Blue Groceries

Online Grocery Store

This is a light copy of the documentation I originally prepared for my Project Report it may help you with better understanding of the project.

Table of Contents

Introduction

The application developed is an Ecommerce Web Application. It is a fictional project, as formulated by me in guidance with my Project Guide Mr. Sanjeev Bora.

Blue Groceries is a leading grocery Store in Starling City and feels the need to extend its operations online. The company is already established and using technical aid to perform its operations offline in, covering over an area of 500 Kms.

The Company already owns 20 well established offline stores at the major parts of the city, being one of the biggest suppliers of grocery items. The growing use of technology and Internet has made them to bring their store online, acting as an endpoint to sell their products directly from the store.

The general idea of this project is to develop a web application that may allow its users to directly purchase products from the store, and pay via electronic means.

2. Objectives

1. A web application that may allow the customers to make purchases online.
2. A secure endpoint for payments.
3. An online selling point for Blue Groceries Stores.
4. The application able to run on both mobiles and Wide Screen Devices
5. The application should facilitate users with online paymenst enabling them to pay, while making purchases.
6. The payment information should be treated with utmost security.
7. The client and server security features.
8. An optimized and quickly responsive application.

9. Tools and Environment

1. A Web Browser (Mozilla Firefox/Google Chrome/ Opera/, Microsoft Edge)
2. Visual Studio Code (A Code Editor) with a bunch of Extension packs.

   Major Extensions

   2.1 MySQL – Connects to MySQL server from within the Code Editor, and enables to Make Queries
3. NodeJS
4. Yarn (A package manager for NodeJs, alternative to default NPM)
5. NodeJS Packages

   5.1 *Express Web Framework*

   5.2 *MySQL Package* – to connect to MySQL Server

   5.3 *Base64* – Encodes & Decodes Base64 strings

   5.4 *EJS* – A templating framework for NodeJS

   5.5 *Express Session* – It helps in maintaining sessions in the express app.

   5.6 *MailgunJS* – It helps sending emails through your Mailgun account.

   5.7 *Express MySQL Session* – It enable storage of sessions created by Express Session into the MySQL Database.

   5.8 *StripeJs* – It acts as a gateway to connect to Stripe and charge the customer.
6. MySQL Database & Server
7. MySQL Front – A feature rich GUI Application that help in manipulating database.
8. Lucid chart – An online browser based diagramming tool.

Development Environment

Windows 10 (x64 bit) Redstone 3 (Version 1709) Build 16299.309

5GB Disk Space

10. Project Analysis

10.1 SRS Document

1. Introduction

    1.1 Purpose

    1.2 Scope

    1.3 Definitions and Acronyms

    1.4 References

    1.5 Document Overview

2. General Description

    2.1 Product Perspective

    2.2 Product Functions

    2.3 User Characteristics

    2.4 Constraints

    2.5 Assumptions and Dependencies

3. Specific Requirements

    3.1 External Interface Requirements

3.1.1 User Interface

3.1.2 Hardware Interfaces

3.1.3 Software Interfaces

3.1.4 Communication Interfaces

3.2 Functional Requirements

3.3 Performance Requirements

3.4 Design Requirements

3.5 Logical Database Requirements

3.6 Software Attribute Requirements

1. Introduction

Blue Groceries is a well-established, leading seller of Grocery items in Starling City. They own over 20 stores in the city. The Company has done a recent research and found the need of having an online system in order to boost their business. A recent survey conducted among the in-store customers, resulted in a positive response to the online store.

The online platform is seen as a major area of development, and will supplement in-store sales of Blue Groceries.

1.1 Purpose

The purpose of this document is to define the system being developed, formerly named Blue Groceries (Blue Groceries – Online Grocery Store). The intended audience of this document are developers, administrators as well as end users, and anyone who wish to use the software, or have an understanding of the system.

1.2 Scope

The plan of launching Blue Groceries as an online platform was envisioned taking into account the growing use of online platforms, in our day to day lives. It was formulated and by the strategic team of Blue Groceries, after conducting a survey among the visitors of the store.

The key goals of Blue Groceries are

- Providing convenient buying options, on both online and offline buying points.

- Providing at the doorstep service to their customers.

- Increase customer loyalty from 15% to 20% within a year.

- Increase sales from 40 to 60%, in a year.

1.3 Definitions and Acronyms

Term	Definition
Application	The system under development, or being developed.
User	These are customers of Blue Groceries that will be using the final application.
Administrator	The person having full control over the system, and is responsible for maintenance, fixes and deployment of the product.
Developer/s	A person (or group) involved in development of the application.

Acronyms

IE: Internet Explorer

1.4 References

IGNOU BCS 051: Introduction to Software Engineering.

Express JS Documentation: http://expressjs.com/

EJS Documentation: http://ejs.co/

Express-sessions Documentation: https://github.com/expressjs/session

MySQL JS Documentation: https://github.com/mysqljs/mysql

Express MySQL Session Documentation:

