Лабораторна робота №2

Для отримання даних та їх відображення були створені наступні контролери та відображення:

Контроллери Cars:

```typescript
'use server'

import {sql} from "@vercel/postgres";
import {revalidatePath} from "next/cache";

export async function GET(id: number = -1) {
    return id === -1 ? (await sql`SELECT *
                                  FROM Cars;`).rows : (await sql`SELECT *
                                                                 FROM Cars
                                                                 WHERE id =
${id};`).rows
}

export async function POST(formData: FormData) {
    try {
        await sql`INSERT INTO Cars (Brand, Model, Year, Price)
                  VALUES (${formData.get('brand')},
                          ${formData.get('model')},
                          ${formData.get('year')},
                          ${formData.get('price')});
        `

        revalidatePath('/cars');
        return true;
    } catch (e) {
        return false;
    }
}

export async function PATCH(formData: FormData) {
    try {
        await sql`UPDATE Cars
                  SET Brand = ${formData.get('brand')},
                      Model = ${formData.get('model')},
                      Year = ${formData.get('year')},
                      Price = ${formData.get('price')}
                  WHERE id = ${formData.get('id')};
        `

        revalidatePath('/cars');
        return true;
    } catch (e) {
        return false;
    }
}

export async function DELETE(formData: FormData) {
    await sql`DELETE
              FROM Cars
              WHERE id = ${formData.get('id')};`
    revalidatePath('/cars');
}
```

Контролери Customers:

```
'use server';
import {sql} from "@vercel/postgres";
import {revalidatePath} from "next/cache";

export async function GET(id: number = -1) {
    return id === -1
        ? (await sql`SELECT *
                     FROM Customers;`).rows
        : (await sql`SELECT *
                     FROM Customers
                     WHERE id = ${id};`).rows;
}

export async function POST(formData: FormData) {
    try {
        await sql`INSERT INTO Customers (FirstName, LastName, Email, Phone)
                  VALUES (${formData.get("firstname")},
                          ${formData.get("lastname")},
                          ${formData.get("email")},
                          ${formData.get("phone")});
        `;
        revalidatePath("/customers");
        return true;
    } catch (e) {
        return false;
    }
}

export async function PATCH(formData: FormData) {
    try {
        await sql`UPDATE Customers
                  SET FirstName = ${formData.get("firstname")},
                      LastName  = ${formData.get("lastname")},
                      Email     = ${formData.get("email")},
                      Phone     = ${formData.get("phone")}
                  WHERE id = ${formData.get("id")};
        `;
        revalidatePath("/customers");
        return true;
    } catch (e) {
        return false;
    }
}

export async function DELETE(formData: FormData) {
    await sql`DELETE
              FROM Customers
              WHERE id = ${formData.get("id")};`;
    revalidatePath("/customers");
}
```

