BLOOMING BLISS FLOWER BOUQUETS

A PROJECT REPORT

Submitted by

DHARSHINI.S (920422205026)

MADHUBALA.K(920422205055)

SUDHAA SHREE.K (920422205109)

In partial fulfillment for the award of the degree of

BACHELOR OF TECHNOLOGY

IN

INFORMATIOTECNOLOGY

DEPARTMENT OF INFORMATION TECHNOLOGY

KAMARAJ COLLEGE OF ENGINEERING AND TECHNOLOGY

(An Autonomous Institution - Affiliated to Anna University, Chennai)

K.VELLAKULAM, VIRUDHUNAGAR - 625 701

NOVEMBER 2024

KAMARAJ COLLEGE OF ENGINEERING AND TECHNOLOGY

(An Autonomous Institution- Affiliated to Anna University, Chennai)

K.VELLAKULAM, VIRUDHUNAGAR - 625 701

BONAFIDE CERTIFICATE

Certified that the project report "**BLOOMING BLISS FLOWER BOUQUETS**" is the bonafide work of **"Dharshini.S(920422205026), MadhuBala.K(920422205055), SudhaaShree.K(920422205109)"** who carried out the project work under my supervision.

SIGNATURE	**SIGNATURE**
Dr. E. VAKAIMALAR	Dr. R. ARTHY
Head of the Department,	SUPERVISOR,
Associate Professor,	Assistant Professor,
Dept. of Information Technology,	Dept. of Information Technology,
Kamaraj College of Engg & Tech, K.Vellakulam,	Kamaraj College of Engg &Tech, K.Vellakulam,
Virudhunagar - 625 701.	Virudhunagar - 625701.

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ABSTRACT

Blooming Bliss Flower Bouquets is an online flower shop that offers a wide variety of fresh blooms, bouquets, and floral arrangements tailored for any occasion. With a user-friendly, ReactJS-powered interface, customers can easily browse and shop from a curated selection of flowers. The platform provides customization options to personalize bouquets, ensuring each arrangement is unique. Fast, same-day delivery guarantees freshness, while secure checkout options allow for a hassle-free shopping experience. Whether it's for a special event, a gift, or simply to brighten your space, Blooming Bliss helps you express emotions beautifully through flowers.

ACKNOWLEDGEMENT

I would like to thank to Dr. E. Vakaimalar, Head of the Department of Information Technology, for their encouragement and valuable insights throughout the development of the "BLOOMING BLISS FLOWER BOUQUETS" project.

Special thank to Dr. R. Arthy, my supervisor, whose guidance and expertise were instrumental in shaping the direction of this project. Their support helped me gain in-depth the knowledge of the MERN stack and its application. This project has been an invaluable learning experience, enhancing both my technical and problem-solving skills.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 HTML (Hyper Text Markup Language)

HTML is the **standard markup language** used to create and structure content on web pages. It forms the **foundational framework** for websites, defining elements that browsers interpret to display web content.

- **Purpose**: Defines content and structure of a webpage using various elements like headings, paragraphs, images, links, and lists.
- **Elements**:
  - **Headings** (<h1> - <h6>): Define the hierarchy of headings on a page.
  - **Paragraphs** (<p>): Block of text.
  - **Images** (<img>): Embeds images.
  - **Links** (<a>): Creates clickable links to other pages.
  - **Lists** (<ul>, <ol>): Organizes content in bullet points or numbered lists.
- **Attributes**: HTML elements can have attributes (like id, class, src, href) that add additional information, such as linking an image or assigning identifiers for styling.
- **Document Structure**:
  - **DOCTYPE Declaration** (<!DOCTYPE html>): Ensures correct HTML rendering.
  - **<html> Element**: The root of the HTML document.
  - **<head> Element**: Contains meta-information (title, character set) and external resources (stylesheets, scripts).
  - **<body> Element**: Contains the visible content of the webpage.

HTML acts as the **skeleton** of web pages, providing the basic structure before CSS (for styling) and JavaScript (for interactivity) are applied.

1.2 CSS (Cascading Style Sheets)

CSS is the **styling language** used to control the visual presentation of HTML elements on web pages. It determines the layout, colors, fonts, and overall aesthetic of a website.

- ❖ **Purpose**: While HTML defines the structure of a webpage, CSS enhances its appearance by applying styles to HTML elements, making the website more visually appealing.
- ❖ **Basic Concepts**:
  - **Selectors**: CSS uses selectors to target specific HTML elements for styling.
    - **Element Selector**: Targets all elements of a specific type (e.g., h1 { color: blue; } to make all <h1> headings blue).
    - **Class Selector**: Targets elements with a specific class (e.g., .header { font-size: 20px; } to style elements with class header).
    - **ID Selector**: Targets an element by its unique ID (e.g., #menu { background-color: grey; } to style an element with the id="menu").
  - **Properties and Values**: CSS applies styles using properties (e.g., color, font-size, margin) and their values.
    - Example: color: red; changes the text color to red, and margin: 20px; adds space around an element.
- ❖ **Layout**:
  - **Box Model**: Describes the space an element occupies, including:
    - **Content**: The actual content inside the element.
    - **Padding**: The space between the content and the element's border.
    - **Border**: The line surrounding the padding (optional).

        - **Margin**: The space outside the border.

    - **Grid**: A 2-dimensional system for creating complex, responsive layouts using rows and columns.

- **Responsiveness**: CSS uses **media queries** to make web pages responsive, meaning they adapt their layout to fit different screen sizes, such as mobile, tablet, or desktop.

1.3 JavaScript

JavaScript (JS) is a **high-level, dynamic programming language** used to add interactivity and dynamic behavior to websites.

