

Final Project Report

MANAGEMENT OF DEVICE

*Professor*: Jahan Ghofraniha

*Course*: CS 157A, Fall 2022

*Section*: 01

*Team member:*

Quan Le

Bao Nguyen

TABLE OF CONTENTS

[I. EXECUTIVE SUMMARY 3](#_gjdgxs)

[II. BACKGROUND/INTRODUCTION 4](#_30j0zll)

[III. PROBLEM STATEMENT 5](#_1fob9te)

[IV. PURPOSE/MOTIVATION 6](#_3znysh7)

[V. DESIGN 7](#_2et92p0)

[VI. IMPLEMENTATION & TEST REPORT 7](#_tyjcwt)

[VII. CONCLUSION 7](#_3dy6vkm)

# I. EXECUTIVE SUMMARY

"Management of Device" is a straightforward full-stack web application that our team created. This program displays data about the business, including information from users' given machines, employees, and clients. If any information about the company is incorrect, they are permitted to edit it. Users can also add and remove customer and employee information, as well as add machines for devices where they have only recently started working together. Users will be better able to control that device by quickly scanning the reviews and information they received.

# II. BACKGROUND/INTRODUCTION

Copiers, printers, scanners, and fax machines are indispensable office equipment for the operation of a company. They are also popular devices for individuals with printing and paperless storage needs. High-end printing equipment needs periodic maintenance, replacement of spare parts and consumables to ensure print quality.

"Management of Device" is software to manage the devices being used by customers in order to improve the quality of repair and maintenance services for customers.

# III. PROBLEM STATEMENT

After purchasing office equipment, the customer needs an after-sales service such as maintenance, replacement of consumables, and spare parts. "Management of Device" is customer device management software that allows sales and service companies to store and query information about customers, their equipment, and device history in order to provide customers with fast and quality service.

# IV. PURPOSE/MOTIVATION

Ease of service calls, assigning the right technician, and identifying the correct device model, are the primary goals of "Management of Device". Customers only need to report the machine ID and symptoms; the technician will respond quickly and accurately to the repair based on the machine information.

"Management of devices" will be expanded with other useful features such as machine location, machine status, and machine failure history to improve service quality.

# V. DESIGN

## **Conceptual Design**

There are six entities: Departments, Customers, Employees, Machines, Models, and States.

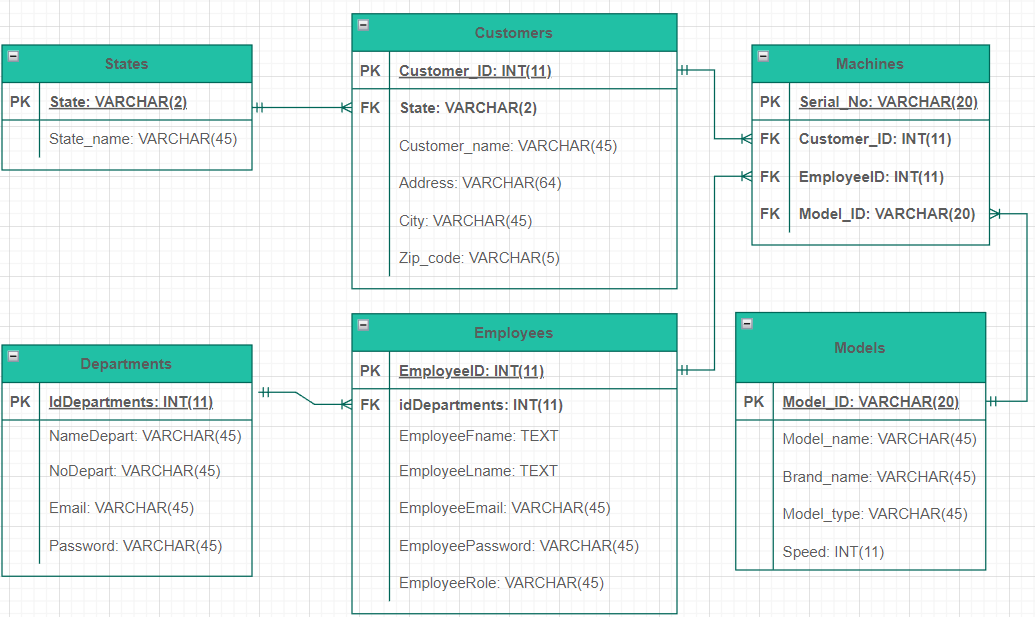
* Departments: idDepartments, nameDepart, noDepart, email, password.
* Customers: Customer\_ID, Customer\_name, Address, City, State, Zip\_code.
* Employees: employeeID, employeeFName, employeeLName, employeeEmail, employeePassword, employeeRole, idDepartments
* Machines: Serial\_no, Customer\_ID, employeeID, Model\_ID
* Models: Model\_ID, Model\_name, Brand\_name, Model\_type
* States: State, State\_name

Relationships between the tables:

1. States to Customers: 1 to many - Primary key: **State**
2. Departments to Employees: 1 to many - Primary key: **idDepartments**
3. Customers to Machines: 1 to many - Primary key: **Customer\_ID**
4. Employees to Machines: 1 to many - Primary key: **EmployeeID**
5. Models to Machines: 1 to many - Primary key: **Model\_ID**

## **Logical Model**

ERD:



## **Physical Model**

**Tables names**

**1. States**:

CREATE TABLE States (

state VARCHAR(2) NOT NULL PRIMARY KEY,

state\_name VARCHAR(45) NOT NULL

);

**2. Departments:**

CREATE TABLE Departments (

idDepartment INT(11) NOT NULL PRIMARY KEY AUTO\_INCREMENT,

nameDepart VARCHAR(45) NOT NULL,

noDepart VARCHAR(45),

email VARCHAR(45) NOT NULL,

password VARCHAR(45) NOT NULL

);

**3. Employees:**

CREATE TABLE Employees (

employeeID INT(11) NOT NULL PRIMARY KEY AUTO\_INCREMENT,

employeeFname TEXT NOT NULL,

employeeLname TEXT NOT NULL,

employeeEmail VARCHAR(45) NOT NULL,

employeePassword VARCHAR(45) NOT NULL,

employeeRole VARCHAR(45) NOT NULL,

idDepartments INT(11) NOT NULL,

FOREIGN KEY (idDepartments) REFERENCES Departments(idDepartments)

ON UPDATE CASCADE

);

**4. Models:**

CREATE TABLE Models (

model\_ID VARCHAR(20) NOT NULL PRIMARY KEY,

model\_name VARCHAR(45) NOT NULL ,

brand\_name VARCHAR(45) NOT NULL,

model\_type VARCHAR(45) NOT NULL,

speed INT(11) NOT NULL

);

**5. Customers:**

CREATE TABLE Customers (

customer\_ID INT(11) NOT NULL PRIMARY KEY AUTO\_INCREMENT,

customer\_name VARCHAR(45) NOT NULL,

address VARCHAR(64) NOT NULL,

city VARCHAR(45) NOT NULL,

state VARCHAR(2) NOT NULL,

zip\_code VARCHAR(5) NOT NULL,

FOREIGN KEY (state) REFERENCES States(state)

ON UPDATE CASCADE

);

**6. Machines:**

CREATE TABLE Machines (

serial\_no VARCHAR(30) NOT NULL PRIMARY KEY,

customer\_ID INT(11) NOT NULL,

employeeID INT(11) NOT NULL,

model\_ID VARCHAR(20) NOT NULL,

FOREIGN KEY (customer\_ID) REFERENCES Customers(customer\_ID)

ON DELETE CASCADE ON UPDATE CASCADE,

FOREIGN KEY (employeeID) REFERENCES Employees(employeeID)

ON UPDATE CASCADE,

FOREIGN KEY (model\_ID) REFERENCES Models(model\_ID)

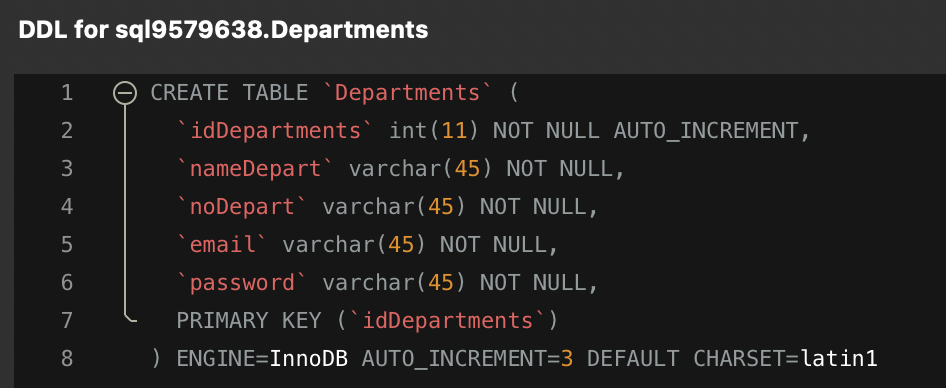
ON UPDATE CASCADE

);

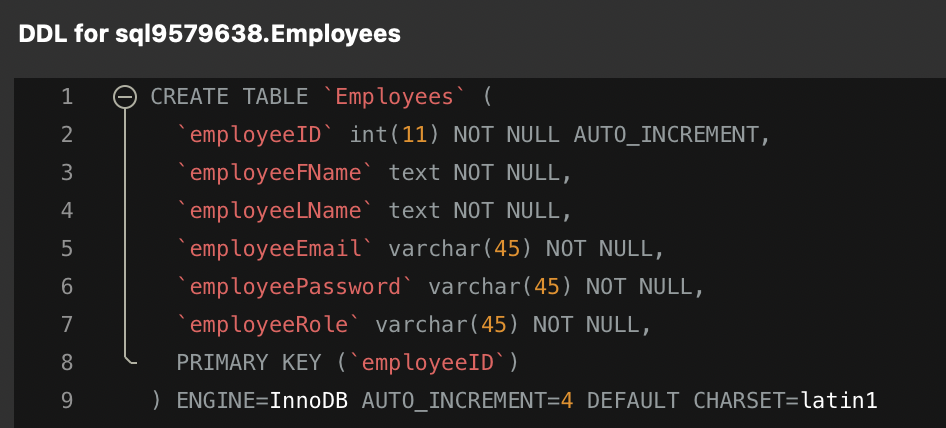
# VI. IMPLEMENTATION & TEST REPORT

## **SQL query**

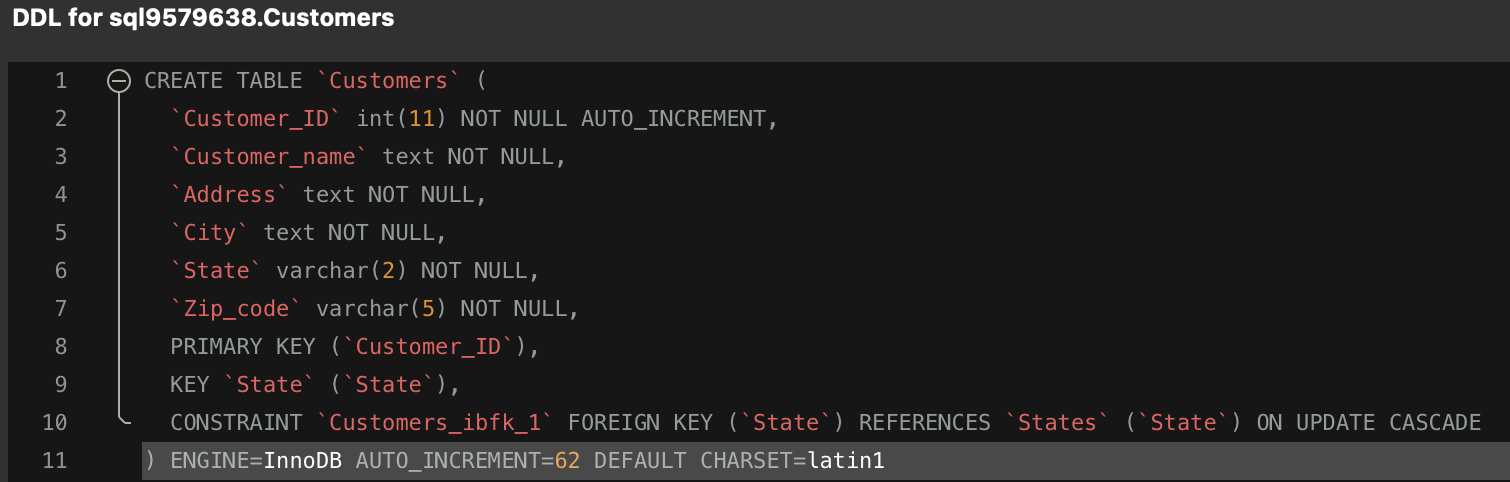
* Query to **create** Departments table:



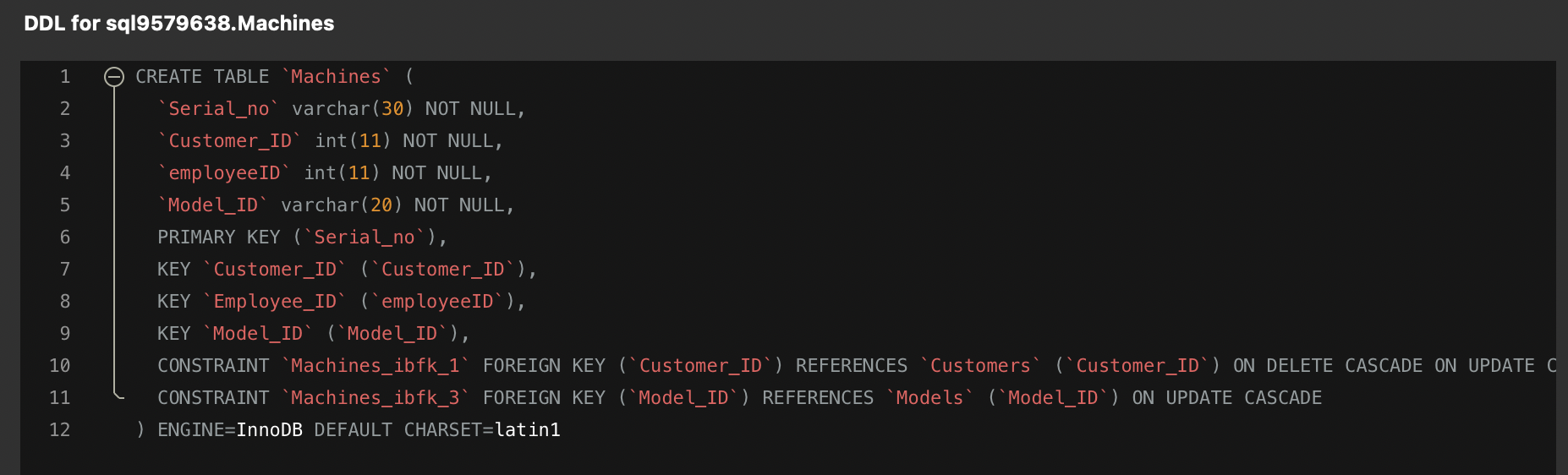
* Query to **create** Employees table:



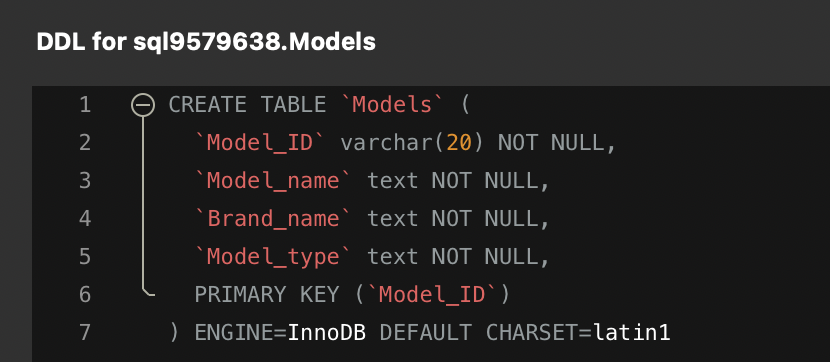
* Query to **create** Customers table:



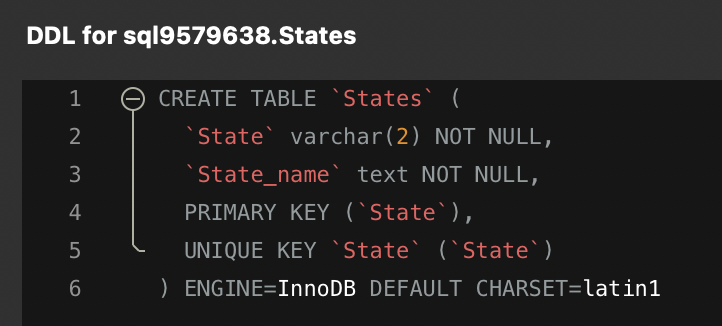
* Query to **create** Machines table:



* Query to **create** Models table:



* Query to **create** States table:

