WEB DEVELOPMENT
MINI PROJECT

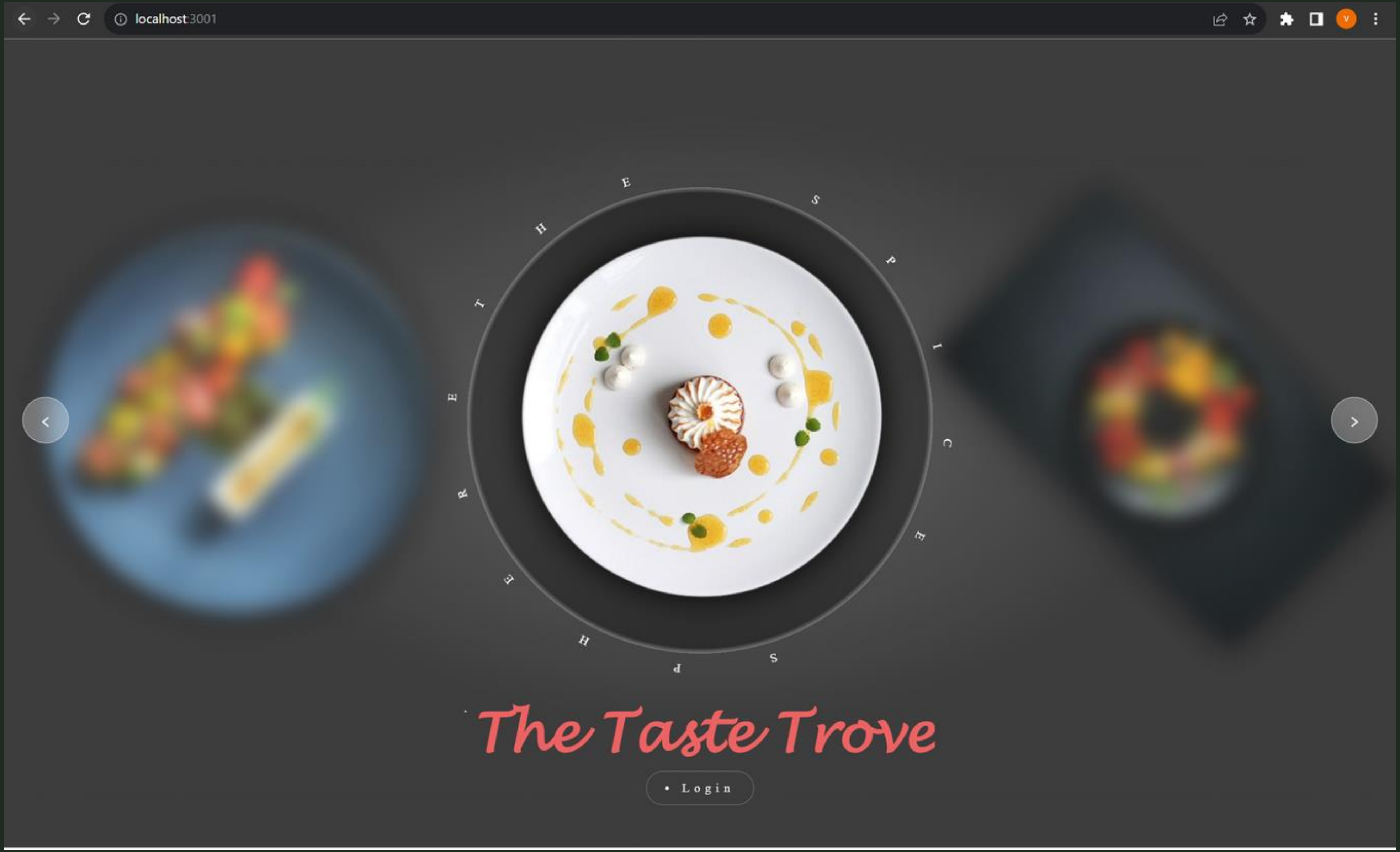
BHUMIL SHAH- 60004220063
ARHAM SHAH-60004220110
CLEON LOPES-60004220071

The Taste Trove

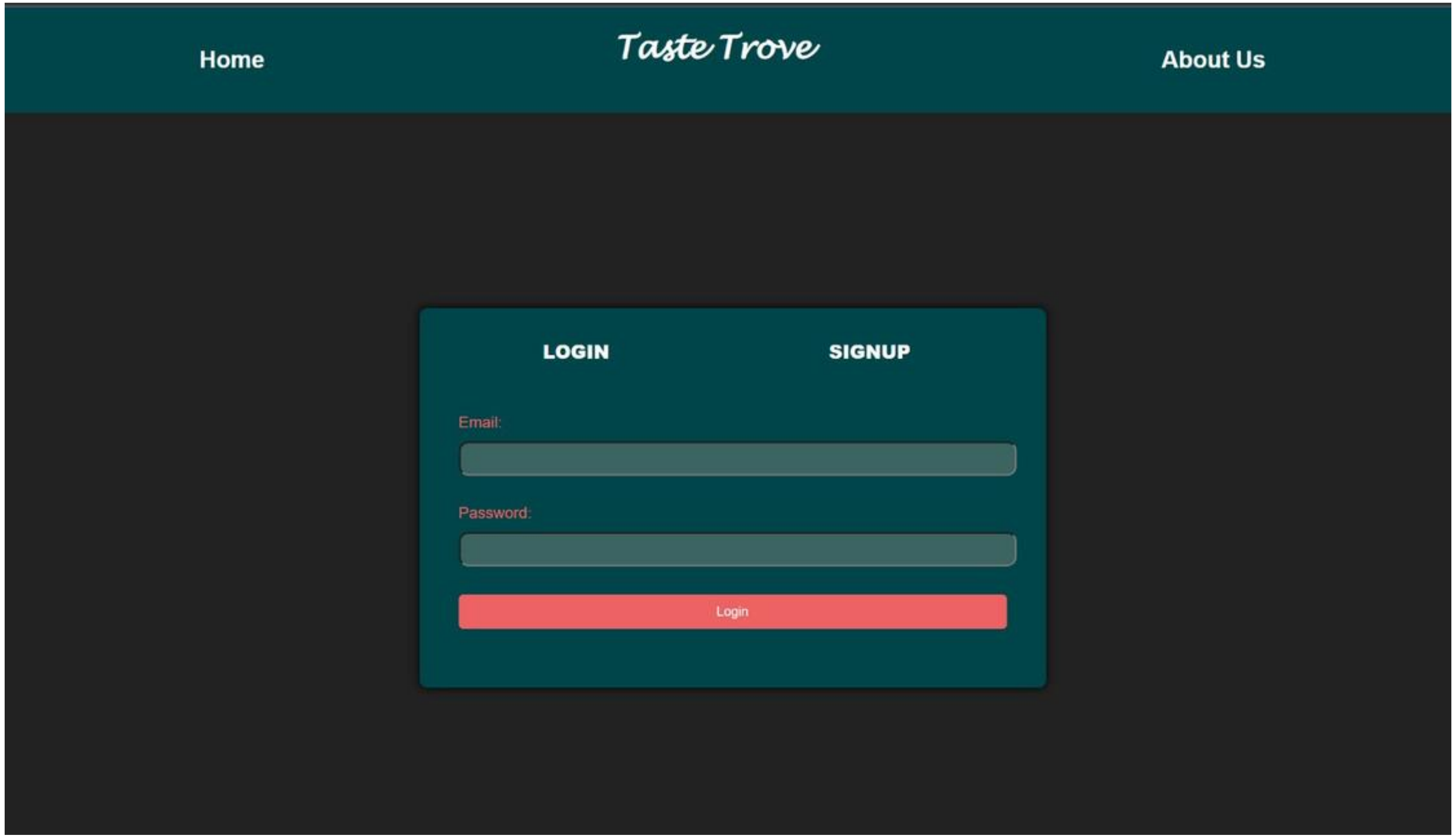
Hello!

Welcome to the Recipe Recommendation System, a flavorful odyssey crafted with the power of MongoDB, HTML, CSS, and JavaScript, Node.js and Express.js.
Our goal is to create a Recipe Recommendation System that transcends traditional cooking platforms. We aim to build a community-driven space where users can discover, contribute, and celebrate the art of cooking.

THE HOME PAGE

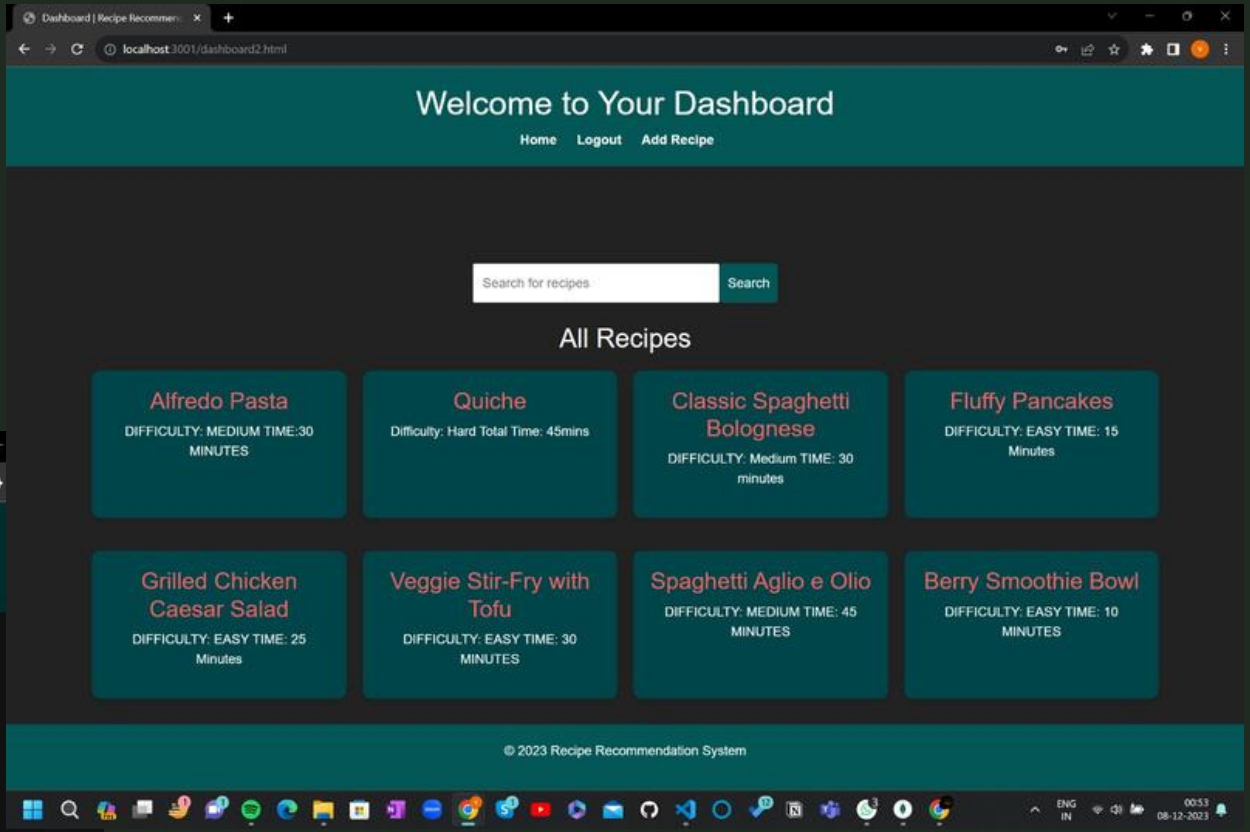


Signup & Login Page



SIGN UP TO JOIN THIS CULINARY SPACE

Digital Recipe Book.



localhost:3001/dashboard2.html

Welcome to Your Dashboard

Home Logout Add Recipe

Search for recipes | Search

All Recipes

Alfredo Pasta
DIFFICULTY: MEDIUM TIME:30 MINUTES

Quiche
Difficulty: Hard Total Time: 45mins

Classic Spaghetti Bolognese
DIFFICULTY: Medium TIME: 30 minutes

Fluffy Pancakes
DIFFICULTY: EASY TIME: 15 Minutes

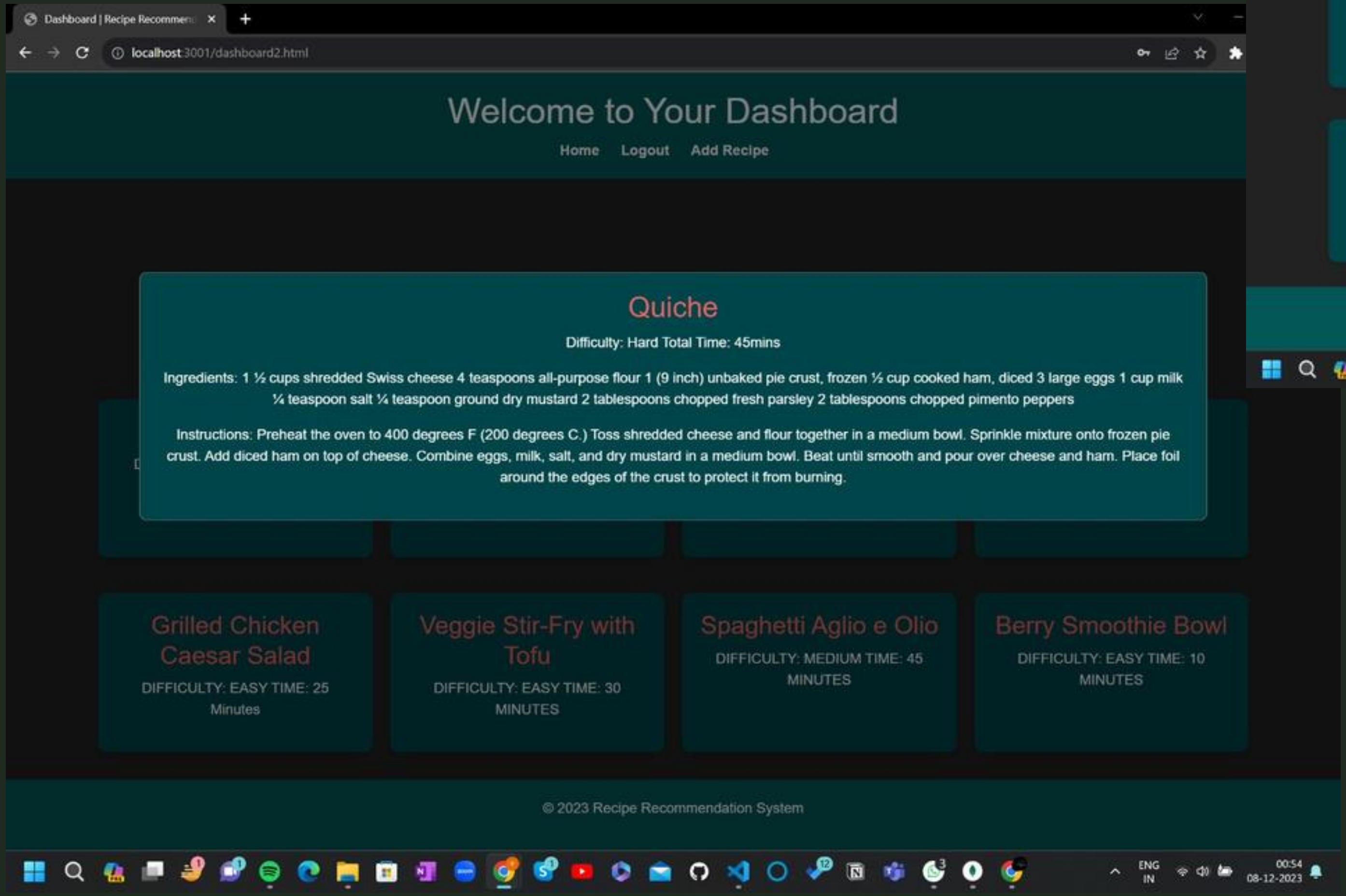
Grilled Chicken Caesar Salad
DIFFICULTY: EASY TIME: 25 Minutes

Veggie Stir-Fry with Tofu
DIFFICULTY: EASY TIME: 30 MINUTES

Spaghetti Aglio e Olio
DIFFICULTY: MEDIUM TIME: 45 MINUTES

Berry Smoothie Bowl
DIFFICULTY: EASY TIME: 10 MINUTES

© 2023 Recipe Recommendation System



localhost:3001/dashboard2.html

Welcome to Your Dashboard

Home Logout Add Recipe

Quiche

Difficulty: Hard Total Time: 45mins

Ingredients: 1 ½ cups shredded Swiss cheese 4 teaspoons all-purpose flour 1 (9 inch) unbaked pie crust, frozen ½ cup cooked ham, diced 3 large eggs 1 cup milk ¼ teaspoon salt ¼ teaspoon ground dry mustard 2 tablespoons chopped fresh parsley 2 tablespoons chopped pimento peppers

Instructions: Preheat the oven to 400 degrees F (200 degrees C.) Toss shredded cheese and flour together in a medium bowl. Sprinkle mixture onto frozen pie crust. Add diced ham on top of cheese. Combine eggs, milk, salt, and dry mustard in a medium bowl. Beat until smooth and pour over cheese and ham. Place foil around the edges of the crust to protect it from burning.

Grilled Chicken Caesar Salad
DIFFICULTY: EASY TIME: 25 Minutes

Veggie Stir-Fry with Tofu
DIFFICULTY: EASY TIME: 30 MINUTES

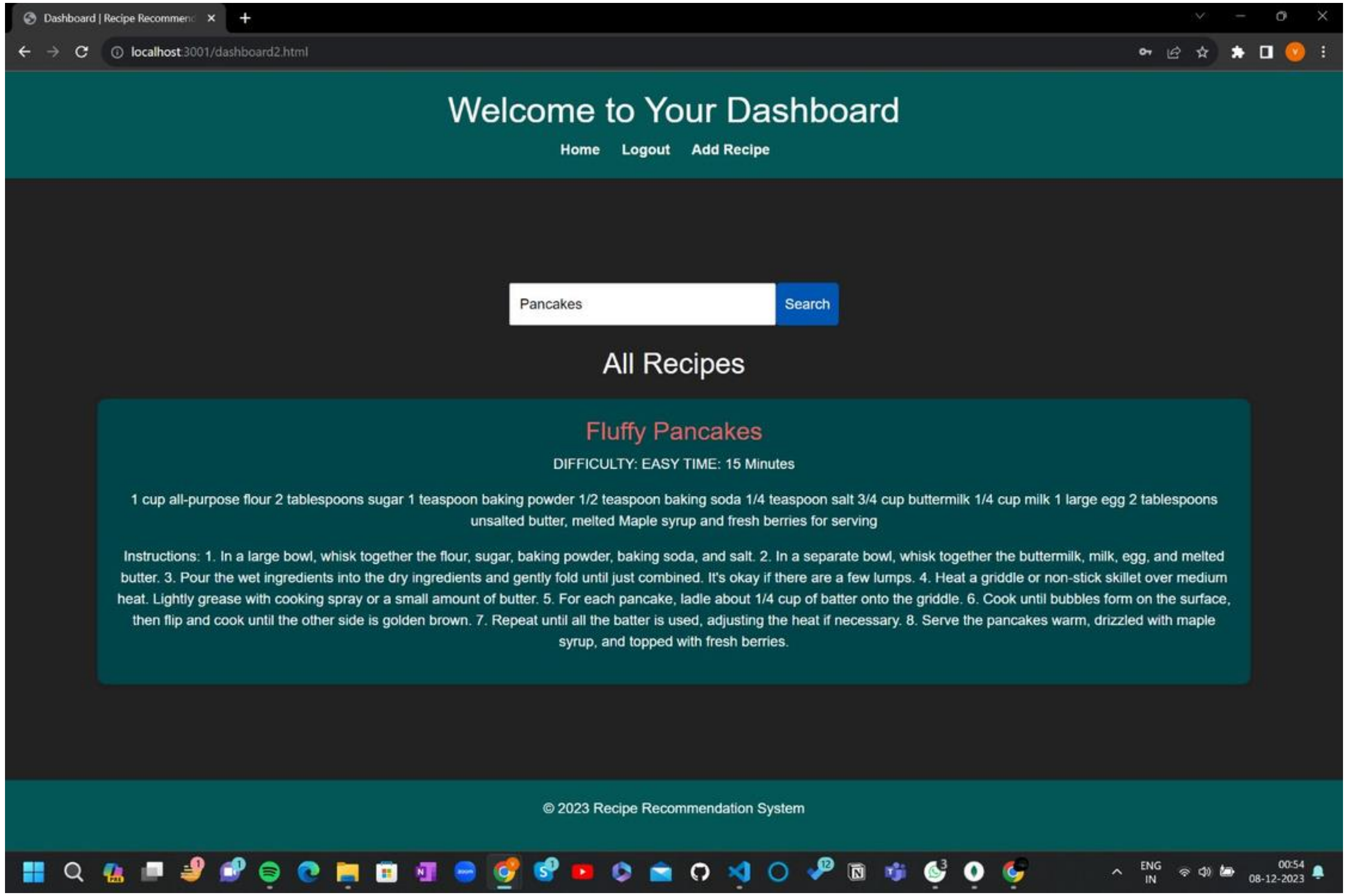
Spaghetti Aglio e Olio
DIFFICULTY: MEDIUM TIME: 45 MINUTES

Berry Smoothie Bowl
DIFFICULTY: EASY TIME: 10 MINUTES

© 2023 Recipe Recommendation System

This section boasts visually appealing displays, complemented by succinct titles and detailed descriptions. The user-friendly layout facilitates effortless navigation, providing users with a seamless and immersive experience as they explore the richness of our culinary community.

Users can also search for a specific recipe.



Welcome to Your Dashboard

All Recipes

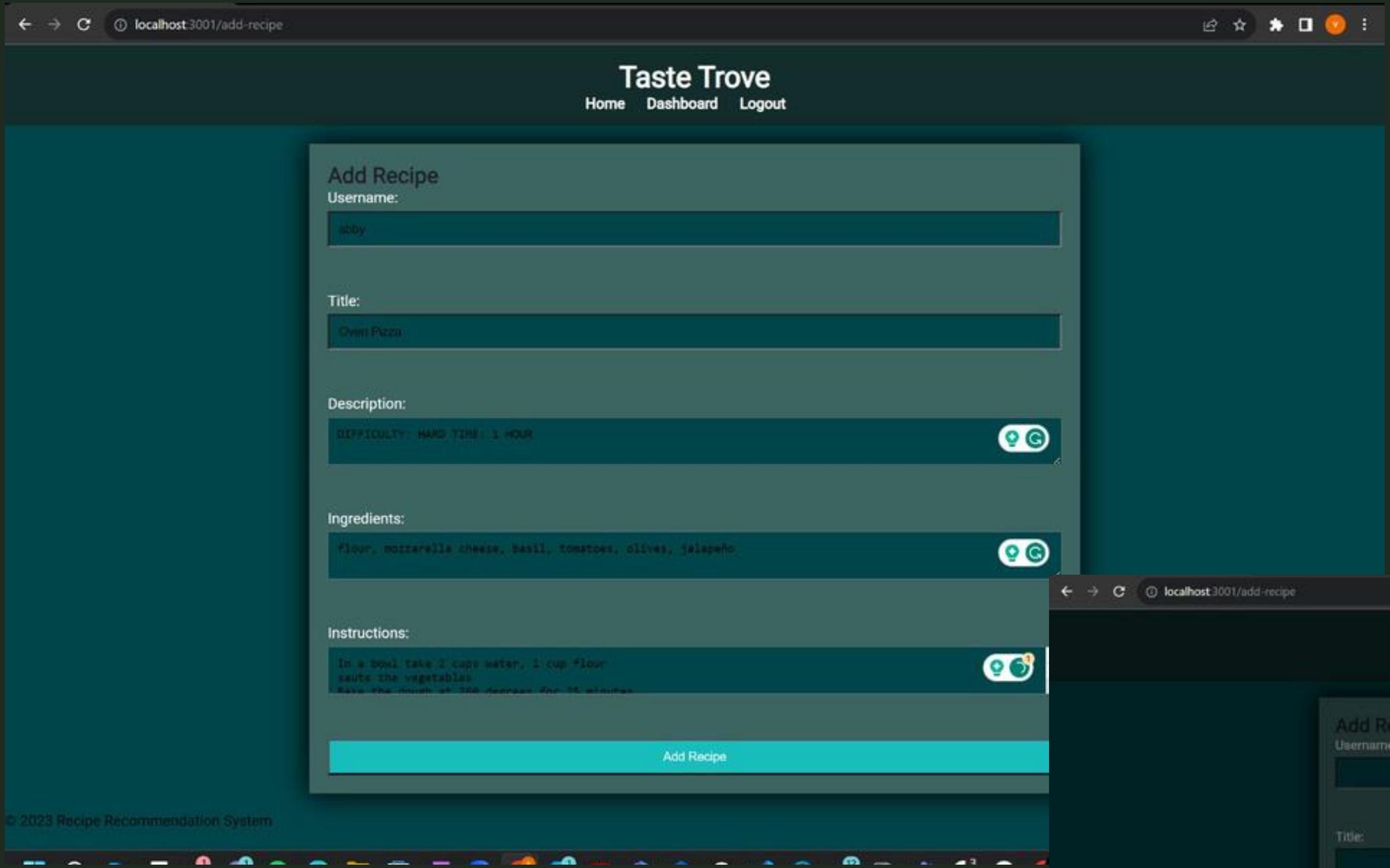
Fluffy Pancakes

DIFFICULTY: EASY TIME: 15 Minutes

1 cup all-purpose flour 2 tablespoons sugar 1 teaspoon baking powder 1/2 teaspoon baking soda 1/4 teaspoon salt 3/4 cup buttermilk 1/4 cup milk 1 large egg 2 tablespoons unsalted butter, melted Maple syrup and fresh berries for serving

Instructions: 1. In a large bowl, whisk together the flour, sugar, baking powder, baking soda, and salt. 2. In a separate bowl, whisk together the buttermilk, milk, egg, and melted butter. 3. Pour the wet ingredients into the dry ingredients and gently fold until just combined. It's okay if there are a few lumps. 4. Heat a griddle or non-stick skillet over medium heat. Lightly grease with cooking spray or a small amount of butter. 5. For each pancake, ladle about 1/4 cup of batter onto the griddle. 6. Cook until bubbles form on the surface, then flip and cook until the other side is golden brown. 7. Repeat until all the batter is used, adjusting the heat if necessary. 8. Serve the pancakes warm, drizzled with maple syrup, and topped with fresh berries.

© 2023 Recipe Recommendation System


localhost:3001/add-recipe
Taste Trove
Home
Dashboard
Logout

Add Recipe

Username: abby

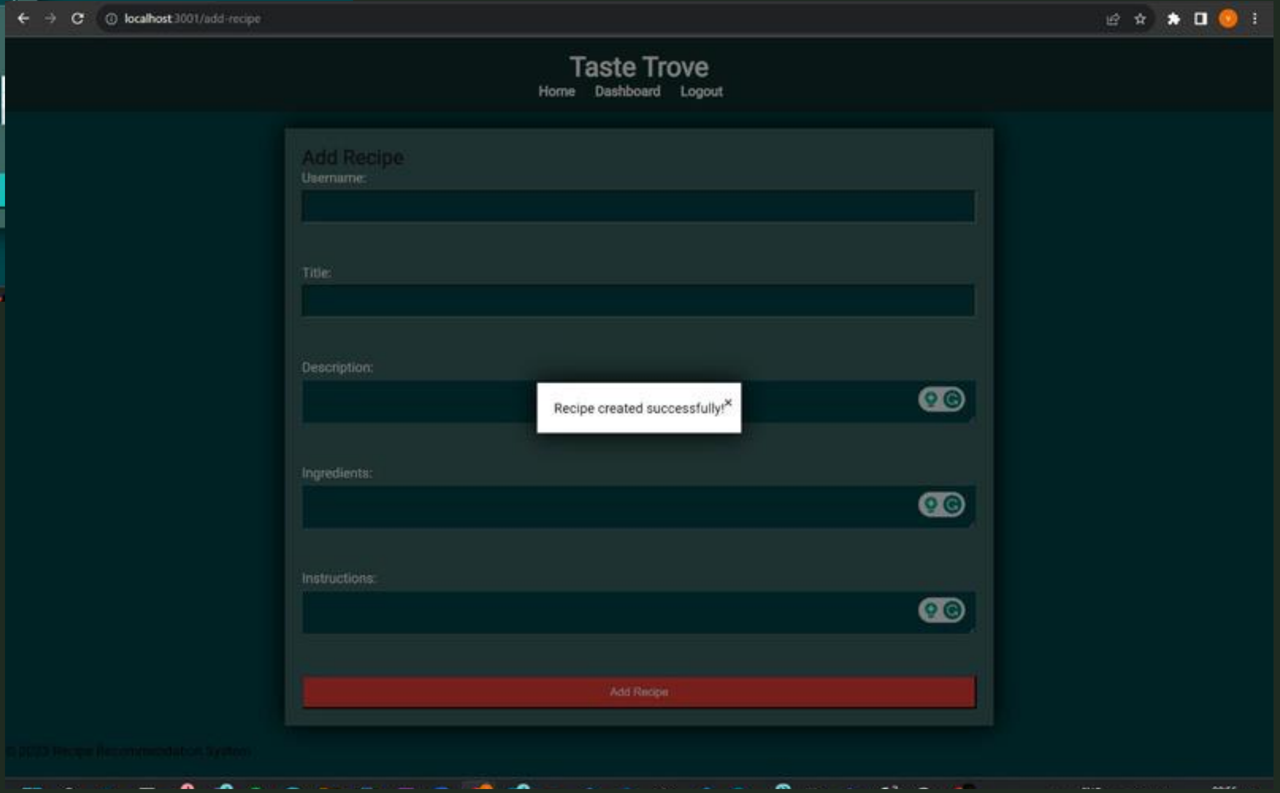
Title: Oven Pizza

Description: DIFFICULTY: HARD TIME: 1 HOUR

Ingredients: flour, mozzarella cheese, basil, tomatoes, olives, jalapeño

Instructions:
In a bowl take 2 cups water, 1 cup flour
saute the vegetables
Bake the dough at 360 degrees for 25 minutes

Add Recipe
© 2023 Recipe Recommendation System



Recipe created successfully!

Users can also add their own recipes.

Healthy, hearty meals await. See you soon!

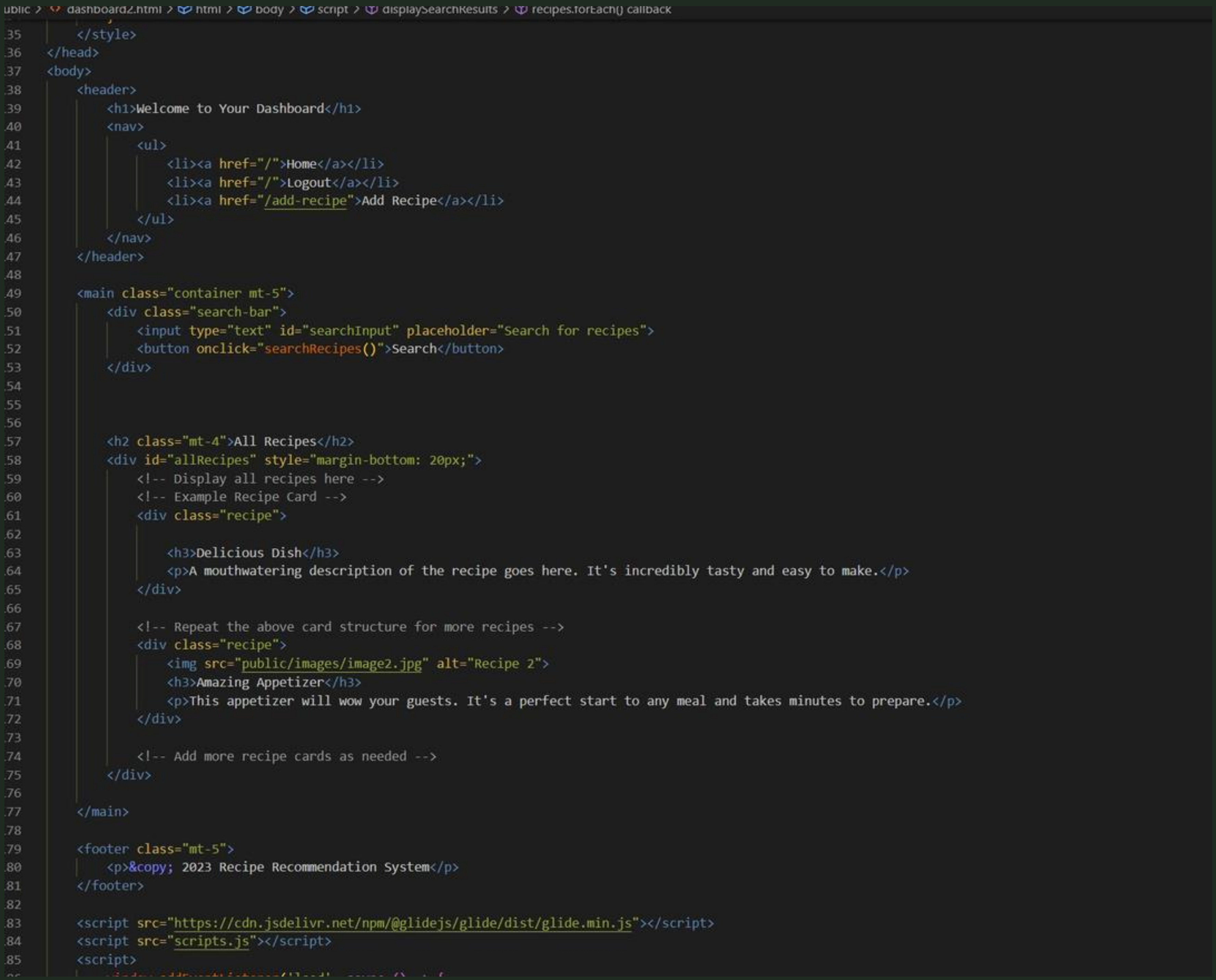
TECH STACK

HTML,CSS,JS,Bootsrap,Node,Express and MongoDB was used along with dependencies like cors,bycrypt and mongoose to implement the website.

It has been shown in the following slides

HTML

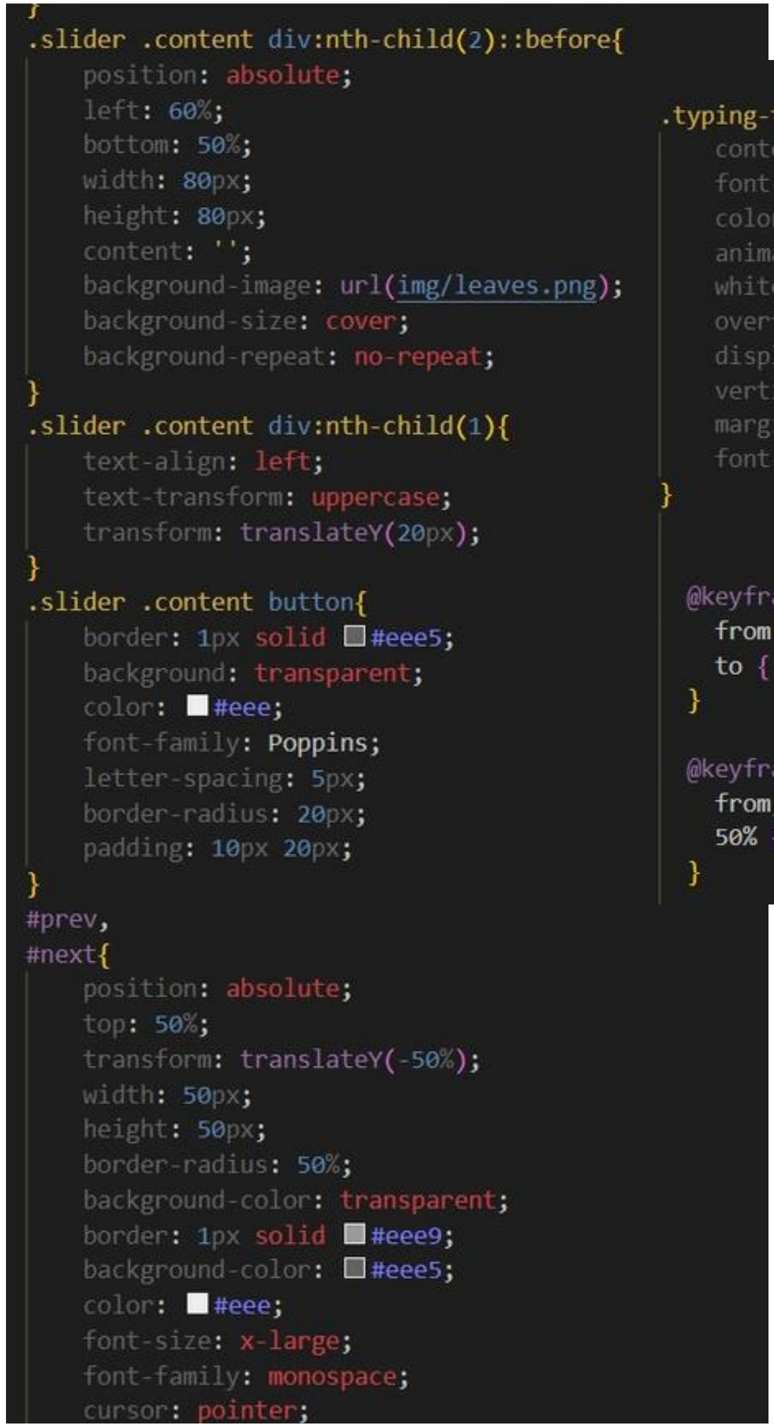
In the development of our Recipe Recommendation System homepage, HyperText Markup Language (HTML) served as the foundational structure, defining the content and layout of the web page. We utilized HTML to create a structured and semantically meaningful document, incorporating essential elements such as headers, images, navigation links, and forms. HTML's role in organizing content and establishing a hierarchical structure facilitated seamless integration with CSS for styling and JavaScript for dynamic functionalities. This hands-on experience reinforced our understanding of HTML's significance in web development, emphasizing the importance of clean, well-organized markup for creating accessible, search engine-friendly, and structurally sound web pages in our Recipe Recommendation System.


ublic > dashboard2.html > html > body > script > displaySearchResults > recipes.forEach() callback

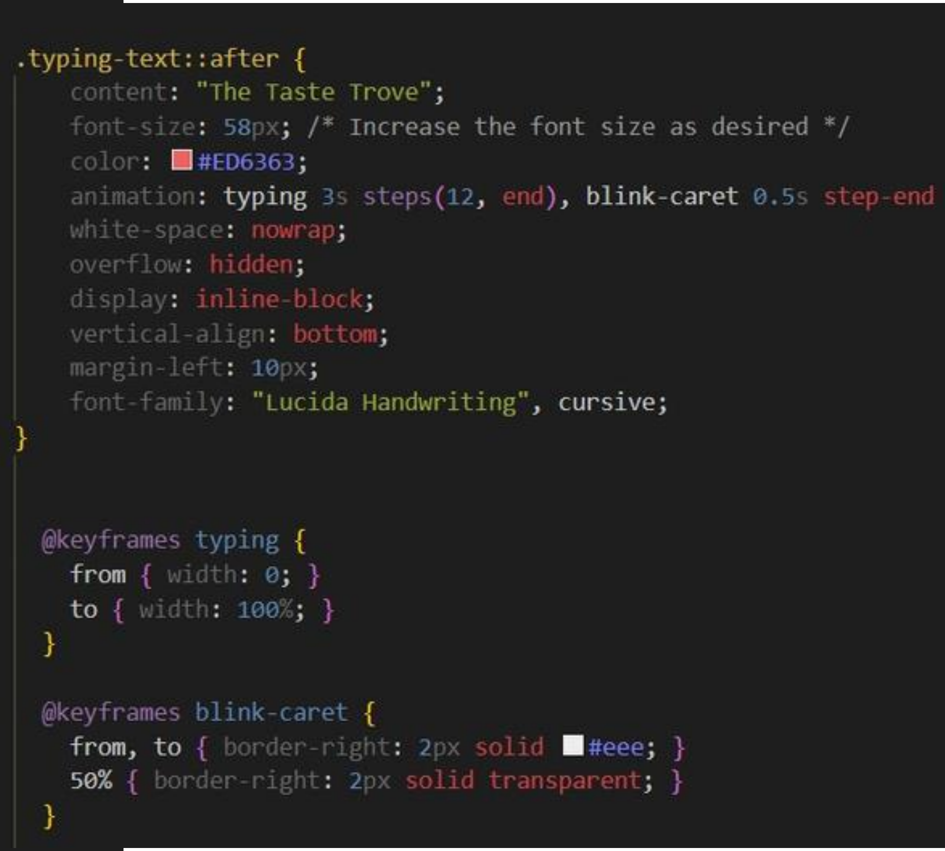
```html
    </style>
</head>
<body>
    <header>
        <h1>Welcome to Your Dashboard</h1>
        <nav>
            <ul>
                <li><a href="/">Home</a></li>
                <li><a href="/">Logout</a></li>
                <li><a href="/add-recipe">Add Recipe</a></li>
            </ul>
        </nav>
    </header>

    <main class="container mt-5">
        <div class="search-bar">
            <input type="text" id="searchInput" placeholder="Search for recipes">
            <button onclick="searchRecipes()">Search</button>
        </div>



        <h2 class="mt-4">All Recipes</h2>
        <div id="allRecipes" style="margin-bottom: 20px;">
            <!-- Display all recipes here -->
            <!-- Example Recipe Card -->
            <div class="recipe">

                <h3>Delicious Dish</h3>
                <p>A mouthwatering description of the recipe goes here. It's incredibly tasty and easy to make.</p>
            </div>

            <!-- Repeat the above card structure for more recipes -->
            <div class="recipe">
                <img src="public/images/image2.jpg" alt="Recipe 2">
                <h3>Amazing Appetizer</h3>
                <p>This appetizer will wow your guests. It's a perfect start to any meal and takes minutes to prepare.</p>
            </div>

            <!-- Add more recipe cards as needed -->
        </div>

    </main>

    <footer class="mt-5">
        <p>&copy; 2023 Recipe Recommendation System</p>
    </footer>

    <script src="https://cdn.jsdelivr.net/npm/@glidejs/glide/dist/glide.min.js"></script>
    <script src="scripts.js"></script>
    <script>
```

CSS & Bootstrap

Cascading Style Sheets (CSS) played a crucial role in defining the visual aspects and layout of the web page. We leveraged CSS to ensure a consistent and visually appealing design, implementing styles that aligned with our branding and provided a cohesive user experience. From defining color schemes, typography, to structuring the layout and responsiveness, CSS empowered us to transform HTML elements into an aesthetically pleasing and user-friendly interface. Through this project, we deepened our understanding of CSS principles, including selectors, properties, and responsive design techniques, allowing us to create a visually compelling and well-styled presentation for our Recipe Recommendation System.



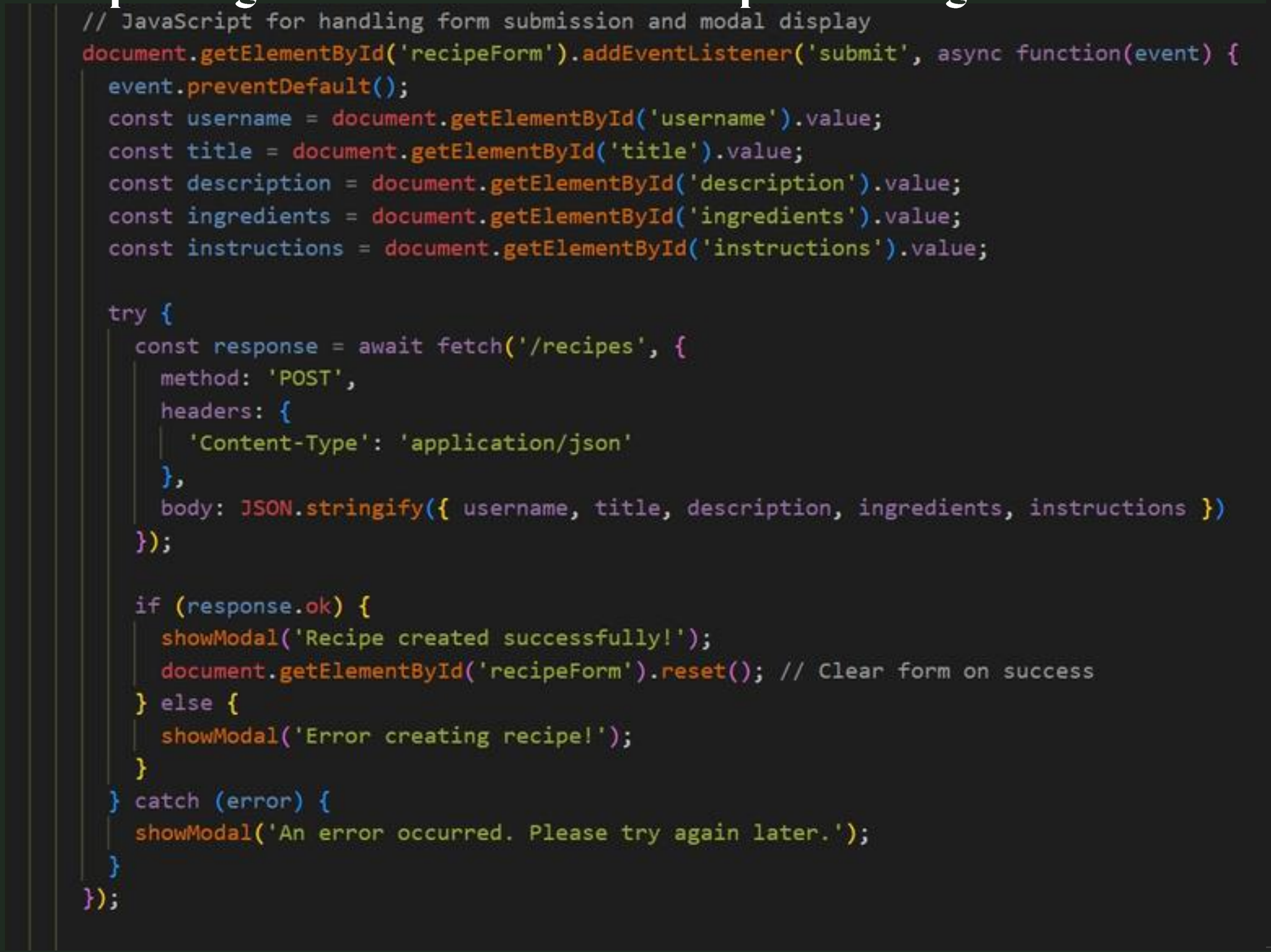
```css
.slider .content div:nth-child(2)::before{
    position: absolute;
    left: 60%;
    bottom: 50%;
    width: 80px;
    height: 80px;
    content: '';
    background-image: url(img/leaves.png);
    background-size: cover;
    background-repeat: no-repeat;
}
.slider .content div:nth-child(1){
    text-align: left;
    text-transform: uppercase;
    transform: translateY(20px);
}
.slider .content button{
    border: 1px solid #eee5;
    background: transparent;
    color: #eee;
    font-family: Poppins;
    letter-spacing: 5px;
    border-radius: 20px;
    padding: 10px 20px;
}
#prev,
#next{
    position: absolute;
    top: 50%;
    transform: translateY(-50%);
    width: 50px;
    height: 50px;
    border-radius: 50%;
    background-color: transparent;
    border: 1px solid #eee9;
    background-color: #eee5;
    color: #eee;
    font-size: x-large;
    font-family: monospace;
    cursor: pointer;
```



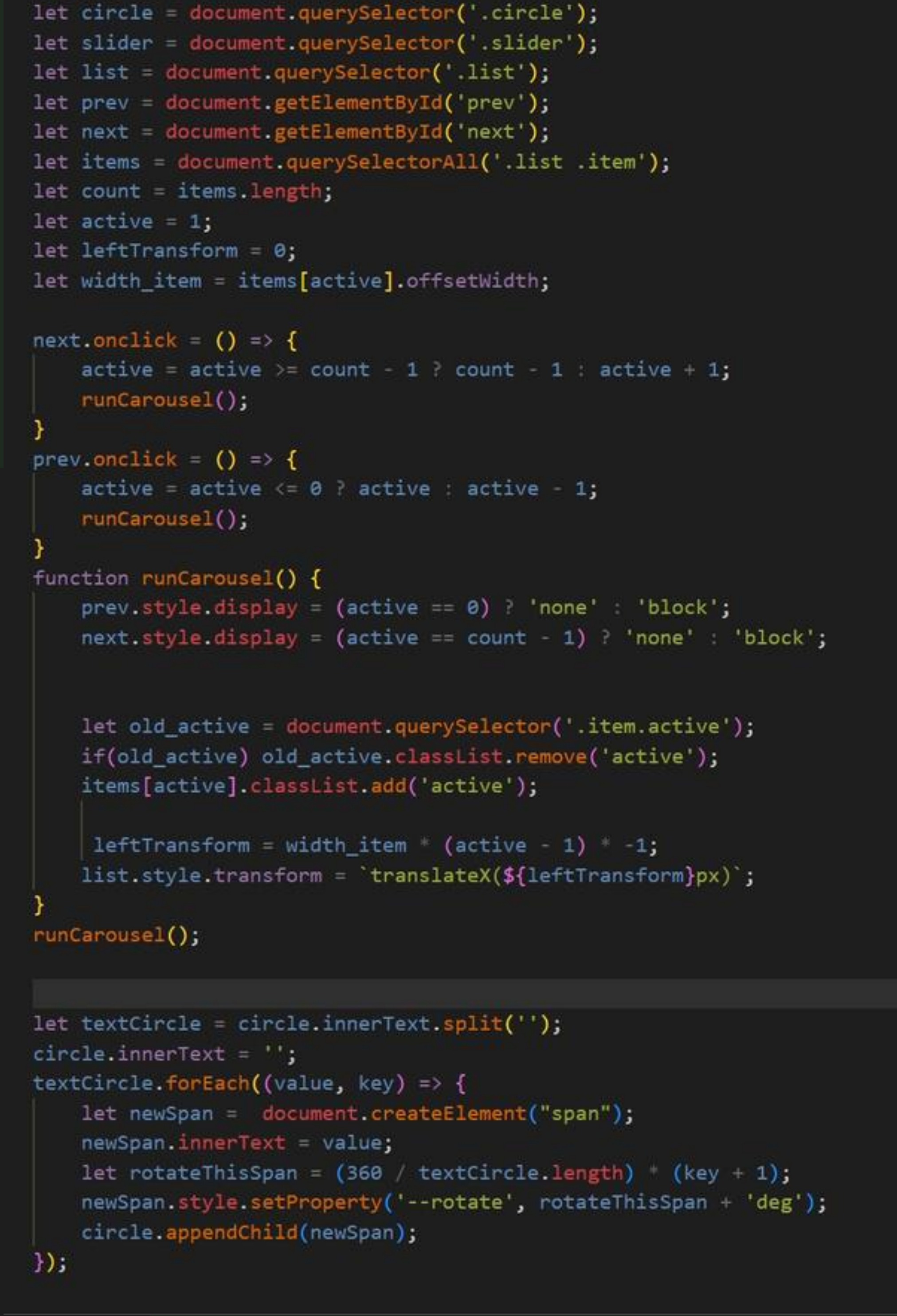
```css
.typing-text::after {
    content: "The Taste Trove";
    font-size: 58px; /* Increase the font size as desired */
    color: #ED6363;
    animation: typing 3s steps(12, end), blink-caret 0.5s step-end
    white-space: nowrap;
    overflow: hidden;
    display: inline-block;
    vertical-align: bottom;
    margin-left: 10px;
    font-family: "Lucida Handwriting", cursive;
}


  @keyframes typing {
    from { width: 0; }
    to { width: 100%; }
  }

  @keyframes blink-caret {
    from, to { border-right: 2px solid #eee; }
    50% { border-right: 2px solid transparent; }
  }
```

JS

JavaScript played a pivotal role in enhancing user interaction and providing dynamic content. Specifically, we utilized JavaScript to create an engaging image slider, allowing users to navigate through featured recipes seamlessly. The implementation of event listeners for next and previous buttons facilitated a smooth, visually appealing transition between recipe images. Through this project, we gained hands-on experience in leveraging JavaScript for client-side scripting, enabling us to enhance the user experience by incorporating interactive elements and responsive design.



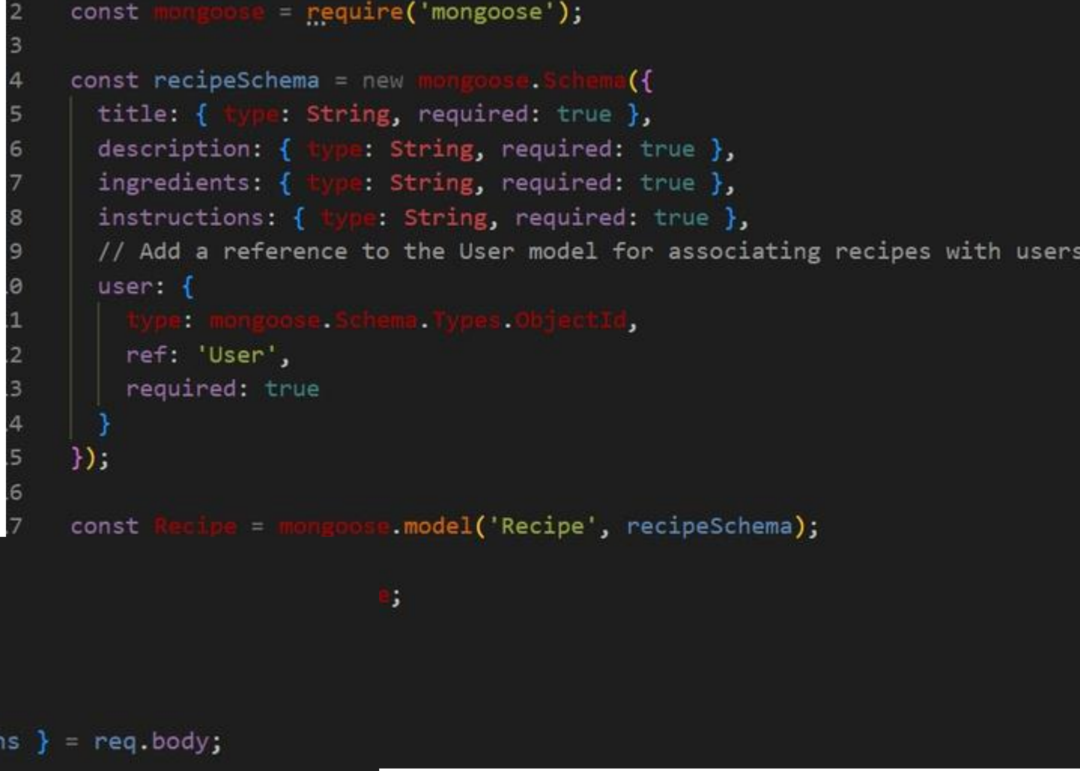
```javascript
// JavaScript for handling form submission and modal display
document.getElementById('recipeForm').addEventListener('submit', async function(event) {
  event.preventDefault();
  const username = document.getElementById('username').value;
  const title = document.getElementById('title').value;
  const description = document.getElementById('description').value;
  const ingredients = document.getElementById('ingredients').value;
  const instructions = document.getElementById('instructions').value;

  try {
    const response = await fetch('/recipes', {
      method: 'POST',
      headers: {
        'Content-Type': 'application/json'
      },
      body: JSON.stringify({ username, title, description, ingredients, instructions })
    });

    if (response.ok) {
      showModal('Recipe created successfully!');
      document.getElementById('recipeForm').reset(); // Clear form on success
    } else {
      showModal('Error creating recipe!');
    }
  } catch (error) {
    showModal('An error occurred. Please try again later.');
  }
});
```



