Boutique Electronics

Rex Fagin

Timur Ozdemirali Guner

Website URL: http://flip2.engr.oregonstate.edu:7785/

Executive Summary

From the creation of our first draft, we received feedback to change all of the primary keys to not be limited (ex: INT instead of INT(10)). We also made sure to change our attribute syntax to all lowercase to avoid confusing them with table names. In our outline we also shorten words like ‘Primary Key’ to PK at the graders suggestion. In addition we wrote more specific explanations of which values can be null and why. Peer revision comments led us to make a variety of changes to our attribute types, such as deciding to use CHAR in instances where we knew each entry had to conform to an exact number of characters (ex: two letters for state). We converted several INT values to CHAR in places where we would not need to do math with values in the future (‘quantity’ should be INT but ‘zip code’ can just be CHAR), as well as other similar type changes.

The first major design decision we had to review was about normalization. Our database thus far satisfied the first and second normalization forms because each of our attributes were in their lowest atomic forms, and we didn’t have any instances of partial primary key dependencies. However, we had tables that included entities such as ‘state’ and ‘zip code’. In these cases, changing a ‘zip code’ of an employee would mean that state would have to be changed at the same time, so it introduced some transitive dependencies. However our database is aimed at a boutique shop, which would be a small business, so for the amount of customers expected in the database, we decided that it would not be worth increasing the complexity of the database simply to satisfy the third normal form in all tables.

When it came to adding or editing ‘Invoices’, we changed the schema (and ERD) to make ‘Employee ID’ nullable. This preserved the ‘invoice’ by only tying it to ‘Customer ID’, which is mandatory in our project. It also solved an issue with not having the Invoice deleted when an Employee is deleted. At this point we included some information text to summarize what each table was.

For the final steps, we made some changes to our UI and CRUD operations. We fixed negative input for ‘qty’ in ‘Invoice/Electronics’, and fixed the search title shown in the search ‘Customer’ modal. We used the pattern attribute along with REGEX on our inputs to prevent users from entering bad data. We got feedback that the foreign key ids were confusing, so we replaced the ids with relevant descriptions in the Invoices and Invoices/Electronics tables. Our final change we made based on user input, was to make the homepage have more relevant information and a better layout. After all fixes, we added modals that contain information on the table including instructions, layout information, and formats.

Project Outline

Boutique electronics sells $5 million worth of high-end electronic equipment annually. Given that their inventory is stocked with premium items, figuring out which items sell the most, and how many items sell will allow them to invest in products that will be most likely to maximize their profits and minimize purchase of inventory that may not sell. They will be able to add new items and dis-continue items that are not selling well. Keeping track of customer purchasing history will also give them insights into which customers spend the most, and may be used to target promotional coupons for best customers in conjunction with their marketing strategy. They additionally could benefit from logging their employee data, and matching individual employees to invoices to see what they have sold. All of these requirements can be satisfied by a database with a web application to allow them to manage this information in an efficient and effective manner.

Database Outline

Below is an outline of the database design for Boutique Electronics. Each section will follow a standard format for easy analysis:

EXAMPLE:

- **Table Name: description of table**
    - **attribute: attribute type, additional options, nullable, key designations**
    - **Relationship - description of the relationship to other entities**

OUTLINE:

- Customers: records of details of the customers that do business with the store
    - customer_id: INT, auto-generated and auto-incremented, UK (unique key), NOT NULL, PK (primary key)
    - first_Name: VARCHAR(35), NOT NULL
    - last_Name: VARCHAR(35), NOT NULL
    - phone_Number: CHAR(10), UK, NOT NULL
    - email: VARCHAR(50), UK, NOT NULL
    - birthday: DATE(), NULL
    - address: VARCHAR(70), NOT NULL
    - apt_number: VARCHAR(10), NULL
    - city: VARCHAR(50), NOT NULL
    - state: CHAR(2), NOT NULL
    - zip: CHAR(5), NOT NULL

    - Relationship - A 1:M relationship between Customers and Invoices is implemented with CustomerID as foreign key in Invoices.The relationship can be null since a customer can be entered into the database without yet having made any purchases and in that case would not yet have any invoices related to them.