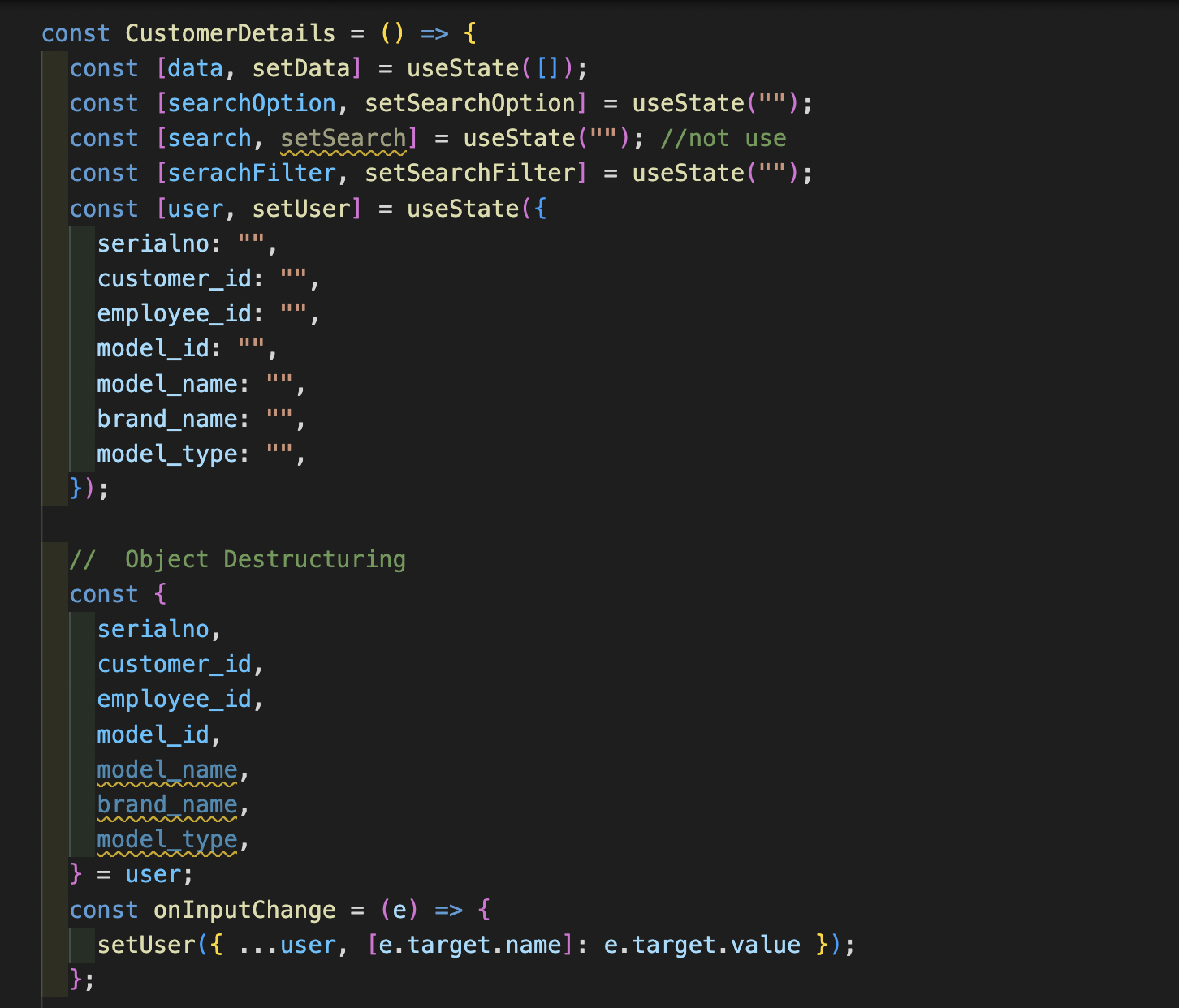


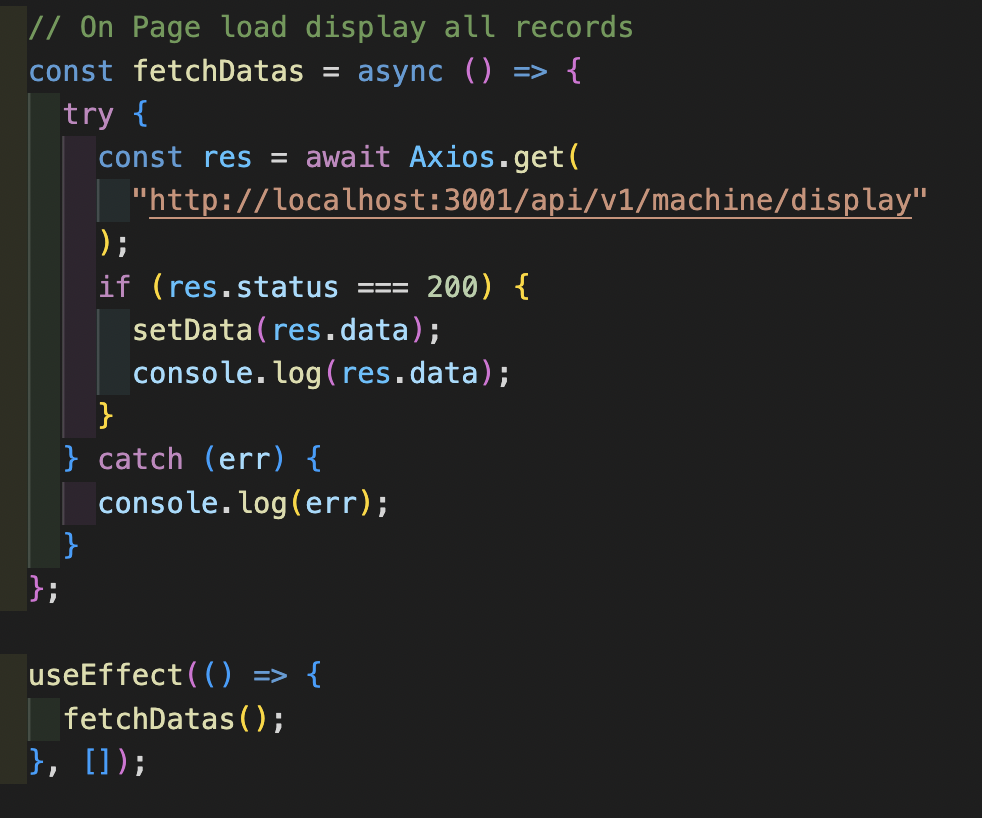
* Query **insert** value into Departments table
* Query **insert** value into Employees table
* Query **insert** value into Customers table
* Query **insert** value into Machines table
* Query **insert** value into Models table
* Query **insert** value into States table
* Query to **update** a Departments details
* Query to **update** a Employees details:
* Query to **update** a Customers details:
* Query to **update** a Machines details:
* Query to **update** a Models details:
* Query to **update** a States details
* Query to **delete** a Departments details
* Query to **delete** a Employees details:
* Query to **delete** a Customers details:
* Query to **delete** a Machines details:
* Query to **delete** a Models details:
* Query to **delete** a States details:

## **Function implementation**

* **Frontend:**

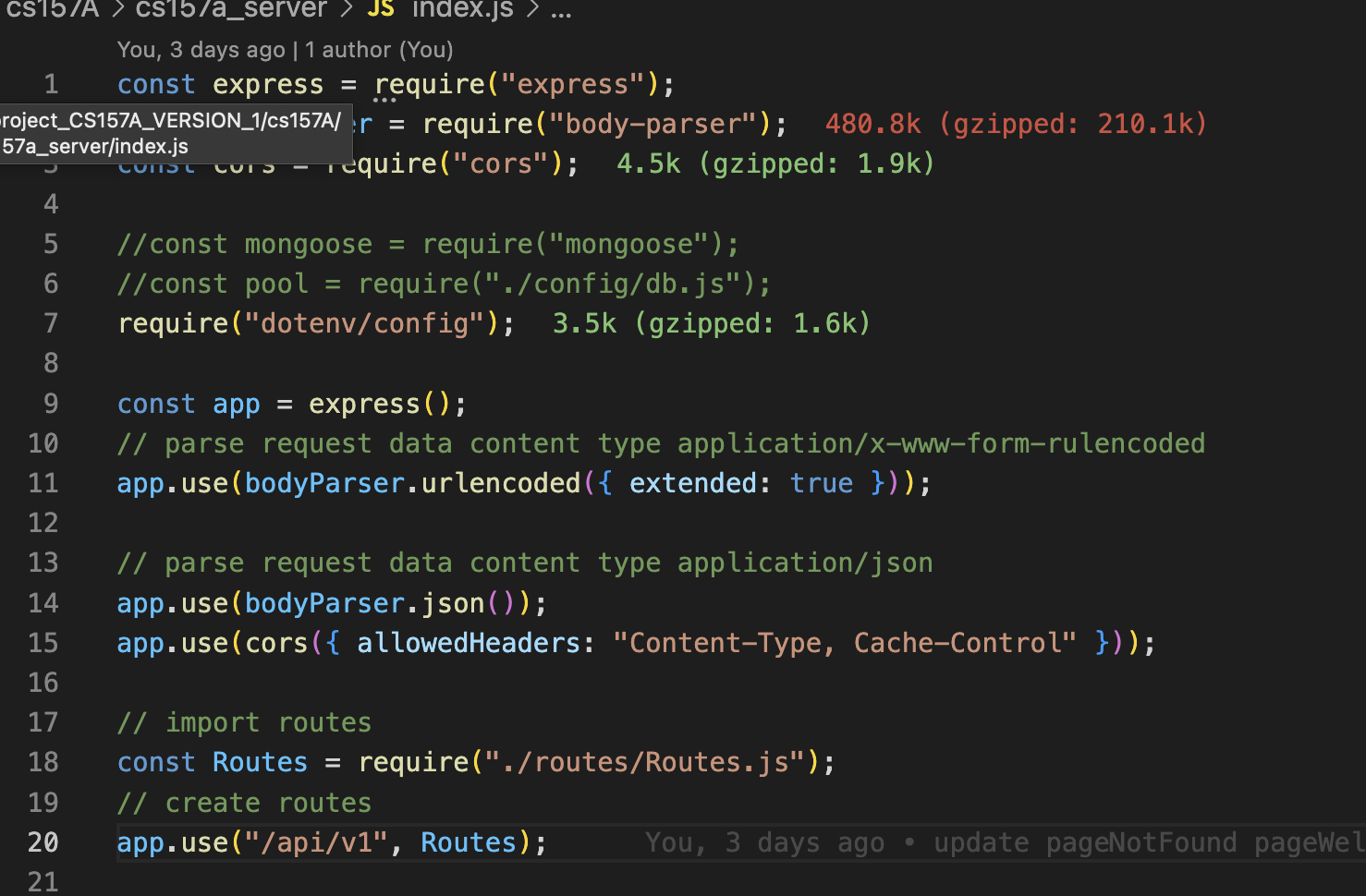
Frontend is built on ReactJS and Bootstrap framework. Every function is responsible for accepting input from users, in multiple forms, and parsing them. Therefore, the backend can recognize the data and execute the corresponding queries.

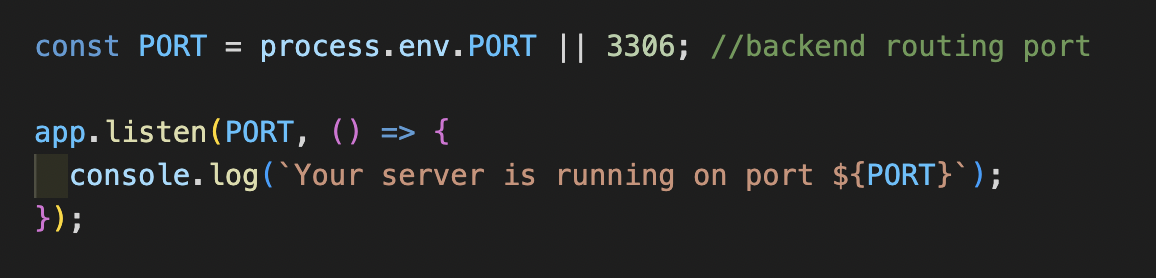


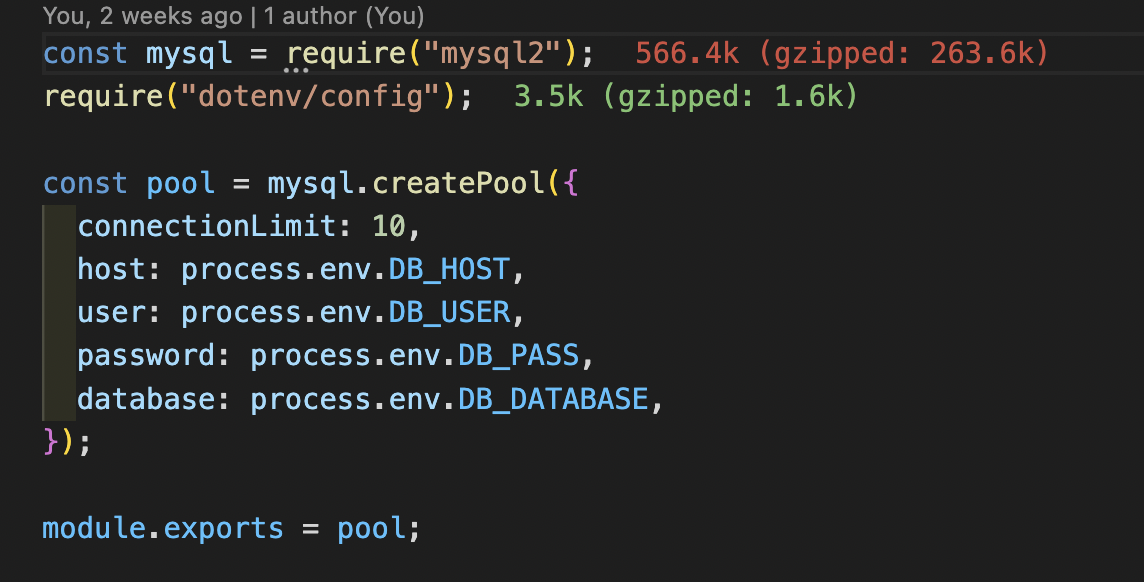


* **Backend:**

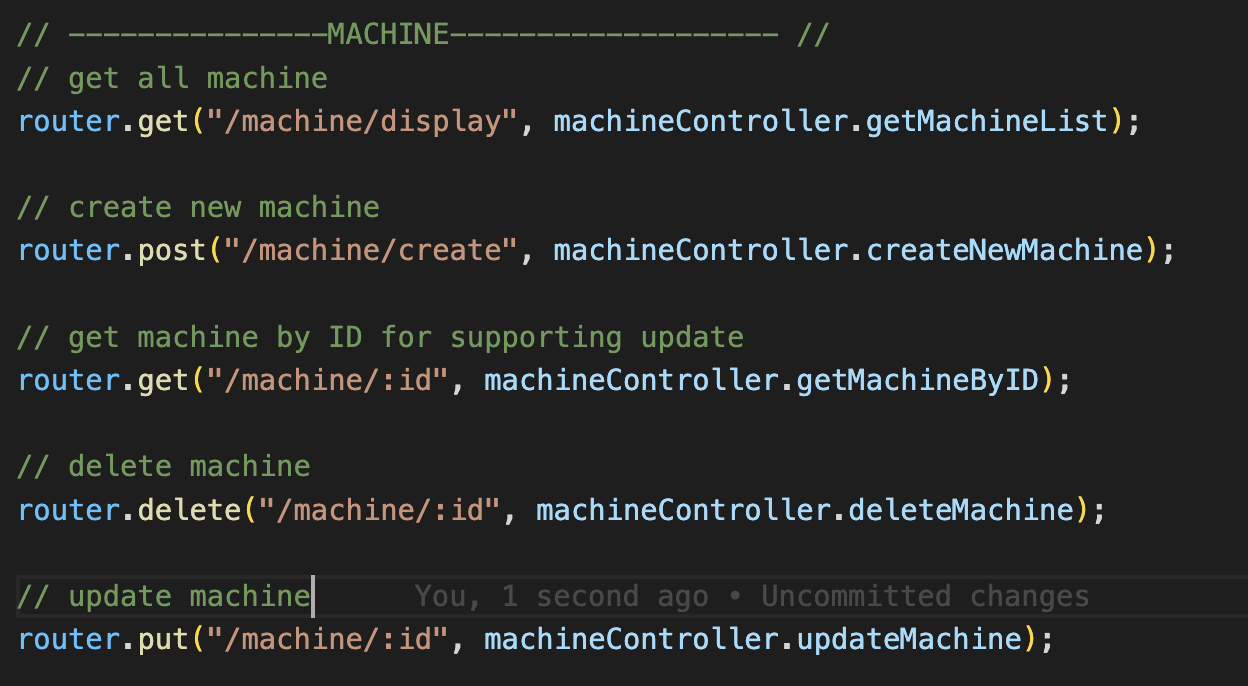
NodeJS, ExpressJS is utilized to set up the backend. A middleware is being used to create the connection between backend and database. In addition, to secure the connection and limit accessing, a file called .env is created to hold all environment variables that are required for accessing the database.

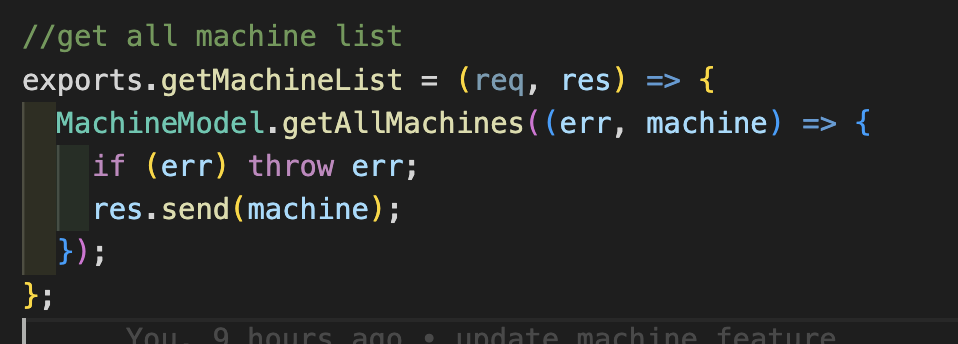






Note that the backend is built as a RESTful API. That being said, HTTP requests such as PUT, POST, UPDATE and DELETE to perform actions like adding, pulling, altering and deleting data off the database. Below is an example of how retrieve all list machine in the database is implemented:

