Nozama E-Commerce

CS6360 Database Design

Jalal Omer

Matthew Standeven, Ivan Macsemchuk, Jonathan Kao, Ryan Bell

05/01/2023

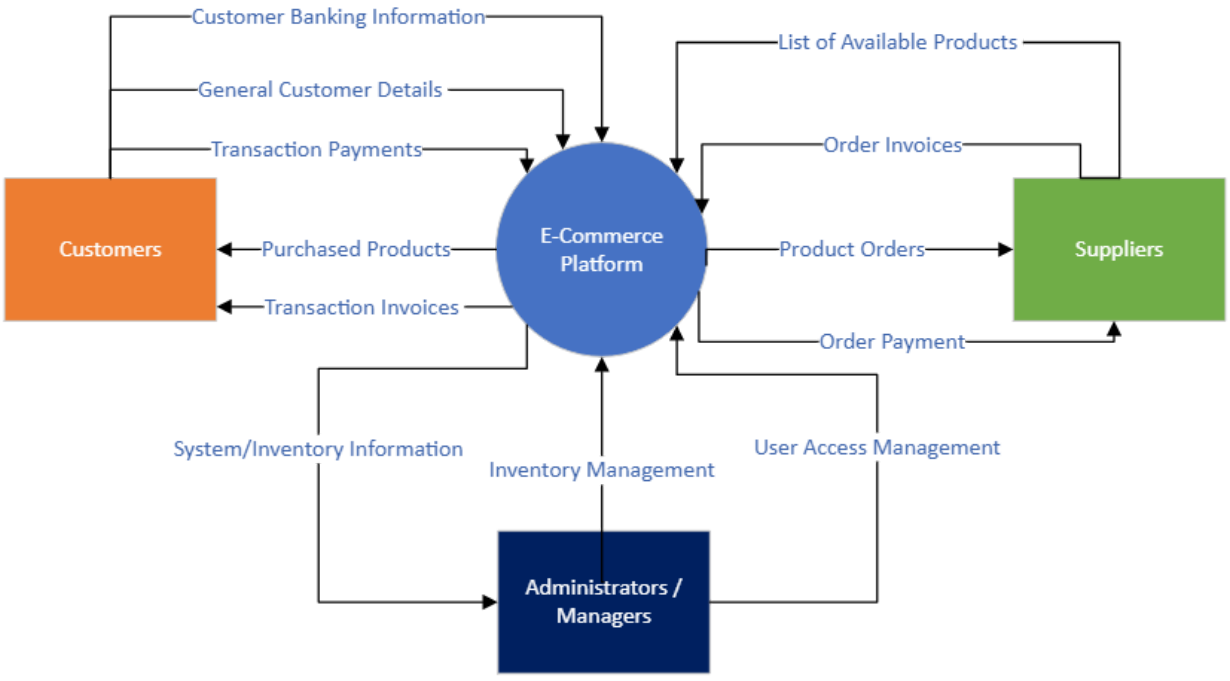
Table of Contents

Introduction

The Nozama E-commerce platform consists of a front-end implemented in Java and a back-end implemented using the MySQL DBMS. This front-end will enable consumers and suppliers to register, login, search for, buy, and sell products. In addition, the front-end will enable administrators to manage all aspects of the Nozama product inventory, transactions and user base. All associated data will be stored in the MySQL database backend and accessed through a JDBC connector.

This project report consists broadly of three sections. The first section concerns the design process of the project including topics such as system requirements, conceptual and logical design, functional dependencies, and normalization. The following section concerns the actual implementation of the database and the database system and how basic interactions can take place with the system in the absence of a front end. Following that we will discuss the implementation of a front end application and more queries and tricks used in the coding as well as various lessons learned. Lastly we will discuss conclusions of the work and future improvements that could come from it.

System Requirements



Functional Requirements

1. **Login Functional Requirements**
    a. The system will allow the user to login
    b. The system will verify the username and password
    c. The system will not allow the user to log in with an invalid username or password
    d. The system will be able to remember usernames and passwords
    e. The system will allow the users to create accounts.
    f. The system will allow users to select whether they are a customer or supplier.
    g. The system will change from the login page to the main product page upon successful user login.

2. **Browsing Functional Requirements**
    a. The system will display all the products including their name, price and rating.
    b. The system shall allow the user to sort items by rating, price, or alphabetically.
    c. The system will allow the user to filter items by rating and/or price.
    d. The system will allow users to open and view more details about a product.
    e. The system will allow users to add products on the main page to a 'shopping cart' list.
    f. The system will be able to remember what items the user has added to their shopping cart previously.
    g. The system will allow the user to check their purchase history.
    h. The system will allow the user to change quantities and remove items from their shopping cart.
    i. The system will allow the user to see the total price of their selected products.
    j. The system will allow the user to purchase all items in their shopping cart.
    k. The system will allow the user to modify their account details and banking information.
    l. The system will allow suppliers to see their products and their purchase history.
    m. The system will allow suppliers to add products to sell.

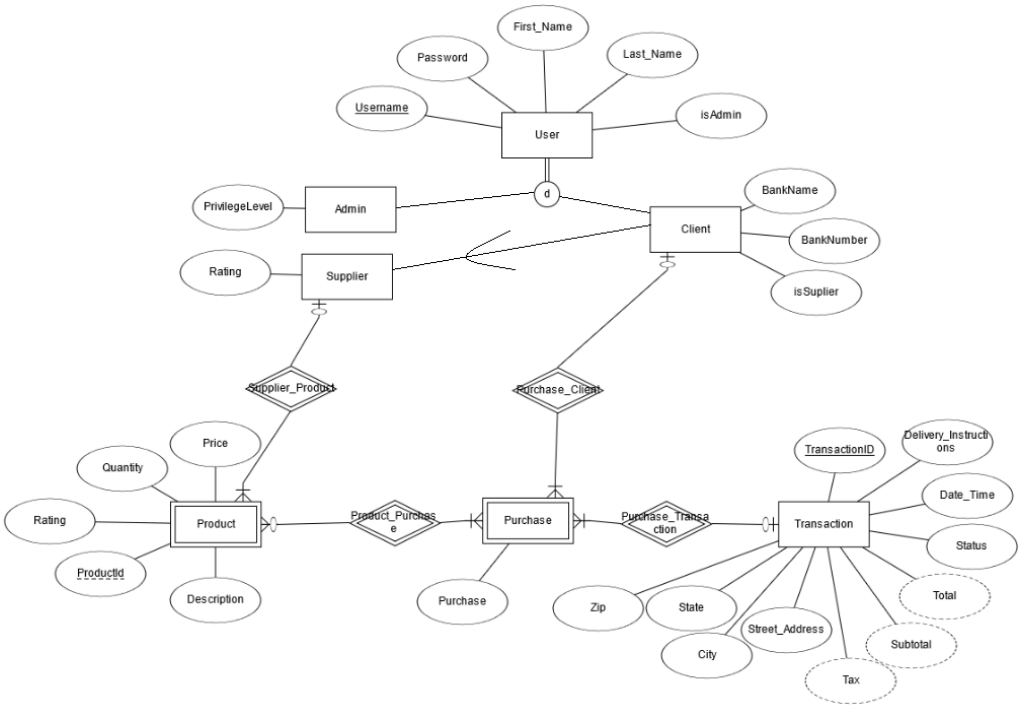
3. **Administrator Functional Requirements**
    a. The system will allow an administrator user to calculate the number of total users, supplier users, and regular users in the system.
    b. The system will allow an administrator to calculate the unique number of products and the total number of products (taking into account quantity) in the system.

c. The system will allow an administrator user to lookup user information, including # items sold/bought, account email address and user ID, etc.
d. The system will allow an administrator user to open any user's shopping cart with the same permissions.
e. The system will allow an administrator to look up the number of items sold per day/per month/per year.
f. The system will allow an administrator to look up the number of items added by all suppliers per day/per month/per year.
g. The system will allow an administrator to open any supplier's product information page with the same permissions.
h. The system will allow an administrator to open any user's information with ability to edit.
i. The system will allow an administrator to make any user who isn't a supplier into one.

Non-Functional Requirements

1. Security - Usernames must be between 4 and 12 characters long.
2. Security - User passwords must be between 6 and 12 characters long.
3. Security - Only administrator users can change passwords / banking information.
4. Security - Administrator users can change user type (user privileges), including changing users to suppliers.
5. Performance - Each front-end page should load within 3 seconds after the corresponding text command.
6. Performance - Product additions/removals/edits should be processed and displayed within one second.
7. Performance - Database queries should be processed and displayed within one second.
8. Availability - Normal users can update their cart and checkout at any time unless the system is undergoing maintenance.
9. Maintainability - A back-up database will be updated on a daily basis for rollback recovery in case of database failure.

Conceptual Design of the Database



Business Rules

1. After a purchase, update the remaining quantities for the products sold.
2. Once a transaction is finalized and its status is changed to ORDER SUBMITTED the price, tax, and subtotal is locked in and cannot be changed if the price of one of the products in the order is changed.
3. Once a transaction is finalized and its status is changed to ORDER SUBMITTED, the quantity of each product will be subtracted from the quantity attribute of the associated product.
4. Before a client can be deleted, all their purchases must be in a RECEIVED state or in a IN CART state. If all purchases are delivered then the client can be deleted. When deleted the IN CART transactions will be deleted. If transactions are submitted, but not delivered then the client can't be deleted.
5. Before a product can be deleted, all the purchases associated with it must be in an IN CART or RECEIVED state. If not, then the product can't be deleted. If all are in RECEIVED state, then the IN CART transactions can be deleted and the product can be deleted.
6. Before a transaction can be deleted it must be in a RECEIVED or IN CART state and all purchases referencing the transaction must be deleted.
7. All the product quantities must be updated before the purchases can be deleted.