- Invoices: records the transaction or purchase made by the customer
    - invoice_id: INT, auto-generated and auto-incremented, UK, NOT NULL, PK
    - customer_id: INT, FK (foreign key), UK, NOT NULL
    - employee_id: INT, FK from Employee, UK, NULL
    - total_sale: DEC(8,2), NOT NULL
    - Relationship - The invoice has a M:1 relationship in which the CustomerID in invoices maps to the CustomerID in Customer. An invoice must map to 1 and only 1 customer
    - Relationship - The Invoice as a M:1 relation in which the EmployeeID (foreign key) in Invoice maps to the EmployeeID in Employees. Invoice must map to 0 or 1 employees
    - Relationship - has a 1:M relationship to the InvoiceElectronic, which is an intermediate table that links Invoice to Electronics for M:M. The link to InvoiceElectronic is based on the InvoiceID (foreign key) to the Invoice ID in Invoice.

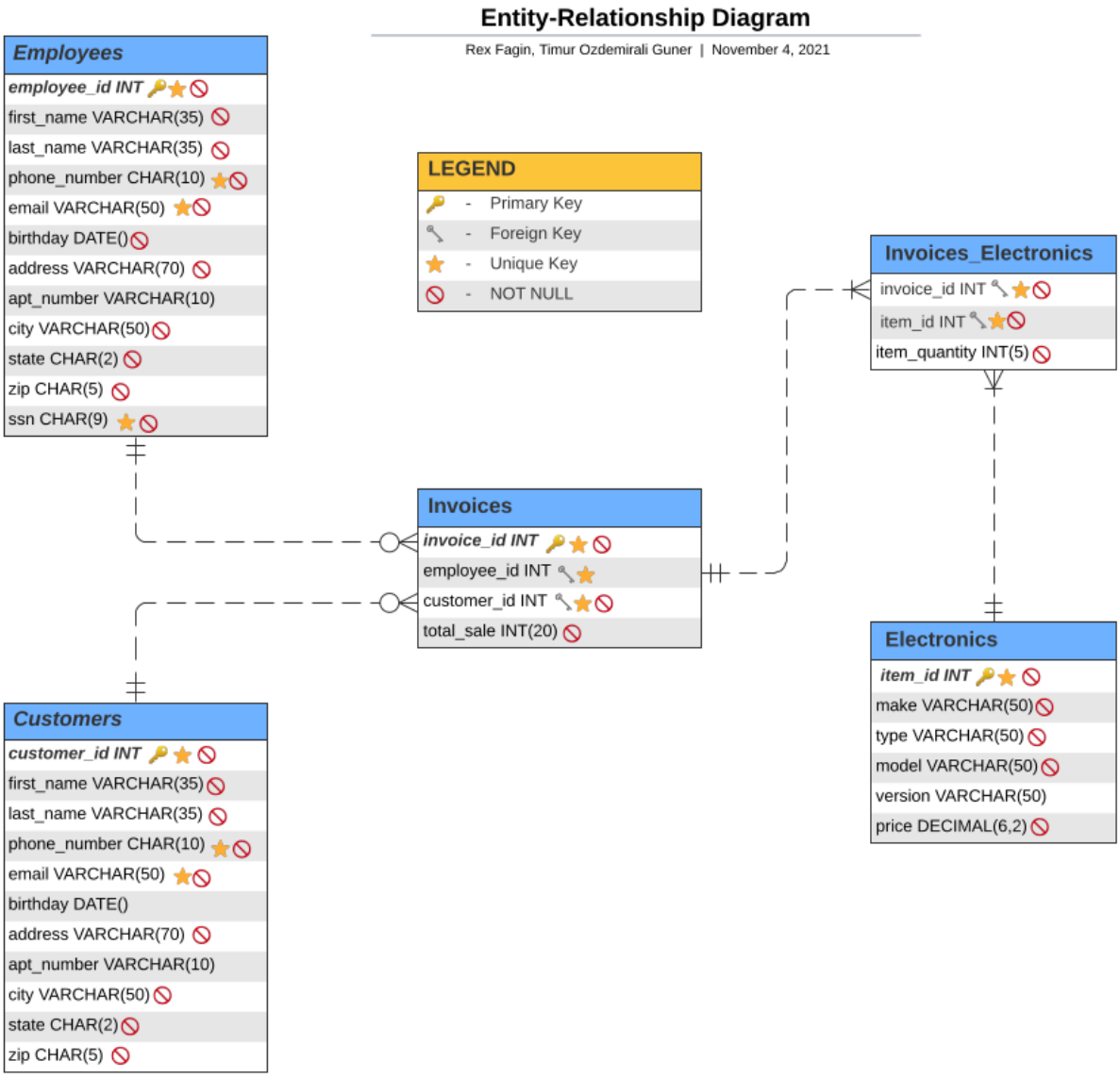
- Employees: records data of employees that work at the business
    - employee_id: INT, auto-generated and auto-incremented, UK, NOT NULL, PK
    - first_name: VARCHAR(35), NOT NULL
    - last_name: VARCHAR(35), NOT NULL
    - phone_number: CHAR(10), UK, NOT NULL
    - email: VARCHAR(50), UK, NOT NULL
    - birthday: DATE(), NOT NULL
    - address: VARCHAR(70), NULL
    - apt_number: VARCHAR(10), NULL
    - city: VARCHAR(50), NOT NULL
    - state: CHAR(2), NOT NULL
    - zip: CHAR(5), NOT NULL
    - ssn: CHAR(9), NOT NULL, UK
    - Relationship - The employee has a 1:M relationship to invoices and is linked by the EmployeeID (foreign key) in the Invoice Table

