Illinois Institute of Technology

CS 425

Database Organization

CS425 Project: Online Distribution Center

Authors
Hector Hernandez
Ramir Aguilos
Rafael Zavala
Hauyu Wang

Professor
Dr. Omar Aldawud

February 28, 2018

Contents

1 Introduction

This project consist of a web solution for an e-commerce web application regarding product distribution. From the web application, customers will have access to querying products from different suppliers. Employees of the web application will have specific views tailored to their tasks. The project itself will be conducted using a waterfall model [1] for development. The waterfall model consists of these ordered steps: Requirements, Design, Implementation, Verification, and Maintenance. The requirements that are set out for the development of a software project are constraints, usages, views, functions, and any other interaction between users and the software's interface. For this project, the requirements are listed as a list of functionalities that different users have with the product distribution system. The design of a software project is the development of blueprints or plans for how the development will be implemented. Design also consists of determining what medium and environment the project will be developed in. For the purposes of this project, the design consists of using Github repository as a functioning interaction between the project developers to create files necessary for the database driven software. The implementation of a software project will be the actual creation of the design plans. The implementation of the project will be a web-based application using a DBMS (database management software) as the back-end of the system and HTML, CSS, and PHP, with NodeJS as the front-end of the application. The verification of a software project is the running of tests to determine its readiness to be deployed as a product or service. For this project, testing will be driven by the methods of unit testing each aspect of the web application and having test users use the application. The maintenance of a web application occurs after a software project has been launched or deployed. Ideally, the maintenance of a web application occurs autonomously as the database and server structure will be able to maintain a state of software homeostasis where users can manage their access to the service without losing the integrity of data stored in the database. When there are cases such as data integrity loss or concurrency issues, the physical maintenance and restructuring software are necessary, which would require project developers to address. For this project, maintenance of the web application can be accomplished through many methods such as having more than one physical drive to store user data or creating 'watchdog' programs to monitor the state and integrity of the databases that do not rely on the main server.

2 Requirements

#	**Requirement Description**	**Testing Criteria**
R1	The application shall allow customers to register with the application. The application shall present the user with a registration form with the following fields: Customer Name, Contact Name, Email, phone, region, address, major product line category (research a list that fits most customers).	User enters registration information and the data is stored in the Customer database.
R2	Once a customer registers with the application the application shall generate a unique customer id.	A unique customer is stored in the database.
R3	The system shall allow the customer to enter a password for her account. The password shall at least be 8 characters long and shall have at least one distinctive character and one uppercase character and a mixed alphanumeric value.	Customers password shall be encrypted and stored in the database.
R4	The customer shall be able be able to visit his/her profile within the application and be able to update his/her profile information only.	Customers information in the database is updated according to user input.

#	**Requirement Description**	**Testing Criteria**
R5	The customer shall be able to query products online. with the following attributes: Name, Description, Categories, Units, Price.	User enters product information and the application shows a list of relevant products.
R6	The application shall allow the Supplier to register with the application. The application shall present the user with a registration form with the following fields: name, email, phone number, and address.	Supplier enters registration information and the data is stored in the Supplier database.
R7	Once the supplier registers their product with the application, the application shall generate a unique product id and will also update the Categories database to include any new categories created or increase the relevance of any categories that exist already.	A unique product is stored in the Products database and the product is visible to any any customer within the application. Categories Database is updated accordingly based on how products are loaded onto the application.
R8	The customer should be able to add/remove products to their cart. Even when the user logs out, the cart is saved with all products from the previous session(s).	Cart database, corresponding to the customer, is updated accordingly. Cart keeps products from the previous session until checkout or explicitly removed by the user. Cart displays price, quantity, title, and image of each desired product. Cart displays total expenses.