https://www.npmjs.com/package/express-mysql-session

Stripe Documentation: https://stripe.com/docs

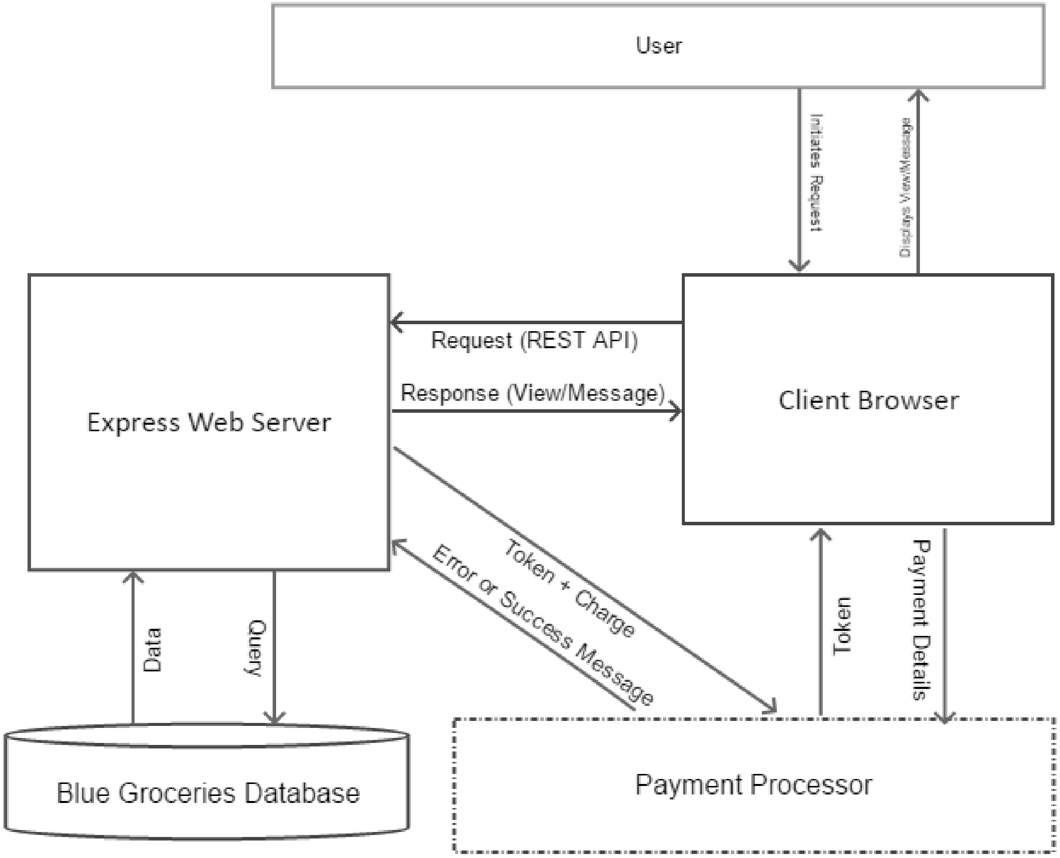
1.5 Document Overview

There are three sections at total in the document. The next section deals with the details regarding application, its functions, and if any Constraints, Assumptions or Dependencies. The final section provides details about functional or non-functional requirements of the application/use-cases

2. General Description

2.1 Product Perspective

The application uses Client-Server MVC Model.



It consists of a browser client acting as an interface between the end users and the Express Web Server. The client interacts with server using the REST API, and the server responds with an appropriate view (or sometimes message) for the user.

A high level block diagram of Blue Groceries is provided above.

2.2 Product Function

Search: This enables users to make searches for available products. It displays all the products that (or a part of which) matches with product name or description.

A message will be displayed if there is no product that matches given search term.

Authentication (Login/Signup): It allows users to authenticate that they are human, and sole owners of their data and account.

Recovery: It lets users to reset their password using their registered email, in case they have lost the old one.

Shopping Cart: The application allows users to add products to a cart, once added to the cart the products can be later purchased at once.

Order Confirmation: The application let users to make any changes to quantity or shipping address before proceeding for payment. The default shipping address is as provided by the user

Checkout: It fetches payment information from the user and generates a token to be sent to the server, in order to charge the user, for the specific amount.

2.3 User Characteristics

Site Users: These are customers (end users) of Blue Groceries, the entire application is focused to provide services to them.

Site Administrator: This is a person or group of person responsible for deployment, any fixes, and regular maintenance to the application. The administrator has total control of the system, with full privileged access.

2.4 Constraints

- The application is only being produced in English (US), so the user must have basic understanding of English in order to use the system.
- The products as well as search results are divided into multiple pages. Each page displays from 1 to 12 results
- IE is partially supported or not supported at all. It is not recommended to use the application on IE. All the modern browsers are supported, on any type of Devices.
- The application needs Javascript support
- The application uses cookies to maintain sessions, so the setCookie permission is required in the client browser.

2.5 Assumptions and Dependencies

Assumptions

These are the premade assumption

- The user has a working Internet Connection, and is connected to the Internet.

- The end user has the knowledge of performing basic tasks on the browser and can navigate through web pages easily.
- Blue Groceries has all the needed Hardware and expertise to support the user load.
- Blue Groceries already has a product database built up for the operations among its stores that will be leveraged.
- All the connected systems i.e. Database Server, Payment processor provide maximum guaranteed uptime, and are running along with the application.
- All the upstream systems, provide the response within agreed time, to ensure optimal performance.

Dependencies

These dependencies are just for the server, there will be no (or minimal) dependencies on the client end.

- Express Package - It is a lightweight NodeJs Framework that provides a robust set of features, for making Web and Mobile Applications.
- EJS (Embedded Javascript) – A templating engine based on Javascript that enables creation of HTML Markup pages, using simple Javascript.
- Express-Sessions - A session middleware for express that makes it easier to implement Sessions in the application.
- Express MySQL Sessions – A Session Storage Javascript package, that helps in storage of Sessions in the MySQL database.

- Stripe – A payment processor that enables payment with the sensitive data never touching the web server.

3. Special Requirements

The following section provides detailed functional and non-functional requirements.

3.1 External Interface Requirement

This section provides the details of all Inputs and Outputs including the hardware, software, communication and mockup prototype

3.1.1 User Interfaces

Blue Groceries contain following major User Interfaces, and a few other supporting pages.

- *Login Page:* It is for authenticating the users already registered i.e. they have a Blue Groceries user account. The page accepts two parameters. The first parameter must be a username or an email, the second parameter is a password.
  The page also displays links to Sign up and for password recovery.
- *Signup Page:* It enables new user to get an account at Blue Groceries, by providing the sufficient amount of information.

The page also displays link for Login, in case the user accidentally landed on the page.

- Search Page: It displays the search results, the results are divided into pages. Each page can have 12 products at most.
- Shopping Cart Page – It displays the products that have been added to the cart. The user can proceed to order all the products at once, or remove any product that he/she don't wants to purchase.
- *Home Page:* It acts as the main page, shows a few products, and other navigable components.
- *Error Page:* It displays that the user has landed on the page that does not exist, or may have been removed. In other words, it comes out, when user tries to access an unavailable resource.
- *Order Confirmation Page:* It allows users to make changes to the quantity, and shipping address before proceeding to payment.
- *Orders Page:* It displays pending, dispatched and delivered orders of the user.

3.1.2 Hardware Interfaces

There are no specific hardware interface requirements to use Blue Groceries, all most all Internet enable devices can use the app with the help of a browser.

The application runs on an application server that is always connected to the Internet.

3.1.3 Software Interfaces

The application should integrate with the following systems in order to function.

- MySQL Database Server: It helps in storage and retrieval of necessary data whenever necessary for the database.
- Payment Processor: It processes payments for the orders and charges customer with the required amount. The payment processor to be used is Stripe. It generates a token from the payment information provided by the user, the token is then used to charge the user.

3.1.4 Communication Interfaces

The existing OS and the network will be used communication between different components of the application. However, the components will be using REST API for making requests.

3.2 Functional Requirements

The following section provides the list of all the major functionalities supported by Blue Groceries.

3.2.1 Functional Requirement 1: Login Operation

Functional Requirement ID	FX1
Title	Login Operation
Requirement Description	• The user enters Email/Username along with a password. • The user hits the login button • The application validates the input provided by the user. • The application passes on the information with through a PUT request for successful Validation. • The server should authenticate the information provided by the user. If the information is correct the user should logged in, the session should be updated the user should be redirected to the home page.
Business Rationale	The registered users make buy orders, edit their information and much more.
Exceptional Scenarios	• If the validation fails, at the client the user must be provided with an appropriate error message. • If the authentication fails on the server the user must be provided with appropriate error message.
Dependencies	None

3.2.2 Functional Requirement 2: Signup Operation

Functional Requirement ID	FX2
Title	Sign Up Operation
Requirement Description	• The user should provide with all the necessary details, require to get an account at Blue Groceries. • The provided information must be validated before submissions. • The information passed to the server must be authenticated for prior existence. • An account should be created for the user.
Business Rationale	• It allows the application to differentiate between users and provide may other services to the user. • It serves as a mean of authentication, for the user. • It terminates the need of providing information with every purchase and automate many tasks.
Exceptional Scenarios	• If the validation fails an error message should be displayed to user about the error. • If the information being provided already exists, the user must be informed about it through an error

	message.
Dependencies	None

3.2.3 *Functional Requirement 3: Search Operation*

Functional Requirement ID	FX3
Title	Search Operation
Requirement Description	• User enters a search term or keyword. • The search keyword is passed to the server, and validated. • The products name and descriptions is matched against the occurrence of the keyword. • The results are displayed in pages, with each page holding 12 products at maximum.
Business Rationale	The users can search for relevant products without having to check all the products they don't intend to buy. It helps the user to discover a product.
Exceptional Scenarios	• If the validation fails, a suitable error message must be provided to the user. • If no product matches the given criteria, an appropriate message must be displayed to the user.
Dependencies	None

3.2.4 Functional Requirement 4: Shopping Cart Operation

Functional Requirement ID	FX4
Title	Shopping Cart Operation
Requirement Description	• It should display all the products added by user in the cart, along with a button to proceed for making an order. • The product details also should be visible, for example, its availability, price and discount (if any). • The user should be able to remove the products in their cart.
Business Rationale	It allows user to select different products and later order them at once, providing a seamless shopping operation.
Exceptional Scenarios	• If the user is not logged in, and tries to add product into the cart, a login prompt must be displayed to the user. • If a guest user tries to access cart page, he must be redirected to login page, or a login prompt should be provided.

Dependencies	FX1: The user must be logged in.

3.2.5 Functional Requirement 5: Order Confirmation Operation

Functional Requirement ID	FX5
Title	Order Confirmation Operation
Requirement Description	• The user can update the quantity for a certain order, before payment. • The user can update shipping information before payment of the order. • The quantity and order are validated before updating the order. • The update orders are displayed to the user.
Business Rationale	The user can make changes as per his preference.
Exceptional Scenarios	• If the quantity is more than items available in the stock, and less than 1, the user must be provided with appropriate error message. • If the address is not valid the user should be provided with the error message.
Dependencies	FX1: The user must be logged in. FX4: The user must have proceed to order the products, from the cart, or individual products.

3.2.6 Functional Requirement Operation 6: Checkout Operation

Functional Requirement ID	FX6
Title	Checkout Operation
Requirement Description	• The final amount, applying discount, shipping charges is displayed to the user. • The user is then asked the payment details. • A token is generated using Stripe Elements. • The generated token is passed to the server. • The token is then used to charge the customer, to the total of products, after applying discounts and delivery charges. • The user is updated about the payment
Business Rationale	The user can pay online, so the company have not to deal with Cash, keep changes and other payment related problems. It provides flexible options for users to make payments.
Exceptional Scenarios	• The user should be provided with card error details if any.

	• The user should be informed if any error occurs during the payment.
Dependencies	FX1: The user is logged in. FX2: The user has confirmed, and updated products quantity and shipping information.

3.2.7 Functional Requirement 7: Edit Profile Operation

Functional Requirement ID	FX7
Title	Edit Profile Operation
Requirement Description	• The user should be able to edit the information they wish to change about their account. • The user inputs the information they wish to change. • The new information is validated for any errors. • The new information is validated and new information is displayed to the user.
Business Rationale	It provides the control of the user over their information, and make changes to it whenever they wish to.
Exceptional Scenarios	• If the user enters the same information, redirect the user to display profile page. • Display an appropriate error message, if any occurs during validation.
Dependencies	FX1: The user must be logged in.

3.2.8 Functional Requirement Operation 8: Account Recovery

Functional Requirement ID	FX8
Title	Recovery Operation
Requirement Description	• The user is sent a form asking either its user name or email. • The user provides the required details. • The input is validated and sent to the server. • It is checked if the user exists, and a recovery link to his registered email is sent automatically. • The user checks his email, and is redirected to a password reset page from there. • The expiration link works only once and is unique. It is destroyed after a use.
Business Rationale	The user can have the ownership back of his data and account, by verifying himself. It also reduces chances of customer calling the company's helpline and requesting the possession back of his account.

Exceptional Scenarios	• The user is to be provided with an error message, if user does not exist. • The user is shown an error page, if he or she uses a fake verification link. • The user is shown an error message after the link expires.
Dependencies	None

3.3 Performance Requirements

The key performance requirements have been listed below.

- The pages must be loaded within a second.
- The search results should not be displayed within a second.
- The inputs must be validated within 0.5 seconds.
- The application should take no more than 3 seconds for creating an order.
- The checkout must be done within 5 seconds after the payment information has been provided.

3.4 Design Constraints

Blue Groceries must follow these design standards.

- The pages must be marked up using HTML5 Standard.
- CSS features must be used to style the elements.
- The pages must be responsive that means they must be viewable on different screens without breaking.
- Material Icons are to be used, which are to be implemented using CSS classes.

3.5 Logical Database Requirements

The database used by Blue Groceries for its offline operations will be leveraged, it will be extended in order to facilitate online operations, any changes if necessary will be made, without breaking any functionality

3.6 Software System Attributes

3.6.1 Reliability

The application should be reliable, it should not crash or break while operations. The data must be properly validated and sent to the server.

3.6.2 Availability: The application must guarantee 99.99% uptime. It must be available 24x7. Any fixes, patches or upgrades, should not always require application to be shut down

3.6.3 Security: The application must have following security features.

- The application must be running on HTTPS (HTTP Secure) connection instead of HTTP.
- The password must be encrypted using asymmetric encryption and only then should be stored in the database.
- The input fields must be validated for SQL Injections, before making query to the database.
- The old password must be provided in order to change the password.
- Each user should maintain a session, even if he is not registered. The session should be updated as soon as user logged in. A session should be created as soon as the user enters the site.
- The user information should not be stored in browser cookies, the cookie on the browser should just act as a token.
- The payment information should not be stored on the server. In fact it should never touch the server, the payment details should be directly passed to the payment processor

3.6.4 Maintainability

The following maintainability features should be supported by Blue Groceries application.

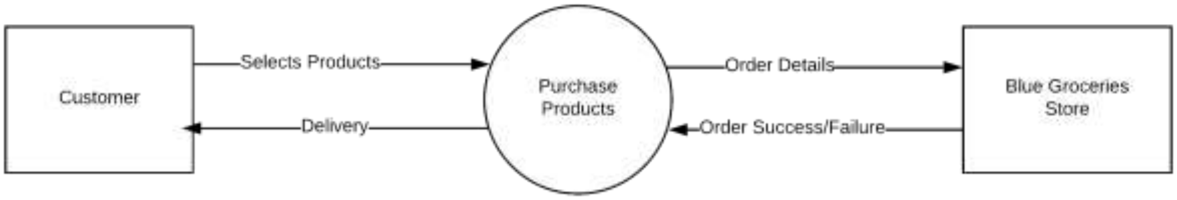
- All code artifacts should have proper documentation.
- All code components must be thoroughly tested with a test coverage of over 80%.
- All code should be commented wherever necessary, to provide easy understanding of the code.
- The application code must be broken into different modules, based on the type of code, its function etc.

10.2 Data Flow Diagrams (DFD)'s

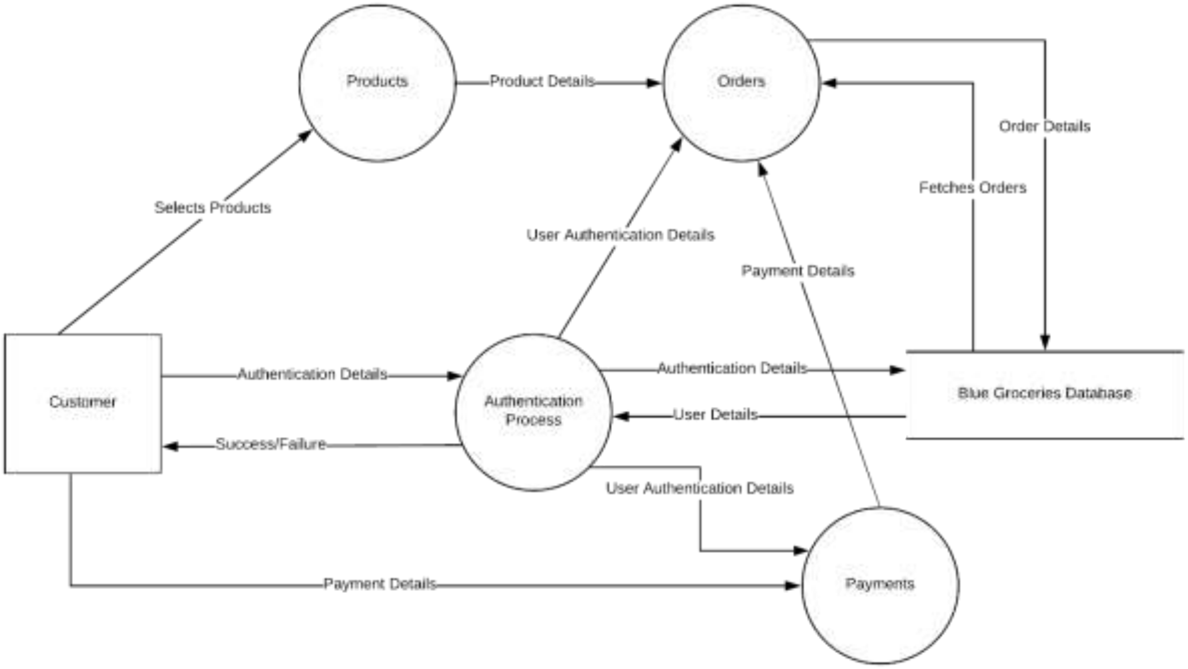
A data flow diagram (DFD) maps out the flow of information for any process or system. It uses defined symbols like rectangles, circles and arrows, plus short text labels, to show data inputs, outputs, storage points and the routes between each destination.

10.2.1 Context Diagram (Level 0)

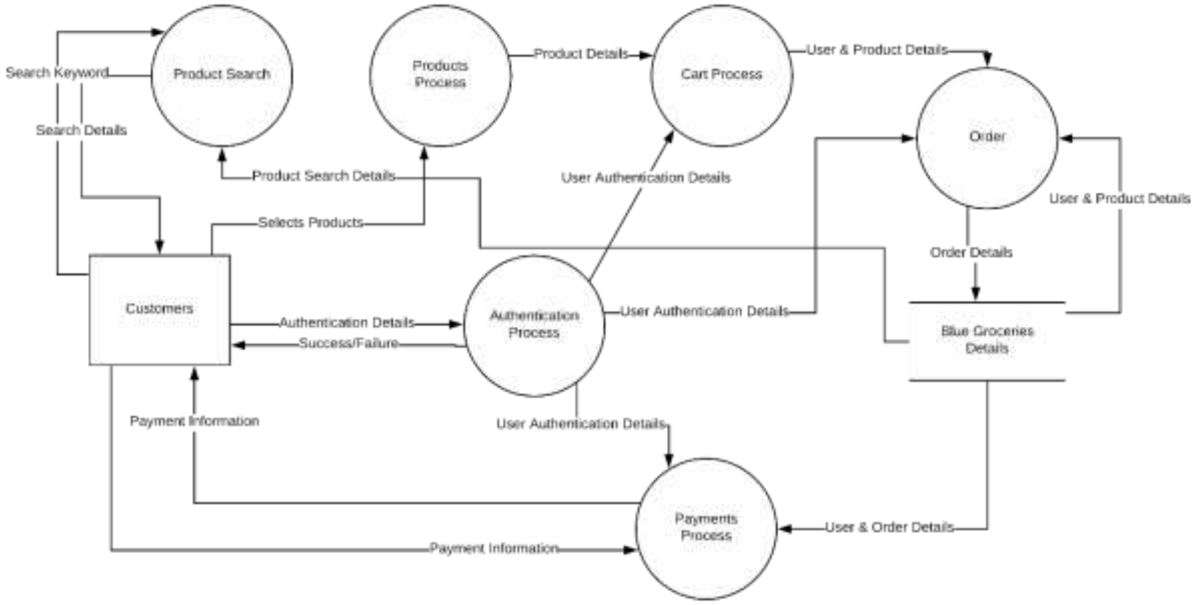
It's a basic overview of the whole system or process being analyzed or modeled. It's designed to be an at-a-glance view, showing the system as a single high-level process



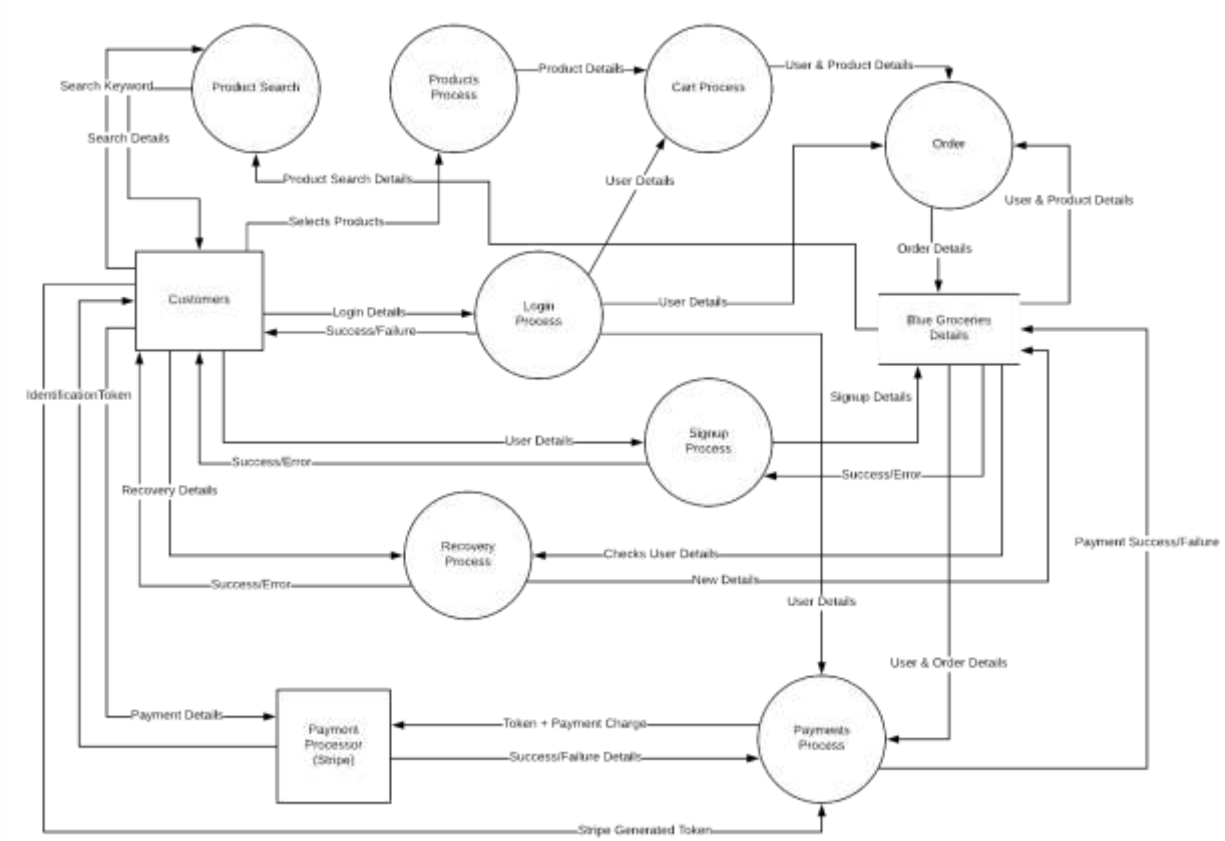
10.2.2 DFD (Level 1)



10.2.3 DFD (Level 2)



10.2.4 DFD (Level 3)



10.3 ERD (Entity Relationship Diagrams)

An Entity Relationship (ER) Diagram is a type of flowchart that illustrates how "entities" such as people, objects or concepts relate to each other within a system. ER Diagrams are most often used to design or debug relational databases in the fields of software engineering, business information systems, education and research.

There are several notation systems for creating ERD Diagrams, which are similar but vary in a few specifics. The style being used in the diagram presented below is "*Crow's Foot/Martin/Information Engineering style*"

Acronyms used in following diagram -

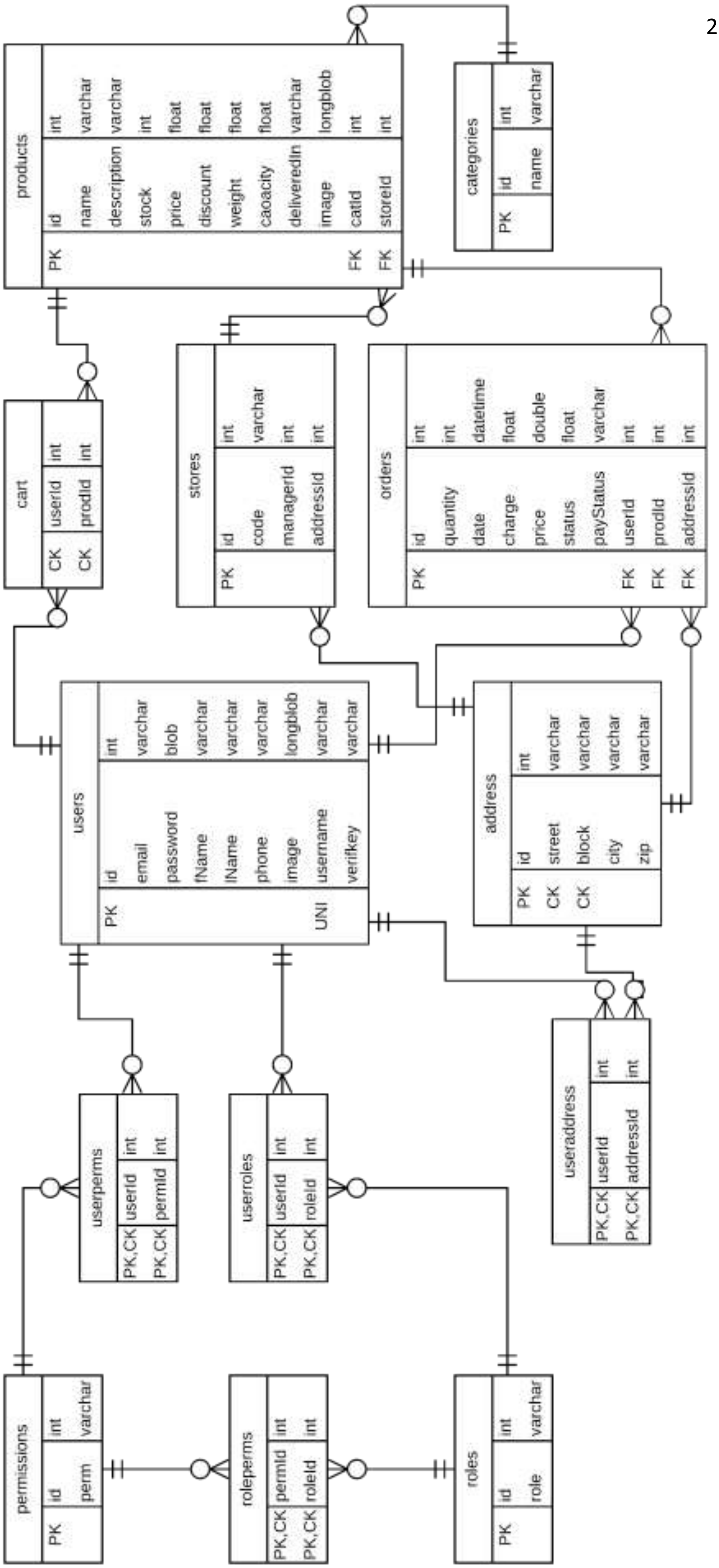
PK – Primary Key

CK – Composite Key

PK, CK – The Composite Key acting as a primary key.

FK – Foreign Key

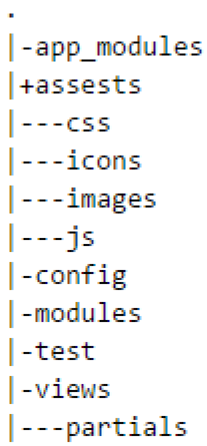
UNI – Unique Key


products
PK id int
name varchar
description varchar
stock int
price float
discount float
weight float
caoacity float
deliveredIn varchar
image longblob
FK catId int
FK storeId int
categories
PK id int
name varchar
cart
CK userId int
CK prodId int
stores
PK id int
code varchar
managerId int
addressId int
orders
PK id int
quantity int
date datetime
charge float
price double
status float
payStatus varchar
FK userId int
FK prodId int
FK addressId int
users
PK id int
email varchar
password blob
fName varchar
lName varchar
phone varchar
image longblob
UNI username varchar
verifkey varchar
address
PK id int
CK street varchar
CK block varchar
city varchar
zip varchar
userperms
PK,CK userId int
PK,CK permId int
userroles
PK,CK userId int
PK,CK roleId int
useraddress
PK,CK userId int
PK,CK addressId int
permissions
PK id int
perm varchar
roleperms
PK,CK permId int
PK,CK roleId int
roles
PK id int
role varchar

11. Project Design

11.1 Modularization Details

The overall structure of the application is described below.



```
.
|-app_modules
|+assests
|---css
|---icons
|---images
|---js
|-config
|-modules
|-test
|-views
|---partials
```

app_modules –This directory contains NodeJS modules that help in overall functioning of the application. These modules are written from scratch or on the top of other modules, and provide directly callable endpoints for performing tasks quickly without the need to write same code over and over.

assests – This directory contains CSS, Javascript and other related files that are to be served to the browser for the app to function properly.

config – This folder contains JSON, configuration files that containes sensitive information, like API Keys, Database Connection Information etc.

modules – This directory contains the core modules of the application, that provide the application a means to function. Each module has a specific set of tasks that is performed when a request at that certain route is made.

test – This module contains the code for testing the functioning of the app.

views – This specific directory contains EJS Snippets, that are compiled into HTML pages, using the data provided to them.

views/partials – This specific contains snippets that are combined or used repeatedly in order to form a complete view.

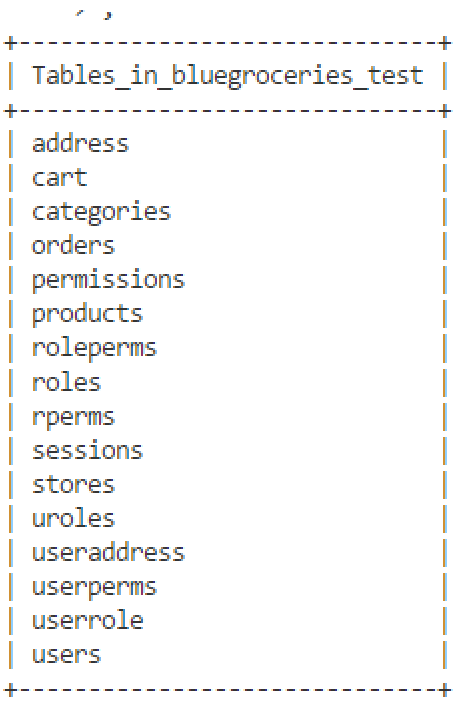
Some main files for the project

app.js > This is the main file and an entry point for the application.

package.json > This file contains all the meta information about the application.

11.2 Data Integrity and Constraints

Data Tables



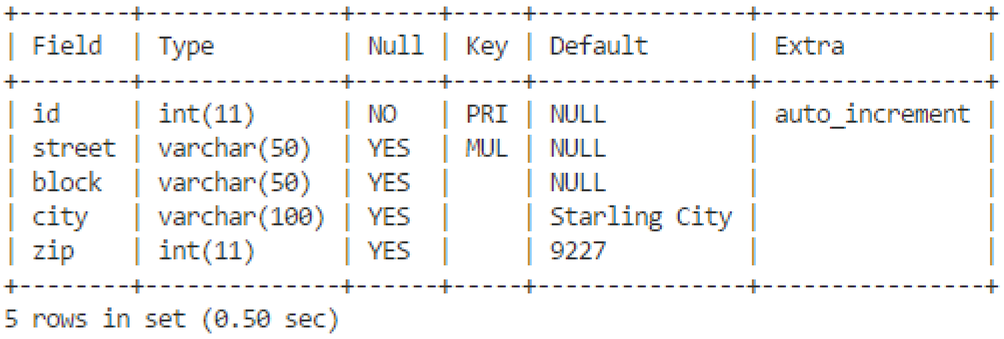
```
+-----------------------------+
| Tables_in_bluegroceries_test |
+-----------------------------+
| address                     |
| cart                        |
| categories                  |
| orders                      |
| permissions                 |
| products                    |
| roleperms                   |
| roles                       |
| rperms                      |
| sessions                    |
| stores                      |
| uroles                      |
| useraddress                 |
| userperms                   |
| userrole                    |
| users                       |
+-----------------------------+
```

This is a screen snippet from bluegroceries Database, using MySQL CLI.

> ‘rperms’ & ‘uroles’ are views instead of tables.

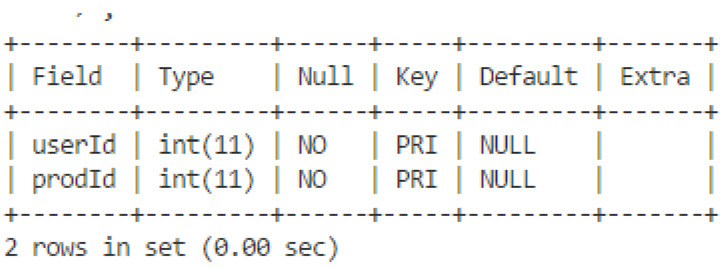
> ‘session’ is a table generated by express-mysql-session package, and will not be discussed.

Address Table



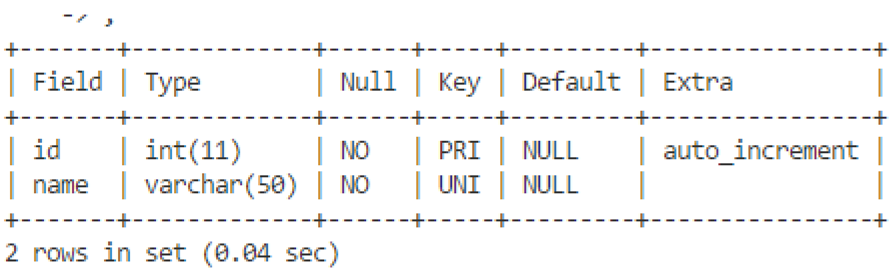
```
+--------+--------------+------+-----+--------------+----------------+
| Field  | Type         | Null | Key | Default      | Extra          |
+--------+--------------+------+-----+--------------+----------------+
| id     | int(11)      | NO   | PRI | NULL         | auto_increment |
| street | varchar(50)  | YES  | MUL | NULL         |                |
| block  | varchar(50)  | YES  |     | NULL         |                |
| city   | varchar(100) | YES  |     | Starling City |                |
| zip    | int(11)      | YES  |     | 9227         |                |
+--------+--------------+------+-----+--------------+----------------+
5 rows in set (0.50 sec)
```

Cart (Junction) Table



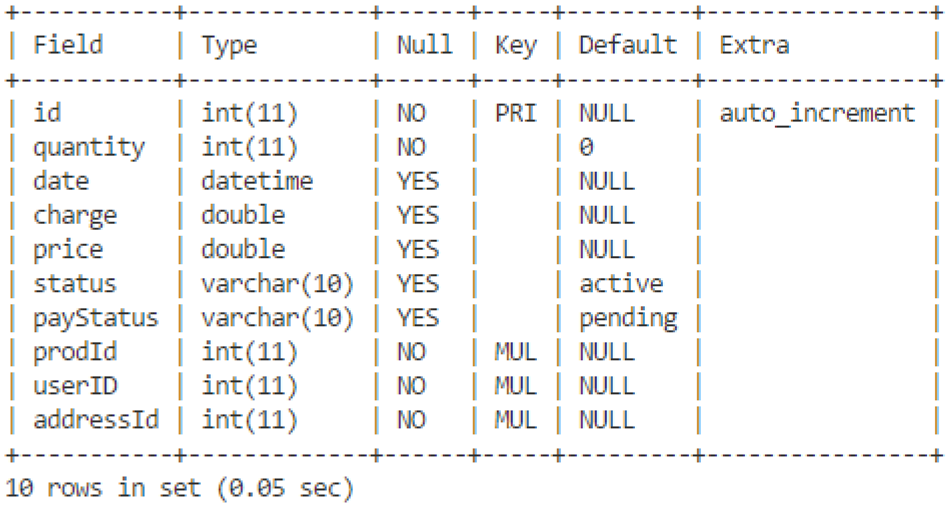
```
+--------+---------+------+-----+---------+-------+
| Field  | Type    | Null | Key | Default | Extra |
+--------+---------+------+-----+---------+-------+
| userId | int(11) | NO   | PRI | NULL    |       |
| prodId | int(11) | NO   | PRI | NULL    |       |
+--------+---------+------+-----+---------+-------+
2 rows in set (0.00 sec)
```

Categories Table



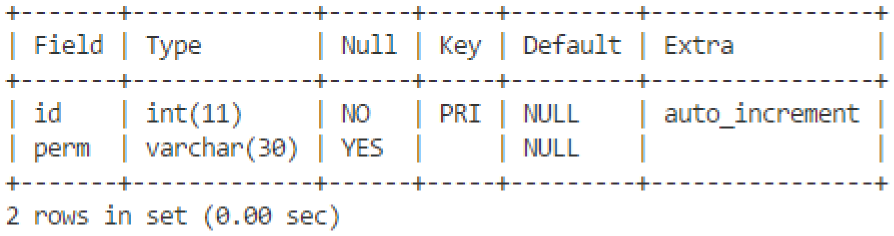
```
+-------+-------------+------+-----+---------+----------------+
| Field | Type        | Null | Key | Default | Extra          |
+-------+-------------+------+-----+---------+----------------+
| id    | int(11)     | NO   | PRI | NULL    | auto_increment |
| name  | varchar(50) | NO   | UNI | NULL    |                |
+-------+-------------+------+-----+---------+----------------+
2 rows in set (0.04 sec)
```

Orders Table



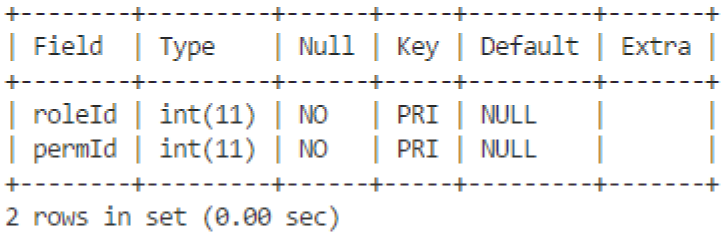
```
+-----------+-------------+------+-----+---------+----------------+
| Field     | Type        | Null | Key | Default | Extra          |
+-----------+-------------+------+-----+---------+----------------+
| id        | int(11)     | NO   | PRI | NULL    | auto_increment |
| quantity  | int(11)     | NO   |     | 0       |                |
| date      | datetime    | YES  |     | NULL    |                |
| charge    | double      | YES  |     | NULL    |                |
| price     | double      | YES  |     | NULL    |                |
| status    | varchar(10) | YES  |     | active  |                |
| payStatus | varchar(10) | YES  |     | pending |                |
| prodId    | int(11)     | NO   | MUL | NULL    |                |
| userID    | int(11)     | NO   | MUL | NULL    |                |
| addressId | int(11)     | NO   | MUL | NULL    |                |
+-----------+-------------+------+-----+---------+----------------+
10 rows in set (0.05 sec)
```

Permissions Table



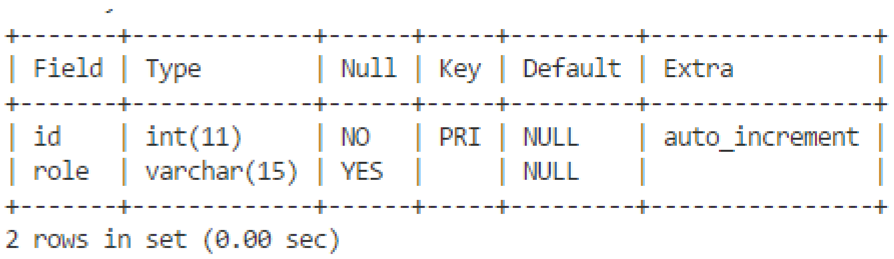
```
+-------+-------------+------+-----+---------+----------------+
| Field | Type        | Null | Key | Default | Extra          |
+-------+-------------+------+-----+---------+----------------+
| id    | int(11)     | NO   | PRI | NULL    | auto_increment |
| perm  | varchar(30) | YES  |     | NULL    |                |
+-------+-------------+------+-----+---------+----------------+
2 rows in set (0.00 sec)
```

Roleperms Table



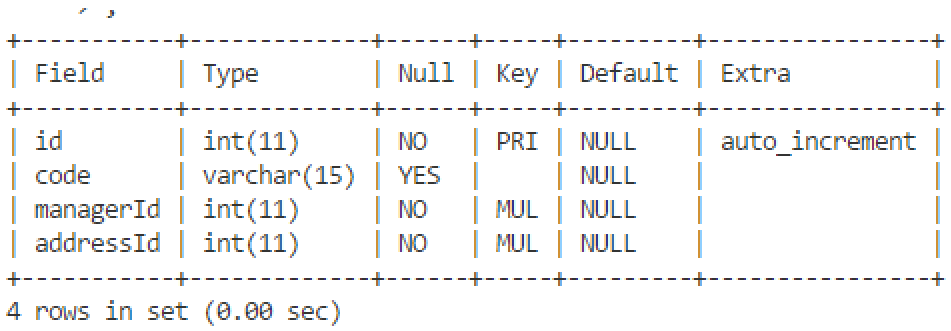
```
+--------+---------+------+-----+---------+-------+
| Field  | Type    | Null | Key | Default | Extra |
+--------+---------+------+-----+---------+-------+
| roleId | int(11) | NO   | PRI | NULL    |       |
| permId | int(11) | NO   | PRI | NULL    |       |
+--------+---------+------+-----+---------+-------+
2 rows in set (0.00 sec)
```

Roles Table



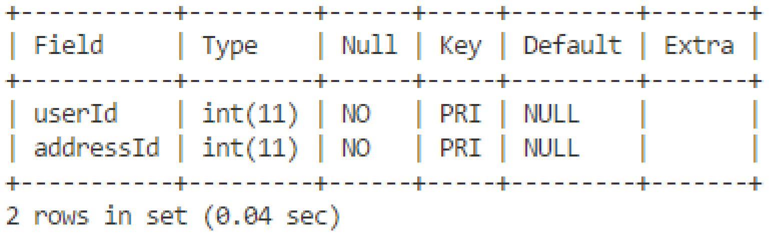
```
+-------+-------------+------+-----+---------+----------------+
| Field | Type        | Null | Key | Default | Extra          |
+-------+-------------+------+-----+---------+----------------+
| id    | int(11)     | NO   | PRI | NULL    | auto_increment |
| role  | varchar(15) | YES  |     | NULL    |                |
+-------+-------------+------+-----+---------+----------------+
2 rows in set (0.00 sec)
```

Stores Table



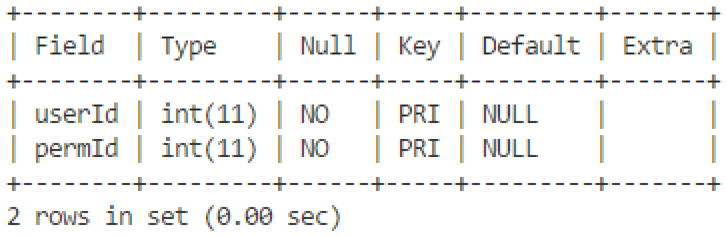
```
+-----------+-------------+------+-----+---------+----------------+
| Field     | Type        | Null | Key | Default | Extra          |
+-----------+-------------+------+-----+---------+----------------+
| id        | int(11)     | NO   | PRI | NULL    | auto_increment |
| code      | varchar(15) | YES  |     | NULL    |                |
| managerId | int(11)     | NO   | MUL | NULL    |                |
| addressId | int(11)     | NO   | MUL | NULL    |                |
+-----------+-------------+------+-----+---------+----------------+
4 rows in set (0.00 sec)
```

UserAddress (Junction) Table



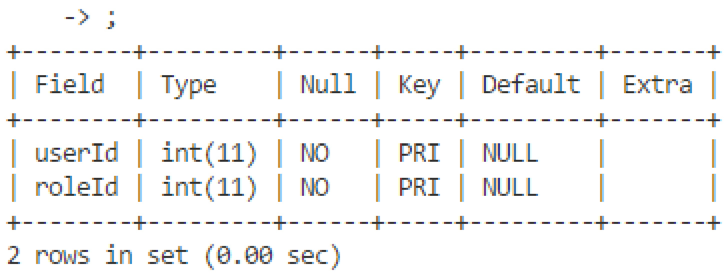
```
+-----------+---------+------+-----+---------+-------+
| Field     | Type    | Null | Key | Default | Extra |
+-----------+---------+------+-----+---------+-------+
| userId    | int(11) | NO   | PRI | NULL    |       |
| addressId | int(11) | NO   | PRI | NULL    |       |
+-----------+---------+------+-----+---------+-------+
2 rows in set (0.04 sec)
```

UserPerm (Junction) Table



```
+--------+---------+------+-----+---------+-------+
| Field  | Type    | Null | Key | Default | Extra |
+--------+---------+------+-----+---------+-------+
| userId | int(11) | NO   | PRI | NULL    |       |
| permId | int(11) | NO   | PRI | NULL    |       |
+--------+---------+------+-----+---------+-------+
2 rows in set (0.00 sec)
```

UserRole (Junction) Table



```
    -> ;
+--------+---------+------+-----+---------+-------+
| Field  | Type    | Null | Key | Default | Extra |
+--------+---------+------+-----+---------+-------+
| userId | int(11) | NO   | PRI | NULL    |       |
| roleId | int(11) | NO   | PRI | NULL    |       |
+--------+---------+------+-----+---------+-------+
2 rows in set (0.00 sec)
```

12. Security Features

1. The application uses REST API for communication with the application, so all the unwanted requests/routes are filtered and redirected to the Error pages.
2. All the data is validated, before and after sending it to the server,
3. Each function/request is made within a session, if the session doesn't exist it is created.
4. The session expires as soon as the user closes the application tab in the browser,
5. The session data is stored on the server rather than client, the client is provided only with session ID Key.
6. The payment/card information never touches the server,its is stored at a PCI Compliant Payment Processor, which then generated a token, and the token is then used to charge the user.
7. All the requests and inputs are checked for SQL injection attacks.
8. The data is verified before being stored.
9. The information about the user is not accessible anyone else, or any other user.

13. Limitations

1. The application is pretty basic, and does not allow moderators or adminstrators to modify database data.
2. The application requires SSL Transport Layer Security in order to work, so in case it is not provided some part of application may not function.
3. The application require Javascript both on client and server end to function properly, inability to do so may lead in broken application/features.
4. The application is not meant and compatible to IE and other outhdated browsers.
5. The application requires setCookie permission to work on the client.
6. The application only serves the purpose of buying products from Blue Groceries.

14. Future Scope

The application is open to further enhancements and features.

1. The roles and permissions can be implemented that may allows cutomers. Moderators, and administartors to work from the application.
2. The application can be optimized for mobile browser, providing quicker rendering.
3. The features for product rating and reviews can be added.
4. Local sellers can be approached to use the platform and sell alongwith the Blue Groceries Stores.

15. Bibliography

The resources used during development of the application and this documentation are as –

1. IGNOU's BCS 051 – Introduction to Software Engineering
2. Express JS Documentation: http://expressjs.com/
3. EJS Documentation: http://ejs.co/
4. Express-sessions Documentation: https://github.com/expressjs/session
5. MySQL JS Documentation: https://github.com/mysqljs/mysql
6. Express MySQL Session Documentation: https://www.npmjs.com/package/express-mysql-session
7. Stripe Documentation: https://stripe.com/docs
8. MailGun Tutorial - https://documentation.mailgun.com/en/latest/
9. MDN (Mozilla Develper Network) - https://developer.mozilla.org/en-US/
10. Medium Tutorials - https://medium.com/
11. The Programmers Hangout - Discord Programming Community Server
12. MySQL Docs - https://dev.mysql.com/doc/
13. Tutorialspoint Docs - https://www.tutorialspoint.com/index.htm