BlinkIt Clone

Submitted to the

Department of Master of Computer Applications in partial fulfilment of the requirements for the Mini Project in

Full Stack Development – MCA26

by

Dattatray Nalawade

1MS24MC024

Under the Guidance of

Komal S K

Department of Master of Computer Applications

RAMAIAH INSTITUTE OF TECHNOLOGY

(Autonomous Institute, Affiliated to VTU)

Accredited by National Board of Accreditation & NAAC with 'A+' Grade

MSR Nagar, MSRIT Post, Bangalore-560054

www.msrit.edu

2025

RAMAIAH
Institute of Technology

DEPARTMENT OF MASTER OF COMPUTER APPLICATIONS

CERTIFICATE

This is to certify that the project entitled **"BlinkIt Clone"** is carried out by **Dattatray Nalawade - 1MS24MC024** student of 2nd semester, in partial fulfillment for the Mini Project in Full Stack development – MCA26, during the academic year 2024-2025.

Komal S K
Guide

Dr. S Ajitha
Head of theDepartment

Name of Examiners **Signature with Date**

1.

2.

ACKNOWLEDGEMENT

With deep sense of gratitude, I am thankful to our Respected HoD, **Dr. S Ajitha** for her pillared support and encouragement. I would like to convey my heartfelt thanks to my Project GuideMr **Komal S K**, Department of Master of Computer Applications, for giving me an opportunity to embark upon this topic and for her/his constant encouragement.

It gives me great pleasure to acknowledge the contribution of all people who helped me directly or indirectly realize it. First I would like to acknowledge the love and tremendous sacrifices that my Parents made to ensure that I always get an excellent education. For this and much more, I am forever in their debt.

I am thankful to all my friends for their moral and material support throughout the course of my project. Finally, I would like to thank all the Teaching and Non-Teaching Staff of Department of Master of Computer Applications for their assistance during various stages of my project work.

Dattatray Nalawade

ABSTRACT

This project presents a comprehensive full-stack e-commerce application that replicates the functionality and user experience of Blinkit, a popular instant grocery delivery platform. The application is designed to provide users with a seamless online shopping experience, enabling them to browse products, manage their shopping carts, and complete transactions efficiently. The project demonstrates the implementation of modern web development practices and serves as a practical application of full-stack development skills using contemporary technologies and frameworks.

The application is built using the MERN stack architecture, comprising MongoDB for database management, Express.js for server-side development, React.js for frontend user interface, and Node.js for backend runtime environment. The technical implementation incorporates advanced authentication mechanisms using JSON Web Tokens (JWT) with both access and refresh tokens to ensure secure user sessions. Additionally, the system integrates essential features such as password recovery functionality, OTP-based email verification for user registration, and robust security protocols to protect user data and maintain system integrity.

The core functionalities of the application include a comprehensive product management system with upload capabilities, an administrative panel for system management, and dynamic category and subcategory organization. Users can experience features such as real-time product browsing, advanced search and filtering options, shopping cart management, and secure checkout processes. The application also incorporates payment gateway integration, order tracking capabilities, and user profile management to provide a complete e-commerce solution that mirrors the functionality of modern instant delivery platforms.

The development approach emphasizes modularity, scalability, and user experience optimization. The project implements responsive design principles to ensure compatibility across various devices and screen sizes, while maintaining performance efficiency for handling multiple concurrent users. The backend architecture is designed to support high-volume transactions and real-time data processing, incorporating best practices for API development, database optimization, and security implementation. The frontend utilizes modern React.js features and state management techniques to deliver an intuitive and interactive user interface.

This BlinkIt clone project represents a significant achievement in full-stack web development, demonstrating proficiency in modern web technologies and e-commerce application development. The implementation showcases the ability to create scalable, secure, and user-friendly web applications that can compete with industry-standard platforms. The project serves as both a learning experience and a practical demonstration of skills in database design, API development, user authentication, payment processing, and responsive web design, making it a valuable addition to any developer's portfolio and a solid foundation for future e-commerce ventures.