Контролери Employees:

```
'use server';
import {sql} from "@vercel/postgres";
import {revalidatePath} from "next/cache";
```

```typescript
export async function GET(id: number = -1) {
    return id === -1
        ? (await sql`SELECT *
                     FROM Employees;`).rows
        : (await sql`SELECT *
                     FROM Employees
                     WHERE id = ${id};`).rows;
}

export async function POST(formData: FormData) {
    try {
        await sql`INSERT INTO Employees (Firstname, Lastname, Position, Salary,
Password)
                  VALUES (${formData.get('firstname')},
                          ${formData.get('lastname')},
                          ${formData.get('position')},
                          ${formData.get('salary')},
                          ${formData.get('password')});
        `;
        revalidatePath('/employees');
        return true;
    } catch (e) {
        return false;
    }
}

export async function PATCH(formData: FormData) {
    try {
        await sql`UPDATE Employees
                  SET Firstname = ${formData.get('firstname')},
                      Lastname  = ${formData.get('lastname')},
                      Position  = ${formData.get('position')},
                      Salary    = ${formData.get('salary')},
                      Password    = ${formData.get('password')}
                  WHERE id = ${formData.get('id')};
        `;
        revalidatePath('/employees');
        return true;
    } catch (e) {
        return false;
    }
}

export async function DELETE(formData: FormData) {
    await sql`DELETE
              FROM Employees
              WHERE id = ${formData.get('id')};`;
    revalidatePath('/employees');
}
```
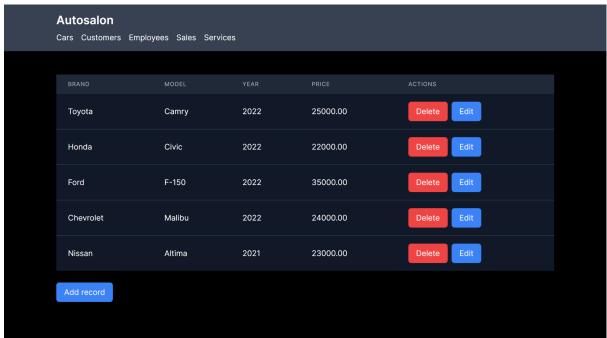
Контролери Sales:

```typescript
'use server';
import {sql} from "@vercel/postgres";
import {revalidatePath} from "next/cache";

export async function GET(id: number = -1) {
    return id === -1
        ? (await sql`SELECT *
```
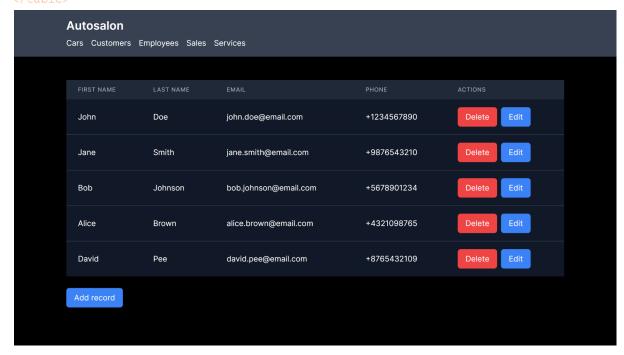
```
                            FROM Sales;`).rows
        : (await sql`SELECT *
                            FROM Sales
                            WHERE id = ${id};`).rows;
}

export async function POST(formData: FormData) {
    try {
        await sql`INSERT INTO Sales (CarID, CustomerID, SaleDate, SalePrice)
                    VALUES (${formData.get('carid')},
                            ${formData.get('customerid')},
                            ${formData.get('saledate')},
                            ${formData.get('saleprice')});
        `;
        revalidatePath('/sales');
        return true;
    } catch (e) {
        console.log(e)
        return false;
    }
}

export async function PATCH(formData: FormData) {
    try {
        await sql`UPDATE Sales
                    SET CarID      = ${formData.get('carid')},
                        CustomerID = ${formData.get('customerid')},
                        SaleDate   = ${formData.get('saledate')},
                        SalePrice  = ${formData.get('saleprice')}
                    WHERE Id = ${formData.get('id')};
        `;
        revalidatePath('/sales');
        return true;
    } catch (e) {
        return false;
    }
}

export async function DELETE(formData: FormData) {
    await sql`DELETE
                FROM Sales
                WHERE Id = ${formData.get('id')};
    `;
    revalidatePath('/sales');
}
```

Контролери Services:

```
'use server';
import {sql} from "@vercel/postgres";
import {revalidatePath} from "next/cache";

export async function GET(id = -1) {
    return id === -1
        ? (await sql`SELECT *
                            FROM Services;`).rows
        : (await sql`SELECT *
                            FROM Services
                            WHERE id = ${id};`).rows;
```

```
}

export async function POST(formData) {
    try {
        await sql`INSERT INTO Services (Service_name, Description, Price)
                    VALUES (${formData.get('service_name')},
                            ${formData.get('description')},
                            ${formData.get('price')};
        `;
        revalidatePath('/services');
        return true;
    } catch (e) {
        console.log(e)
        return false;
    }
}

export async function PATCH(formData) {
    try {
        await sql`UPDATE Services
                    SET Service_Name = ${formData.get('service_name')},
                        Description  = ${formData.get('description')},
                        Price        = ${formData.get('price')}
                    WHERE id = ${formData.get('id')};
        `;
        revalidatePath('/services');
        return true;
    } catch (e) {
        return false;
    }
}

export async function DELETE(formData) {
    await sql`DELETE
              FROM Services
              WHERE id = ${formData.get('id')};
    `;
    revalidatePath('/services');
}
```

