Recipe-Book

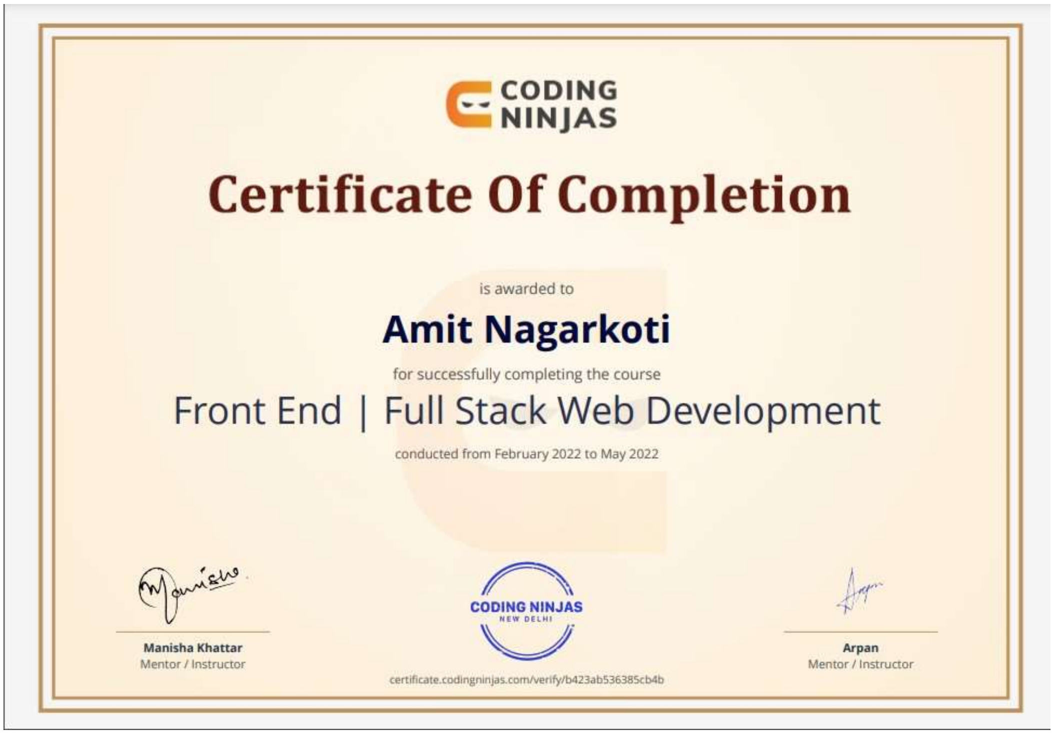
Final year Project report

B.Tech (Computer Science Engineering)

Submitted by:

Amit Nagarkoti
1810048

(Department of Computer Engineering)
SUSCET , TANGORI



CODING NINJAS

Certificate Of Completion

is awarded to

Amit Nagarkoti

for successfully completing the course

Front End | Full Stack Web Development

conducted from February 2022 to May 2022

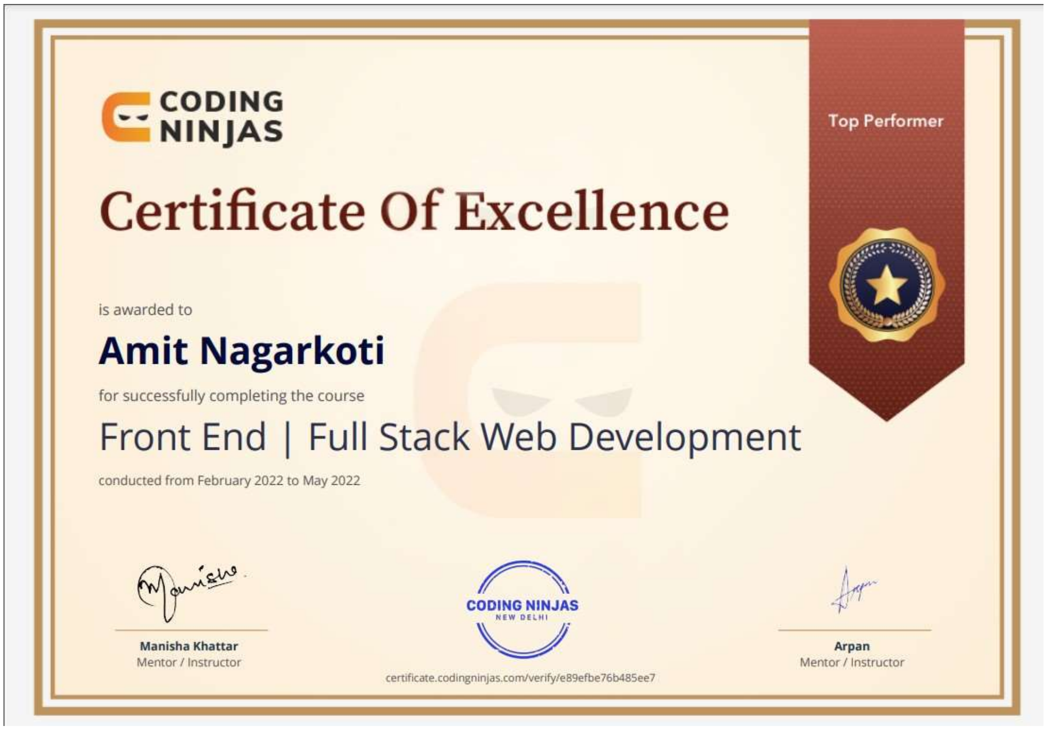
Manisha Khattar
Mentor / Instructor

CODING NINJAS
NEW DELHI

Arpan
Mentor / Instructor

certificate.codingninjas.com/verify/b423ab536385cb4b

Verify:https://certificate.codingninjas.com/verify/b423ab536385cb4b



CODING NINJAS

Top Performer

Certificate Of Excellence

is awarded to

Amit Nagarkoti

for successfully completing the course

Front End | Full Stack Web Development

conducted from February 2022 to May 2022

Manisha Khattar
Mentor / Instructor

CODING NINJAS
NEW DELHI

Arpan
Mentor / Instructor

certificate.codingninjas.com/verify/e89efbe76b485ee7

https://certificate.codingninjas.com/verify/e89efbe76b485ee7

CANDIDATE'S DECLARATION

I hereby certify that the work which is being presented in the project report entitled "**Recipe-Book**"by "**Amit Nagarkoti**" in partial fulfillment of requirements for the award of degree of Bachelors in Technology in Computer Science Engineering submitted in the Faculty of Engineering, at "**SUSCET**", Tangori is an authentic record of my ownwork carried out during a period from February, 2022 to May, 2022.

Amit Nagarkoti

This is to certify that the above statement made by the candidate is correct to the best of my/ourknowledge.

Deepinder Kaur

ACKNOWLEDGEMENT

I am very thankful to **Arpan Garg**. for giving me the opportunity to undertake my PROJECT semester at their prestigious **Coding Ninjas Pvt. Ltd**. It was a very good learning experience for me to have worked at this project involved many new and unique practices and challenges. I would also like to give my heart-felt thanks to **Manisha Khattar(Instructor)** and **Arpan Gaarg** (**mentor)**, who guided and encouraged me all through the training and imparted in-depth knowledge of the project. , Head of Department of - Computer Science engineering, SUSCET who was my faculty adviser for being so helpful during training.

Name of Student **Signature of Student**

Amit Nagarkoti

INDEX

t

INTRODUCTION

The aim of this Cooking Recipe Portal project is to construct an online system by use of which a user can learn various kinds of the recipe. This system can be beneficial for a restaurant or to the instructor who teaches the cooking as they can spread their recipe to the other or to a common person sitting at home looking to try and eat something he/she normally doesn’t. The user can also post their recipe on the website. As the Cooking Recipe Portal system will be based on the internet, the user can learn the recipe from any place and can view the recipe any number of times. The users can give feedback to the recipe.

OBJECTIVE OF THE PROJECT

The Objective of the project is simple,

to make users upload their recipe and present it in the front of the world

And

To make users checkout different categories of recipes and try them

SCOPE OF THE SYSTEM: -

Any Food Cooking instructor can adapt different cuisines from the website and teach them to his/her students.

Any restaurant can checkout different recipes and can add them to their menu with asking permission to the person who has uploaded that recipe.

We can add new features as and when we require.

Reusability of this application is also possible.

Hardware and Software Requirement

To run this project successfully the minimum requirement is also kept in mind. Following are their requirements -

Hardware requirements: -

Processor: Pentium(R) Dual-Core CPU E5700 @ 3.00GHz

RAM 512MB and Above

HDD 20 GB Hard Disk Space and Above

Software requirements: -

Operating System: Windows XP, 7 or higher

Visual Studio

Node.js

Express Server

MongoDB database

Browser- Chrome

System Design

INTRODUCTION: - Software design sits at the technical kernel of the software engineering process and is applied regardless of the development paradigm and area of application. Design is the first step in the development phase for any engineered product or system. The designer's goal is to produce a model or representation of an entity that will later be built. Beginning, once system requirement has been specified and analyzed, system design is the first of the three technical activities -design, code and test that is required to build and verify software.

The importance can be stated with a single word "Quality". Design is the place where quality is fostered in software development. Design provides us with representations of software that can assess for quality. Design is the only way that we can accurately translate a customer's view into a finished software product or system. Software design serves as a foundation for all the software engineering steps that follow. Without a strong design we risk building an unstable system – one that will be difficult to test, one whose quality cannot be assessed until the last stage.

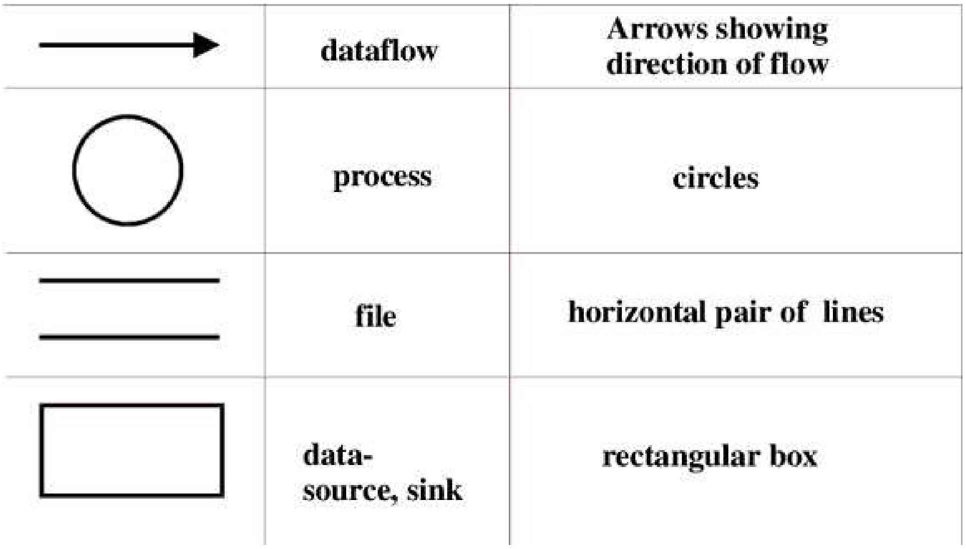
During design, progressive refinement of data structure, program structure, and procedural details are developed reviewed and documented. System design can be viewed from either technical or project management perspective. From the technical point of view, design is comprised of four activities – architectural design, data structure design, interface design and procedural design.

Data Flow Diagram of the Proposed System: -

A data flow diagram is a graphical view of how data is processed in a system term of input and output.

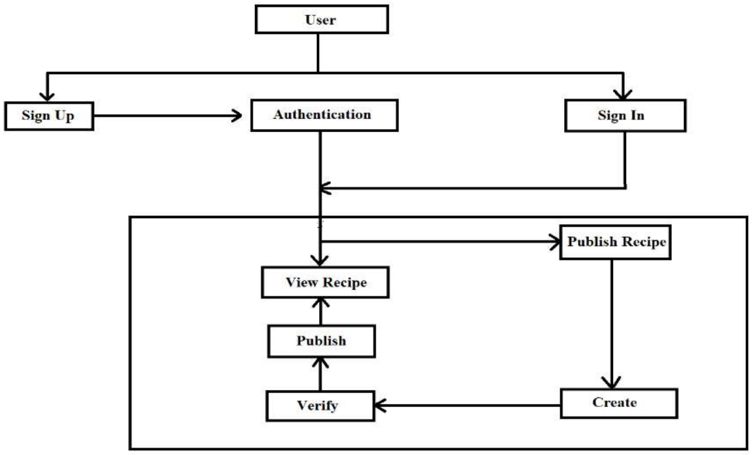
The data flow diagram (DFD) contains some symbol for drawing the data flow diagram.

Data flow diagram symbols: -



→	dataflow	Arrows showing direction of flow
○	process	circles
=	file	horizontal pair of lines
□	data-source, sink	rectangular box

DFD for the Users: -



Feasibility

Preliminary investigation examines project feasibility, the likelihood the system will be useful to the organization. The main objective of the feasibility study is to test the Technical, Operational and Economical feasibility for adding new modules and debugging old running system. All system is feasible if they are unlimited resources and infinite time. There are aspects in the feasibility study portion of the preliminary investigation:

- Technical Feasibility
- Operation Feasibility
- Economic Feasibility

Technical Feasibility

The technical issue usually raised during the feasibility stage of the investigation includes the following:

- Does the necessary technology exist to do what is suggested?
- Do the proposed equipment's have the technical capacity to hold the data required to use the new system?
- Will the proposed system provide adequate response to inquiries, regardless of the number or location of users?
- Can the system be upgraded if developed?
- Are there technical guarantees of accuracy, reliability, ease of access and data security?

Earlier no system existed to cater to the needs of 'Secure Infrastructure Implementation System'. The current system developed is technically feasible. It is a web-based user interface. Thus, it provides an easy access to the users. The database's purpose is to create, establish and maintain a workflow among various entities in order to facilitate all concerned users in their various capacities or roles. Permission to the users would be granted based on the roles specified. Therefore, it provides the technical guarantee of accuracy, reliability and security. The software and hard requirements for the development of this project are not many and are available as free as open source. The work for the project is done with the current equipment and existing software technology. Necessary bandwidth exists for providing fast feedback to the users irrespective of the number of users using the system.

Operational Feasibility

Proposed projects are beneficial only if they can be turned out into information system. That will meet the organization's operating requirements. Operational feasibility aspects of the project are to be taken as an important part of the project implementation. Some of the important issues raised are to test the operational feasibility of a project includes the following: -

- Is there sufficient support for the management from the users?

- Will the system be used and work properly if it is being developed and

Implemented?

- Will there be any resistance from the user that will undermine the possible

application benefits?

This system is targeted to be in accordance with the above-mentioned issues. Beforehand, the management issues and user requirements have been taken into consideration. So, there is no question of resistance from the users that can undermine the possible application benefits.

Economic Feasibility

A system can be developed technically and that will be used if installed must still be a good investment for the organization. In the economic feasibility, the development cost in creating the system is evaluated against the ultimate benefit derived from the new systems. Financial benefits must equal or exceed the costs.

The system is economically feasible. It does not require any addition hardware or software. Since the interface for this system is developed using the existing resources and technologies. There is nominal expenditure and economic feasibility for certain.

System Analysis

After analyzing the requirements of the task to be performed, the next step is to analyze the problem and understand its context. The first activity in the phase is studying the existing system and other is to understand the requirements and domain of the new system. Both the activities are equally important, but the first activity serves as a basis of giving the functional specifications and then successful design of the proposed system. Understanding the properties and requirements of a new system is more difficult and requires creative thinking and understanding of existing running system is also difficult, improper understanding of present system can lead diversion from solution.

STUDY OF THE SYSTEM

GUI'S

In the flexibility of the uses the interface has been developed a graphics concept in mind, associated through a browser interface. The GUI'S at the top level have been categorized as

- Administrative user interface
- The operational or generic user interface

The administrative user interface concentrates on the consistent information that is practically, part of the organizational activities and which needs proper authentication for the data collection. The interfaces help the administrations with all the transactional states like Data insertion, Data deletion and Date updating along with the extensive data search capabilities.

The operational or generic user interface helps the users upon the system in transactions through the existing data and required services. The operational user interface also helps the ordinary users in managing their own information helps the ordinary users in managing their own information in a customized manner as per the assisted flexibilities.

System Testing and Implementation

INTRODUCTION

Software testing is a critical element of software quality assurance and represents the ultimate review of specification, design and coding. In fact, testing is the one step in the software engineering process that could be viewed as destructive rather than constructive.

A strategy for software testing integrates software test case design methods into a well-planned series of steps that result in the successful construction of software. Testing is the set of activities that can be planned in advance and conducted systematically. The underlying motivation of program testing is to affirm software quality with methods that can economically and effectively apply to both strategic to both large and small-scale systems.

STRATEGIC APPROACH TO SOFTWARE TESTING

The software engineering process can be viewed as a spiral. Initially system engineering defines the role of software and leads to software requirement analysis where the information domain, functions, behavior, performance, constraints and validation criteria for software are established. Moving inward along the spiral, we come to design and finally to coding. To develop computer software, we spiral in along streamlines that decrease the level of abstraction on each turn.