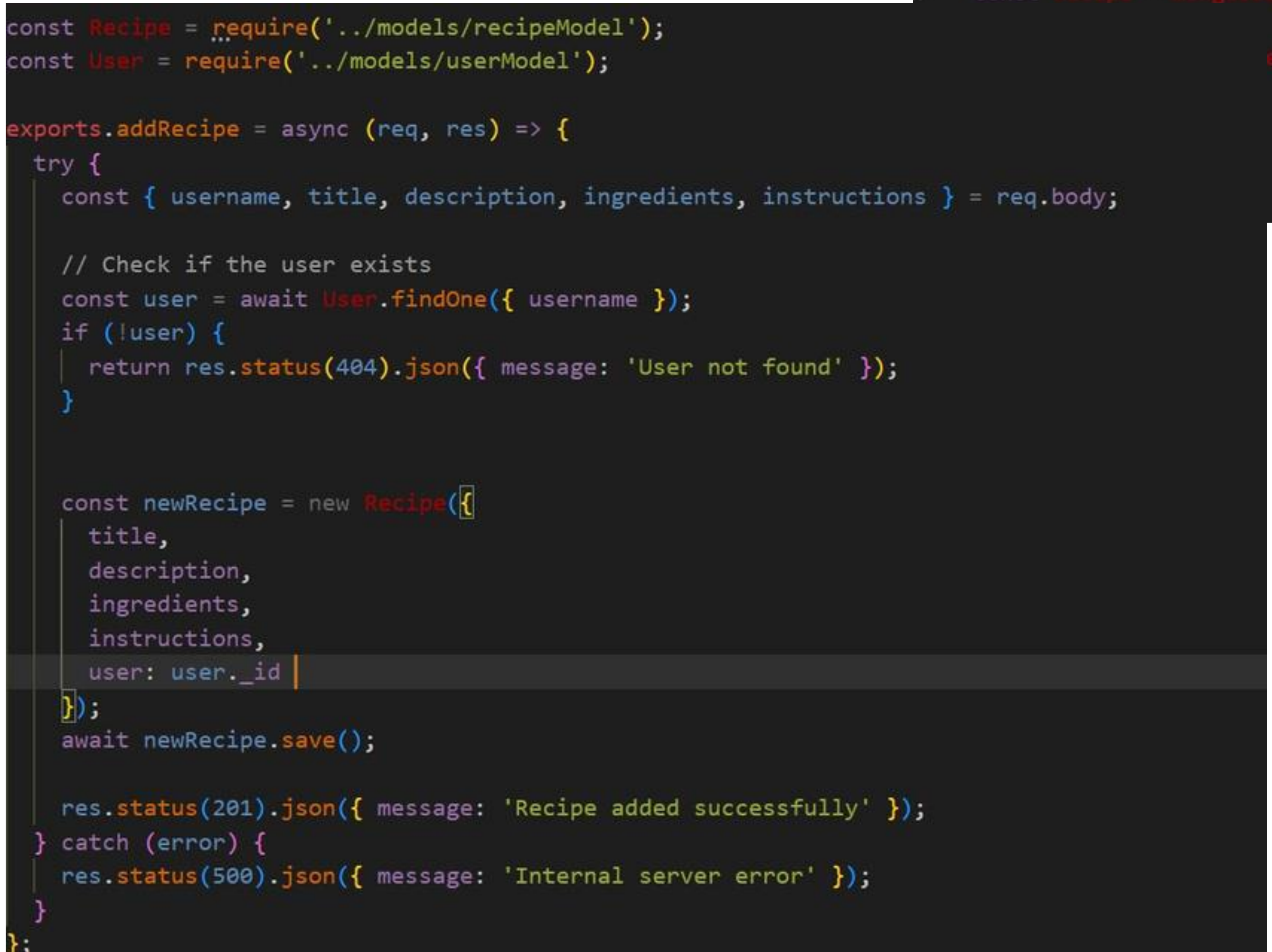
```javascript
let circle = document.querySelector('.circle');
let slider = document.querySelector('.slider');
let list = document.querySelector('.list');
let prev = document.getElementById('prev');
let next = document.getElementById('next');
let items = document.querySelectorAll('.list .item');
let count = items.length;
let active = 1;
let leftTransform = 0;
let width_item = items[active].offsetWidth;

next.onclick = () => {
    active = active >= count - 1 ? count - 1 : active + 1;
    runCarousel();
}
prev.onclick = () => {
    active = active <= 0 ? active : active - 1;
    runCarousel();
}
function runCarousel() {
    prev.style.display = (active == 0) ? 'none' : 'block';
    next.style.display = (active == count - 1) ? 'none' : 'block';


    let old_active = document.querySelector('.item.active');
    if(old_active) old_active.classList.remove('active');
    items[active].classList.add('active');

     leftTransform = width_item * (active - 1) * -1;
    list.style.transform = `translateX(${leftTransform}px)`;
}
runCarousel();


let textCircle = circle.innerText.split('');
circle.innerText = '';
textCircle.forEach((value, key) => {
    let newSpan =  document.createElement("span");
    newSpan.innerText = value;
    let rotateThisSpan = (360 / textCircle.length) * (key + 1);
    newSpan.style.setProperty('--rotate', rotateThisSpan + 'deg');
    circle.appendChild(newSpan);
});
```

MongoDB

In implementing our signup function, MongoDB serves as the reliable backend database, seamlessly storing and managing user credentials. Leveraging the flexibility of MongoDB's NoSQL structure, we efficiently store and retrieve user data, ensuring scalability and responsiveness.

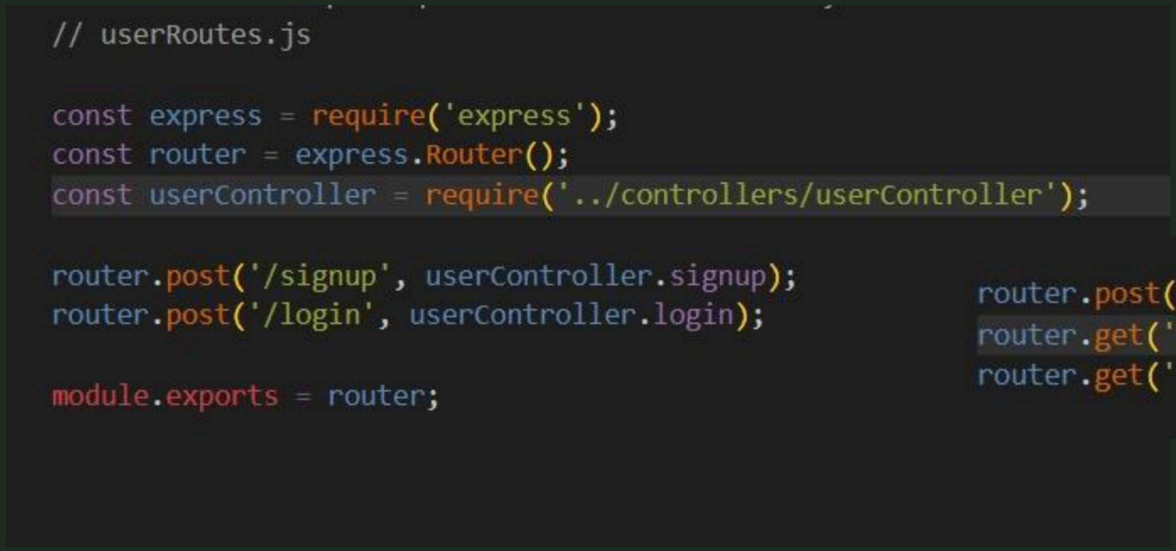


```
const mongoose = require('mongoose');

const recipeSchema = new mongoose.Schema({
  title: { type: String, required: true },
  description: { type: String, required: true },
  ingredients: { type: String, required: true },
  instructions: { type: String, required: true },
  // Add a reference to the User model for associating recipes with users
  user: {
    type: mongoose.Schema.Types.ObjectId,
    ref: 'User',
    required: true
  }
});

const Recipe = mongoose.model('Recipe', recipeSchema);

                                e;
```

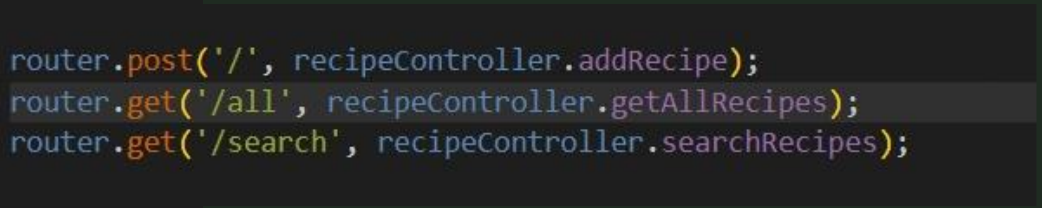


```
const Recipe = require('../models/recipeModel');
const User = require('../models/userModel');

exports.addRecipe = async (req, res) => {
  try {
    const { username, title, description, ingredients, instructions } = req.body;

    // Check if the user exists
    const user = await User.findOne({ username });
    if (!user) {
      return res.status(404).json({ message: 'User not found' });
    }


    const newRecipe = new Recipe({
      title,
      description,
      ingredients,
      instructions,
      user: user._id
    });
    await newRecipe.save();

    res.status(201).json({ message: 'Recipe added successfully' });
  } catch (error) {
    res.status(500).json({ message: 'Internal server error' });
  }
};
```

Routes

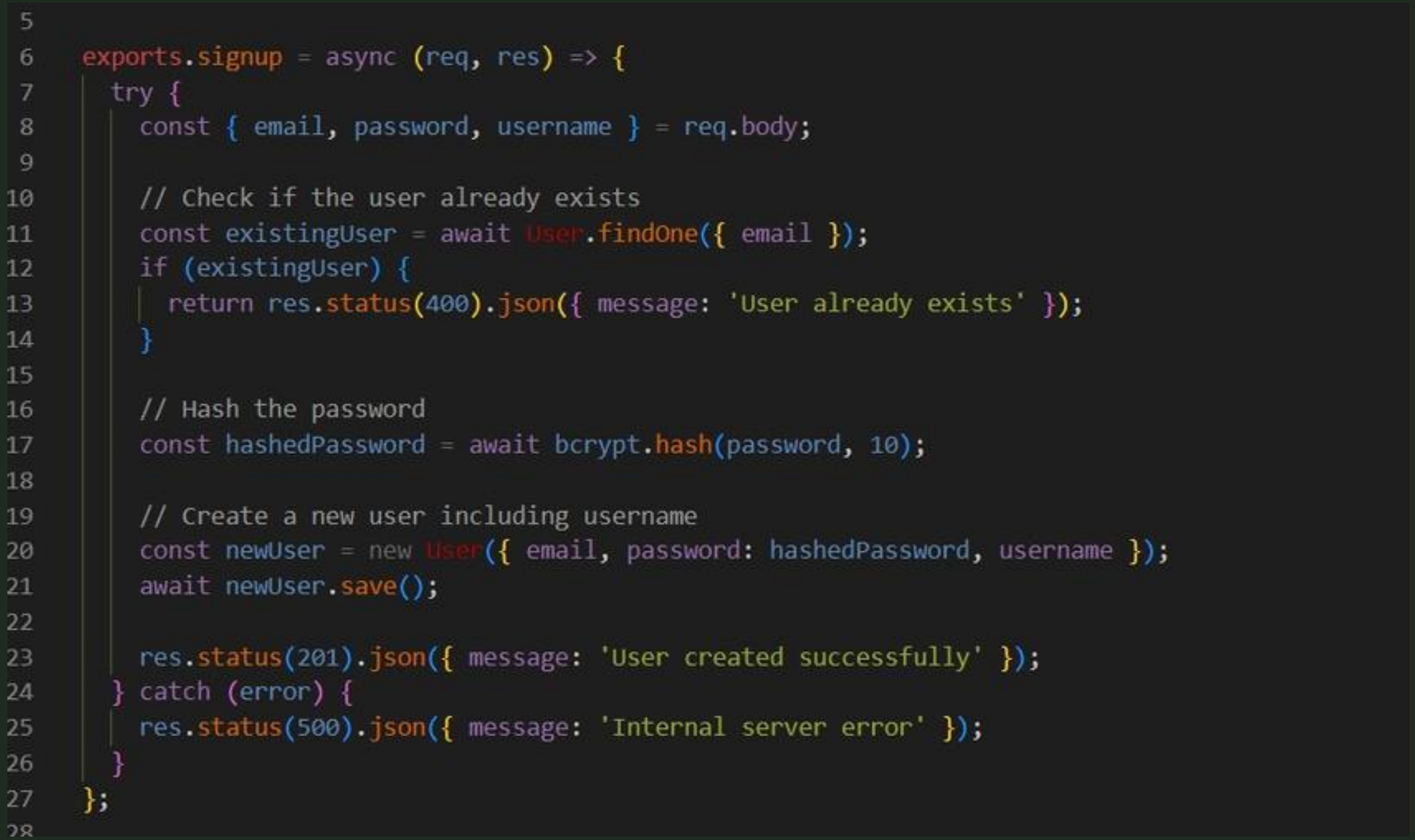


```
// userRoutes.js

const express = require('express');
const router = express.Router();
const userController = require('../controllers/userController');

router.post('/signup', userController.signup);
router.post('/login', userController.login);

module.exports = router;
```



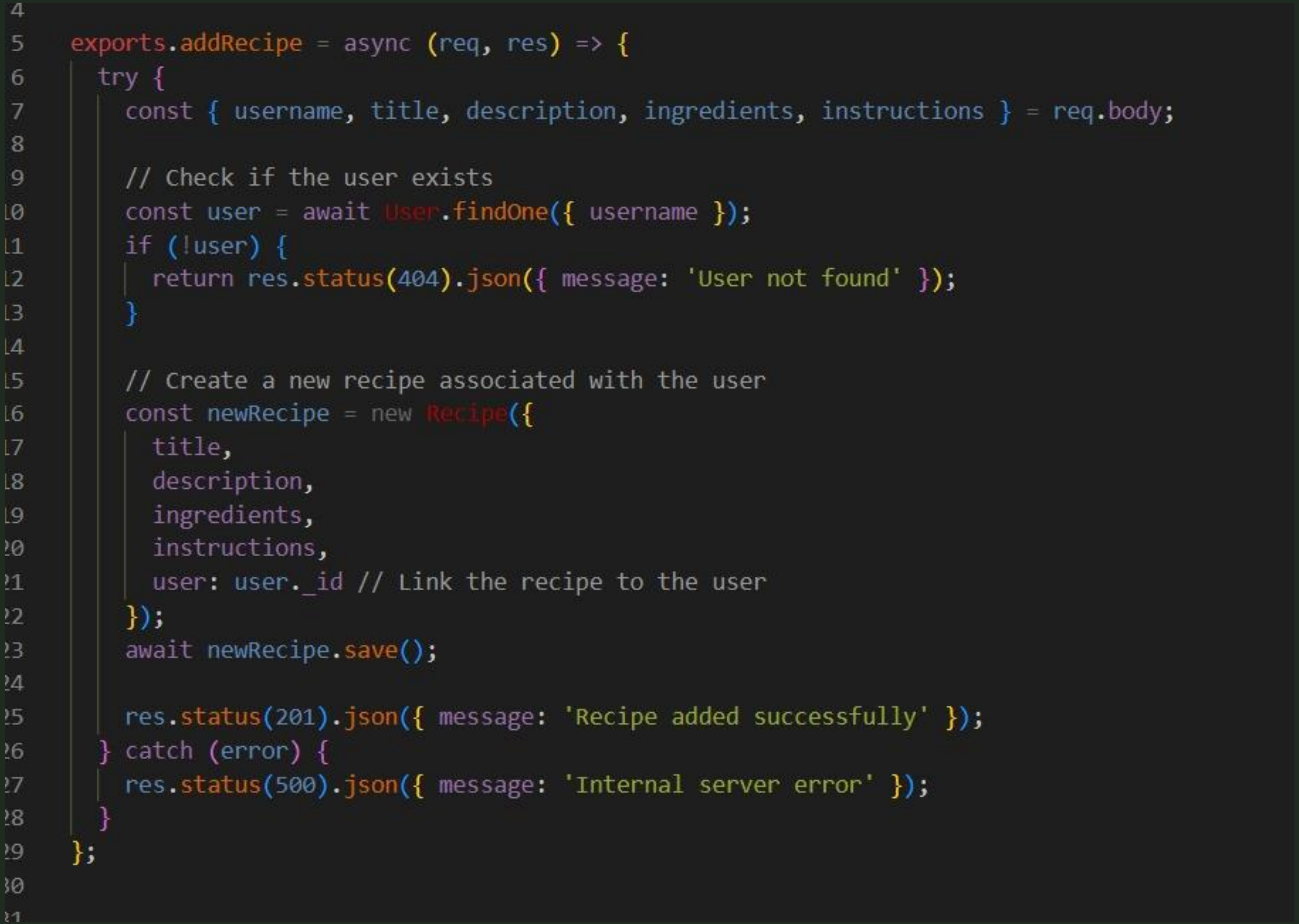
```
router.post('/', recipeController.addRecipe);
router.get('/all', recipeController.getAllRecipes);
router.get('/search', recipeController.searchRecipes);
```

Signup Using bycrypt Express



```

exports.signup = async (req, res) => {
  try {
    const { email, password, username } = req.body;

    // Check if the user already exists
    const existingUser = await User.findOne({ email });
    if (existingUser) {
      return res.status(400).json({ message: 'User already exists' });
    }

    // Hash the password
    const hashedPassword = await bcrypt.hash(password, 10);

    // Create a new user including username
    const newUser = new User({ email, password: hashedPassword, username });
    await newUser.save();

    res.status(201).json({ message: 'User created successfully' });
  } catch (error) {
    res.status(500).json({ message: 'Internal server error' });
  }
};
```

Adding Recipe



```javascript

exports.addRecipe = async (req, res) => {
  try {
    const { username, title, description, ingredients, instructions } = req.body;

    // Check if the user exists
    const user = await User.findOne({ username });
    if (!user) {
      return res.status(404).json({ message: 'User not found' });
    }

    // Create a new recipe associated with the user
    const newRecipe = new Recipe({
      title,
      description,
      ingredients,
      instructions,
      user: user._id // Link the recipe to the user
    });
    await newRecipe.save();

    res.status(201).json({ message: 'Recipe added successfully' });
  } catch (error) {
    res.status(500).json({ message: 'Internal server error' });
  }
};

```