- Electronics: this is the merchandise of the business
    - item_id: INT, auto-generated and auto-incremented, UK, NOT NULLI, PK
    - make: VARCHAR(50), NOT NULL
    - type: VARCHAR(50), NOT NULL
    - model: VARCHAR(50), NOT NULL

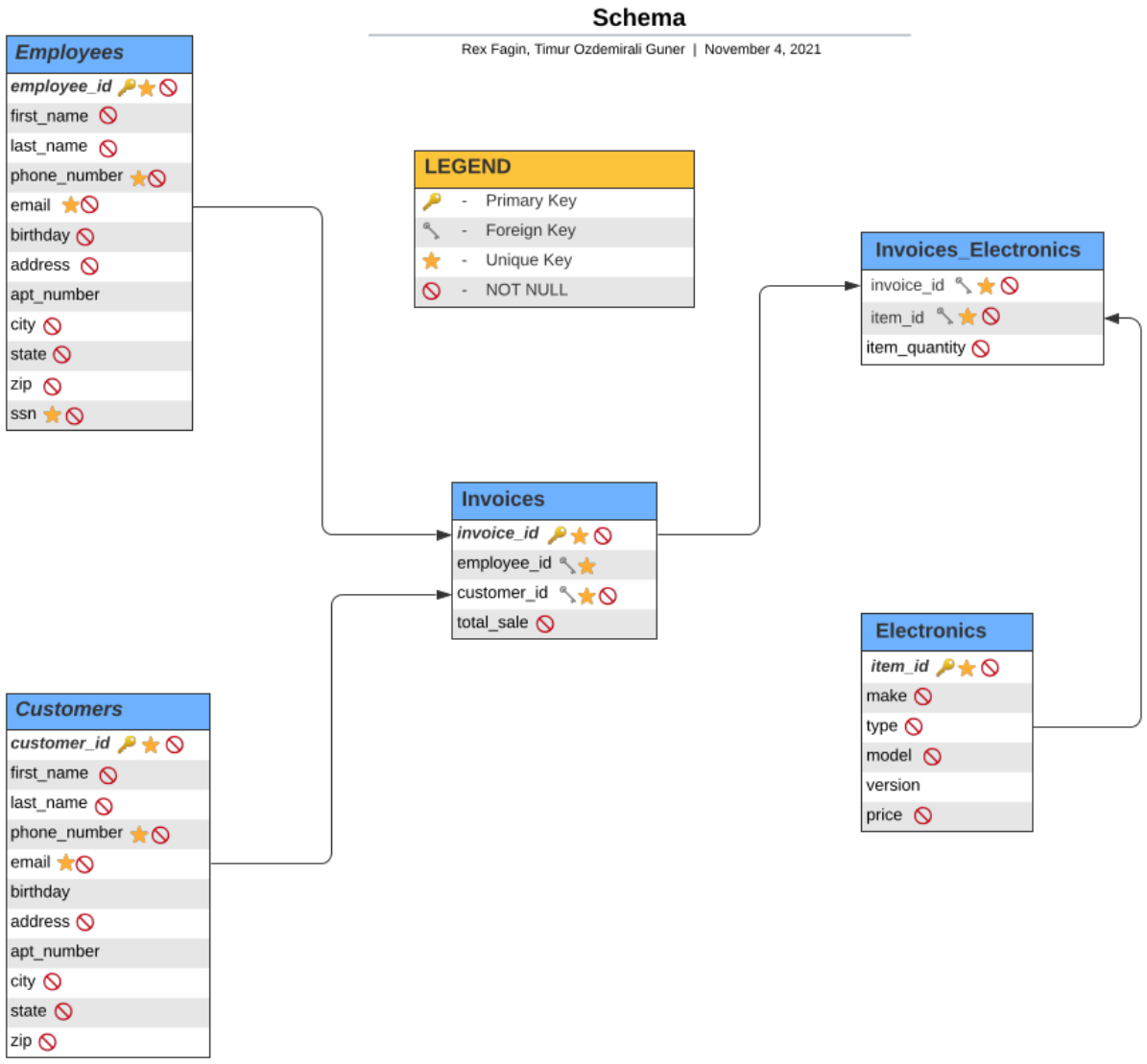
    - version: VARCHAR(50), NULL
    - price: DECIMAL(6,2), NOT NULL
    - Relationship - Has a 1:M relationship with InvoiceElectronics that is mapped by the ItemID in electronics to the ItemID (foreign key) in InvoiceElectronics. InvoiceElectronics is the intermediate table that maps the M:M between Invoices and Electronics