A strategy for software testing may also be viewed in the context of the spiral. Unit testing begins at the vertex of the spiral and concentrates on each unit of the software as implemented in source code. Testing progress by moving outward along the spiral to integration testing, where the focus

is on the design and the construction of the software architecture. Talking another turn on outward on the spiral we encounter validation testing where requirements established as part of software requirements analysis are validated against the software that has been constructed. Finally, we arrive at system testing, where the software and other system elements are tested as a whole.

Unit Testing

Unit testing focuses verification effort on the smallest unit of software design, the module. The unit testing, we have is white box oriented and some modules the steps are conducted in parallel.

1. WHITE BOX TESTING

This type of testing ensures that

- All independent paths have been exercised at least once
- All logical decisions have been exercised on their true and false sides
- All loops are executed at their boundaries and within their operational bounds
- All internal data structures have been exercised to assure their validity.

To follow the concept of white box testing we have tested each form. we have created independently to verify that Data flow is correct, all conditions are exercised to check their validity, all loops are executed on their boundaries.

2. CONDITIONAL TESTING

In this part of the testing each of the conditions were tested to both true and false aspects. And all the resulting paths were tested. So that each path that may be generate on particular condition is traced to uncover any possible errors.

3. DATA FLOW TESTING

This type of testing selects the path of the program according to the location of definition and use of variables. This kind of testing was used only when some local variable was declared. The definition-use chain method was used in this type of testing. These were particularly useful in nested statements.

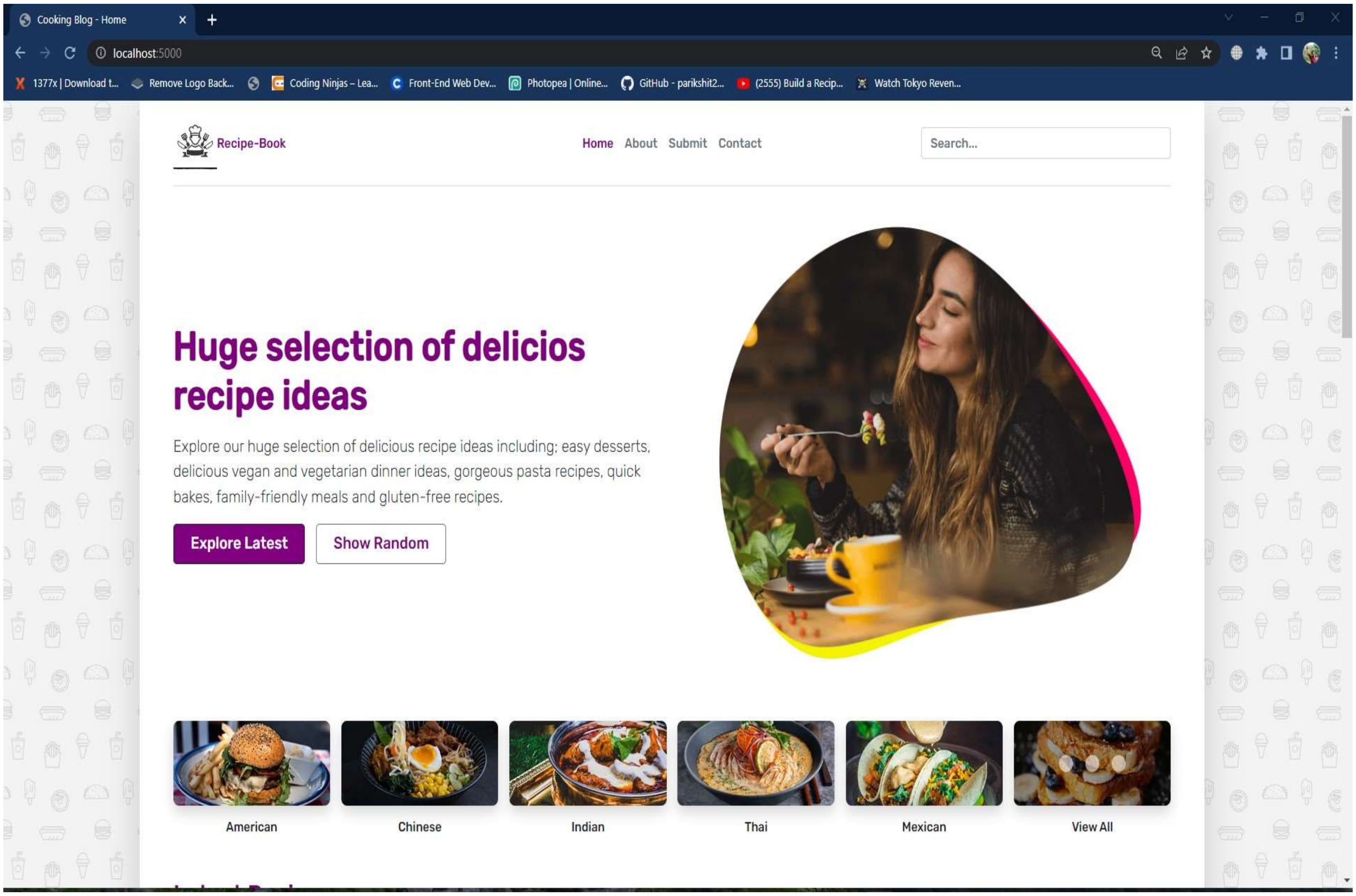
4. LOOP TESTING

In this type of testing all the loops are tested to all the limits possible. The following exercise was adopted for all loops:

- All the loops were tested at their limits, just above them and just below them.
- All the loops were skipped at least once.
- For nested loops test the inner most loop first and then work outwards.
- For concatenated loops the values of dependent loops were set with the help of connected loop.
- Unstructured loops were resolved into nested loops or concatenated loops and tested as above.

Each unit has been separately tested by the development team itself and all the input have been validated.

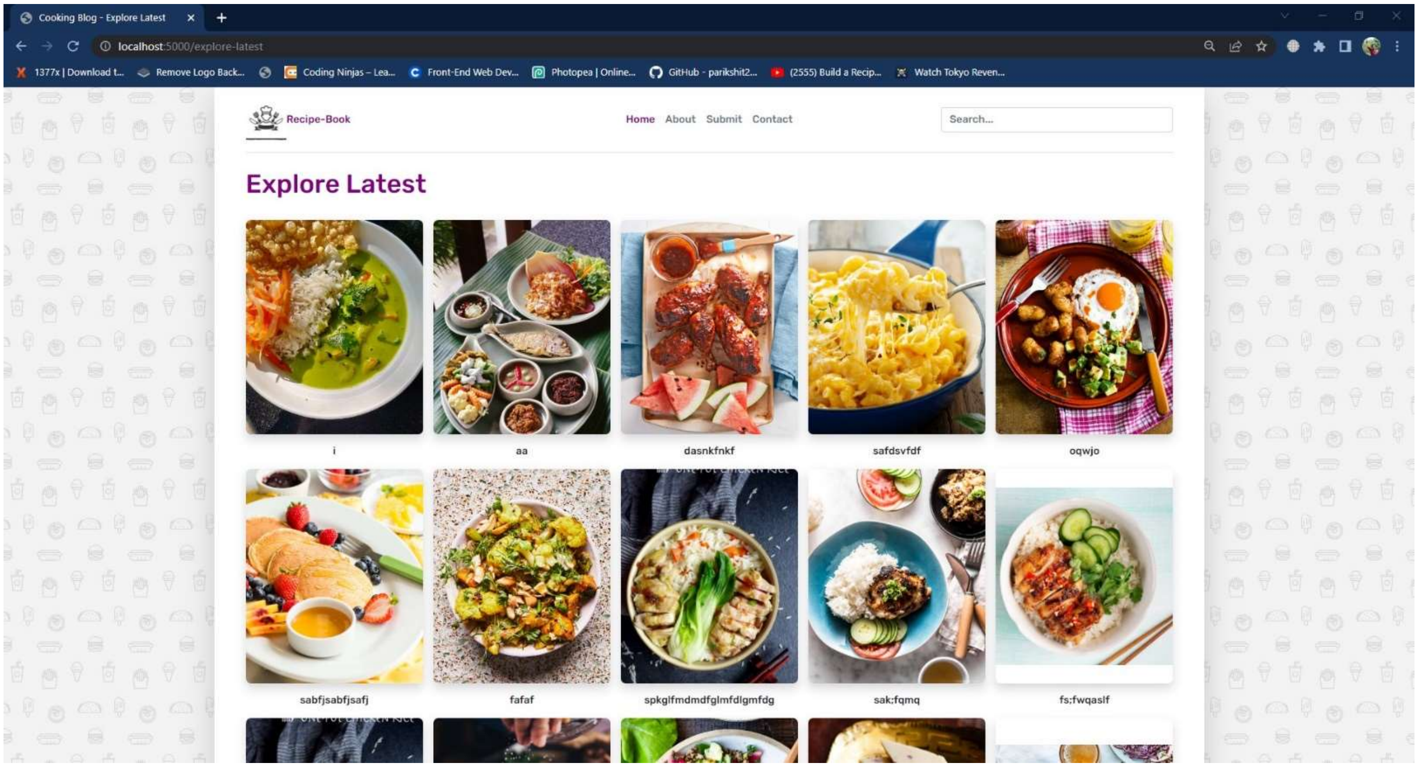
Screenshots of the Website: -



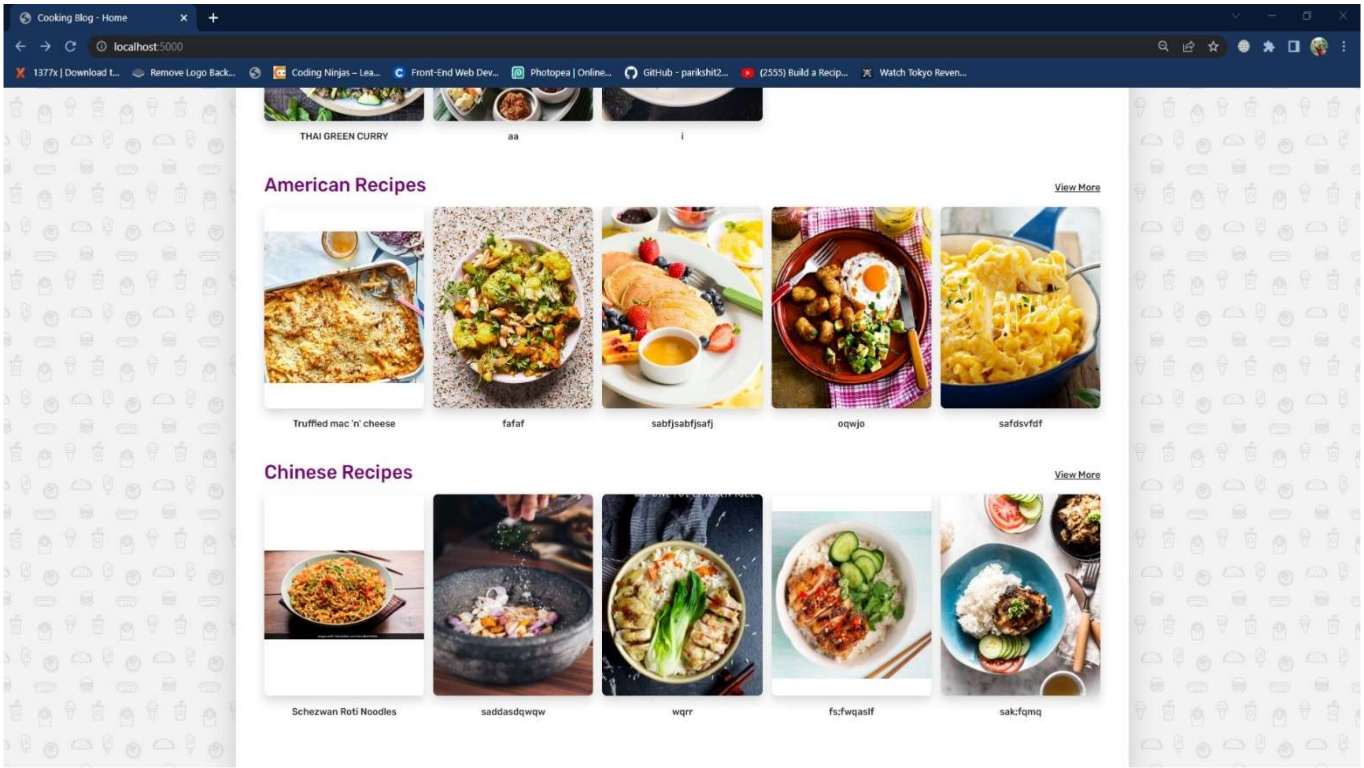
Huge selection of delicios recipe ideas

Explore our huge selection of delicious recipe ideas including; easy desserts, delicious vegan and vegetarian dinner ideas, gorgeous pasta recipes, quick bakes, family-friendly meals and gluten-free recipes.

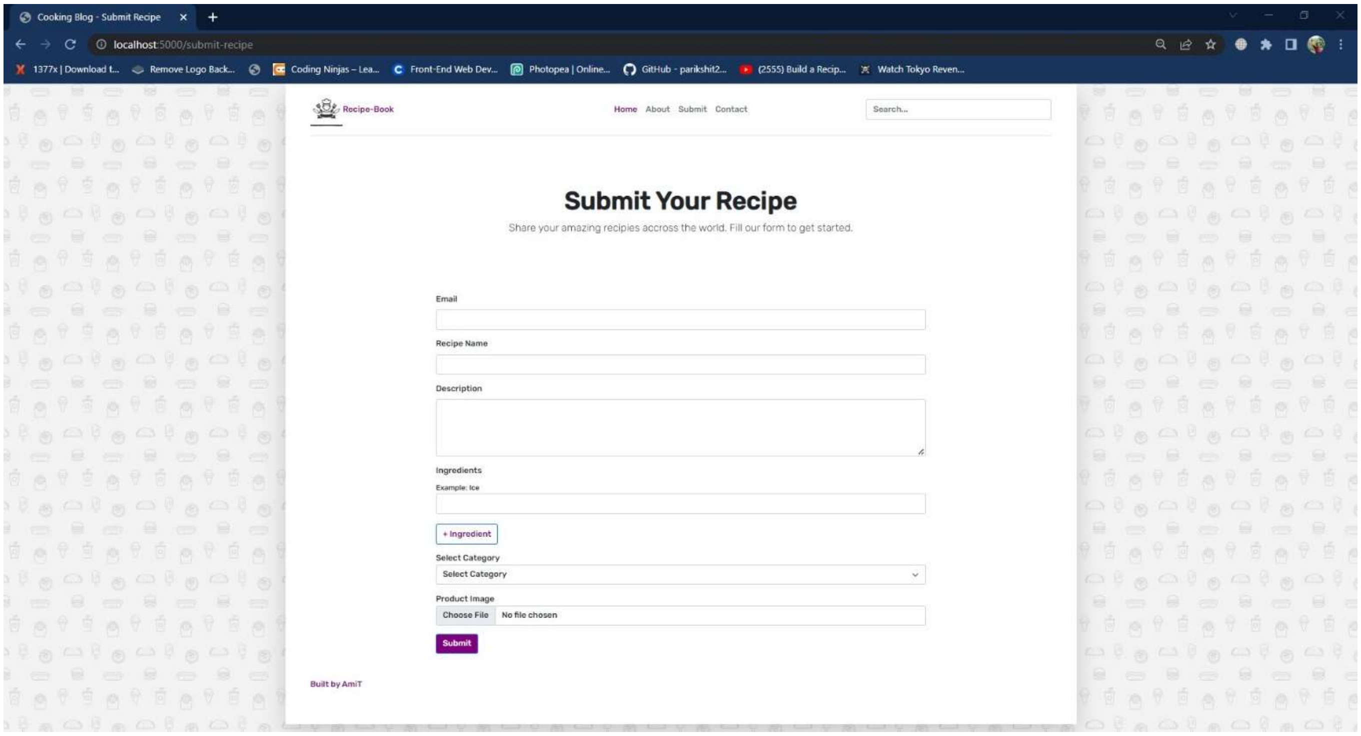
Homepage



Explore Page



Recipes by Categories



Submit Your Recipe

Share your amazing recipies accross the world. Fill our form to get started.