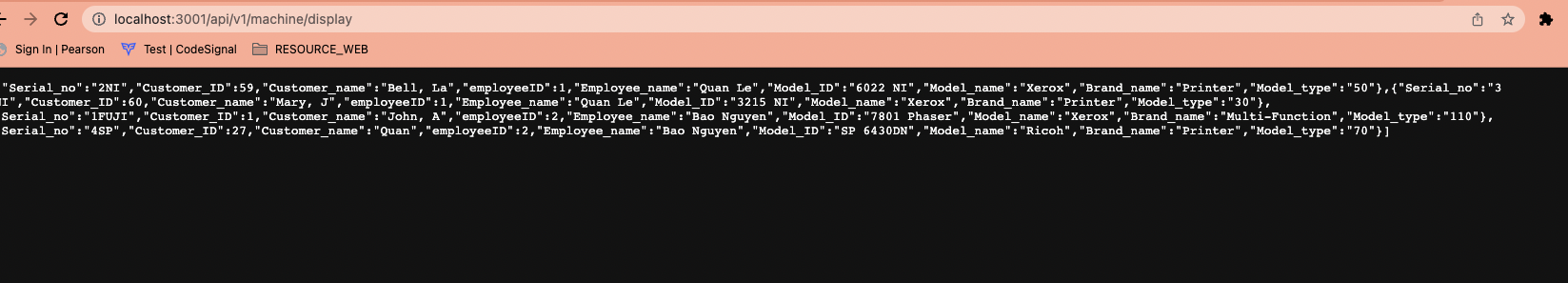


****

****

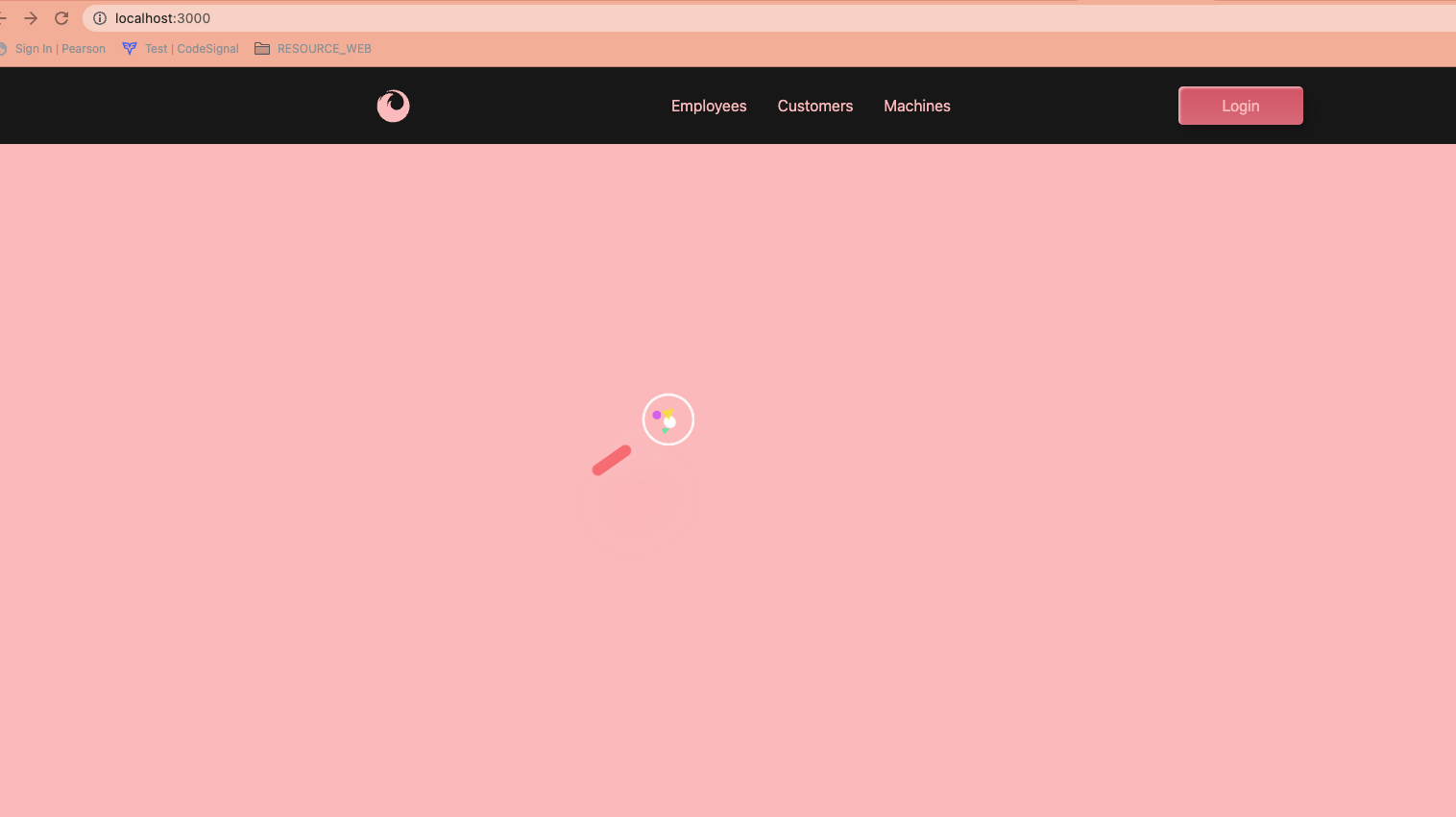
## **Test Data**

* Retrieve all list machine

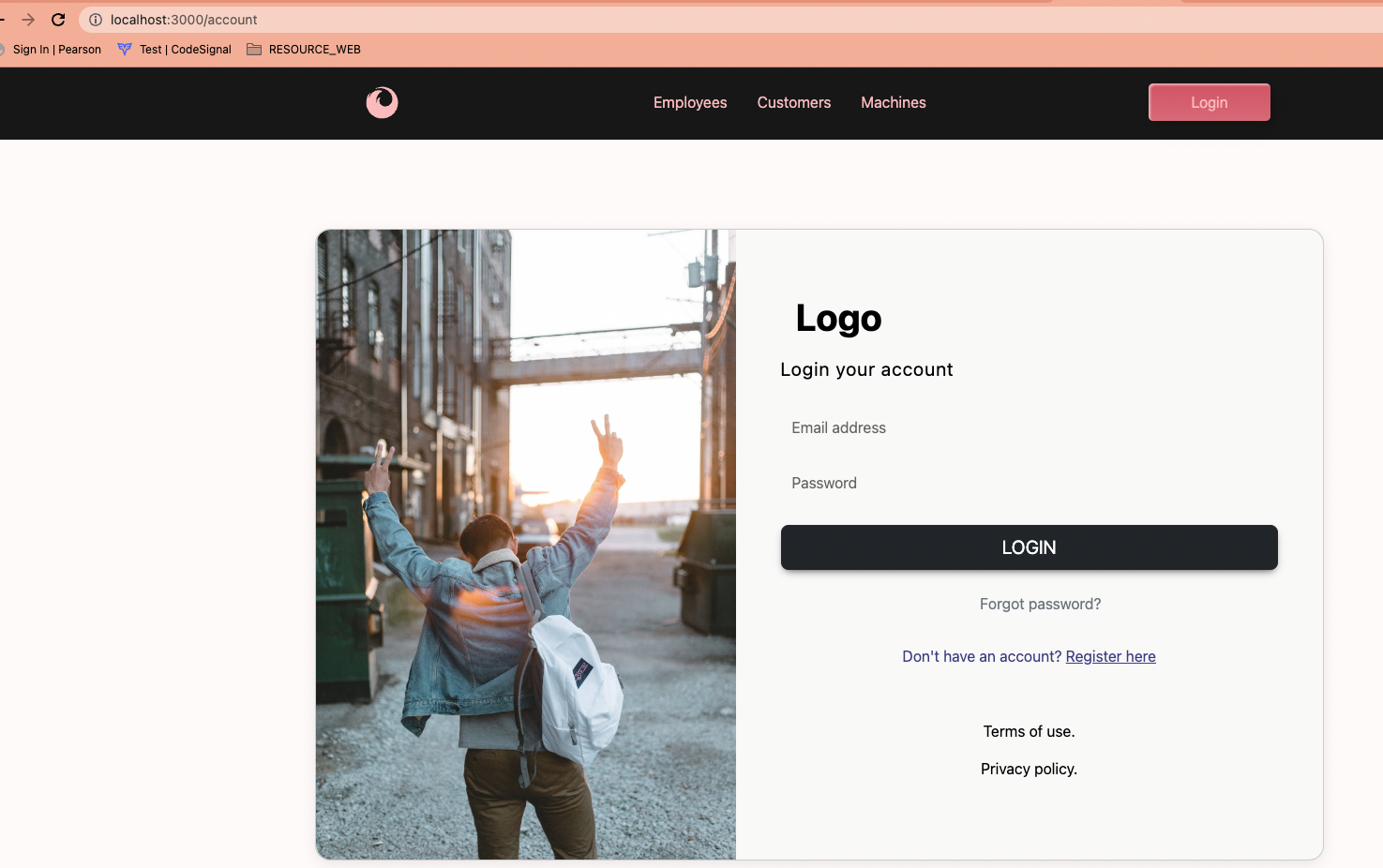


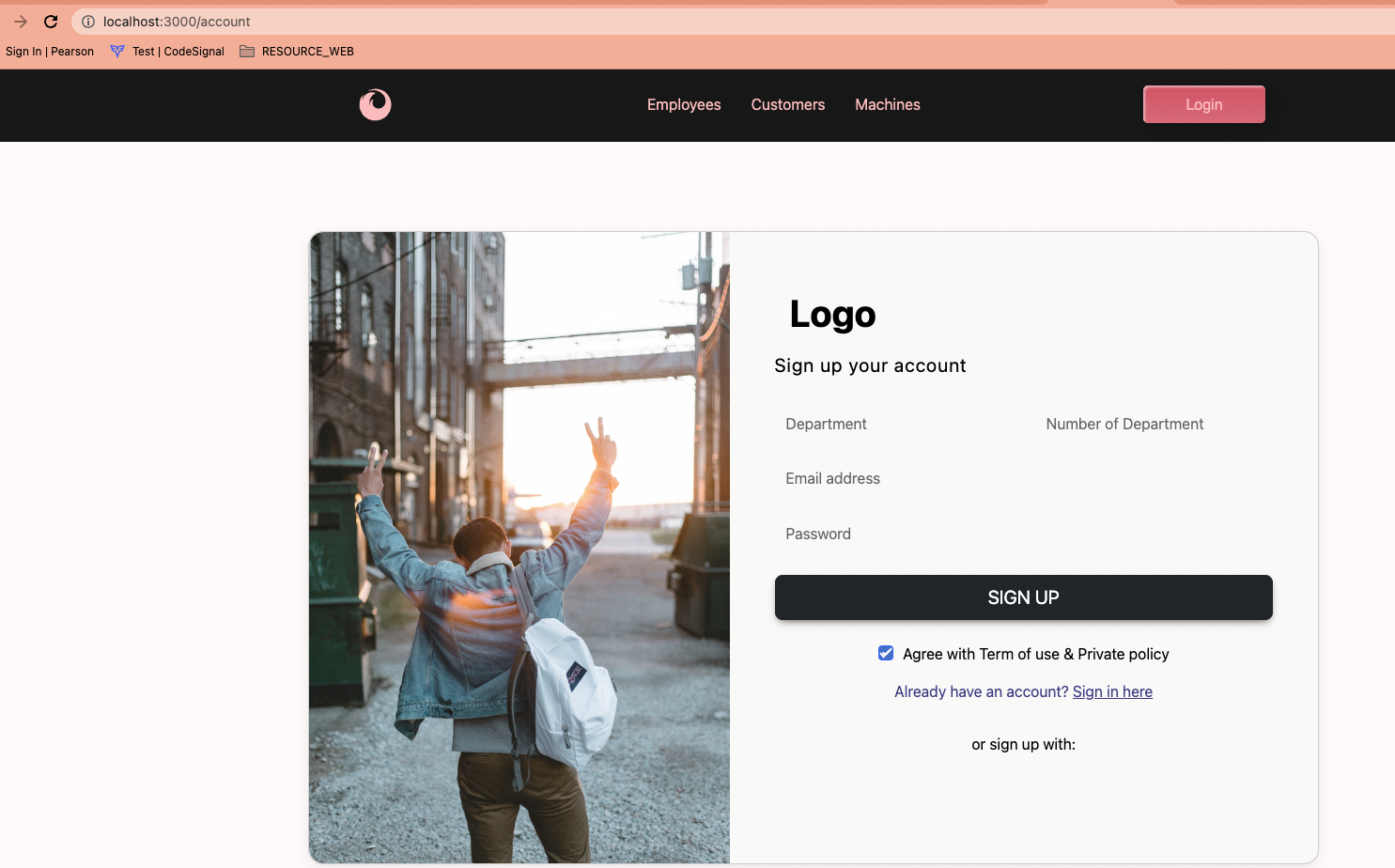
## **User interface and output**

### ***Main screen:***

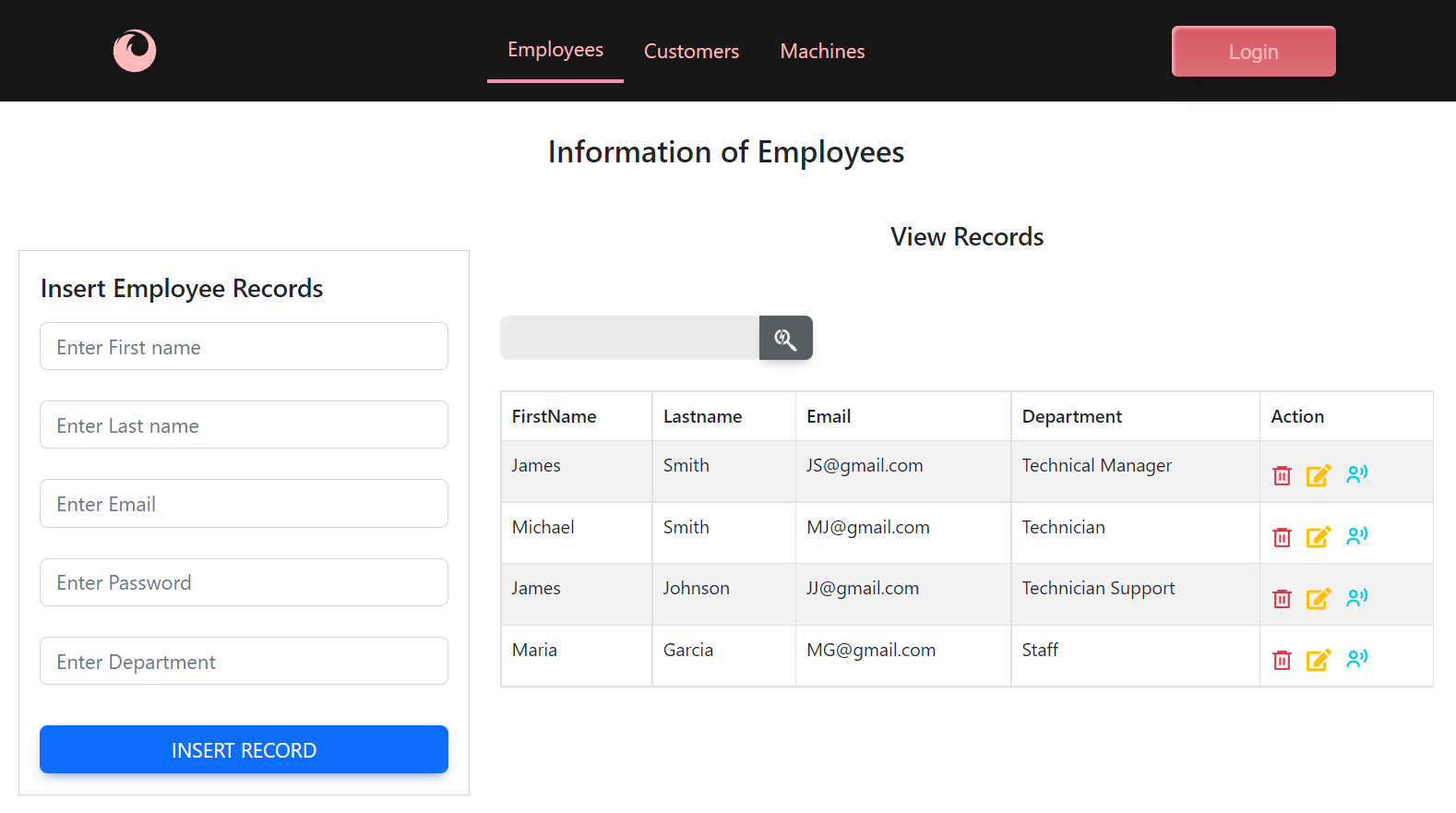
****

### ***Login and Signup:***

****

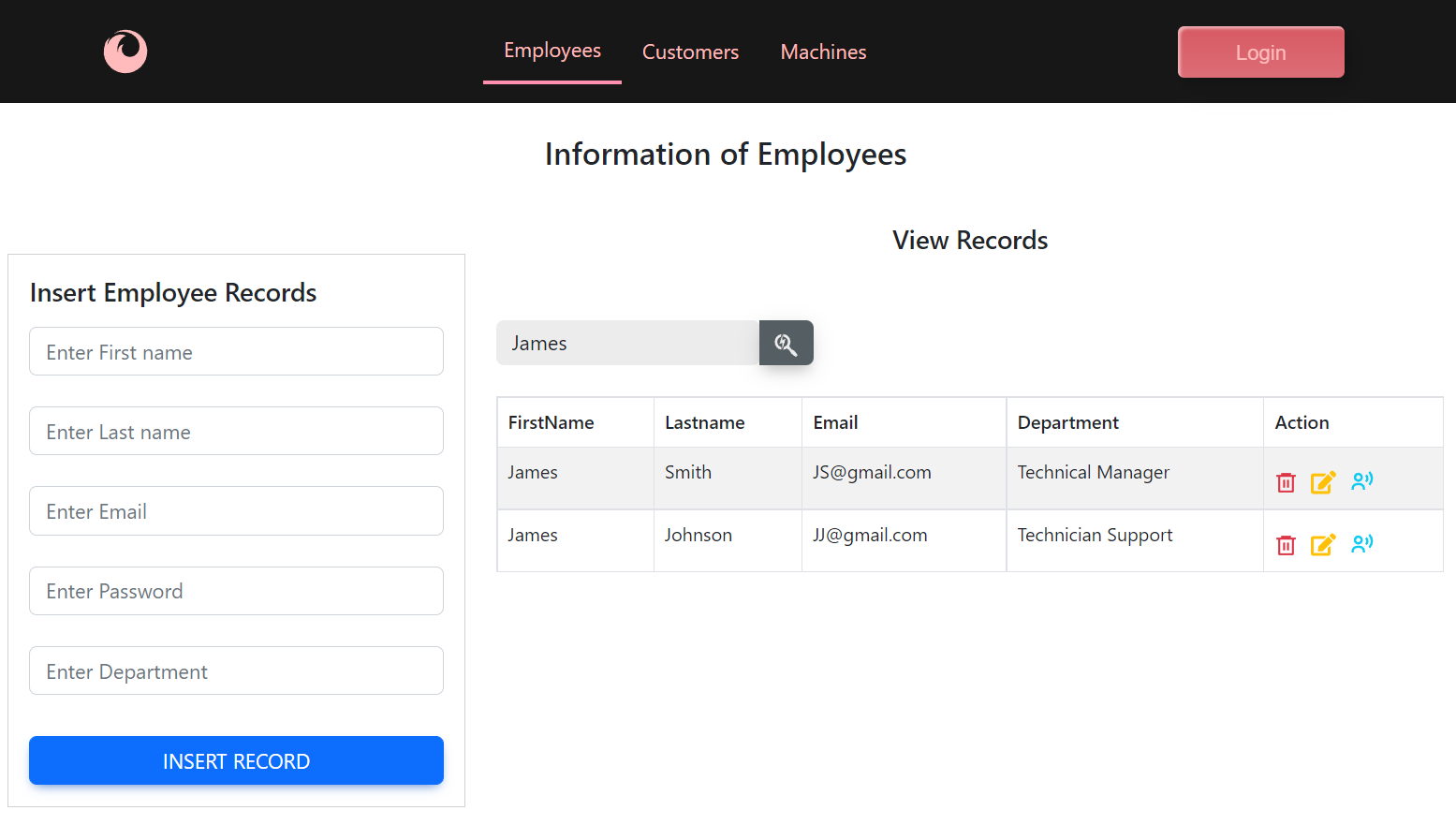
****

### ***Employees list:***

****

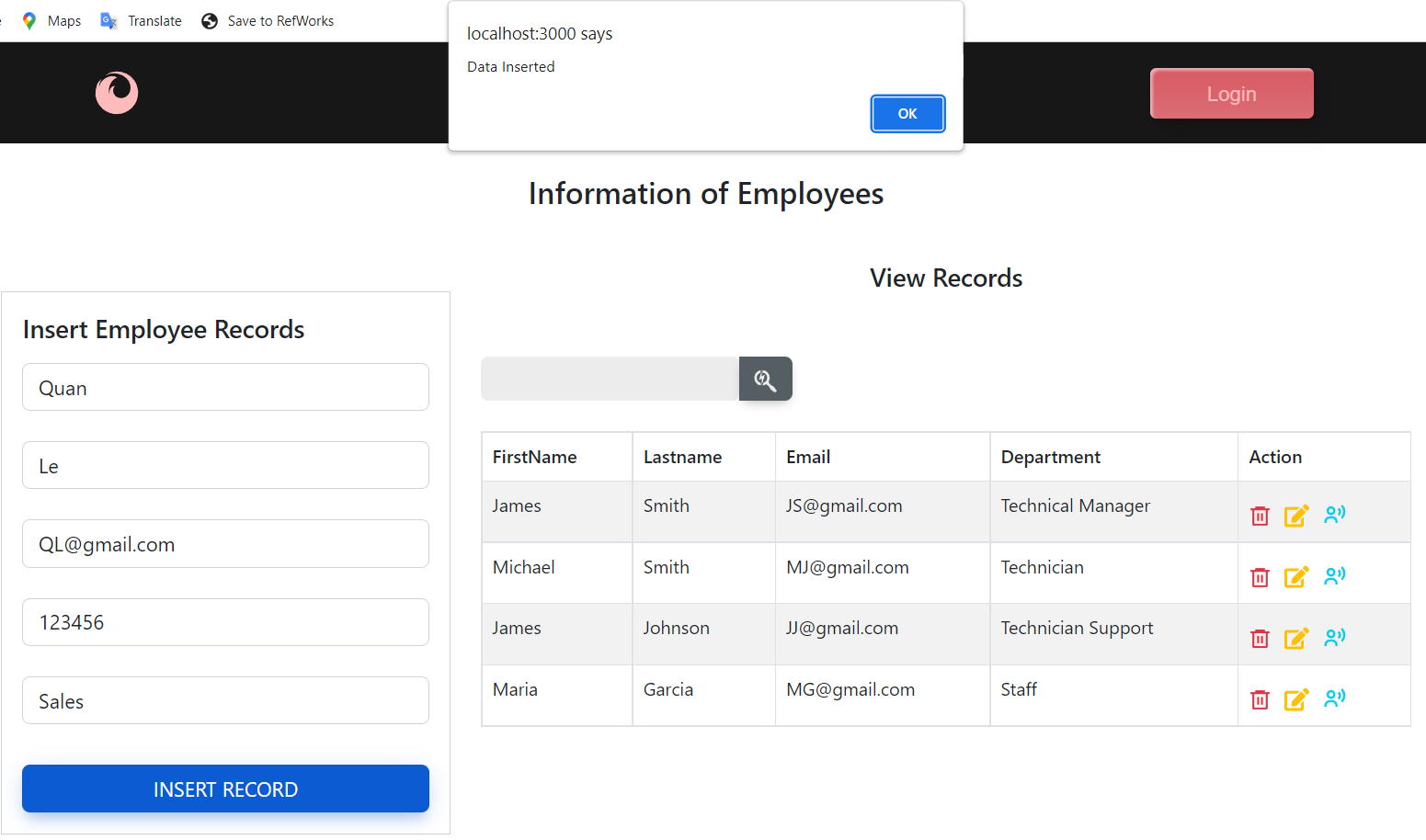
#### Search An Employee

Type “James” in Search box, system will list all the employees with the first name is “James”