- **Purpose**: JavaScript enables **interactivity** on websites by responding to user input, allowing for real-time updates, animations, and dynamic content changes without needing to reload the page.
- **Key Features**:
    - **DOM Manipulation**: JavaScript can interact with the **Document Object Model** (DOM), which represents the structure of HTML documents. This allows developers to dynamically change content, styles, or structure.
        - Example: Changing the text of a button or showing/hiding elements based on user interaction.
    - **Event Handling**: JavaScript enables websites to respond to various **events**, such as clicks, form submissions, scrolling, and keyboard input.
        - Example: When a user clicks a "Submit" button, JavaScript can validate form data before sending it to the server.
    - **AJAX (Asynchronous JavaScript and XML)**: AJAX allows websites to update data **asynchronously** without reloading the entire page, providing faster and more interactive experiences.

        - Example: Loading new content when scrolling down a page (infinite scrolling) or updating a list of tasks without reloading the page.

    - **API Interaction**: JavaScript can **fetch data** from external APIs (e.g., weather data, social media feeds) and display it dynamically on the webpage.

- **Frameworks and Libraries**:

    - **React**, **Angular**: Popular JavaScript frameworks/libraries that simplify building complex user interfaces.

    - **Node.js**: Extends JavaScript to **server-side** development, allowing developers to use JavaScript for backend processes as well.

1.4 MERN STACK

MongoDB: A highly flexible, NoSQL database used to store journal entries and user data. It enables the Flower Shop App to scale efficiently, supporting large amounts of data without requiring rigid structures, which is ideal for managing diverse travel logs and user details.

Express.js: A minimal, fast, and robust web application framework built on Node.js. It manages the app's back-end logic, handling server-side routing and interactions with the database. Express simplifies connecting the front-end React components with the database, enabling smooth data flow.

React.js: A powerful front-end JavaScript library that allows for the creation of dynamic user interfaces. In the Flower Shop App, React's component-based architecture ensures that features like writing, editing, and reading journals are rendered efficiently and updated in real time without refreshing the entire page.

Node.js: A versatile JavaScript runtime environment that runs on the server side. It processes user requests, manages routes, and interacts with the MongoDB database, allowing for a smooth, non-blocking performance. Node.js ensures that the app handles multiple user interactions and back-end processes effectively, making it highly scalable.

1.5 How the MERN stack works?

The **MERN stack**—which includes MongoDB, Express.js, React.js, and Node.js powers the Flower Shop App, letting users create, edit, and share their travel stories. Here’s how it all works together:

React.js (Front-End): Users interact with the app through React, writing, editing, and reviewing journal entries. Each part of the interface is a reusable component that updates dynamically without refreshing the page. It sends data to the server using HTTP requests.

Express.js (Back-End): Express handles incoming requests from React, routes them to the right endpoints (like saving or fetching journals), and ensures data is validated before heading to the database.

Node.js (Back-End Runtime): Running on Node.js, the server processes the requests from React and interacts with the database. It ensures multiple users can work smoothly at the same time.

MongoDB (Database): MongoDB stores all journal entries, user data, and travel info as documents, making it easy to manage, scale, and retrieve the data through CRUD operations.

1.6 How It All Comes Together:

1. **React Sends Order Data to the Server:** When a customer creates or edits an order in the flower shop (such as choosing flowers, specifying delivery details, and adding a personalized message), React gathers all the input data. This data is then sent to the backend using HTTP requests, typically through libraries like Axios or Fetch.

2. **Express Processes the Order Request**: On the server side, Express.js handles incoming requests, processes the data, and validates it to ensure accuracy (like checking stock availability for chosen flowers). It then routes the request to the appropriate endpoint, depending on whether the action is to create a new order, update an existing order, or retrieve order details.

3. **MongoDB Stores the Order Information**: After processing, Express forwards the order data to MongoDB. The order, including details like selected flowers, delivery date, customer information, and custom messages, is stored as a document within an orders collection. MongoDB's flexible schema accommodates various order details and any additional specifications.

4. **Server Responds, React Updates the UI:** Once the order is successfully saved in MongoDB, the server sends a response back to React. React then updates the UI, either by displaying a confirmation message or by updating the order list to reflect the new or edited order. This process is handled dynamically, ensuring a smooth user experience without the need to refresh the entire page.