Email

Recipe Name

Description

Ingredients

Example: Ice

Select Category

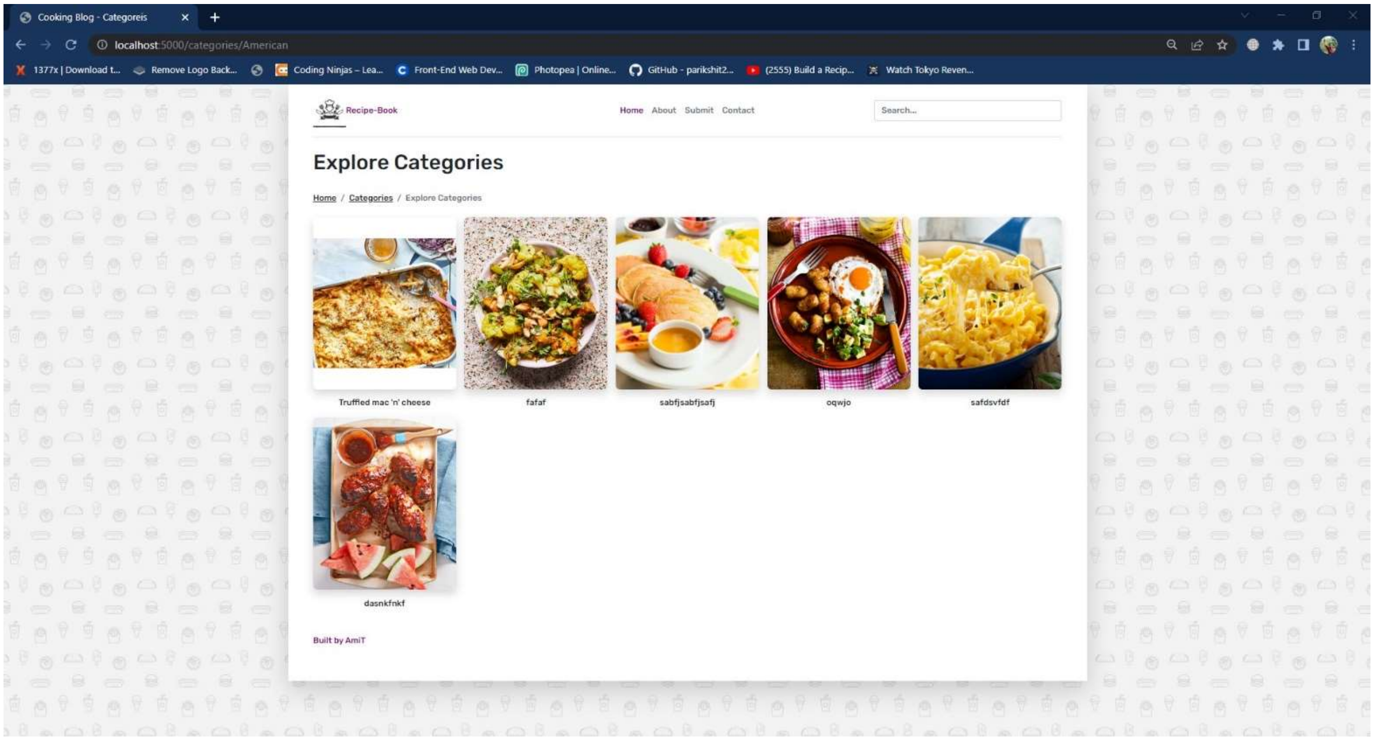
Select Category

Product Image

No file chosen

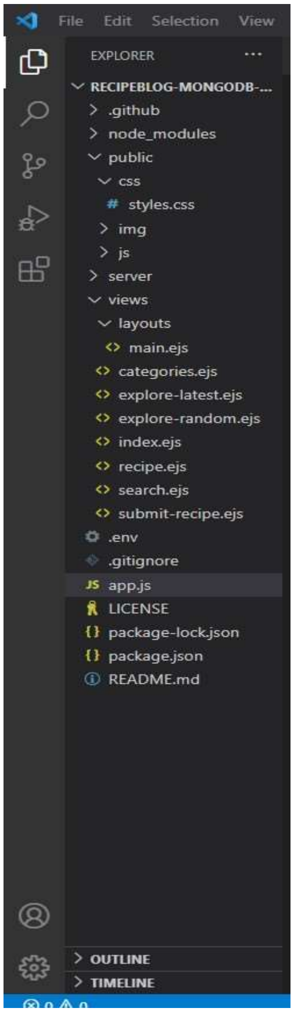
Built by AmiT

Submit Form



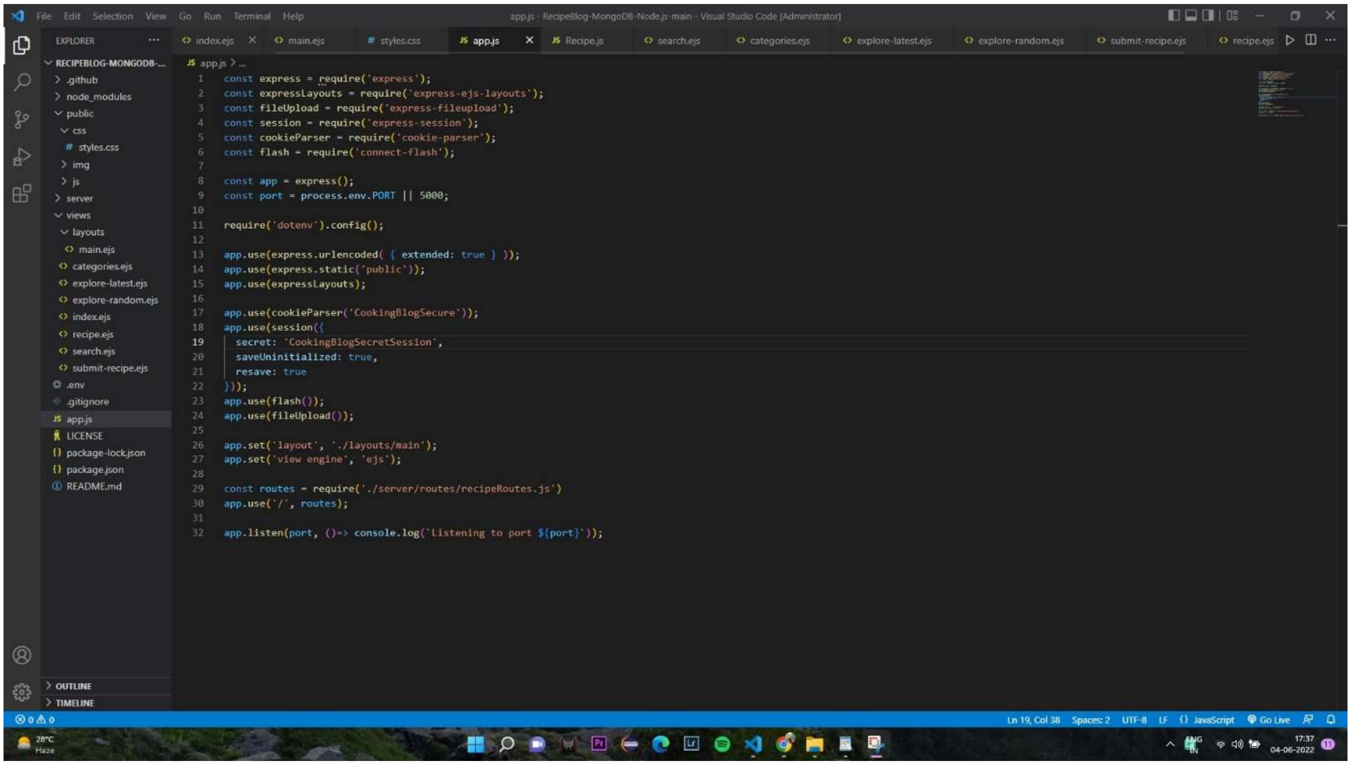
Explore Category

Screenshots of the Code: -



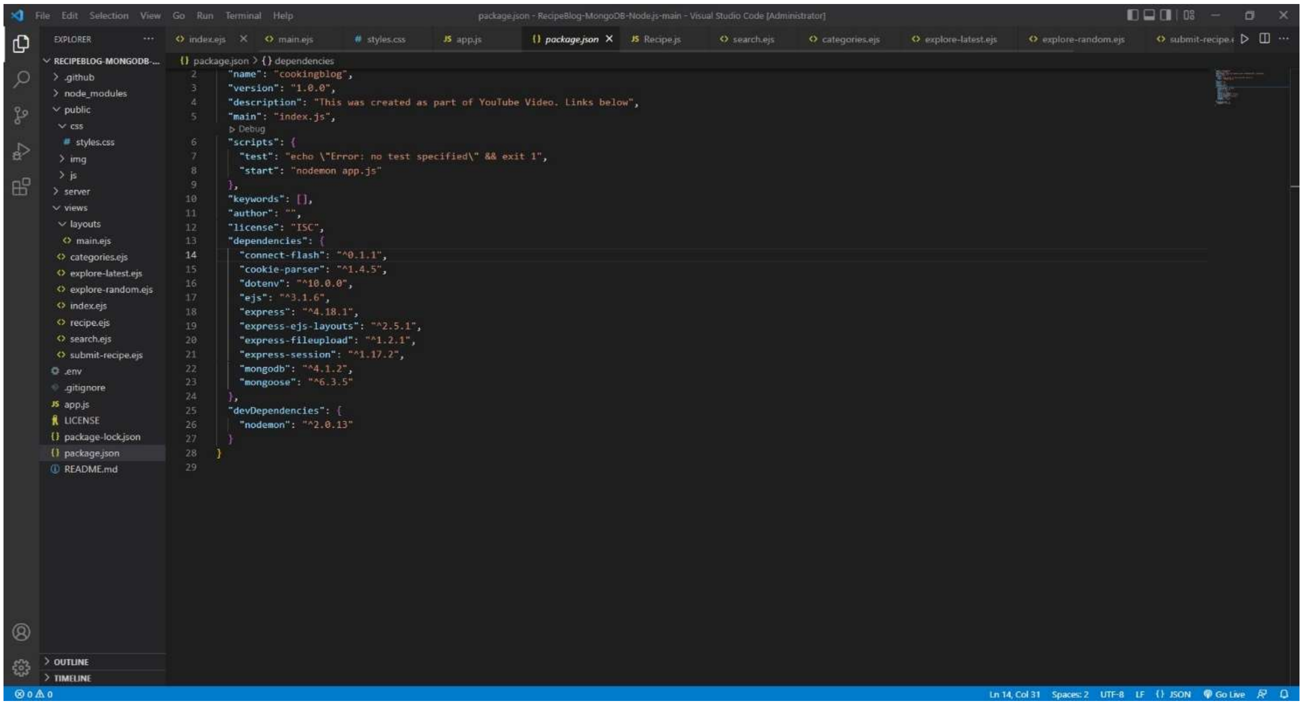
```
RECIPEBLOG-MONGODB-...
  > .github
  > node_modules
  v public
    v css
      # styles.css
    > img
    > js
  > server
  v views
    v layouts
      <> main.ejs
    <> categories.ejs
    <> explore-latest.ejs
    <> explore-random.ejs
    <> index.ejs
    <> recipe.ejs
    <> search.ejs
    <> submit-recipe.ejs
  .env
  .gitignore
  JS app.js
  LICENSE
  {} package-lock.json
  {} package.json
  README.md
```

Files Created



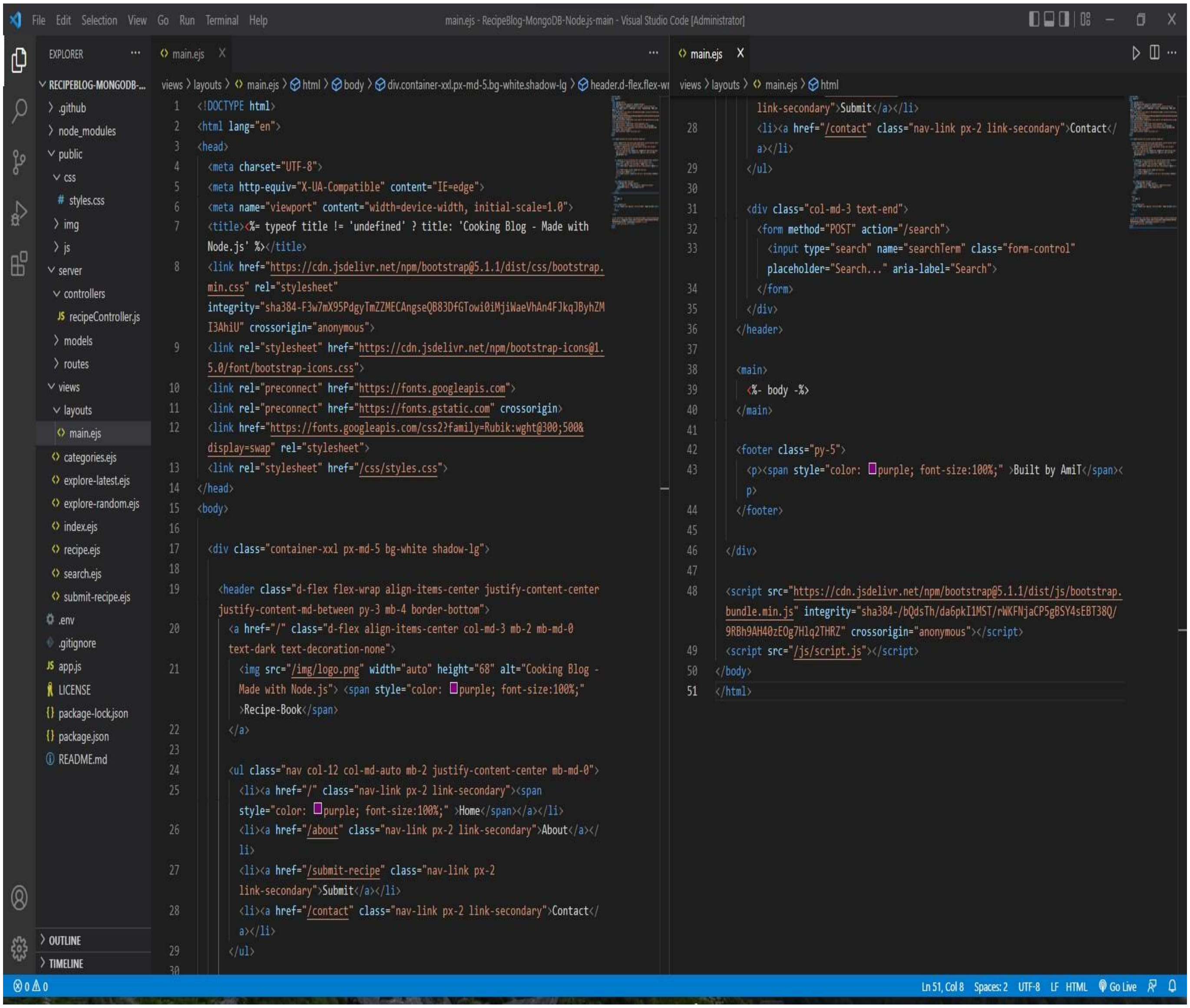
```javascript
const express = require('express');
const expressLayouts = require('express-ejs-layouts');
const fileUpload = require('express-fileupload');
const session = require('express-session');
const cookieParser = require('cookie-parser');
const flash = require('connect-flash');

const app = express();
const port = process.env.PORT || 5000;

require('dotenv').config();

app.use(express.urlencoded( { extended: true } ));
app.use(express.static('public'));
app.use(expressLayouts);

app.use(cookieParser('CookingBlogSecure'));
app.use(session({
  secret: 'CookingBlogSecretSession',
  saveUninitialized: true,
  resave: true
}));
app.use(flash());
app.use(fileUpload());

app.set('layout', './layouts/main');
app.set('view engine', 'ejs');

const routes = require('./server/routes/recipeRoutes.js')
app.use('/', routes);

app.listen(port, ()=> console.log(`Listening to port ${port}`));
```

Express.js



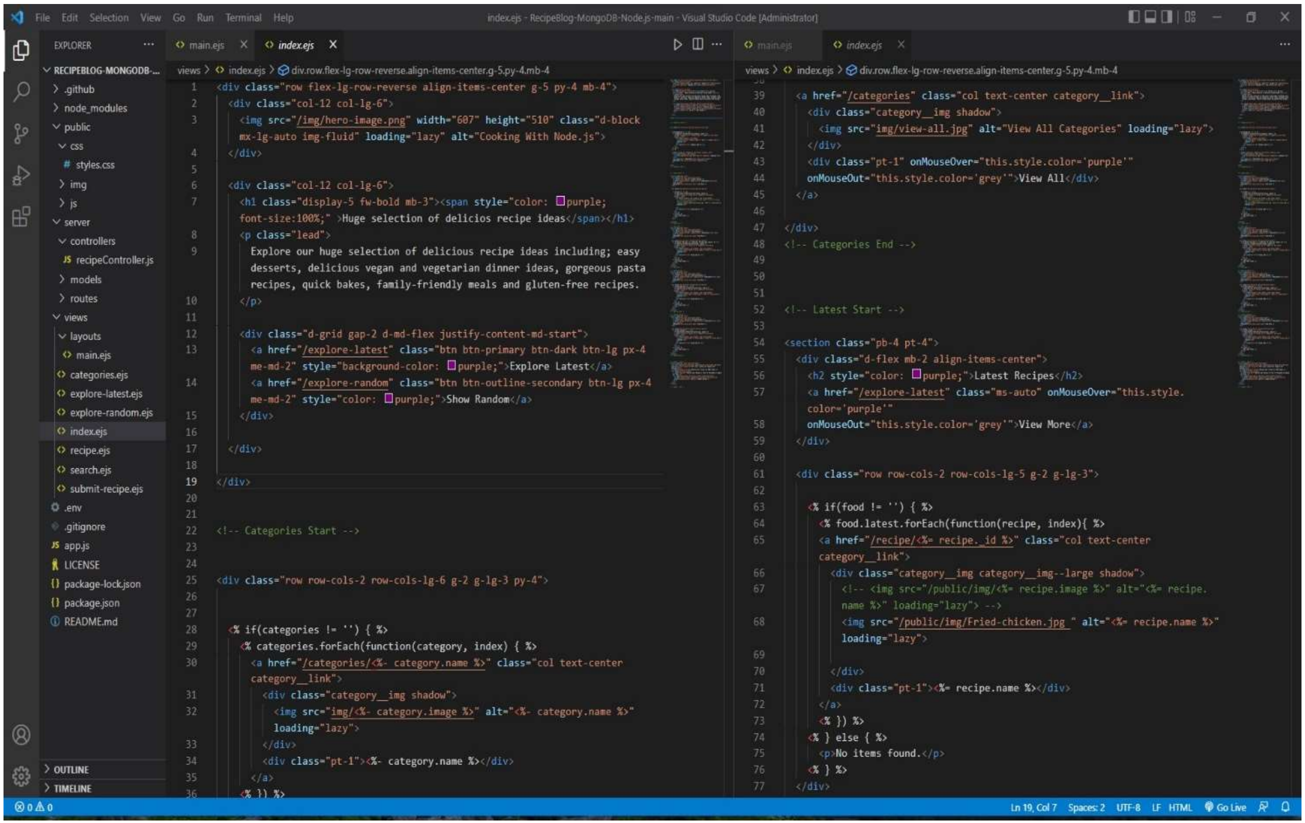
```json
  "name": "cookingblog",
  "version": "1.0.0",
  "description": "This was created as part of YouTube Video. Links below",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1",
    "start": "nodemon app.js"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "dependencies": {
    "connect-flash": "^0.1.1",
    "cookie-parser": "^1.4.5",
    "dotenv": "^10.0.0",
    "ejs": "^3.1.6",
    "express": "^4.18.1",
    "express-ejs-layouts": "^2.5.1",
    "express-fileupload": "^1.2.1",
    "express-session": "^1.17.2",
    "mongodb": "^4.1.2",
    "mongoose": "^6.3.5"
  },
  "devDependencies": {
    "nodemon": "^2.0.13"
  }
}
```