- Invoice_Electronics:
    - invoice_id: INT, UK, FK, NOT NULL
    - item_id: INT, UK, FK, NOT NULL
    - item_quantity: INT(5), NOT NULL
    - Relationship - bridges that M:M relationship between Electronics and Invoices. It does this by 1 and only 1 relationship with Electronics on the ItemID foreign key and 1 and only 1 relationship with Invoices based on the InvoiceID foreign key, with the combination of the two as the primary key for the table.

Entity Relationship Diagram

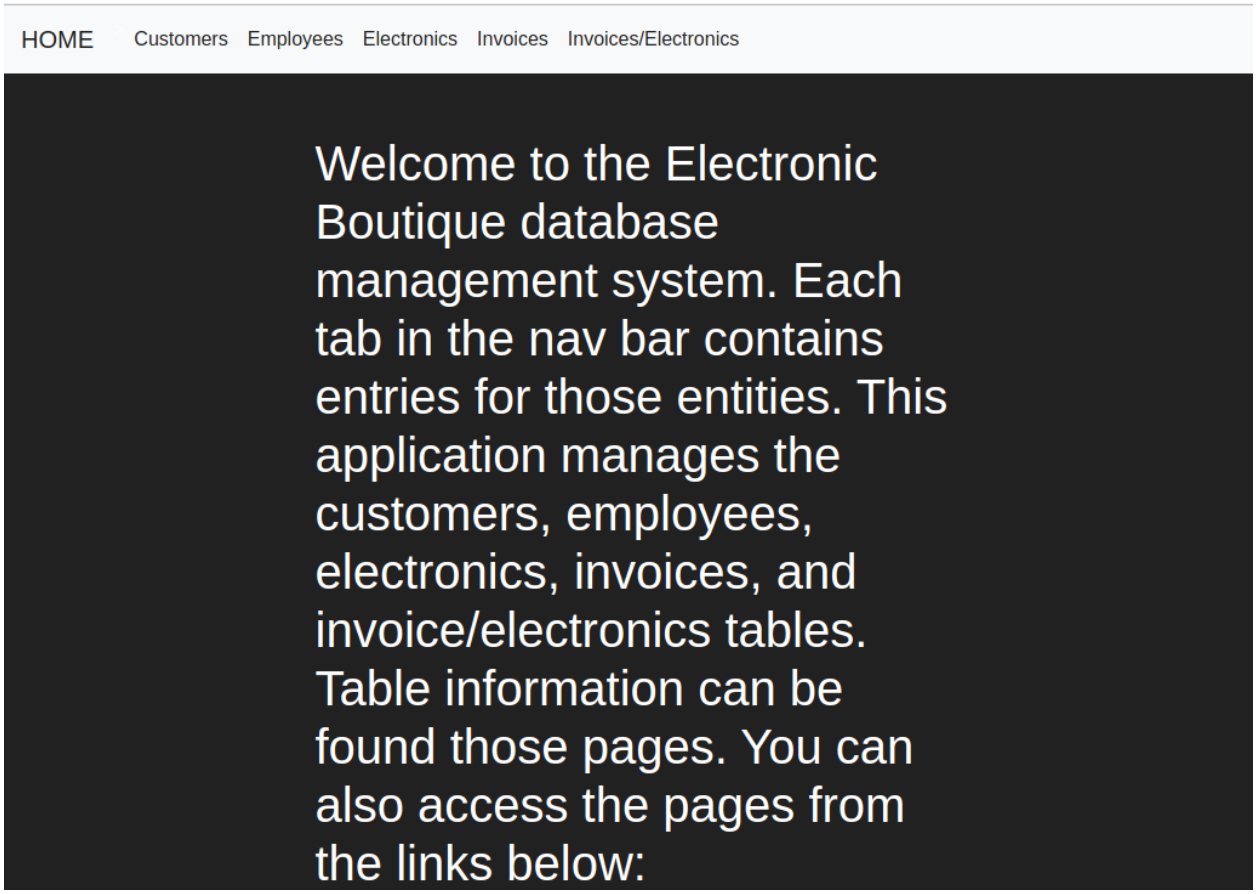


Schema

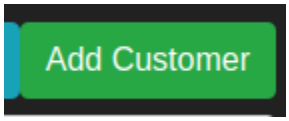

Schema
Rex Fagin, Timur Ozdemirali Guner | November 4, 2021
Employees
employee_id
first_name
last_name
phone_number
email
birthday
address
apt_number
city
state
zip
ssn
LEGEND
- Primary Key
- Foreign Key
- Unique Key
- NOT NULL
Invoices_Electronics
invoice_id
item_id
item_quantity
Invoices
invoice_id
employee_id
customer_id
total_sale
Electronics
item_id
make
type
model
version
price
Customers
customer_id
first_name
last_name
phone_number
email
birthday
address
apt_number
city
state
zip