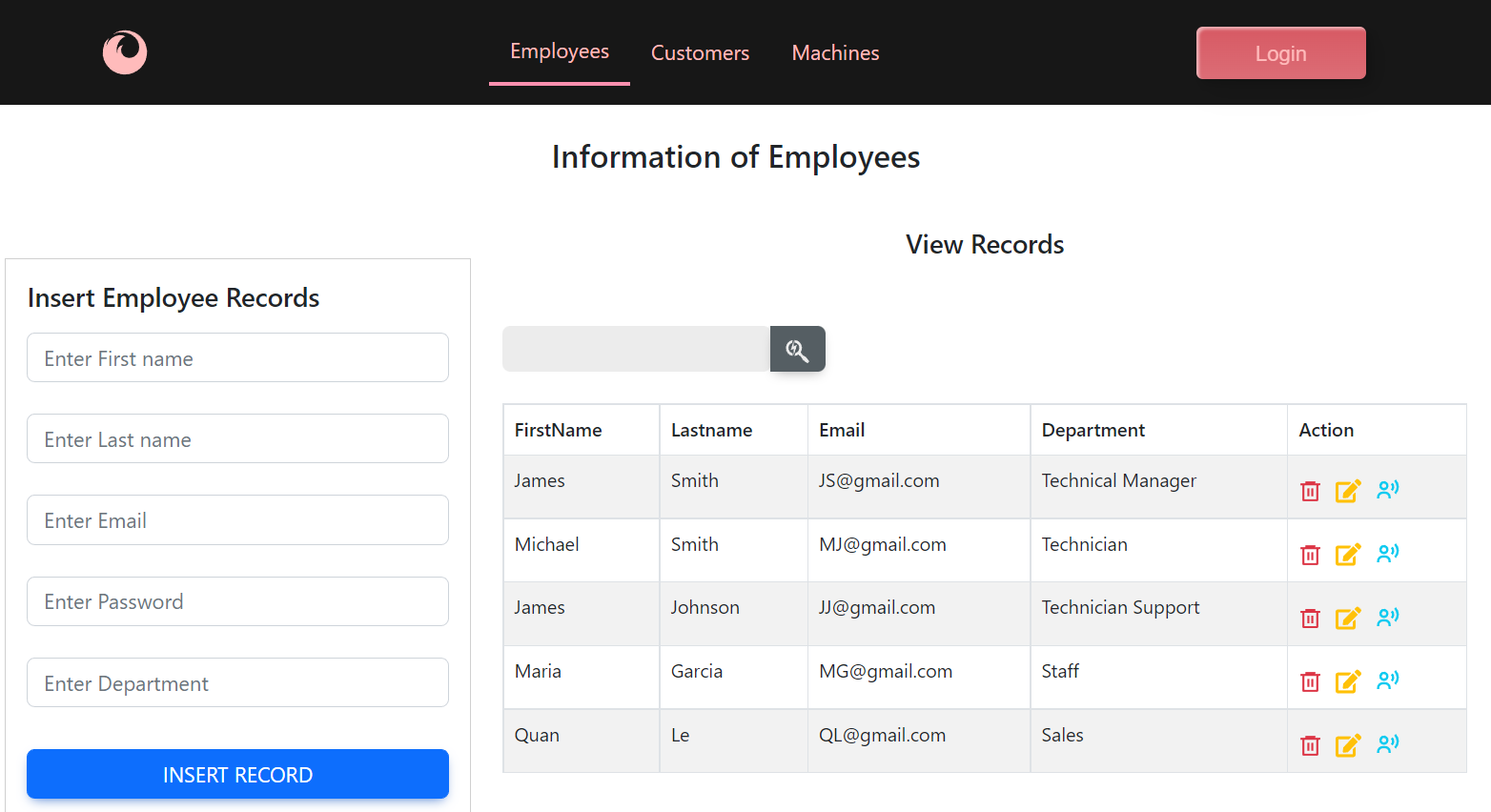
****

#### Insert An Employees:

Type Firstname: “Quan”, Lastname: “Le”, Email: “[QL@gmail.com](mailto:QL@gmail.com)”, Password: “12345”, Department: “Sales” then click “INSERT RECORD” button.

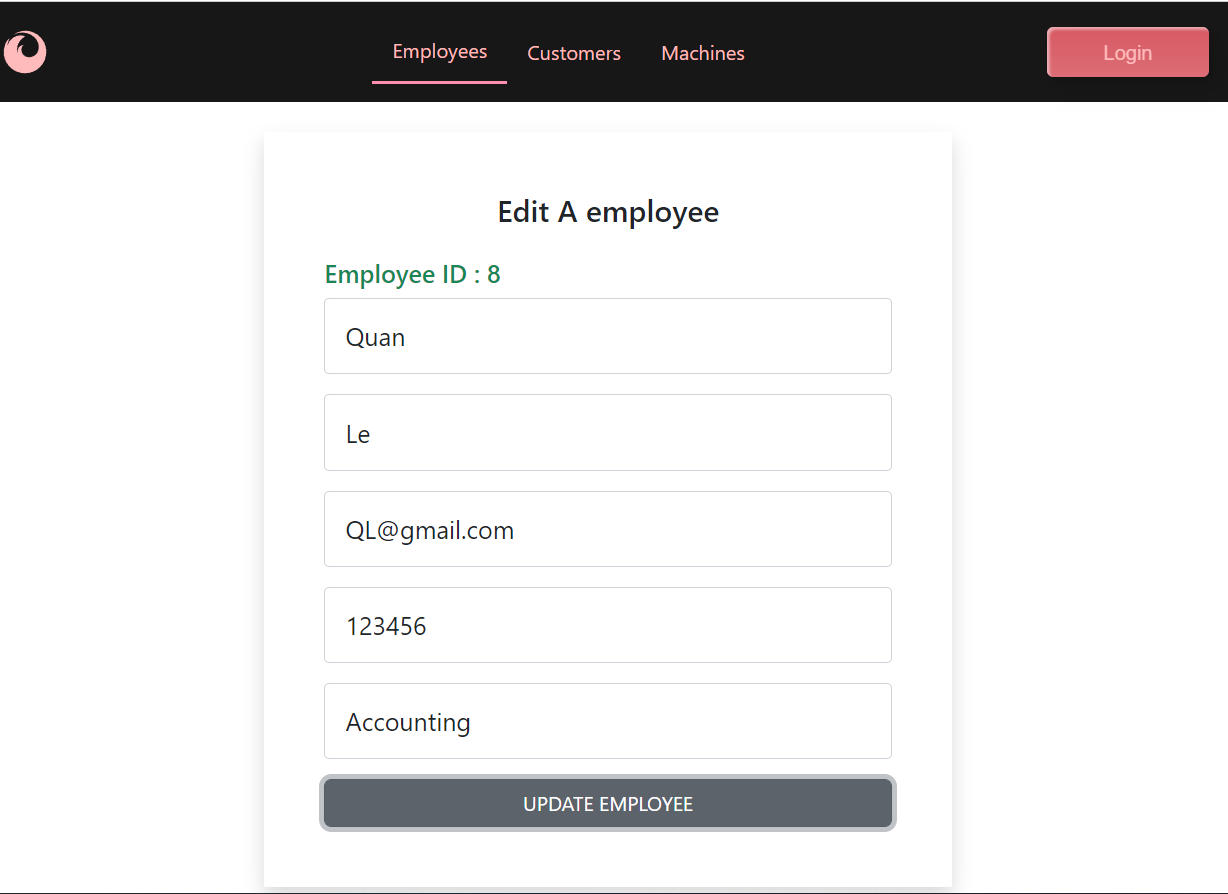
****

New record has been inserted

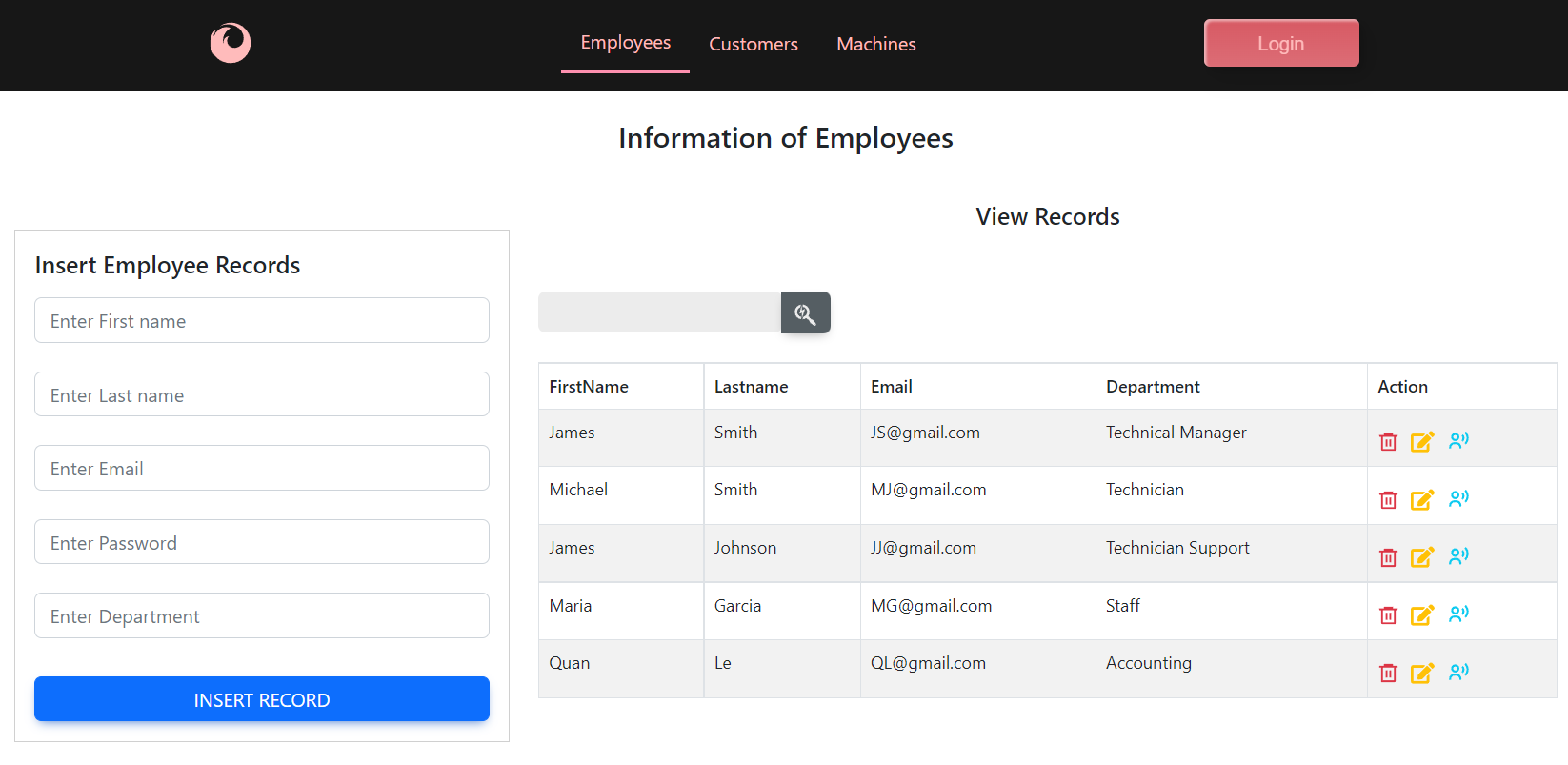


#### Update a employee details:

Change Quan’s department from “Sales” to “Accounting” by clicking on  edit icon, then click on “UPDATE EMPLOYEE”

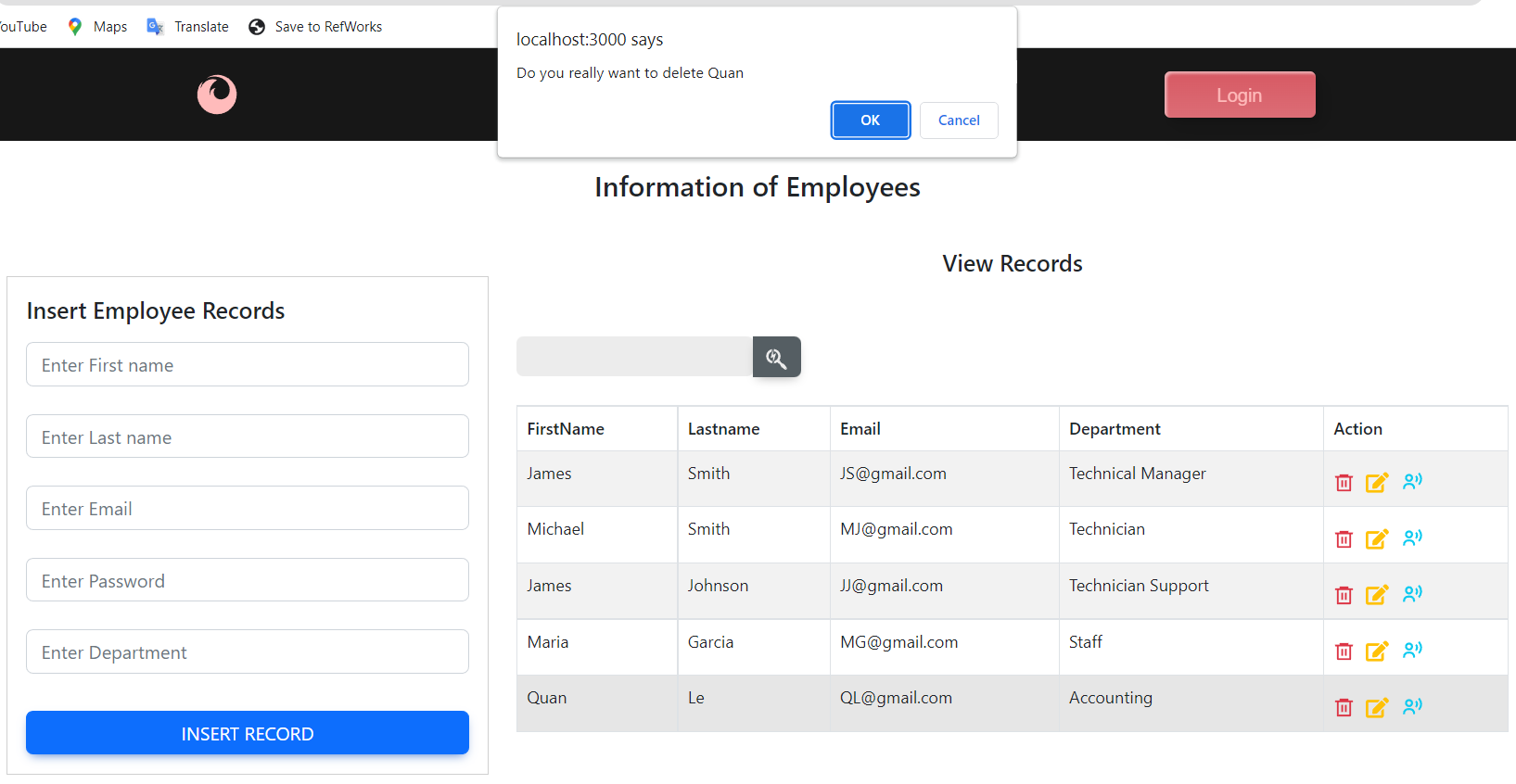


Quan’s department has been changed to “Accounting”