Dependencies



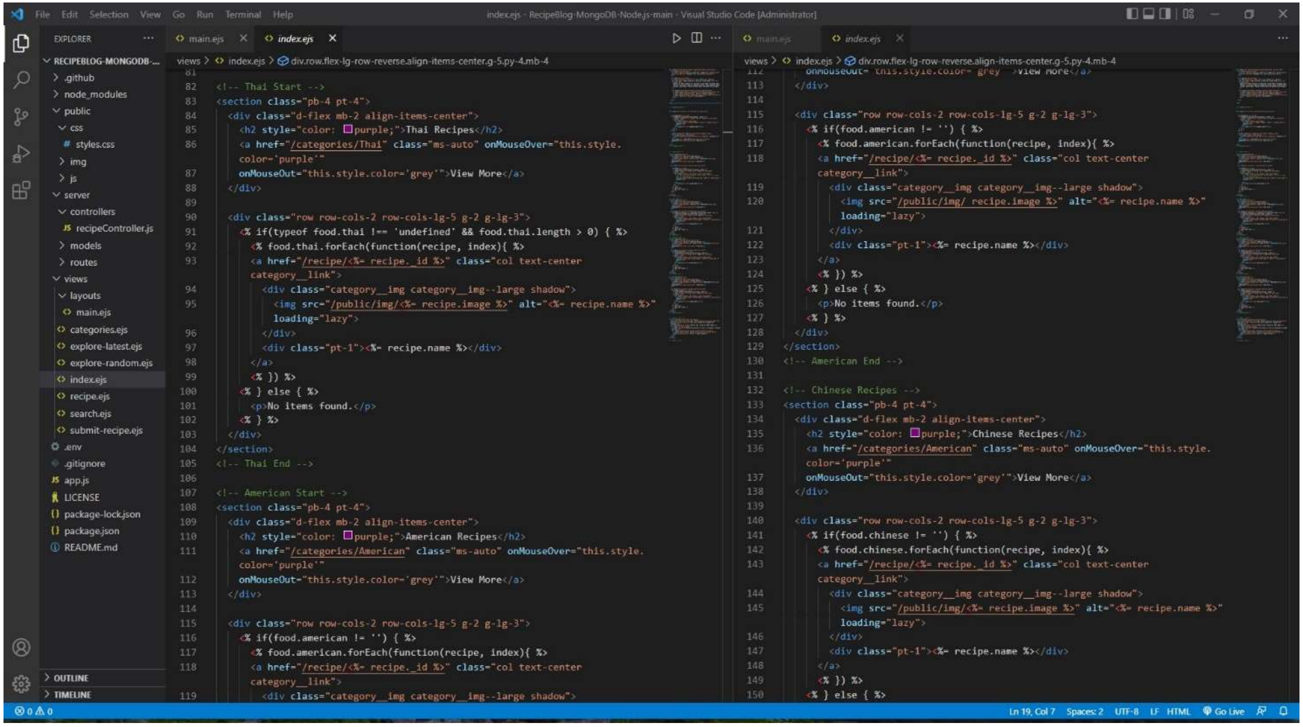
```html
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title><%= typeof title != 'undefined' ? title: 'Cooking Blog - Made with Node.js' %></title>
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.1/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-F3w7mX95PdgyTmZZMECAngseQB83DfGTowi0iMjiWaeVhAn4FJkqJByhZMI3AhiU" crossorigin="anonymous">
  <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap-icons@1.5.0/font/bootstrap-icons.css">
  <link rel="preconnect" href="https://fonts.googleapis.com">
  <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
  <link href="https://fonts.googleapis.com/css2?family=Rubik:wght@300;500&display=swap" rel="stylesheet">
  <link rel="stylesheet" href="/css/styles.css">
</head>
<body>

  <div class="container-xxl px-md-5 bg-white shadow-lg">

    <header class="d-flex flex-wrap align-items-center justify-content-center justify-content-md-between py-3 mb-4 border-bottom">
      <a href="/" class="d-flex align-items-center col-md-3 mb-2 mb-md-0 text-dark text-decoration-none">
        <img src="/img/logo.png" width="auto" height="68" alt="Cooking Blog - Made with Node.js"> <span style="color: purple; font-size:100%;" >Recipe-Book</span>
      </a>

      <ul class="nav col-12 col-md-auto mb-2 justify-content-center mb-md-0">
        <li><a href="/" class="nav-link px-2 link-secondary"><span style="color: purple; font-size:100%;" >Home</span></a></li>
        <li><a href="/about" class="nav-link px-2 link-secondary">About</a></li>
        <li><a href="/submit-recipe" class="nav-link px-2 link-secondary">Submit</a></li>
        <li><a href="/contact" class="nav-link px-2 link-secondary">Contact</a></li>
      </ul>

      <div class="col-md-3 text-end">
        <form method="POST" action="/search">
          <input type="search" name="searchTerm" class="form-control" placeholder="Search..." aria-label="Search">
        </form>
      </div>
    </header>

    <main>
      <%- body -%>
    </main>

    <footer class="py-5">
      <p><span style="color: purple; font-size:100%;" >Built by AmiT</span></p>
    </footer>

  </div>

  <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.1/dist/js/bootstrap.bundle.min.js" integrity="sha384-/bQdsTh/da6pkI1MST/rWKFNjaCP5gBSY4sEBT38Q/9RBh9AH40zEOg7Hlq2THRZ" crossorigin="anonymous"></script>
  <script src="/js/script.js"></script>
</body>
</html>
```

Main EJS file



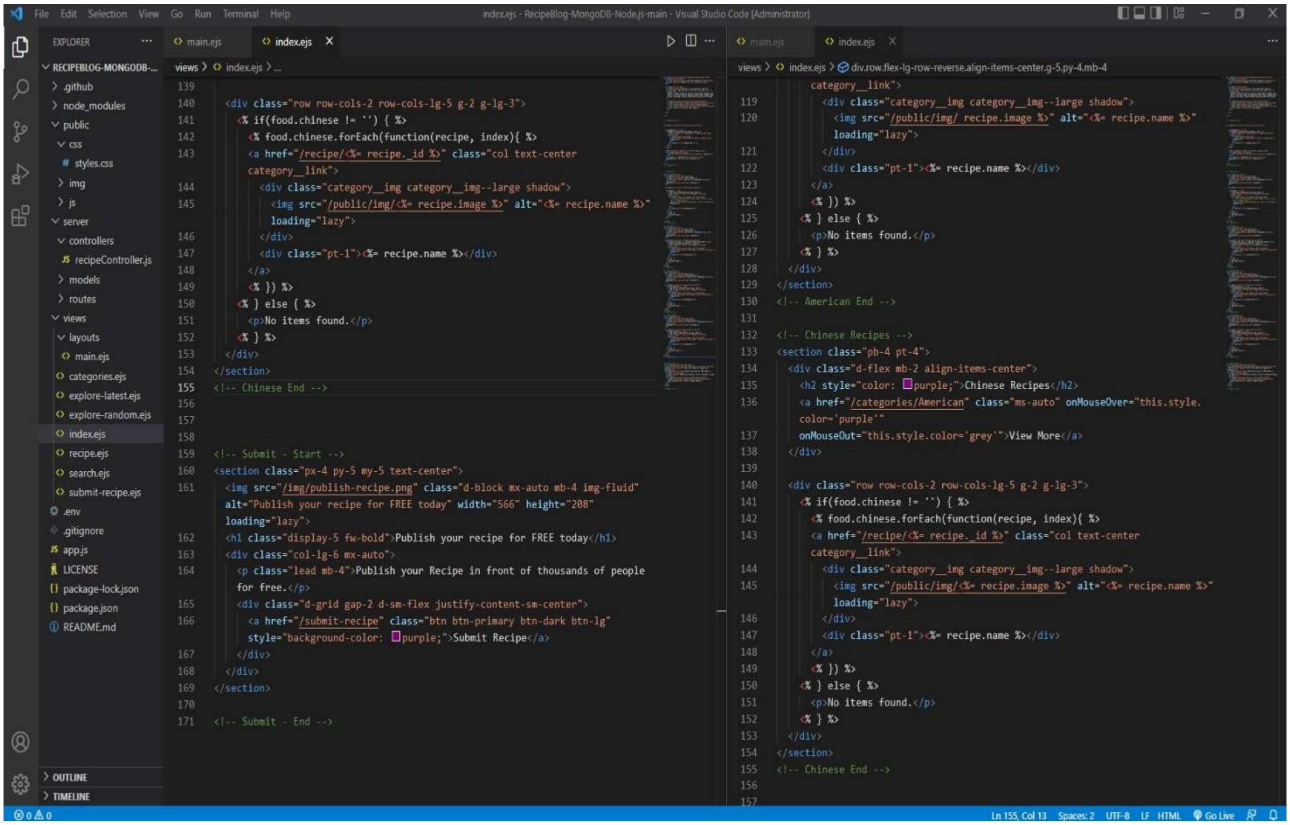
```
<div class="row flex-lg-row-reverse align-items-center g-5 py-4 mb-4">
  <div class="col-12 col-lg-6">
    <img src="/img/hero-image.png" width="607" height="510" class="d-block mx-lg-auto img-fluid" loading="lazy" alt="Cooking With Node.js">
  </div>

  <div class="col-12 col-lg-6">
    <h1 class="display-5 fw-bold mb-3"><span style="color: ■purple; font-size:100%;" >Huge selection of delicios recipe ideas</span></h1>
    <p class="lead">
      Explore our huge selection of delicious recipe ideas including; easy desserts, delicious vegan and vegetarian dinner ideas, gorgeous pasta recipes, quick bakes, family-friendly meals and gluten-free recipes.
    </p>

    <div class="d-grid gap-2 d-md-flex justify-content-md-start">
      <a href="/explore-latest" class="btn btn-primary btn-dark btn-lg px-4 me-md-2" style="background-color: ■purple;">Explore Latest</a>
      <a href="/explore-random" class="btn btn-outline-secondary btn-lg px-4 me-md-2" style="color: ■purple;">Show Random</a>
    </div>

  </div>

</div>


<!-- Categories Start -->


<div class="row row-cols-2 row-cols-lg-6 g-2 g-lg-3 py-4">


  <% if(categories != '') { %>
    <% categories.forEach(function(category, index) { %>
      <a href="/categories/<%- category.name %>" class="col text-center category__link">
        <div class="category__img shadow">
          <img src="img/<%- category.image %>" alt="<%- category.name %>" loading="lazy">
        </div>
        <div class="pt-1"><%- category.name %></div>
      </a>
    <% }) %>
```

```
  <a href="/categories" class="col text-center category__link">
    <div class="category__img shadow">
      <img src="img/view-all.jpg" alt="View All Categories" loading="lazy">
    </div>
    <div class="pt-1" onMouseOver="this.style.color='purple'"
    onMouseOut="this.style.color='grey'">View All</div>
  </a>

</div>
<!-- Categories End -->



<!-- Latest Start -->

<section class="pb-4 pt-4">
  <div class="d-flex mb-2 align-items-center">
    <h2 style="color: ■purple;">Latest Recipes</h2>
    <a href="/explore-latest" class="ms-auto" onMouseOver="this.style.color='purple'"
    onMouseOut="this.style.color='grey'">View More</a>
  </div>

  <div class="row row-cols-2 row-cols-lg-5 g-2 g-lg-3">

    <% if(food != '') { %>
      <% food.latest.forEach(function(recipe, index){ %>
      <a href="/recipe/<%= recipe._id %>" class="col text-center category__link">
        <div class="category__img category__img--large shadow">
          <!-- <img src="/public/img/<%= recipe.image %>" alt="<%= recipe.name %>" loading="lazy"> -->
          <img src="/public/img/Fried-chicken.jpg " alt="<%= recipe.name %>" loading="lazy">

        </div>
        <div class="pt-1"><%= recipe.name %></div>
      </a>
      <% }) %>
    <% } else { %>
      <p>No items found.</p>
    <% } %>
  </div>
```



```
<!-- Thai Start -->
<section class="pb-4 pt-4">
  <div class="d-flex mb-2 align-items-center">
    <h2 style="color: ■purple;">Thai Recipes</h2>
    <a href="/categories/Thai" class="ms-auto" onMouseOver="this.style.color='purple'"
    onMouseOut="this.style.color='grey'">View More</a>
  </div>

  <div class="row row-cols-2 row-cols-lg-5 g-2 g-lg-3">
    <% if(typeof food.thai !== 'undefined' && food.thai.length > 0) { %>
      <% food.thai.forEach(function(recipe, index){ %>
      <a href="/recipe/<%= recipe._id %>" class="col text-center category__link">
        <div class="category__img category__img--large shadow">
          <img src="/public/img/<%= recipe.image %>" alt="<%= recipe.name %>" loading="lazy">
        </div>
        <div class="pt-1"><%= recipe.name %></div>
      </a>
      <% }) %>
    <% } else { %>
      <p>No items found.</p>
    <% } %>
  </div>
</section>
<!-- Thai End -->

<!-- American Start -->
<section class="pb-4 pt-4">
  <div class="d-flex mb-2 align-items-center">
    <h2 style="color: ■purple;">American Recipes</h2>
    <a href="/categories/American" class="ms-auto" onMouseOver="this.style.color='purple'"
    onMouseOut="this.style.color='grey'">View More</a>
  </div>

  <div class="row row-cols-2 row-cols-lg-5 g-2 g-lg-3">
    <% if(food.american != '') { %>
      <% food.american.forEach(function(recipe, index){ %>
      <a href="/recipe/<%= recipe._id %>" class="col text-center category__link">
        <div class="category__img category__img--large shadow">
```