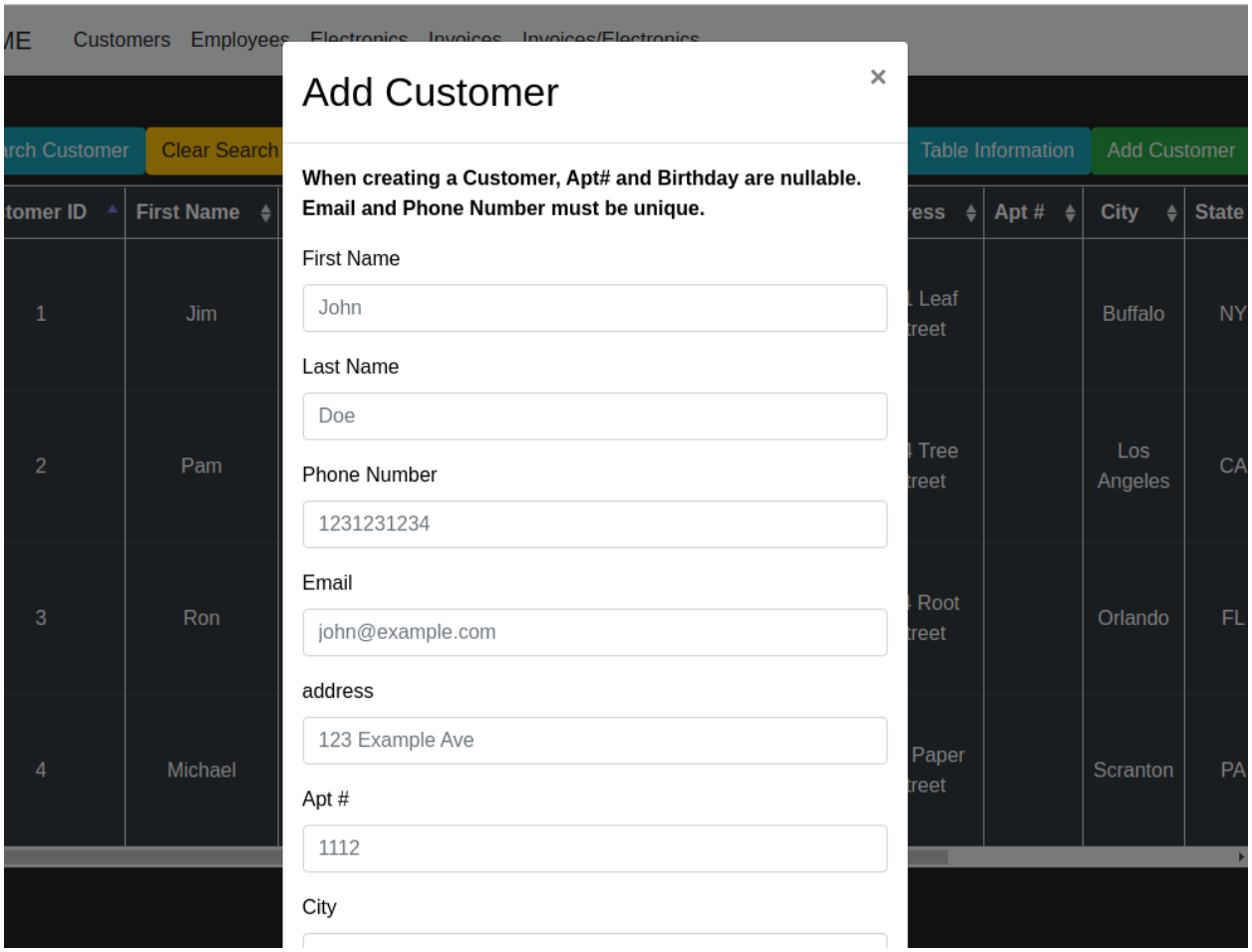
Screenshots

HOMPEPAGE AND DESIGN:

