COMP 3700 Project 1

Tripp Isbell
cai0004@auburn.edu
Github

1 User Story: Add Product

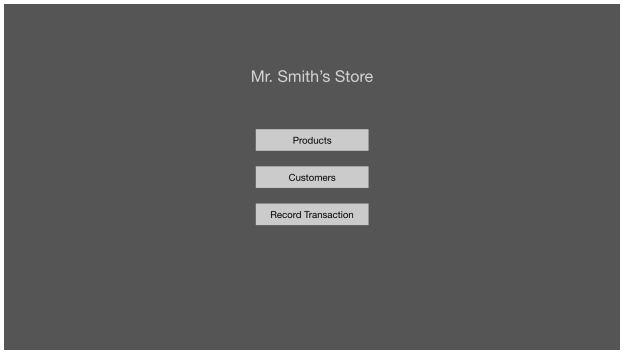
Use Case: add a product into the system
Actors: employees
Goals: update database to include new product
Related use cases: adding a customer to the database or recording a transaction (below) **Pre-conditions:** interface is functional and connected to underlying database
Postconditions: The product database is updated with the item
Steps:

(1) the user clicks a button to display the product database
(2) the system displays the database
- the user clicks add product

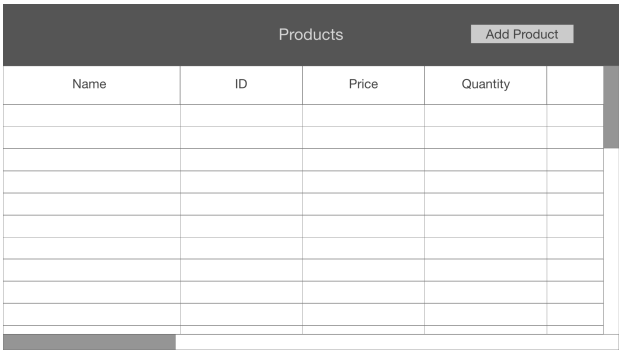
(3) the system displays a screen with text fields for the product info
- the user enters the information and clicks an add button

(4) the system updates the database and displays a confirmation message
- the user clicks confirm

(1) the system returns to the main menu

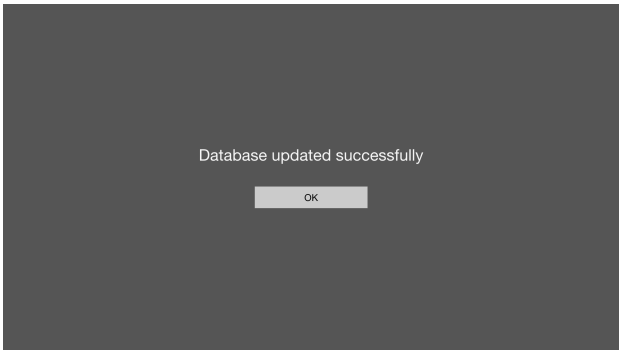


(1)



(2)

(3)



(4)

Use Case: update a product in the system

Actors: employees
Goals: update a product in the database
Preconditions: the target product exists in the database
Postconditions: the database is updated with desired changes
steps:

(1) the user clicks a button to display the product database
(2) the system displays the database
- the user (double) clicks on the desired field to edit
- the system responds by making the field editable
- the user enters desired changes and presses enter
- the system updates the underlying database as well as the GUI

the (1) and (2) views here are the same (1) and (2) views on the previous page

SQL code to create table, insert, update, and delete

```
CREATE TABLE IF NOT EXISTS Product (
   Barcode integer PRIMARY KEY,
   Name text,
   Price real,
   Quantity real,
   Supplier text);
INSERT INTO Product (Barcode, Name, Price, Quantity, Supplier)
   VALUES (1, 'iphone', 899.99, 50, 'Apple');
UPDATE Product SET
   Barcode = 1,
   Name = 'iphone',
   Price = 799.99,
   Quantity = 40,
   Supplier = 'Apple'
WHERE Barcode = 1;
DELETE FROM Product WHERE Barcode = 1;
```

2 User Story: Add Customer

Use case: add a customer into the system
Actors: employees
Goals: update database to include new customer
Related use cases: adding a product or transaction
Preconditions, postconditions: Same as above just replace "product" with "customer"
Steps:

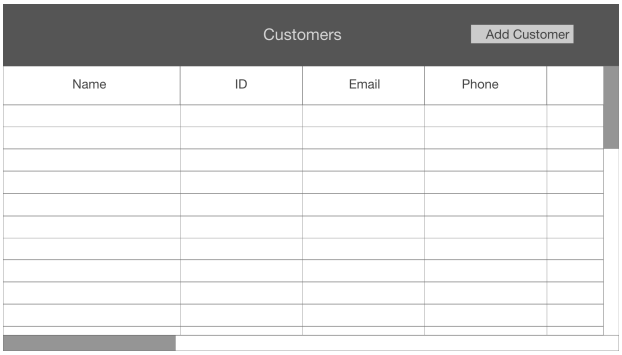
(1) the user clicks a button to display the customer database
(2) the system displays the database
- the user clicks a plus button

(3) the system displays a screen with text fields for the customer info
- the user enters the information and clicks an add button

(4) the system updates the database and displays a confirmation message
- the user clicks confirm

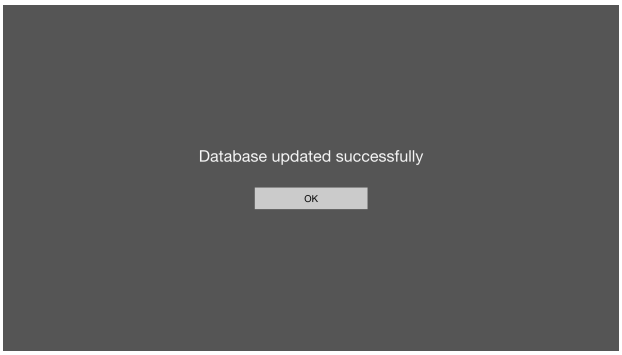
(1) the system returns to the main menu

(1)



(2)

(3)



(4)

Use Case: update a customer in the system
Actors: employees
Goals: update a customer entity in the database
Preconditions: the specific customer exists in the database
Postconditions: the database is updated with desired changes
steps:

(1) the user clicks a button to display the customer database
(2) the system displays the database
- the user (double) clicks on the desired field to edit
- the system responds by making the field editable
- the user enters desired changes and presses enter
- the system updates the underlying database as well as the GUI

the (1) and (2) views here are the same (1) and (2) views on the previous page

SQL code to create table, insert, update, and delete

```
CREATE TABLE IF NOT EXISTS Customer (
   CustomerID integer PRIMARY KEY,
   Name text,
   Email text,
   Phone text,
   Address text,
   PaymentInfo text);
-- Register a new customer named Alice
INSERT INTO Customer (CustomerID, Name, Email, Phone, Address, PaymentInfo)
   VALUES (1, 'Alice', 'a@gmail.com', '555-5555', '1 Main St', 'credit');
-- Alice wishes to change her identity
UPDATE Customer SET
   CustomerID = 1,
   Name = 'Alice',
   Email = 'a@outlook.com',
   Phone = '666-6666'
   Address = '1 College St',
   PaymentInfo = 'cash',
WHERE CustomerID = 1;
-- Alice wishes to discontinue our services
DELETE FROM Customer WHERE CustomerID = 1;
```

3 User Story: Add Transaction

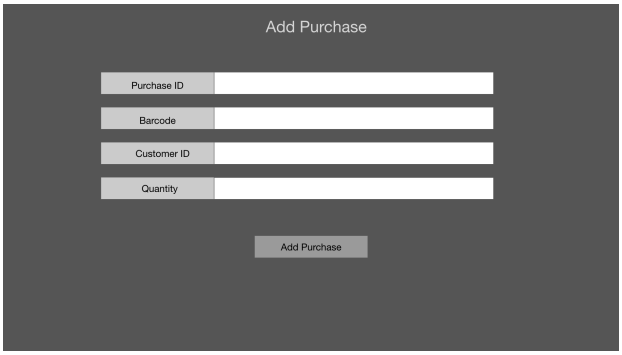
Use Case: record a transaction
Actors: employees
Goals: update database to include transaction
steps:

(1) the user clicks a button to record a transaction
(2) the system displays a screen with text fields for transaction
- the user enters in the information and clicks ok

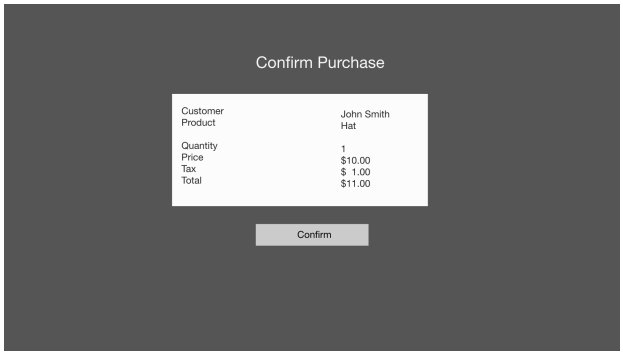
(3) the system displays the receipt with names and prices for confirmation
- the user reviews the information and clicks ok

(4) the system saves the purchase to the database and displays a success message

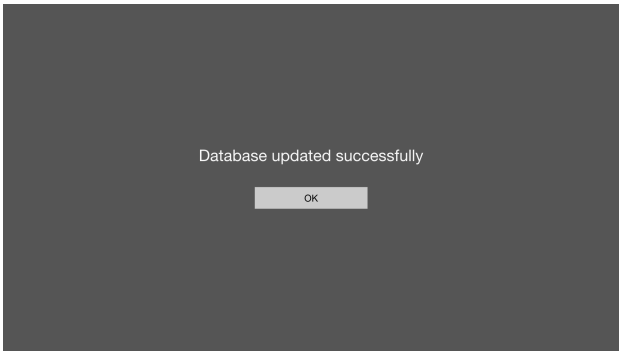
(1)



(2)



(3)



(4)

Use Case: refund (delete) a transaction
Actors: employees
Goals: delete purchase from database
steps:

(1) the user clicks a button to view transaction database
(2) the system displays the purchase database
- the user right clicks (or something) on a purchase to delete it

(3) the system displays the purchase receipt with a button to delete
- the user clicks the delete button

(2) the system removes the purchase from the database and updates the view

Same views as above except the (1) view (main menu) includes a "view transaction" button.

SQL code to create table, insert, update, and delete

```
CREATE TABLE IF NOT EXISTS Purchase (
   PurchaseID integer PRIMARY KEY,
   Date text,
   Barcode integer,
   CustomerID integer,
   Quantity real,
   Price real,
   FOREIGN KEY(Barcode) REFERENCES Product(Barcode),
   FOREIGN KEY(CustomerID) REFERENCES Customer(CustomerID)
   );

-- Price calculated by application based on quantity and product
-- Date calculated based on current date
INSERT INTO Purchase (PurchaseID, Date, Barcode, CustomerID, Quantity, Price)
   VALUES (1, '10/16/2019', 1, 1, 5, 999.99);

UPDATE Purchase SET
   PurchaseID = 1,
   Date = '10/16/2019',
   Barcode = 1,
   CustomerID = 1,
   Quantity = 10,
   Price = 1999.98,
WHERE PurchaseID = 1;

-- Issue a refund[1]
DELETE FROM Purchase WHERE PurchaseID = 1;
```

[1]Disclaimer: the system does not actually issue refunds, the actual transfer of payment must be carried out by the employee operating the system