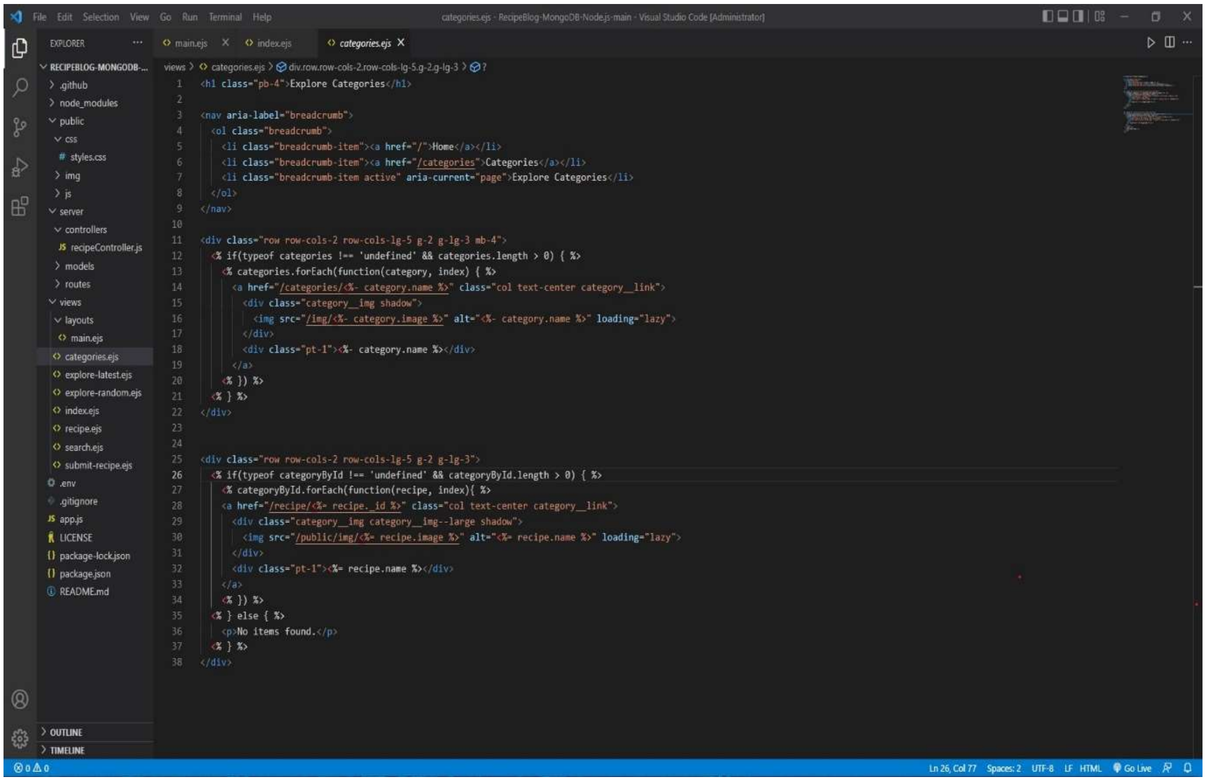
```
    onMouseOut="this.style.color='grey'">View More</a>
  </div>

  <div class="row row-cols-2 row-cols-lg-5 g-2 g-lg-3">
    <% if(food.american != '') { %>
      <% food.american.forEach(function(recipe, index){ %>
      <a href="/recipe/<%= recipe._id %>" class="col text-center category__link">
        <div class="category__img category__img--large shadow">
          <img src="/public/img/ recipe.image %>" alt="<%= recipe.name %>" loading="lazy">
        </div>
        <div class="pt-1"><%= recipe.name %></div>
      </a>
      <% }) %>
    <% } else { %>
      <p>No items found.</p>
    <% } %>
  </div>
</section>
<!-- American End -->

<!-- Chinese Recipes -->
<section class="pb-4 pt-4">
  <div class="d-flex mb-2 align-items-center">
    <h2 style="color: ■purple;">Chinese Recipes</h2>
    <a href="/categories/American" class="ms-auto" onMouseOver="this.style.color='purple'"
    onMouseOut="this.style.color='grey'">View More</a>
  </div>

  <div class="row row-cols-2 row-cols-lg-5 g-2 g-lg-3">
    <% if(food.chinese != '') { %>
      <% food.chinese.forEach(function(recipe, index){ %>
      <a href="/recipe/<%= recipe._id %>" class="col text-center category__link">
        <div class="category__img category__img--large shadow">
          <img src="/public/img/<%= recipe.image %>" alt="<%= recipe.name %>" loading="lazy">
        </div>
        <div class="pt-1"><%= recipe.name %></div>
      </a>
      <% }) %>
    <% } else { %>
```



```html

  <div class="row row-cols-2 row-cols-lg-5 g-2 g-lg-3">
    <% if(food.chinese != '') { %>
      <% food.chinese.forEach(function(recipe, index){ %>
      <a href="/recipe/<%= recipe._id %>" class="col text-center
      category__link">
        <div class="category__img category__img--large shadow">
          <img src="/public/img/<%= recipe.image %>" alt="<%= recipe.name %>"
          loading="lazy">
        </div>
        <div class="pt-1"><%= recipe.name %></div>
      </a>
      <% }) %>
    <% } else { %>
      <p>No items found.</p>
    <% } %>
  </div>
</section>
<!-- Chinese End -->



<!-- Submit - Start -->
<section class="px-4 py-5 my-5 text-center">
  <img src="/img/publish-recipe.png" class="d-block mx-auto mb-4 img-fluid"
  alt="Publish your recipe for FREE today" width="566" height="208"
  loading="lazy">
  <h1 class="display-5 fw-bold">Publish your recipe for FREE today</h1>
  <div class="col-lg-6 mx-auto">
    <p class="lead mb-4">Publish your Recipe in front of thousands of people
    for free.</p>
    <div class="d-grid gap-2 d-sm-flex justify-content-sm-center">
      <a href="/submit-recipe" class="btn btn-primary btn-dark btn-lg"
      style="background-color: purple;">Submit Recipe</a>
    </div>
  </div>
</section>

<!-- Submit - End -->
```

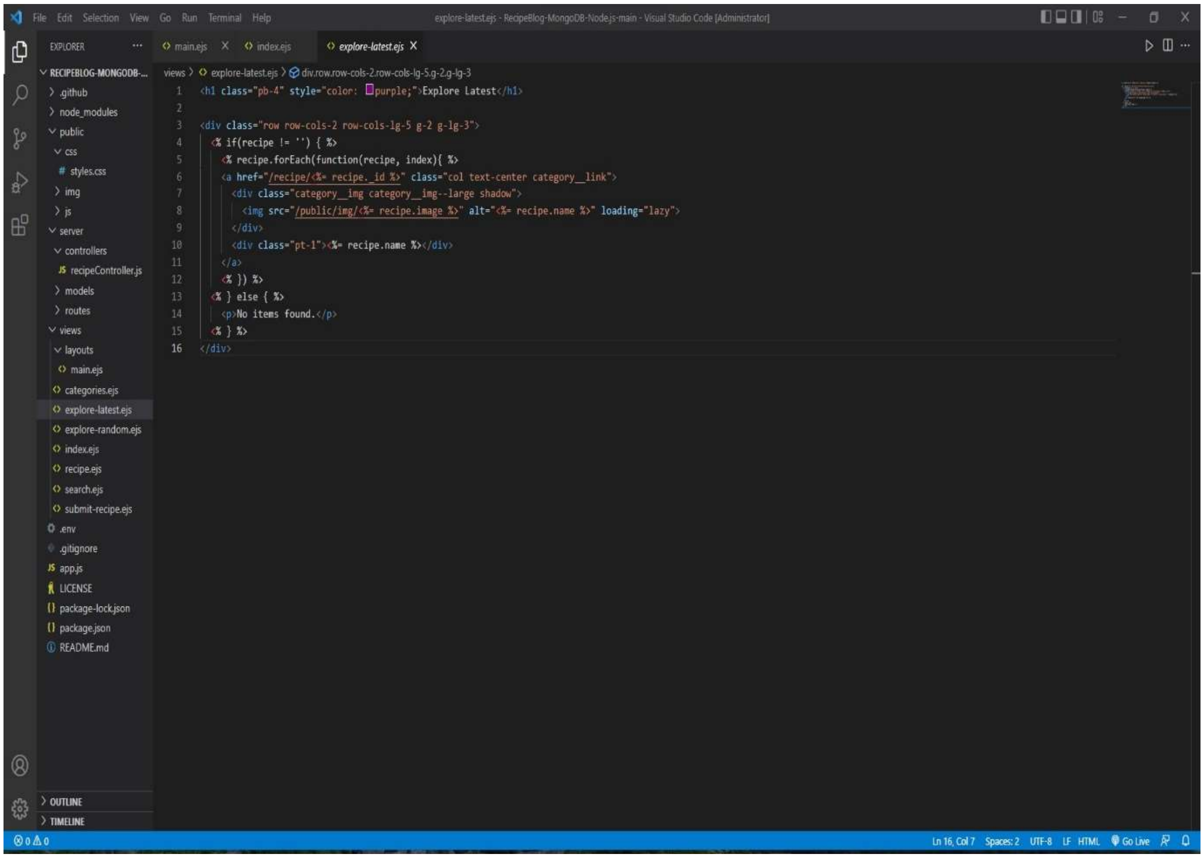
```html
      category__link">
        <div class="category__img category__img--large shadow">
          <img src="/public/img/ recipe.image %>" alt="<%= recipe.name %>"
          loading="lazy">
        </div>
        <div class="pt-1"><%= recipe.name %></div>
      </a>
      <% }) %>
    <% } else { %>
      <p>No items found.</p>
    <% } %>
  </div>
</section>
<!-- American End -->

<!-- Chinese Recipes -->
<section class="pb-4 pt-4">
  <div class="d-flex mb-2 align-items-center">
    <h2 style="color: purple;">Chinese Recipes</h2>
    <a href="/categories/American" class="ms-auto" onMouseOver="this.style.
    color='purple'"
    onMouseOut="this.style.color='grey'">View More</a>
  </div>

  <div class="row row-cols-2 row-cols-lg-5 g-2 g-lg-3">
    <% if(food.chinese != '') { %>
      <% food.chinese.forEach(function(recipe, index){ %>
      <a href="/recipe/<%= recipe._id %>" class="col text-center
      category__link">
        <div class="category__img category__img--large shadow">
          <img src="/public/img/<%= recipe.image %>" alt="<%= recipe.name %>"
          loading="lazy">
        </div>
        <div class="pt-1"><%= recipe.name %></div>
      </a>
      <% }) %>
    <% } else { %>
      <p>No items found.</p>
    <% } %>
  </div>
</section>
<!-- Chinese End -->

```

Index.ejs File (layout of the webpage)



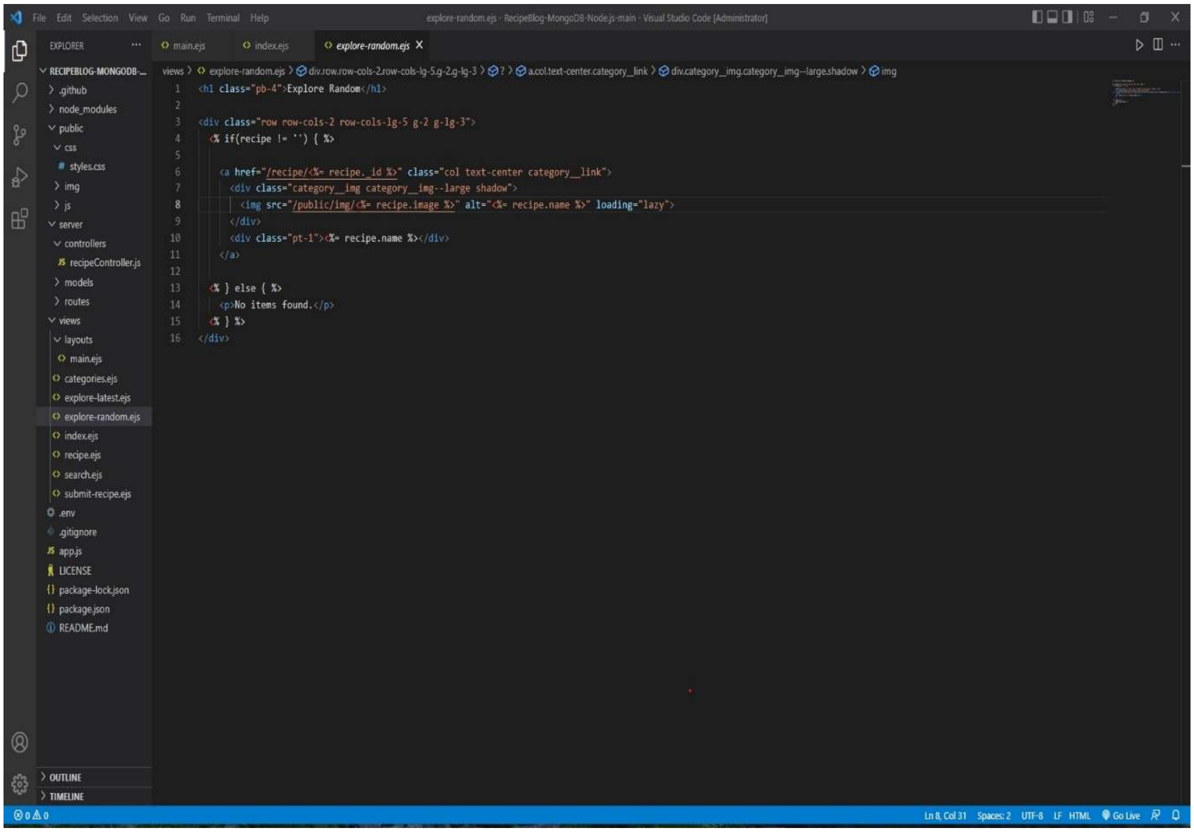
```
<h1 class="pb-4">Explore Categories</h1>

<nav aria-label="breadcrumb">
  <ol class="breadcrumb">
    <li class="breadcrumb-item"><a href="/">Home</a></li>
    <li class="breadcrumb-item"><a href="/categories">Categories</a></li>
    <li class="breadcrumb-item active" aria-current="page">Explore Categories</li>
  </ol>
</nav>

<div class="row row-cols-2 row-cols-lg-5 g-2 g-lg-3 mb-4">
  <% if(typeof categories !== 'undefined' && categories.length > 0) { %>
    <% categories.forEach(function(category, index) { %>
      <a href="/categories/<%- category.name %>" class="col text-center category__link">
        <div class="category__img shadow">
          <img src="/img/<%- category.image %>" alt="<%- category.name %>" loading="lazy">
        </div>
        <div class="pt-1"><%- category.name %></div>
      </a>
    <% }) %>
  <% } %>
</div>


<div class="row row-cols-2 row-cols-lg-5 g-2 g-lg-3">
  <% if(typeof categoryById !== 'undefined' && categoryById.length > 0) { %>
    <% categoryById.forEach(function(recipe, index){ %>
    <a href="/recipe/<%= recipe._id %>" class="col text-center category__link">
      <div class="category__img category__img--large shadow">
        <img src="/public/img/<%= recipe.image %>" alt="<%= recipe.name %>" loading="lazy">
      </div>
      <div class="pt-1"><%= recipe.name %></div>
    </a>
    <% }) %>
  <% } else { %>
    <p>No items found.</p>
  <% } %>
</div>
```

Categories Section Ejs File



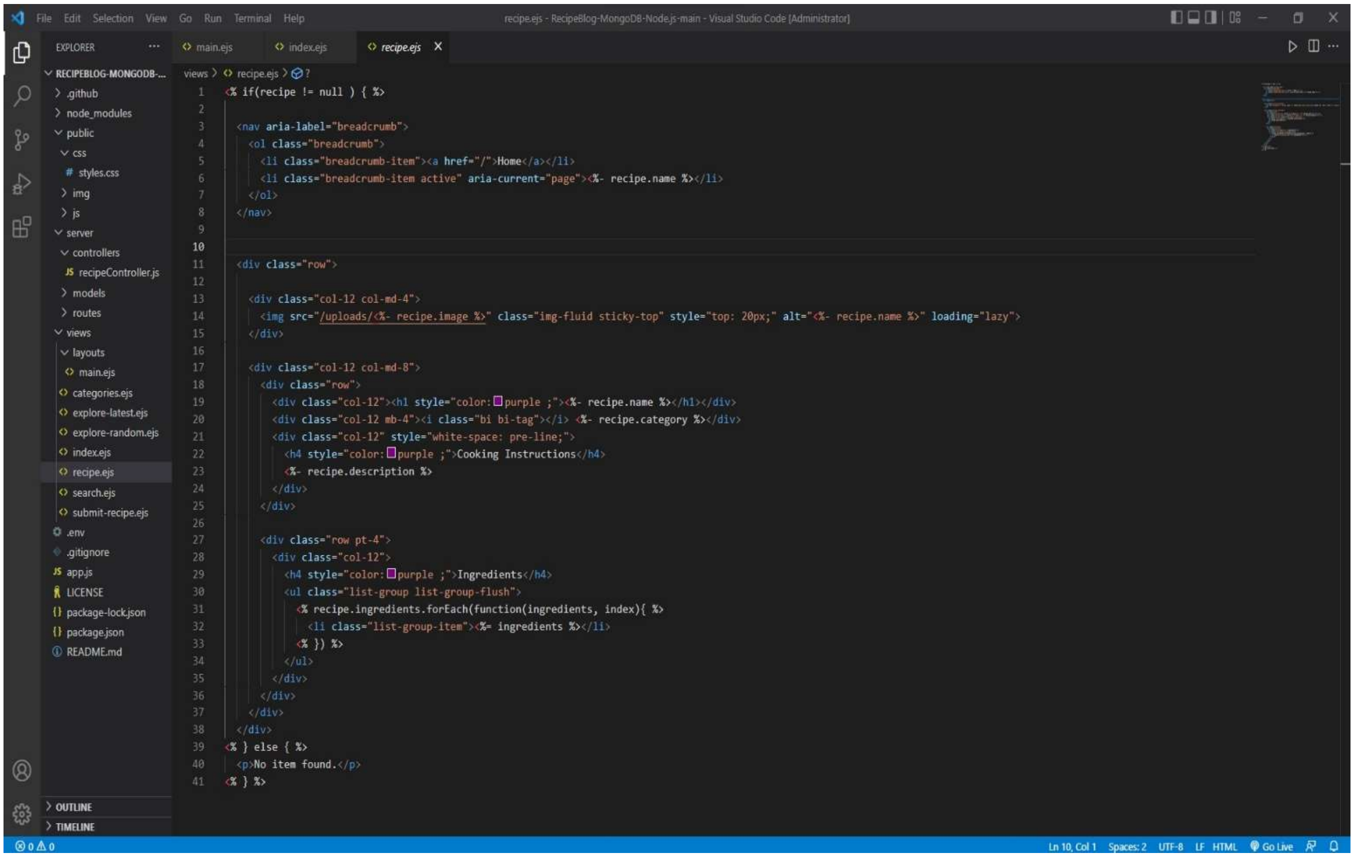
```
<h1 class="pb-4" style="color: purple;">Explore Latest</h1>

<div class="row row-cols-2 row-cols-lg-5 g-2 g-lg-3">
  <% if(recipe != '') { %>
    <% recipe.forEach(function(recipe, index){ %>
    <a href="/recipe/<%= recipe._id %>" class="col text-center category__link">
      <div class="category__img category__img--large shadow">
        <img src="/public/img/<%= recipe.image %>" alt="<%= recipe.name %>" loading="lazy">
      </div>
      <div class="pt-1"><%= recipe.name %></div>
    </a>
    <% }) %>
  <% } else { %>
    <p>No items found.</p>
  <% } %>
</div>
```