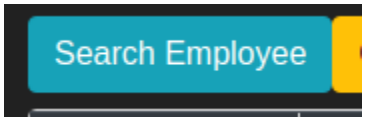


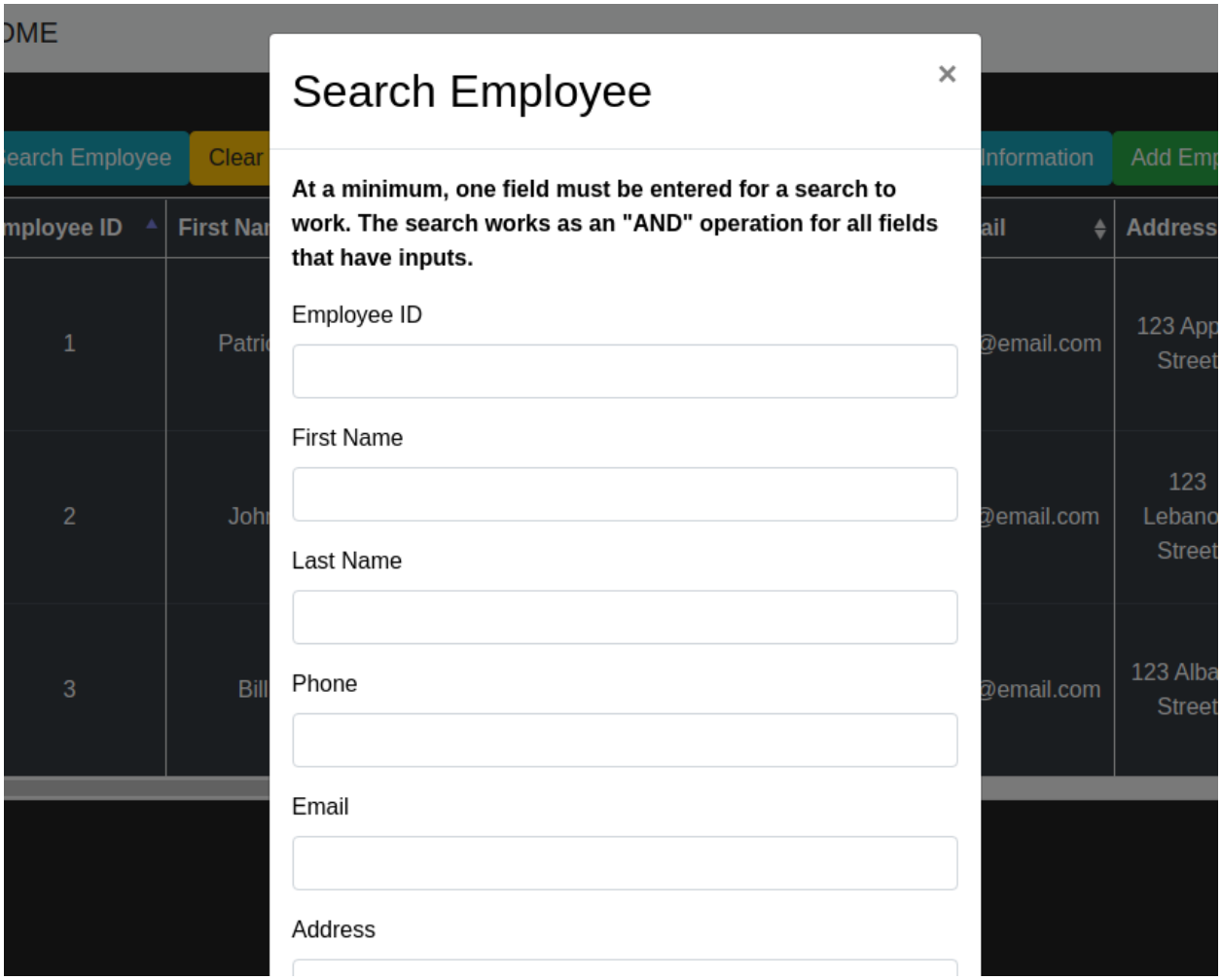
CUSTOMERS - ADD CUSTOMER:



