7

Visitor Log

Man: 15, Woman: 10 - Saved on: 3/4/2025, 12:57:00 AM

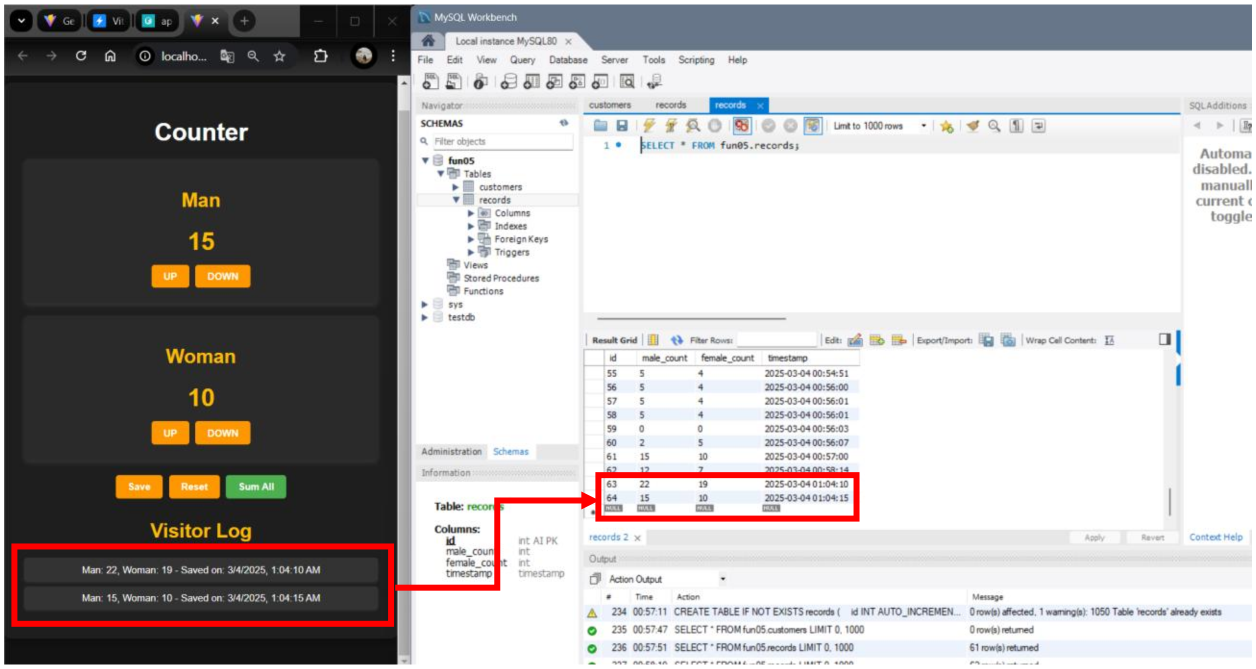
Man: 12, Woman: 7 - Saved on: 3/4/2025, 12:58:14 AM

```
SELECT * FROM fun05.records;
```

id	male_count	female_count	timestamp
53	5	6	2025-03-04 00:54:40
54	5	3	2025-03-04 00:54:44
55	5	4	2025-03-04 00:54:51
56	5	4	2025-03-04 00:56:00
57	5	4	2025-03-04 00:56:01
58	5	4	2025-03-04 00:56:01
59	0	0	2025-03-04 00:56:03
61	15	10	2025-03-04 00:57:00
62	12	7	2025-03-04 00:58:14
NULL	NULL	NULL	NULL

```
#    Time      Action                                                  Message
233  00:57:11  USE fun05                                               0 row(s) affected
234  00:57:11  CREATE TABLE IF NOT EXISTS records (   id INT AUTO_INCREMEN...  0 row(s) affected, 1 warning(s): 1050 Table 'records' already exists
235  00:57:47  SELECT * FROM fun05.customers LIMIT 0, 1000             0 row(s) returned
```

Reset and Save new data : On web also clear the data but all of them will be save at Database (SQL)



Counter

Man

15

Woman

10

Visitor Log

Man: 22, Woman: 19 - Saved on: 3/4/2025, 1:04:10 AM

Man: 15, Woman: 10 - Saved on: 3/4/2025, 1:04:15 AM

```
SELECT * FROM fun05.records;
```

id	male_count	female_count	timestamp
55	5	4	2025-03-04 00:54:51
56	5	4	2025-03-04 00:56:00
57	5	4	2025-03-04 00:56:01
58	5	4	2025-03-04 00:56:01
59	0	0	2025-03-04 00:56:03
60	2	5	2025-03-04 00:56:07
61	15	10	2025-03-04 00:57:00
63	22	19	2025-03-04 01:04:10
64	15	10	2025-03-04 01:04:15
NULL	NULL	NULL	NULL

```
#    Time      Action                                                  Message
234  00:57:11  CREATE TABLE IF NOT EXISTS records (   id INT AUTO_INCREMEN...  0 row(s) affected, 1 warning(s): 1050 Table 'records' already exists
235  00:57:47  SELECT * FROM fun05.customers LIMIT 0, 1000             0 row(s) returned
236  00:57:51  SELECT * FROM fun05.records LIMIT 0, 1000               61 row(s) returned
```