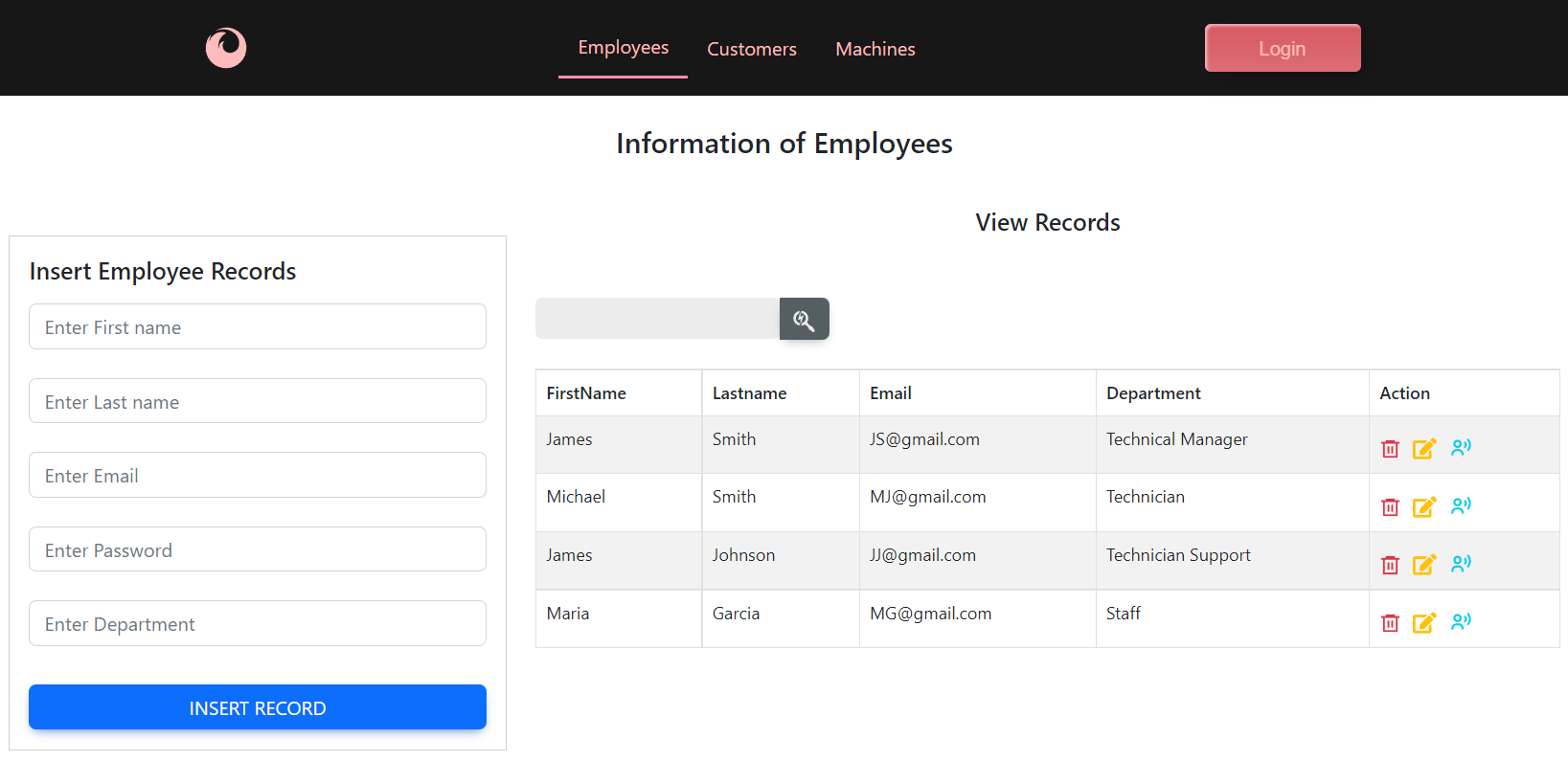


#### Delete an employee:

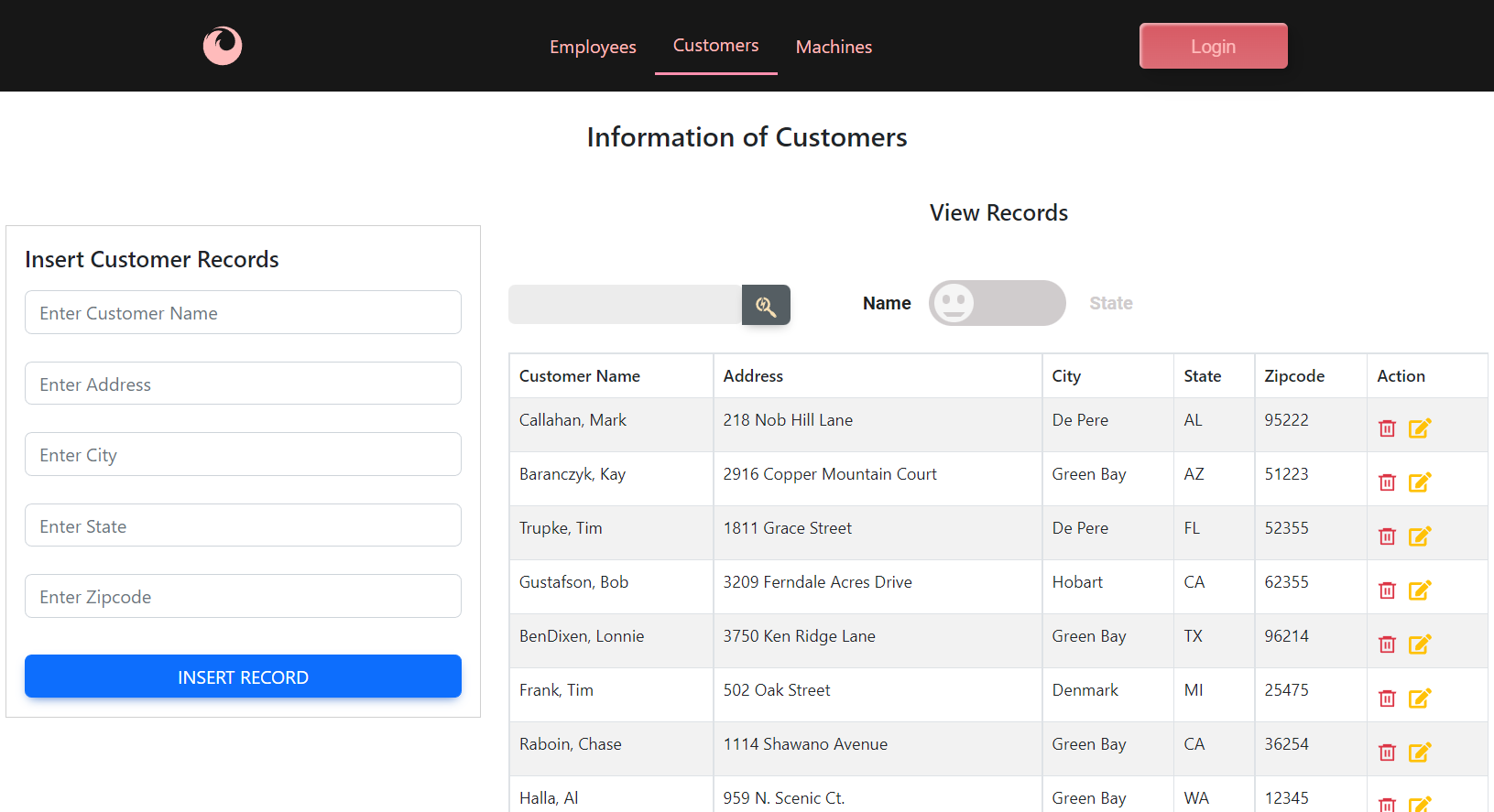
Click on  icon at Quan’s row, then click on OK from the confirmation dialog.



Quan’s row has been deleted.

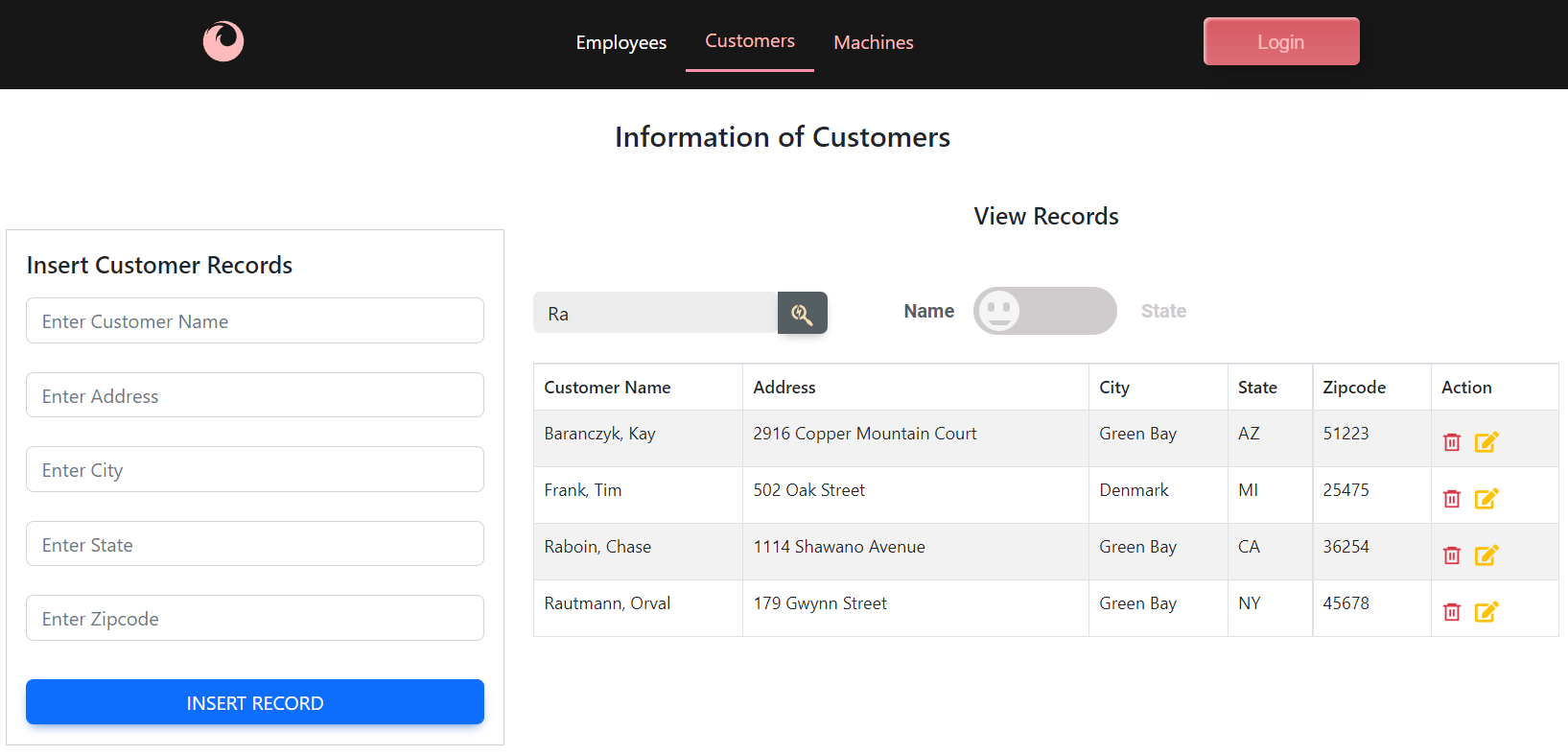
****

### ***Customer List***

****

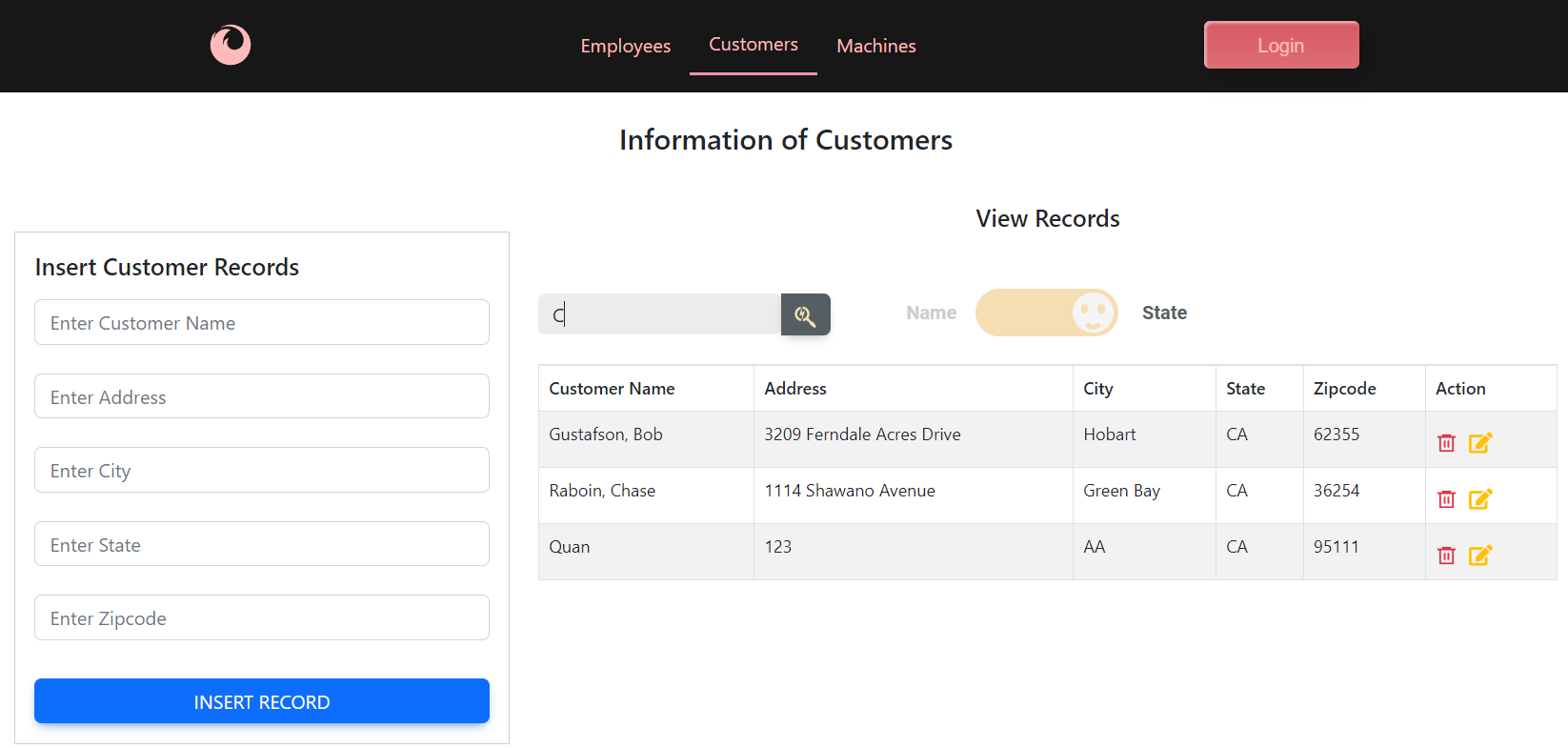
#### Search A Customer

* **search by Name:**Click on “Name” on the left switch button  , then type any part of the value of the Customer name column in the search box. “Ra” matches Ba**ra**nczy, F**ra**nk, **Ra**boin and **Ra**utnann

****

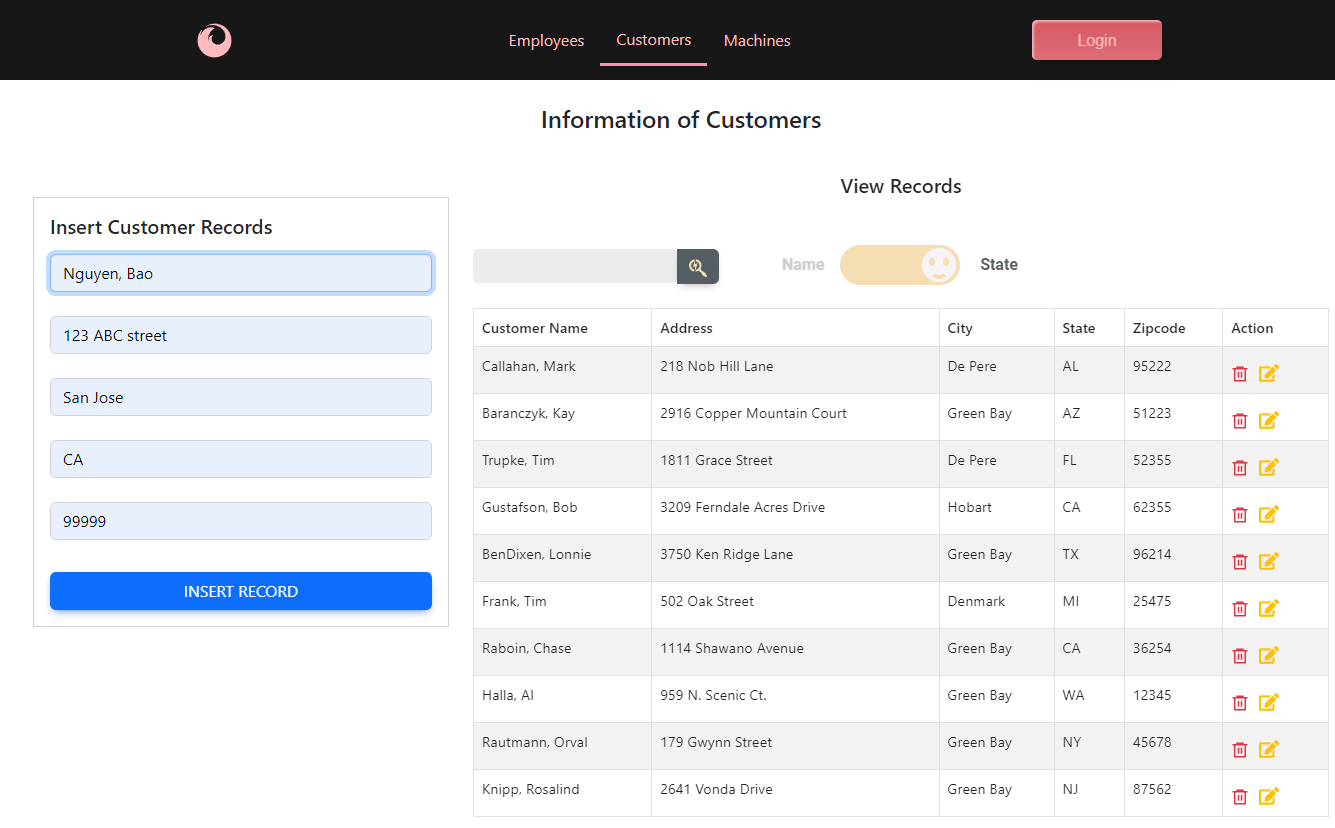
* **search by State:**

Click on “State” on the right switch button , then type any part of the value of the State column in the search box. “C” matches “CA”