CHAPTER 2

METHODOLOGY

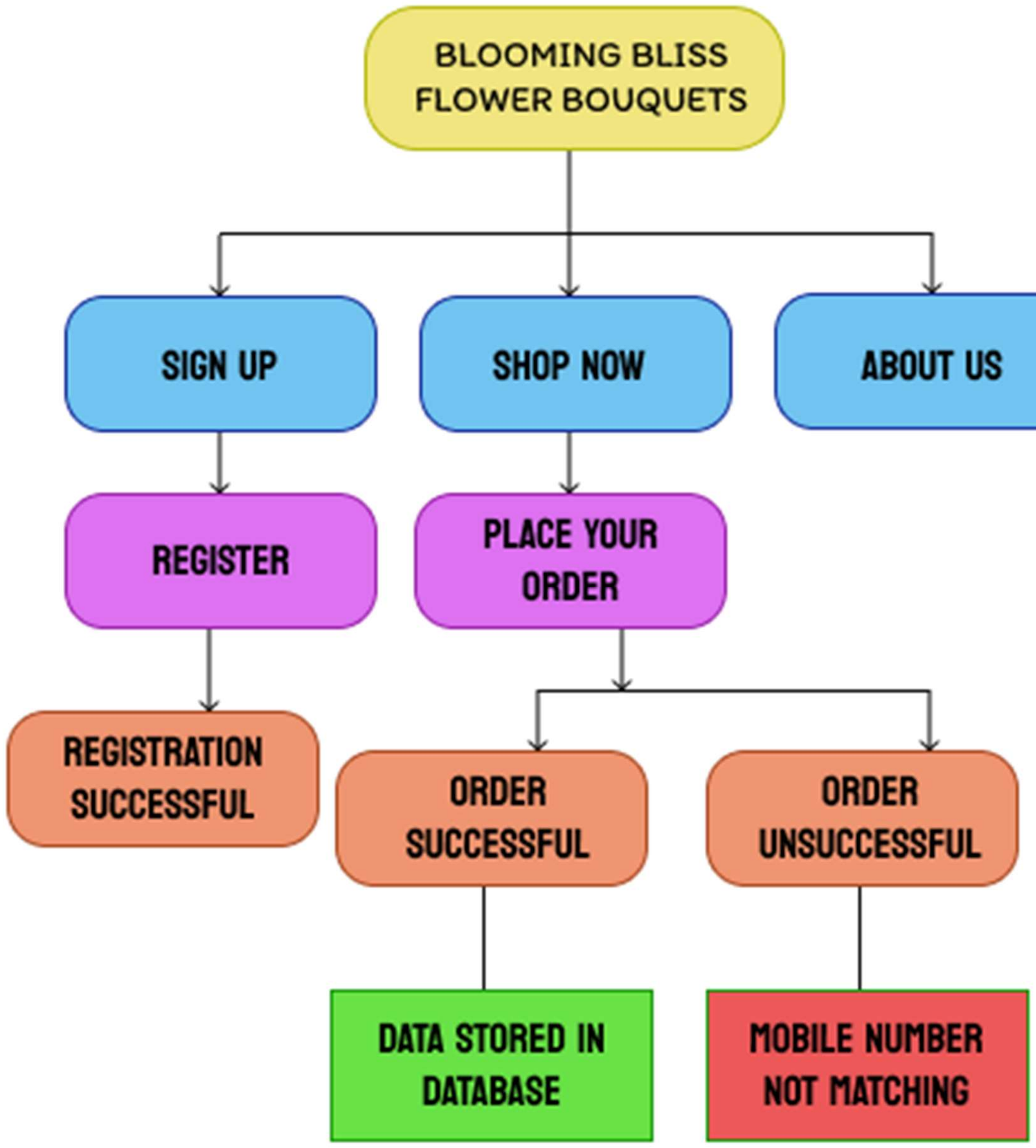
2.1 OBJECTIVE

The objective of the Flower Shop App is to offer customers a seamless platform for browsing, customizing, and purchasing beautiful floral arrangements, while also providing real-time order tracking and personalization options. This creates a personalized and interactive shopping experience, making it easy for customers to find the perfect flowers for any occasion.

2.2 PROBLEM STATEMENT

The problem addressed by the Flower Shop App is the absence of a convenient, user-friendly platform where customers can easily browse, customize, and purchase floral arrangements. Many traditional flower shopping experiences lack personalization, real-time order tracking, and efficient online ordering options, which can lead to a disconnected and time-consuming experience for customers trying to find the perfect flowers for their needs.

2.3 BLOCK DIAGRAM



2.4 MODULE EXPLANATION

User Interface (React Front-End):

- **Browse Products**: This component displays the available floral arrangements in an organized format. Customers can view flowers sorted by categories, price, or other relevant filters. Each product listing includes details like name, description, price, and available customizations.

- **Product Details**: This module shows more detailed information about a selected flower or bouquet. Users can view larger images, read a detailed description, choose custom options (such as size or add-ons), and add the item to their shopping cart.

- **Shopping Cart**: Provides a summary of selected products. Users can adjust quantities, remove items, and see the total cost. Changes made in the cart are dynamically reflected without page reloads, and users can proceed to checkout from this module.

- **Checkout Form**: Collects customer details for billing, shipping, and payment. Input fields include name, delivery address, payment information, and any special delivery instructions. Upon submission, the data is processed and saved to the back-end.

- **Order Confirmation & Tracking**: After a successful order, this component displays an order summary with tracking information. Users can check order status and expected delivery dates without leaving the page.

Back-End (Order Management Form):

- **Order Management Form**: Handles order details, allowing customers to submit their name, delivery address, payment info, and other order-related details. Once the form is completed, data is processed by the back-end server (Node.js/Express) and saved in the MongoDB database.

Back-End Logic (Express.js):

- API Management: Express manages API routes and processes requests from the front-end. It validates incoming data (such as payment and delivery information), handles responses, and maintains communication between React and MongoDB.

Server-Side Logic (Node.js):

- Order Processing: Manages back-end operations like handling incoming requests, managing database interactions, and ensuring the app runs asynchronously to handle multiple customer orders efficiently.

Database (MongoDB):

- Order & Product Data Storage: MongoDB stores details of orders, products, and customer information. It offers flexibility for storing text, dates, images, and customizations, allowing for efficient data retrieval and updates as customers place or modify orders**.**

CHAPTER 3

RESULT AND DISCUSSIONS

The Flower Shop App effectively provides users with a convenient platform to browse, customize, and purchase floral arrangements. The front-end, developed with React, offers a smooth and responsive user experience, allowing customers to navigate through products, add items to the cart, and complete purchases with real-time updates and without page reloads.

On the back-end, Node.js and Express.js manage requests efficiently, ensuring that order details, customer information, and product data are securely processed and stored in MongoDB. The app's functionality includes essential features like product browsing, order customization, shopping cart management, and checkout, creating a cohesive and interactive shopping experience.

Overall, the Flower Shop App successfully addresses the need for a streamlined online flower shopping experience, demonstrating its effectiveness in providing users with a comprehensive and personalized platform for selecting and purchasing floral arrangements.

SCREENSHOTS

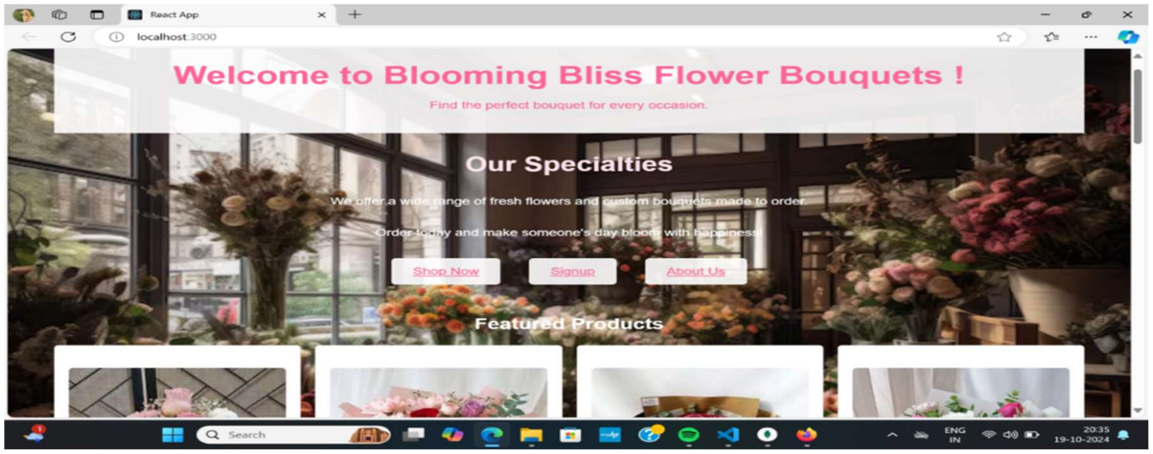

React App
localhost:3000
Welcome to Blooming Bliss Flower Bouquets !
Find the perfect bouquet for every occasion.
Our Specialties
We offer a wide range of fresh flowers and custom bouquets made to order.
Order today and make someone's day bloom with happiness!
Shop Now
Signup
About Us
Featured Products

Figure 3.1 Home page

This is a homepage for "Blooming Bliss Flower Bouquets." It features a welcoming banner, highlights the store's flower and bouquet customization services, and offers options to shop, sign up, or learn more about the business.

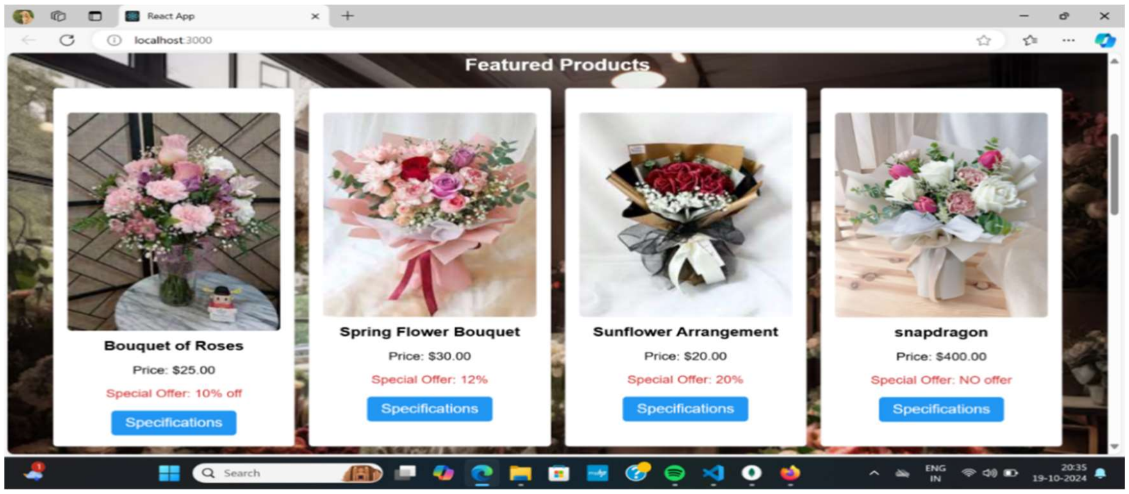
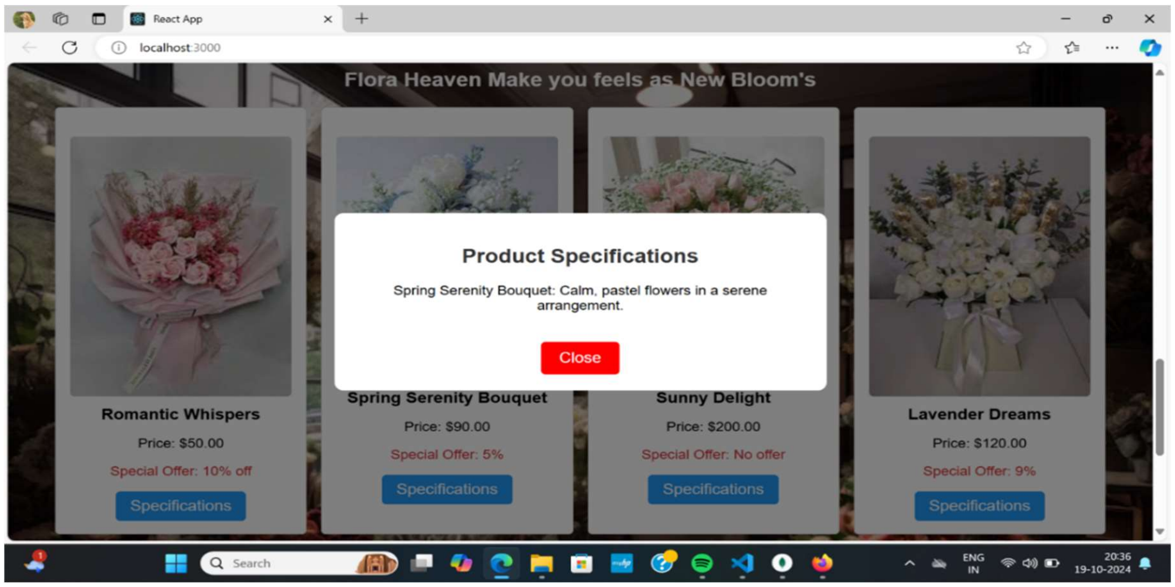


Figure 3.2 Featured products

This section showcases featured floral products with images, prices, and special discount offers for select bouquets. Each product has a "Specifications" button for further details


React App
localhost:3000
Flora Heaven Make you feels as New Bloom's

Product Specifications

Spring Serenity Bouquet: Calm, pastel flowers in a serene arrangement.

Close

Romantic Whispers
Price: $50.00
Special Offer: 10% off
Specifications

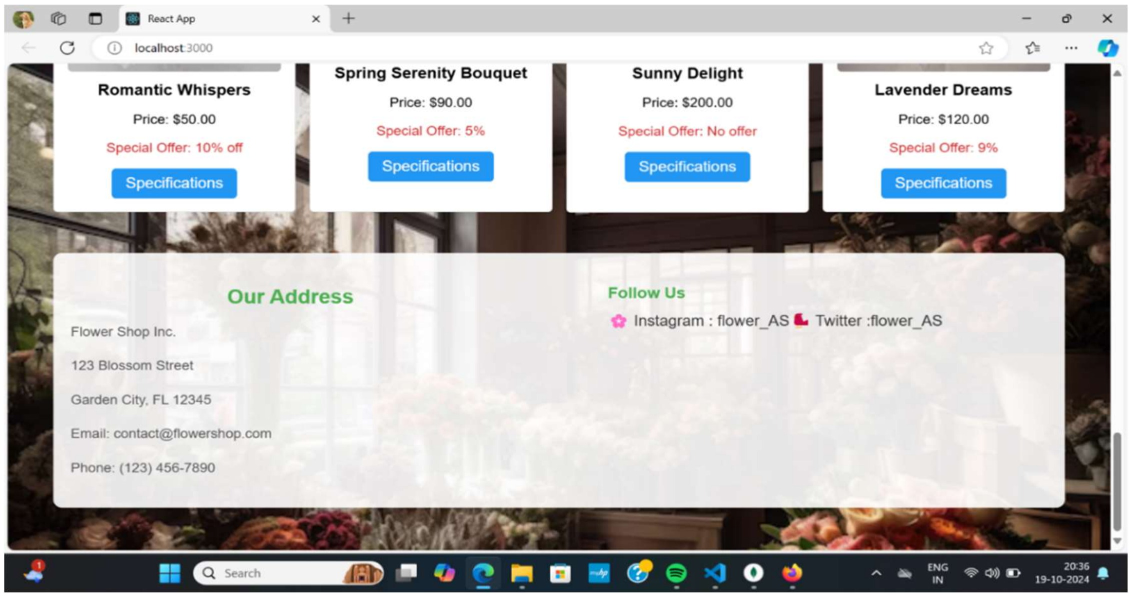
Spring Serenity Bouquet
Price: $90.00
Special Offer: 5%
Specifications

Sunny Delight
Price: $200.00
Special Offer: No offer
Specifications

Lavender Dreams
Price: $120.00
Special Offer: 9%
Specifications

Figure 3.3 Product specifications

This image displays a pop-up window showing the product specifications


React App
localhost:3000

Romantic Whispers
Price: $50.00
Special Offer: 10% off
Specifications

Spring Serenity Bouquet
Price: $90.00
Special Offer: 5%
Specifications

Sunny Delight
Price: $200.00
Special Offer: No offer
Specifications

Lavender Dreams
Price: $120.00
Special Offer: 9%
Specifications

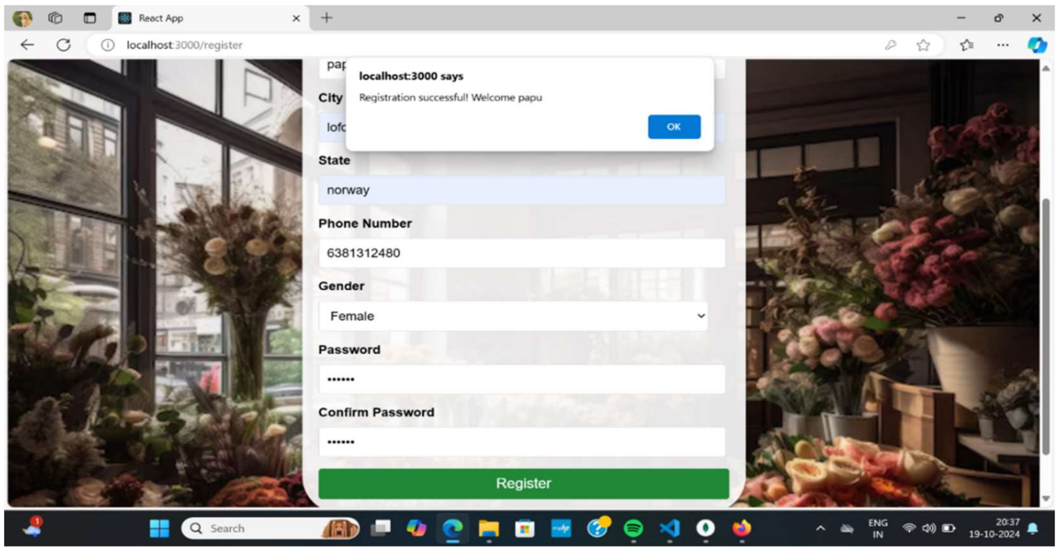
Our Address

Flower Shop Inc.
123 Blossom Street
Garden City, FL 12345
Email: contact@flowershop.com
Phone: (123) 456-7890

Follow Us
Instagram : flower_AS Twitter :flower_AS

Figure 3.4 Contact page

This image displays our bouquet shop’s address and our contact details


localhost:3000/register

localhost:3000 says

Registration successful! Welcome papu

City

lofc

State

norway

Phone Number

6381312480

Gender

Female

Password

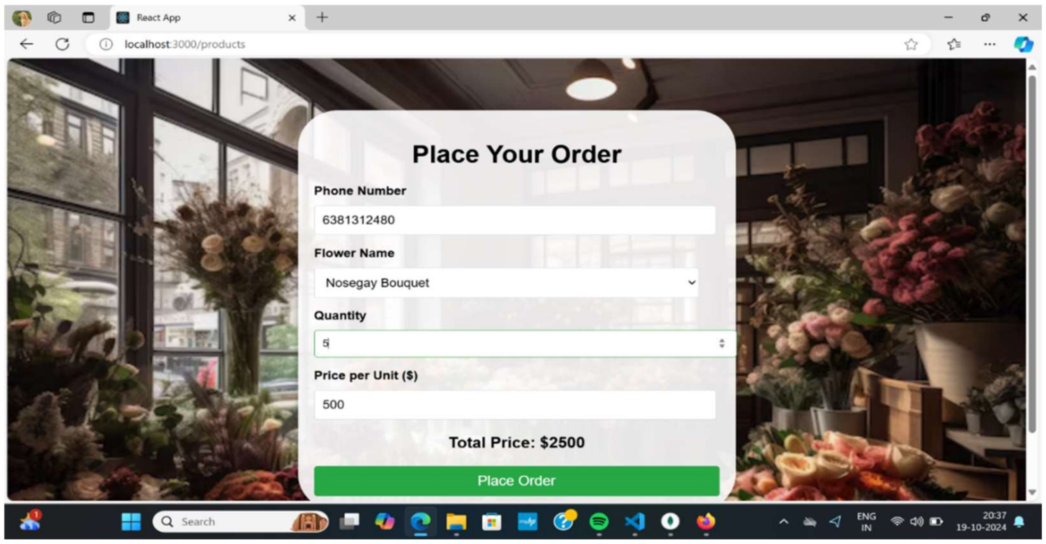
••••••

Confirm Password

••••••

Figure 3.5 Registration page

This image shows that once all the details filled in the form , the registration completes and welcomes you to the shopping page.


localhost:3000/products

Place Your Order

Phone Number

6381312480

Flower Name

Nosegay Bouquet

Quantity

5

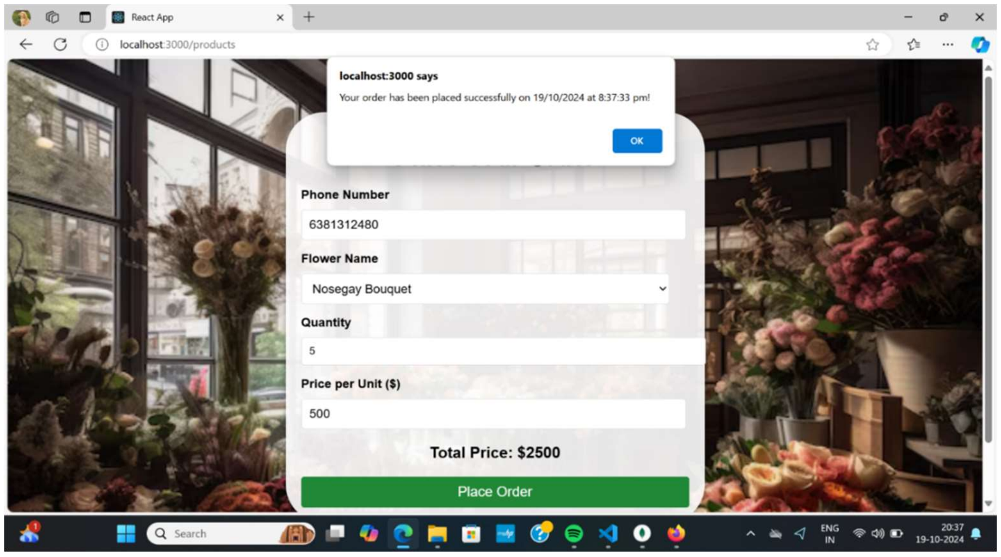
Price per Unit ($)

500

Total Price: $2500

Figure 3.6 Place your order page

This image shows that to place your order and displays the total price of your orders made.


localhost:3000 says

Your order has been placed successfully on 19/10/2024 at 8:37:33 pm!

OK

Phone Number

6381312480

Flower Name

Nosegay Bouquet

Quantity

5

Price per Unit ($)

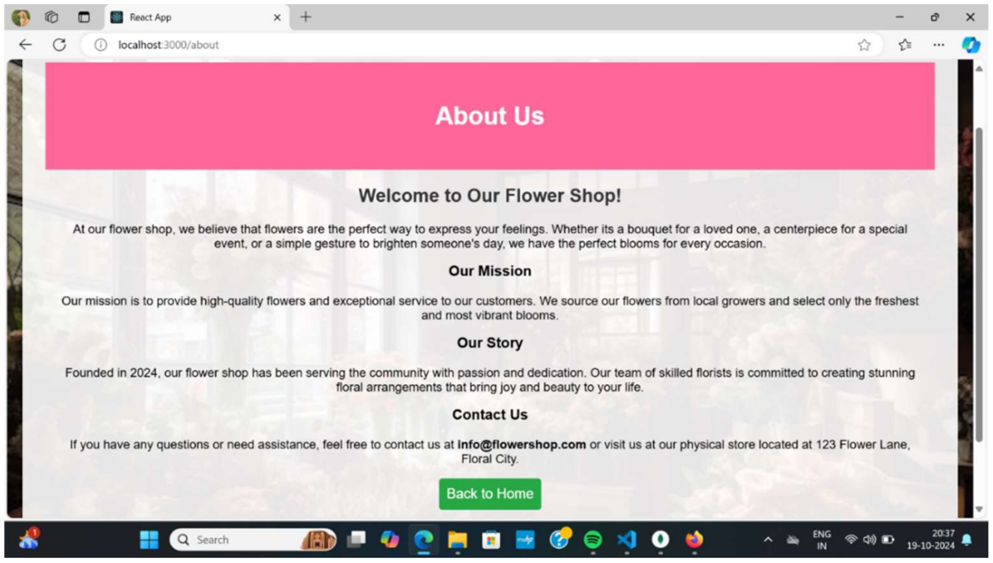
500

Total Price: $2500

Place Order

Figure 3.7 Order successful page

This image shows that your order have been placed successfully .


About Us

Welcome to Our Flower Shop!

At our flower shop, we believe that flowers are the perfect way to express your feelings. Whether its a bouquet for a loved one, a centerpiece for a special event, or a simple gesture to brighten someone's day, we have the perfect blooms for every occasion.

Our Mission

Our mission is to provide high-quality flowers and exceptional service to our customers. We source our flowers from local growers and select only the freshest and most vibrant blooms.

Our Story

Founded in 2024, our flower shop has been serving the community with passion and dedication. Our team of skilled florists is committed to creating stunning floral arrangements that bring joy and beauty to your life.

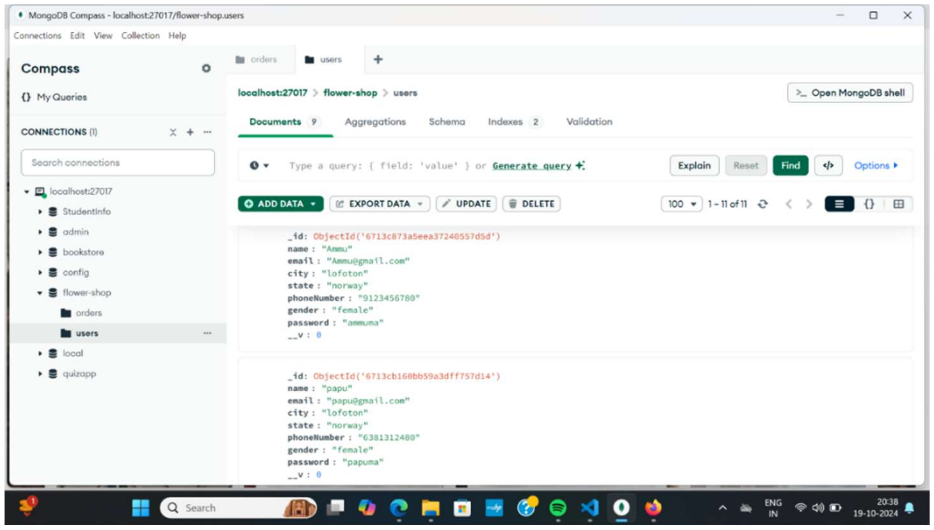
Contact Us

If you have any questions or need assistance, feel free to contact us at **info@flowershop.com** or visit us at our physical store located at 123 Flower Lane, Floral City.

Back to Home

Figure 3.8 About us page

This image shows that details of our bouquet shop

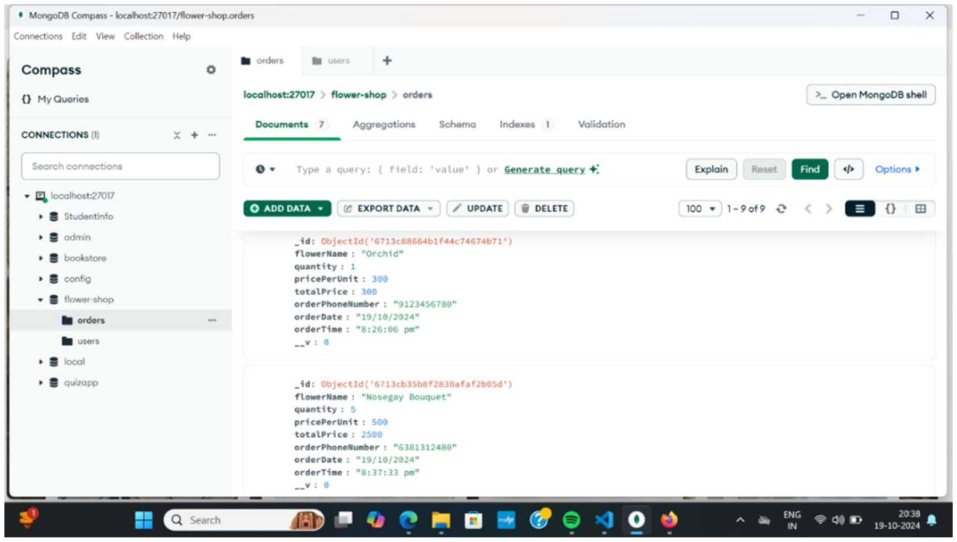


```
_id: ObjectId('6713c873a5eea37240557d5d')
name : "Ammu"
email : "Ammu@gmail.com"
city : "lofoton"
state : "norway"
phoneNumber : "9123456780"
gender : "female"
password : "ammuma"
__v : 0

_id: ObjectId('6713cb160bb59a3dff757d14')
name : "papu"
email : "papu@gmail.com"
city : "lofoton"
state : "norway"
phoneNumber : "6381312480"
gender : "female"
password : "papuma"
__v : 0
```

