Clothing Store Point of Sale System
Data Management

Prepared by: Alan Shami

April 21, 2023

System Description

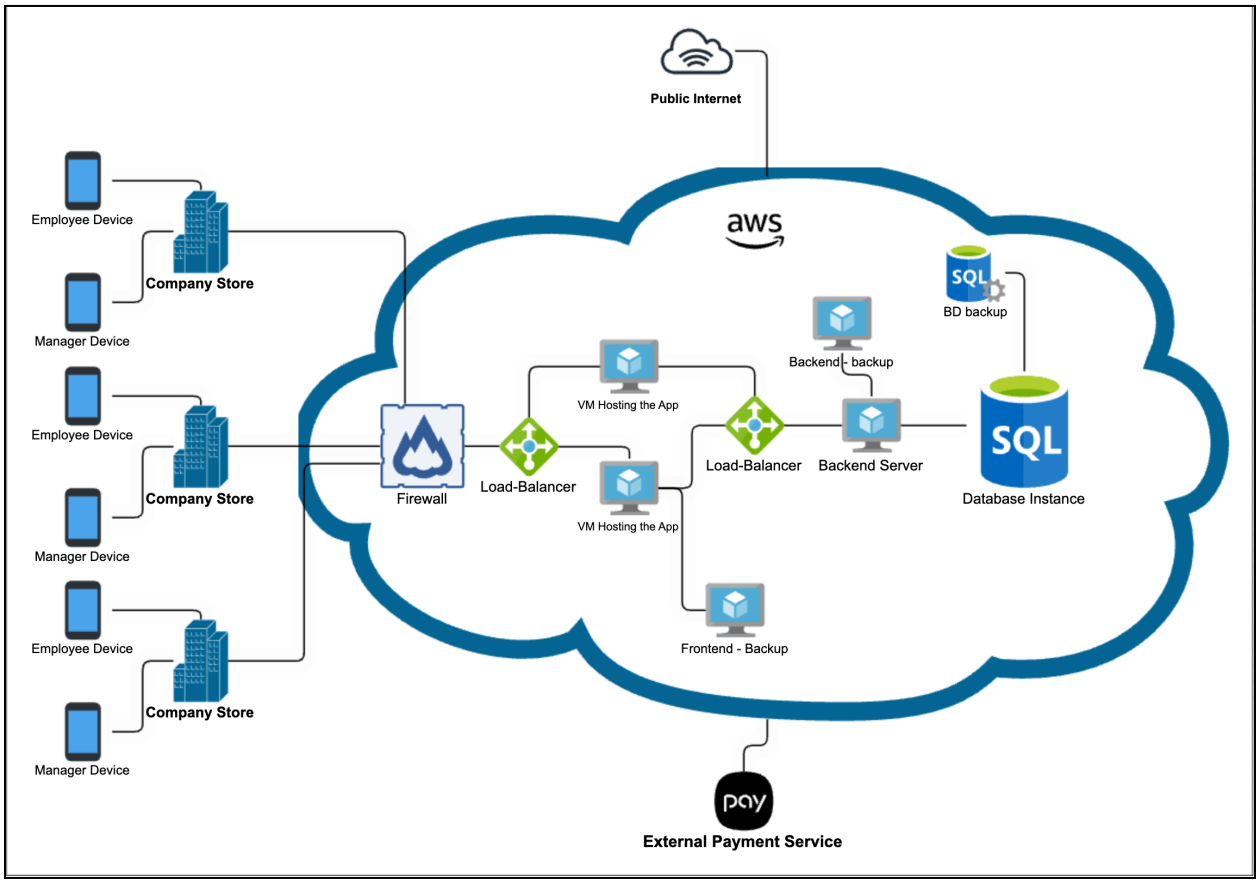
This mobile app works across multiple platforms (iOS and Android) and is intended for internal use only by company employees. The app is designed for a clothing store with multiple locations across the city and has two levels of user access: employees and managers. Employees can make sales and issue refunds, taking into consideration sales tax in both cases. Items can be identified by scanning the barcode using the device camera as a scanner. The app can also accept different payment methods through an external payment service provider. However, refunds will be cash only due to company policy. After each transaction (whether a purchase or a refund transaction), the store's inventory should be updated across all locations. All store chains' inventory data will be stored and backed up on a public cloud provider. The database will utilize a relational database system, which will consist of tables to cover all the system's data. This will enable the app to allow employees to look up items by ID, item name, or date added to the inventory. Some records or data will be accessible only to privileged users (administrators), such as transaction history and sales numbers.