Представлення Cars:

```
<table className="min-w-full">
    <thead>
    <tr>
        <th className="px-6 py-3 bg-gray-50 dark:bg-gray-800 text-left text-xs
leading-4 font-medium text-gray-500 dark:text-gray-400 uppercase tracking-wider">
            Brand
        </th>
        <th className="px-6 py-3 bg-gray-50 dark:bg-gray-800 text-left text-xs
leading-4 font-medium text-gray-500 dark:text-gray-400 uppercase tracking-wider">
            Model
        </th>
        <th className="px-6 py-3 bg-gray-50 dark:bg-gray-800 text-left text-xs
leading-4 font-medium text-gray-500 dark:text-gray-400 uppercase tracking-wider">
            Year
        </th>
        <th className="px-6 py-3 bg-gray-50 dark:bg-gray-800 text-left text-xs
leading-4 font-medium text-gray-500 dark:text-gray-400 uppercase tracking-wider">
            Price
        </th>
```

```
        <th className="px-6 py-3 bg-gray-50 dark:bg-gray-800 text-left text-xs
leading-4 font-medium text-gray-500 dark:text-gray-400 uppercase tracking-wider">
            Actions
        </th>
    </tr>
  </thead>
  <tbody className="bg-white dark:bg-gray-900 divide-y divide-gray-200
dark:divide-gray-700">
    {data.map((car) => (
      <tr key={car.id}>
        <td className="px-6 py-4 whitespace-no-wrap">{car.brand}</td>
        <td className="px-6 py-4 whitespace-no-wrap">{car.model}</td>
        <td className="px-6 py-4 whitespace-no-wrap">{car.year}</td>
        <td className="px-6 py-4 whitespace-no-wrap">{car.price}</td>
        <td className="px-6 py-4 whitespace-no-wrap">
                        </td>
      </tr>
    ))}
  </tbody>
</table>
```


Представлення Customers:

```
<table className="min-w-full">
  <thead>
    <tr>
      <th className="px-6 py-3 bg-gray-50 dark:bg-gray-800 text-left text-xs
leading-4 font-medium text-gray-500 dark:text-gray-400 uppercase tracking-wider">
          First Name
      </th>
      <th className="px-6 py-3 bg-gray-50 dark:bg-gray-800 text-left text-xs
leading-4 font-medium text-gray-500 dark:text-gray-400 uppercase tracking-wider">
          Last Name
      </th>
      <th className="px-6 py-3 bg-gray-50 dark:bg-gray-800 text-left text-xs
leading-4 font-medium text-gray-500 dark:text-gray-400 uppercase tracking-wider">
          Email
      </th>
```

```
      <th className="px-6 py-3 bg-gray-50 dark:bg-gray-800 text-left text-xs
leading-4 font-medium text-gray-500 dark:text-gray-400 uppercase tracking-wider">
        Phone
      </th>
      <th className="px-6 py-3 bg-gray-50 dark:bg-gray-800 text-left text-xs
leading-4 font-medium text-gray-500 dark:text-gray-400 uppercase tracking-wider">
        Actions
      </th>
    </tr>
    </thead>
    <tbody className="bg-white dark:bg-gray-900 divide-y divide-gray-200
dark:divide-gray-700">
    {data.map((customer) => (
      <tr key={customer.id}>
        <td className="px-6 py-4 whitespace-no-wrap">{customer.firstname}</td>
        <td className="px-6 py-4 whitespace-no-wrap">{customer.lastname}</td>
        <td className="px-6 py-4 whitespace-no-wrap">{customer.email}</td>
        <td className="px-6 py-4 whitespace-no-wrap">{customer.phone}</td>
        <td className="px-6 py-4 whitespace-no-wrap">
                    </td>
      </tr>
    ))}
    </tbody>
</table>
```


Представлення Employees:
```
<table className="min-w-full">
    <thead>
    <tr>
      <th className="px-6 py-3 bg-gray-50 dark:bg-gray-800 text-left text-xs
leading-4 font-medium text-gray-500 dark:text-gray-400 uppercase tracking-wider">
        First Name
      </th>
      <th className="px-6 py-3 bg-gray-50 dark:bg-gray-800 text-left text-xs
leading-4 font-medium text-gray-500 dark:text-gray-400 uppercase tracking-wider">
        Last Name
      </th>
```

```jsx
        <th className="px-6 py-3 bg-gray-50 dark:bg-gray-800 text-left text-xs
leading-4 font-medium text-gray-500 dark:text-gray-400 uppercase tracking-wider">
            Position
        </th>
        <th className="px-6 py-3 bg-gray-50 dark:bg-gray-800 text-left text-xs
leading-4 font-medium text-gray-500 dark:text-gray-400 uppercase tracking-wider">
            Salary
        </th>
        <th className="px-6 py-3 bg-gray-50 dark:bg-gray-800 text-left text-xs
leading-4 font-medium text-gray-500 dark:text-gray-400 uppercase tracking-wider">
            Password
        </th>
        <th className="px-6 py-3 bg-gray-50 dark:bg-gray-800 text-left text-xs
leading-4 font-medium text-gray-500 dark:text-gray-400 uppercase tracking-wider">
            Actions
        </th>
    </tr>
    </thead>
    <tbody className="bg-white dark:bg-gray-900 divide-y divide-gray-200
dark:divide-gray-700">
    {employeesData.map((employee) => (
        <tr key={employee.employeeid}>
            <td className="px-6 py-4 whitespace-no-wrap">{employee.firstname}</td>
            <td className="px-6 py-4 whitespace-no-wrap">{employee.lastname}</td>
            <td className="px-6 py-4 whitespace-no-wrap">{employee.position}</td>
            <td className="px-6 py-4 whitespace-no-wrap">{employee.salary}</td>
            <td className="px-6 py-4
whitespace-no-wrap">{'*'.repeat(employee.password?.length || 0)}</td>
            <td className="px-6 py-4 whitespace-no-wrap">

            </td>
        </tr>
    ))}
    </tbody>
</table>
```


