A Project Report on Application

"VEGIBLE"

For the course

FULL STACK DEVELOPMENT

Submitted by :

Aryan (201500143)

Nishant Dwivedi (201500443)

Shatrughan Singh Bisen (201500646)

Sourabh (201500703)

Guided by: Mr. Bhanu Kapoor (Technical Trainer)

Contents: -

Acknowledgement

It gives us a great sense of pleasure to present the synopsis of the B.Tech mini project undertaken during B.Tech III Year. This project is going to be an acknowledgement to the inspiration, drive and technical assistance will be contributed to it by many individuals. We owe special debt of gratitude to Mr. Bhanu Kapoor , Technical Trainer , for providing us with an encouraging platform to develop this project, which thus helped us in shaping our abilities towards a constructive goal and for his constant support and guidance to our work.

His sincerity, thoroughness and perseverance has been a constant source of inspiration for us. We believe that he will shower us with all his extensively experienced ideas and insightful comments at different stages of the project & also taught us about the latest industry-oriented technologies. We also do not like miss the opportunity to acknowledge the contribution of all faculty members of the department for their kind guidance and co-operation.

Aryan (201500143)

Nishant Dwivedi (201500443)

Shatrughan Singh Bisen (201500646)

Sourabh (201500703)

Declaration

We hereby declare that the project work entitled **"Vegible"** submitted to the GLA University, is a record of an original work done by us groupmates under the guidance of **Mr. Bhanu Kapoor**, and this project work is submitted in the partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in Computer Science & Engineering. The results embodied in this thesis have not been submitted to any other University or Institute for the award of any degree or diploma.

Signed and submitted by

Aryan (201500143)
Nishant Dwivedi (201500443)
Shatrughan Singh Bisen (201500646)
Sourabh (201500703)

INTRODUCTION

Welcome to the introduction of our full-stack project on a Vegetable e-commerce website built using the MERN stack.

Our platform provides a user-friendly interface with features like search functionality, product filtering, shopping cart management, payment processing, and order tracking. Additionally, we have integrated advanced security features to protect user data and ensure secure online transactions.

Our website has been developed with a responsive design, making it accessible on multiple devices such as desktops, laptops, tablets, and smartphones. We have used modern development practices to ensure that our website is fast, reliable, and scalable to meet the growing demands of our users.

In this project, we have used MongoDB as our database to store the product and user data. Express.js provides a robust framework for building APIs, and Node.js is used to run server-side JavaScript code. React has been used for the front-end development of our website, providing a dynamic and interactive user experience.

Overall, this Vegetable e-commerce website is a comprehensive project that demonstrates the full potential of the MERN stack. We hope you enjoy exploring our project and appreciate the effort we have put in to make it a reality.

Development Tools:

- Languages: React, Nodejs, expressjs.
- We used VS Code IDE as our editor.
- Connection Built with Socket.io.
- Data Stored with MongoDB and MongoDB Compass.

Software and Hardware Requirements:

- An internet connection.
- A Mac, Linux, or Windows 10 or Windows 11 computer.
- A web browser like Chrome or Microsoft Edge.

Features :

1. Secure Log in-out (password encryption)
2. Secure Vegetable Shopping (End to End Encrypted)
3. Simple and User-Friendly UI
4. Free to use
5. Not restricted to any one browser

HYPOTHESIS

Node.JS

Node JS is a JavaScript platform for building fast, scalable, network applications built on Google's V8 Engine. Node is single threaded and built around the paradigm of noneblocking IO. With Node.js each incoming request by the user is handled by one single thread in opposition to the multithreaded techniques used by PHP to scale the operations. Each request handled by this thread is coupled with a callback function that is called upon completion of the task. This is possible due to the fundamental support of JavaScript for events, Asynchronous operations, and callbacks; and Node.js puts JavaScript on the server side.

Node.js single-thread model vs traditional servers

Express.js

Although Node is capable of independently act as a web server, there are frameworks designed to make it more powerful and efficient -the most popular being Express.js. It is fast, minimal, and has several applications built for it. Express was chosen for several reasons among them are:

- Minimalistic. Express makes it possible to build an HTTP server very easily by wrapping the backend code of Node.js, and it makes the building of a RESTful API very simple.
- Express supports middleware which are functions called in a sequential order on a request or response. For example, a middleware I am using is body parser, which parses the body of incoming HTTP requests with a form submission. This allows me to access the parameters of the request directly. Middleware can do anything, and they end with next(), which calls the next middleware.