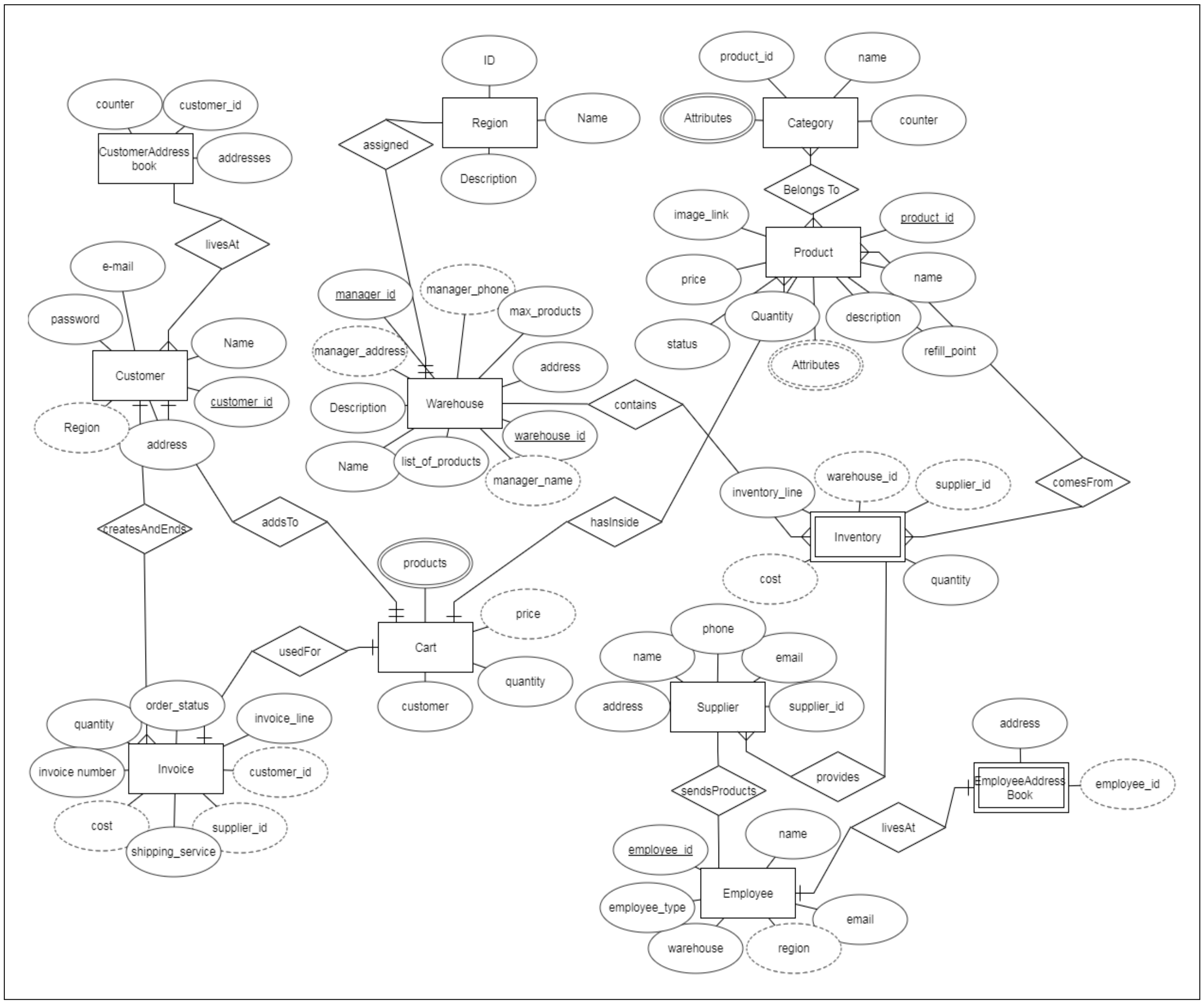
#	**Requirement Description**	**Testing Criteria**
R9	The customer shall have the option (button) to remove all products from the cart in one instance.	A pop-up window confirms the deletion of all products in cart, then cart is emptied entirely.
R10	The customer shall have the option to checkout with PayPal or credit/debit card.	The application verifies the transaction. Once the transaction is verified, an e-mail confirmation of their placed ordered is sent to the customer. An e-mail confirmation is sent to the seller.
R11	Customer gets asked if he/she wants to save their payment information for future orders.	Pop-up window confirms if the user wants to save payment information; if yes, payment information is stored.
R12	The customer shall be able to monitor the status of their order.	The application allows the customer to view orders placed; the application
R13	The customer shall be able to search for products explicitly using a search engine within the application.	The application searches the Categories or Products databases to display items.
R14	The customer shall be able to view their order history.	The application provides a list of all orders made by the customer and the status and time stamp of each order.
R15	The application shall display the status of each product.	The application searches through the Product Database and assigns the status (In stock, Out of Stock, etc...) to the product based on quantity.