Представлення Sales:

```jsx
<table className="min-w-full">
```

```
    <thead>
    <tr>
        <th className="px-6 py-3 bg-gray-50 dark:bg-gray-800 text-left text-xs
leading-4 font-medium text-gray-500 dark:text-gray-400 uppercase tracking-wider">
            CarID
        </th>
        <th className="px-6 py-3 bg-gray-50 dark:bg-gray-800 text-left text-xs
leading-4 font-medium text-gray-500 dark:text-gray-400 uppercase tracking-wider">
            CustomerID
        </th>
        <th className="px-6 py-3 bg-gray-50 dark:bg-gray-800 text-left text-xs
leading-4 font-medium text-gray-500 dark:text-gray-400 uppercase tracking-wider">
            Sale Date
        </th>
        <th className="px-6 py-3 bg-gray-50 dark:bg-gray-800 text-left text-xs
leading-4 font-medium text-gray-500 dark:text-gray-400 uppercase tracking-wider">
            Sale Price
        </th>
        <th className="px-6 py-3 bg-gray-50 dark:bg-gray-800 text-left text-xs
leading-4 font-medium text-gray-500 dark:text-gray-400 uppercase tracking-wider">
            Actions
        </th>
    </tr>
    </thead>
    <tbody className="bg-white dark:bg-gray-900 divide-y divide-gray-200
dark:divide-gray-700">
    {data.map((sale) => (
        <tr key={sale.saleid}>
            <td className="px-6 py-4 whitespace-no-wrap">{sale.carid}</td>
            <td className="px-6 py-4 whitespace-no-wrap">{sale.customerid}</td>
            <td className="px-6 py-4
whitespace-no-wrap">{sale.saledate.toUTCString()}</td>
            <td className="px-6 py-4 whitespace-no-wrap">{sale.saleprice}</td>
            <td className="px-6 py-4 whitespace-no-wrap">

            </td>
        </tr>
    ))}
    </tbody>
</table>
```

Представлення Services:

```jsx
<table className="min-w-full">
    <thead>
    <tr>
        <th className="px-6 py-3 bg-gray-50 dark:bg-gray-800 text-left text-xs
leading-4 font-medium text-gray-500 dark:text-gray-400 uppercase tracking-wider">
            Service Name
        </th>
        <th className="px-6 py-3 bg-gray-50 dark:bg-gray-800 text-left text-xs
leading-4 font-medium text-gray-500 dark:text-gray-400 uppercase tracking-wider">
            Description
        </th>
        <th className="px-6 py-3 bg-gray-50 dark:bg-gray-800 text-left text-xs
leading-4 font-medium text-gray-500 dark:text-gray-400 uppercase tracking-wider">
            Price
        </th>
        <th className="px-6 py-3 bg-gray-50 dark:bg-gray-800 text-left text-xs
leading-4 font-medium text-gray-500 dark:text-gray-400 uppercase tracking-wider">
            Actions
        </th>
    </tr>
    </thead>
    <tbody className="bg-white dark:bg-gray-900 divide-y divide-gray-200
dark:divide-gray-700">
    {data.map((service) => (
        <tr key={service.id}>
            <td className="px-6 py-4 whitespace-no-wrap">{service.service_name}</td>
            <td className="px-6 py-4 whitespace-no-wrap">{service.description}</td>
            <td className="px-6 py-4 whitespace-no-wrap">{service.price}</td>
            <td className="px-6 py-4 whitespace-no-wrap">


            </td>
        </tr>
    ))}
    </tbody>
</table>
```

Autosalon

Cars Customers Employees Sales Services

SERVICE NAME	DESCRIPTION	PRICE	ACTIONS
Regular Maintenance Service	Scheduled maintenance to keep the vehicle in good condition.	100.00	Delete Edit
Brake Pad Replacement	Replacement of brake pads and brake system work.	150.00	Delete Edit
Engine Diagnostics	Engine diagnostics to detect issues and facilitate repairs.	75.00	Delete Edit
Oil and Filter Change	Routine oil and filter change to maintain engine performance.	50.00	Delete Edit
Climate System Repair	Repair and servicing of the vehicles climate control system.	120.00	Delete Edit

Add record