Latest-recipe Section Ejs File



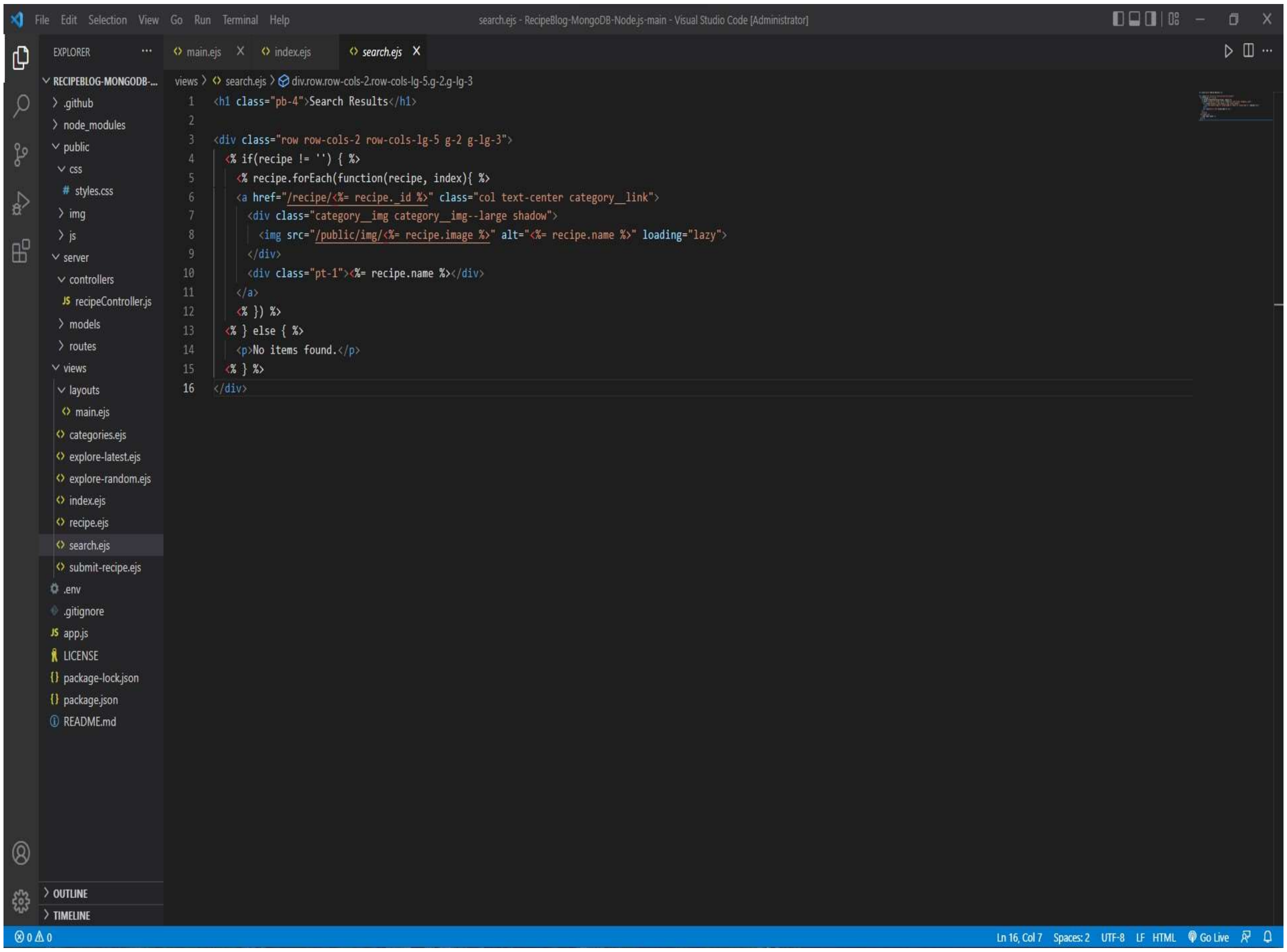
```
<h1 class="pb-4">Explore Random</h1>

<div class="row row-cols-2 row-cols-lg-5 g-2 g-lg-3">
  <% if(recipe != '') { %>

    <a href="/recipe/<%= recipe._id %>" class="col text-center category__link">
      <div class="category__img category__img--large shadow">
        <img src="/public/img/<%= recipe.image %>" alt="<%= recipe.name %>" loading="lazy">
      </div>
      <div class="pt-1"><%= recipe.name %></div>
    </a>

  <% } else { %>
    <p>No items found.</p>
  <% } %>
</div>
```

Random-recipe Section Ejs File



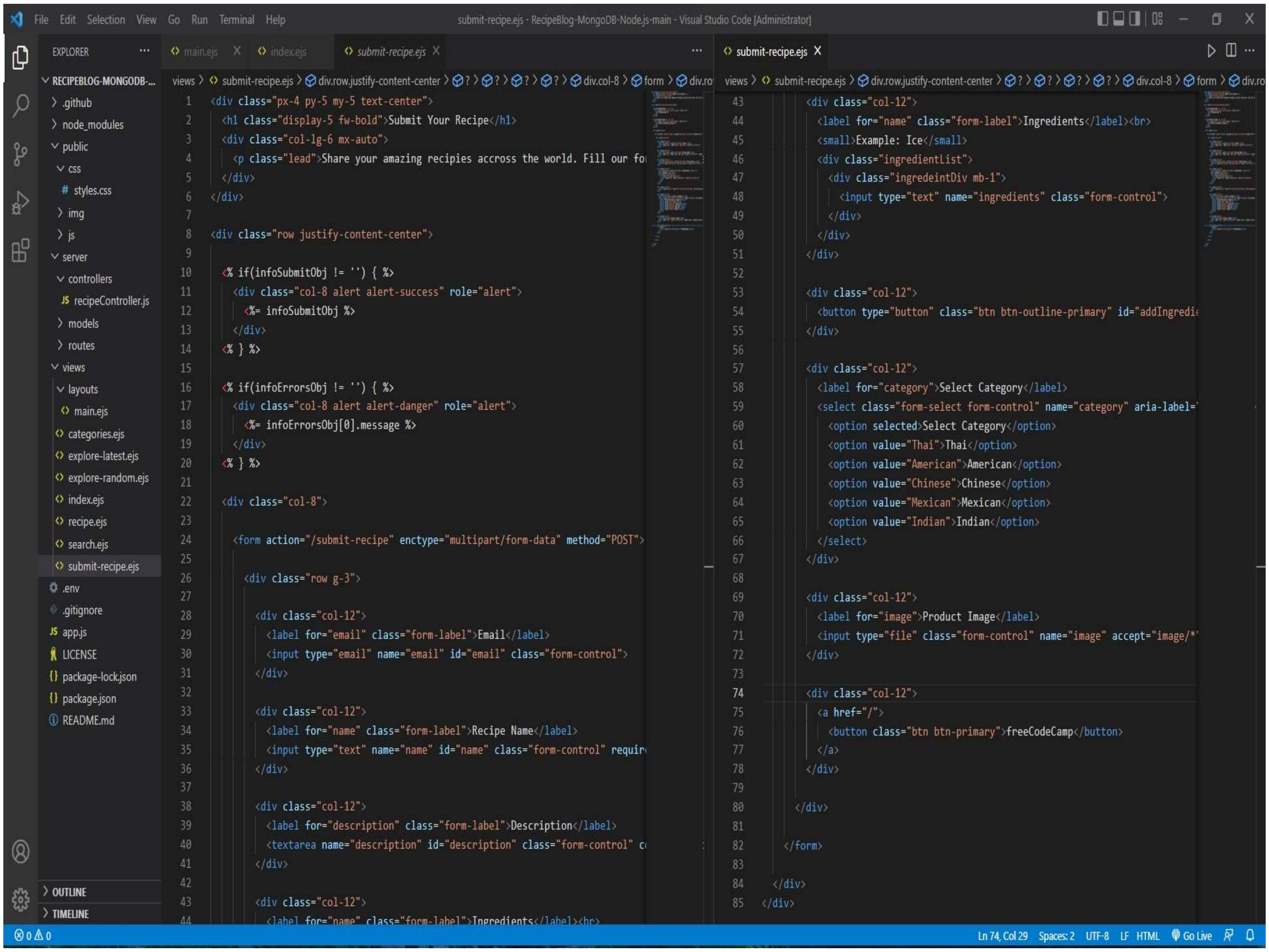
```
<% if(recipe != null ) { %>

  <nav aria-label="breadcrumb">
    <ol class="breadcrumb">
      <li class="breadcrumb-item"><a href="/">Home</a></li>
      <li class="breadcrumb-item active" aria-current="page"><%- recipe.name %></li>
    </ol>
  </nav>


  <div class="row">

    <div class="col-12 col-md-4">
      <img src="/uploads/<%- recipe.image %>" class="img-fluid sticky-top" style="top: 20px;" alt="<%- recipe.name %>" loading="lazy">
    </div>

    <div class="col-12 col-md-8">
      <div class="row">
        <div class="col-12"><h1 style="color:purple ;"><%- recipe.name %></h1></div>
        <div class="col-12 mb-4"><i class="bi bi-tag"></i> <%- recipe.category %></div>
        <div class="col-12" style="white-space: pre-line;">
          <h4 style="color:purple ;">Cooking Instructions</h4>
          <%- recipe.description %>
        </div>
      </div>

      <div class="row pt-4">
        <div class="col-12">
          <h4 style="color:purple ;">Ingredients</h4>
          <ul class="list-group list-group-flush">
            <% recipe.ingredients.forEach(function(ingredients, index){ %>
              <li class="list-group-item"><%= ingredients %></li>
            <% }) %>
          </ul>
        </div>
      </div>
    </div>
  </div>
<% } else { %>
  <p>No item found.</p>
<% } %>
```

Recipe Section Ejs File



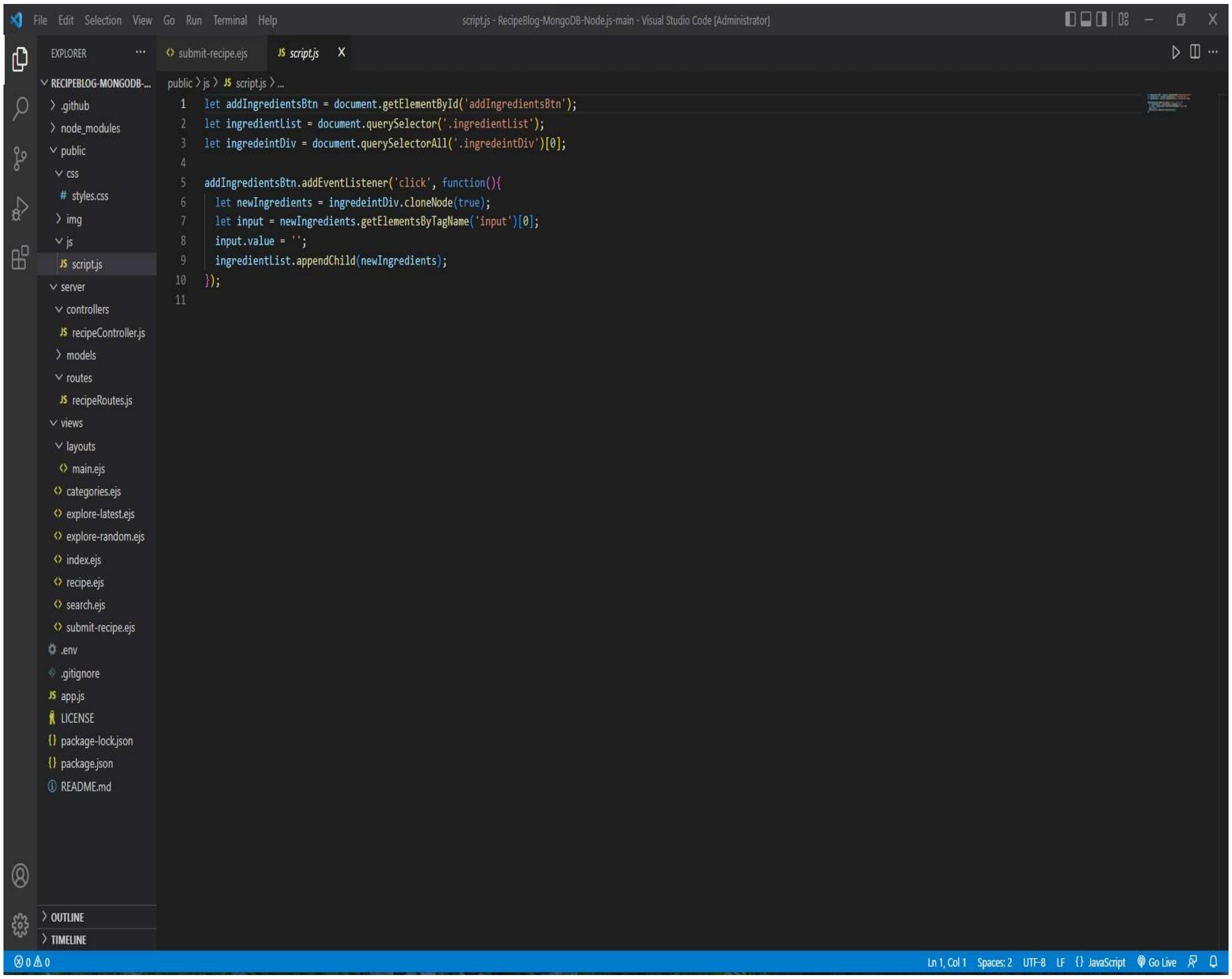
```
<h1 class="pb-4">Search Results</h1>

<div class="row row-cols-2 row-cols-lg-5 g-2 g-lg-3">
  <% if(recipe != '') { %>
    <% recipe.forEach(function(recipe, index){ %>
    <a href="/recipe/<%= recipe._id %>" class="col text-center category__link">
      <div class="category__img category__img--large shadow">
        <img src="/public/img/<%= recipe.image %>" alt="<%= recipe.name %>" loading="lazy">
      </div>
      <div class="pt-1"><%= recipe.name %></div>
    </a>
    <% }) %>
  <% } else { %>
    <p>No items found.</p>
  <% } %>
</div>
```