Table of Contents

1. Introduction

The rapid evolution of e-commerce has transformed the way consumers access goods and services, particularly in the realm of instant grocery delivery. This project focuses on developing a full-stack clone of Blinkit, a prominent platform known for its quick delivery of groceries and essentials. By replicating its core features, the application aims to demonstrate practical implementation of modern web technologies while addressing real-world challenges in online retail. Built using the MERN stack (MongoDB, Express.js, React.js, and Node.js), the project integrates secure authentication, real-time product management, and user-centric functionalities to create an efficient e-commerce ecosystem. This introduction sets the stage for understanding the project's scope, technical architecture, and its significance in the context of full-stack development for MCA coursework.

In today's digital landscape, platforms like Blinkit have set high standards for speed, reliability, and user experience in grocery delivery. The motivation behind this clone is to explore and implement key aspects of such systems, including database management for inventory, API development for seamless interactions, and frontend design for intuitive navigation. The project not only serves as a learning tool but also highlights the integration of security measures like JWT authentication and OTP verification. Through this endeavor, the application bridges theoretical knowledge with practical application, showcasing how full-stack development can solve complex problems in e-commerce.

1.1. Problem Definition

The primary challenge in modern e-commerce, especially for instant delivery services, lies in creating a scalable system that handles high-volume transactions while ensuring security and user satisfaction. Traditional shopping methods are time-consuming and limited by physical constraints, leading to inefficiencies in product availability and delivery speed. This project addresses the problem of building a comprehensive platform that mimics Blinkit's functionality, including real-time inventory updates, secure payment processing, and efficient order management, without relying on proprietary systems.

A key issue is managing user authentication and data privacy in an environment prone to cyber threats, where weak security can lead to breaches and loss of trust. The application tackles this

by implementing JWT with access and refresh tokens, alongside password recovery and email verification via OTP. Additionally, the problem extends to administrative oversight, such as product uploads and category management, which must be intuitive yet robust to prevent errors in a dynamic inventory system.

Another dimension of the problem is optimizing performance for diverse user devices, ensuring responsiveness and quick load times amid varying network conditions. The project defines the need for a modular architecture that supports future scalability, like adding more features without disrupting core operations. By solving these interconnected issues, the Blinkit clone provides a practical solution that enhances accessibility to groceries while demonstrating full-stack proficiency in addressing e-commerce pain points.

2. Implementation

The implementation of the BlinkIt Clone project follows a structured full-stack development approach using the MERN stack to create a robust e-commerce platform for instant grocery delivery. The backend is developed with Node.js and Express.js, handling server-side logic, API endpoints, and database interactions through MongoDB for storing user data, products, orders, and categories. Key features include secure user authentication implemented via JSON Web Tokens (JWT) with access and refresh tokens, ensuring session management and protection against unauthorized access. The frontend, built with React.js, utilizes components for dynamic rendering of product listings, shopping carts, and user profiles, integrated with state management tools like Redux for efficient data flow. Deployment considerations include cloud services for hosting, with API routes optimized for performance to handle real-time updates and high traffic, demonstrating a scalable architecture that mirrors industry standards in e-commerce applications.

Database schema design plays a crucial role in the implementation, with MongoDB collections defined for users, products, orders, and admins, incorporating relationships for categories and subcategories. Security measures are embedded throughout, such as bcrypt for password hashing and OTP generation for email verification during registration and password recovery. The application implements RESTful APIs for CRUD operations on products, including file uploads for images using middleware like Multer. On the frontend, responsive design is achieved through CSS frameworks like Tailwind CSS, ensuring compatibility across devices, while React Router manages navigation between pages like home, product details, cart, and admin dashboard. Integration of payment gateways simulates real transactions, with error handling and validation to enhance reliability.

The overall implementation emphasizes modularity, with separate folders for backend routes, models, controllers, and frontend components, facilitating easy maintenance and scalability. Testing involves unit tests for APIs using tools like Jest and frontend validation with React Testing Library to ensure functionality. The project incorporates best practices such as environment variables for sensitive data and CORS configuration for secure cross-origin requests. This comprehensive approach results in a functional clone that provides users with seamless browsing, adding items to cart, checkout processes, and admin controls for managing inventory, all while maintaining performance and security in a full-stack environment.

2.1. Code snippets

To illustrate the backend authentication implementation, consider the user registration endpoint in Express.js, which handles OTP generation and email verification. The code uses Nodemailer for sending emails and bcrypt for secure password storage, ensuring data integrity during signup.

Server

ConnectDB.js

```
import mongoose from "mongoose";
import dotenv from 'dotenv'
dotenv.config()

if(!process.env.MONGODB_URI){
throw new Error(
"Please provide MONGODB_URI in the .env file"
)
}

async function connectDB(){
try {
await mongoose.connect(process.env.MONGODB_URI)
console.log("connect DB")
} catch (error) {
console.log("Mongodb connect error",error)
process.exit(1)
}
}

export default connectDB
```

Admin.js

```
import UserModel from "../models/user.model.js"

export const admin = async(request,response,next)=>{
   try {
      const  userId = request.userId

      const user = await UserModel.findById(userId)

      if(user.role !== 'ADMIN'){
          return response.status(400).json({
             message : "Permission denial",
             error : true,
             success : false
          })
      }

      next()

   } catch (error) {
       return response.status(500).json({
          message : "Permission denial",
          error : true,
          success : false
       })
   }
}
```

Index.js

```
import express from 'express'
import cors from 'cors'
import dotenv from 'dotenv'
dotenv.config()
import cookieParser from 'cookie-parser'
```

```
import morgan from 'morgan'
import helmet from 'helmet'
import connectDB from './config/connectDB.js'
import userRouter from './route/user.route.js'
import categoryRouter from './route/category.route.js'
import uploadRouter from './route/upload.router.js'
import subCategoryRouter from './route/subCategory.route.js'
import productRouter from './route/product.route.js'
import cartRouter from './route/cart.route.js'
import addressRouter from './route/address.route.js'
import orderRouter from './route/order.route.js'

const app = express()
app.use(cors({
    credentials : true,
    origin : process.env.FRONTEND_URL
}))
app.use(express.json())
app.use(cookieParser())
app.use(morgan())
app.use(helmet({
    crossOriginResourcePolicy : false
}))

const PORT = 8080 || process.env.PORT

app.get("/",(request,response)=>{
    ///server to client
    response.json({
        message : "Server is running " + PORT
    })
})

app.use('/api/user',userRouter)
```

```
app.use("/api/category",categoryRouter)
app.use("/api/file",uploadRouter)
app.use("/api/subcategory",subCategoryRouter)
app.use("/api/product",productRouter)
app.use("/api/cart",cartRouter)
app.use("/api/address",addressRouter)
app.use('/api/order',orderRouter)

connectDB().then(()=>{
    app.listen(PORT,()=>{
        console.log("Server is running",PORT)
    })
})
```

Client

Index.css

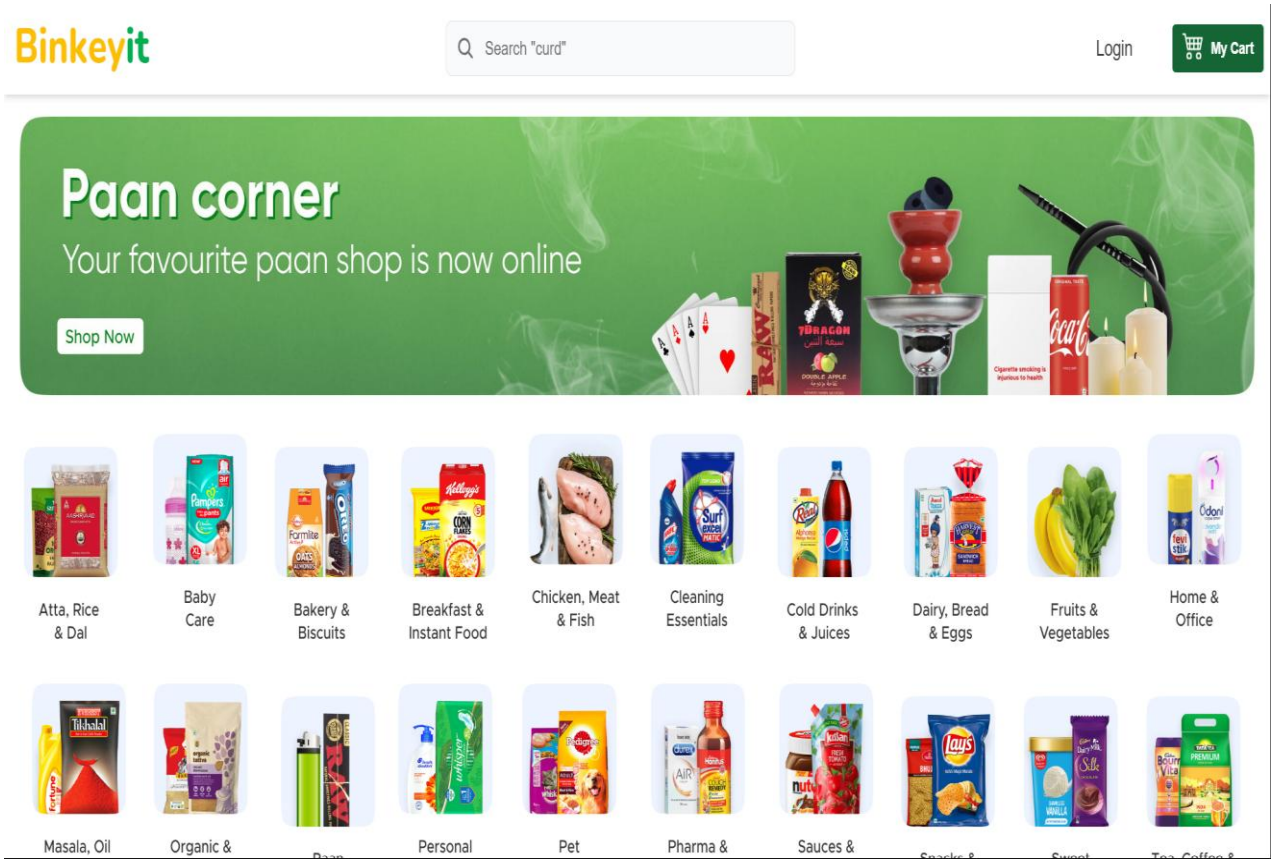
```
@tailwind base;
@tailwind components;
@tailwind utilities;

@import
url('https://fonts.googleapis.com/css2?family=Poppins:ital,wght@0,100;0,200;0,300;0,400;0,
500;0,600;0,700;0,800;0,900;1,100;1,200;1,300;1,400;1,500;1,600;1,700;1,800;1,900&displa
y=swap');

html{
    scroll-behavior: smooth;
}

body{
    font-family: "Poppins", sans-serif;
    font-weight: 100;
    font-style: normal;
```

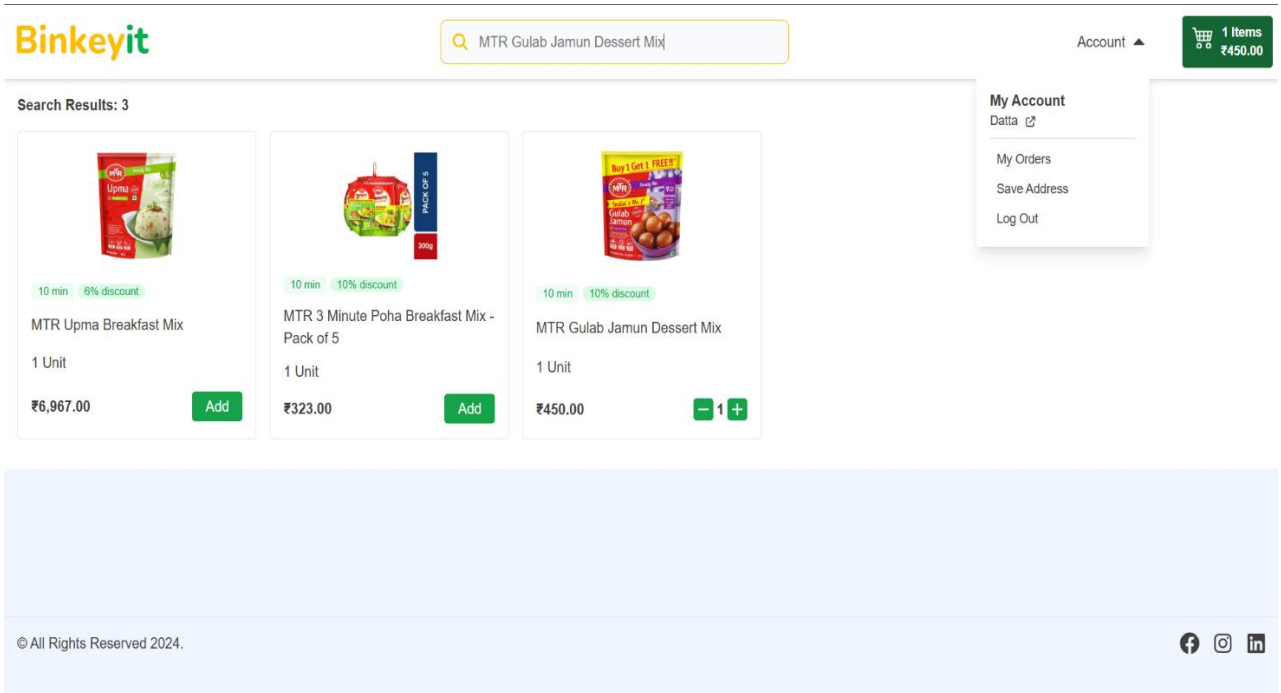
```
    @apply bg-blue-50 text-neutral-700 scroll-smooth
}

.scrollbar-none::-webkit-scrollbar{
    display: none;
}

.scrollbarCustom::-webkit-scrollbar{
    width: 10px;
    @apply bg-slate-100
}
.scrollbarCustom::-webkit-scrollbar-thumb{
    @apply bg-blue-100 rounded-full
}
```

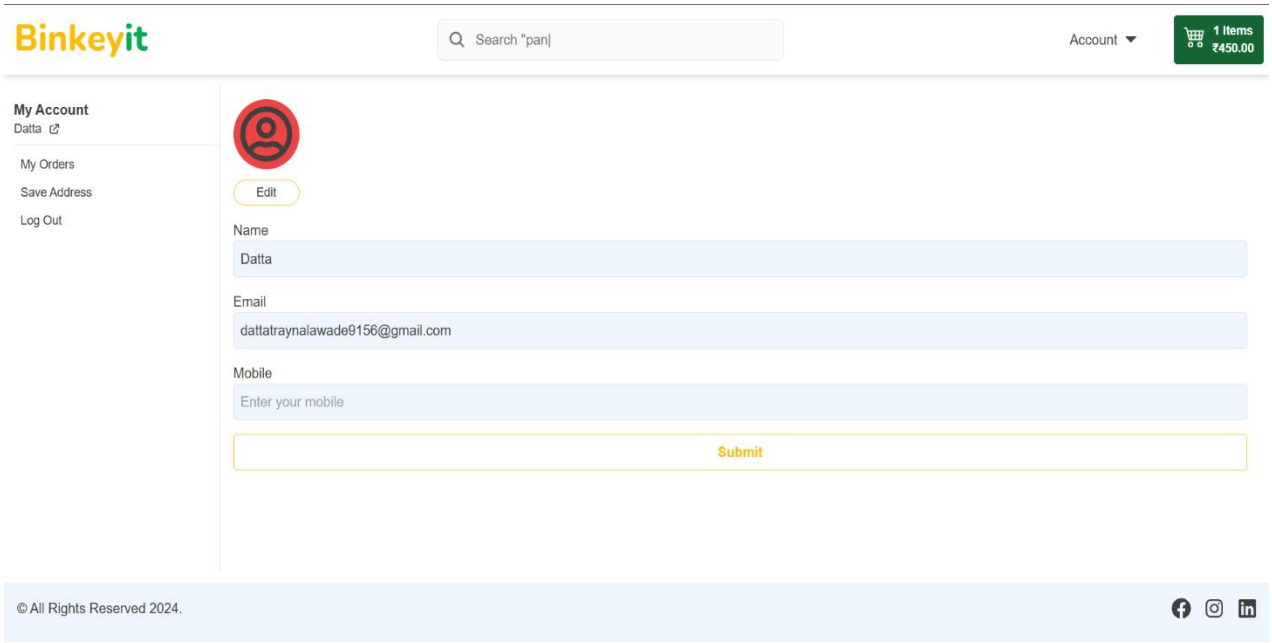
CartProduct.js

```
import { createSlice } from "@reduxjs/toolkit";

const initialState = {
    cart : []
}

const cartSlice = createSlice({
    name : "cartItem",
    initialState : initialState,
    reducers : {
        handleAddItemCart : (state,action)=>{
            state.cart = [...action.payload]
        },
    }
})

export const { handleAddItemCart } = cartSlice.actions
export default cartSlice.reducer
```

3. User Interface Screenshots






Binkeyit
Search "pan|
Account
1 Items
₹450.00
My Account
Datta
My Orders
Save Address
Log Out
Edit
Name
Datta
Email
dattatraynalawade9156@gmail.com
Mobile
Enter your mobile
Submit
© All Rights Reserved 2024.


Get ready for
Raksha Bandhan
Explore a wide
range of Rakhis
Shop now

4. Conclusion and Future Enhancement

Conclusion

The BlinkIt Clone project successfully demonstrates the development of a full-stack e-commerce application that emulates the core functionalities of a leading instant grocery delivery platform. By leveraging the MERN stack—MongoDB for efficient data storage, Express.js and Node.js for robust backend operations, and React.js for an interactive frontend—the application provides users with a seamless experience in browsing products, managing carts, and completing secure transactions. Key implementations include JWT-based authentication with access and refresh tokens, OTP verification for user registration and password recovery, and an administrative dashboard for product uploads, category management, and order oversight. This project not only addresses the demands of modern online retail but also showcases proficiency in integrating payment gateways, real-time inventory updates, and responsive design principles to ensure accessibility across devices. Through modular code structure and best practices in API development, the application achieves scalability and maintainability, making it a practical example of full-stack engineering in the e-commerce domain.

Overall, the BlinkIt Clone serves as a comprehensive learning endeavor, bridging theoretical concepts with real-world application in web development. It highlights the importance of security protocols to protect user data, performance optimizations for handling high traffic, and user-centric features that enhance shopping convenience. The successful integration of features like dynamic search filters, shopping cart persistence, and order tracking underscores the project's alignment with industry standards, while demonstrating the developer's ability to create a functional, secure, and efficient platform. This mini-project fulfills the requirements of full-stack development coursework, providing a solid foundation for understanding e-commerce systems and paving the way for more advanced implementations in future endeavors.

Future Enhancement

While the current version of the BlinkIt Clone delivers essential e-commerce functionalities, several enhancements could expand its capabilities and user appeal:

1. Mobile Responsiveness and App Development: Implement full mobile optimization using frameworks like React Native to create a dedicated mobile application, enabling on-the-go shopping with push notifications for order updates and promotions.

2. Advanced Search and Recommendation Engine: Integrate machine learning algorithms to provide personalized product recommendations based on user browsing history and purchase patterns, along with enhanced search capabilities using Elasticsearch for faster, more accurate results.

3. Multi-Vendor Support and Marketplace Features: Evolve the platform into a multi-vendor marketplace by allowing third-party sellers to register, manage their inventories, and process orders, with added commission tracking and dispute resolution tools.

4. Integration of Additional Payment Options and Logistics: Expand payment gateways to include digital wallets, cryptocurrencies, and buy-now-pay-later services, while incorporating real-time logistics APIs for accurate delivery tracking and estimated time arrivals.

5. Analytics Dashboard and User Insights: Add an advanced analytics module for administrators to monitor sales trends, user behavior, and inventory levels, with reporting tools and visualizations using libraries like Chart.js, to support data-driven decision-making.

6. Enhanced Security and Compliance Features: Incorporate two-factor authentication, GDPR compliance for data privacy, and AI-based fraud detection to further strengthen security, ensuring trust in high-volume transactions.

7. Social Features and Community Building: Introduce user reviews, ratings, social sharing options, and loyalty programs to foster community engagement, encouraging repeat visits and word-of-mouth promotion.

These enhancements would transform the BlinkIt Clone into a more versatile and competitive e-commerce solution, suitable for real-world deployment and scalable growth in the instant delivery sector.