#	**Requirement Description**	**Testing Criteria**
R16	The application shall allow registered customers to sign-in via a sign-in page.	The sign-in page searches through the customer database to authenticate the user.
R17	The application shall automatically display the most popular categories on the home page.	The application should keep track of what categories and products are frequently searched for.
R18	The application shall allow warehouse managers to register with the application. The application shall present the warehouse manager with a registration form with the following attributes: Name, address, warehouse_id, special password (provided by the company).	The data is stored in the warehouse database and a unique manager id created.
R19	Warehouse managers can view customers in their region.	The application shall allow warehouse managers to view a list of customers (Name, address, region, and e-mail) in his/her region only.
R20	The application shall allow warehouse managers to view the inventory in their warehouse.	The application should provide a list of all products that are in their respective warehouse.
R21	The application shall emphasize to the warehouse manager which products need to be prioritized for shipping.	By checking the time stamps of the orders or shipping service used, the application will output a filtered list of products.

R22	Administrators have the privileges of modifying warehouses, product, employee, invoices, and customer data.	Administrator account(s) with these privileges will be created.

3 Design

3.1 Entity Relationship Diagram



3.2 Instantiation

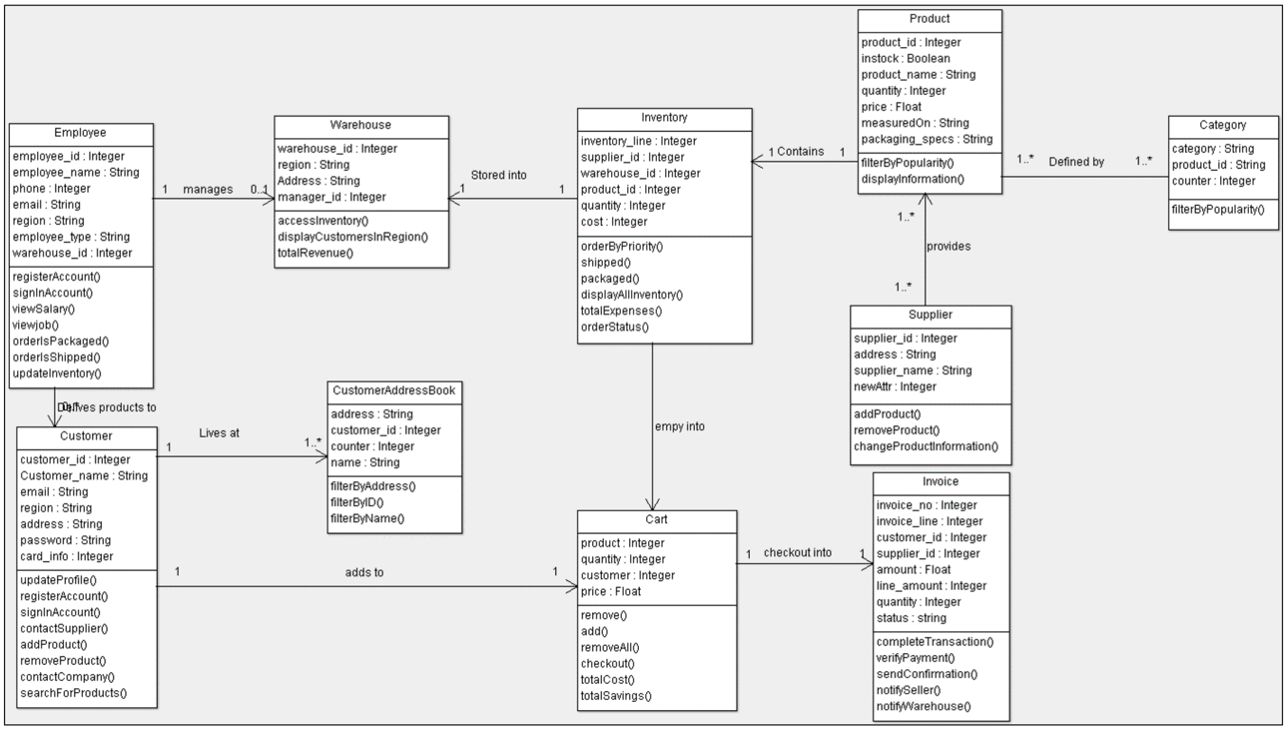
The following code represents the instantiation of the database shown in the previous ERD diagram.

```
DROP DATABASE IF EXISTS ecommerce;

CREATE DATABASE IF NOT EXISTS  ecommerce;

USE ecommerce;


/*

Creating Customer table

*/

DROP TABLE IF EXISTS Customer;

CREATE TABLE IF NOT EXISTS Customer

   (

address varchar(50) not null,

customer_name  varchar(50) not null,

customer_id  int AUTO_INCREMENT not null,

email varchar(50) not null,

password varchar(50) not null,-- We will eventually hash this

PRIMARY KEY (customer_id)

   );


/*

Creating CustomerAddressBook table

*/

DROP TABLE IF EXISTS CustomerAddressBook;

CREATE TABLE IF NOT EXISTS CustomerAddressBook

   (

    address varchar(50),

counter int UNSIGNED NOT NULL default 0,
customer_id  int ,
FOREIGN KEY(customer_id) REFERENCES Customer(customer_id)
   );

DROP TABLE IF EXISTS Product;

/*
Creating Product table
*/
CREATE TABLE IF NOT EXISTS Product
   (
description varchar(250),
image_link varchar(250),
price DECIMAL(6,2) NOT null,
product_id  int AUTO_INCREMENT ,
product_name varchar(250) ,
quantity int not null,
units varchar(50),
refill_point varchar(50),
status varchar(10),
primary key (product_id)
   );

/*
Creating Category table
*/
DROP TABLE IF EXISTS Category;
CREATE TABLE IF NOT EXISTS Category
```

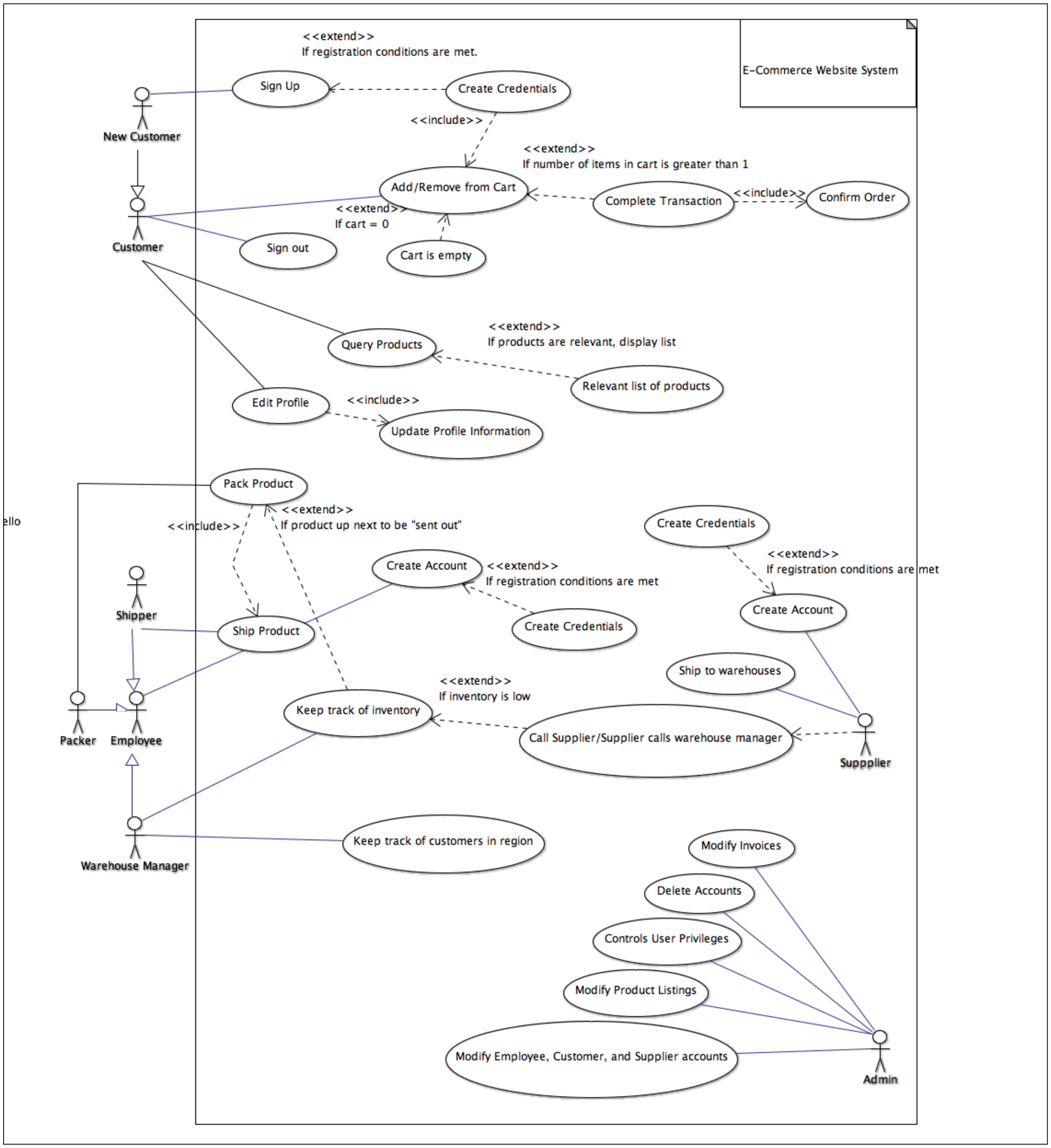
```
   (
attributes varchar(250),
category varchar(20),
counter int UNSIGNED NOT NULL default 0,
product_id  int not null,
FOREIGN KEY(product_id) REFERENCES Product(product_id)
   );
/*
Creating Region
*/
DROP TABLE IF EXISTS Region;
CREATE TABLE IF NOT EXISTS Region
   (
region_id int NOT NULL,
region_name varchar(250),
description varchar(250),
PRIMARY KEY(region_id)
   );

/*
Creating Cart
*/
DROP TABLE IF EXISTS Cart;
CREATE TABLE IF NOT EXISTS Cart
       (
customer_id int  ,
product_id int not null,
quantity int not null,
FOREIGN KEY(customer_id) REFERENCES Customer(customer_id),
FOREIGN KEY(product_id) REFERENCES Product(product_id)
   );
/*
Creating Supplier
*/
DROP TABLE IF EXISTS Supplier;
CREATE TABLE IF NOT EXISTS Supplier
   (
address varchar(50) not null,
email varchar(50),
phone varchar(20),
supplier_id  int AUTO_INCREMENT,
supplier_name varchar(50) NOT NULL,
primary key (supplier_id)

   );


/*
Creating Employee
*/
DROP TABLE IF EXISTS Employee;
CREATE TABLE IF NOT EXISTS Employee
   (
email varchar(50),
employee_type  varchar(20),
employee_id int NOT NULL,
region varchar(50) NOT NULL,


employee_name int,

primary key (employee_id)

   );


/*

Creating EmployeeAddressBook table

*/

DROP TABLE IF EXISTS EmployeeAddressBook;

CREATE TABLE IF NOT EXISTS EmployeeAddressBook

   (

employee_id  int not null,

address varchar(50),

FOREIGN KEY(employee_id) REFERENCES Employee(employee_id)

   );


/*

Creating warehouse

*/

DROP TABLE IF EXISTS Warehouse;

CREATE TABLE IF NOT EXISTS Warehouse

   (

region   varchar(50),

warehouse_id int NOT NULL,

address varchar(50) NOT NULL,

manager_id int,

primary key (warehouse_id),
FOREIGN KEY(manager_id) REFERENCES Employee(employee_id)
   );



/*
Creating Inventory
*/
DROP TABLE IF EXISTS Inventory;
CREATE TABLE IF NOT EXISTS Inventory
   (
inventory_line  int AUTO_INCREMENT,
cost DECIMAL(6,2) NOT NULL,
quantity int NOT NULL,
product_id int,
warehouse_id int ,
supplier_id int,
FOREIGN KEY (supplier_id) REFERENCES Supplier(supplier_id),
FOREIGN KEY (product_id) REFERENCES Product(product_id),
FOREIGN KEY (warehouse_id) REFERENCES Warehouse(warehouse_id),
primary key (inventory_line)

   );


/*
Creating Invoice
*/
DROP TABLE IF EXISTS Invoice;
CREATE TABLE IF NOT EXISTS Invoice
```

```
   (
invoice_no  int AUTO_INCREMENT,
invoice_line int NOT NULL,
amount DECIMAL(6,2) NOT NULL,
product_id int,
supplier_id int,
quantity int NOT NULL,
FOREIGN KEY (supplier_id) REFERENCES Supplier(supplier_id),
FOREIGN KEY (product_id) REFERENCES Product(product_id),
primary key (invoice_no)

   );
```

3.3 UML Class Diagram

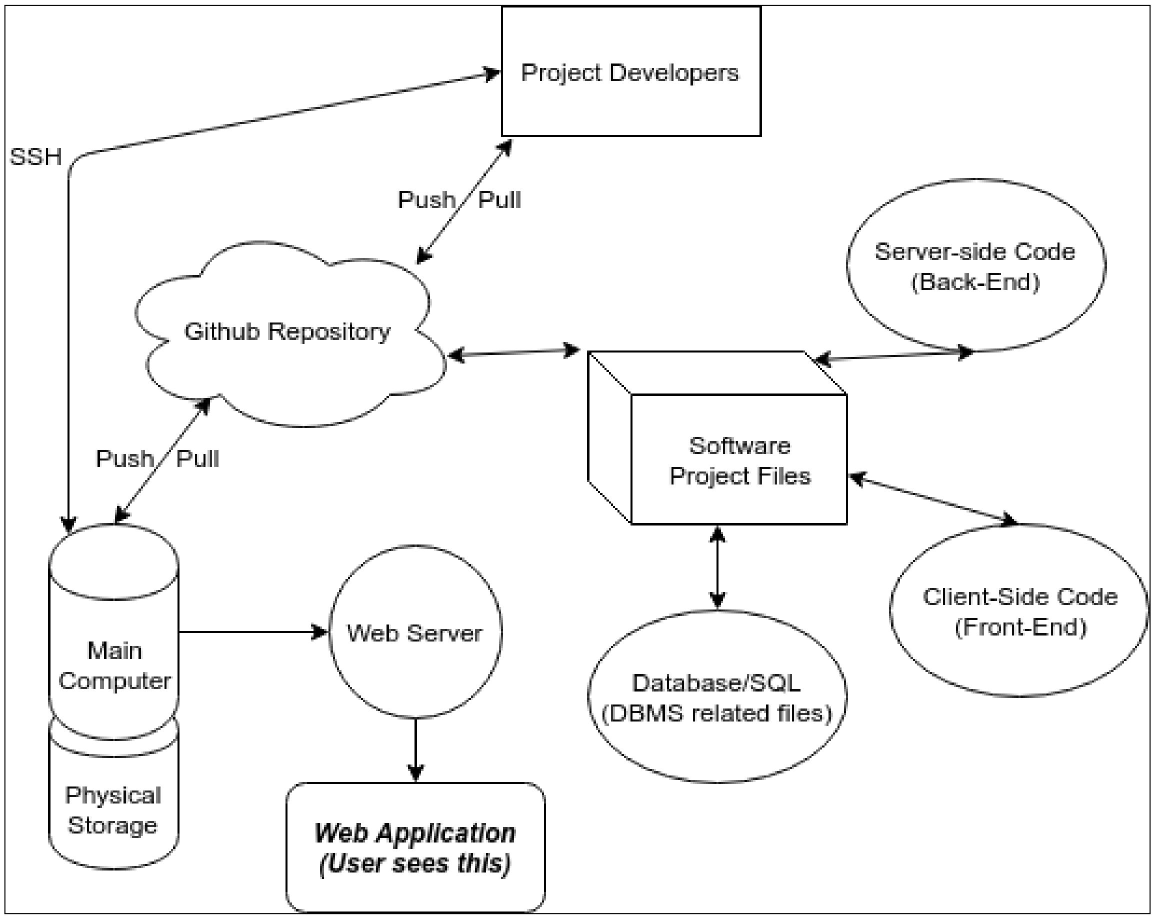


3.4 UML Case Diagram



4 Implementation

The implementation of the project will be a web-based application using JDBC program files as a back-end and Node.js, HTML, CSS, and PHP program files as a front-end. The development environment that the project developers will use is the UNIX operating system environment and tools connected with Github serving as a functioning repository for back-end and front-end code to be updated as the project progresses. The web application will be hosted on an Apache web server linked to a main computer with Ubuntu Linux as its operating system. This main computer can also be accessed via SSH for the project developers. Below is a diagram for the method of implementation:



5 Verification

The verification of the project will be performed by implementing different testing methods [2] with the web application. Firstly, unit testing will be implemented by testing individual aspects of the web application. For example, a customer is able to complete transaction to his or her request. So the implementation of unit testing here would exist as making sure each individual part of the transaction occurs correctly. In accordance to the ERD, the 'Invoice' is an entity with data that is updated atomically; specifically, this is the record of a transaction a customer may request. To unit test the entire transaction, code will be debugged with each iteration of the program until the transaction results in a feasible output (i.e. the invoice). By implementing unit testing during the development of the web application, each individual function of the web application will have higher success of functioning correctly because each function would have been tested to work individually. System testing is also a necessity for the project because system testing captures the users' perspectives and views, which is important in keeping integrated yet private data. System testing is best described as having a box with an input and output where specific users have specific views. In this project for example, if a customer orders a product, a customer would like to know what warehouse it is coming from but may not need to know who the supplier of the product is. Integration testing is the methodology of testing where developers test to see if their software works with different hardware and interfaces. To actually have users of the application, users must be able to use their own devices and hardware to access the application, which makes integral testing necessary. Performance testing is a method of testing to see the efficiency of the application. For this project, performance testing will be implemented by having the product distribution service have relevant information readily available for different users.

6 Maintenance

The maintenance of the project will be performed with various concepts to keep the deployed application stable. On a micro level, the development of the back-end (server side content) using JDBC will serve to maintain the integrity of the data as it is updated per user request. On the macro level, the maintenance of the project may consist of the project fixing bugs that users may find from interactions with the software, which would be referred to as patches. Additionally, maintenance includes the updating of present software's features with new features that make the user's experience easier or more efficient.

References

[1] *Waterfall Model*

`http://testingfreak.com/waterfall-model-software-testing`

`-advantages-disadvantages-waterfall-model/`

[2] *Testing Methodology*

`https://www.guru99.com/testing-methodology.html`