

A Project/Dissertation Report
on
HOME ESSENTIALS: AN ONLINE GROCERY STORE

*Submitted in partial fulfilment of the
requirement for the award of the degree of*

Bachelor of Technology (CSE)



(Established under Galgotias University Uttar Pradesh Act No. 14 of 2011)

Under The Supervision of
Dr. Amit Kumar Goel
Professor

Submitted By
Rahul Bakshi
18SCSE1010253

SCHOOL OF COMPUTING SCIENCE AND ENGINEERING
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
GALGOTIAS UNIVERSITY, GREATER NOIDA
INDIA
DECEMBER, 2021



**SCHOOL OF COMPUTING SCIENCE AND ENGINEERING
GALGOTIAS UNIVERSITY, GREATER NOIDA**

CANDIDATE'S DECLARATION

I hereby certify that the work which is being presented in the thesis/project/dissertation, entitled **“Home Essentials: An Online Grocery Store”** in partial fulfilment of the requirements for the award of the B.Tech (CSE) submitted in the School of Computing Science and Engineering of Galgotias University, Greater Noida, is an original work carried out during the period of July, 2021 to December 2021, under the supervision of **Dr. Amit Kumar Goel**, Professor, Department of Computer Science and Engineering of School of Computing Science and Engineering, Galgotias University, Greater Noida.

The matter presented in the thesis/project/dissertation has not been submitted by me for the award of any other degree of this or any other places.

Rahul Bakshi

18SCSE1010253

This is to certify that the above statement made by the candidates is correct to the best of my knowledge.

Dr, Amit Kumar Geol

Professor

ACKNOWLEDGEMENT

It gives me immense pleasure in bringing out this synopsis of the project entitled “Home Essentials: An Online Grocery Store”. Firstly, I would like to thank my teacher and guide **Dr. Amit Kumar Goel**, Professor, Galgotias University who gave me his valuable suggestions and ideas when we were in need of them. He encouraged me to work on this project. I’m also grateful to Galgotias University for giving me the opportunity to work with him and providing me the necessary resources for the project.

I am immensely grateful to all involved in this project as without their inspiration and valuable suggestion it would not have been possible to develop the project within the prescribed time.

With sincere thanks,

Rahul Bakshi

CERTIFICATE

The Final Thesis/Project/ Dissertation Viva-Voce examination of Rahul Bakshi 18SCSE1010253 has been held on _____ and his/her work is recommended for the award of Bachelor of Technology in Computer Science and Engineering.

Signature of Examiner(s)

Signature of Supervisor(s)

Signature of Project Coordinator

Signature of Dean

Date:

Place: Greater Noida

Abstract

The project Home Essential is an E-commerce platform for all the grocery needs. Online shopping has been reaching new products and markets each day. India has been adapting to these trends very well and there are millions of people who shop online today for various products ranging from electronics, health care, clothing etc. Shopping online for daily needs is the newer trends that been emerging over the time of past few years.

This project is a web-based grocery store where the user will be able to shop daily needs and requirements. The user can search for the product of their requirements and order them only using this platform. This has wide range of functionalities ranging from being able to search product, adding the selected items to the shopping cart, a checkout section and integration of payment gateway to pay for the order online saving the time and efforts of grocery shopping. This project is a web application and built using MERN (MongoDB, ExpressJS, React, NodeJS) and other utility framework like Tailwind CSS. The main objective of the project is to manage the details of Products, Customer, Shipping, Payment, Category. It manages all the information about Products, Sales, Category, Products. The purpose of the project is to build an application program to reduce the manual work for managing the Products, Customer, Sales, Shipping. It tracks all the details about the Shipping, Payment, Category.

Contents

Title	Page No.
Candidates Declaration	I
Acknowledgement	II
Certificate	III
Abstract	IV
List of Figures	V
Chapter 1 Introduction	1
1.1 Introduction	
1.2 Aim	
1.3 Identification of Need	
1.4 Literary Survey	
1.4.1 Feasibility Analysis	
Chapter 2 Project Design	4
2.1 User Flow Diagram	
2.2 ER Diagram	
Chapter 3 Tools and Technologies Used	6
3.1 Frontend	8
3.1.1 HTML	
3.1.2 Tailwind CSS	
3.1.3 JavaScript	
3.1.4 React	
3.2 Backend	12
3.2.1 Node JS	
3.2.2 Express JS	
3.3 Database	14
3.3.1 Mongo DB	
3.3.2 Mongoose	
3.4 Tools	16
3.4.1 VS Code	
3.4.2 Node Package Manager	
3.4.2 Postman	

Chapter 4	Functionality/Working of Project	20
	4.1 Module Description	
	4.1.1 Product Listing	
	4.1.2 Product Description	
	4.1.3 Shopping Cart	
	4.1.4 User Authentication	
	4.1.5 Purchase Flow	
	4.2 Database	28
	4.2.1 MongoDB Atlas and Compass	
	4.2.2 Collections	
	4.3 API Testing	33
	4.4 Instructions	35
Chapter 5	Result and Discussion	36
	5.1 Testing Methods	
	5.1.1 White Box Testing	
	5.1.2 Black Box Testing	
Chapter 6	Conclusion and Future Scope	39
	5.1 Conclusion	
	5.2 Future Scope	
	Reference	41

List of Figures

S. No.	Title	Page No.
1	User Flow Diagram	4
2	ER Diagram	5
3	MERN Stack Architecture	7
4	Product Listing Page	20
5	Product Description Page	21
6	Shopping Cart Page	22
7	User Login Page	23
8	User Sign Up Page	24
9	Shipping Details Page	25
10	Payment Method Page	26
11	Place Order Page	27
12	MongoDB Collections	29
13	Order Collection	30
14	Product Collection	31
15	User Collection	32
16	Product Route	33
17	Product Route with id	33
18	Order Details with id	34
19	User Token	34
20	Running Server	35
21	Open App in Browser	35

CHAPTER 1

INTRODUCTION

1.1 Introduction

The grocery shopping is probably one of the very essential parts for every household. It the first thing one does on receiving a pay check or salary. Now a days people tend shop for their needs from the comfort of your home itself using number of available Ecommerce Store around in matter of some click. The purchase ranges from technical gadgets, to skin care products, to many other things. The new addition to the list over the span last few years is India's e-grocery market is expected to be attractive for the next few years with more than 25 percent of the growth in the organized grocery segment projected to come from e-commerce. More than 25 percent of the organized grocery market's growth over the next few years could come from online shopping. For traditional modern trade players, this means e-commerce is no longer an option. It's a necessity. With the upcoming surge for the e-grocery shopping there we will requirement for more Ecommerce web site to bring the product from local retailer to the consumers over the web. This project is a web application and built using MERN (MongoDB, ExpressJS, React, NodeJS) and other utility framework like Tailwind CSS.

1.2 Aim

The Home Essentials a Grocery Shop Management System has been developed to override the problems prevailing in the practicing manual system. This software is supported to eliminate and, in some cases, reduce the hardships faced by this existing system. Moreover, this system is designed for the particular need of the company to carry out operations in a smooth and effective manner.

1.3 Identification of Need

The old manual system was suffering from a series of drawbacks. Since whole of the system was to be maintained with hands the process of keeping, maintaining and retrieving the information was very tedious and lengthy. The records were never used to be in a systematic order. there used to be lots of difficulties in associating any particular transaction with a particular context. If any information was to be found it was required to go through the different registers, documents there would never exist anything like report generation. There would always be unnecessary consumption of time while entering records and retrieving records, one more problem was that it was very difficult to find errors while entering the records. Once the records were entered it was very difficult to update these records.

The reason behind it is that there is lot of information to be maintained and have to be kept in mind while running the business for this reason we have provided features Present system is partially automated (computerized), actually existing system is quite laborious as one has to enter same information at three different places.

1.4 Literary Survey

1.4.1 Feasibility Study:

After doing the project Grocery Shop Management System, study and analysing all the existing or required functionalities of the system, the next task is to do the feasibility study for the project. All projects are feasible - given unlimited resources and infinite time.

Feasibility study includes consideration of all the possible ways to provide a solution to the given problem. The proposed solution should satisfy all the user requirements and should be flexible enough so that future changes can be easily done based on the future upcoming requirements.

A. Economic Feasibility

Development of this application is highly economically feasible. This is a very important aspect to be considered while developing a project based on minimum possible cost factor. The only thing is to be done is making an environment for the development with an effective supervision. If it is done so we can attain the maximum usability of the corresponding resources. Therefore, the system is economically feasible.

B. Technical Feasibility

On running feasibility analysis, the project was found to be technically feasible for the users as this application is a web-based application which means its platform independent and can be easily accessed using any operating system as windows, MacOS, Linux, Chrome OS and this application is also built with responsive design so it would be easily accessible using operating system such Android, IOS using just a web browser such Google Chrome, Mozilla Firefox, Safari, Opera etc. So, it would be fairly easy for the users to access the this without the limitation of any technology or device

CHAPTER 2 Product Design

2.1 User Flow Diagram

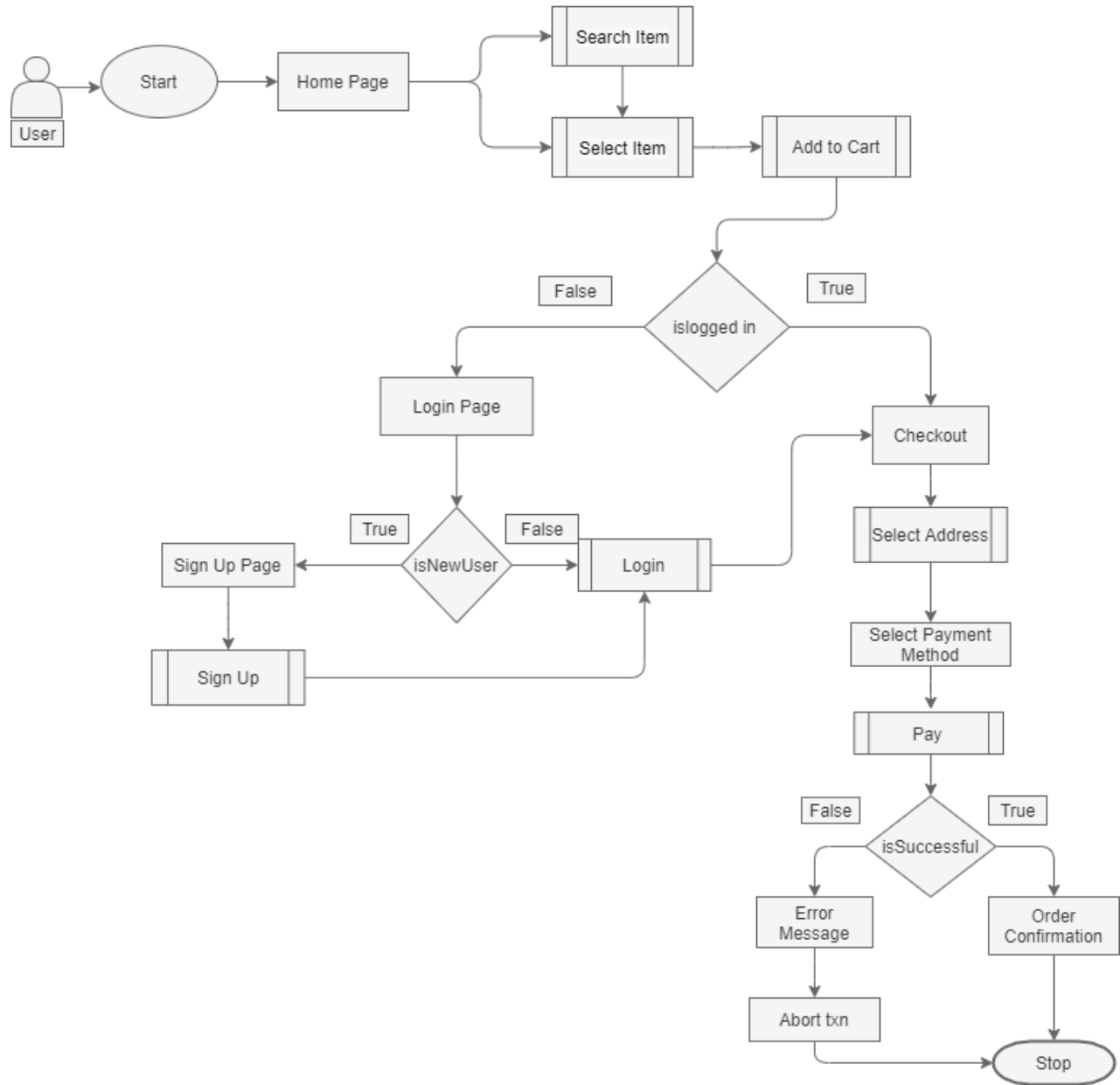


Figure 1
User Flow Diagram

2.1 ER – Diagram

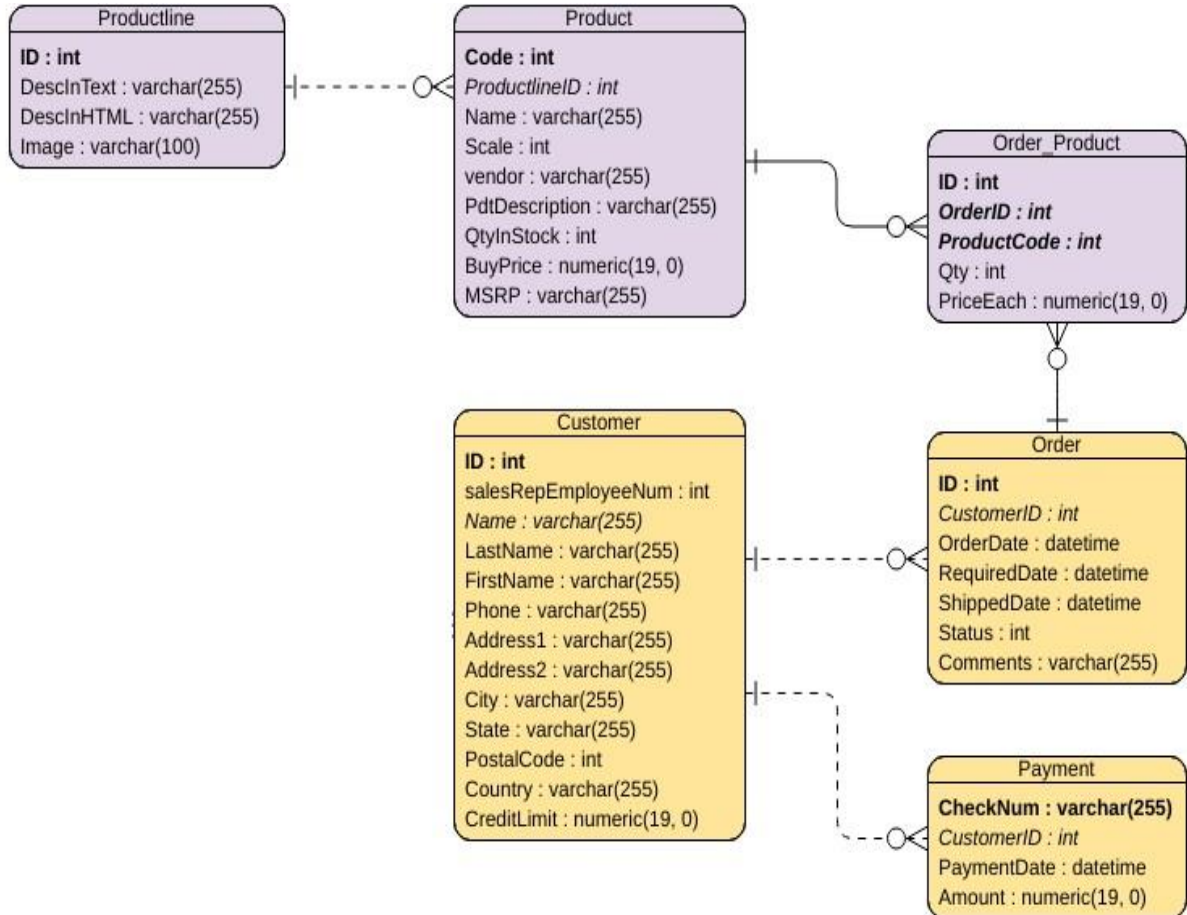


Figure 2
ER Diagram

CHAPTER 3

TOOLS AND TECHNOLOGIES USED

The main thing about an e-commerce website development is the trunk end functionality. You can find hundreds of tens of thousands of requests that ping every minute from all over the world. The backend must support the data fetching from the server and display it in the front end. In addition to that, there are certainly a lot of other elements such as for instance logistics, payment gateways, supplier management, and more. These sophisticated features take a toll on the entire performance of the site. So, to transport out such complex tasks and to keep powerful round the clock, the trunk end should be robust and scalable; otherwise, the front end also collapses.

Following are the technologies used for the development of this project:

MERN stack is a web development framework. It consists of MongoDB, ExpressJS, ReactJS, and NodeJS as its working components. Here are the details of what each of these components is used for in developing a web application when using MERN stack:

MongoDB: A document-oriented, No-SQL database used to store the application data.

NodeJS: The JavaScript runtime environment. It is used to run JavaScript on a machine rather than in a browser.

ExpressJS: A framework layered on top of NodeJS, used to build the backend of a site using NodeJS functions and structures. Since NodeJS was not developed to make websites but rather run JavaScript on a machine, ExpressJS was developed.

ReactJS: A library created by Facebook. It is used to build UI components that create the user interface of the single page web application.

The user interacts with the ReactJS UI components at the application front-end residing in the browser. This frontend is served by the application backend residing in a server, through ExpressJS running on top of NodeJS.

Any interaction that causes a data change request is sent to the NodeJS based Express server, which grabs data from the MongoDB database if required, and returns the data to the frontend of the application, which is then presented to the user.



Figure 3
MERN Stack Architecture

3.1 Frontend

3.1.1 HTML

HTML is a *markup language* that defines the structure of your content. HTML consists of a series of elements, which you use to enclose, or wrap, different parts of the content to make it appear a certain way, or act a certain way. The enclosing tags can make a word or image hyperlink to somewhere else, can italicize words, can make the font bigger or smaller, and so on.

HTML (HyperText Markup Language) is the most basic building block of the Web. It defines the meaning and structure of web content. "Hypertext" refers to links that connect web pages to one another, either within a single website or between websites. Links are a fundamental aspect of the Web.

3.1.2 Tailwind CSS

Tailwind is a utility-first CSS framework. In contrast to other CSS frameworks like Bootstrap or Materialize CSS it doesn't come with predefined components. Instead, Tailwind CSS operates on a lower level and provides you with a set of CSS helper classes. By using these classes, you can rapidly create custom design with ease. Tailwind CSS is not opinionated and lets you create your own unique design.

Tailwind CSS works by scanning all of your HTML files, JavaScript components, and any other templates for class names, generating the corresponding styles and then writing them to a static CSS file.

It's fast, flexible, and reliable — with zero-runtime.

Installation:

The simplest and fastest way to get up and running with Tailwind CSS from scratch is with the Tailwind CLI tool.

1. Install Tailwind CSS:

Install `tailwindcss` via npm, and create your `tailwind.config.js` file.

In the terminal run the following commands

```
npm install -D tailwindcss
```

```
npx tailwindcss init
```

2. Configure your template path:

Add the paths to all of your template files in your `tailwind.config.js` file

```
module.exports = {  
  content: ["/src/**/*.{html,js}"],  
  theme: {  
    extend: {},  
  },  
  plugins: [],  
}
```

3. Add the Tailwind directives to your CSS:

Add the `@tailwind` directives for each of Tailwind's layers to your main CSS file.

```
@tailwind base;
```

```
@tailwind components;
```

```
@tailwind utilities;
```

4. Start the Tailwind CLI build process:

Run the CLI tool to scan your template files for classes and build your CSS.

In the terminal run the following commands

```
npx tailwindcss -i ./src/input.css -o ./dist/output.css -watch
```

3.1.3 React

React is a JavaScript library for building user interfaces. React stands at the intersection of design and programming. It lets you take a complex user interface, and break it down into nestable and reusable pieces called “components” that fit well together.

- React makes it painless to create interactive UIs. Design simple views for each state in your application, and React will efficiently update and render just the right components when your data changes. Declarative views make your code more predictable, simpler to understand, and easier to debug.
- Build encapsulated components that manage their own state, then compose them to make complex UIs. Since component logic is written in JavaScript instead of templates, you can easily pass rich data through your app and keep state out of the DOM.

Installation:

Create React App is a comfortable environment for learning React, and is the best way to start building a new single-page application in React.

It sets up your development environment so that you can use the latest JavaScript features, provides a nice developer experience, and optimizes your app for production. You'll need to have Node \geq 14.0.0 and npm \geq 5.6 on your machine.

To create a project, run:

```
npx create-react-app my-app  
cd my-app  
npm start
```

Then open <http://localhost:3000/> to see your app.

When you're ready to deploy to production, create a minified bundle with

```
npm run build.
```

3.1.4 JavaScript

JavaScript is a high-level, interpreted scripting language that conforms to the ECMAScript specification. JavaScript has curly-bracket syntax, dynamic typing, prototype-based object-orientation, and first-class functions. Alongside HTML and CSS, JavaScript is one of the core technologies of the World Wide Web. JavaScript enables interactive web pages and is an essential part of web applications. The vast majority of websites use it, and major web browsers have a dedicated JavaScript engine to execute it. As a multi-paradigm language, JavaScript supports event-driven, functional, and imperative (including object-

oriented and prototype-based) programming styles. It has APIs for working with text, arrays, dates, regular expressions, and the DOM, but the language itself does not include any I/O, such as networking, storage, or graphics facilities. It relies upon the host environment in which it is embedded to provide these features.

Initially only implemented client-side in web browsers, JavaScript engines are now embedded in many other types of host software, including server-side in web servers and databases, and in non-web programs such as word processors and PDF software, and in runtime environments that make JavaScript available for writing mobile and desktop applications, including desktop widgets.

The terms Vanilla JavaScript and Vanilla JS refer to JavaScript not extended by any frameworks or additional libraries. Scripts written in Vanilla JS are plain JavaScript code. Google's Chrome extensions, Opera's extensions, Apple's Safari 5 extensions, Apple's Dashboard Widgets, Microsoft's Gadgets, Yahoo! Widgets, Google Desktop Gadgets, and Serence Klipfolio are implemented using JavaScript.

3.2 Backend

3.2.1 NodeJS

Node.js is an open-source and cross-platform JavaScript runtime environment. It is a popular tool for almost any kind of project!

Node.js runs the V8 JavaScript engine, the core of Google Chrome, outside of the browser. This allows Node.js to be very performant.

A Node.js app runs in a single process, without creating a new thread for every request. Node.js provides a set of asynchronous I/O primitives in its standard

library that prevent JavaScript code from blocking and generally, libraries in Node.js are written using non-blocking paradigms, making blocking behavior the exception rather than the norm.

When Node.js performs an I/O operation, like reading from the network, accessing a database or the filesystem, instead of blocking the thread and wasting CPU cycles waiting, Node.js will resume the operations when the response comes back. This allows Node.js to handle thousands of concurrent connections with a single server without introducing the burden of managing thread concurrency, which could be a significant source of bugs.

Node.js has a unique advantage because millions of frontend developers that write JavaScript for the browser are now able to write the server-side code in addition to the client-side code without the need to learn a completely different language.

3.2.2 Express JS

Express is the most popular Node web framework, and is the underlying library for a number of other popular Node web frameworks. It provides mechanisms to:

- Write handlers for requests with different HTTP verbs at different URL paths (routes).
- Integrate with "view" rendering engines in order to generate responses by inserting data into templates.
- Set common web application settings like the port to use for connecting, and the location of templates that are used for rendering the response.

- Add additional request processing "middleware" at any point within the request handling pipeline.

While Express itself is fairly minimalist, developers have created compatible middleware packages to address almost any web development problem. There are libraries to work with cookies, sessions, user logins, URL parameters, POST data, security headers, and many more.

Installation:

Use the following command to install express:

```
npm install express --save
```

3.3 Database

3.3.1 MongoDB

MongoDB is a document-oriented NoSQL database used for high volume data storage. Instead of using tables and rows as in the traditional relational databases, MongoDB makes use of collections and documents. Documents consist of key-value pairs which are the basic unit of data in MongoDB. Collections contain sets of documents and function which is the equivalent of relational database tables. MongoDB is a database which came into light around the mid-2000s.

Features of MongoDB:

1. Each database contains collections which in turn contains documents. Each document can be different with a varying number of fields. The size and content of each document can be different from each other.

2. The document structure is more in line with how developers construct their classes and objects in their respective programming languages. Developers will often say that their classes are not rows and columns but have a clear structure with key-value pairs.
3. The rows (or documents as called in MongoDB) doesn't need to have a schema defined beforehand. Instead, the fields can be created on the fly.
4. The data model available within MongoDB allows you to represent hierarchical relationships, to store arrays, and other more complex structures more easily.
5. Scalability – The MongoDB environments are very scalable. Companies across the world have defined clusters with some of them running 100+ nodes with around millions of documents within the database.

3.3.2 Mongoose

Mongoose is an Object Document Mapper (ODM). This means that Mongoose allows you to define objects with a strongly-typed schema that is mapped to a MongoDB document.

Mongoose provides an incredible amount of functionality around creating and working with schemas. Mongoose currently contains eight Schema Types that a property is saved as when it is persisted to MongoDB. They are:

- String
- Number
- Date
- Buffer
- Boolean
- Mixed
- ObjectId
- Array

3.4 Tools

3.4.1 VS Code

Visual Studio Code is a source-code editor made by Microsoft for Windows, Linux and macOS. Features include support for debugging, syntax highlighting, intelligent code completion, snippets, code refactoring, and embedded Git. Users can change the theme, keyboard shortcuts, preferences, and install extensions that add additional functionality. Visual Studio Code was first announced on April 29, 2015, by Microsoft at the 2015 Build conference. A preview build was released shortly thereafter.

On November 18, 2015, the source of Visual Studio Code was released under the MIT License, and made available on GitHub. Extension support was also announced. On April 14, 2016, Visual Studio Code graduated from the public preview stage and was released to the Web.[12] Microsoft has released most of Visual Studio Code's source code on GitHub under the permissive MIT License, while the releases by Microsoft are proprietary freeware.

In the Stack Overflow 2021 Developer Survey, Visual Studio Code was ranked the most popular developer environment tool, with 70% of 82,000 respondents reporting that they use it.

3.4.2 NPM (Node Package Manager):

npm is the standard package manager for Node.js. In January 2017 over 350000 packages were reported being listed in the npm registry, making it the biggest

single language code repository on Earth, and you can be sure there is a package for (almost!) everything.

It started as a way to download and manage dependencies of Node.js packages, but it has since become a tool used also in frontend JavaScript.

The public npm registry is a database of JavaScript packages, each comprised of software and metadata. Open-source developers and developers at companies use the npm registry to contribute packages to the entire community or members of their organizations, and download packages to use in their own projects.

A **package** is a file or directory that is described by a `package.json` file. A package must contain a `package.json` file in order to be published to the npm registry. For more information on creating a `package.json` file, see "Creating a `package.json` file".

npm manages downloads of dependencies of your project.

Installing all dependencies:

```
npm install
```

it will install everything the project needs, in the `node_modules` folder, creating it if it's not existing already.

Installing a single package:

```
npm install <package-name>
```

3.4.3 Postman

Postman is an application used for API testing. It is an HTTP client that tests HTTP requests, utilizing a graphical user interface, through which we obtain different types of responses that need to be subsequently validated.

Postman offers many endpoint interaction methods. The following are some of the most used, including their functions:

- GET: Obtain information
- POST: Add information
- PUT: Replace information
- PATCH: Update certain information
- DELETE: Delete information

When testing APIs with Postman, we usually obtain different response codes. Some of the most common include:

100 Series: Temporal responses, for example, ‘102 Processing’.

200 Series: Responses where the client accepts the request and the server processes it successfully, for instance, ‘200 Ok’.

300 Series: Responses related to URL redirection, for example, ‘301 Moved Permanently.’

400 Series: Client error responses, for instance, ‘400 Bad Request’.

500 Series: Server error responses, for example, ‘500 Internal Server Error.’

Postman gives the possibility to group different requests. This feature is known as ‘collections’ and helps organize tests. These collections are folders where requests are stored and can be structured in whichever way the team prefers. It is also

possible to export-import them. Postman also allows us to create different environments through the generation/use of variables; for example, a URL variable that is aimed towards different test environments (dev-QA), enabling us to execute tests in different environments using existing requests.

CHAPTER 4

Functionality/ Working of the Project

3.1 Module Description

The project has a large scope and has number of functionality to it. Thus project have a multiple number of module to it. This website is divided up in modules according to the user flow. The general user flow starts with landing/ product listing page, product description page, shopping cart, authentication, shipping details, payment method selection, order checkout page.

3.1.1 Product Listing

This is the landing page of the website i.e the first page that is viewed by the user on visiting this website. The user is able to look through the product available for purchase and decide upon the desirable product of choice. The page has a navbar with different nagivation option as Sign In and Cart. If the user is logged in the username will be displayed on the navigation bar itself are a Sign In button will be displayed.

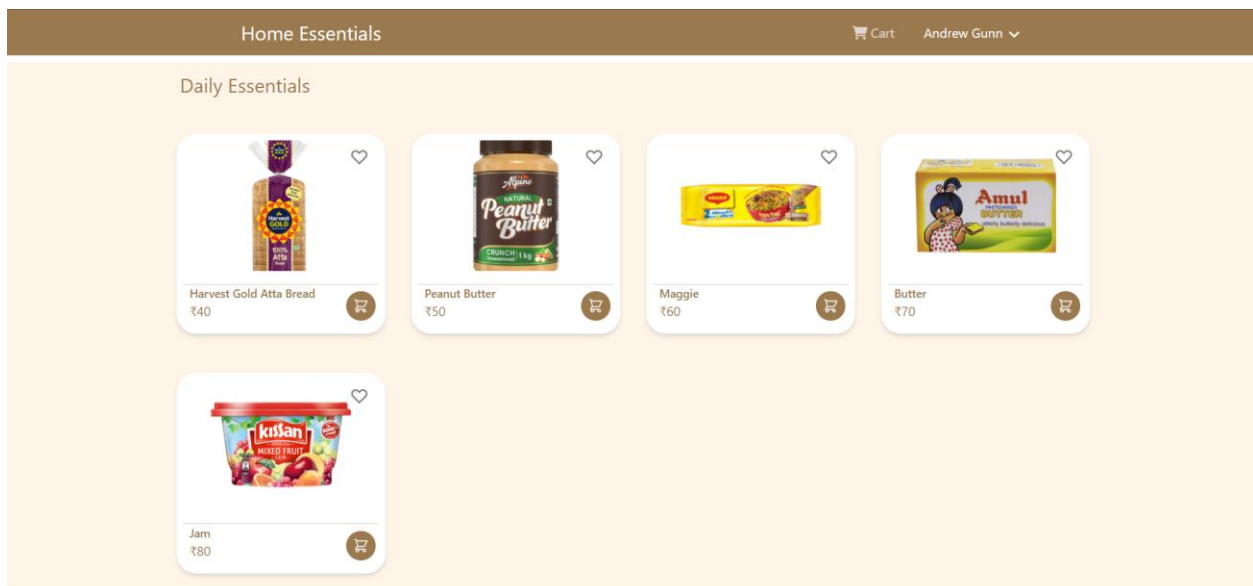


Figure 4
Product Listing Page

3.1.2 Product Description

From the product listing page when the users chooses the desired product they are navigated to the product description page. The product page as the name suggest has the description related to the product of choice. This page has product image Product name , product description and quantity dropdown menu to choose the desired quantity of the product. Once the users have the quantity set they can hit the add to cart button to add the selected product to the shopping cart.

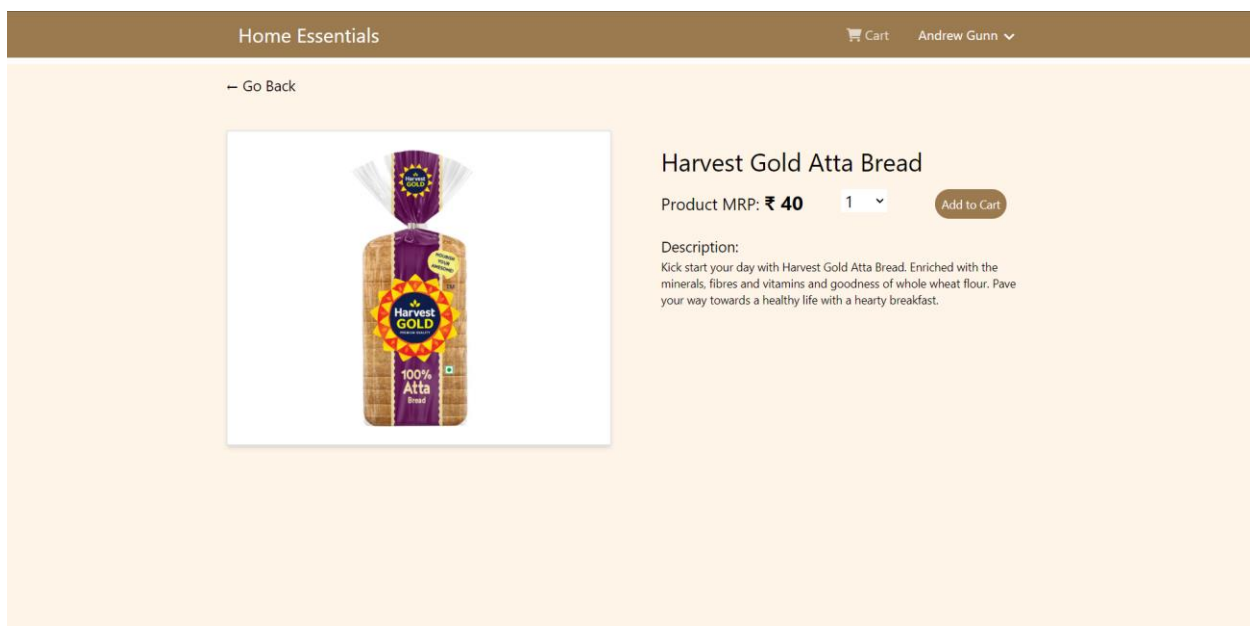


Figure 5
Product Description Page

3.1.3 Shopping Cart

Once the users hit the add to cart button they are navigated to the shopping cart page. The shopping cart page has the list of all the product selected by the user with selected quantity, The page also has a summary section where the summary of the purchase is listed for the user convenience to have a quick glance on there purchase history. The page also has a continue shopping navigation link which takes users back to the product listing page in case they want to add more product to the cart.

If the everything seems fine to the users they can hit the checkout button to proceed further with there purchase.

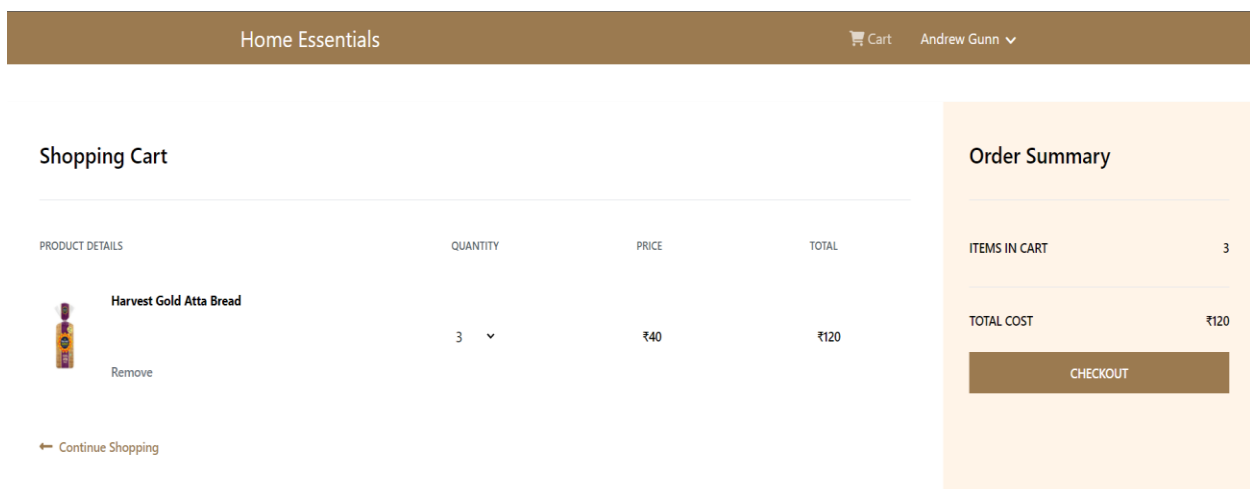
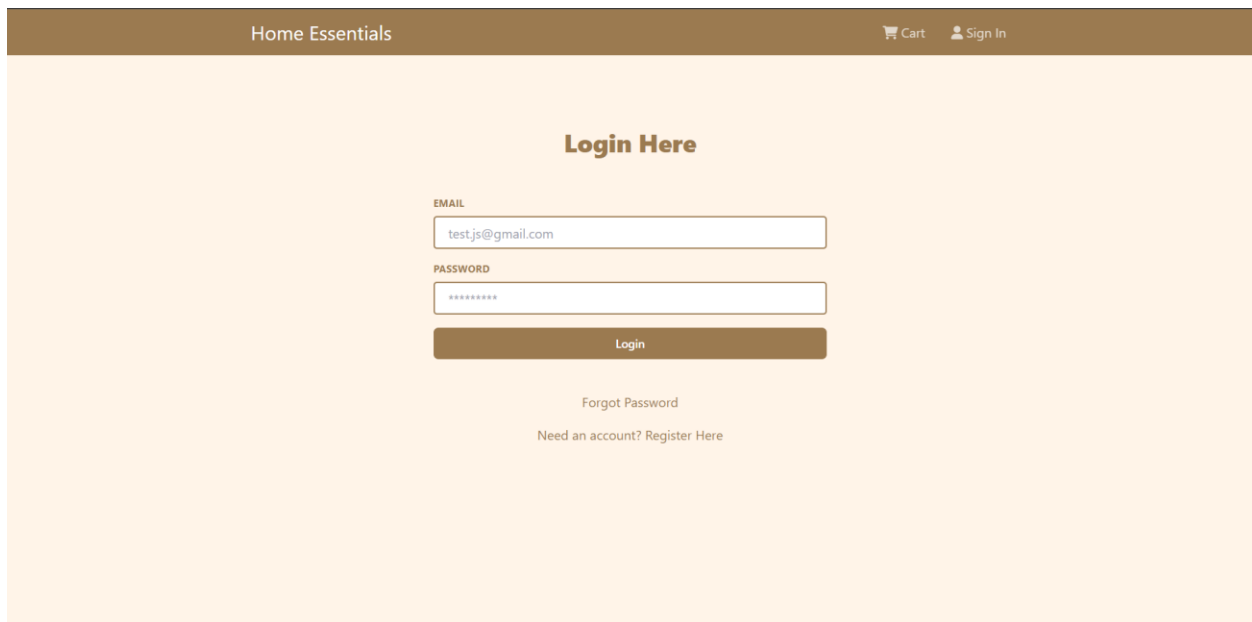


Figure 6
Shopping Cart Page

3.1.4 User Authentication

Login Page:

Once the users hit the checkout button on the shopping cart page there are redirected to the login page for authentication. If the user is already a customer the user can easily login with there credential to proceed with the purchase of the product selected by them .

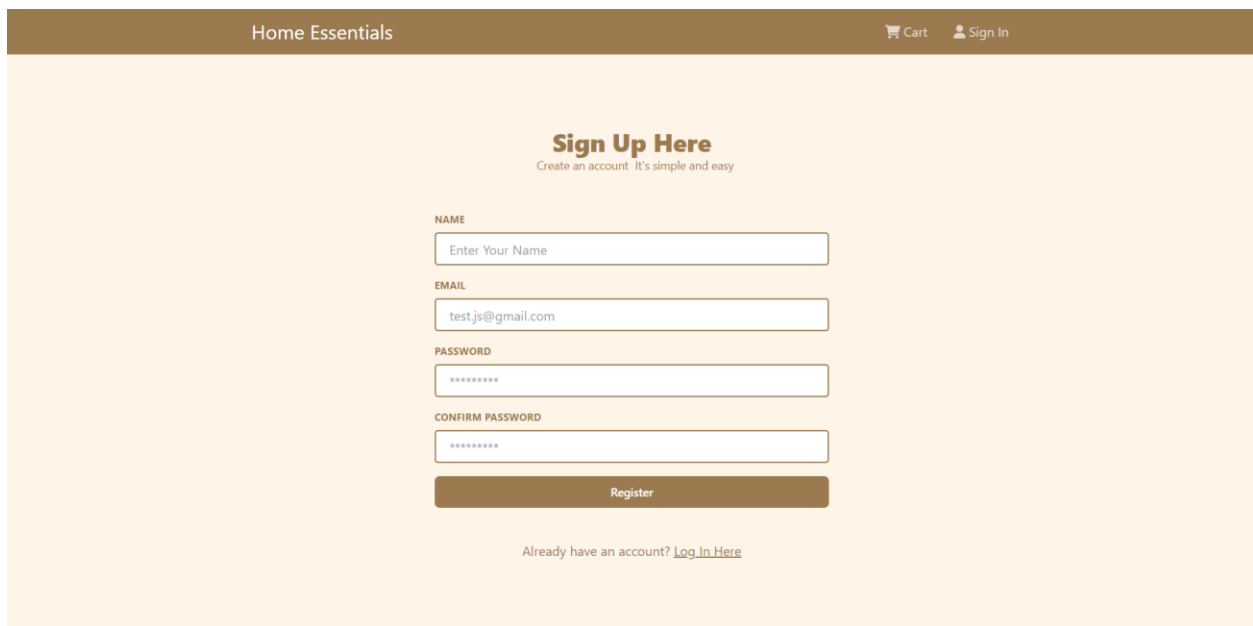


The image shows a user login page for 'Home Essentials'. The page has a dark brown header with 'Home Essentials' on the left and 'Cart' and 'Sign In' on the right. The main content area is light beige. In the center, there is a 'Login Here' heading. Below it are two input fields: 'EMAIL' with the value 'testjs@gmail.com' and 'PASSWORD' with masked characters '*****'. A dark brown 'Login' button is positioned below the password field. At the bottom of the form area, there are two links: 'Forgot Password' and 'Need an account? Register Here'.

Figure 7
User Login Page

Sign Up Page:

If the user doesn't already have an user account they are redirected to the Sign Up page where they can register. The user is rrequired to provide username,email, password and then confirm password to register for a user account



The image shows a web page for 'Home Essentials' with a sign-up form. The page has a dark brown header with the text 'Home Essentials' on the left and 'Cart' and 'Sign In' on the right. The main content area is light orange and features the heading 'Sign Up Here' with the subtext 'Create an account. It's simple and easy'. Below this are four input fields: 'NAME' with the placeholder 'Enter Your Name', 'EMAIL' with the placeholder 'test.js@gmail.com', 'PASSWORD' with a masked password '*****', and 'CONFIRM PASSWORD' with a masked password '*****'. A dark brown 'Register' button is positioned below the fields. At the bottom, there is a link: 'Already have an account? [Log In Here](#)'.

Figure 8
User Sign Up Page

3.1.5 Purchase Flow

Shipping Details:

Once the user is authenticated, they have done the first step of the purchase flow now are redirected to the shipping page details page they where are asked about their shipping details such as address, city , country , postal code etc. These are then stored to the local storage for future user reference. Once the users have filed in all the details they are redirected to the payment method selection page.

Home Essentials Cart Andrew Gunn ▼

SIGN IN SHIPPING PAYMENT PLACE ORDER

Shipping Details

ADDRESS

CITY

POSTAL CODE

COUNTRY

Figure 9
Shipping Details Page

Select Payment Method:

Once the users successfully enter their shipping details then they are further redirected to the page for payment method selection. By default paypal and credit card option is checked already as it currently the mode of payment for the project. But more methods can be added on the further stages of the project development. Once the payment method is decided the users can hit the submit button to proceed with there purchase.

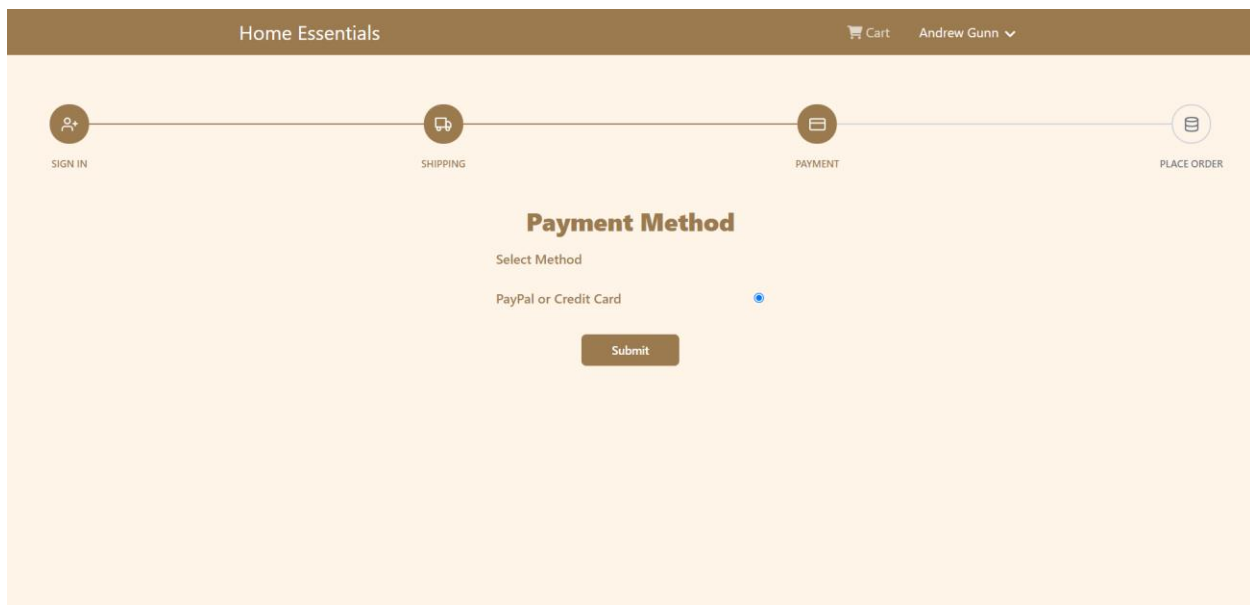


Figure 10
Payment Method Page

After shipping details are saved and payment method is selected then the users get to the place order screen where they can finally view a summary of their purchase with all the details and the total cost of the product after adding all the added service charges.

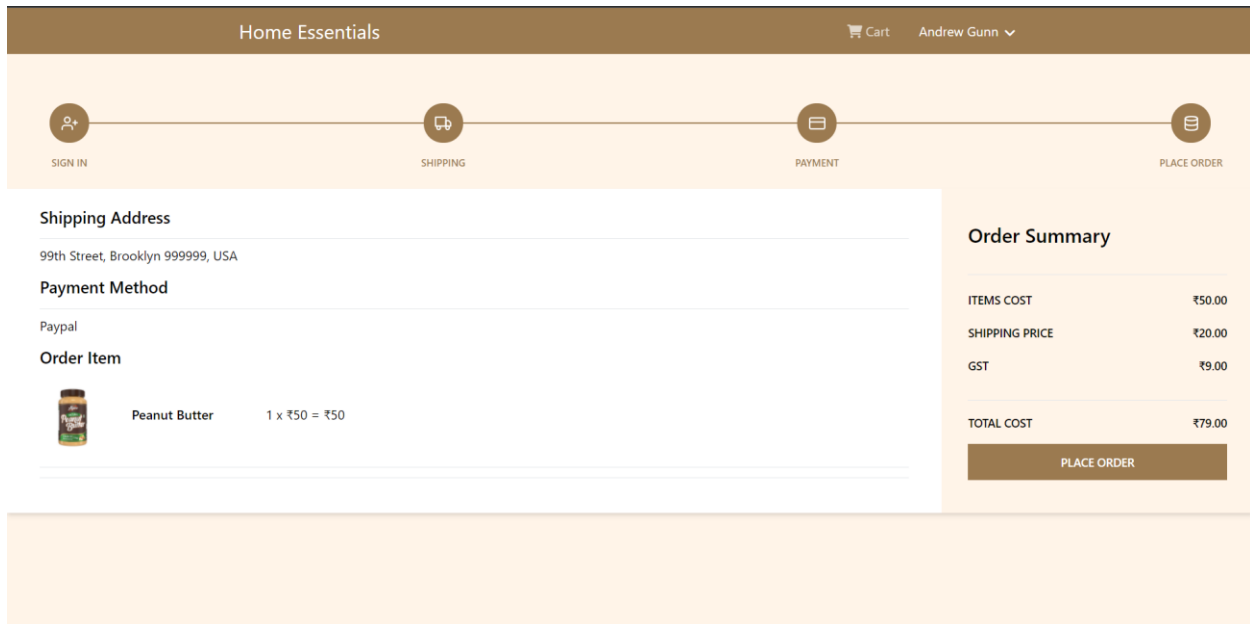


Figure 11
Place Order Page

A navigation bar goes along with this whole purchase flow with icons displaying at what point a user the user is at in the purchase. These navigation bar with checkout flow has sign in , shipping, payment and place order as the display icons showing how the purchase flow goes in. The icon are by default inactive until the user reached that particular point of purchase in the purchase flow.

4.2 Database

A “cloud database” can be one of two distinct things: a traditional or NoSQL database installed and running on a cloud virtual machine (be it public cloud, private cloud, or hybrid cloud platforms), or a cloud provider’s fully managed database-as-a-service (DBaaS) offering. The former, running your own self-managed database in a cloud environment, is really no different from operating a traditional database. Cloud DbaaS, on the other hand, is the natural database equivalent of software-as-a-service (SaaS): pay as you go, and only for what you use, and let the system handle all the details of provisioning and scaling to meet demand, while maintaining consistently high performance.

4.2.1 MongoDB Atlas and Compass:

MongoDB Atlas

The project uses MongoDB Atlas as the cloud database provider. MongoDB Atlas, a fully managed cloud database for modern applications. Atlas is the best way to run MongoDB, the leading modern database. MongoDB’s document model is the fastest way to innovate, bringing flexibility and ease of use to the database.

MongoDB Compass

MongoDB Compass is a powerful GUI for querying, aggregating, and analysing your MongoDB data in a visual environment. Compass is free to use and source available, and can be run on macOS, Windows, and Linux.

4.2.2 Collections:

Collections is grouping of MongoDB documents. A collection is the equivalent of an RDBMS table. A collection exists within a single database. Collections do not enforce a schema. Documents within a collection can have different fields. Typically, all documents in a collection have a similar or related purpose.

In the project database there are currently different collections to store different database from the website.

The current used collections are as under:

- Orders Collection
- Products Collection
- Users Collection

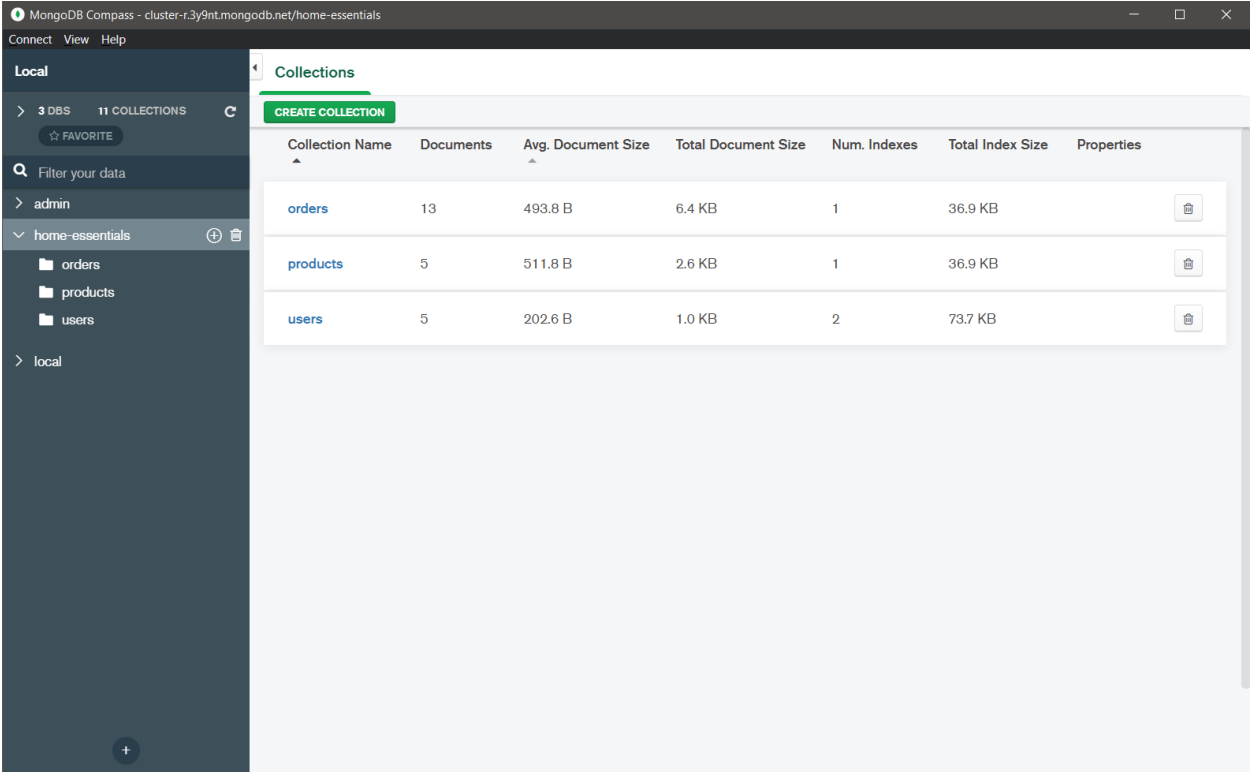


Figure 12
MongoDB Collections

Order Collection:

Order Collection is the collection of all the orders placed by users with the entry field of data as order id , user id,order item, shipping address, payment method, tax price, total cost, is paid , is delivered.

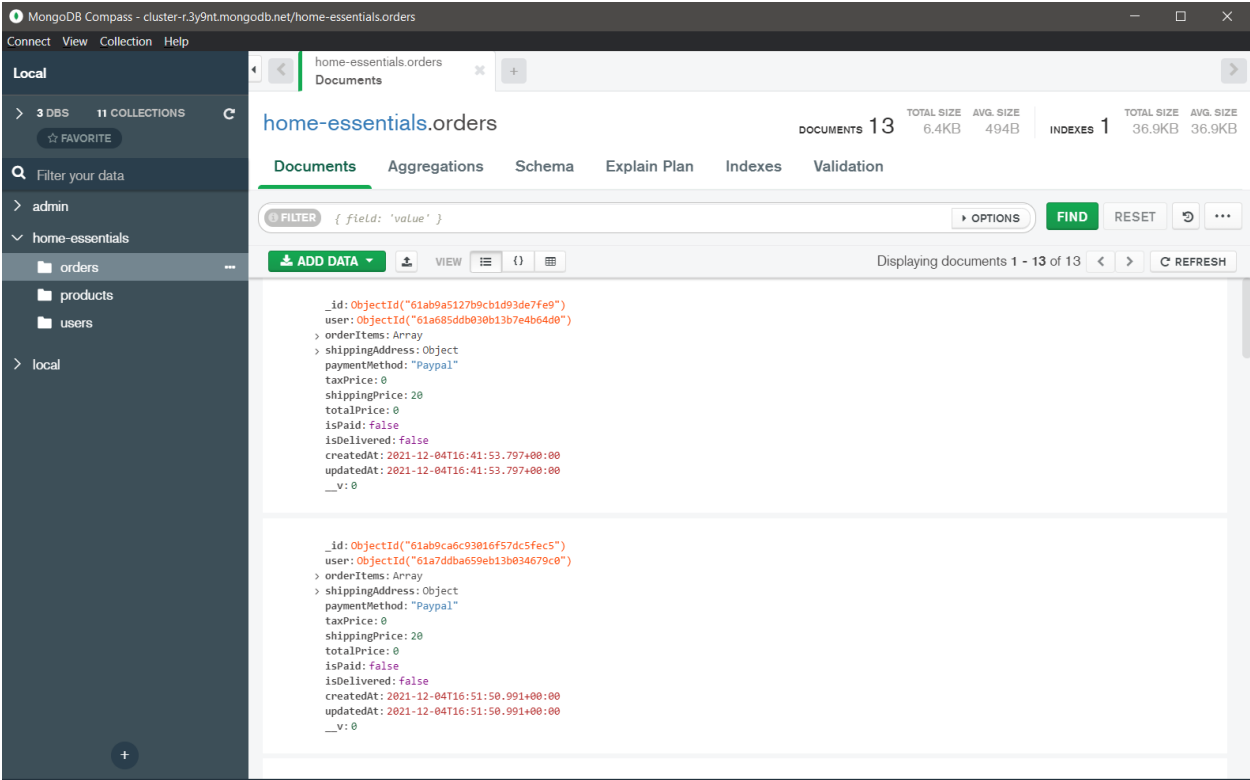


Figure 13
Order Collection

Product Collection:

Product Collection is the collection of all the products listed on the website with the entry field of data as product id, description , inStock status, product price, name , image.

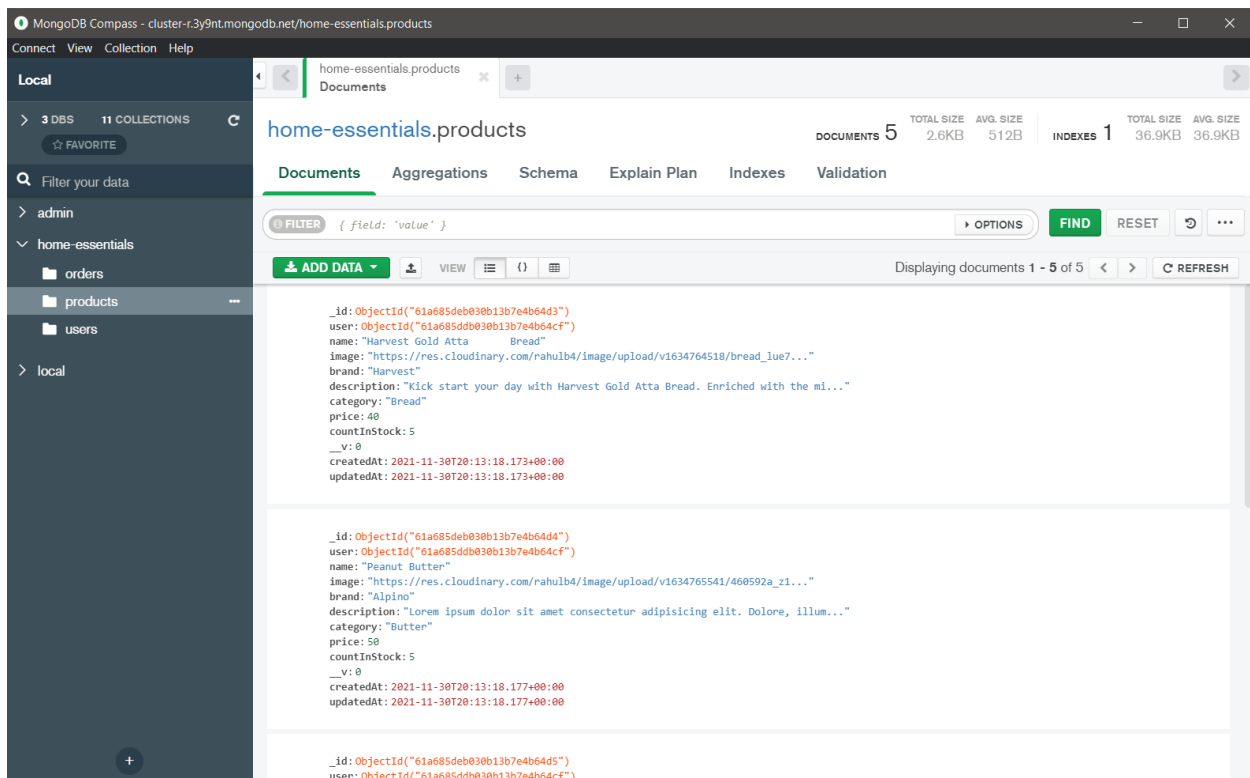


Figure 14
Product Collection

User Collection:

User collection is the collection of all the registered user with the field of there credential such as email,name,password , user id , isadmin etc.

Passwords are first encrypted then stored in the data based in its encoding form.

The project uses bcrypt to has the passwords.

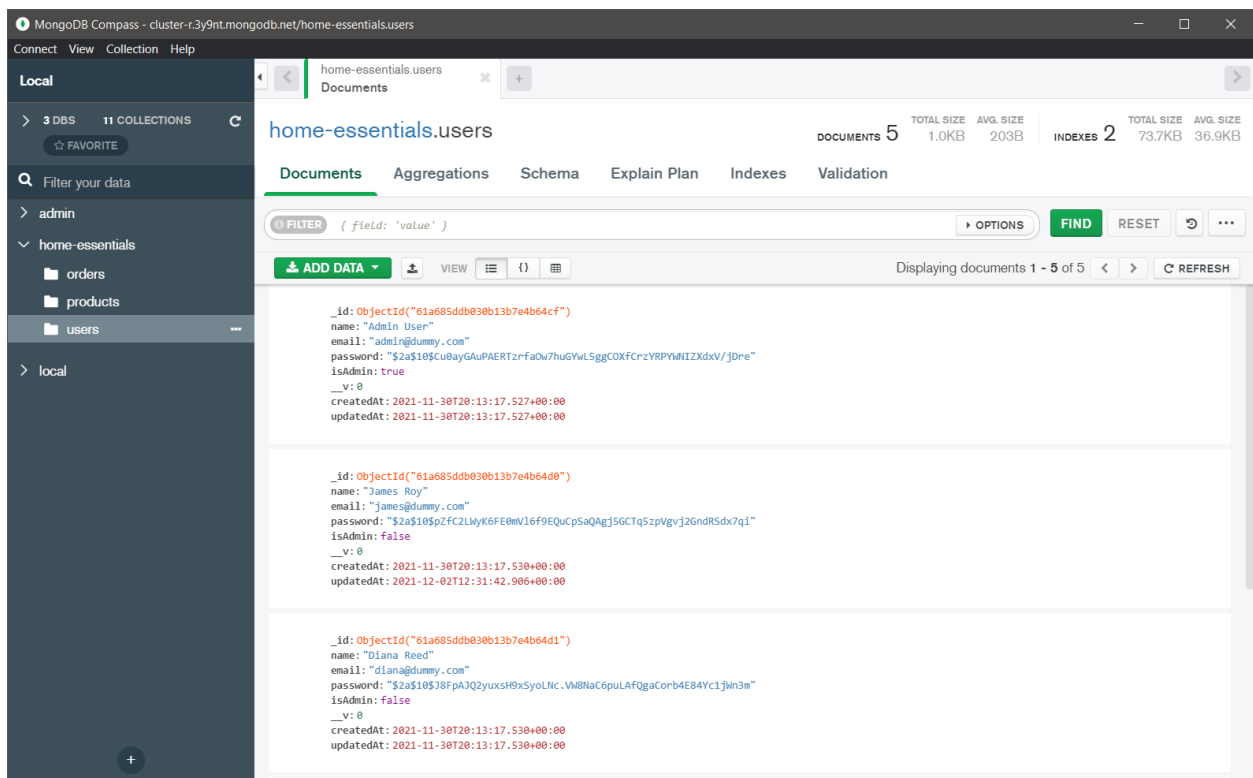


Figure 15
User Collection

4.3 API Testing

Fetching the product list

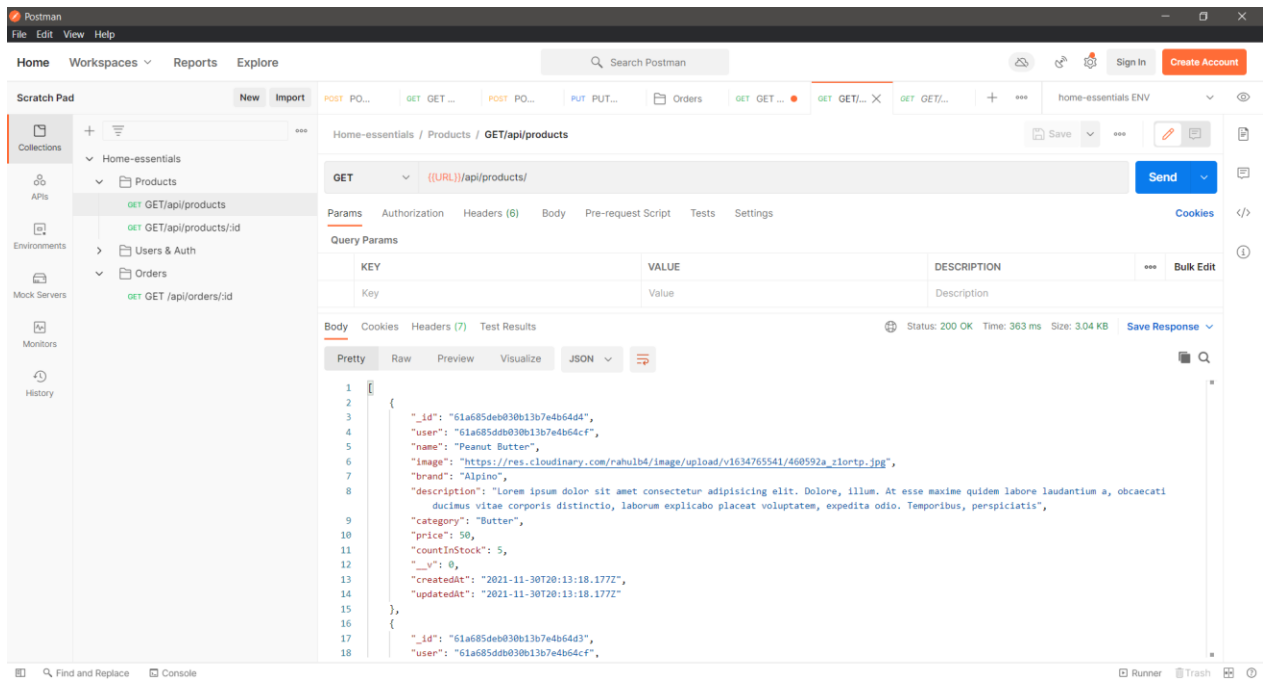


Figure 16 Product Route

Fetching a specific product with its product id

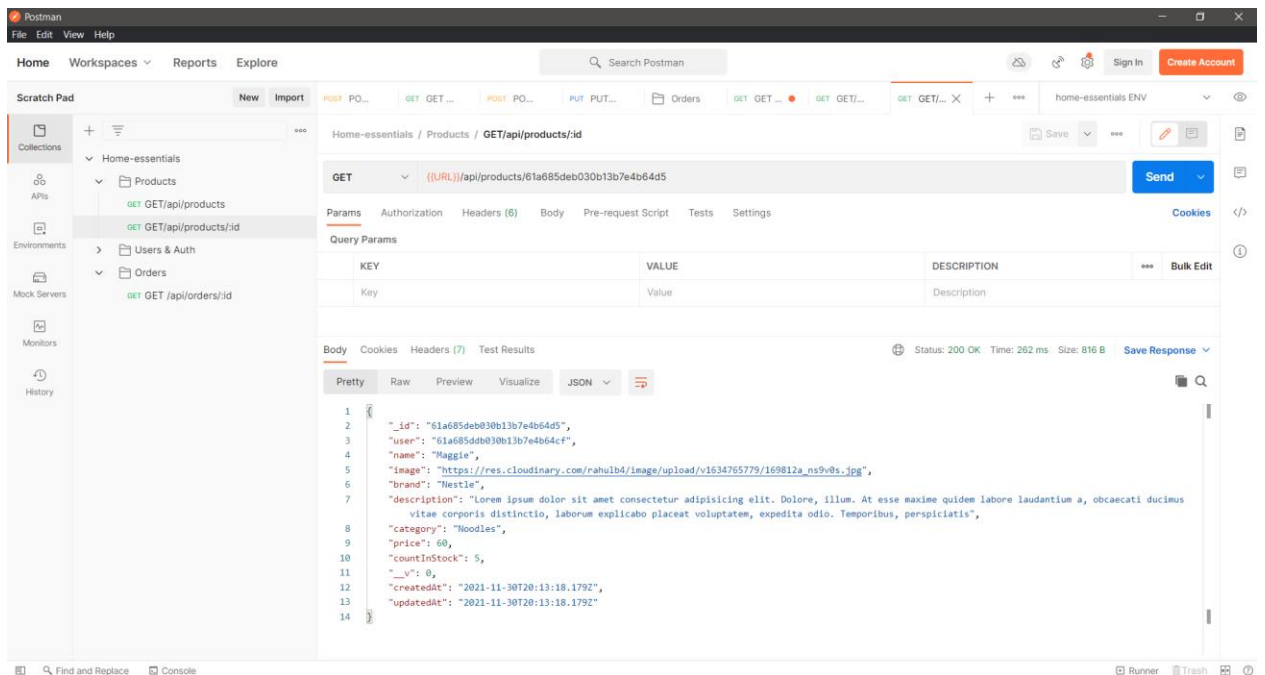


Figure 17 Product Route with id

4.4 Instructions:

Following are the instructions to run the project

In the terminal once all packages are installed run the below command to run the development server.

```
npm run dev
```



```
TERMINAL  PROBLEMS  OUTPUT  DEBUG CONSOLE

-> npm run dev

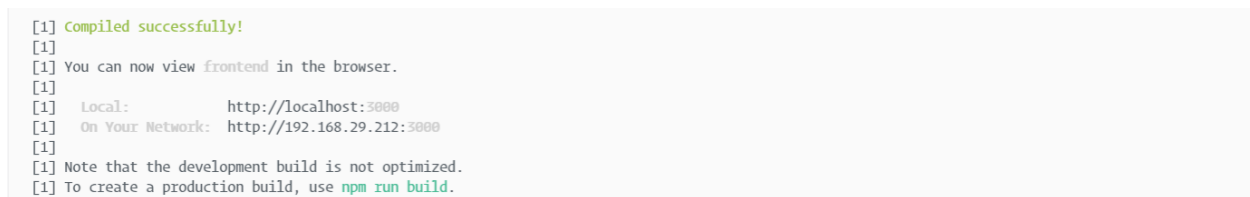
> home-essentials@1.0.0 dev
> concurrently "npm run server" "npm run client"

[0]
[0] > home-essentials@1.0.0 server
[0] > nodemon backend/server
[0]
[1]
[1] > home-essentials@1.0.0 client
[1] > npm start --prefix frontend
[1]
[1]
[1] > frontend@0.1.0 start
[1] > craco start
[1]
[0] [nodemon] 2.0.14
[0] [nodemon] to restart at any time, enter `rs`
[0] [nodemon] watching path(s): *.*
[0] [nodemon] watching extensions: js,mjs,json
[0] [nodemon] starting `node backend/server.js`
[0] Server running in development mode on port 5000
[0] MongoDB Connected: cluster-r-shard-00-02.3y9nt.mongodb.net
[1] i [wds]: Project is running at http://192.168.29.212/
[1] i [wds]: webpack output is served from
[1] i [wds]: Content not from webpack is served from D:\React\home-essentials\frontend\public
[1] i [wds]: 404s will fallback to /
[1] Starting the development server...
[1]
```

Figure 20 Running Server

Once the server is up and running and database is connected

Then open <http://localhost:3000/> to see the app.



```
[1] Compiled successfully!
[1]
[1] You can now view frontend in the browser.
[1]
[1] Local:          http://localhost:3000
[1] On Your Network: http://192.168.29.212:3000
[1]
[1] Note that the development build is not optimized.
[1] To create a production build, use npm run build.
```

Figure 21 Open App in Browser

CHAPTER 5

RESULT ANALYSIS

5.1 Testing Methods

The completion of a system will be achieved only after it has been thoroughly tested. Though this gives a feel the project is completed, there cannot be any project without going through this stage. Hence in this stage it is decided whether the project can undergo the real time environment execution without any break downs, therefore a package can be rejected even at this stage.

If testing is conducted successfully (according to the objectives stated previously) it will uncover errors in the software. As a secondary benefit, testing demonstrates that software functions appear to be working according to specification, that behavioral and performance requirements appear to have been met. In addition, data collected as testing is conducted provide a good indication of software reliability and some indication of software quality as a whole. But testing cannot show the absence of errors and defects, it can show only that software errors and defects are present. It is important to keep this (rather gloomy) statement in mind as testing is being conducted.

While deciding on the focus of testing activities, study project priorities. For example, for an online system, pay more attention to response time. Spend more time on the features used frequently. Decide on the effort required for testing based on the usage of the system. If the system is to be used by a large number of users, evaluate the impact on users due to a system failure before deciding on the effort.

This creates two problems

- Time delay between the cause and appearance of the problem.
- The effect of the system errors on files and records within the system.

The purpose of the system testing is to consider all the likely variations to which it will be suggested and push the systems to limits. The testing process focuses on the logical intervals of the software ensuring that all statements have been tested and on functional interval is conducting tests to uncover errors and ensure that defined input will produce actual results that agree with the required results. Program level testing, modules level testing integrated and carried out.

There are two major types of testing they are:

- White Box Testing.
- Black Box Testing.

5.1.1 White Box Testing

White box sometimes called “Glass box testing” is a test case design uses the control structure of the procedural design to drive test case. Using white box testing methods, the following tests were made on the system

- a) All independent paths within a module have been exercised once. In the system, ensuring that case was selected and executed checked all case structures. The bugs that were prevailing in some part of the code where fixed
- b) All logical decisions were checked for the truth and falsity of the values.

5.1.2 Black Box Testing

Black box testing focuses on the functional requirements of the software. The black box testing enables the software engineering to derive a set of input conditions that will fully exercise all functional requirements for a program. Black box testing is not an alternative to white box testing rather it is complementary approach that is likely to uncover a different class of errors that white box methods like.

- Interface errors.
- Performance in data structure.
- Performance errors.
- Initializing and termination errors
- Authentication Errors.

CHAPTER 6

CONCLUSION

6.1 Conclusion

This project is only a humble venture to satisfy the needs in a shop. Several user-friendly coding has also been adopted. This package shall prove to be a powerful package in satisfying all the requirements of the organization. The objective of software planning is to provide a frame work that enables the manger to make reasonable estimates made within a limited time frame at the beginning of the software project and should be updated regularly as the project progresses. This website provides a computerized version of grocery management system which will benefit the users as well as the visitor of the shop. It makes entire process online where users can buy various product. It also has a facility for common user by login into the system where user can login and can see status of ordered item as well request for items.

6.2 Future Work

The project has a very vast scope in future. The project can be implemented on intranet in future. Project can be updated in near future as and when requirement for the same arises, as it is very flexible in terms of expansion. With the proposed software the client is now able to manage and hence run the entire work in a much better, accurate and error free manner.

The project can be updated over the time the technologies in the current trends thus making the app more user accessible.

The following are the future scope for the project.

- Payment Method Integration using PayPal/Razor pay
- Can be added inventory management system

- A user specific profile section
- Feature to check delivery status
- And many features can be added this project to make it more robust.

References

1. Sabari Shankar R, Naresh K Kumar S,” *A Descriptive Analysis of Consumer Perception on Online Grocery Shopping*”, IJRSI, March 2018
2. <https://www.businesswire.com/news/home/20190729005605/en/Indian-Online-Grocery-Market-Outlook-2019-2023-Historic>
3. <https://reactjs.org/>
4. <https://tailwindcss.com/>
5. <https://docs.atlas.mongodb.com/>
6. <https://developer.mozilla.org/en-US/docs/Web>
7. <https://mongoosejs.com/docs/>
8. <https://learning.postman.com/docs/getting-started/introduction/>
9. <https://nodejs.dev/>
10. https://developer.mozilla.org/en-US/docs/Learn/Server-side/Express_Nodejs/Introduction