MongoDB

Additionally on the backend, there should be a database that stores user data. It would be preferable to user a JSON –NoSQL- based database to leverage the benefits of using JavaScript across the stack when the same objects stored in the DB can be processed by the server and the front-end without any additional conversion. I started my testing with a pioneer of NoSQL databases, and a member of the MEAN stack, MongoDB. Mongo is just the database software itself, and a driver is needed to connect node with the database instance and provide a layer for I/O with the database. Preferable also, there should be defined schemas for the database documents.

There are several advantages for using MongoDB such as:

- On its own, MongoDB is schema less, which provides the users with an easier way to append objects into the database. It also certain documents within the collection may have different values from others. For example, a user can have many number of phone numbers' fields in their document as they own without any need to change the other documents in the same collection.
- Indexing. This is a feature of MongoDB in which any field that is 'required' in a collection, if indexed, would be added to a sorted array with the values of this field in all documents. This way searching the collection can be made very quickly.

React:

React is a declarative, efficient, and flexible JavaScript library for building user interfaces. 'V' denotes the view in MVC. ReactJS is an opensource, component-based front-end library responsible only for the view layer of the application. It is maintained by Facebook.

React uses a declarative paradigm that makes it easier to reason about your application and aims to be both efficient and flexible. It designs simple views for each state in your application, and React will efficiently update and render just the right component when your data changes. The declarative view makes your code more predictable and easier to debug.

A React application is made of multiple components, each responsible for rendering a small, reusable piece of HTML. Components can be nested within other components to allow complex applications to be built out of simple building blocks. A component may also maintain an internal state – for example, a Tab List component may store a variable corresponding to the currently open tab.

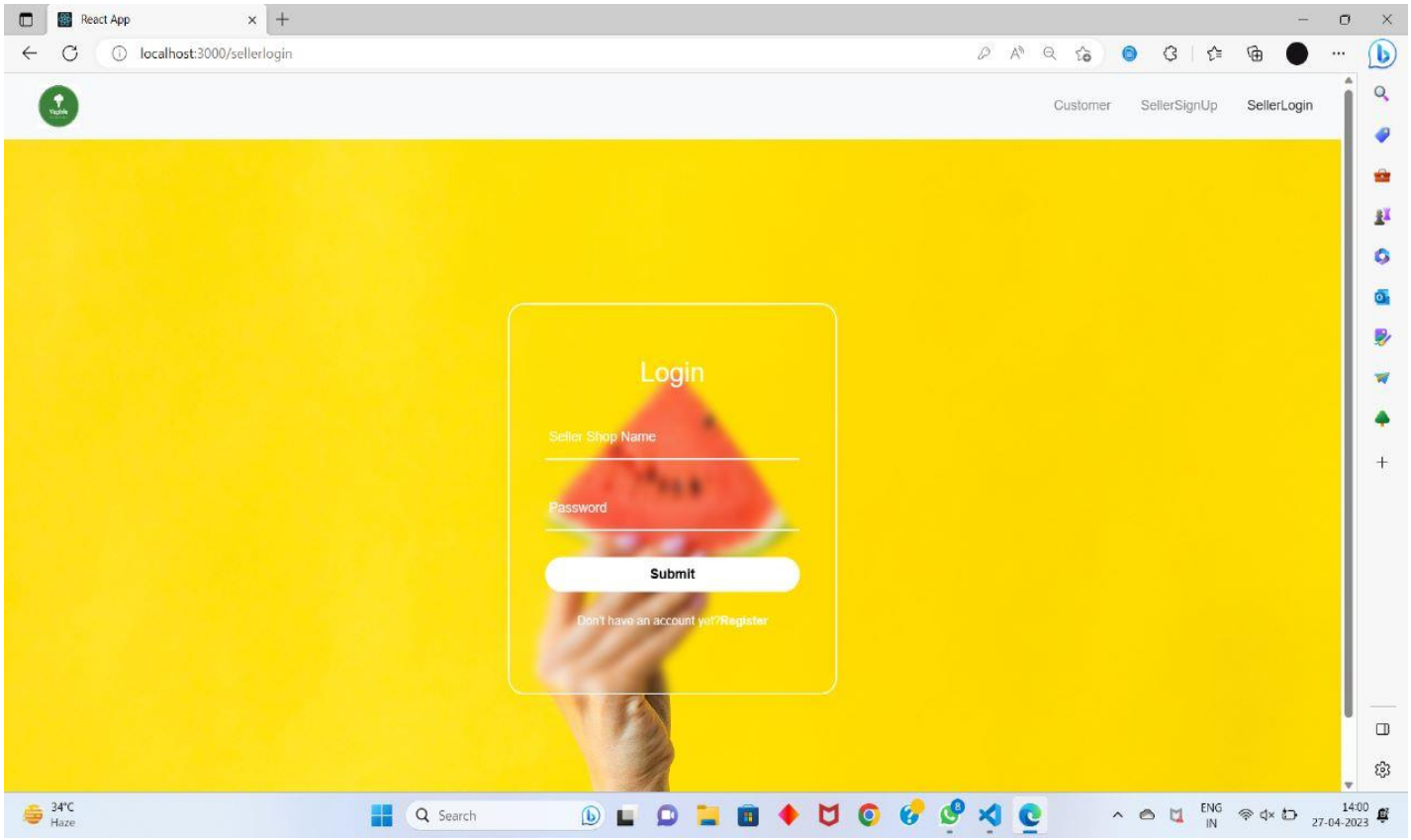
Snapshots:

