SHOPSMART

Project Title: shopsmart: your digital grocerystore experience

Team Id :LTVIP2025TMID59031

Team lead:Chekka Rochan

Team member 1:Bodapati Hima Varshini

Team member 2:Boddu Deepika

Team member 3:Bolla Rushwanth

College Name: Seshadri rao gudlavalleruengineering college

Rollno's: 23481A4219,23481A1231,23481A1232, 23481A1233

Introduction:

"Welcome to our Grocery Web App, your one-stopshopforall your grocery needs! With our user-friendly interface and wide selection of high-qualityproducts, we aim to make your grocery shoppingexperience convenient and enjoyable. Whether you're



looking for fresh produce, pantry staples, or householdessentials, our app has you covered. Exploreour virtualaisles, add items to your cart with ease, andhaveyourgroceries delivered right to your doorstep. Experiencethefuture of grocery shopping with our GroceryWebApptoday!"

Description:

Our Grocery Web App is more than just a convenientwayto shop for groceries—it's a comprehensivesolutiondesigned to enhance every aspect of your shoppingexperience. Imagine a virtual supermarket at yourfingertips, offering an extensive selection of high-qualityproducts that cater to your every need.

From fresh produce sourced directly fromlocal farmstopantry staples and household essentials, our appprovideseverything you need to keep your kitchen well-stocked. Ouruser-friendly interface ensures that you cannavigateourvirtual aisles effortlessly, easily finding what you'relookingfor and adding items to your cart with a simpleclick.

But the convenience doesn't stop there. Withour app, youcan schedule deliveries at your preferred time, ensuringthat your groceries arrive when it's most convenientforyou. Say goodbye to long lines and tediousgrocerytrips—our app brings the supermarket to you, savingyou



time and hassle.

We understand that everyone's dietary preferencesandrequirements are different, which is why our appoffersawide range of options, including organic, gluten-free, andvegan products. You can also discover newandexcitingproducts through our app, making grocery shoppinganadventure rather than a chore.

Our commitment to quality extends beyondour productsto our customer service. Our dedicated teamisalwaysavailable to assist you with any questions or concernsyoumay have, ensuring that your shopping experienceisnothing short of exceptional.

Experience the future of grocery shoppingwithourGrocery Web App. Download it today anddiscovertheconvenience of fresh, high-quality groceries deliveredrightto your doorstep.

Scenario Based Case Study:

Meet Priya, a busy professional with ahecticschedule who values convenience andefficiencyinher daily life. Priya loves to cook and prefersusingfresh ingredients in her meals. However, her tight



schedule often makes it challenging for her tofindthe time to visit grocery stores regularly.

Priya's Solution: The Grocery WebApp.

Priya discovers the Grocery Web App, aone-stopsolution for all her grocery needs. The appoffersawide selection of high-quality products, includingfresh produce, pantry staples, and householdessentials, all available at her fingertips.

User Registration and Authentication:Priyaregistersan account on the app, providing her basicdetailsand preferences. She logs in securely usinghercredentials, ensuring her privacy and security.

Product Listings:Priya browses throughtheapp'sextensive list of products, organized neatlyintocategories for easy navigation. She canquicklyfindwhat she's looking for and add items toher virtual cart with a simple tap.

Personalized Recommendations:The appusesPriya's purchase history and preferencestoprovidepersonalized recommendations, helpingher discovernew products and brands that align withher tastes.



Convenient Delivery Options:Priya canchoosetohave her groceries delivered to her doorstepat atime that suits her schedule. She can alsoselectexpress delivery for urgent needs.

Secure Payment Gateway:The app integrateswithasecure payment gateway, allowing Priyatopayforher groceries online using various payment methods,including credit/debit cards and digital wallets.

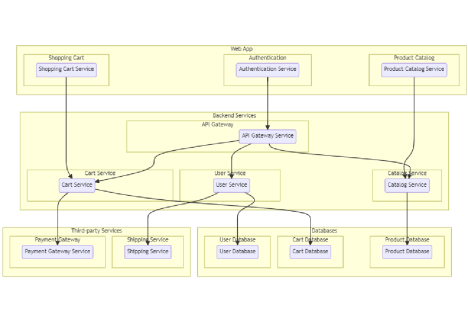
Order Tracking:Priya receives a confirmationofherorder along with a tracking link that allowshertomonitor the status of her delivery in real-time.

Customer Support:The app providesexcellentcustomer support, with a dedicatedteamavailableto assist Priya with any queries or concernsshemayhave.

Priya's Experience: Thanks to the GroceryWebApp,Priya can now enjoy the convenience of havingfresh,high-quality groceries delivered right toher doorstep,saving her time and effort. She can focusonwhatmatters most to her—cooking deliciousmealsforherself and her loved ones.



Technical Architecture:~~-~~

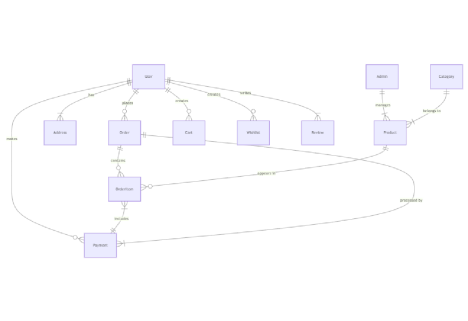


The technical architecture of an grocery-webappapptypically involves a client-server model, wherethefrontend represents the client andthebackendserves as the server. The frontend is responsibleforuser interface, interaction, and presentation, whilethe backend handles data storage, businesslogic,and integration with external services likepaymentgateways and databases. Communicationbetween



the frontend and backend is typicallyfacilitatedthrough APIs, enabling seamless dataexchangeandfunctionality.

ER~~-~~Diagram:



The technical architecture of an grocery-webappapptypically involves a client-server model, wherethefrontendrepresents the client and the backend serves astheserver.The frontend is responsible for user interface, interaction, and presentation, while the backend handles datastorage, business logic, and integration with external serviceslike



payment gateways and databases. Communicationbetween the frontend and backend is typicallyfacilitatedthrough APIs, enabling seamless data exchangeandfunctionality.

Key Features:

Product Catalog: Our grocery-webapp appprovidesanextensive product catalog with various categoriesandsubcategories. Users can easily search, browse, andfilterproducts based on their preferences, makingit effortlessto find the desired items.

Shopping Cart and Checkout: The appincludesashopping cart feature that enables users toaddproducts, review their cart, and proceed to checkout. Thecheckoutprocess offers multiple payment options, ensuringasmooth and secure transaction experience. Product Reviews and Ratings: Customers canprovidefeedback and rate products, helping other usersmakeinformed purchasing decisions. This featurefostersasense of community and trust among users. Order Tracking: Once an order is placed, userscantrackits status in real-time. They receive updatesonorderprocessing, shipping, and delivery, providingtransparencyand peace of mind.

Admin Dashboard: For administrators, our grocery-webapp app offers a comprehensive dashboardto



manage products, inventory, orders, andcustomerinformation. It provides insights into sales performance, stock levels, and customer analytics, enablingefficientbusiness operations.

Order Management: The app manages the order lifecycle, including order placement, tracking, and statusupdates.Users can view their order history, track shipments, andrequest returns or cancellations.

Search and Filtering: Users can search for productsusingkeywords and apply filters to narrowdownthesearchresults based on criteria such as price range, brand, orcustomer ratings.

Shopping Cart and Checkout: The appincludesashopping cart feature that enables users toaddproducts, review their cart, and proceed to checkout. Thecheckoutprocess offers multiple payment options, ensuringasmooth and secure transaction experience. Product Reviews and Ratings: Customers canprovidefeedback and rate products, helping other usersmakeinformed purchasing decisions. This featurefostersasense of community and trust among users. Order Tracking: Once an order is placed, userscantrackits status in real-time. They receive updatesonorderprocessing, shipping, and delivery, providingtransparencyand peace of mind.

Admin Dashboard: For administrators, our grocery-webapp app offers a comprehensive dashboardtomanage products, inventory, orders, andcustomerinformation. It provides insights into sales performance,



stock levels, and customer analytics, enablingefficientbusiness operations.

Order Management: The app manages the order lifecycle, including order placement, tracking, and statusupdates.Users can view their order history, track shipments, andrequest returns or cancellations.

Search and Filtering: Users can search for productsusingkeywords and apply filters to narrowdownthesearchresults based on criteria such as price range, brand, orcustomer ratings.

PRE REQUISITES:

To develop a full-stack Ecommerce App for FurnitureToolusing React js, Node.js,Express js and MongoDB, thereareseveral prerequisites you should consider. Herearethekey prerequisites for developing such an application:

Node.js and npm: Install Node.js, which includesnpm(Node Package Manager), on your development machine.Node.js is required to run JavaScript on the server side. • Download: https://nodejs.org/en/download/ • Installationinstructions:https://nodejs.org/en/download/package- manager/

MongoDB: Set up a MongoDB database tostorehoteland booking information. Install MongoDBlocallyor useacloud-based MongoDB service.



• Download:https://www.mongodb.com/try/download/community• Installationinstructions:https://docs.mongodb.com/manual/installation/

Express.js: Express.js is a web application frameworkforNode.js. Install Express.js to handle server-siderouting, middleware, and API development.

• Installation: Open your command prompt or terminal andrun the following

command: npm install express

React js: React is a JavaScript library for buildingclient-side applications.

And Creating Single Page Web-Appliaction

Getting Started

Create React App is an officially supportedwaytocreatesingle-page React applications. It offers a modernbuildsetup with no configuration.

Quik Start

npm create vite@latest

cd my-app

npm install

npm run dev



If you've previously installed create-react-appgloballyvia npm install -g create-react-app, we recommendyouuninstall the package using npmuninstall -gcreate-react-app or yarn global remove create-react-apptoensurethat npx always uses the latest version.

Create a new React project:

• Choose or create a directory where you want tosetupyour React project.

• Open your terminal or command prompt. • Navigate to the selected directory using the cdcommand.

• Create a new React project by runningthefollowingcommand: npx create-react-app your-app-name.Waitforthe project to be created:

• This command will generate the basic project structureand install the necessary dependencies

Navigate into the project directory:

• After the project creation is complete, navigateintotheproject directory by running the following command: cdyour~~-~~app~~-~~name

Start the development server:

• To launch the development server and seeyour Reactapp in the browser, run the following command: npmrundev

• The npm start will compile your app andstartthedevelopment server.



• Open your web browser and navigate to seeyour Reactapp.

You have successfully set up React on your machineandcreated a new React project. You can nowstart buildingyour app by modifying the generated project filesinthesrcdirectory.

Please note that these instructions provide abasicsetupfor React. You can explore more ad- vancedconfigurationsand features by referring to the official .

HTML, CSS, and JavaScript: Basic knowledgeof HTMLfor creating the structure of your app, CSSfor styling, andJavaScript for client-side interactivity is essential.

Database Connectivity: Use a MongoDBdriver oranObject-Document Mapping (ODM) library like Mongoosetoconnect your Node.js server with the MongoDBdatabaseand perform CRUD (Create, Read, Update, Delete)operations.

Front~~-~~end Library: Utilize React to build theuser-facingpart of the application, including products listings, bookingforms, and user interfaces for the admin dashboard.

Version Control: Use Git for version control, enablingcollaboration and tracking changes throughout thedevelopment process. Platforms like GitHubor Bitbucketcan host your repository.



• Git: Download and installation instructions canbefoundat:

Development Environment: Choose a codeeditor orIntegrated Development Environment (IDE) that suitsyourpreferences, such as Visual Studio Code, SublimeText, orWebStorm.

Roles and Responsibility

User:~~-~~

● Registration and Authentication: Usersare

responsible for creating an account ontheplatform and securely logging in toaccessitsfeatures.

● Browsing and Shopping: Users canbrowseproducts, add themto their cart, andproceedtocheckout for purchasing.

● Payment: Users are responsible for makingpayments for their orders using theavailablepayment methods.



● Order Management: Users can viewtheir orderhistory, track their deliveries, andmanagetheiraccount details.

● Feedback and Reviews: Users canprovidefeedback on products and servicesandleavereviews to help other users make informeddecisions.

● Compliance: Users are expected toadheretotheplatform's terms and conditions andprivacypolicy.

Admin:~~-~~

● User Management: Admins canmanageuseraccounts, including creating, updating, anddeleting accounts as necessary.

● Product Management: Admins areresponsiblefor managing the platform's product listings,including adding newproducts, updatingexisting ones, and removing outdatedproducts.

● Order Management: Admins canviewandmanage all orders placed on theplatform,including processing payments, trackingdeliveries, and handling returns or refunds.

● Content Management: Admins canmanagetheplatform's content, including creatingand



updating informational pages, blogposts, andother content.

● Analytics and Reporting: Admins cangeneratereports and analyze data to gain insightsintotheplatform's performance and user behavior.

● Compliance and Security: Adminsareresponsible for ensuring that theplatformcomplies with relevant laws and regulationsandthat user data is kept secure.

● Customer Support: Admins can providesupportto users, including respondingtoinquiries,resolving issues, and handling complaints.

● Marketing and Promotion: Adminscancreateand manage marketing campaignsandpromotions to attract and retain users.

Admin & User Flow:





The project flow for a grocery-web appinvolvesuseractions such as browsing products, addingitemstothe cart, proceeding to checkout, providingshippingdetails, selecting payment methods, makingpayments, and receiving order confirmation. Adminactions include managing products, viewingandprocessing orders, managing customers, andupdating product details.



PROJECT ~~FLOW:-~~ Before starting to work on this project Milestone 1: Project Setup and Configuration:1. Install required tools and software: ● Node.js.

● MongoDB.

● Create-react-app.

2. Create project folders and files: ● Client folders.

● Server folders.

3. Install Packages:

Frontend npm Packages ● Axios.

● React-Router –dom.

● Bootstrap.



● React-Bootstrap.

● React-icons.

Backend npm Packages ● Express.

● Mongoose.

● Cors.

Milestone 2: Backend Development:

● Setup express server

1. Create index.js file in the server (backendfolder).

2. Create a .env file and define port numbertoaccess it globally.

3. Configure the server by addingcors, body-parser.

● User Authentication:

• Create routes and middleware for user registration, login, and logout.

• Set up authentication middleware toprotect routes that require user authentication.



● Define API Routes:

• Create separate route files for different API functionalities such as users orders, andauthentication.

• Define the necessary routes for listingproducts, handling user registrationand

login,managing orders, etc.

• Implement route handlers using Express.jstohandle requests and interact with thedatabase.

● Implement Data Models: • Define Mongoose schemas for the different data entities like products, users, and

orders.

• Create corresponding Mongoose modelstointeract with the MongoDB database. • Implement CRUD operations (Create, Read, Update, Delete) for each model to performdatabase operations.

● User Authentication:

• Create routes and middleware for user registration, login, and logout.

• Set up authentication middleware toprotect routes that require user authentication.



● Error Handling:

• Implement error handling middleware tocatchand handle any errors that occur duringthe

API requests.

• Return appropriate error responses withrelevant error messages and HTTPstatuscodes.

Milestone 3: Database:

1. Configure MongoDB:

● Install Mongoose.

● Create database connection.

● Create Schemas & Models. 2. Connect database to backend: Now, make sure the database is connectedbefore performing any of the actions throughthebackend.

3. Configure Schema:

Firstly, configure the SchemasforMongoDB database, to store thedatainsuch



pattern. Use the data fromthe ERdiagramstocreate the schemas.

The schemas are looks like for theApplication.const mongoose = require('mongoose');

const userSchema = new mongoose.Schema({

firstname: { type: String},

lastname: { type: String },

username: { type: String, unique: true },

email: { type: String},

password: { type: String }

});

// category schema

const categorySchema = new mongoose.Schema({

category: { type: String, required: true, unique: true, },

description: { type: String, }

});

const productSchema = new mongoose.Schema({

productname: { type: String, required: true },

description: { type: String, required: true },

price: { type: Number, required: true },

image: { type: String, required: true },

category: { type: String, ref: 'Category', required: true },

countInStock: { type: Number, required: true, min: 0 },

rating: { type: Number, required: true },

dateCreated: { type: Date, default: Date.now }

});

const addToCartSchema = new mongoose.Schema({

userId: { type: String, required: true },

productId: { type: String, required: true },

quantity: { type: Number, minimum: 1, required: true, default: 1 }, });

const orderSchema = new mongoose.Schema({

firstname: { type: String, required: true },

lastname: { type: String, required: true },

user: { type: String, ref: 'User', required: true },

phone: { type: String, required: true },



productId: { type: String, required: true },

productName: { type: String, required: true },

quantity: { type: String, default: 1 },

price: { type: String, required: true },

status: { type: String, enum: ['Pending', 'Confirmed', 'Shipped', 'Delivered', 'Canceled',], default: 'Pending' },

paymentMethod: { type: String, required: true },

address: { type: String, required: true },

createdAt: { type: Date, default: Date.now }

});

const models = {

Users: mongoose.model('User', userSchema),

Category: mongoose.model('Category', categorySchema),

Product: mongoose.model('Product', productSchema),

AddToCart: mongoose.model('AddToCart', addToCartSchema),

Order: mongoose.model('Order', orderSchema),

};

module.exports = models;

Milestone 4: Frontend Development:

1. Setup React Application: • Create React application.

• Configure Routing.

• Install required libraries. 2. Design UI components:



• Create Components.

• Implement layout and styling. • Add navigation.

3. Implement frontend logic: • Integration with API endpoints. • Implement data binding.

Milestone 5: Project Implementation:

Finally, after finishing coding the projectswerunthewhole project to test it’s working processandlookfor bugs. Now, let’s have a final look at theworkingof our Darshan Ease.

Landing page:-



Login Page:-

Items Page:-



My Cart:-

My Orders Page:-



My History Page:-



Place Order Page:-

Admin Dashboard Page:





Users Page:-

Add Product page:-



Admin Orders Page:-



The demo of the app is available at:-

https://drive.google.com/file/d/1HLowcIqs2d8lxTprS2jqPmR4AO nUW8xD/view?usp=drive\_link

\*\*\*\*END\*\*\*\*