Figure 3.9 Database connectivity

This image shows that the user page have been connected to the database

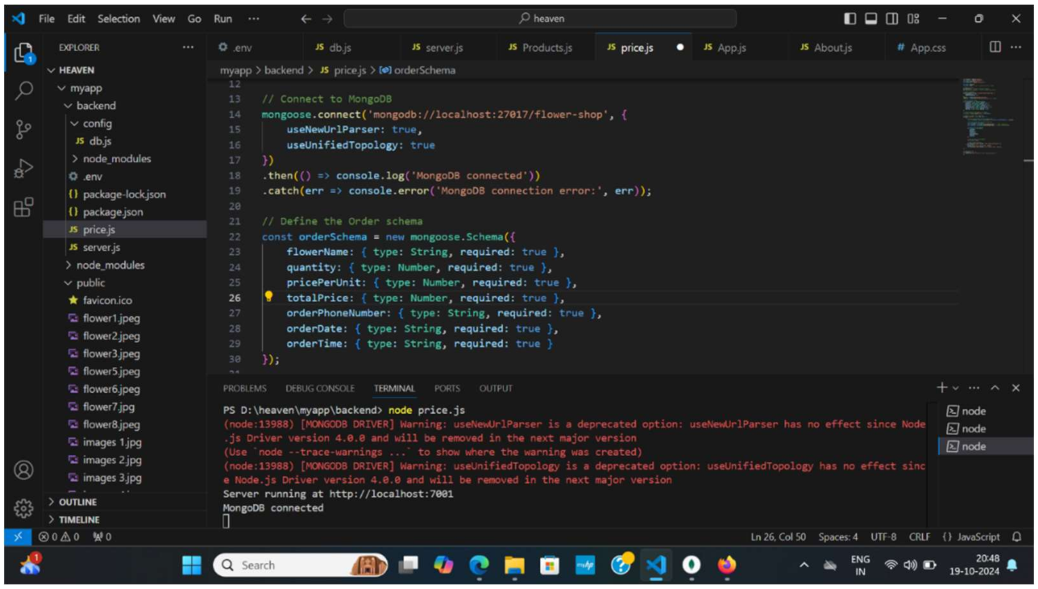


```
_id: ObjectId('6713c88664b1f44c74674b71')
flowerName : "Orchid"
quantity : 1
pricePerUnit : 300
totalPrice : 300
orderPhoneNumber : "9123456780"
orderDate : "19/10/2024"
orderTime : "8:26:06 pm"
__v : 0

_id: ObjectId('6713cb35b8f2830afaf2b05d')
flowerName : "Nosegay Bouquet"
quantity : 5
pricePerUnit : 500
totalPrice : 2500
orderPhoneNumber : "6381312480"
orderDate : "19/10/2024"
orderTime : "8:37:33 pm"
__v : 0
```

Figure 3.10 Database connectivity

This image shows that the order page have been connected to the database



```
// Connect to MongoDB
mongoose.connect('mongodb://localhost:27017/flower-shop', {
    useNewUrlParser: true,
    useUnifiedTopology: true
})
.then(() => console.log('MongoDB connected'))
.catch(err => console.error('MongoDB connection error:', err));

// Define the Order schema
const orderSchema = new mongoose.Schema({
    flowerName: { type: String, required: true },
    quantity: { type: Number, required: true },
    pricePerUnit: { type: Number, required: true },
    totalPrice: { type: Number, required: true },
    orderPhoneNumber: { type: String, required: true },
    orderDate: { type: String, required: true },
    orderTime: { type: String, required: true }
});
```

```
PS D:\heaven\myapp\backend> node price.js
(node:13988) [MONGODB DRIVER] Warning: useNewUrlParser is a deprecated option: useNewUrlParser has no effect since Node
.js Driver version 4.0.0 and will be removed in the next major version
(Use `node --trace-warnings ...` to show where the warning was created)
(node:13988) [MONGODB DRIVER] Warning: useUnifiedTopology is a deprecated option: useUnifiedTopology has no effect sinc
e Node.js Driver version 4.0.0 and will be removed in the next major version
Server running at http://localhost:7001
MongoDB connected
```

Figure 10.11 Database connectivity

This image shows that the code have been connected to the database

CHAPTER 4

CONCLUSION

In conclusion, building a flower bouquet shop website using React.js for the frontend and MongoDB as the database provides an efficient, modern, and scalable solution. React's dynamic UI capabilities allow for a smooth and responsive user experience, while MongoDB's flexibility supports efficient storage and retrieval of product and customer data. Together, they create a robust and interactive platform for managing orders, displaying products, and engaging users, making it ideal for a growing online flower business.

CHAPTER 5

REFERENCES

1.React Documentation

https://reactjs.org/docs/getting-started.html

2.MongoDB Documentation

https://www.mongodb.com/docs/

3.MERN Stack Tutorial

https://www.freecodecamp.org/news/mern-stack-tutorial/

4.Express.js Documentation

https://expressjs.com/

5.Mongoose Documentation

https://mongoosejs.com/docs/guide.html