Login page:

The login page is one of the most commonly used features of a web application. It allows users to authenticate themselves before they can access the rest of the application.

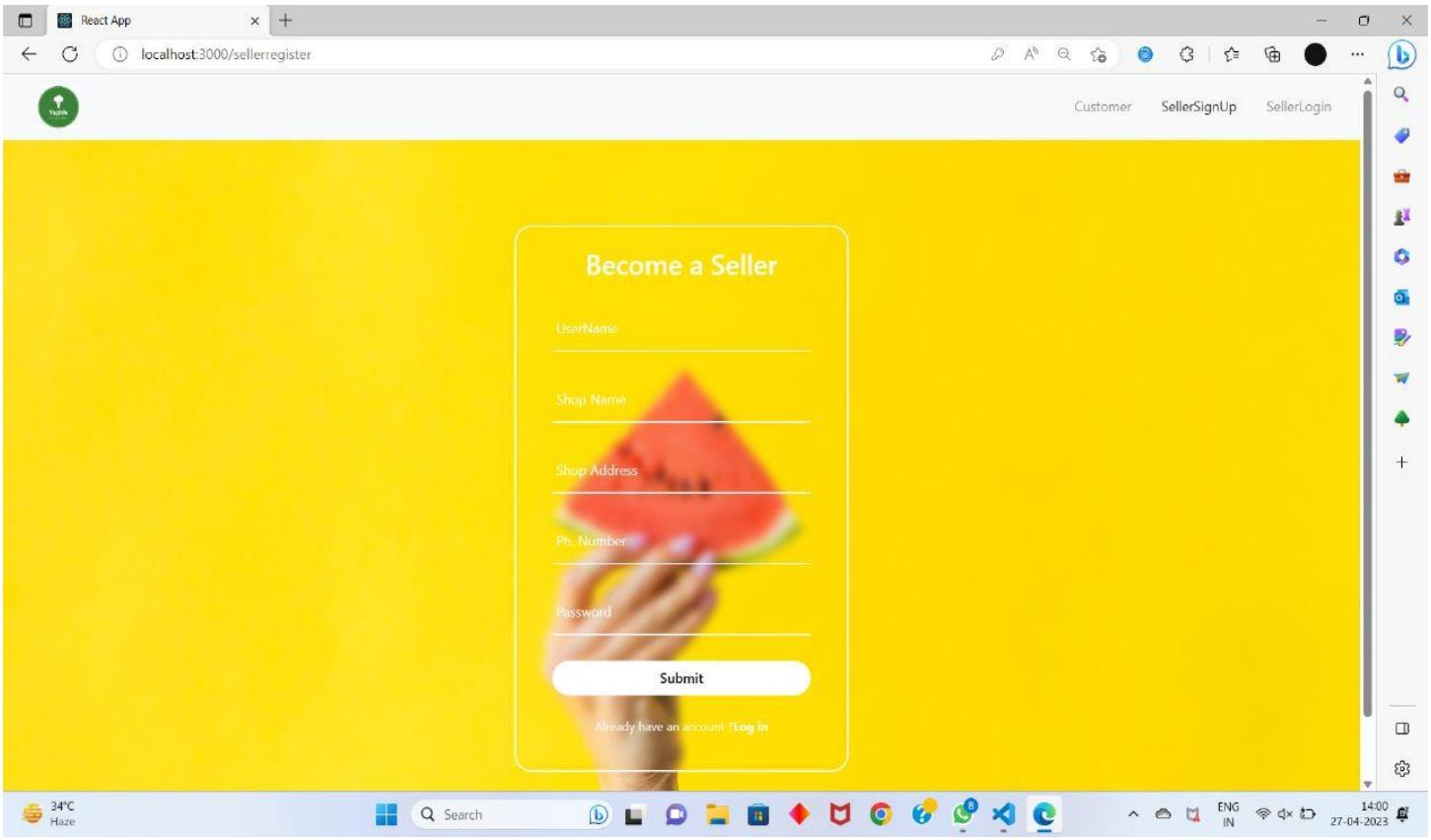
A login page typically has a form where the user can enter their username and password. Once the user has entered their credentials, the form is submitted to the server for authentication.