Search bar Ejs File



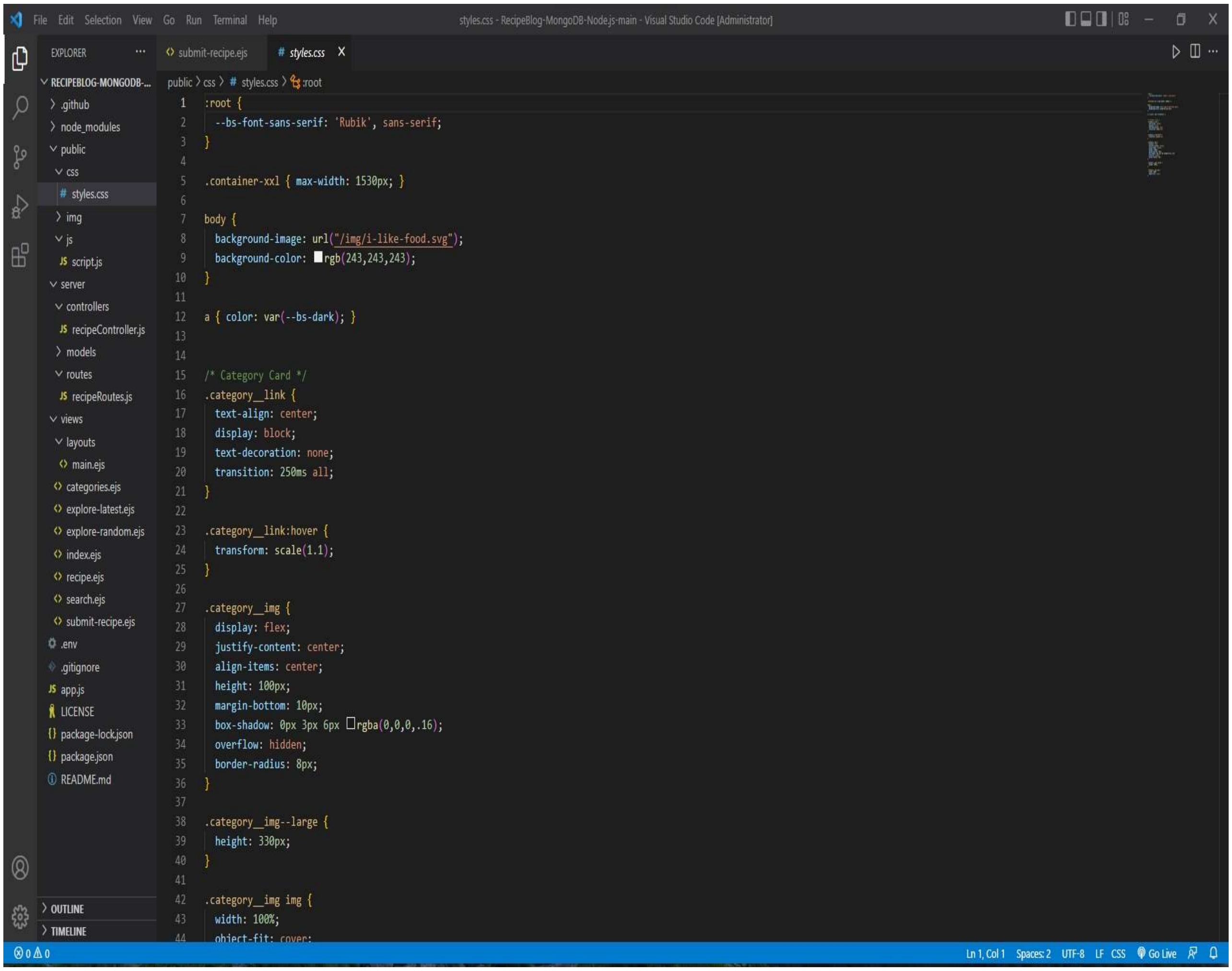
```
<div class="px-4 py-5 my-5 text-center">
  <h1 class="display-5 fw-bold">Submit Your Recipe</h1>
  <div class="col-lg-6 mx-auto">
    <p class="lead">Share your amazing recipies accross the world. Fill our fo
  </div>
</div>

<div class="row justify-content-center">

  <% if(infoSubmitObj != '') { %>
    <div class="col-8 alert alert-success" role="alert">
      <%= infoSubmitObj %>
    </div>
  <% } %>

  <% if(infoErrorsObj != '') { %>
    <div class="col-8 alert alert-danger" role="alert">
      <%= infoErrorsObj[0].message %>
    </div>
  <% } %>

  <div class="col-8">

    <form action="/submit-recipe" enctype="multipart/form-data" method="POST">

      <div class="row g-3">

        <div class="col-12">
          <label for="email" class="form-label">Email</label>
          <input type="email" name="email" id="email" class="form-control">
        </div>

        <div class="col-12">
          <label for="name" class="form-label">Recipe Name</label>
          <input type="text" name="name" id="name" class="form-control" requir
        </div>

        <div class="col-12">
          <label for="description" class="form-label">Description</label>
          <textarea name="description" id="description" class="form-control" c
        </div>

        <div class="col-12">
          <label for="name" class="form-label">Ingredients</label><br>
          <small>Example: Ice</small>
          <div class="ingredientList">
            <div class="ingredeintDiv mb-1">
              <input type="text" name="ingredients" class="form-control">
            </div>
          </div>
        </div>

        <div class="col-12">
          <button type="button" class="btn btn-outline-primary" id="addIngredie
        </div>

        <div class="col-12">
          <label for="category">Select Category</label>
          <select class="form-select form-control" name="category" aria-label="
            <option selected>Select Category</option>
            <option value="Thai">Thai</option>
            <option value="American">American</option>
            <option value="Chinese">Chinese</option>
            <option value="Mexican">Mexican</option>
            <option value="Indian">Indian</option>
          </select>
        </div>

        <div class="col-12">
          <label for="image">Product Image</label>
          <input type="file" class="form-control" name="image" accept="image/*"
        </div>

        <div class="col-12">
          <a href="/">
            <button class="btn btn-primary">freeCodeCamp</button>
          </a>
        </div>

      </div>

    </form>

  </div>
</div>
```

Recipe-Submit Form Ejs File



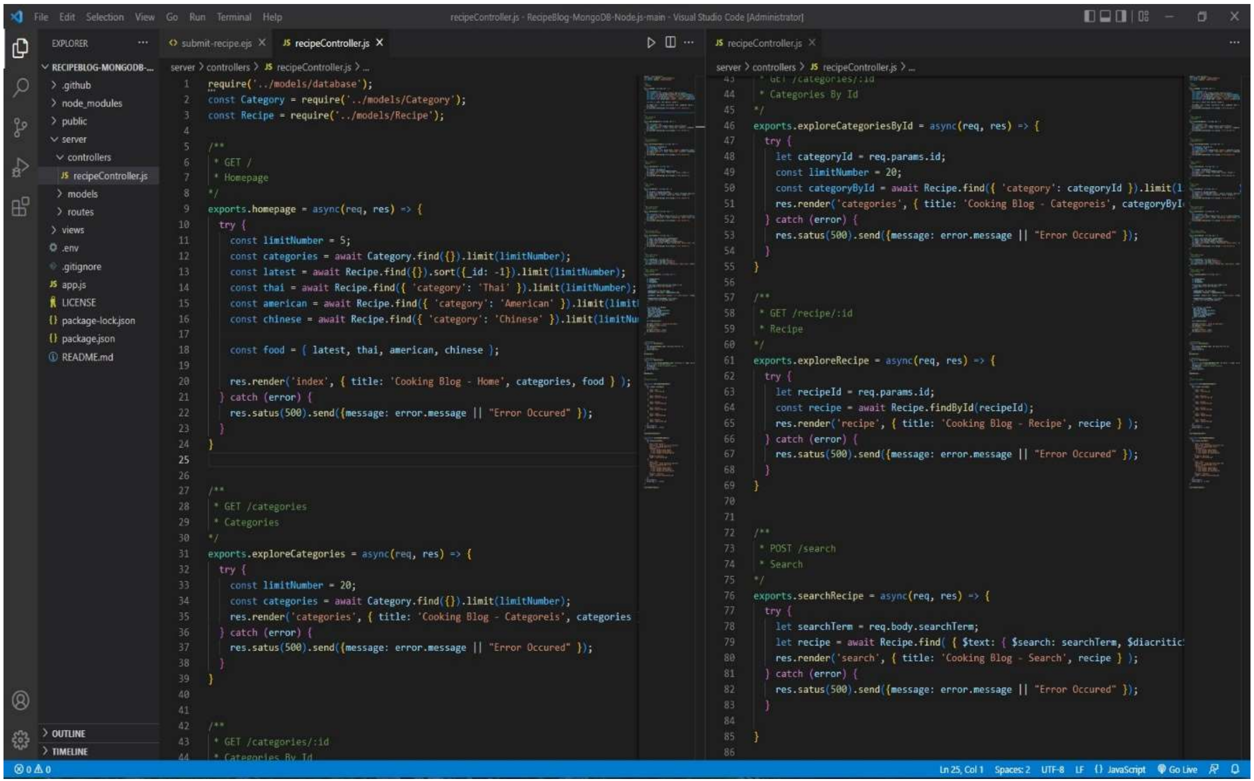
```javascript
let addIngredientsBtn = document.getElementById('addIngredientsBtn');
let ingredientList = document.querySelector('.ingredientList');
let ingredeintDiv = document.querySelectorAll('.ingredeintDiv')[0];

addIngredientsBtn.addEventListener('click', function(){
  let newIngredients = ingredeintDiv.cloneNode(true);
  let input = newIngredients.getElementsByTagName('input')[0];
  input.value = '';
  ingredientList.appendChild(newIngredients);
});
```

Script.js File



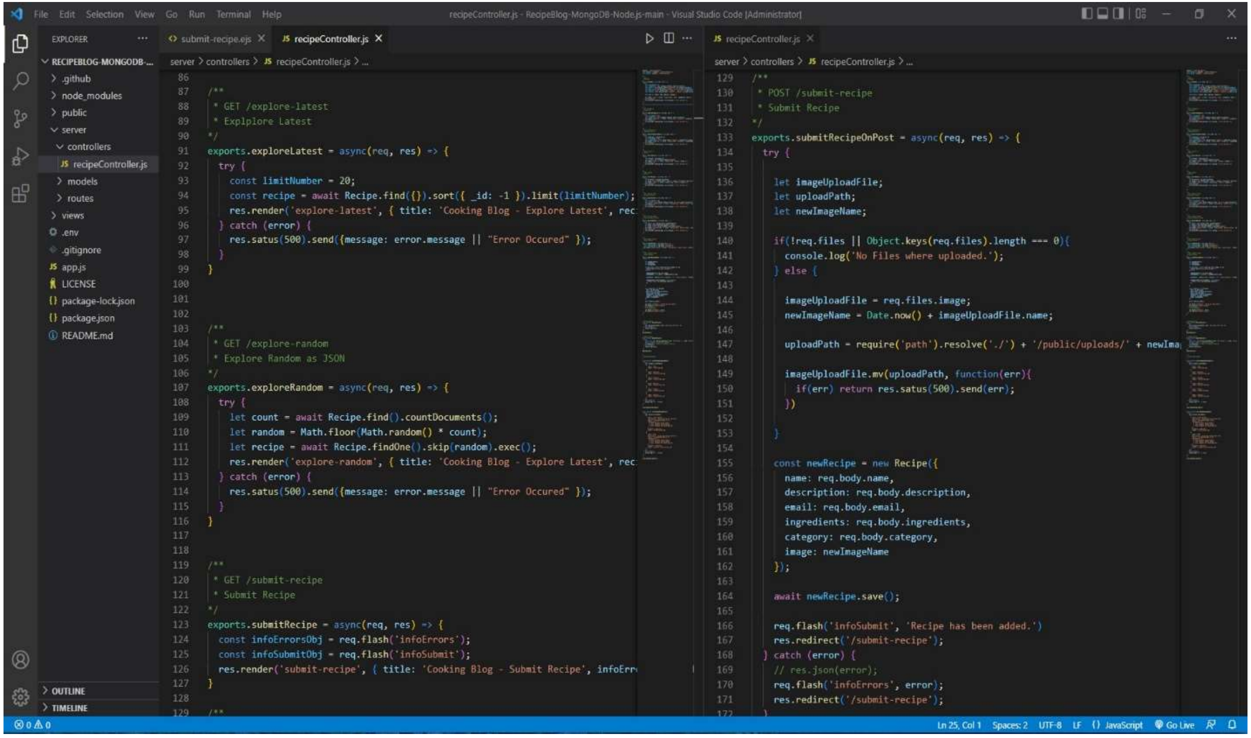
```css
:root {
  --bs-font-sans-serif: 'Rubik', sans-serif;
}

.container-xxl { max-width: 1530px; }

body {
  background-image: url("/img/i-like-food.svg");
  background-color: rgb(243,243,243);
}

a { color: var(--bs-dark); }


/* Category Card */
.category__link {
  text-align: center;
  display: block;
  text-decoration: none;
  transition: 250ms all;
}

.category__link:hover {
  transform: scale(1.1);
}

.category__img {
  display: flex;
  justify-content: center;
  align-items: center;
  height: 100px;
  margin-bottom: 10px;
  box-shadow: 0px 3px 6px rgba(0,0,0,.16);
  overflow: hidden;
  border-radius: 8px;
}

.category__img--large {
  height: 330px;
}

.category__img img {
  width: 100%;
  object-fit: cover;
```

CSS File


File Edit Selection View Go Run Terminal Help
recipeController.js - RecipeBlog-MongoDB-Node.js-main - Visual Studio Code [Administrator]
EXPLORER
RECIPEBLOG-MONGODB-...
.github
node_modules
public
server
controllers
recipeController.js
models
routes
views
.env
.gitignore
app.js
LICENSE
package-lock.json
package.json
README.md
OUTLINE
TIMELINE
submit-recipe.ejs
recipeController.js
server > controllers > recipeController.js > ...
Ln 25, Col 1 Spaces: 2 UTF-8 LF JavaScript Go Live

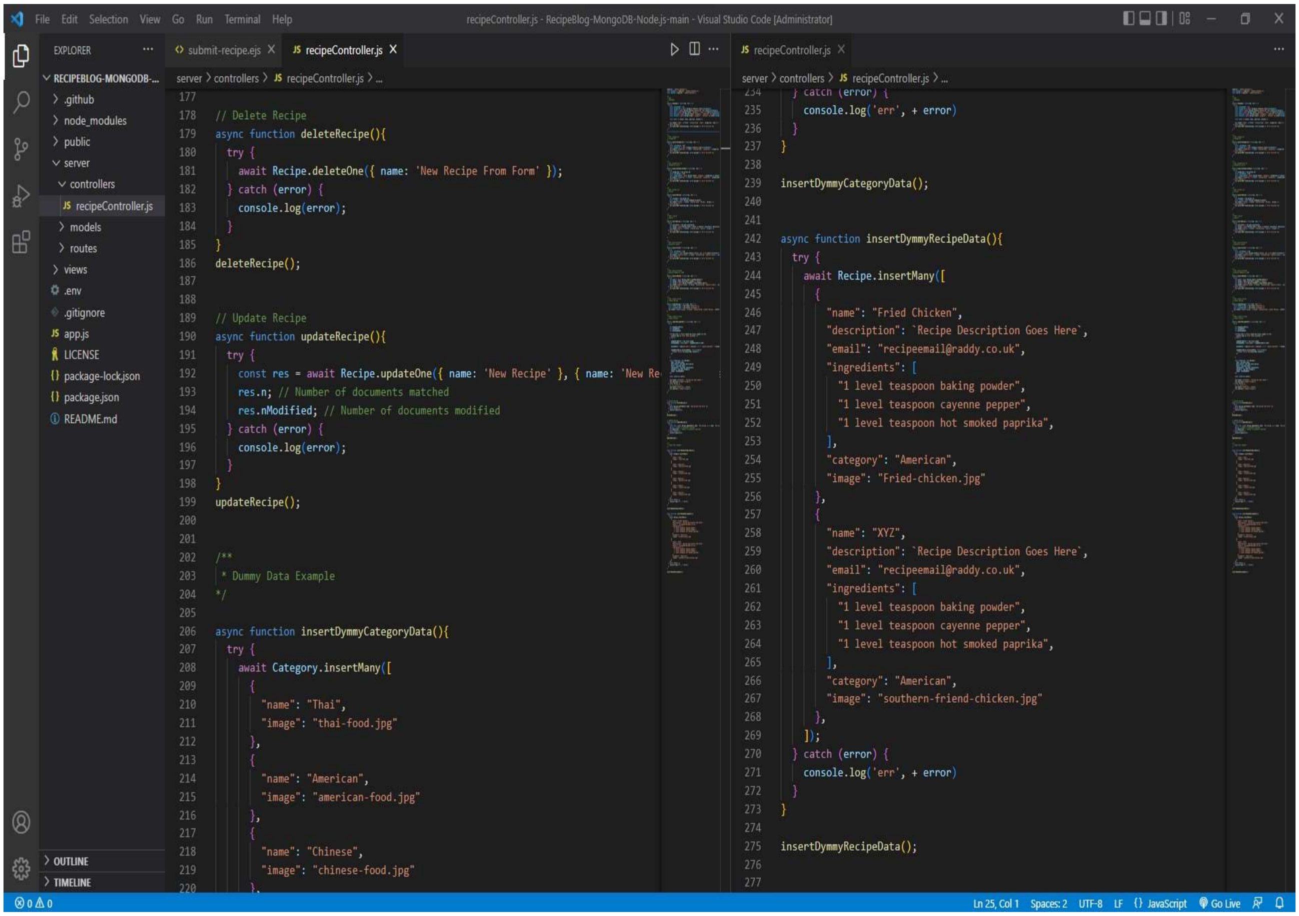
```javascript
require('../models/database');
const Category = require('../models/Category');
const Recipe = require('../models/Recipe');

/**
 * GET /
 * Homepage
*/
exports.homepage = async(req, res) => {
  try {
    const limitNumber = 5;
    const categories = await Category.find({}).limit(limitNumber);
    const latest = await Recipe.find({}).sort({_id: -1}).limit(limitNumber);
    const thai = await Recipe.find({ 'category': 'Thai' }).limit(limitNumber);
    const american = await Recipe.find({ 'category': 'American' }).limit(limit
    const chinese = await Recipe.find({ 'category': 'Chinese' }).limit(limitNu

    const food = { latest, thai, american, chinese };

    res.render('index', { title: 'Cooking Blog - Home', categories, food } );
  } catch (error) {
    res.satus(500).send({message: error.message || "Error Occured" });
  }
}



/**
 * GET /categories
 * Categories
*/
exports.exploreCategories = async(req, res) => {
  try {
    const limitNumber = 20;
    const categories = await Category.find({}).limit(limitNumber);
    res.render('categories', { title: 'Cooking Blog - Categoreis', categories
  } catch (error) {
    res.satus(500).send({message: error.message || "Error Occured" });
  }
}


/**
 * GET /categories/:id
 * Categories By Id
```

```javascript
 * Categories By Id
*/
exports.exploreCategoriesById = async(req, res) => {
  try {
    let categoryId = req.params.id;
    const limitNumber = 20;
    const categoryById = await Recipe.find({ 'category': categoryId }).limit(l
    res.render('categories', { title: 'Cooking Blog - Categoreis', categoryBy
  } catch (error) {
    res.satus(500).send({message: error.message || "Error Occured" });
  }
}

/**
 * GET /recipe/:id
 * Recipe
*/
exports.exploreRecipe = async(req, res) => {
  try {
    let recipeId = req.params.id;
    const recipe = await Recipe.findById(recipeId);
    res.render('recipe', { title: 'Cooking Blog - Recipe', recipe } );
  } catch (error) {
    res.satus(500).send({message: error.message || "Error Occured" });
  }
}


/**
 * POST /search
 * Search
*/
exports.searchRecipe = async(req, res) => {
  try {
    let searchTerm = req.body.searchTerm;
    let recipe = await Recipe.find( { $text: { $search: searchTerm, $diacritic
    res.render('search', { title: 'Cooking Blog - Search', recipe } );
  } catch (error) {
    res.satus(500).send({message: error.message || "Error Occured" });
  }

}