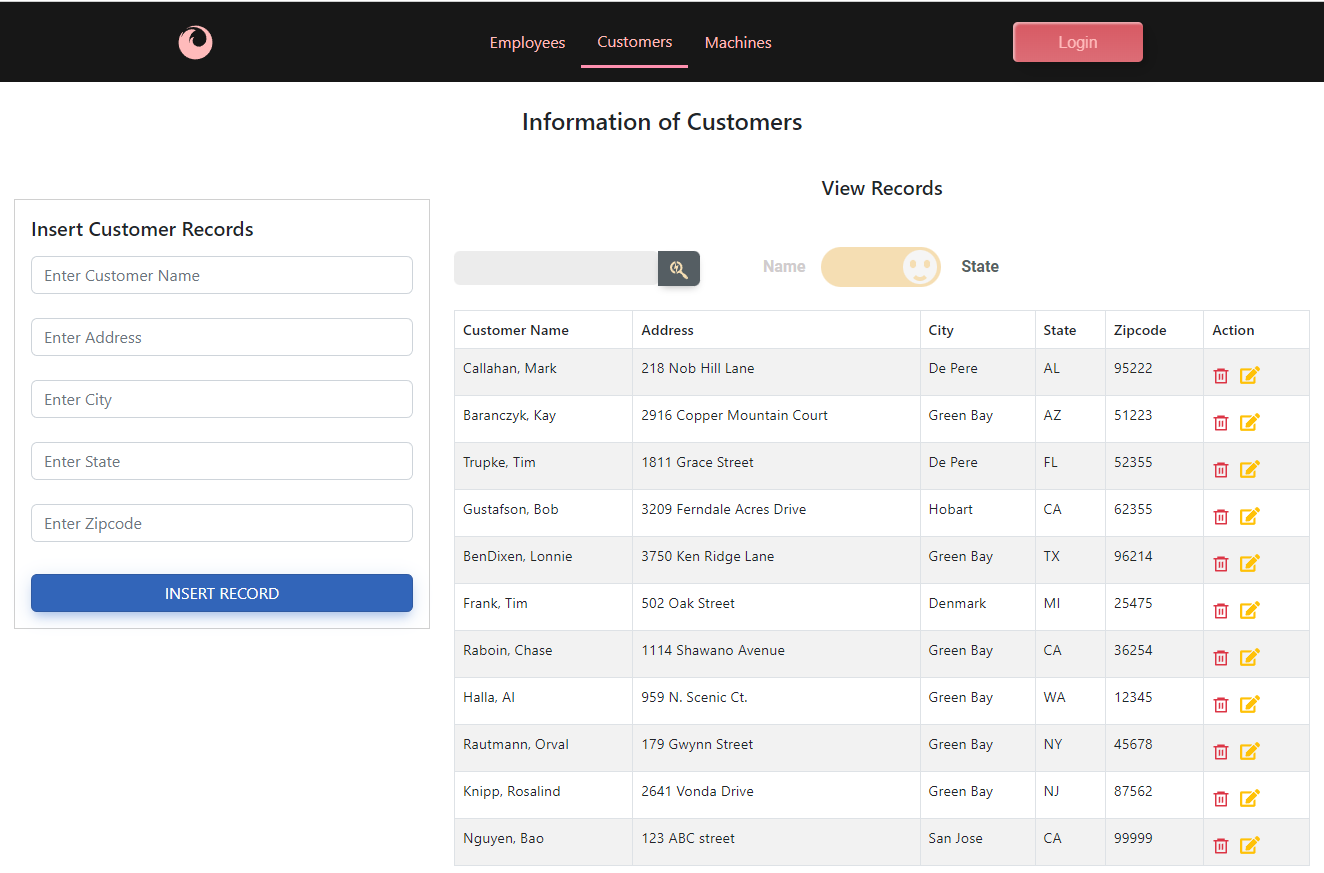
****

#### Insert A Customer

same as inserting Employee above

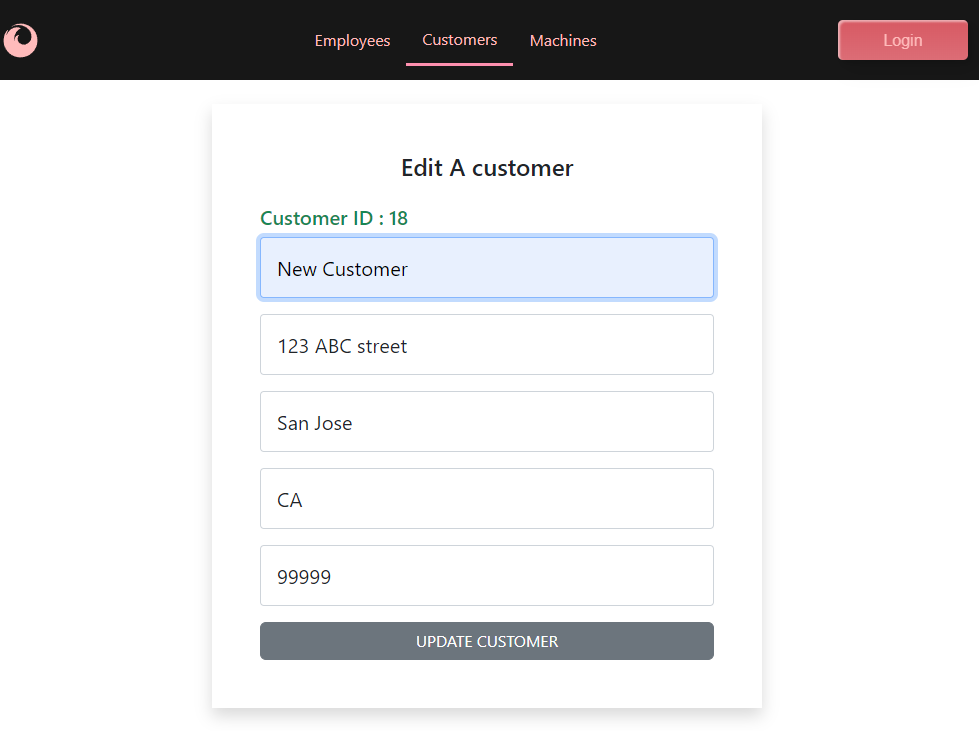
****

New customer has been inserted.

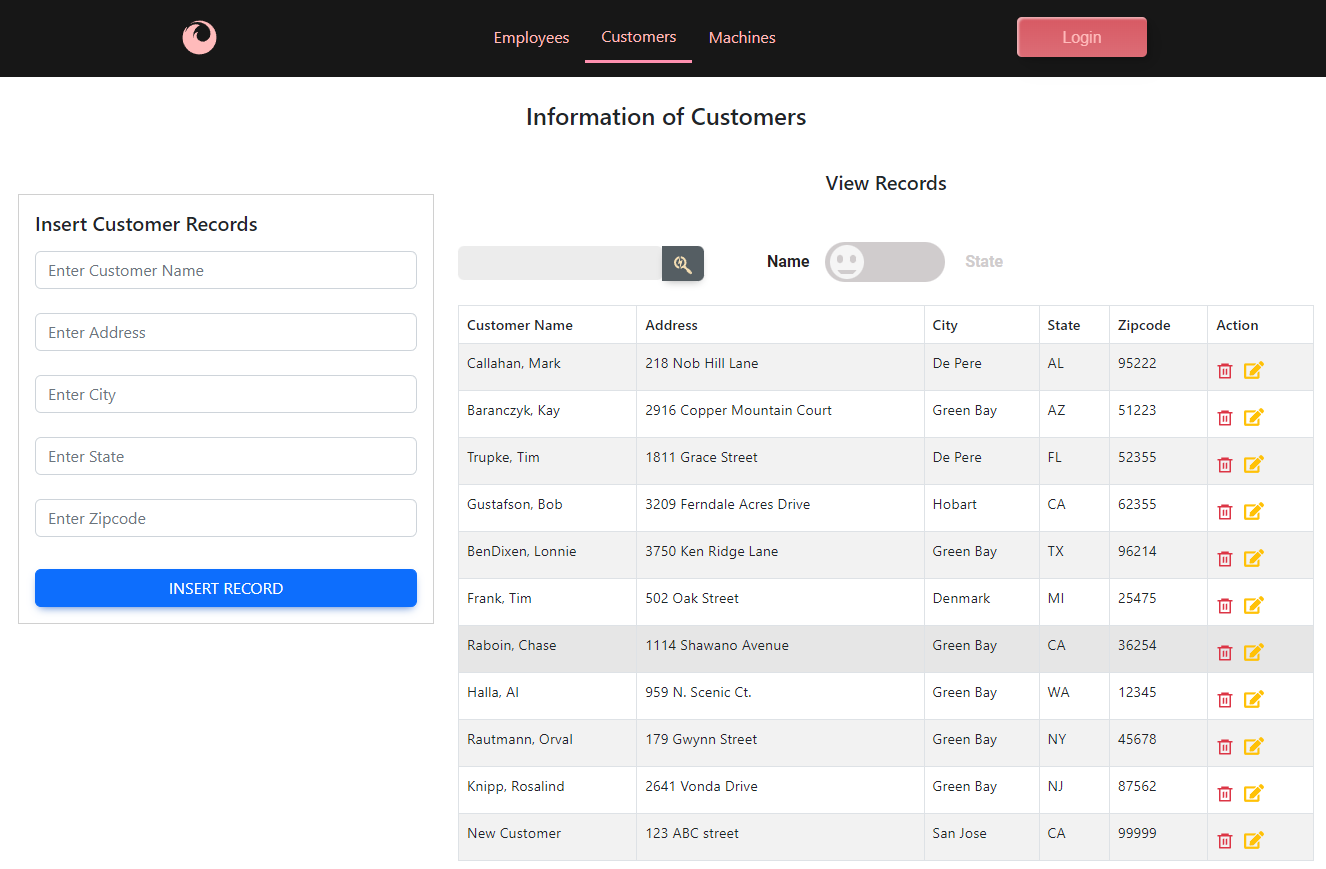
****

#### Update **A Customer**

Same as updating Employee above: Change “Nguyen, Bao” to “New Customer”

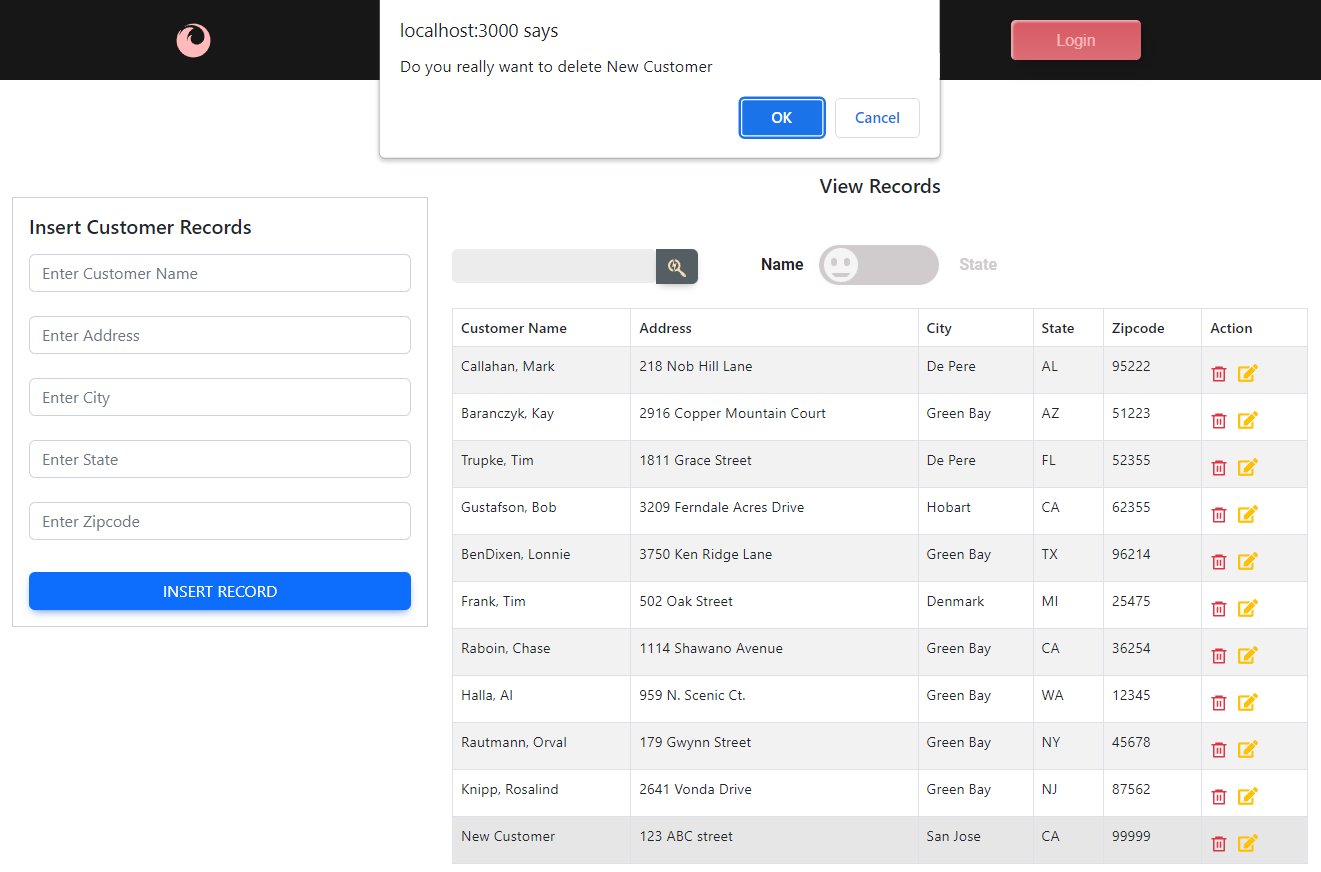


Nguyen, Bao has been changed to New Customer

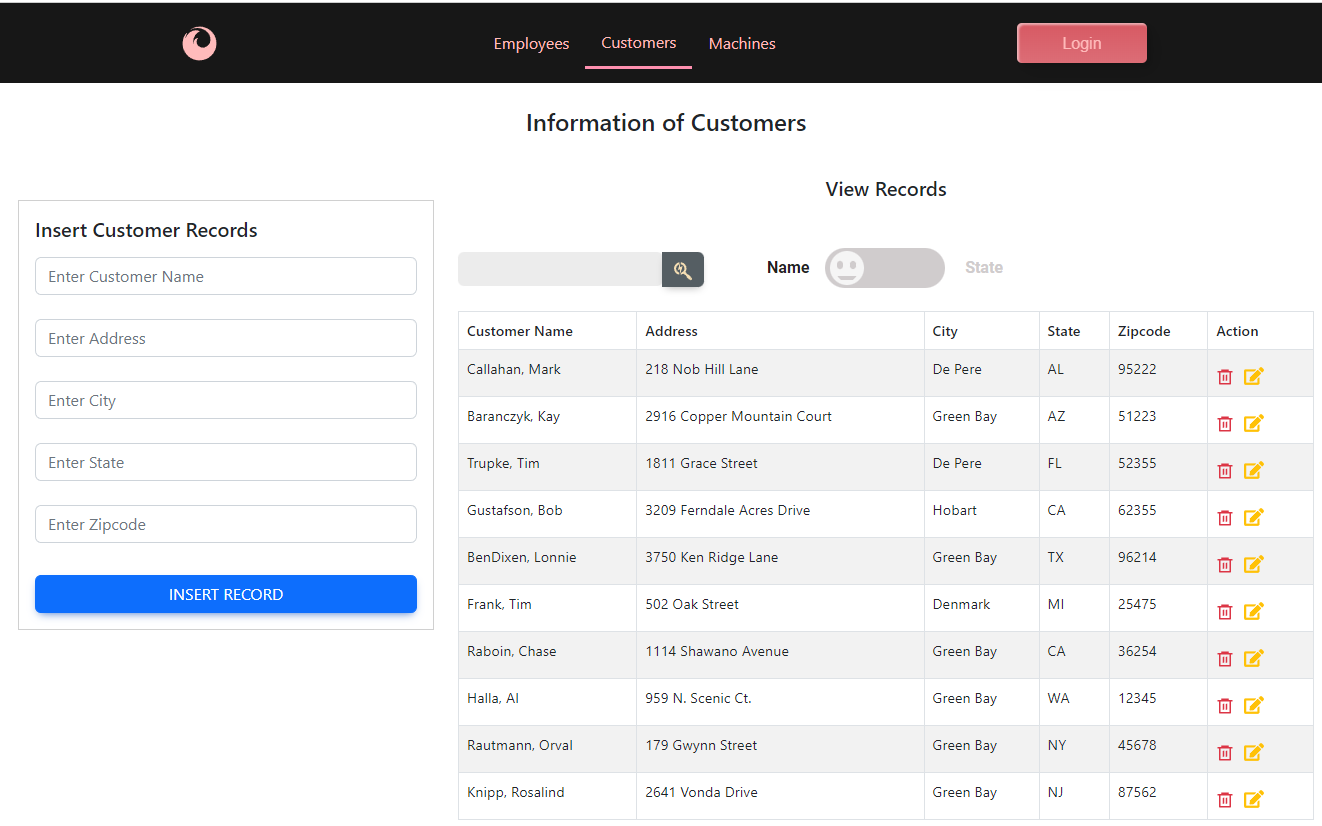


#### Delete A Customer

Same as delete Employee: Delete “New Customer” row.

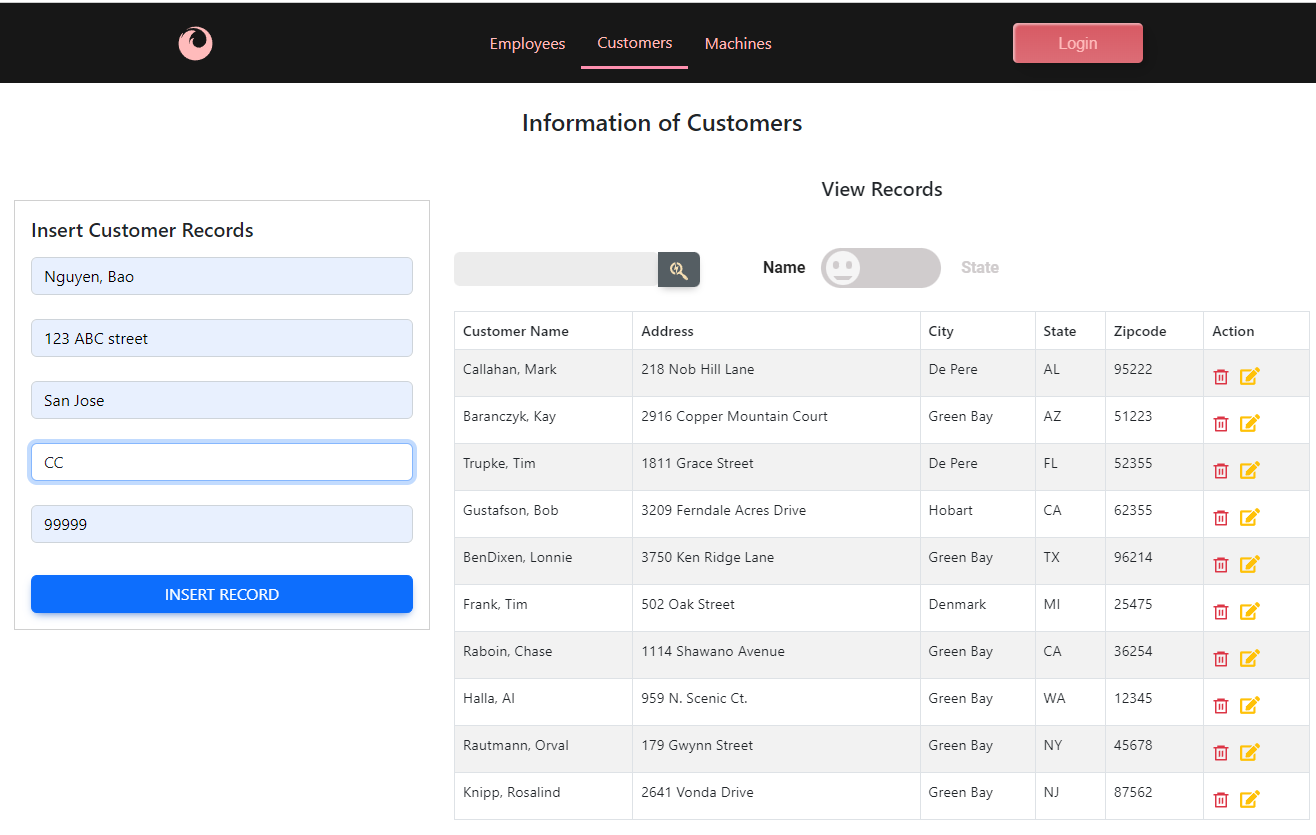
****

“New customer” row has been deleted.

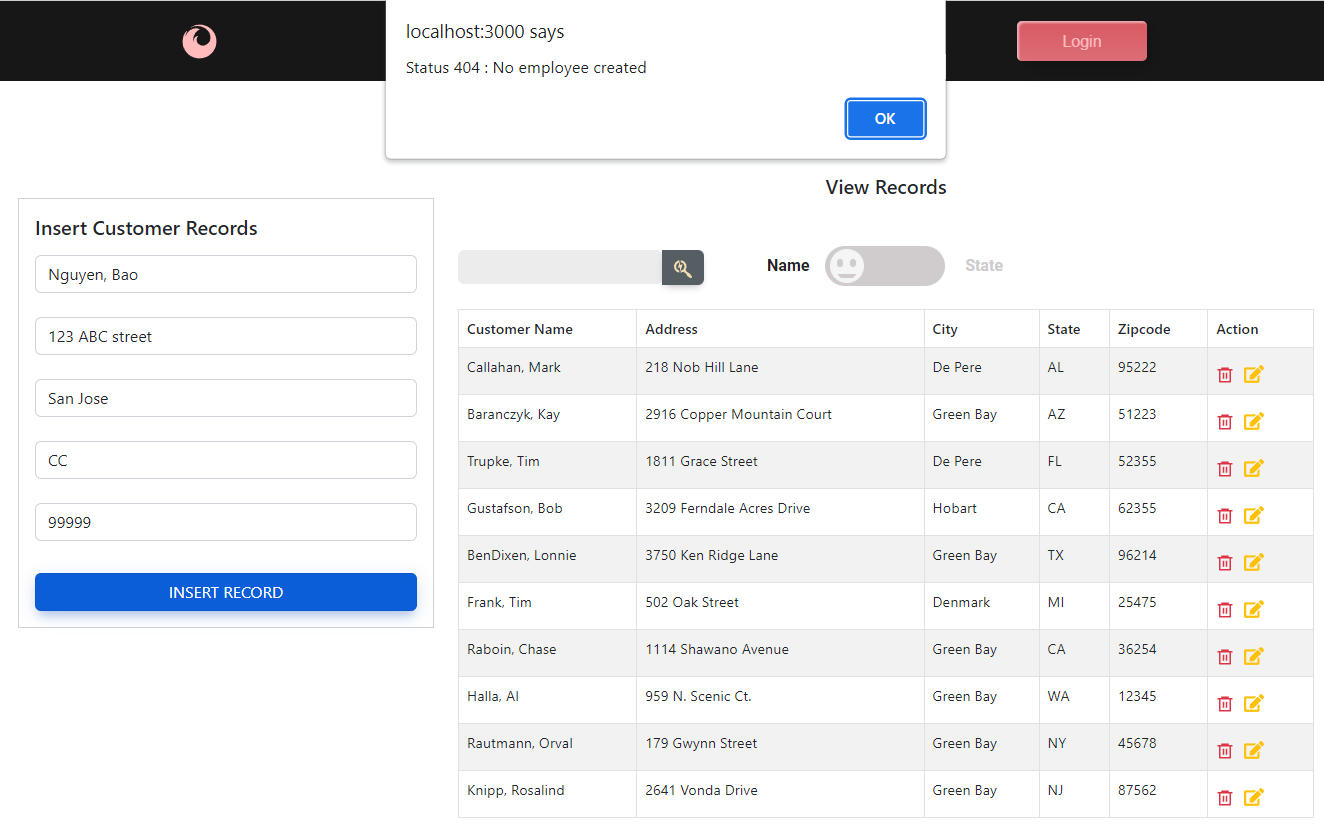


#### Referential constraint

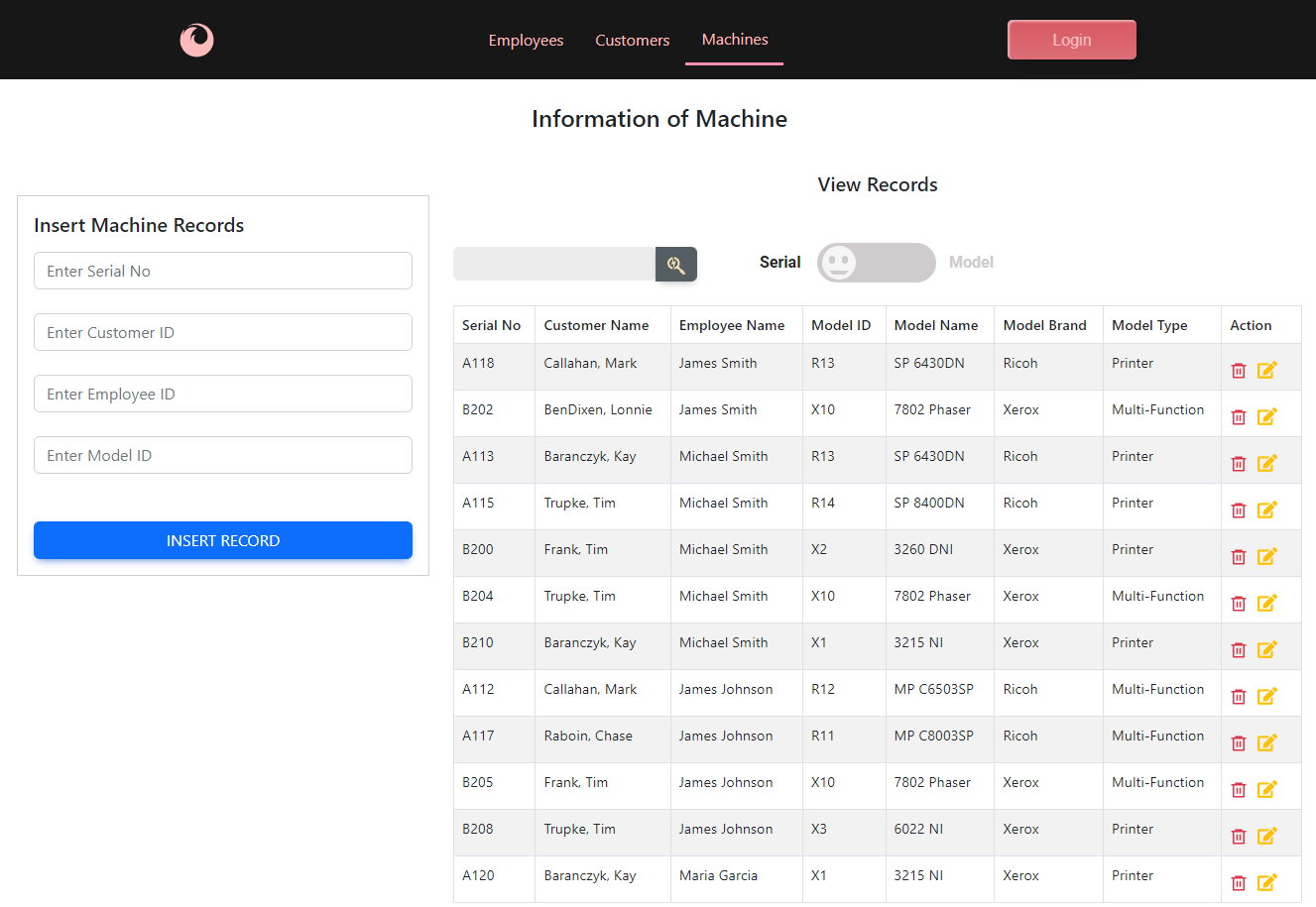
There is a referential constraint with the State table.  
Insert a new customer with a wrong State, for example “CC”



Raising an error

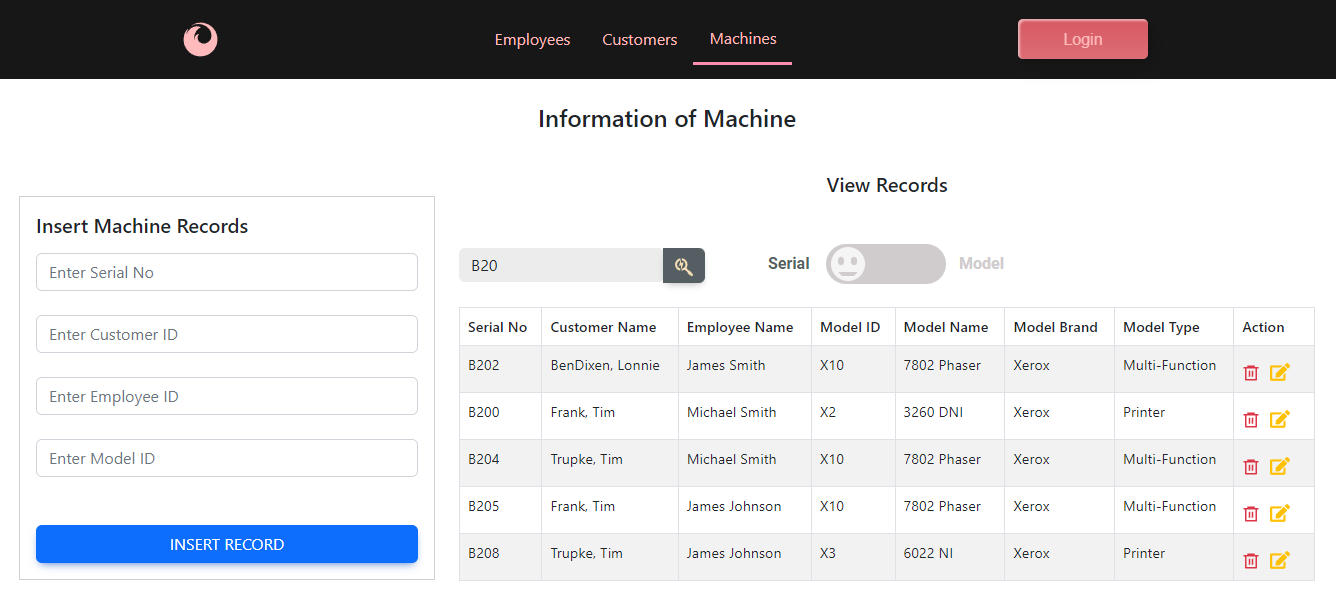


### ***Machine list***

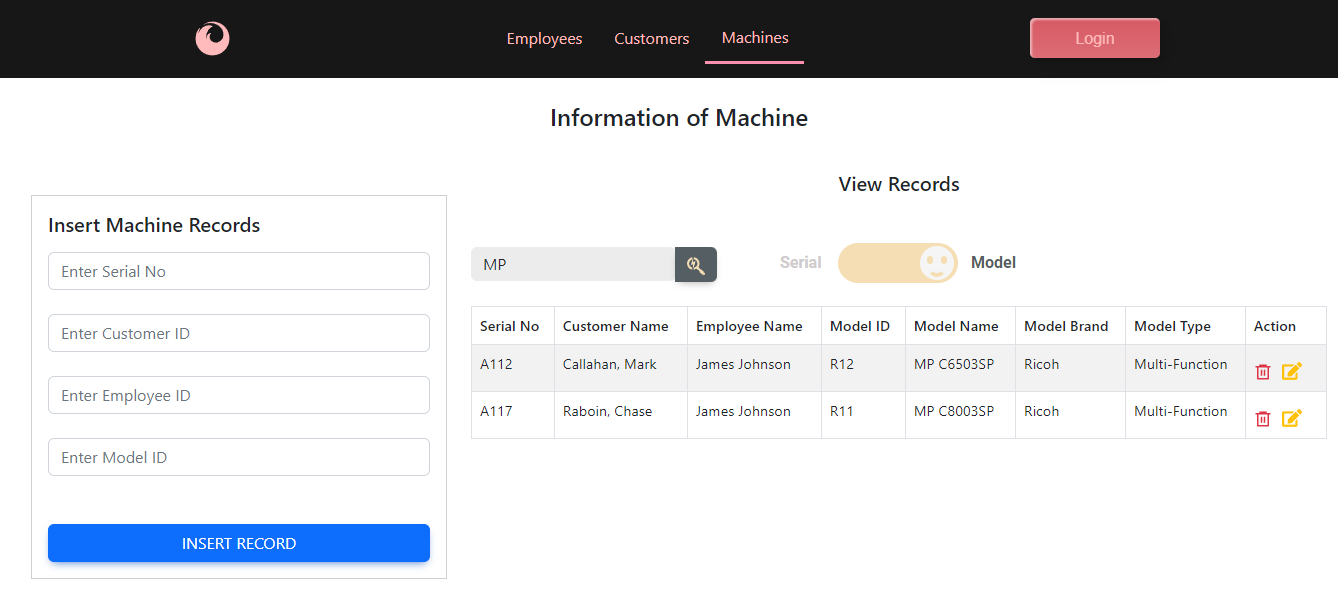
Join four tables: Machines, Models, Employees and Customers.  
****

#### Search A Machine

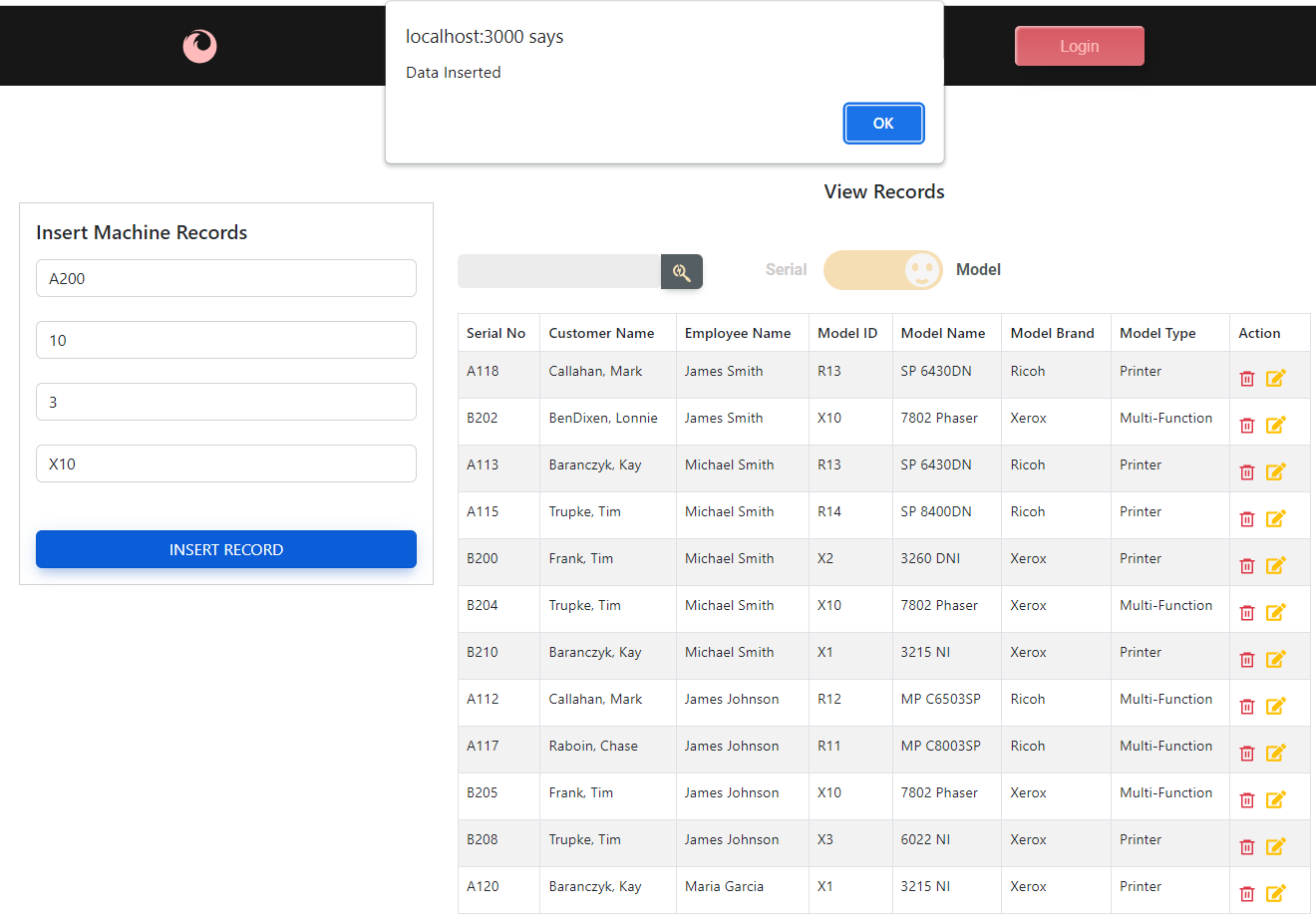
* **Search by SerialNo:** Matches “B20”

****

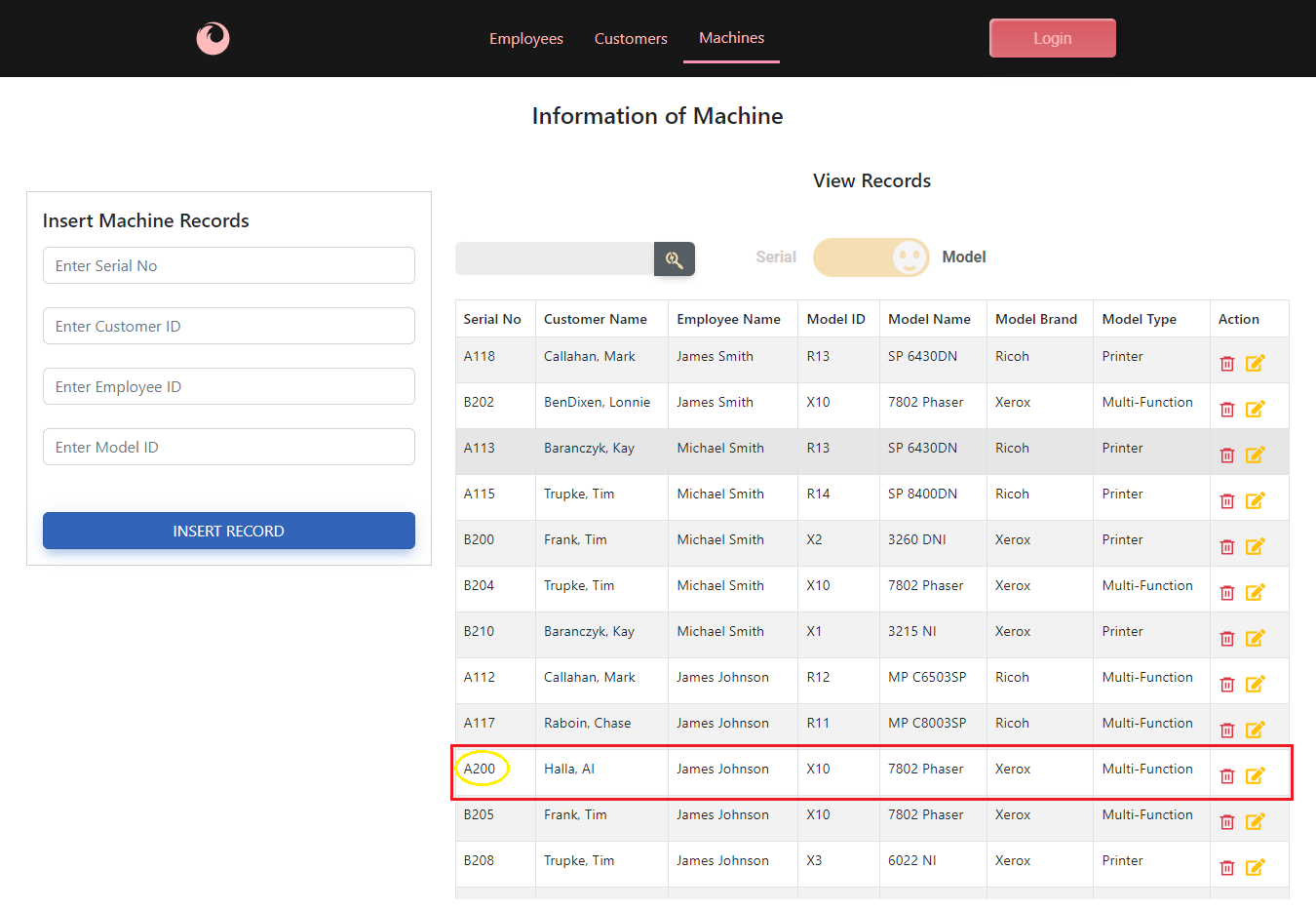
* **search by Model:** matches “MP”

****

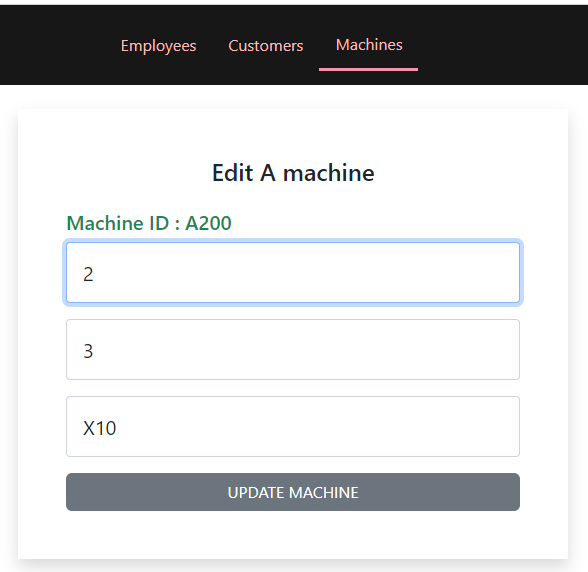
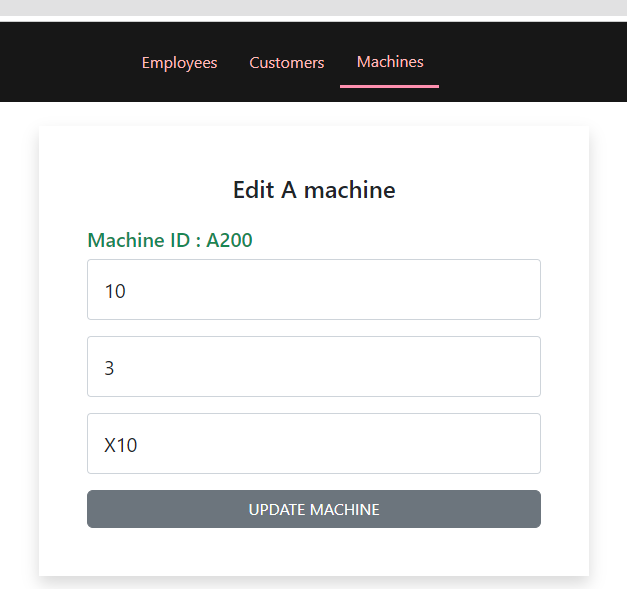
#### Insert A Machine

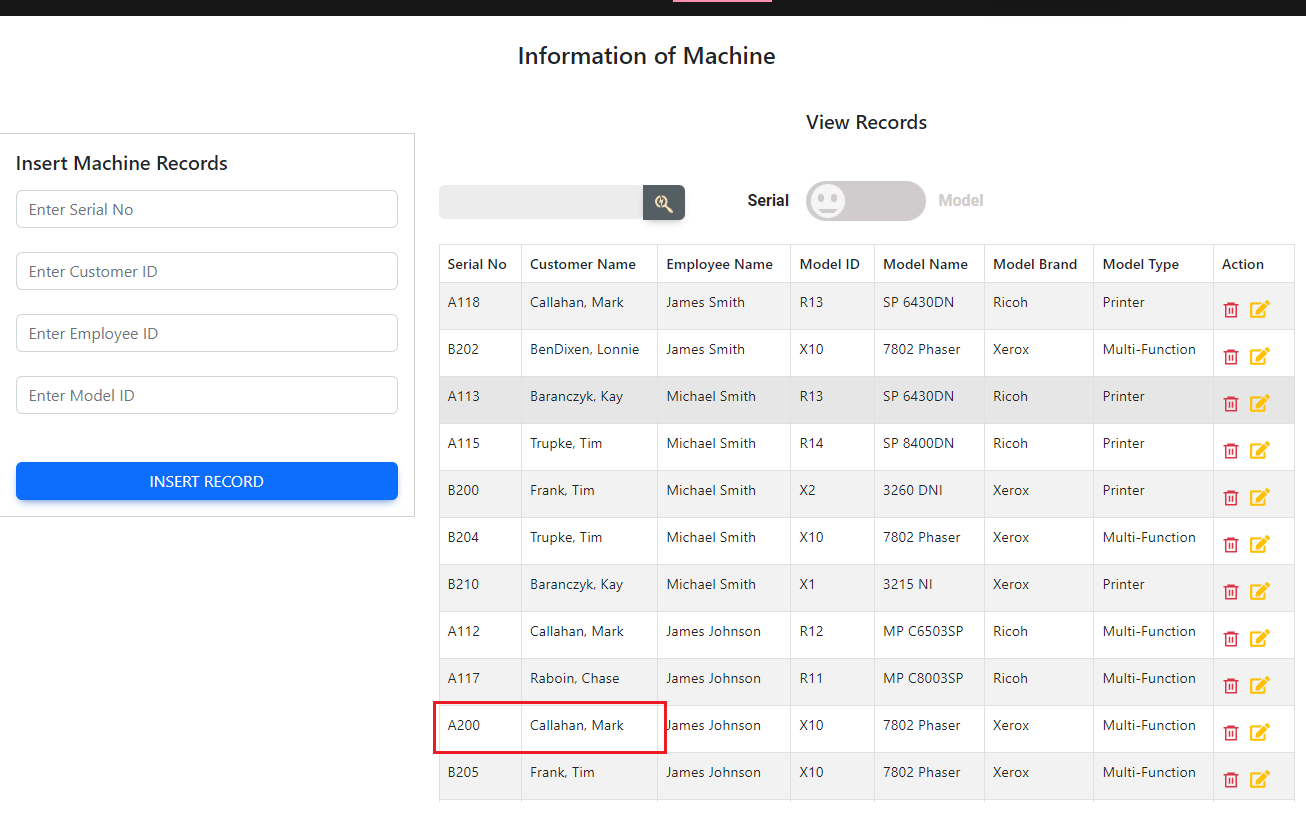


New machine has been inserted



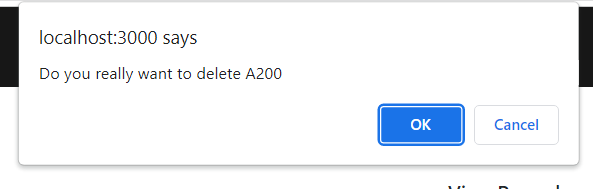
#### Update A Machine:

Change Machine SN: A200 from customer ID 10 ( Halla, Al) to customer ID 2 (Callahan, Mark)

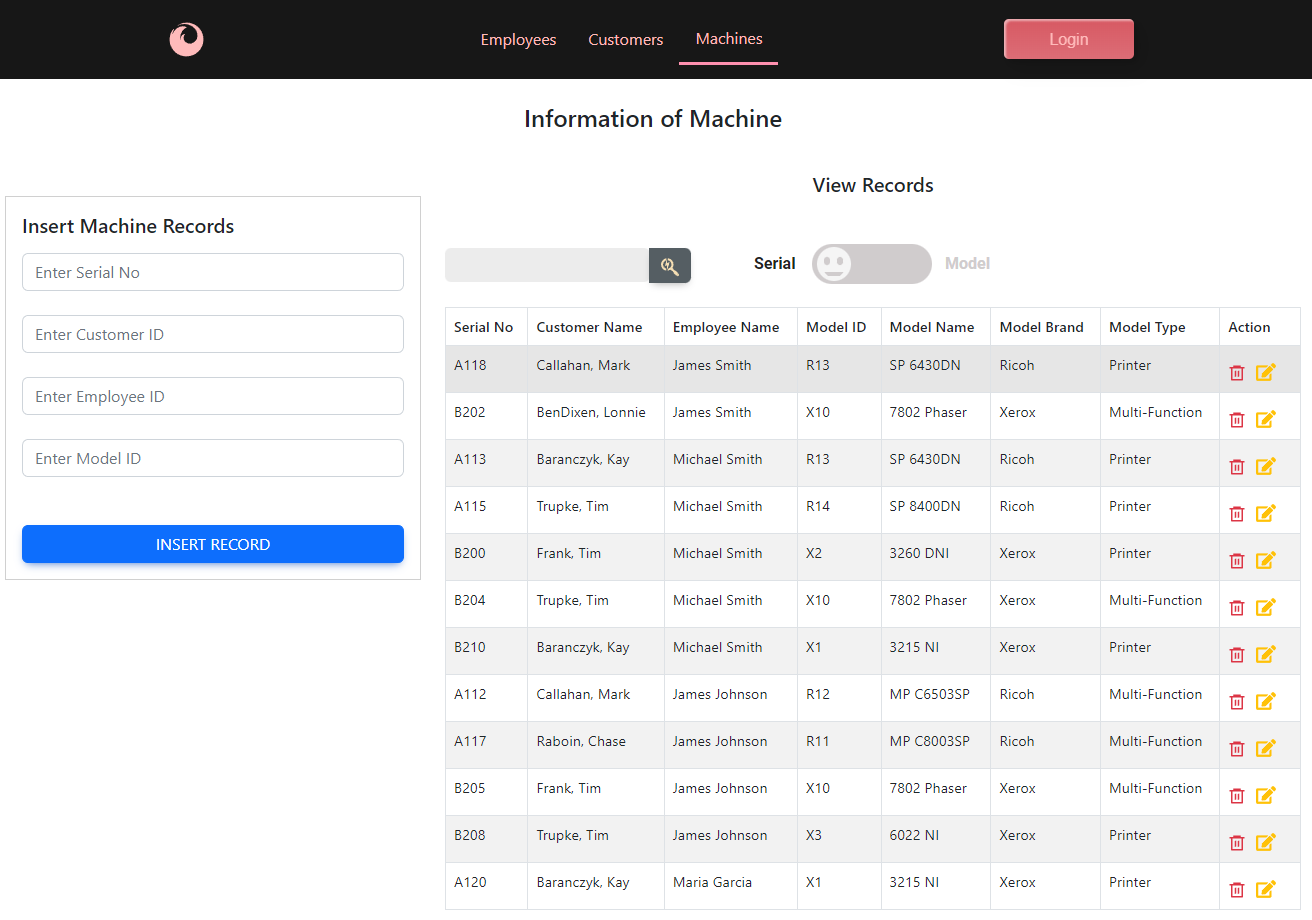


#### Delete A Machine:

Same as delete Employee: Delete “A200” row.

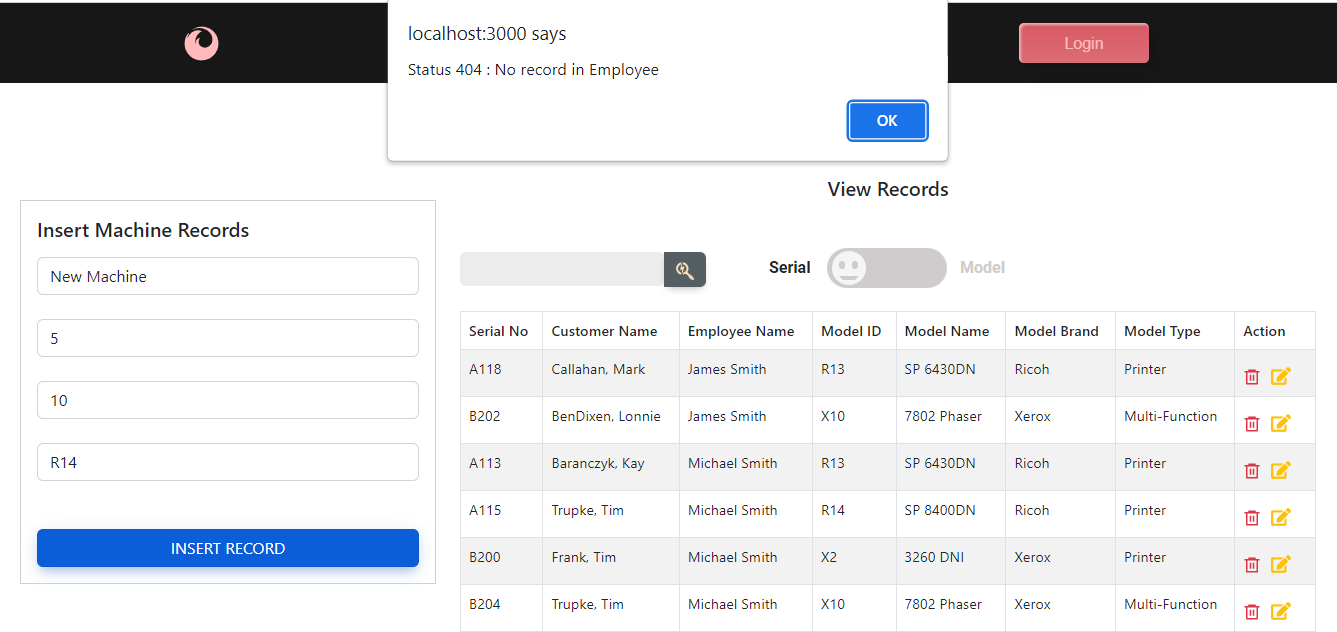


Row A200 has been deleted



#### Referential constraint

There are referential constraints with the Models, Customers and Employees tables.  
Insert a new machine with a wrong Employees, for example “10”



* **ON DELETE CASCADE:**

If deleting a customer, for example Customer ID 3 (Baranczyk, Kay), all her three machines are deleted, too.

# VII. CONCLUSION

The database of this project is expandable to many tables and more features such as storage tables of machine failure history, various service contracts of customer machines. The tables have a structure of 3rd normalization with strict relations and referential constraints contained in the MySQL database. This project is just the beginning of a software engineering process that aims to provide an efficient service and customer management tool for office equipment sales and service companies.

# VIII. APPENDIX

Github: <https://github.com/QuanLew/cs157A/tree/QLBranch>

Link: In Progress

Reference:

* <https://getbootstrap.com/docs/5.2/getting-started/introduction/>
* https://mdbootstrap.com/docs/react/