If the authentication is successful, the user is redirected to the main application page. If the authentication fails, the user is typically given an error message and the opportunity to try again.



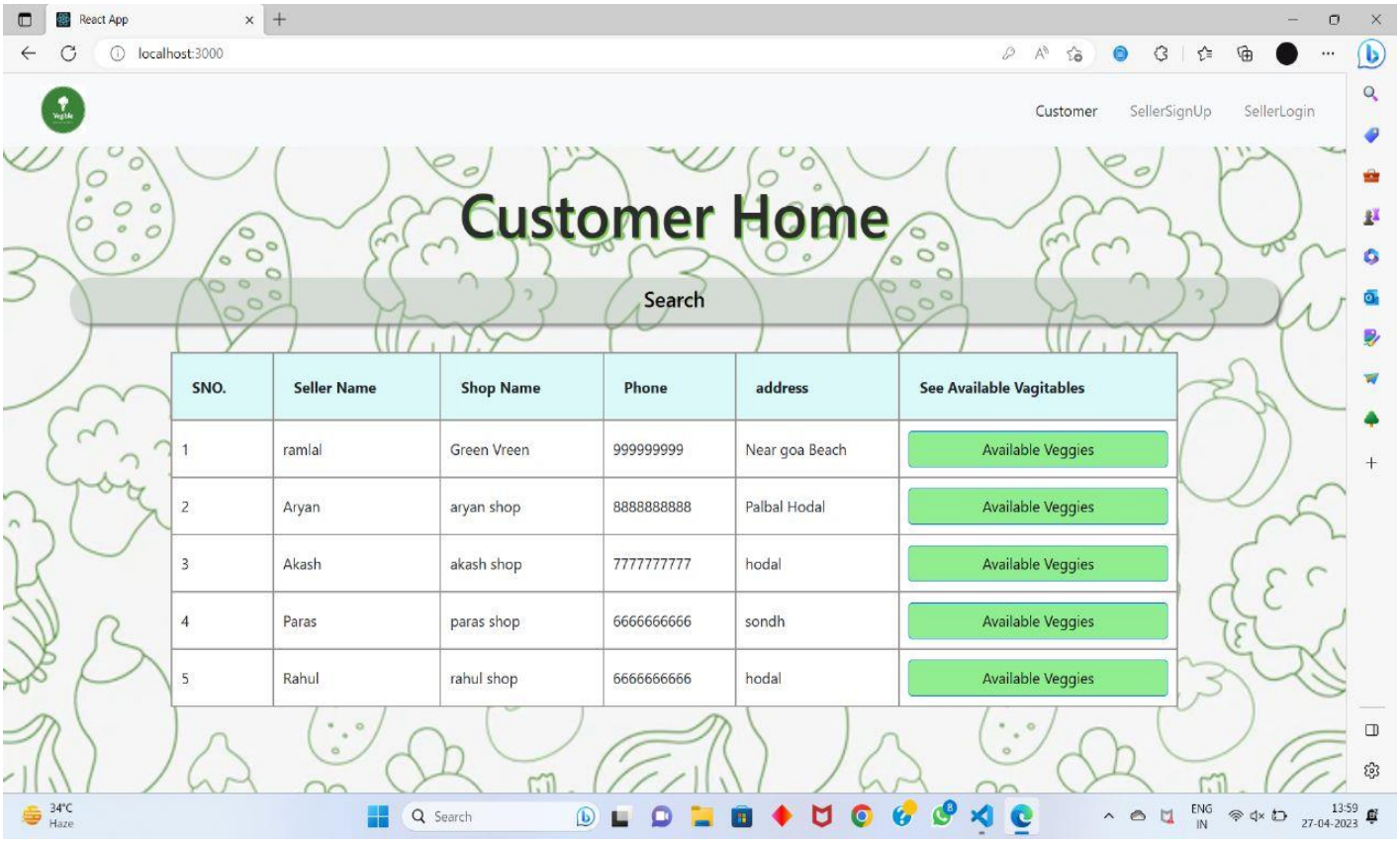
Registration Page:

The registration page is a web page on a website that allows users to sign up for a user account on that website. This page typically contains a form that allows the user to enter their desired username, password, email address, and other personal information. Once the user has filled out the form and submitted it, they will usually be redirected to a confirmation page that tells them their account has been created and provides further instructions on how to activate it.



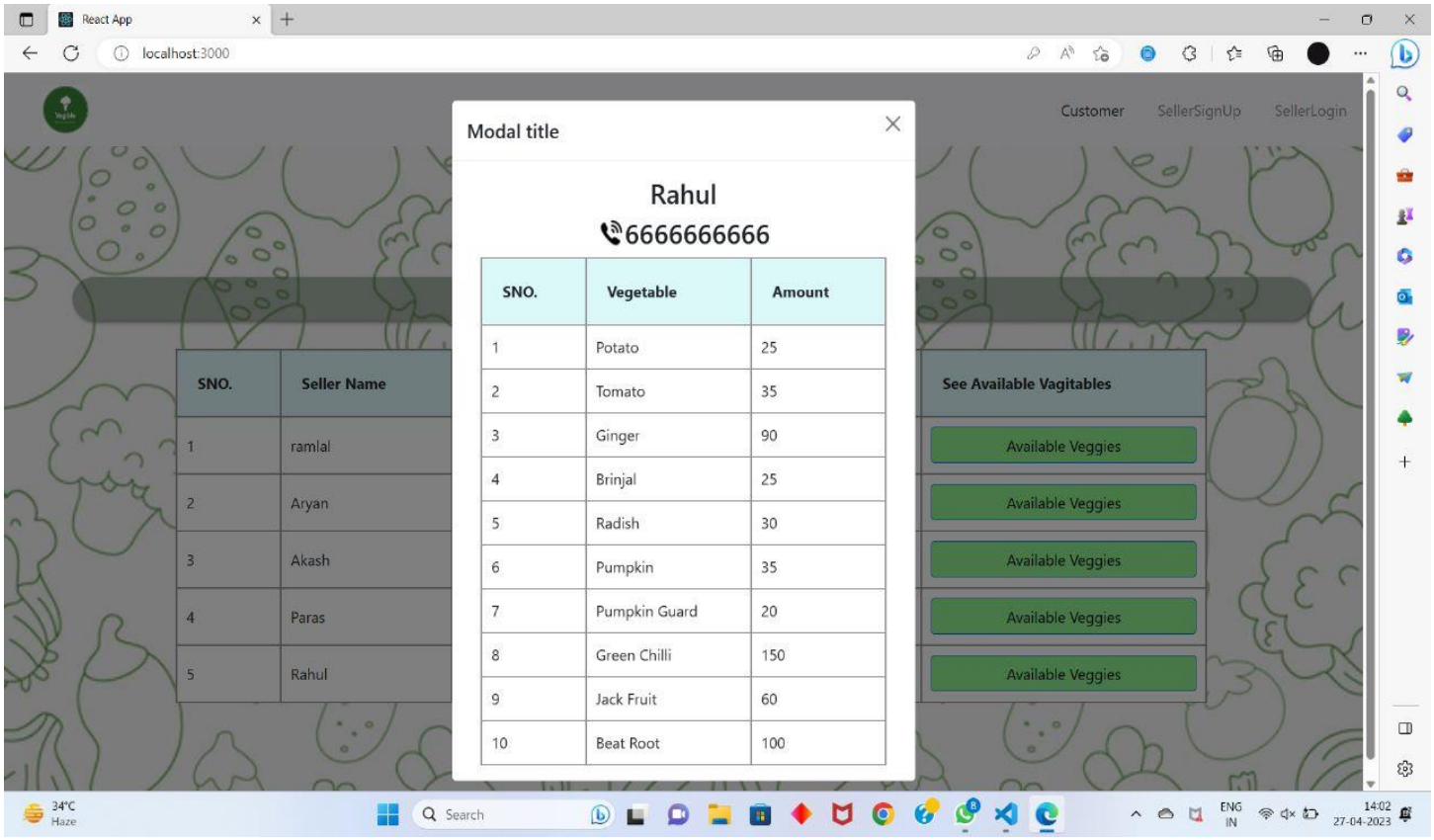
Customer Home:

This page comes right after the registration page allows user to choose seller.



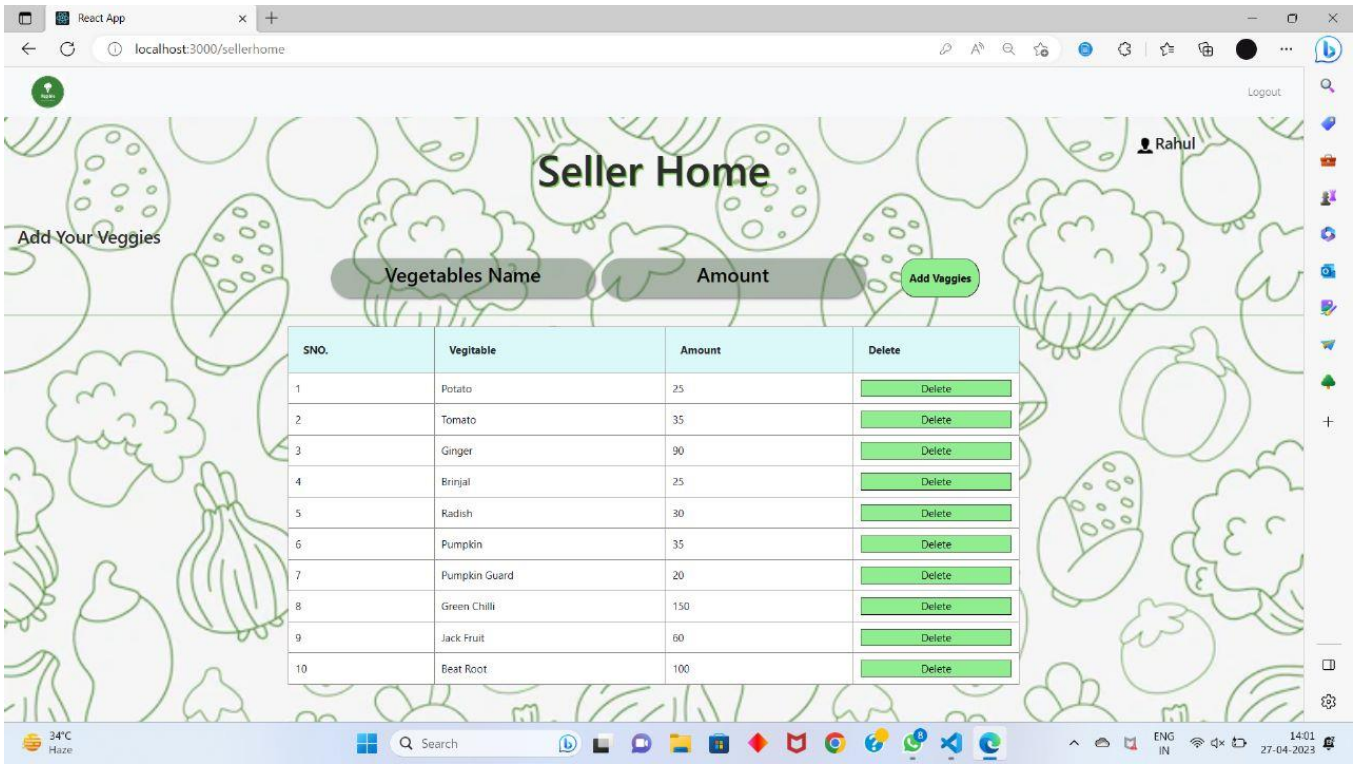
List of Vegetable:

This is where user can see themselves to the very first interface of the application. Usernames are searchable and they can use them to select the vegetables of registered sellers.



Seller Home:

Here will be the data of seller that which vegetables he has uploaded on website.



Future Scope:

1. Customized Service
2. Marketing
3. Social Networking
4. Business

References:

Books:

- Full-Stack
  Modern Full-Stack Development Pro MERN Stack
- React
  The Road to Learn React React Explained
- Full-Stack React Projects

Websites:

- https://reactjs.org/
- https://www.w3schools.com/
- https://getbootstrap.com/

Faculty Guidelines:

Mr. Bhanu Kapoor (Technical Trainer, GLA University)

GitHub Repository link:

https://github.com/sourabhthakur87/Vegibles/