Software Architecture Overview

Cloud Architecture and Architectural Diagram:

The application will be hosted on a public cloud provider (AWS) for easy control, security, cost-effectiveness, and management. The mobile app will be hosted on a virtual server, and it will make use of the security group (the cloud firewall) to limit access to the app to only within the company’s internal network so that any public access will be denied by default. At first, there will be two servers that host the app, to increase availability, however, auto-scaling will be enabled in case the traffic is increased or new establishment of new stores. The frontend server will be backed up periodically in case of a disaster, or loss of data. The two initial servers will be connected to the backend server, which will be initially one server and has all the system logic with enabled auto-scaling as well, by a load-balancer to distribute the load among the servers (in case the backend servers increase), and also to increase security. Also, the facing access will go through another load-balancer to distribute the traffic among the hosting servers and

increase availability and decrease network lag. The backend server will also be backed up for the same reason as the frontend server. The system will use a relational database, in this case, MySQL, to better access and organize the data.



Data Management Strategy

The database system will make use of MySQL for better storage, organization, and performance of the system when querying the data. MySQL is chosen because of its reliability, scalability, and ease of use. MySQL offers a robust solution for managing the complex relationships between the different components of the system, such as employees, stores, items, inventory, and transactions.

The design of the Clothing Store Point of Sale System database schema using MySQL includes the following tables:

1. **Store:**

The Store database contains three pieces of information. These include the store name, the store address, and the store ID number. The ID number is used throughout the other databases to place items, employees, and transactions at the corresponding store locations. This table will have a one-to-many relationship with the following tables:

a. `'employee'` table: Since each store will have many employees.
b. `'transactions'` table: Since each store will have many transactions.

2. Employee:

The Employee database holds all information about employees. That includes basic data such as first and last name, as well as an employee ID number. It also contains the employee's email, password, and ID number of the store they work at. Lastly, it contains whether or not the employee is an administrator or a basic employee, which determines how much access they might have. This table will have a one-to-many relationship with the following tables:

a. `'item'` table: Since employees need to access items within their store and other stores.
b. `'transactions'` table: Since employees will make the transactions.

3. Item:

The Item database holds all information about specific items or products of the store. It contains the item name, price, size, color, quantity, date added, and ID number. All of these aspects can be used to search for a specific item in the inventory. This table will have a One-to-many relationship with the following tables:

a. `'transaction'` table: Since all items will have some sort of a transaction, (sell or refund).
b. `'inventory'` table: All items need to be stored and checked for inventory.

4. Transactions:

The Transaction database contains information related to each transaction. The data is thorough so that each transaction is differentiated. The aspects of the database include the date of the transaction, transaction ID number, cost of the sale, sales tax, and the total cost. It also contains information about whether the transaction was a purchase made or a refund, and it notes the payment method used to complete it (cash, credit, debit, gift card). Lastly, it includes information related to the location at which the transaction occurred, such as the store ID number and the ID number of the employee who completed the transaction.

5. Inventory:

The Inventory database holds more basic information about items and products, such as their ID numbers and their quantity. Additionally, it includes the ID number of the store location where the item is located.

Database Design Diagram:

This diagram will illustrate the design of the database and the type of relationships between each of the tables: