Database Description

The database is stored in a MySQL database and currently stored on Humber College's Apollo Linux server. The tables themselves are all created with SQL scripts that can be moved to another active Linux Server if necessary. The database consists of two main tables with three subsidiary tables that create entries based on the different functionality of the system.

The main tables are 'accountsJRC' used for storing account information of each user and 'Inventory' for holding data for each item registered in the system. Every account has a unique character string (UID) tied to each entry acting as the primary key distinguishing each account along with personal data for administrative reference and stored encrypted credentials. All the inventory has a unique number (SID) to identify each item as the primary key, information to identify each item in a human readable form is also available in each entry as well as a quantity value to maintain availability of each item.

The first subsidiary table is 'Transaction' used to maintain every order created by user interaction. Each order has a generated transaction ID (TID) tied to each individual order and gains foreign keys from the two main table's primary keys as foreign keys to identify the user the order is tied to and every item associated with it and the quantity of each. The other fields contain a timestamp for logging interactions and a status field to execute the desired functionality of the system.

The next table is 'ListHead' that is used to save lists so users can use for efficient functionality when creating orders. Each entry has a unique ID (TID) to identify in the system with the rest of the fields storing detailed information about the list being stored. The primary key from 'accountsJRC' is used to keep track of the creator of each entry - this field and the 'course' field can be used to organize lists for more efficient searching by clients.

The last table is ‘ListItems’ used to store the items and quantities associated in each entry of saved list. They are grouped by the TID received from ‘ListHead’ to organize all items in one list. Each item is stored by the SID value associated with the inventory and a quantity desired in the list.

All the tables are interacted through multiple PHP scripts that perform all the functions required for the system’s functionality. They are only accessible when certain fields are given to each particular script. This is done so the fields can be filtered server side to prevent database injections and to control the type of interaction that can be done with the database.

Unit Testing and Production Testing

Initial testing requires every PHP script to execute its intended functions. Each file is tested with web browser and access to the server through SSH connection to monitor the results and output. Afterwards the devices interacting with the scripts (mobile device, NFC Reader, web application) all require the ability to post requests and then ensure each device can post or retrieve the necessary information from each function.

Testing the entire system requires the devices involved to be able to temporarily store the ID of the transaction. Upon interaction with each other it will change the state of each order to communicate to both client and administrator the actions required during each state. Every device interacts with the orders differently through each state. The mobile application puts the transaction into a “P” state - which stands for a pending order. This allows the employee to double check their inventory beforehand and prep the order before the person arrives at the point of service’. The hardware prototype has 2 states: “Q” - which stands for “Queued” and “QR” - which stands for “Queued for Return”. This is activated when the person arrives at the point of service, taps their phone with the application open and sends the NDEF message containing the transaction ID. If the message starts with an ‘R’ that means the person who scanned their phone intends to return their items. To finalize the transaction, the employee at the point of service will need to double check any items coming in or out. Here a web application will be used to set the state of the transaction to “O” - which stands for “Out” and “R” - which stands for returned.

The hardware prototype created for this system contains a check sequence to ensure that the components connected via I2C are active as well as the Wi-Fi/internet connection allowing the device to communicate to the database. If there are issues, a notification would be displayed in the terminal or displayed on the LCD. The check sequence is important to reduce the amount of time for technical troubleshooting and can be used during production testing to speed up the process.

The mobile application is tested through ensuring that it can interact smoothly with the tables in the database and post/receive required information. All functionality in the application is tested individually first, then with each other to ensure data is being controlled as intended. Finally, the application is tested interacting with the hardware to complete the process.

Network and Security

For each device their connection to networks is crucial to setup specifically to control security and ensure no unintended breaches can be made. Currently the hardware is provided with a WiFi certificate in the configuration files to ensure secure connection to the Humber College network. This is important for setup per product to keep the log-in credentials encrypted specifically to each business. Increasing secure network access is essential involving any database that stores data for businesses or personal information. Currently through testing the hardware is assigned a reserved IP address when the MAC address is identified and used to troubleshoot issues via SSH.

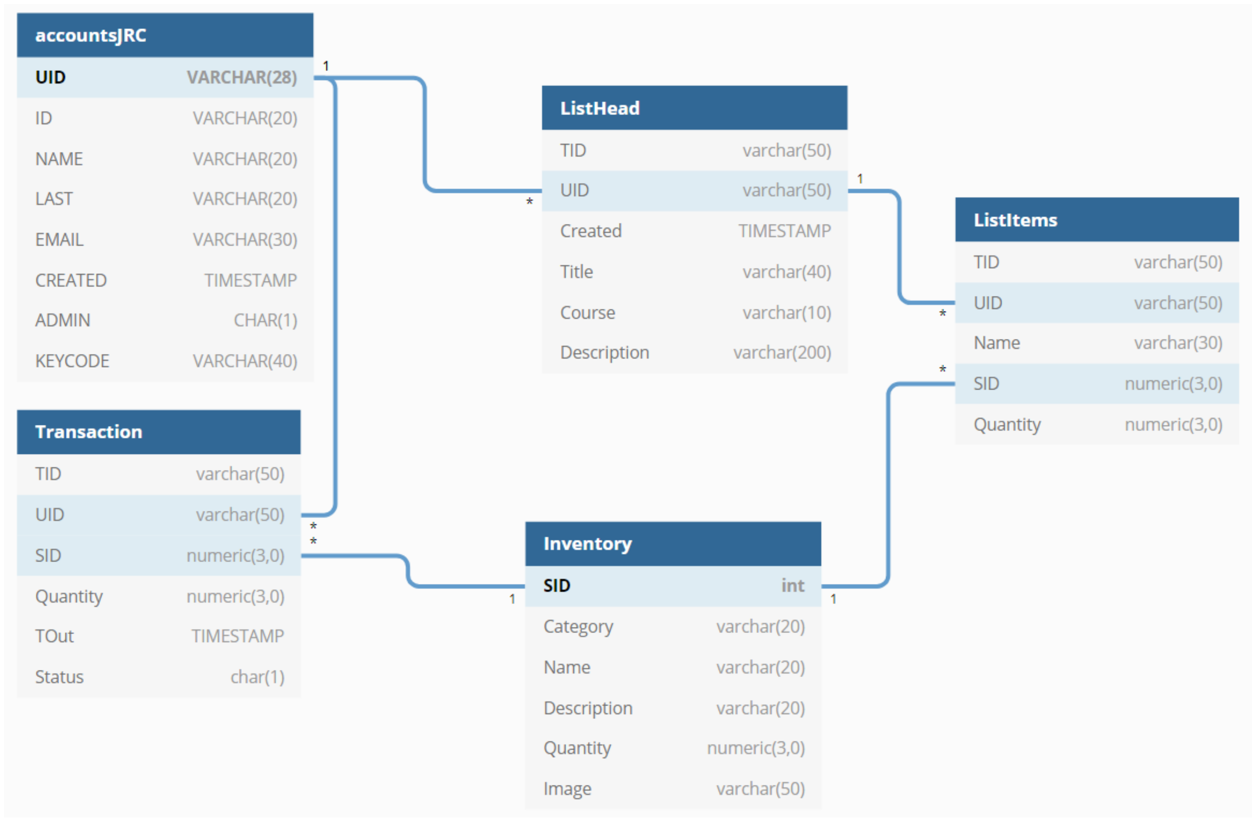
Every interaction from clients must be limited to specifically control access to the database to only what is required. Account information and every transaction is identified with uniquely generated character strings. No access is permitted unless an account is recognized with their corresponding hashed password that is generated and recreated with the PHP scripts stored on the server. To use this product in a business, heavy consideration over security is essential to protect all data used by the respective business.

Currently, the server is only utilizing http which is unsafe as anyone with access to applications similar to Wireshark can see what data is flowing from the hardware prototype or from the mobile application. To increase security for businesses, HTTPS with a SSL certificate must be used to encrypt the data being transferred so that it adds a layer of security that prevents anyone from seeing the data over a network.

Database Description for poster

The database was designed to manage accounts associated with the system (client/administrator) as well as the product/inventory to be used in client interaction. Every entry in the two main tables store unique IDs for every entry that are used as foreign keys in all the other tables to control functionality in the system. Three secondary tables use the main tables to store the data for all functionality intended to be done by the system. The tables can only be accessed through the MySQL account holder or the PHP script that has been designed to query and update the tables when necessary.

The secondary tables are queried for their foreign keys provided by the main tables to conduct transactions for clients, save lists for efficient functionality and store various data fields for client and administrator reference. Additionally, with the foreign keys queries from these tables can retrieve information to be used to access additional information required from the main tables.


accountsJRC
UID VARCHAR(28)
ID VARCHAR(20)
NAME VARCHAR(20)
LAST VARCHAR(20)
EMAIL VARCHAR(30)
CREATED TIMESTAMP
ADMIN CHAR(1)
KEYCODE VARCHAR(40)
Transaction
TID varchar(50)
UID varchar(50)
SID numeric(3,0)
Quantity numeric(3,0)
TOut TIMESTAMP
Status char(1)
ListHead
TID varchar(50)
UID varchar(50)
Created TIMESTAMP
Title varchar(40)
Course varchar(10)
Description varchar(200)
ListItems
TID varchar(50)
UID varchar(50)
Name varchar(30)
SID numeric(3,0)
Quantity numeric(3,0)
Inventory
SID int
Category varchar(20)
Name varchar(20)
Description varchar(20)
Quantity numeric(3,0)
Image varchar(50)
1
*
1
*
*
*
1
1
*