Integrity Constraints

1. Entity Integrity Constraints
    a. A user entity, and the associated admin and client entities, are uniquely identified by their username whose value cannot be NULL and is a varchar 4-12.
    b. A product entity is uniquely identified by its supplier and its productID, neither of whom's values can be NULL.
    c. A purchase entity is uniquely identified by a username, productID, and transactionID, none of whose values can be NULL.
    d. A transaction entity is uniquely identified by its transactionID whose value cannot be NULL.
    e. An admin entity is uniquely identified by its username whose value cannot be NULL.
2. Referential Integrity Constraints'
    a. User
        i. Admin username FK will reject on update (username can't change), and cascade on delete.
        ii. Client username FK will reject on update (username can't change), and cascade on delete.
    b. Product
        i. Product supplier FK will reject on update (supplier can't change, a new product will have to be added to replace an old one), and reject on delete.
    c. Purchase
        i. Client FK will reject on update because a username can't be changed. It will reject on delete because if a client is deleted we can't delete their purchases until we're certain they've been delivered.
        ii. Product FK will reject on update because product ID can't be changed. It will reject on delete because if a product is deleted we need to make sure they've all been delivered before deleting the purchase.
        iii. Transaction FK will reject on update because a transaction ID can't be changed. It will reject on delete because if a transaction is deleted it still needs a record of the quantity sold to maintain product quantities.
    d. Transaction
        i. No FKs.
3. Domain Integrity Constraints
    a. User
        i. Passwords will be varchar 6-12 and cannot be null.

ii. First and last names will be varchar 1-20 each and at least the last name cannot be null.
iii. Admin privilege level should be an int between 1-5.
iv. Bank name is a varchar 1-25
v. Bank number is a varchar 1-20
vi. A supplier's rating must be between 1 and 5.
vii. Supplier or Admin flag is a tinyint 0 or 1

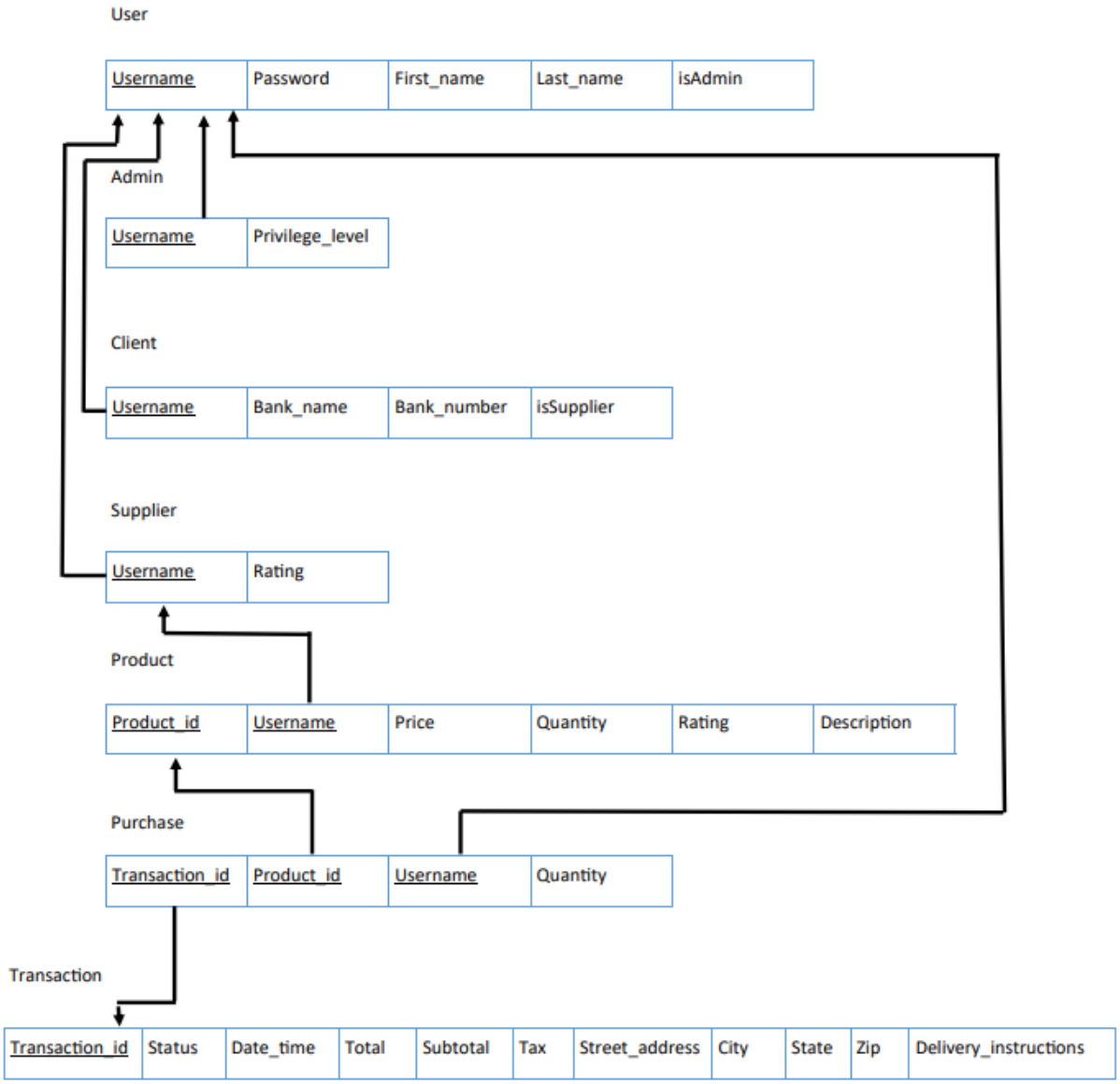
b. Product
   i. A product cannot be purchased if that would reduce the quantity of that product below 0.
   ii. Product description is a varchar 1-100
   iii. Product price is a decimal with 2 digits
   iv. Product quantity is an int >= 0
   v. A product's rating must be between 1 and 5.

c. Purchase
   i. Purchase quantity is an int >= 1

d. Transaction
   i. Status is an enumerated type including {IN CART, ORDER SUBMITTED, SHIPPED, RECEIVED}
   ii. Date is a date type of when the transaction was made
   iii. Time is a time type of when the transaction was made
   iv. Price is a composite type including total, subtotal, and tax
   v. Total is a float of the subtotal added with the tax. It can be derived.
   vi. Subtotal is a float of the sum of all product prices and can be derived.
   vii. Tax is the subtotal x 6.25% and can be derived
   viii. Shipping address is a composite type
   ix. Street address is a varchar 3-50
   x. City is a varchar 2-30
   xi. State is a char 2-30
   xii. Zip is a char 5
   xiii. Delivery instructions is a varchar 0-500

Logical Database Schema

Logical Database Schema:



Commands to Create Database:

CREATE DATABASE NozamaDB;

USE NozamaDB;

CREATE TABLE User (
    Username VARCHAR(20) NOT NULL,
    Password VARCHAR(12) NOT NULL,

```
    First_name VARCHAR(20),
    Last_name VARCHAR(20) NOT NULL,
    isAdmin tinyint(1),
    CONSTRAINT CHK_UsernameLength CHECK (LENGTH(Username) >= 3 AND
    LENGTH(Username) <= 20),
    CONSTRAINT CHK_PasswordLength CHECK (LENGTH(Password) >= 4 AND
    LENGTH(Password) <= 12),
    PRIMARY KEY (Username)
);
```

SHOW COLUMNS FROM User;

```
mysql> show columns from user;
+------------+-------------+------+-----+---------+-------+
| Field      | Type        | Null | Key | Default | Extra |
+------------+-------------+------+-----+---------+-------+
| Username   | varchar(20) | NO   | PRI | NULL    |       |
| Password   | varchar(12) | NO   |     | NULL    |       |
| First_name | varchar(20) | YES  |     | NULL    |       |
| Last_name  | varchar(20) | NO   |     | NULL    |       |
| isAdmin    | tinyint(1)  | YES  |     | NULL    |       |
+------------+-------------+------+-----+---------+-------+
5 rows in set (0.01 sec)
```

```
CREATE TABLE Admin(
    Username VARCHAR(20) NOT NULL,
    Privilege_level enum('1', '2', '3', '4', '5'),
    PRIMARY KEY (Username),
    FOREIGN KEY (Username) REFERENCES User(Username)
    ON DELETE CASCADE ON UPDATE RESTRICT
);
```

```
mysql> show columns from admin;
+-----------------+---------------------------+------+-----+---------+-------+
| Field           | Type                      | Null | Key | Default | Extra |
+-----------------+---------------------------+------+-----+---------+-------+
| Username        | varchar(20)               | NO   | PRI | NULL    |       |
| Privilege_level | enum('1','2','3','4','5') | YES  |     | NULL    |       |
+-----------------+---------------------------+------+-----+---------+-------+
2 rows in set (0.00 sec)
```

```
CREATE TABLE Client (
    Username VARCHAR(20) NOT NULL,
    Bank_name VARCHAR(25),
    Bank_number VARCHAR(20) NOT NULL,
    isSupplier tinyint(1),
    PRIMARY KEY (Username),
```

```
    FOREIGN KEY(Username) references User(Username)
    ON DELETE CASCADE ON UPDATE RESTRICT
);
```

```
mysql> show columns from client;
+-------------+-------------+------+-----+---------+-------+
| Field       | Type        | Null | Key | Default | Extra |
+-------------+-------------+------+-----+---------+-------+
| Username    | varchar(20) | NO   | PRI | NULL    |       |
| Bank_name   | varchar(25) | YES  |     | NULL    |       |
| Bank_number | varchar(20) | NO   |     | NULL    |       |
| isSupplier  | tinyint(1)  | YES  |     | NULL    |       |
+-------------+-------------+------+-----+---------+-------+
4 rows in set (0.00 sec)
```

```
CREATE TABLE Supplier (
    Username VARCHAR(20) NOT NULL,
    Rating DECIMAL(2,1),
    PRIMARY KEY (Username),
    FOREIGN KEY (Username) REFERENCES User(Username)
    ON DELETE CASCADE ON UPDATE RESTRICT
);
```

```
mysql> show columns from supplier;
+----------+--------------+------+-----+---------+-------+
| Field    | Type         | Null | Key | Default | Extra |
+----------+--------------+------+-----+---------+-------+
| Username | varchar(20)  | NO   | PRI | NULL    |       |
| Rating   | decimal(2,1) | YES  |     | NULL    |       |
+----------+--------------+------+-----+---------+-------+
2 rows in set (0.00 sec)
```

```
CREATE TABLE Product (
    Product_id INT NOT NULL,
    Username VARCHAR(20) NOT NULL,
    Price DECIMAL(10, 2),
    Quantity INT NOT NULL,
    Rating DECIMAL(2,1),
    Description VARCHAR(100),
    CONSTRAINT CHK_NegativeQuantity CHECK (Quantity >=0),
    PRIMARY KEY(Product_id),
    FOREIGN KEY(Username) REFERENCES Supplier(Username)
    ON DELETE CASCADE ON UPDATE RESTRICT
);
```

```
mysql> show columns from product;
+-------------+---------------+------+-----+---------+-------+
| Field       | Type          | Null | Key | Default | Extra |
+-------------+---------------+------+-----+---------+-------+
| Product_id  | int           | NO   | PRI | NULL    |       |
| Username    | varchar(20)   | NO   | MUL | NULL    |       |
| Price       | decimal(10,2) | YES  |     | NULL    |       |
| Quantity    | int           | NO   |     | NULL    |       |
| Rating      | decimal(2,1)  | YES  |     | NULL    |       |
| Description | varchar(100)  | YES  |     | NULL    |       |
+-------------+---------------+------+-----+---------+-------+
6 rows in set (0.00 sec)
```

```
CREATE TABLE Transaction (
        Transaction_id INT NOT NULL,
        Status ENUM('IN_CART', 'ORDER_SUBMITTED', 'SHIPPED', 'RECEIVED') NOT NULL,
        Date_time Datetime NOT NULL,
        Total DECIMAL(10, 2) NOT NULL,
        Subtotal DECIMAL(10, 2) NOT NULL,
        Tax DECIMAL(10, 2) NOT NULL,
        Street_address VARCHAR(50) NOT NULL,
        City VARCHAR(30) NOT NULL,
        State VARCHAR(30) NOT NULL,
        Zip CHAR(5) NOT NULL,
        Delivery_instructions VARCHAR(100) NOT NULL,
        CONSTRAINT CHK_AddressLength CHECK (LENGTH(Street_address) >= 3),
        CONSTRAINT CHK_CityLength CHECK (LENGTH(City) >= 3),
        PRIMARY KEY(Transaction_id)
);
```

```
mysql> show columns from transaction;
+----------------------+-----------------------------------------------------------+------+-----+---------+-------+
| Field                | Type                                                      | Null | Key | Default | Extra |
+----------------------+-----------------------------------------------------------+------+-----+---------+-------+
| Transaction_id       | int                                                       | NO   | PRI | NULL    |       |
| Status               | enum('IN_CART','ORDER_SUBMITTED','SHIPPED','RECEIVED')    | NO   |     | NULL    |       |
| Date_time            | datetime                                                  | NO   |     | NULL    |       |
| Total                | decimal(10,2)                                             | NO   |     | NULL    |       |
| Subtotal             | decimal(10,2)                                             | NO   |     | NULL    |       |
| Tax                  | decimal(10,2)                                             | NO   |     | NULL    |       |
| Street_address       | varchar(50)                                               | NO   |     | NULL    |       |
| City                 | varchar(30)                                               | NO   |     | NULL    |       |
| State                | varchar(30)                                               | NO   |     | NULL    |       |
| Zip                  | char(5)                                                   | NO   |     | NULL    |       |
| Delivery_instructions | varchar(100)                                             | NO   |     | NULL    |       |
+----------------------+-----------------------------------------------------------+------+-----+---------+-------+
11 rows in set (0.00 sec)
```

```
CREATE TABLE Purchase (
        Transaction_id INT NOT NULL,
        Product_id INT NOT NULL,
        Username VARCHAR(20) NOT NULL,
```

```
        Quantity INT NOT NULL,
        CONSTRAINT CHK_PositiveQuantity CHECK (QUANTITY >= 1),
        PRIMARY KEY (Transaction_id, Product_id, Username),
        FOREIGN KEY(Transaction_id) REFERENCES Transaction(Transaction_id)
        ON DELETE RESTRICT ON UPDATE RESTRICT,
        FOREIGN KEY (Product_id) REFERENCES Product(Product_id)
        ON DELETE RESTRICT ON UPDATE RESTRICT,
        FOREIGN KEY(Username) REFERENCES User(Username)
        ON DELETE RESTRICT ON UPDATE RESTRICT
);
```

```
mysql> show columns from purchase;
+----------------+-------------+------+-----+---------+-------+
| Field          | Type        | Null | Key | Default | Extra |
+----------------+-------------+------+-----+---------+-------+
| Transaction_id | int         | NO   | PRI | NULL    |       |
| Product_id     | int         | NO   | PRI | NULL    |       |
| Username       | varchar(20) | NO   | PRI | NULL    |       |
| Quantity       | int         | NO   |     | NULL    |       |
+----------------+-------------+------+-----+---------+-------+
4 rows in set (0.00 sec)
```