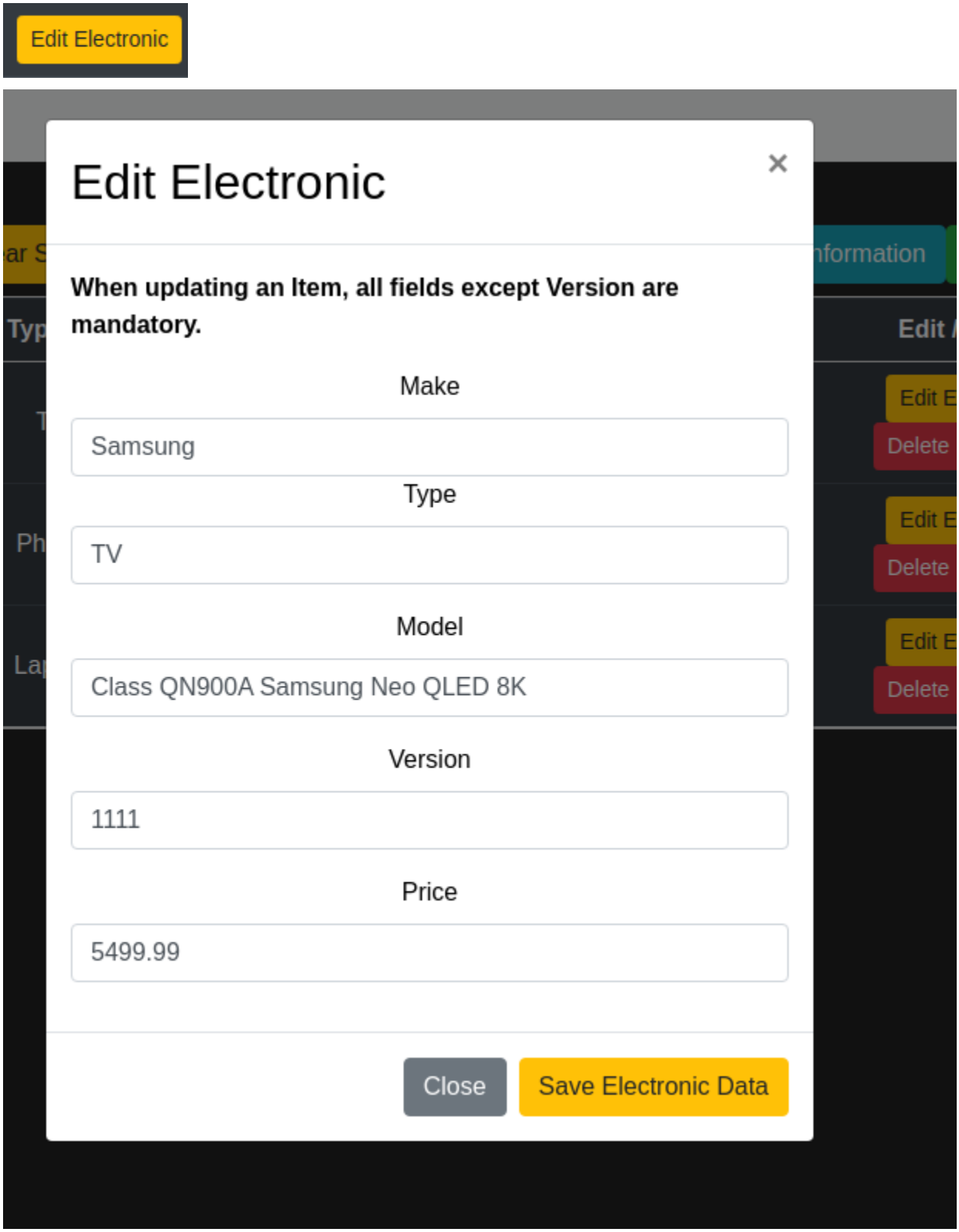


EMPLOYEES - SEARCH EMPLOYEE:

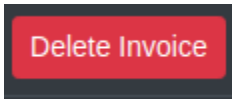


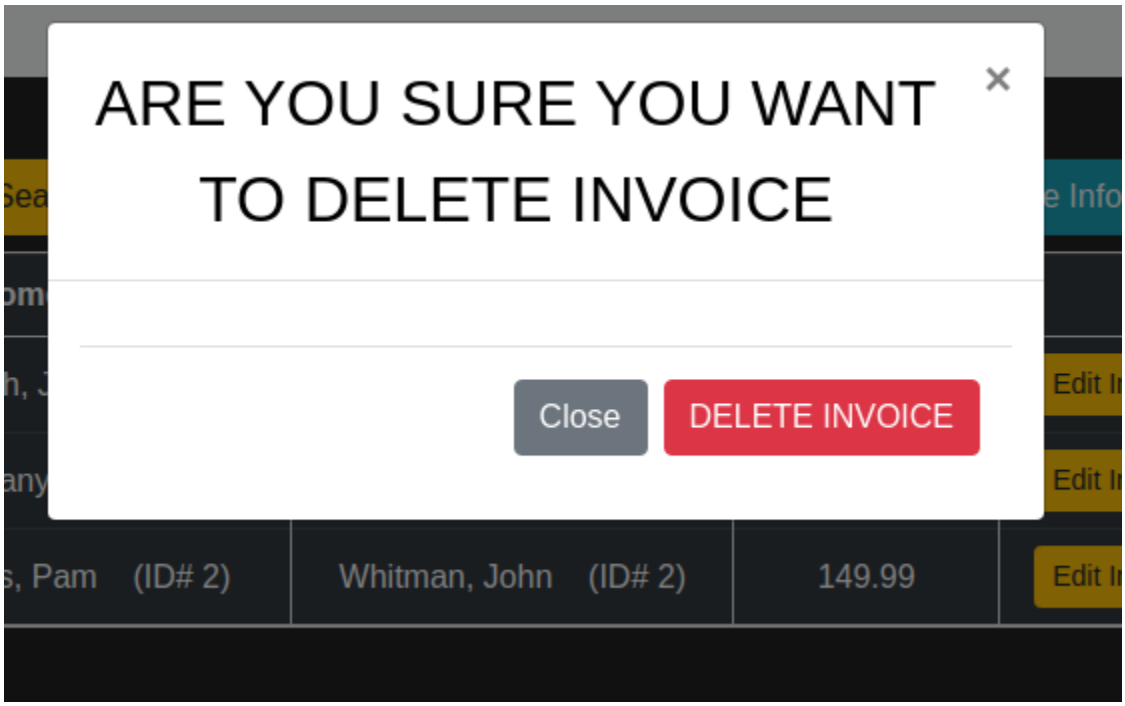


ELECTRONICS - EDIT ELECTRONIC:



INVOICES - DELETE INVOICE:





INVOICE / ELECTRONICS - SELECT: