Icon

Description automatically generated

System Design Document

<Joy’s Toys>

<Seth White>

CSC-289 Programming Capstone Project

Guilford Technical Community College

Professor Scott Ilkenhons, MS

|  |  |
| --- | --- |
| Document ID | SYSTEM DESIGN-v0.1 |
| Version Number | 0.1 |
| Issue Date | Friday, November 4, 2022 |
| Classification | Private |

**Table of Contents**

[List of Tables 4](#_Toc134364643)

[List of Figures 5](#_Toc134364644)

[1 Introduction 7](#_Toc134364645)

[1.1 Purpose of Document 7](#_Toc134364646)

[1.2 Acronyms and Abbreviations 7](#_Toc134364647)

[2 SWOT Analysis 8](#_Toc134364648)

[3 Database Design 9](#_Toc134364649)

[4 Application/Website Detailed Design 17](#_Toc134364650)

[5 Network and Physical Architecture Design Overview 23](#_Toc134364651)

[5.1 Background Information 23](#_Toc134364652)

[5.2 System Evolution Description 23](#_Toc134364653)

[5.3 Disaster Recovery Plan 24](#_Toc134364654)

[6 Training Plan 25](#_Toc134364655)

[7 Future Considerations 26](#_Toc134364656)

# List of Tables

2.1 [SWOT Analysis](#_SWOT_Analysis)

# List of Figures

3 [Database Design](#_Database_Design)

3.2 Database Function

4 [Application Design](#_Application/Website_Detailed_Design)

Figure 4.1

Figure 4.2

Figure 4.3

Figure 4.4

Figure 4.5

Figure 4.6

Figure 4.7

Figure 4.8

5 [Network and Physical Architecture Design Overview](#_Network_and_Physical)

6 [Training Plan](#_Training_Plan)

7 [Future Considerations](#_Future_Considerations)

# Introduction

This is the Software Design Document for the development of the Web Application for Joy’s Toys. Refer to this document for the list of concerns that will be/have been addressed.

## Purpose of Document

This SDD will identify opportunities that our application will address for the business *Joy’s Toys.* Throughout the development process this document will serve as a source of guidance. This document divides itself into a SWOT Analysis, Database Design, Application/Website Detailed Design, External Interface Design, an overview of Network and Physical Architecture Design, and a Training Plan.

The SWOT Analysis identifies the Strengths, Weaknesses, Opportunities, and Threats of our application.

The Database Design provides an abstract overview of our database.

The Application/Website Detailed Design will contain an abstract overview/layout of the design of the application.

The External Interface Design section will describe the architecture and design of the External Interface.

The overview of the Network and Physical Architecture Design will address the hardware requirements and network design and set up for this application.

Finally, the Training Plan will include a structured plan to train employees on how to use this system. Training will be broken down into sessions that employees will attend.

The system name is Joy’s Toys. The app will be identifiable through key features of Inventory Tracking, Customer Account creation and management as well as access for customers to product availability, Organization for Employees, eCommerce, and a system to access and modify financial data for specified employees.

## Acronyms and Abbreviations

|  |  |
| --- | --- |
| GUI | Graphical User Interface |
| SDD | System Design Document |
| ERD | Entity Relationship Diagram |
| APP  CRUD  APP  DTO | Web/Mobile Application  Create, Read, Update, Delete  Application Programable Interface  Data Transferrable Object |
|  |  |

# SWOT Analysis

|  |  |
| --- | --- |
| Strengths   * Good reputation among local community * Familiarity with customer base | Weaknesses   * Organization/Structure * Tracking * Accessibility for customers and employees |
| Opportunities   * Inventory tracking * eCommerce(selling, buying, etc) * Daily Balancing(reconciling credits and debits) * Payroll * Accounts Receivable * Accounts Payable * Order forms * Customer receipt generator * Customer Request form * New Customers * Memoization for employees * Customer Accounts (online accounts, buying history, etc) | Threats   * Other online competitors * Supplier shortages/lack of storage space * Losing employees to other businesses |

# Database Design

\*Entity Relationship Diagram\*

Diagram

Description automatically generated

This ERD details the tables and illustrates the relationships between these tables. This will be the overall organization of Joy’s Toys database thus far.

The tables involved are:

Customer

Payment Information

Customer\_Order

Customer\_Transaction

Customer\_Order\_Product\_Association

Products

Categories

Inventory\_Order

Inventory\_Transaction

Restocking\_Logistics

Delivery

Supplier

Employee

Job\_Role

The database was created using PostgreSQL and managed using the PgAdmin application.

A Customer is a defined object for a customer that will be able to access the app, which will in turn use information that the Customer will provide to the database for purchases and profile purposes.

A customer’s payment information will be stored in the Payment Information table, which will be related to Customer through a foreign key in the Payment Information table.

The Products table contains all of the products that will be offered by Joy’s Toys. It will contain the name, category name, and a file location for the image of the product among other relevant bits of information. The file location in the database will be served to the Client using a file server. For this application, we have decided to go with Apache using XAMPP as a file server.

Customers can have multiple Orders, but only one Customer per Order. Orders may also contain many Products, and many Products can be in many Orders. This will be accomplished through an Associative Entity (or Junction table), called Customer\_Order\_Product\_Association.

For the purpose of simplicity, one Category can contain Many Products and a Product can only be in one Category.

Customer\_Order contains the order information of one particular order. You can only have one Customer\_Order at a time. Customer\_Order contains the list of products ordered, along with the total amount of the order, and a timestamp of when the order was placed.

Once a Customer\_Order is complete, a Customer\_Transaction is generated for the customer. Only one Customer\_Transaction is generated per order, which will contain the Order ID number along with Payment Information. Customer\_Transaction will serve the purpose of being the table used to generate a receipt for the customer, which should be displayed to the screen once a transaction between the Joys Toys App and the Customer has been made.

A Many-to-Many relationship is required between Products and Inventory\_Order, as One Inventory\_Order can contain Many Products and One Product can be ordered Many times.

Therefore, another junction table called Restocking\_Logisitics was created. This junction table manages the relationship between Inventory\_Order and Products, with control given to Inventory\_Order. The junction table also contains an additional column that maintains the quantity of a given Product that is ordered.

Inventory is tracked through the Products and Inventory\_Order table. Once a Product decrements to a certain amount in Quantity the designated Employee can make an order for more inventory. There will be logic provided in the backend to send a notification to the designated employee who is in charge of placing orders for stocking. Inventory\_Order will also contain an ID, an order date, the purchasing price per unit, and the Supplier\_ID.

The Inventory\_Order table is linked by foreign key to the Delivery table. The Delivery table contains an ID, a Supplier ID, the expected delivery date, and the aforementioned Inventory order ID. To add clarification, it is the Supplier who ultimately controls the delivery date, so this will require employees to make sure that this information is put into the table once they have received order confirmation from a Supplier.

The Supplier table will be linked by foreign key to the Delivery table. The Supplier table will contain a name, address, phone number, and email of the Supplier.

An Employee will contain standard information about an employee, such as name, address, phone number, email, bank account number, etc…

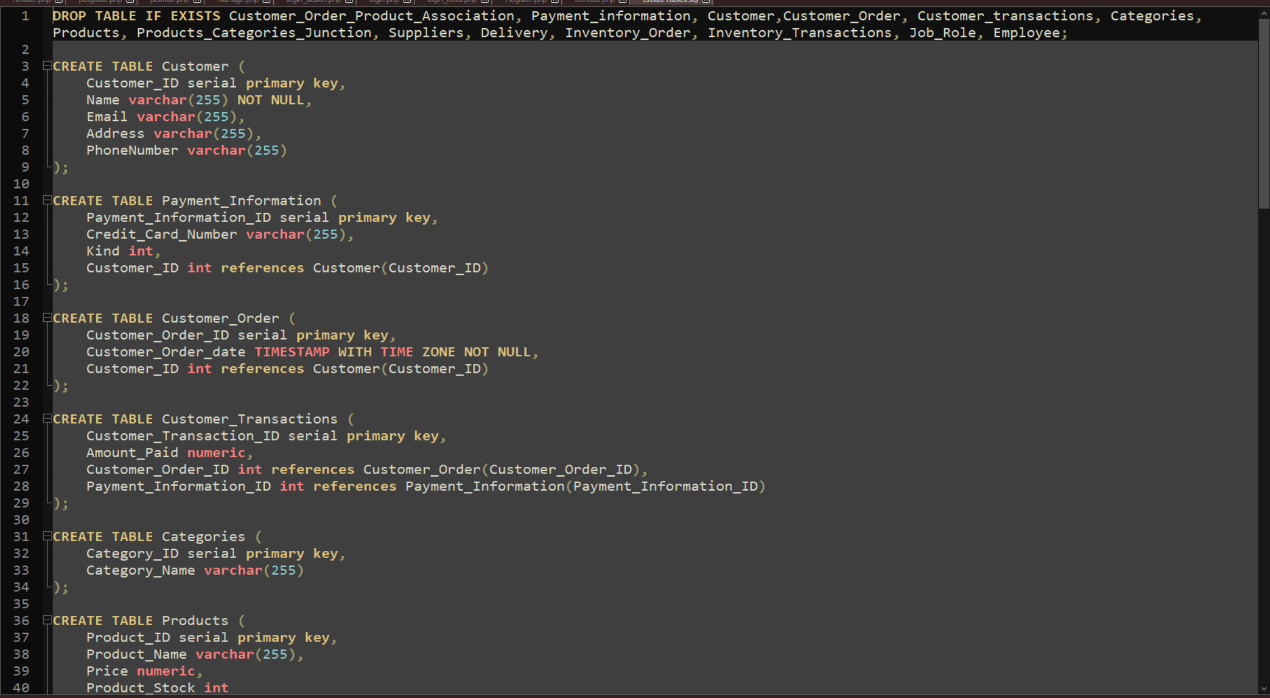
The number of hours worked by a particular Employee will be part of the Employee object as well. One Employee may only have one Job Role, however a specific Job Role can be assigned to many different Employees.

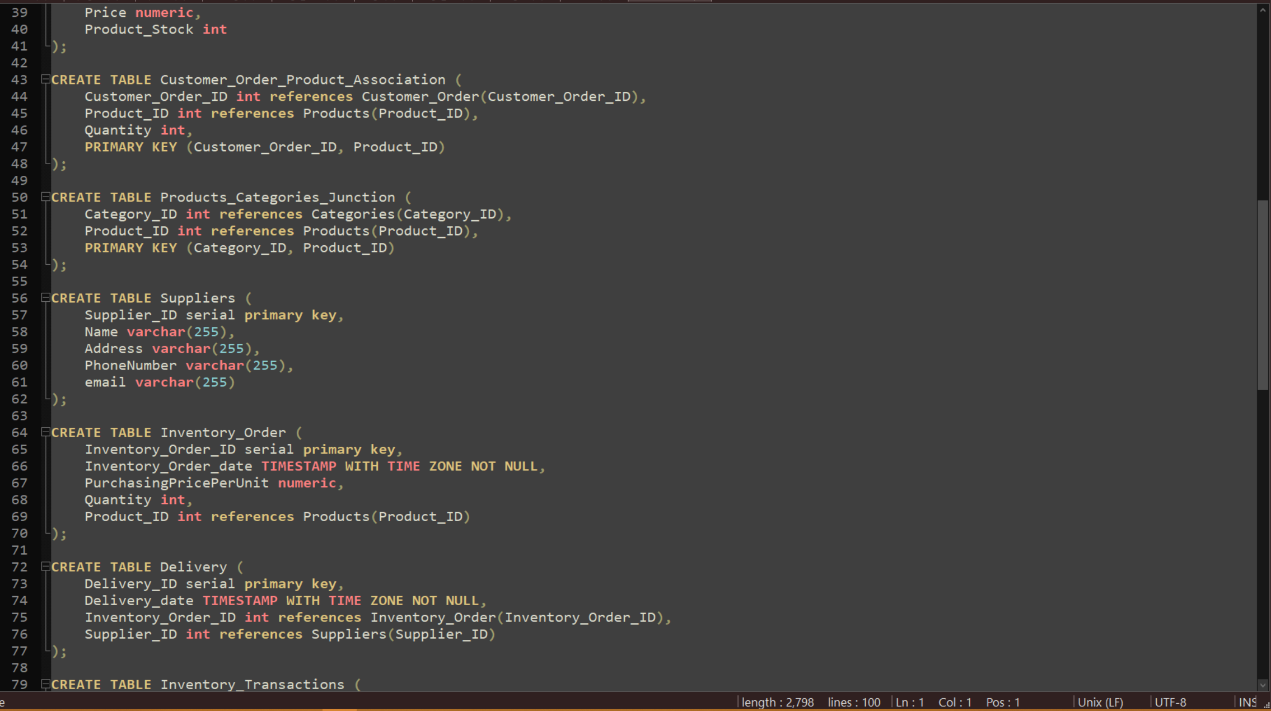
A specific Employee is granted access to complete an Inventory\_ Order. These are transactions that order more inventory of one specific item from a specific Supplier. Once an Inventory\_Order is complete an Inventory\_Transaction is generated that corresponds to that particular Inventory\_Order. Only one transaction per one order. Along with the Inventory\_Transaction, a Delivery\_date will be assigned from the particular Supplier and stored in the Delivery object. Only one supplier can supply one Delivery\_date per order.

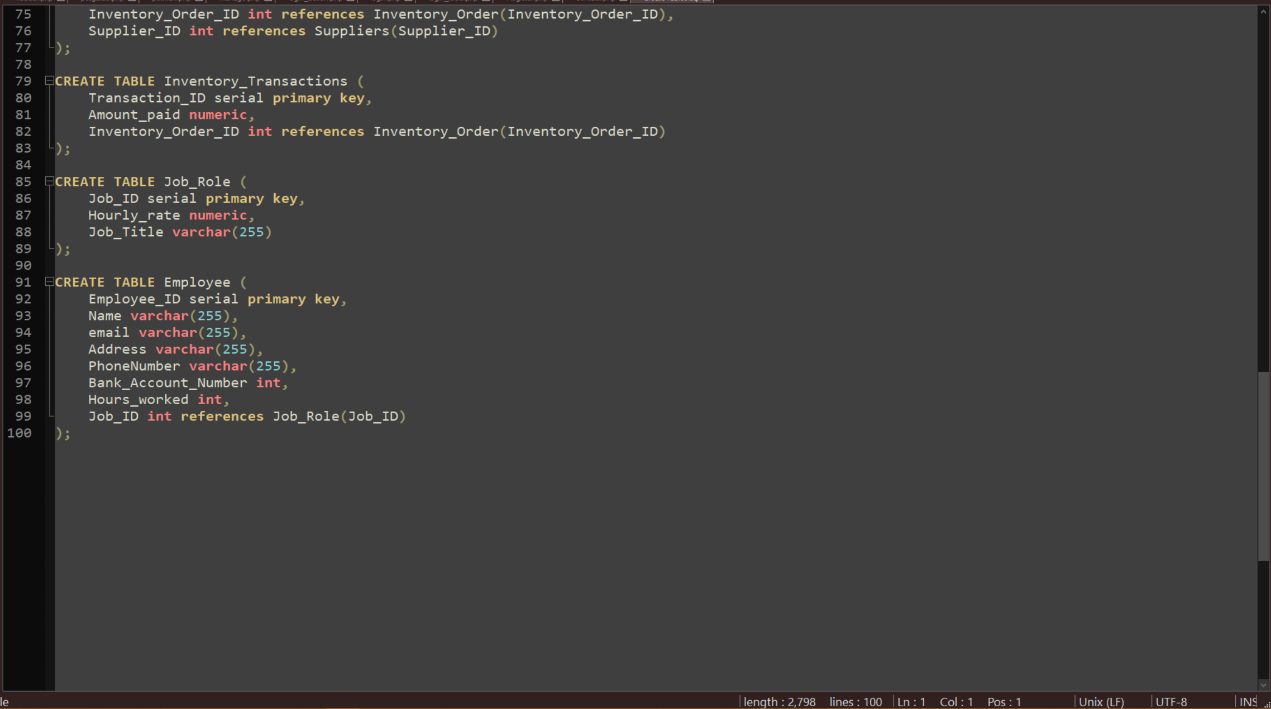
3.2 Database Function:

Below are images of the function and design of the database.

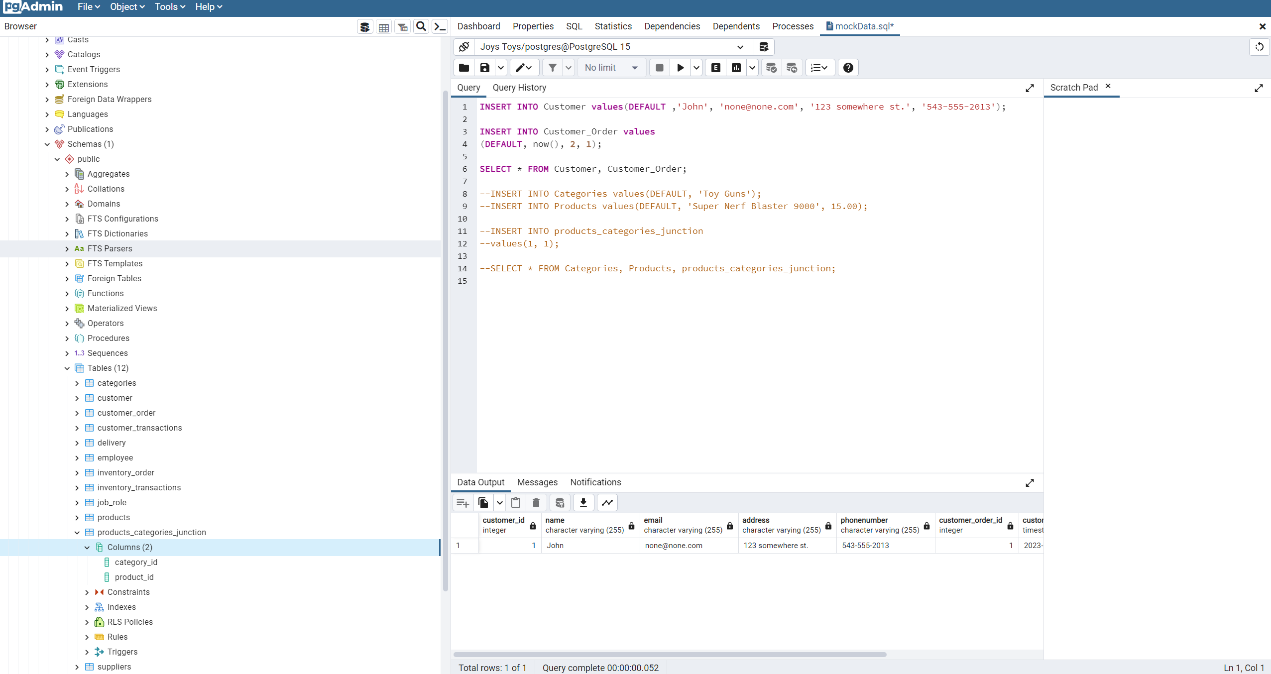
Table Creation:





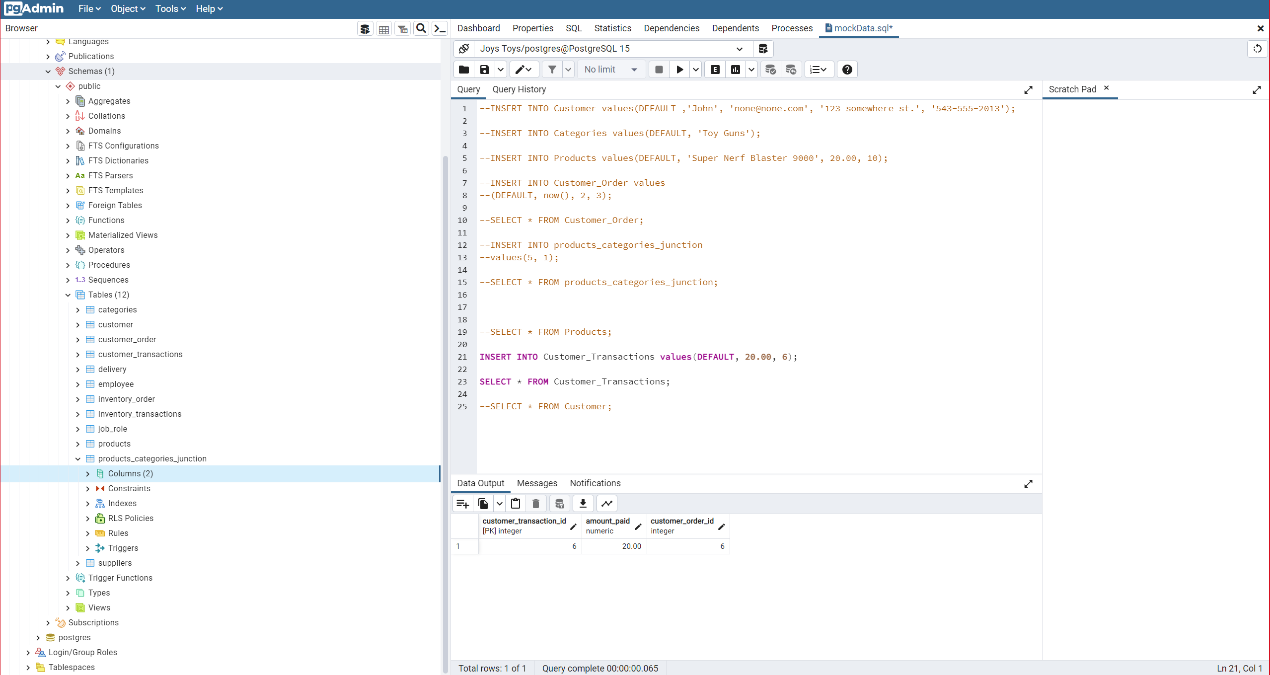


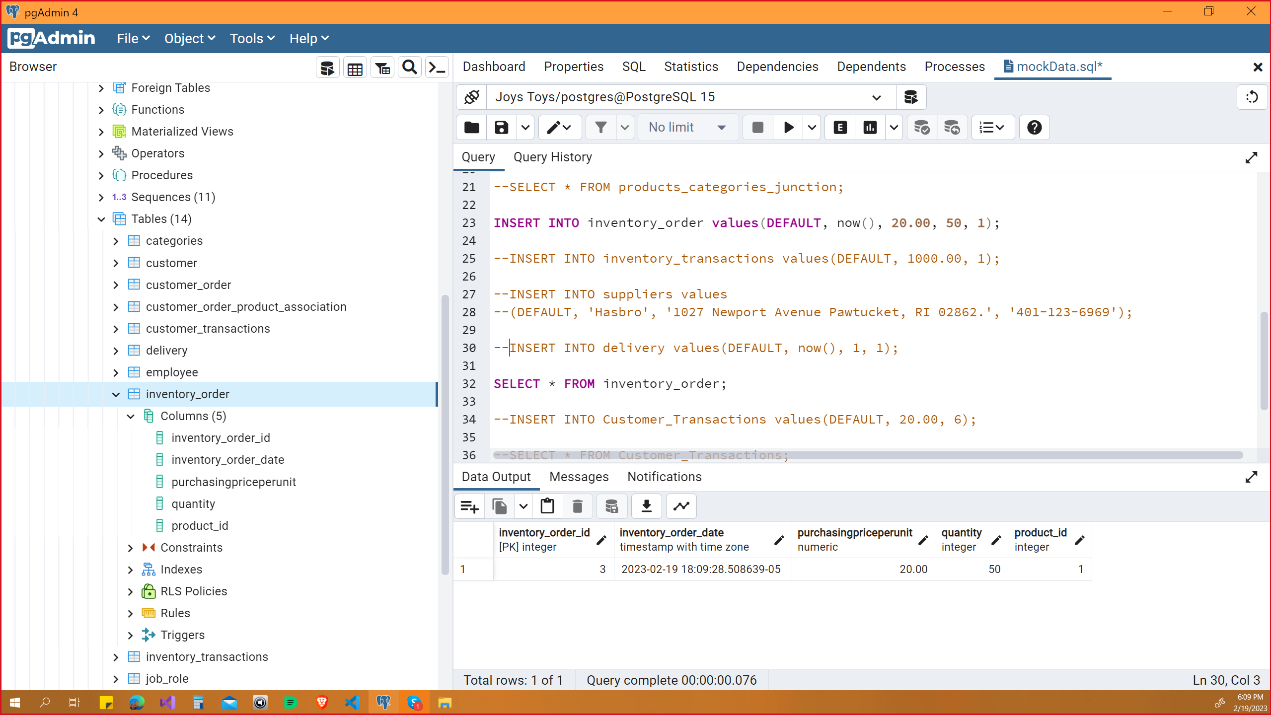
Customer and Customer\_Order:



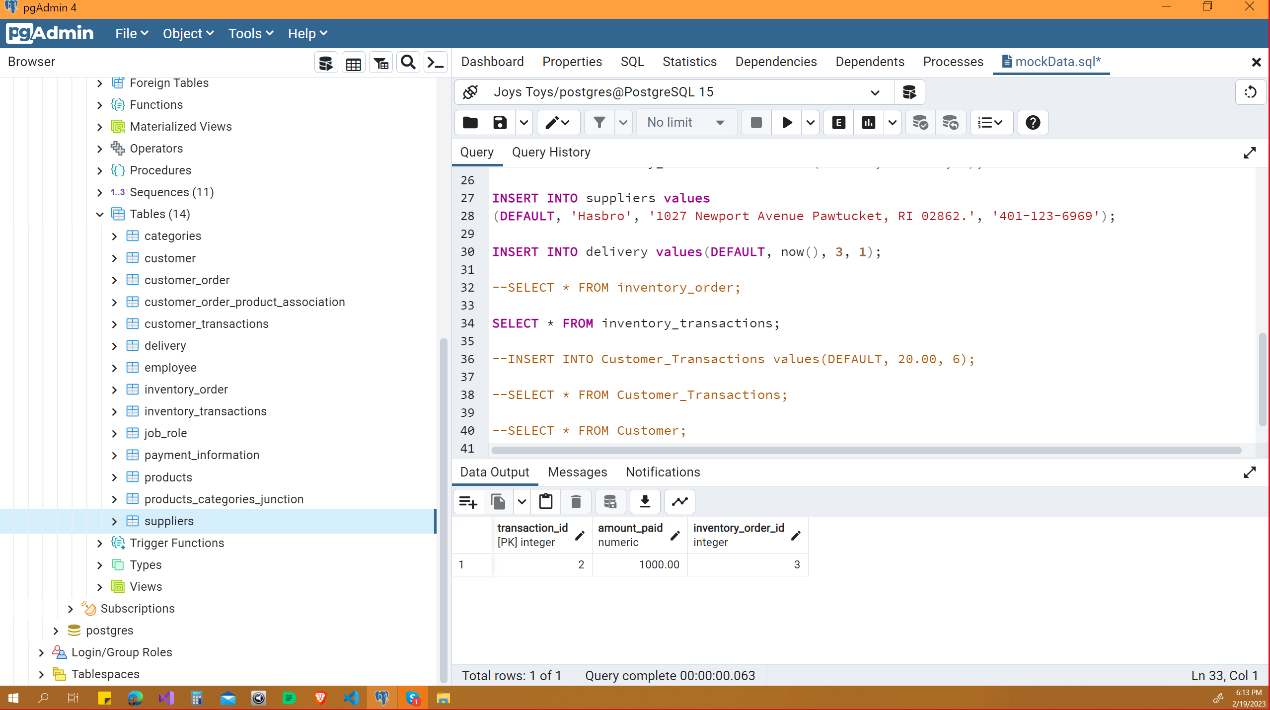
Customer\_Transaction:

Graphical user interface, application, Word

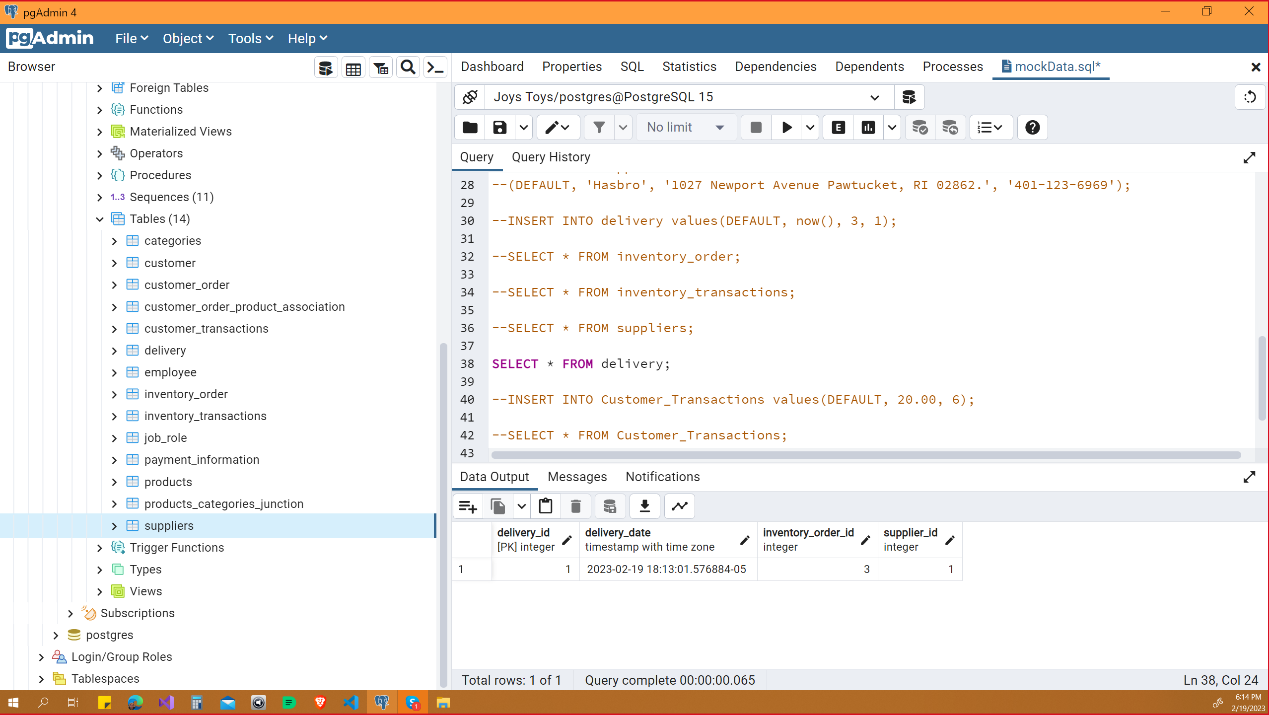
Description automatically generatedRestocking\_Logistics:

Inventory\_Order:

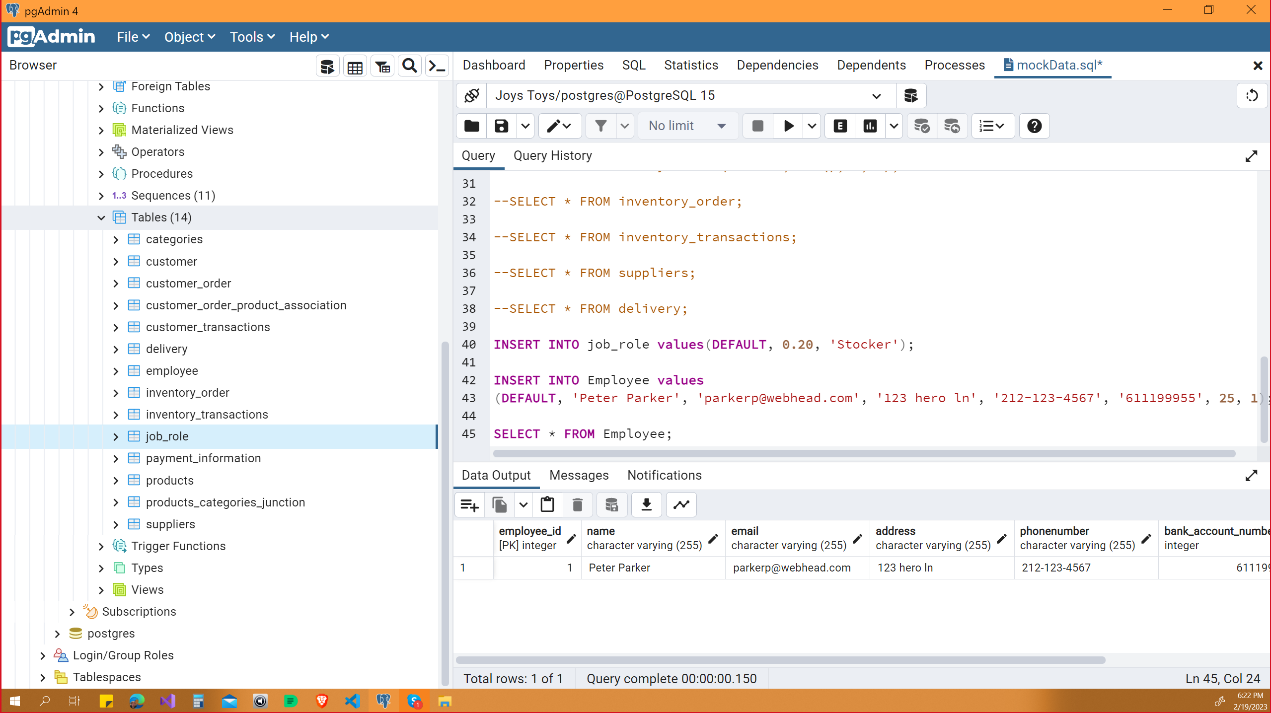
Inventory\_Transaction:



Delivery:



Employee and Job\_Role:



# Application/Website Detailed Design

The application follows the Client-Server Architecture. The Server communicates with the database and exposes the web APIs for which the Client then talks to. The client of the application is implemented with the React framework. The information in the database is summoned by the Server and sent to the Client, and vice-versa. The Client displays the necessary information to the end-user.

The Server is implemented in Java SpringBoot. Its implementation involves packages named “entities”, “repositories”, “services”, “DTO”, “Config” and “controllers”. Entities model the tables, and its records and fields. The repositories, using the Jpa repository, implicitly constructs the SQL statements necessary to perform CRUD operations. Finally, controllers are responsible for responding to and making requests. In our case, the client will interact with the exposed APIs by the controllers from the server.

The DTO package contains the necessary DTO’s that will convert into the appropriate object in the server for either database storage or sent as a Response Entity from an endpoint.

The Config package contains the security configuration for login by customers. It will be required to have an account to make purchases with the Joy’s Toys App.

In the browser, the user is greeted by the homepage. The homepage shows the logo, along with the button to login to the App and the shopping cart which holds all of the products that the Customer selects. The shopping cart is clickable and will display the items held and the button to checkout when clicked. If checkout is clicked, then the user is taken to the checkout page for further order processing. The homepage also contains a section titled “Best Sellers”, which are the best-selling items according to the server logic, and the categories which contains the respective products. Once a category is clicked, a page for the clicked category is generated for the user. When the user clicks login, the application is supposed to take the user to a login portal.

The images that are supplied to the pages are done using Apache with XAMPP as a fileserver, as noted in the database section.

The overall schema of the application is modeled by Figure 4.8. This is the UML diagram of the Joys Toys Application, where user interaction and intended functionality are modeled.

The shop will be visible to the public, where potential customers may browse the categories and the products contained within. Customers who wish to make purchases will need to create an account, which will contain purchase history and payment information. Customers may edit their accounts in order to provide the experience they want.

Employees will also have separate access to the app, with necessary privileges to access information and to perform specified tasks.

Currently, the application is incomplete. See section 8 regarding [future considerations](#_Future_Considerations) for work still needing to be done on the application.

Figure 4.1 shows the server implementation In SpringBoot.

Figure 4.2 shows the data in JSON format that is summoned from SpringBoot.

Figure 4.3 shows the client implementation in React.

Figure 4.4 and Figure 4.5 show the webpage that is generated with React connected to SpringBoot.

Figure 4.6 shows the page generated when a Category is selected by the user.

Figure 4.7 shows the cart when the checkout button is clicked

Figure 4.8 shows the overall schema of the application

Text

Description automatically generated

Figure 4.1

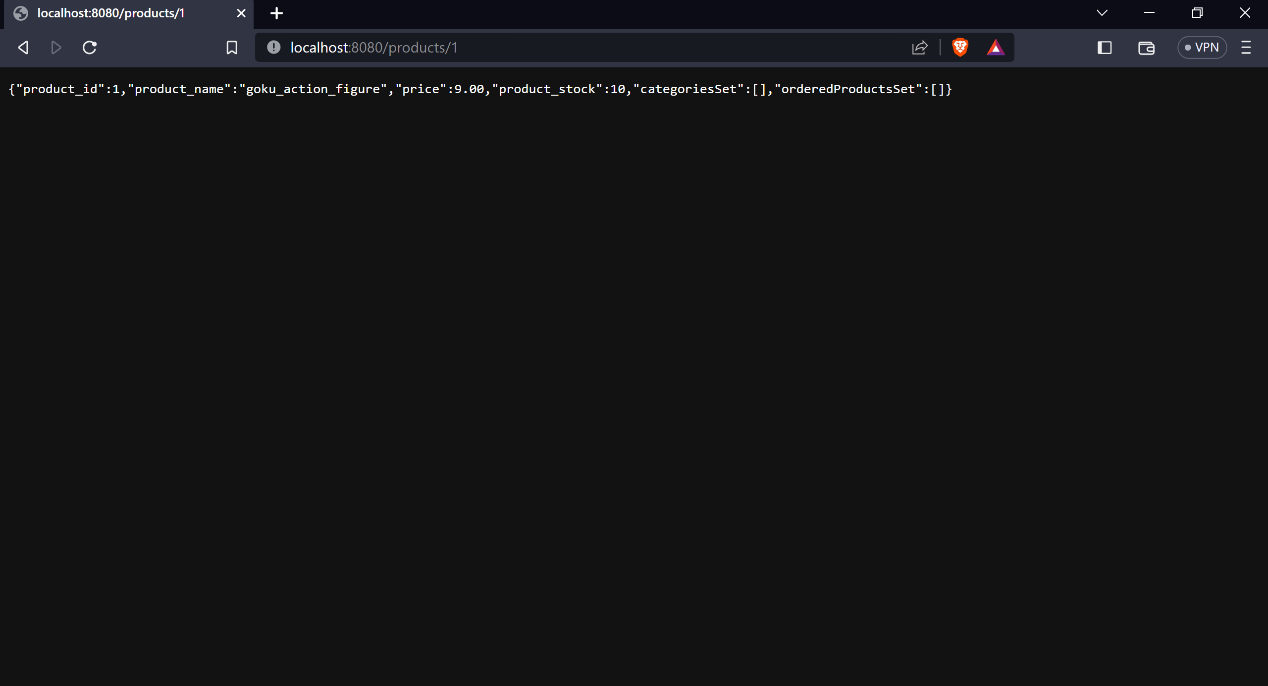
Figure 4.2



Figure 4.3

Graphical user interface, application

Description automatically generated

Figure 4.4

A picture containing graphical user interface

Description automatically generated

Figure 4.5

Graphical user interface

Description automatically generated

Figure 4.6

Graphical user interface

Description automatically generated

Figure 4.7

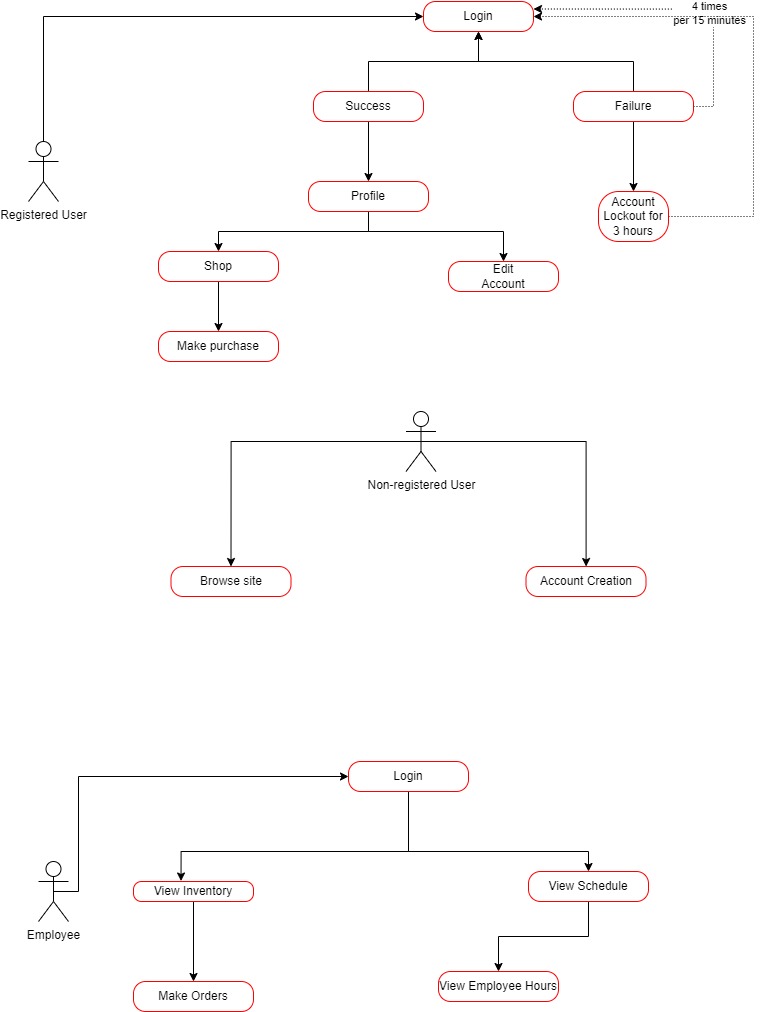


Figure 4.8

# Network and Physical Architecture Design Overview

## Background Information

Currently, Joy’s Toys do not have any computers in the building, as well as no app that customers/employees can access and make purchases, changes, or track history with. We are attempting to build an app that will do the aforementioned for Joy’s Toys. The app will require certain hardware to be introduced to the business.

In order to create the app for Joy’s Toys, the hardware/software will include a Server, a network firewall, a modem and router, 4-5 tablets, and a thin-client desktop for data entry. The mobile app will be accessed through a mobile phone.

## System Evolution Description

\*Network Diagram\*

A picture containing text, sky

Description automatically generated

Currently, Joy’s Toys Network Architecture is nonexistent. A Server will be added to provide services to the app, while the modem and router shall provide access to the internet. The network firewall will be placed in between the internet and the Server for extra protection. Clients will then be able to access the Server via wireless and wired connection.

A backup server will be provided, in the case of disasters occurring.

## Disaster Recovery Plan

A list of hardware that will be required for Joys Toys web infrastructure will include:

* A stable internet connection
* A server for the web application
* A backup server for recovery
* 1 router
* A firewall
* Thin clients, including two desktops, and pad mobile devices

The likelihood of a natural disaster occurring is relatively low. Even still, we will provide a level of redundancy for the case that a natural disaster does occur.

The web server will be rented from a server site, which means that it will be located at an offsite facility. So in the case of a natural disaster occurring, the data for Joys Toys Application will be secure.

The backup server will also be kept at an offsite location. This will provide a level of redundancy for data recovery, in the case that something disastrous happens at the server site location.

Due to construction of the React framework, many dependencies are included in the construction of the Joys Toys application. Fortunately, React is a well-maintained framework making it very easy to manage the security to these dependencies. Most updates cover many of the security vulnerabilities. An upgrade schedule will be created and adhered to. This will fall under the category of “regularly scheduled maintenance” which shall occur according to the upgrade schedule.

Spring Boot utilizes hibernate, which uses prepared SQL statements. As well as providing a level of abstraction, this also provides a level of security against SQL injection.

Payment information will be stored on the server and backup server which will then be protected by the Oauth2 authentication process.

Occasionally, downtime may occur. Whether the cause is regularly scheduled maintenance, network issues, or an issue with the server itself. As far as server issues are concerned, the backup server provides a level of redundancy. Thus, if an issue with the server does occur, then the backup server will take over the main role of responsibility in providing functionality of the web application.

As for other issues, an investigation shall be conducted once an issue is detected. Any issue with the network, or otherwise will be fixed as promptly as possible.

# Training Plan

Joy Johnson: Joy will have full access to the application. Therefore, she will require the most training out of all the employees, save for Bea.

The training will include

* Computer Training
* Mobile phone training
* Application training
  + How to use both the web application and the mobile application
  + Admin role creation and assignment
  + Security Training

Bea Bernard: Bea is the accountant, so she will have full access to the information the database holds. This will include special authorization within the app itself, where she will be able to access financial tables and records, stock and inventory, as well as employee data. Bea’s training will include:

* Application Training
  + How to use both the web application and the mobile application
  + Admin role creation and assignment
  + Security Training
* Security Training

Peter Parker: Peter will be responsible for stock/inventory. He will be able to access and order inventory through the app. Peter’s training will include:

* Application training
  + How to use the web and mobile application
  + How to access inventory information and make orders
* Customer Service
  + Customer training and support

Gwen Stacy: Gwen will be the Customer Sales Representative, so she will be the first point of contact between customers and Joy’s Toys. Therefore, she will need:

* Application Training
  + Sales
  + Stock
  + Receipt
* Customer Service support
  + Helping customers use the app

Plan of Adoption:

Once the application is in functioning capacity, emails from current customers of Joy’s Toys will be gathered. Once that step is complete, an email shall be mass distributed to the emails of the customers of Joy’s Toys that will contain information regarding availability, download instructions, and how-to-use instructions for the App.

# Future Considerations

In the current state of the App, the items are being summoned from the database and presented on the webpage. However, the shop functionality, user creation/login process, best-seller functionality, employee portal/homepage, and customer portal/homepage have yet be implemented.

The App is also currently not dealing with session management. This needs to be added so that a persistent state of the customer can be maintained throughout use of the application, including making purchases. This will be solved primarily on the client. Although, more endpoints must be created in the server in order to appropriately handle specific requests that will be made by the client.

Payment processing needs to be implemented as well, in order to handle customer orders.

The best seller functionality requires another endpoint be created in the server to implement the necessary logic for the intended result.

Overall design should also be improved upon, as currently the App is just a bare site with minimal decoration.

On the opposite side of the application, the appropriate endpoints and logic need to be implemented in the server for Inventory Orders and employee functionality. Inventory orders should only be made and dealt with by employees with the required level of authorization. Inventory Orders will also go through the same payment processor as customers, provided that each supplier will be able to accept payment via these methods.

Finally, better overall security will need to be implemented. Once customer service and shop functionality, along session management are implemented a higher degree of security will be required in order to protect our customers and employees from outside threats.