Database SQL Constraints:

The following triggers enforce the constraint that no single transaction can ever be associated with more than one username and the constraint that no client can simultaneously have two shopping carts. If any insert or update operation would cause either of these conditions to occur the system outputs an appropriate error message and the transaction is rolled back.

```
DELIMITER $$

CREATE TRIGGER check_purchase_update
AFTER UPDATE ON Purchase
FOR EACH ROW
BEGIN
   IF EXISTS (SELECT transaction_ID, MAX(username)
         FROM purchase
         GROUP BY transaction_ID
         HAVING COUNT(DISTINCT username) > 1) THEN
      SIGNAL SQLSTATE '45000'
      SET MESSAGE_TEXT = 'Update Error: More than one username associated with transaction';
   END IF;
END;
$$
```

```
DELIMITER ;

DELIMITER $$

CREATE TRIGGER check_purchase_insert
AFTER INSERT ON Purchase
FOR EACH ROW
BEGIN
   IF EXISTS (SELECT transaction_ID, MAX(username)
         FROM purchase
         GROUP BY transaction_ID
         HAVING COUNT(DISTINCT username) > 1) THEN
      SIGNAL SQLSTATE '45000'
      SET MESSAGE_TEXT = 'Insertion Error: More than one username associated with transaction';
   END IF;
END;
$$

DELIMITER ;

DELIMITER $$

CREATE TRIGGER check_transaction_shopping_cart_insert
AFTER INSERT ON Transaction
FOR EACH ROW
BEGIN
   IF EXISTS (SELECT username
FROM (
         SELECT Transaction.transaction_id, Purchase.username
         FROM Transaction
         INNER JOIN Purchase ON Transaction.transaction_ID = Purchase.transaction_ID
         WHERE Status = 'IN_CART'
         GROUP BY Transaction.transaction_id, username
) AS subquery
GROUP BY username
HAVING COUNT(*) > 1) THEN
      SIGNAL SQLSTATE '45000'
      SET MESSAGE_TEXT = 'Insert Error: User already has a transaction representing their shopping
cart';
   END IF;
END;
$$

DELIMITER ;
```

```
DELIMITER $$

CREATE TRIGGER check_transaction_shopping_cart_update
AFTER UPDATE ON Transaction
FOR EACH ROW
BEGIN
   IF EXISTS (SELECT username
FROM (
        SELECT Transaction.transaction_id, Purchase.username
        FROM Transaction
        INNER JOIN Purchase ON Transaction.transaction_ID = Purchase.transaction_ID
        WHERE Status = 'IN_CART'
        GROUP BY Transaction.transaction_id, username
) AS subquery
GROUP BY username
HAVING COUNT(*) > 1) THEN
      SIGNAL SQLSTATE '45000'
      SET MESSAGE_TEXT = 'Update Error: User already has a transaction representing their shopping
cart';
   END IF;
END;
$$

DELIMITER ;
```

Database Operations and Volumes:

Since the Nozama database is an e-commerce platform, normal operations have to do with the buying and selling of products and the managing of users. These operations can be broken up by functionality required of each entity.

User Entity:

- Query to read username and password during login
- Query to compare proposed username to all others during registration
- Transaction to enter a new user with their associated information
- Transaction to update user's name or password

Admin Entity:

- Query to check if user is an admin
- Transaction to make a user an admin

Client Entity:

- Transaction to make a user into a client during account creation
- Query to count the number of clients for the admin page
- Query to get information about a client for the user page

- Transaction to change the banking information of a client
- Transaction to change a client to a supplier from the admin page

Supplier Entity:

- Query to get the information from a supplier for the supplier's page
- Transaction to add a supplier during account creation
- Query to get the number of suppliers across the platform

Product Entity:

- Query to get the products for a specific supplier
- Query to get information from a product for display
- Query to get all the products in different orders or with various filtering criteria
- Transaction to reduce the quantity of a product after a purchase
- Transaction to update various aspects of a product for the supplier
- Transaction to add a new product from a supplier
- Queries to get the number of products and their value for the admin page

Transaction Entity:

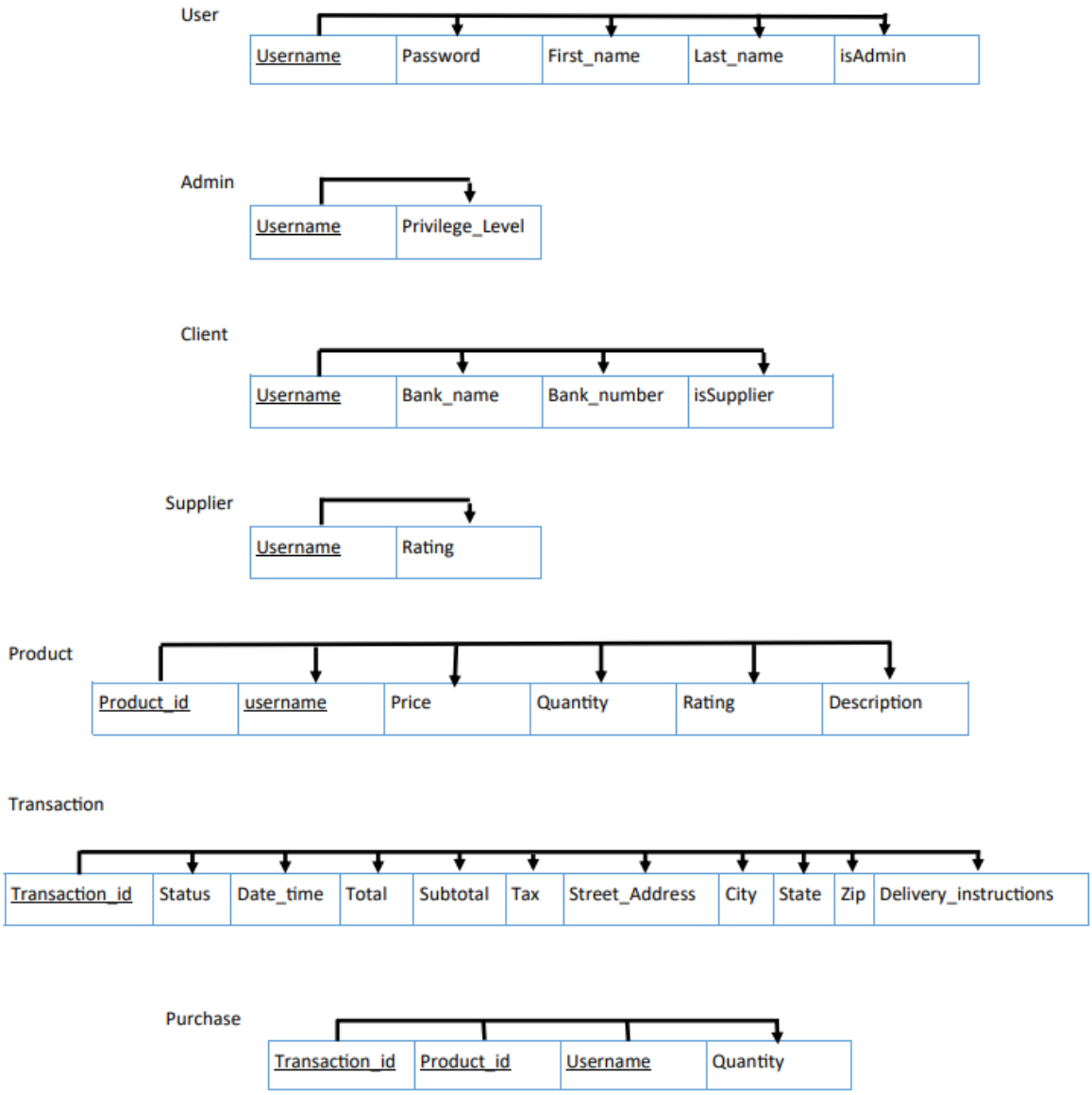
- Queries to get the list of transactions from a specific user that can be displayed in the shopping cart
- Queries to get the list of purchases from a user which is displayed in the previous purchases page
- Query to get a specific transaction which can be used to support other queries
- Transaction to delete a transaction if a user removes some items from their shopping cart
- Transaction to update a transaction if a user decides to buy what's in their cart
- Transaction to add a transaction when a user adds something to their cart

Purchase Entity:

- Query to get purchases with a specific user and transaction id to support other queries
- Query to get all the purchases of a specific product for the suppliers use
- Query to get the quantity of a purchase to maintain product quantities for the supplier
- Transaction to update the quantity of a purchase if a user moves it in or out of the shopping cart
- Transaction to delete purchases associate with a transaction if a user removes it from their shopping cart
- Transaction to add a purchase if the user adds an item to their shopping cart
- Query to get the total cost of a purchase

Our default dataset has 990 clients which includes 96 suppliers. There are 1000 products with a quantity summing to 507,281.

Functional Dependencies and Database Normalization



The Database System

In order to use the Nozama database system, the following prerequisites must be installed: Maven, MySQL 8.0, MySQL Workbench 8.0, JDK 20, and JDBC drivers. The following instructions create and populate a database in MySQL for use with the Nozama database system, compile the Nozama front-end application source code, create a corresponding .jar file, and launch the front-end application of the database.

Database Setup:

1. Unzip the ‘nozama-appendix.zip’ file into a folder called ‘nozama-appendix’.
2. Modify both the root user’s username and password to ‘root’ since the front-end application connects through JDBC assuming this. Alternatively, modify all references to "jdbc:mysql://localhost:3306/nozamadb", "root", "root" in the ‘Controller’ folder of the source code to match your desired MySQL username and password.
3. Log in to MySQL command prompt using the username and password specified in step 2.
4. Run the ‘CREATE DATABASE nozamadb’ command.
5. Open MySQL Workbench and open a local instance.
6. Go to File -> Open SQL Script and then navigate to the ‘dbdata’ folder within ‘nozama-appendix’ and select the.sql file ‘nozamadb.sql’.
7. Navigate to the query corresponding to the nozamadb.sql file and go to Query->Execute. This will execute the commands in the .sql file and the nozamadb database will now be properly populated.

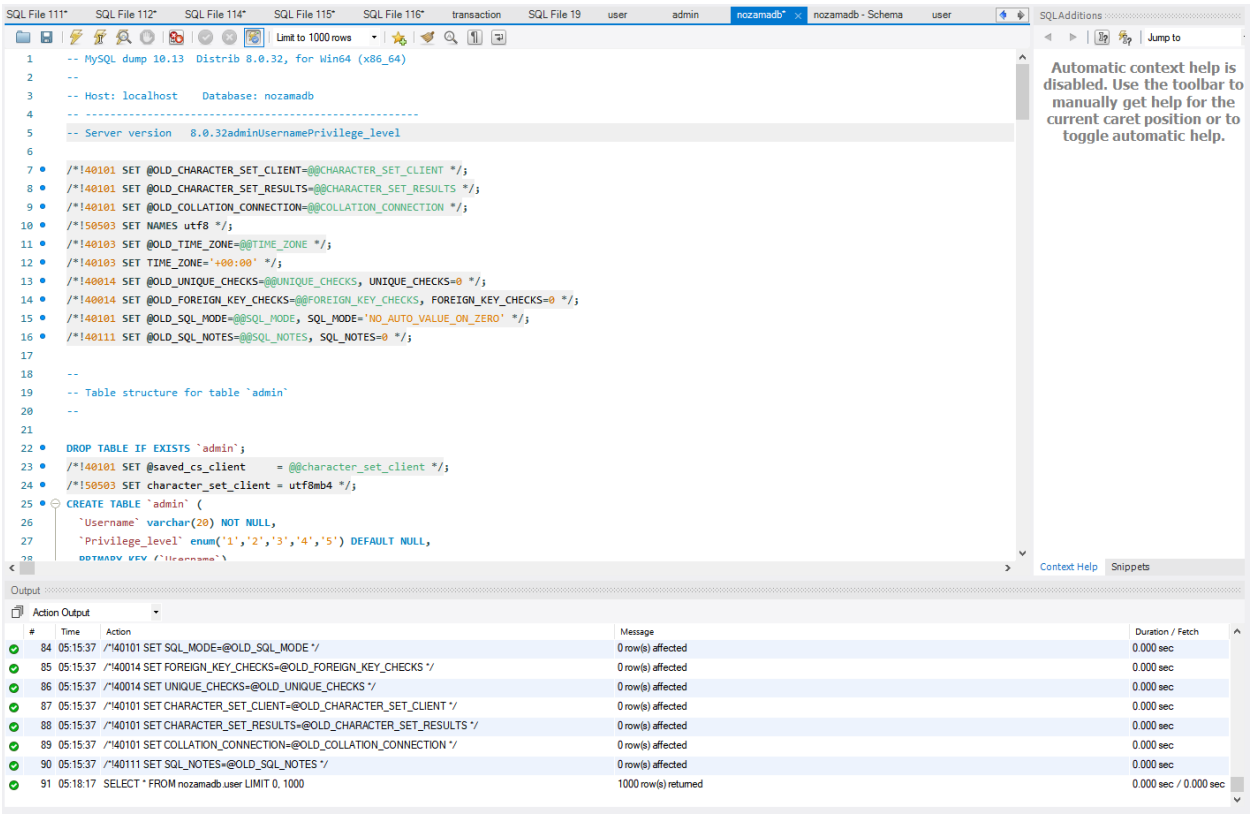
Java Front-End Application:

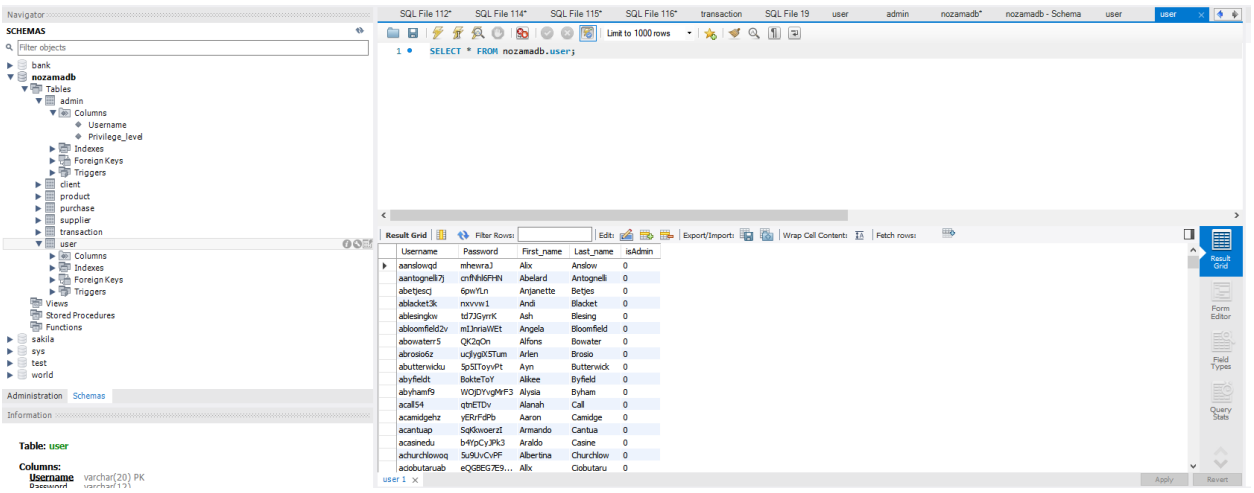
1. Navigate to the ‘project’ folder within ‘nozama-appendix’.
2. Run the ‘mvn package’ command in a terminal to compile the source code to generate a .jar file.
3. Execute the following command from the ‘nozama-appendix’ folder: ‘java -jar target/Nozama-1.0-SNAPSHOT.jar’ in the terminal.
4. The front-end application will start and appear as a terminal window at the login/registration page.

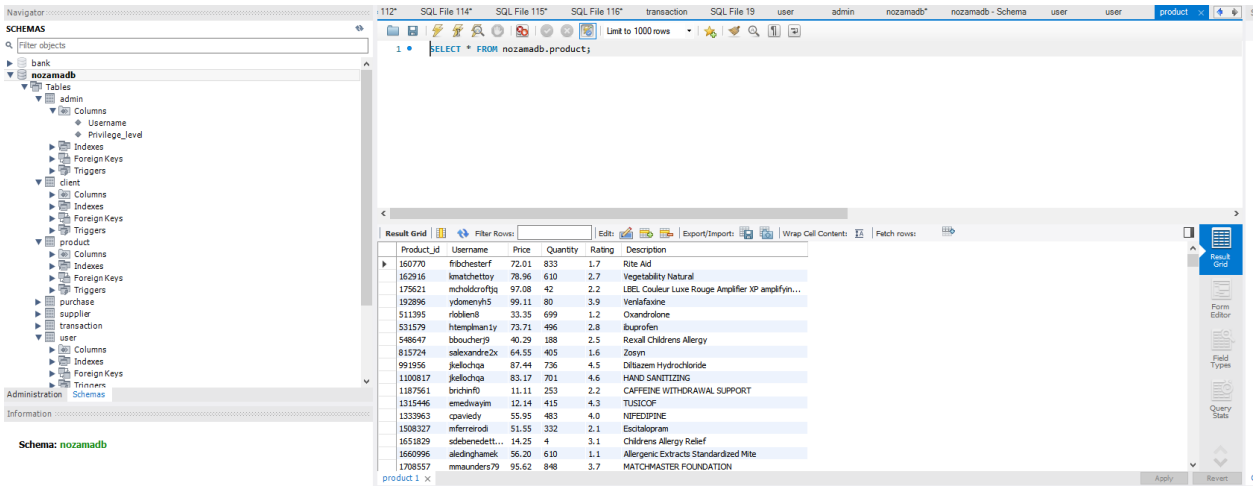
System Installation / General Screenshots:

Database System Setup

The first screenshot shows the nozamadb database being constructed and populated using the nozamadb.sql file and the subsequent screenshots show the database schema and queries for the user and product data.

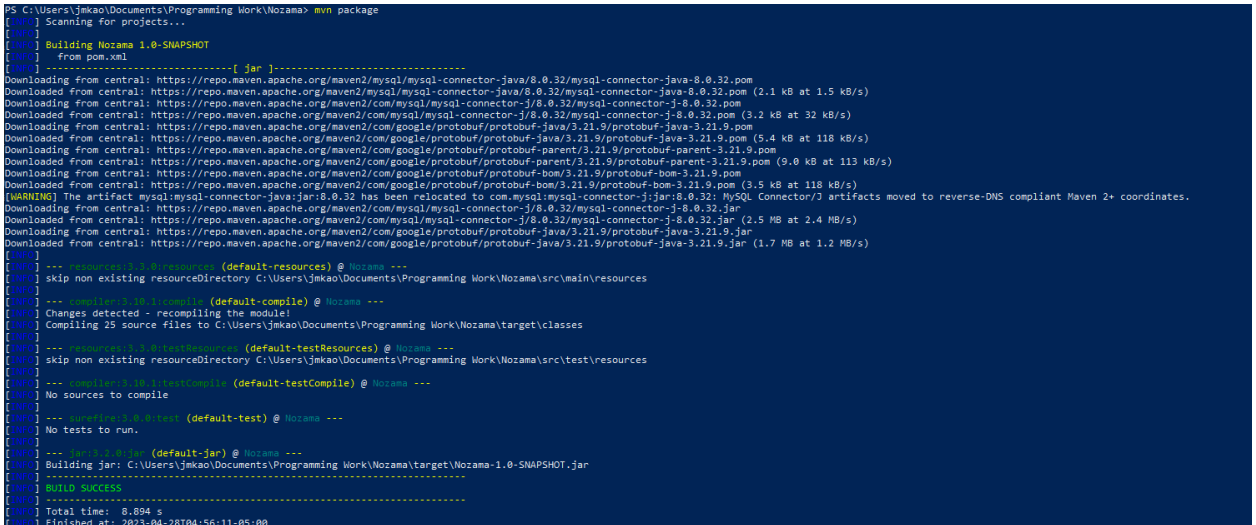

SQL File 111* SQL File 112* SQL File 114* SQL File 115* SQL File 116* transaction SQL File 19 user admin nozamadb* nozamadb - Schema user
Limit to 1000 rows
SQLAdditions
Jump to
Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.
Context Help Snippets
-- MySQL dump 10.13 Distrib 8.0.32, for Win64 (x86_64)
-- Host: localhost Database: nozamadb
-- Server version 8.0.32adminUsernamePrivilege_level
/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!50503 SET NAMES utf8 */;
/*!40103 SET @OLD_TIME_ZONE=@@TIME_ZONE */;
/*!40103 SET TIME_ZONE='+00:00' */;
/*!40014 SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0 */;
/*!40014 SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0 */;
/*!40101 SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='NO_AUTO_VALUE_ON_ZERO' */;
/*!40111 SET @OLD_SQL_NOTES=@@SQL_NOTES, SQL_NOTES=0 */;
-- Table structure for table `admin`
DROP TABLE IF EXISTS `admin`;
/*!40101 SET @saved_cs_client = @@character_set_client */;
/*!50503 SET character_set_client = utf8mb4 */;
CREATE TABLE `admin` (
`Username` varchar(20) NOT NULL,
`Privilege_level` enum('1','2','3','4','5') DEFAULT NULL,
Output
Action Output
Time Action Message Duration / Fetch
84 05:15:37 /*!40101 SET SQL_MODE=@OLD_SQL_MODE */ 0 row(s) affected 0.000 sec
85 05:15:37 /*!40014 SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS */ 0 row(s) affected 0.000 sec
86 05:15:37 /*!40014 SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS */ 0 row(s) affected 0.000 sec
87 05:15:37 /*!40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */ 0 row(s) affected 0.000 sec
88 05:15:37 /*!40101 SET CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */ 0 row(s) affected 0.000 sec
89 05:15:37 /*!40101 SET COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */ 0 row(s) affected 0.000 sec
90 05:15:37 /*!40111 SET SQL_NOTES=@OLD_SQL_NOTES */ 0 row(s) affected 0.000 sec
91 05:18:17 SELECT * FROM nozamadb.user LIMIT 0, 1000 1000 row(s) returned 0.000 sec / 0.000 sec


Navigator
SCHEMAS
Filter objects
bank
nozamadb
Tables
admin
Columns
Username
Privilege_level
Indexes
Foreign Keys
Triggers
client
product
purchase
supplier
transaction
user
Columns
Indexes
Foreign Keys
Triggers
Views
Stored Procedures
Functions
sakila
sys
test
world
Administration Schemas
Information
Table: user
Columns:
Username varchar(20) PK
SQL File 112* SQL File 114* SQL File 115* SQL File 116* transaction SQL File 19 user admin nozamadb* nozamadb - Schema user user
Limit to 1000 rows
1 SELECT * FROM nozamadb.user;
Result Grid
Filter Rows:
Edit:
Export/Import:
Wrap Cell Content:
Fetch rows:
Username Password First_name Last_name isAdmin
aanslowqd mhewraJ Alix Anslow 0
aantognelli7j cnfNhl6FHN Abelard Antognelli 0
abetjescj 6pwYLn Anjanette Betjes 0
ablacket3k nxvvw1 Andi Blacket 0
ablesingkw td7JGyrnK Ash Blesing 0
abloomfield2v ml3rrlaWEt Angela Bloomfield 0
abowaterr5 QK2qOn Alfons Bowater 0
abrosio6z ucJlygiX5Tum Arlen Brosio 0
abutterwicku 5pSITcyvPt Ayn Butterwick 0
abyfieldt BokteToY Alikee Byfield 0
abyhamf9 WOjDYvgMrF3 Alysia Byham 0
acall54 qtnETDv Alanah Call 0
acamidgehz yERrFdPb Aaron Camidge 0
acantuap SqKkwoerzI Armando Cantua 0
acasinedu b4YpCyJPk3 Araldo Casine 0
achurchlowoq 5u9UvCvPF Albertina Churchlow 0
acobutaruab eQGBEG7E9... Alix Clobutaru 0
Result Grid
Form Editor
Field Types
Query Stats
Apply
Revert
user 1



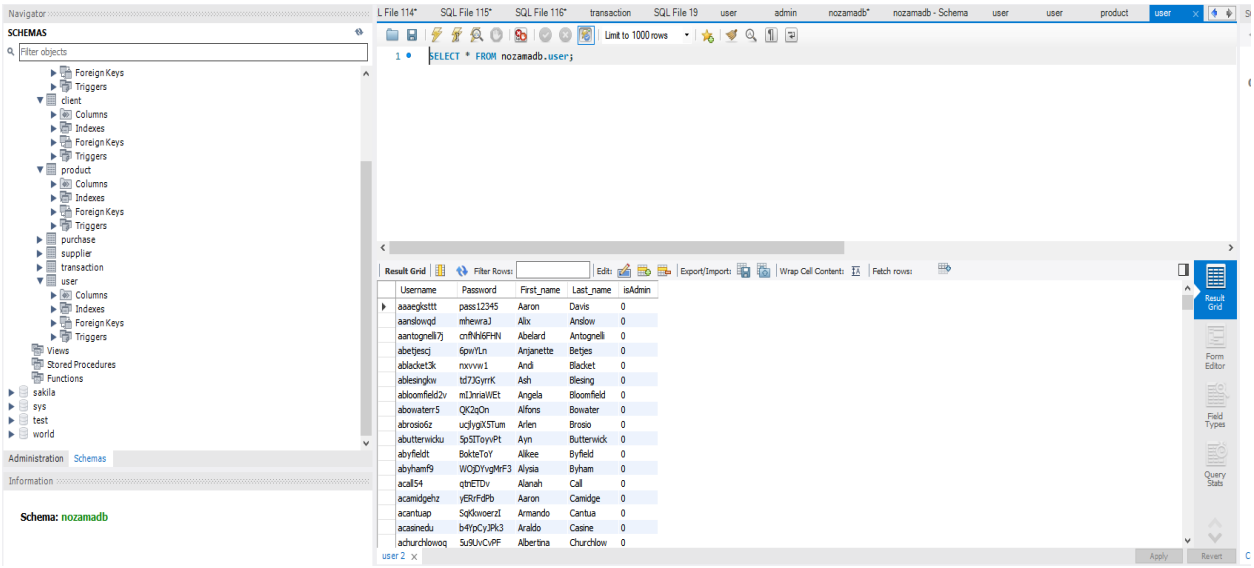
Front-End Setup and Application Start-Up

The following screenshots show how Maven is used to compile and package the Nozama source code into a .jar file and then subsequent execution of the .jar file to start-up the front-end application. Login for a generic user to access the main product page is shown along with registration for a generic user. Then a screenshot of the updated Nozama database with the newly registered user is shown in MySQL Workbench. Detailed descriptions and screenshots for how the rest of the front-end system is used by users are provided in the 'User Application Interface' section of the report.



```
PS C:\Users\jmkao\Documents\Programming Work\Nozama> java -jar target/Nozama-1.0-SNAPSHOT.jar
**********************************************************************************************************************************************************************************************
                                                                                Nozama

1: Login
2: Register
Enter input: 1
**********************************************************************************************************************************************************************************************
                                                                                Nozama

Login Page
Enter username: aanslowqd
Enter password: mhewraJ
**********************************************************************************************************************************************************************************************
                                                                                Nozama

Main Product Page
Current filters: None
No sorting applied

1: Rite Aid
$72.01     Rating:1.7

2: Vegetability Natural
$78.96     Rating:2.7

3: LBEL Couleur Luxe Rouge Amplifier XP amplifying SPF 15
$97.08     Rating:2.2

4: Venlafaxine
$99.11     Rating:3.9

5: Oxandrolone
$33.35     Rating:1.2

6: ibuprofen
$73.71     Rating:2.8

7: Rexall Childrens Allergy
$40.29     Rating:2.5

8: Zosyn
$64.55     Rating:1.6

9: Diltiazem Hydrochloride
$87.44     Rating:4.5

10: HAND SANITIZING
$83.17     Rating:4.6


Enter command('help' for command list): help
**********************************************************************************************************************************************************************************************
```

```
**********************************************************************************************************************************************************************************************
                                                                                Nozama

1: Login
2: Register
Enter input: 2
**********************************************************************************************************************************************************************************************
                                                                                Nozama

Registration Page
Enter username: aaaegksttt
Enter password: pass12345
Enter first name: Aaron
Enter last name: Davis
Enter bank name: Chase
Enter bank number: 93591592151
Would you like to be a supplier?(Y/N): Y
**********************************************************************************************************************************************************************************************
                                                                                Nozama

Main Product Page
Current filters: None
No sorting applied

1: Rite Aid
$72.01     Rating:1.7

2: Vegetability Natural
$78.96     Rating:2.7

3: LBEL Couleur Luxe Rouge Amplifier XP amplifying SPF 15
$97.08     Rating:2.2

4: Venlafaxine
$99.11     Rating:3.9

5: Oxandrolone
$33.35     Rating:1.2

6: ibuprofen
$73.71     Rating:2.8

7: Rexall Childrens Allergy
$40.29     Rating:2.5

8: Zosyn
$64.55     Rating:1.6

9: Diltiazem Hydrochloride
$87.44     Rating:4.5

10: HAND SANITIZING
$83.17     Rating:4.6
```



Additional Queries and Views

SQL Queries

The product listing page will show the product id, description, price, rating, and the supplier username and rating for all products in the Nozama database. These results will be ordered by first product rating descending then by supplier rating descending.

```
1 •  select product_id, Description, price, a.rating as product_rating, b.username as supplier, b.rating as supplier_rating
2    from product a
3    inner join supplier b
4    on a.username = b.username
5    order by a.rating desc, b.rating desc
```

Result Grid

product_id	Description	price	product_rating	supplier	supplier_rating
91959675	Atracurium Besylate	73.12	5.0	lmcbainb0	4.7
89010758	Head and Shoulders 2in1	47.16	5.0	despada6p	4.3
33843764	Antipyrine and Benzocaine	90.82	5.0	mferreirodi	3.6
49341049	NITROGEN	78.41	5.0	salexandre2x	3.5
29427666	Levothyroxine Sodium	39.18	5.0	jkaye1u	2.7
92878970	DAYTIME NIGHTTIME COLD/FLU	4.47	5.0	kmatchettoy	2.4
47004020	Fusarium oxysporum	31.42	5.0	mcholdcroftjq	2.4
45598557	Isopropyl alcohol	34.53	5.0	jpebworthn1	1.5
65490503	Benazepril Hydrochloride	74.69	5.0	gcraker7o	1.4
52176141	mirtazapine	11.43	5.0	mwroughtod	1.4
99192665	berkley and jensen ibuprofen pm	83.49	4.9	badamthwait...	4.8
57056603	Bellagio Sunscreen SPF 30	30.82	4.9	rbenduhnqw	4.8
58197830	IBUPROFEN	44.59	4.9	gpohlakfy	4.7

Result 8

Output

Action Output

#	Time	Action	Message
211	15:52:20	select * from transaction LIMIT 0, 1000	628 row(s) returned
212	15:52:41	select * from purchase LIMIT 0, 1000	1000 row(s) returned
213	15:53:18	select product_id, Description, price, a.rating as product_rating, b.username as supplier, b.rating as sup...	Error Code: 1052. Column 'rating' in order clause is ambiguous
214	15:53:27	select product_id, Description, price, a.rating as product_rating, b.username as supplier, b.rating as sup...	1000 row(s) returned
215	15:53:38	select product_id, Description, price, a.rating as product_rating, b.username as supplier, b.rating as sup...	1000 row(s) returned
216	15:53:47	select product_id, Description, price, a.rating as product_rating, b.username as supplier, b.rating as sup...	1000 row(s) returned

The shopping cart page will display product, price, and supplier details for the current user for any items placed in the cart that have yet to be purchased. This query retrieves this information for the username ‘brevingtongv’.

```
select first_name, last_name, d.product_id, description, d.rating, b.quantity, d.price, d.price*b.quantity as purchase_price, d.username, subtotal, tax, total
from transaction a
natural join purchase b
natural join client c
inner join product d
on b.Product_id = d.Product_id
inner join user e
on c.username = e.username
where b.username = 'brevingtongv'
and status = 'IN_CART'
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

first_name	last_name	product_id	description	rating	quantity	price	purchase_price	username	subtotal	tax	total
Boonie	Revington	33410058	Rye Grass Perennial Standardized	2.3	7	39.95	279.65	slindmanh1	388.60	31.09	419.69
Boonie	Revington	41357167	Clonidine Hydrochloride	1.6	5	21.79	108.95	dstoitedz	388.60	31.09	419.69

Result 22

Output

Action Output

#	Time	Action	Message	Duration / Fe
246	16:25:57	select * from transaction LIMIT 0, 1000	628 row(s) returned	0.000 sec / (
247	16:26:45	select Description, b.quantity, d.price, d.price*b.quantity as purchase_price, subtotal, tax, total from tran...	2 row(s) returned	0.000 sec / (
248	16:28:04	select first_name, last_name, description, b.quantity, d.price, d.price*b.quantity as purchase_price, subt...	2 row(s) returned	0.000 sec / (
249	16:28:28	select first_name, last_name, description, b.quantity, d.price, d.price*b.quantity as purchase_price, d.rati...	2 row(s) returned	0.000 sec / (
250	16:33:00	select first_name, last_name, d.product_id, description, b.quantity, d.price, d.price*b.quantity as purchas...	2 row(s) returned	0.000 sec / (
251	16:33:20	select first_name, last_name, d.product_id, description, d.rating, b.quantity, d.price, d.price*b.quantity as...	2 row(s) returned	0.000 sec / (

The previous purchase page will display the previous purchases of the current user with information regarding the transaction, product, and price. This query retrieves this information for the client ‘Barbey Boucher’.

```
select first_name, last_name, Transaction_id, status, d.product_id, description,
    d.rating, b.quantity, d.price, d.price*b.quantity as purchase_price,
    d.username, subtotal, tax, total, SUBSTRING(bank_number, -5) as CC_4_digits, Date_time
from transaction a
natural join purchase b
natural join client c
inner join product d
on b.Product_id = d.Product_id
inner join user e
on c.username = e.username
where b.username = 'bboucherj9'
and status != 'IN_CART'
order by Transaction_id
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

first_name	last_name	Transaction_id	status	product_id	description	rating	quantity	price	purchase_price	username	subtotal	tax	total	CC_4_digits	Date_time
Barbey	Boucher	6897661	RECEIVED	34531687	Cultivated Rye Grass	1.5	9	4.19	37.71	jcowchafp	37.71	3.02	40.73	0000	2020-10-21 06:08:00
Barbey	Boucher	84671324	ORDER_SUBMITTED	23065038	Metamucil	3.5	8	33.55	268.40	cpaviedy	268.40	21.47	289.87	0000	2022-02-21 21:03:00
Barbey	Boucher	98092343	RECEIVED	67101772	SENSAI FLUID FINISH FF206	4.5	1	87.30	87.30	ydomenyh5	87.30	6.98	94.28	0000	2023-01-07 23:50:00

Result 38

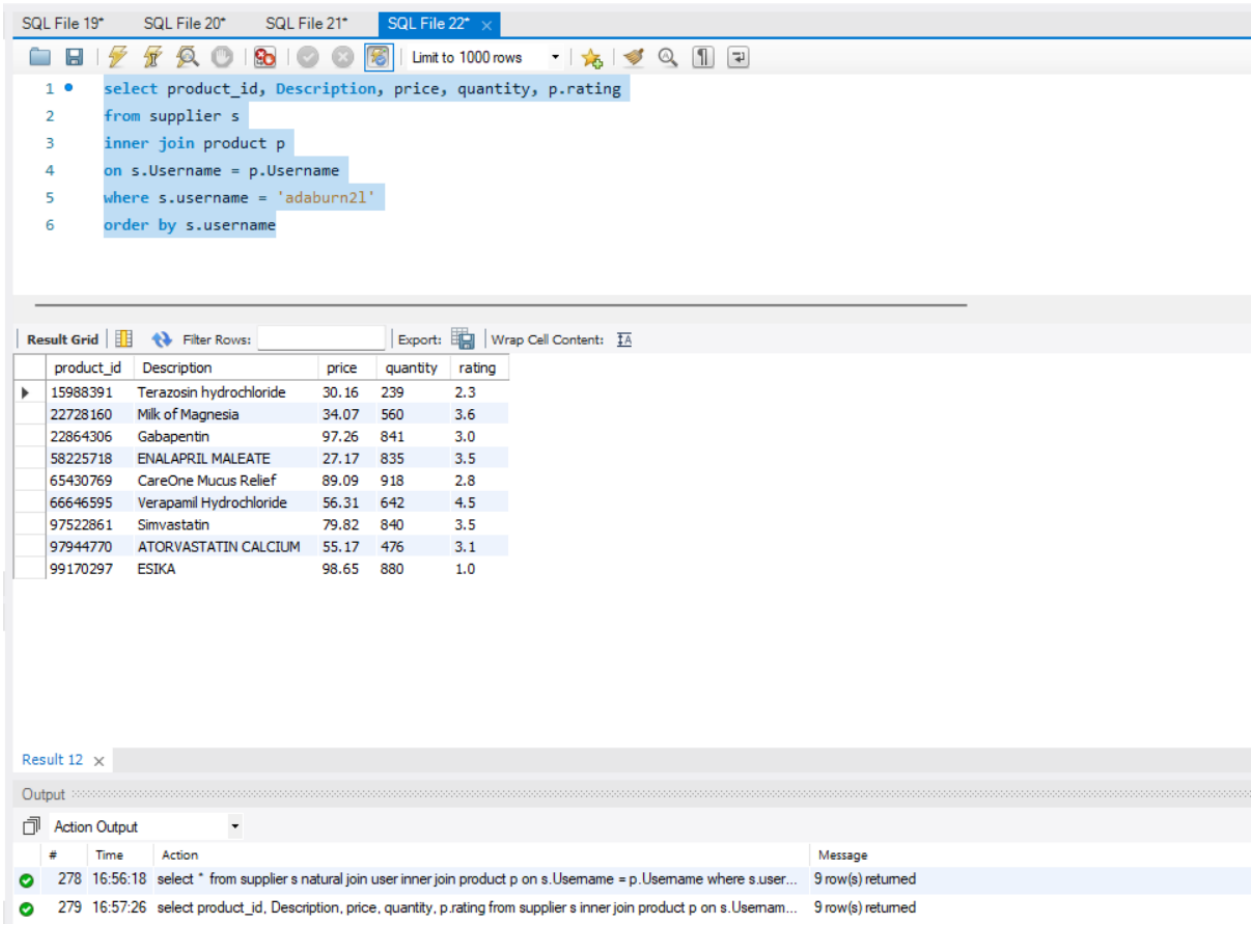
Read Only

Output

Action Output

#	Time	Action	Message	Duration / Fetch
262	16:46:03	select first_name, last_name, Transaction_id, status, d.product_id, description, d.rating, b.quantity, d.pri...	3 row(s) returned	0.000 sec / 0.000 sec
263	16:46:25	select first_name, last_name, Transaction_id, status, d.product_id, description, d.rating, b.quantity, d.pri...	3 row(s) returned	0.000 sec / 0.000 sec

The supplier page will allow suppliers to view their product listings and make changes. This query retrieves the products offered by the supplier user ‘adaburn21’.



SQL Views

The system_info_display view will show general database information such as the count of each entity in the database. This includes the number of total users, the number of users of each user type such as client, supplier, and admin, and inventory information such as the number of products, product purchases, and client transactions in the Nozama database. This view will be used by Nozama users that are admins.

```sql
CREATE VIEW system_count_info AS
SELECT 'Users' AS "Entity Name", COUNT(*) AS Count FROM user
UNION ALL
SELECT 'Clients' AS "Entity Name", COUNT(*) AS Count FROM client
UNION ALL
SELECT 'Suppliers' AS "Entity Name", COUNT(*) AS Count FROM supplier
UNION ALL
SELECT 'Administrators' AS "Entity Name", COUNT(*) AS Count FROM admin
UNION ALL
SELECT 'Products' AS "Entity Name", COUNT(*) AS Count FROM product
UNION ALL
SELECT 'Purchases' AS "Entity Name", COUNT(*) AS Count FROM purchase
UNION ALL
SELECT 'Transactions' AS "Entity Name", COUNT(*) AS Count FROM transaction;

```

```sql
SELECT * FROM SYSTEM_COUNT_INFO
```

Result Grid | Filter Rows: | Export: | Wrap Cell Cont

Entity Name	Count
Users	1000
Clients	990
Suppliers	95
Administrators	10
Products	1000
Purchases	944
Transactions	602

SYSTEM_COUNT_INFO 3

The SALES_BY_STATE view will display business information such as the total number of sales in dollars for transactions that have been processed with the orders being shipped or received. Additionally the total amount of sales in dollars for transactions that have been submitted but have yet to be processed are shown. The results are grouped and ordered alphabetically by state. This view will be used by Nozama users that are admins.

```sql
CREATE OR REPLACE VIEW SALES_BY_STATE AS
SELECT t.state, SUM(t.total) AS total_completed_order_sales, SUM(s.total) AS total_submitted_order_sales
FROM (
    SELECT transaction_ID, state, SUM(total) AS total
    FROM transaction
    WHERE status = 'RECEIVED' OR status = 'SHIPPED'
    GROUP BY transaction_ID, state
) AS t
LEFT JOIN (
    SELECT transaction_ID, state, SUM(total) AS total
    FROM transaction
    WHERE status = 'ORDER_SUBMITTED'
    GROUP BY transaction_ID, state
) AS s
ON t.state = s.state
GROUP BY t.state
ORDER BY t.state;
```

```sql
SELECT * FROM SALES_BY_STATE
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

state	total_completed_order_sales	total_submitted_order_sales
Alabama	9825.64	7130.00
Alaska	375.90	NULL
Arizona	5985.52	12569.70
Arkansas	474.38	585.14
California	218001.00	182420.80
Colorado	15136.72	14461.02
Connecticut	6251.28	4741.90
Delaware	54.35	NULL
District of Columbia	22032.80	17401.56
Florida	156050.40	135460.40
Georgia	18922.68	13461.60
Hawaii	427.96	NULL
Idaho	768.92	NULL
Illinois	8893.58	10645.02
Indiana	5856.66	5256.24
Iowa	966.69	203.82
Kansas	2314.15	1479.85
Kentucky	14806.40	10325.85
Louisiana	1305.19	NULL
Maryland	10469.64	15307.14
Massachusetts	4691.88	5626.02
Michigan	2657.83	NULL
Minnesota	20983.41	13410.48
Missouri	8643.48	11295.76
Montana	101.38	83.59

The SUPPLIER_DISPLAY view will display supplier information such as the username and rating of suppliers along with the number of products each supplier currently has on the Nozama platform. This view will be used by Nozama users that are admins.

```sql
CREATE OR REPLACE VIEW SUPPLIER_DISPLAY as
SELECT s.username as 'Supplier', FORMAT(AVG(s.rating), 1) AS 'Supplier Rating', COUNT(*) AS 'Number of Products'
FROM supplier s
JOIN product p ON s.username = p.username
GROUP BY s.username;
```

```
1 •  SELECT * FROM SUPPLIER_DISPLAY
```

Result Grid | Filter Rows: | Export: | Wrap Cell

Supplier	Supplier Rating	Number of Products
acantuap	3.5	8
adaburn2l	4.1	9
aledinghamek	1.7	12
amathelonqn	4.4	8
badamthwaitel3	4.8	8
baisbett5r	4.9	9
bboldison85	1.4	12
bboucherj9	2.8	10
bfilippovke	1.7	9
bmacgilmartincq	4.7	11
bmarwickka	1.0	12
brichinf0	1.4	14
bshilitonrl	2.2	14

The INVENTORY_DISPLAY view will display the Nozama platform's current product inventory by showing product descriptions, quantities, and suppliers. This view will be used by Nozama users that are admins.

```
1 •  CREATE OR REPLACE VIEW INVENTORY_DISPLAY as
2    SELECT username as 'Supplier', description as 'Product Description', quantity as 'Quantity'
3    FROM product
4    ORDER BY username;
```

```
1 •  SELECT * FROM INVENTORY_DISPLAY
```

Result Grid | Filter Rows: | Export:

Supplier	Product Description	Quantity
acantuap	ORIGINAL POWER REFIRMING AMPOULE	668
acantuap	SHISEIDO THE MAKEUP FOUNDATION	920
acantuap	UltraMax	181
acantuap	ENALAPRIL MALEATE	209
acantuap	Famotidine	814
acantuap	Dexamethasone	796
acantuap	Sani-Foam Instant Hand Sanitizer	79
acantuap	METFORMIN HYDROCHLORIDE	31
adaburn2l	ESIKA	880
adaburn2l	CareOne Mucus Relief	918

RY_DISPLAY 1 ×

User Application Interface

The user interface has been developed as a java project. The general design is based around the various pages the users, admins, and suppliers can navigate too. For each of these pages we've created an

instance of a “page” class which is connected with other files for controlling the pages and the database interactions. The front end interacts with the database using JDBC.

Login/registration page:

Our program starts at the login/registration page. This page prompts the user with a selection of either signing in to an already existing account or creating a new account. If the user has an existing account they can select the regular login option and will be taken to the next prompt. This prompt will ask for their username and password. If the user enters a username that doesn’t exist an error reflecting that will be presented and they’ll be able to try again. The same will happen if their password is wrong. After successfully logging in to the service they will be presented with the main page of the application.

If a user doesn’t already have an existing account they are able to make one using the registration option at the first prompt. When this option is selected they’ll be prompted to provide their username which must be unique from any other existing in the database. This is to prevent the unlikely scenario where two people create an account with the same username and password. The username is between 4-12 characters long. They are then prompted for a password which must be 6-12 characters long. Next they are prompted for their first and last name, bank name, and bank account number. Additionally, they will be able to choose to be a supplier or not. Once this is completed they’re able to be brought to the main page of the application.

Main product page:

The main product page begins by showing the customer the first page of the products. They will be able to visit four locations from here *at any time* by entering the right option to the prompt. (EX: “navigate <page num>”) The pages the user can view are the shopping cart, the previous purchase page, and the user info page. If the user is also a supplier they will be able to access an additional location from this point, the supplier page.

```
Enter command('help' for command list): help
********************************************************************************************************************
                                                                                                          Nozama
Main Product Page
Current filters: Prices lower than 50.0, Ratings greater than 3.0
No sorting applied

Command list

- 'navigate <page_num>'                          Navigates to page page_num.
    Page numbers:
        1: Main Product Page
        2: Shopping Cart Page
        3: Previous Purchases Page
        4: User Info Page
        5: Supplier Page
- 'next'                                         Navigates to the next page of listings.
- 'previous'                                     Navigates to the previous page of listings.
- 'view <product_id>'                            Opens that product.
- 'checkout <product_id> x'                      Add x quantity of that product to the shopping cart.
- 'sortby <order>'                               Sorts products.
    Order options:
        lowprices       Low to high prices
        highprices       High to low prices
        lowratings      Low to high ratings
        highratings     High to low ratings
        alpha           A-Z alphabetical
        clear           Clears order choice
- 'filterby <filter1> <filter2>...'              Filters products by 1 or more filters.
    Filter options:
        (ALL values of x must be decimal, so 4 would have to be 4.0)
        price>x         Price greater than x
        price<x         Price less than x
        rating>x        Rating greater than x
        rating<x        Rating less than x
        clear           Clears filters
```

The product listings page will display 10 products at a time, each with a number associated with it. If the user wants to view the next 10 listings they will enter a command. (EX: “next”) If a user wants to see more about the items beyond the price and name of the item they can enter the command with the product number to view the product description. (EX: “view <product ID>) If they want to add an item to their shopping cart they can enter the associated command with the product number and the number of products they’d like checked out. (EX: “checkout <product ID> <num to add to cart>”). The user can modify the sorting of the products with a sortby command. For example, ‘sortby lowprices’ will give products listed from price low to high; ‘sortby highratings’ will list products by rating high to low. There are also more complex filtering options. For example, the user could sort for all products that are less than $50.00 with a rating higher than 3.0 with the ‘filterby price<50.0 rating>3.0’ command. In this way filters can stack. This was implemented by dynamically building the appropriate query from the user input to allow for maximum combinations.

```
Enter command('help' for command list): filterby price<50.0 rating>3.0
********************************************************************************

Main Product Page
Current filters: Prices lower than 50.0, Ratings greater than 3.0
No sorting applied

1: TUSICOF
$12.14     Rating:4.3

2: Childrens Allergy Relief
$14.25     Rating:3.1

3: No7 Protect and Perfect Foundation Sunscreen Broad Spectrum SPF 15 Calico
$26.54     Rating:3.8

4: Ibuprofen
$42.91     Rating:3.6
```

User info page:

The user can navigate to the user info page from the main page at any time. When they do they'll be presented with their current user info, including their name, username, and bank info. They also may have their rating as a supplier displayed if they are a supplier. From this page the user will be presented with three actions they can perform using listed commands. For all actions on this page the user will first need to provide their password. The first action is to change their password. If the user chooses this option they will be presented with a prompt to enter their current password, then the new password. After that they will be given a confirmation that the action was successful. The second option the user will be presented with is the option to change their name which operates in a very similar way. Lastly they can change their bank info including the bank's name and account number.

The user will be able to exit this page at any time by entering the command for navigating to a different page like the main page, previous purchase page, or shopping cart.

```
                                                                                Nozama
User Info Page

Ryan Bell
Username: rbell
Rating: 0.0
Bank name: Bank of Test
Bank account number: 123456789123


Enter command('help' for command list): help
********************************************************************************
                                                                                Nozama
User Info Page

Command list

- 'navigate <page_num>'                          Navigates to page page_num.
    Page numbers:
        1: Main Product Page
        2: Shopping Cart Page
        3: Previous Purchases Page
        4: User Info Page
- 'changepass'                                   Change your password.
- 'changename'                                   Change your first and/or last name.
- 'changebankinfo'                               Change your bank information.
Press enter to quit.
```

Previous purchase page:

The previous purchase page is accessible from any page by using the 'navigate <page num>' command. From here, the user can navigate to the main page, shopping cart, or the user info page by entering the corresponding navigate command. This page will present a list of their prior purchases grouped by transaction and ordered by date. There is a clear visual grouping of purchases by transaction so the user can see what products were purchased together. It shows relevant details about the transaction such as transaction ID, date, total cost, shipping address, and the transaction status. The transaction status will include: in cart, order submitted, shipped, and received. For this project, we are assuming that all items in a transaction are shipped together. It will also show product information including the product name, product price, quantity, description, and product seller.

```
                                                                    Nozama
Previous Purchases Page

Transaction on 2020-07-04          id: 922
Total: $287.19 (Subtotal: $265.92, Tax: $21.27)
Status: SHIPPED
Shipping address: 14935 Manley Point. New Haven, Connecticut 48705
        8 Headache Relief PM. $29.6 each. Seller: hchater3p
        1 ESIKA. $29.12 each. Seller: sitzhakjs


Enter command('help' for command list):
```

Shopping cart page:

The user can navigate to or from the shopping cart page at any time with the navigate command. From the shopping cart the user can complete a purchase and be directed to the invoice page. In the shopping cart page, the user will be able to view all products added to the shopping cart, the quantity of each product, and relevant information about the product. This information will include the product name, price, seller, rating, and description. The user will be able to modify the cart through several functions. The user can clear the cart which will remove all products currently in the cart with the 'clear' command. The quantity of each item can be adjusted within the cart with 'quantity <productID> n'. Any items can be removed from the cart on this page with 'remove from cart <productID>'. There will be an option to checkout by entering 'checkout'. When this command is entered, the page will display a prompt requesting payment and shipping information and request confirmation from the customer. The payment information will be automatically collected from the database if the customer chooses to pay with their banking account. Otherwise, the customer will be prompted for their credit card number, name, expiration date, and security code if they choose to pay with a credit card. The shipping prompt will include fields for a street address, city, state, zip code, and a free-form section for specific delivery instructions. Upon confirmation, the user will be redirected to the invoice page and the databases will be updated accordingly.

```
                                                                                                    Nozama
Shopping Cart Page

Command list

- 'navigate <page_num>'                                  Navigates to page page_num.
    Page numbers:
        1: Main Product Page
        2: Shopping Cart Page
        3: Previous Purchases Page
        4: User Info Page
- 'remove <product_id>'                                  Removes that product from the cart
- 'quantity <product_id> x'                              Changes that product's quantity to x
- 'checkout'                                             Proceed to payment
- 'clear'                                                Clears the cart
Press enter to quit.

********************************************************************************************************************
                                                                                                    Nozama
Shopping Cart Page

1.
9 Anew Clinical
$38.75  Rating:1.5
Seller: hsandal74

2.
10 Tretinoin
$71.77  Rating:2.9
Seller: ydomenyh5


Enter command('help' for command list):
```

```
Enter command('help' for command list): checkout
Bank account registered. Bank: jcb. Account number: 3573224738798100
Would you like to use this account?(Y/N):
y
Incorrect input. Try again: Y
Enter shipping address: 123 Road St.
Enter city: Anytown
Enter state: wisconsin
Enter zip: 53086
Enter any specific instructions for delivery: No.
********************************************************************************
```

Invoice page:

The invoice page will be displayed on checkout. This page will include relevant product, transaction, and customer data. Product name, price, and quantity will be displayed for all products within the transaction. TransactionID, date, time, and derived fields such as subtotal, tax, and total will be displayed. Customer data, address and relevant bank or credit card information will appear above the product listing. From the invoice page, the user can navigate to the main page, user info page, or the previous purchase page.

```
                                                                                    Nozama
Invoice Page

Transaction on 2023-04-19          id: 669828
Total: $1151.77 (Subtotal: $1066.45, Tax: $85.32)
Status: ORDER_SUBMITTED
Shipping address: 123 Road St.. Anytown, wisconsin 53086
Payment method:
Bank account ending in 0
        9 Anew Clinical. $38.75 each. Seller: hsandal74
        10 Tretinoin. $71.77 each. Seller: ydomenyh5

Enter command('help' for command list):
```

Supplier page:

The supplier page will only be available to users who are also suppliers. If a customer creates a normal account, but would like to become a supplier later they can contact an administrator to do so. The supplier page will only be accessible from the main page. It will serve as the page for suppliers to perform their supplier exclusive duties. The page includes a listing of their products and their information to view. Because of the way we implemented the database they are not able to actually modify any of the products they add. They likewise cannot delete anything, however they do have the option to add a new product. The product information page will show product information including the product name, product price, quantity, and description. From this page a supplier will also be able to see the purchase history of that product. The product purchase history will show relevant details about the transaction such as transaction ID, date, shipping address, last four digits of the credit card number, and the transaction status.

```
--------------------------------------------------------------------------------------------
                                                                                    Nozama
Supplier Page

There are no products

Enter command('help' for command list): add
Enter name: Blue Compass
Enter price: 18.00
Enter stock quantity: 24
********************************************************************************************
                                                                                    Nozama
Supplier Page

1: Blue Compass
$18.0     Rating:0.0
Quantity:24


Enter command('help' for command list):
```

Admin page:

The admin page will only be available to users who are also administrators. It is accessible from the main page. The admin page will be a point of navigation to three different subpages. The first will be a system information page which will contain relevant information about the system including number of total users, supplier users, and regular users in the system, the unique number of products and the total number of products (taking into quantity) in the system, number of items sold per day/per month/per year, and the number of items added by all suppliers per day/per month/per year. The second page will be a page where the administrator can look up any supplier and enter their supplier page with the same editing rights. The third page will be a page for the administrator to look up any user and view and edit their information and user type, as well as access their shopping cart page with the same options.

```
                                                                    Nozama
Admin Page

Command list

- 'navigate <page_num>'                          Navigates to page page_num.
    Page numbers:
        1: Main Product Page
        2: Shopping Cart Page
        3: Previous Purchases Page
        4: User Info Page
- 'systeminfo'                                   Show system information.
- 'viewsupplier <username>'                      Open that user's supplier page.
- 'userinfo <username>'                          Open that user's information page.
- 'viewcart <username>'                          Open that user's shopping cart.

Enter command('help' for command list): systeminfo
Number of clients: 992
Number of suppliers: 98
Number of unique products: 1002
Number of total products: 507281
Average number of products sold per day since 04/03/2020: 5
Average number of products sold per month since 04/03/2020: 140
Average number of products sold per year since 04/03/2020: 1724
Average number of products added per day since 04/03/2020: 457
Average number of products added per month since 04/03/2020: 13710
Average number of products added per year since 04/03/2020: 169094
Press enter to continue.
```

The options that involve going into another person's account are accomplished by using that user's username and password behind the scenes so the system believes it's actually the user. It's because of that that you can only visit one page at a time before returning to the admin page. It helps maintain organization in the front end.

List of User Functions and Associated SQL Implementation:

1. User Login - User login information is retrieved using a select query on the user table for the given username to determine if the given password is correct and which type of user is associated with the account.
2. User Registration - User registration is executed with an insert query into the user table.
3. Product Page Display - The information for the product page display is retrieved using a select query on the product table.
    a. Sort Products – Sorting of the products on the product page is implemented using the 'where' clause in conjunction with the select query on the product table.
    b. Filter Products – Filtering of the products on the product page is implemented using the 'order by' clause in conjunction with the select query on the product table.
4. Specific Product Description Display – The information for the specific product description display is retrieved using a select query on the product table for that specific productID.
5. Add Product to Cart – Adding a product to the shopping cart is implemented using select queries on the transaction and product tables to determine conditionals and insert queries on the purchase and transaction tables.
6. User Info Display – Displaying the user's information is implemented using a select statement on the user table on the associated username.
7. Change User Password - Changing a user's password is implemented using an update SQL query for the associated username.
8. Change Username - Changing a user's name is implemented using an update SQL query for the associated username.
9. Change User Bank Information - Changing a user's bank information is implemented using an update SQL query on the client table for the associated username.
10. Previous User Purchases Display – Previous user purchase information is retrieved using select queries on the transaction and purchase tables for the associated username.
11. User Shopping Cart Display –The user's shopping cart information is retrieved using a select statement on the transaction table for the associated username.
12. Remove Product from Shopping Cart – Removing a product from the shopping cart is implemented using delete queries on the transaction and purchase tables.
13. Change Product Quantity in Shopping Cart – Changing the quantity of a product in the shopping cart is implemented using an update query on the purchase table.
14. Clear Shopping Cart – Clearing the shopping cart is implemented using delete queries on the transaction and purchase tables.

15. Shopping Cart Checkout – The shopping cart checkout option uses select queries on the product, purchase and client tables to determine whether there is enough quantity of the product for the transaction and to gather user banking information. An update query is then used to update the relevant transaction information in the transaction table.
16. Transaction Invoice Display – The transaction invoice display information is retrieved using a select query on the transaction table for the associated username.
17. Supplier User Display – The supplier user's display information is retrieved using a select query on the product table where the username matches the supplier's username.
18. Add Supplier Product - The add supplier product function is implemented using an insert query into the product table.
19. Supplier Product History Display - The supplier's product history information is retrieved using select queries on the purchase and transaction tables for all products associated with the supplier.
20. Admin User Display
    a. Access Supplier User Page – This is implemented using the same SQL query as is used for the normal supplier user page.
    b. Access User Information Page – This is implemented using the same SQL query as is used for the normal access information page.
    c. Access User Shopping Cart – – This is implemented using the same SQL query as is used for the normal user shopping cart page.
21. System Information Display – The information used for system information display is retrieved with several SQL queries which typically use a select statement with an aggregate function such as SUM or COUNT. This information is then further processed into further statistics.

Conclusions and Future Work

With regards to feedback on this database project, designing and implementing the Nozama e-commerce platform was a valuable learning experience for improving our skills in developing user-friendly and efficient database systems. Going through the database design process was useful for understanding the functional and non-functional requirements for an e-commerce platform, and what current standard practices are for the design and development of database systems and user application interfaces. Additionally, we were able to gain practical experience in common database tasks such as drawing ER diagrams, relational database construction, population, and implementation, database normalization and front-end application programming with SQL database connectivity. We also gained insight into the complexities that are involved in creating a good user experience for different types of

users, such as customers, suppliers, and administrators, and we learned how to implement commonly needed features and queries in industry for statistical analysis, searching, filtering, and sorting of database data using SQL, Java and JDBC.

To further improve the project, future implementations could be modified to support more payment options for customers, such as PayPal or other alternative payment options, while more advanced improvements could include implementing a recommendation engine based on user browsing and previous purchases or tuning database queries using index structures. Another possible improvement would be to implement the relational database in a way that allows supplier users to modify and delete their products from the database. One potential implementation of this feature would be to have the front-end application only retrieve information for "active" products that represent the most recent versions of Nozama products. Previous versions of the product that have been deleted or modified by their suppliers would then be stored in the database as historical data. This would provide more control and flexibility for suppliers, while ensuring the integrity and consistency of previous transactions and purchases within the database.

Overall, this project provided a solid introduction to the database systems development process and general e-commerce platform design and implementation which will be useful for our general database understanding and ability to develop more complex database systems in the future.

References

[1] R. Elmasri and S. B. Navathe, Fundamental Database Systems, 7th ed. Boston, MA, USA: Pearson Education, Inc., 2016.