```


File Edit Selection View Go Run Terminal Help
recipeController.js - RecipeBlog-MongoDB-Node.js-main - Visual Studio Code [Administrator]
EXPLORER
RECIPEBLOG-MONGODB-...
.github
node_modules
public
server
controllers
recipeController.js
models
routes
views
.env
.gitignore
app.js
LICENSE
package-lock.json
package.json
README.md
OUTLINE
TIMELINE
submit-recipe.ejs
recipeController.js
server > controllers > recipeController.js > ...
Ln 25, Col 1 Spaces: 2 UTF-8 LF JavaScript Go Live

```javascript

/**
 * GET /explore-latest
 * Explplore Latest
*/
exports.exploreLatest = async(req, res) => {
  try {
    const limitNumber = 20;
    const recipe = await Recipe.find({}).sort({ _id: -1 }).limit(limitNumber);
    res.render('explore-latest', { title: 'Cooking Blog - Explore Latest', rec
  } catch (error) {
    res.satus(500).send({message: error.message || "Error Occured" });
  }
}



/**
 * GET /explore-random
 * Explore Random as JSON
*/
exports.exploreRandom = async(req, res) => {
  try {
    let count = await Recipe.find().countDocuments();
    let random = Math.floor(Math.random() * count);
    let recipe = await Recipe.findOne().skip(random).exec();
    res.render('explore-random', { title: 'Cooking Blog - Explore Latest', rec
  } catch (error) {
    res.satus(500).send({message: error.message || "Error Occured" });
  }
}


/**
 * GET /submit-recipe
 * Submit Recipe
*/
exports.submitRecipe = async(req, res) => {
  const infoErrorsObj = req.flash('infoErrors');
  const infoSubmitObj = req.flash('infoSubmit');
  res.render('submit-recipe', { title: 'Cooking Blog - Submit Recipe', infoErr
}

/**
```

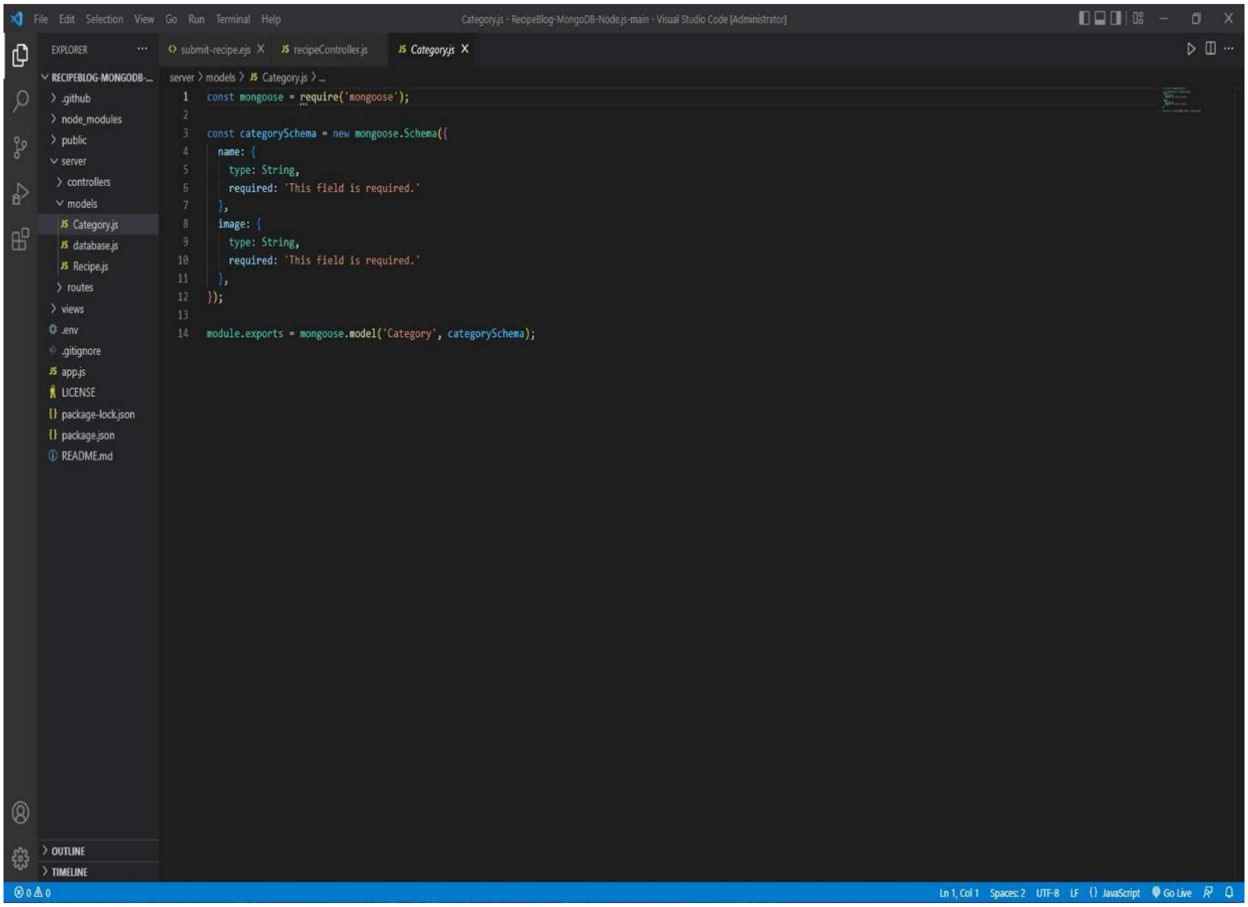
```javascript
/**
 * POST /submit-recipe
 * Submit Recipe
*/
exports.submitRecipeOnPost = async(req, res) => {
  try {

    let imageUploadFile;
    let uploadPath;
    let newImageName;

    if(!req.files || Object.keys(req.files).length === 0){
      console.log('No Files where uploaded.');
    } else {

      imageUploadFile = req.files.image;
      newImageName = Date.now() + imageUploadFile.name;

      uploadPath = require('path').resolve('./') + '/public/uploads/' + newIma

      imageUploadFile.mv(uploadPath, function(err){
        if(err) return res.satus(500).send(err);
      })

    }

    const newRecipe = new Recipe({
      name: req.body.name,
      description: req.body.description,
      email: req.body.email,
      ingredients: req.body.ingredients,
      category: req.body.category,
      image: newImageName
    });

    await newRecipe.save();

    req.flash('infoSubmit', 'Recipe has been added.')
    res.redirect('/submit-recipe');
  } catch (error) {
    // res.json(error);
    req.flash('infoErrors', error);
    res.redirect('/submit-recipe');
```


File Edit Selection View Go Run Terminal Help
recipeController.js - RecipeBlog-MongoDB-Node.js-main - Visual Studio Code [Administrator]
EXPLORER
RECIPEBLOG-MONGODB-...
.github
node_modules
public
server
controllers
recipeController.js
models
routes
views
.env
.gitignore
app.js
LICENSE
package-lock.json
package.json
README.md
OUTLINE
TIMELINE
submit-recipe.ejs
recipeController.js
server > controllers > recipeController.js > ...
Ln 25, Col 1 Spaces: 2 UTF-8 LF {} JavaScript Go Live

```javascript

// Delete Recipe
async function deleteRecipe(){
  try {
    await Recipe.deleteOne({ name: 'New Recipe From Form' });
  } catch (error) {
    console.log(error);
  }
}
deleteRecipe();


// Update Recipe
async function updateRecipe(){
  try {
    const res = await Recipe.updateOne({ name: 'New Recipe' }, { name: 'New Re
    res.n; // Number of documents matched
    res.nModified; // Number of documents modified
  } catch (error) {
    console.log(error);
  }
}
updateRecipe();


/**
 * Dummy Data Example
*/

async function insertDymmyCategoryData(){
  try {
    await Category.insertMany([
      {
        "name": "Thai",
        "image": "thai-food.jpg"
      },
      {
        "name": "American",
        "image": "american-food.jpg"
      },
      {
        "name": "Chinese",
        "image": "chinese-food.jpg"
      },
```

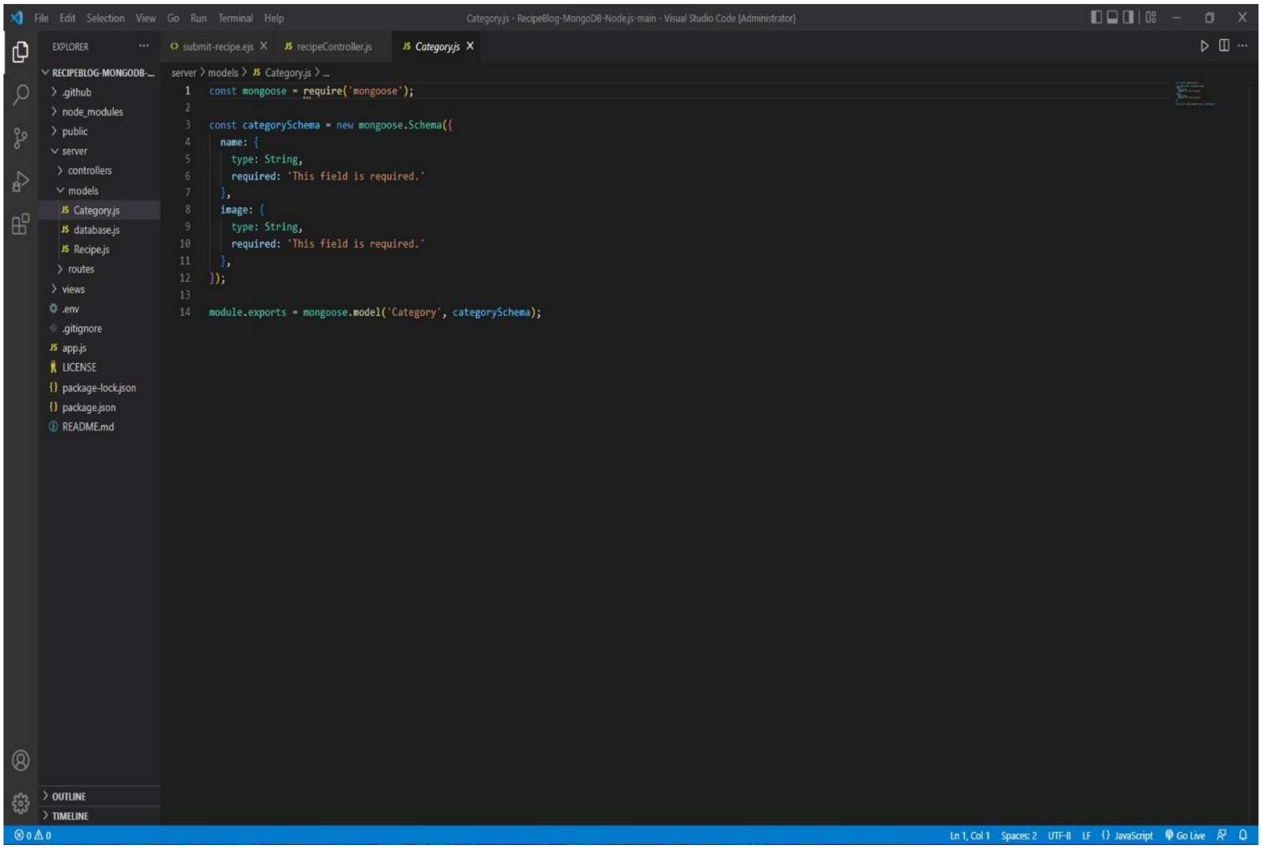
```javascript
  } catch (error) {
    console.log('err', + error)
  }
}

insertDymmyCategoryData();


async function insertDymmyRecipeData(){
  try {
    await Recipe.insertMany([
      {
        "name": "Fried Chicken",
        "description": `Recipe Description Goes Here`,
        "email": "recipeemail@raddy.co.uk",
        "ingredients": [
          "1 level teaspoon baking powder",
          "1 level teaspoon cayenne pepper",
          "1 level teaspoon hot smoked paprika",
        ],
        "category": "American",
        "image": "Fried-chicken.jpg"
      },
      {
        "name": "XYZ",
        "description": `Recipe Description Goes Here`,
        "email": "recipeemail@raddy.co.uk",
        "ingredients": [
          "1 level teaspoon baking powder",
          "1 level teaspoon cayenne pepper",
          "1 level teaspoon hot smoked paprika",
        ],
        "category": "American",
        "image": "southern-friend-chicken.jpg"
      },
    ]);
  } catch (error) {
    console.log('err', + error)
  }
}

insertDymmyRecipeData();


```

RecipeController.js File (for recipe submission)



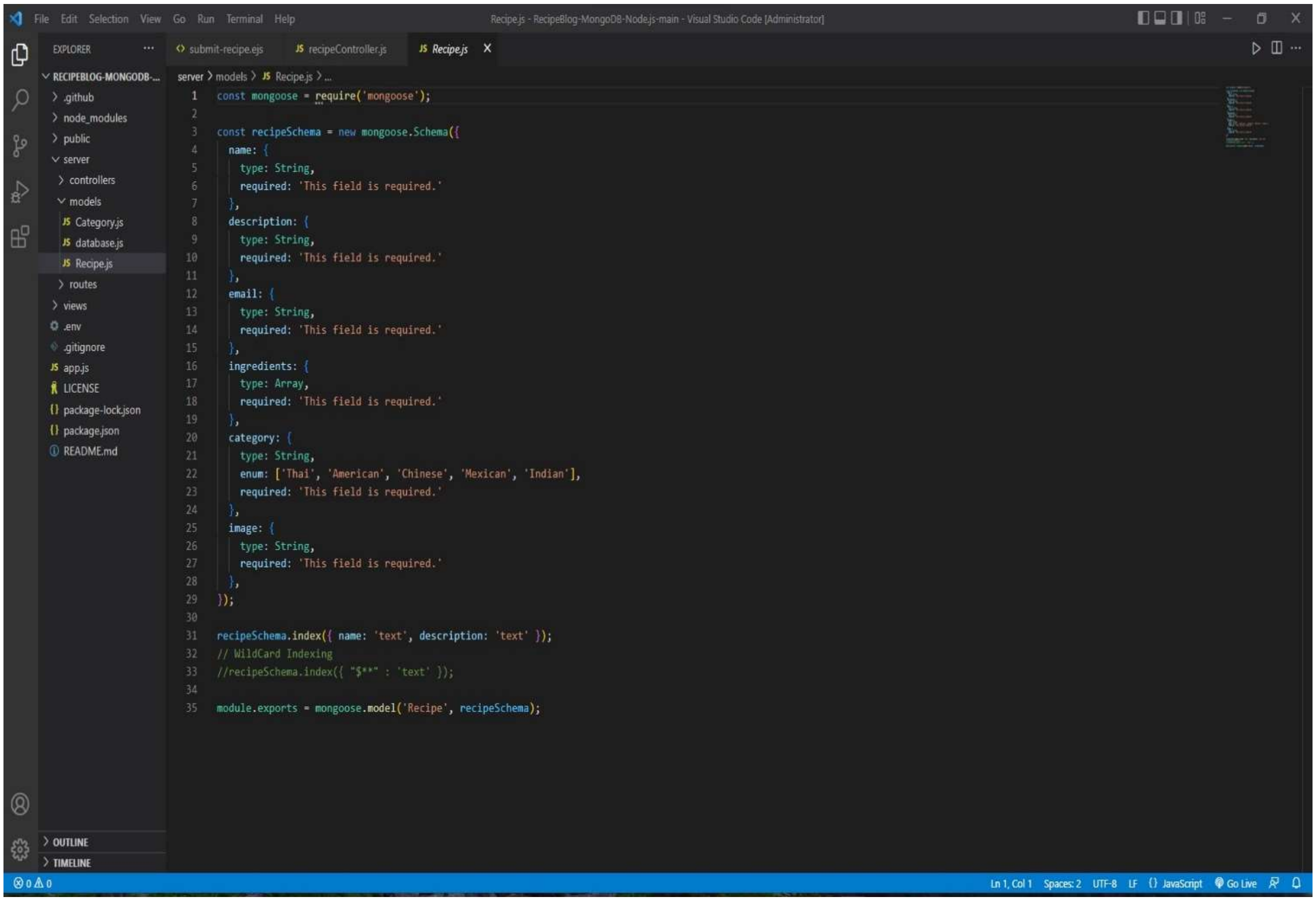
```javascript
const mongoose = require('mongoose');

const categorySchema = new mongoose.Schema({
  name: {
    type: String,
    required: 'This field is required.'
  },
  image: {
    type: String,
    required: 'This field is required.'
  },
});

module.exports = mongoose.model('Category', categorySchema);
```

Category.js File (for grouping similar cuisines)



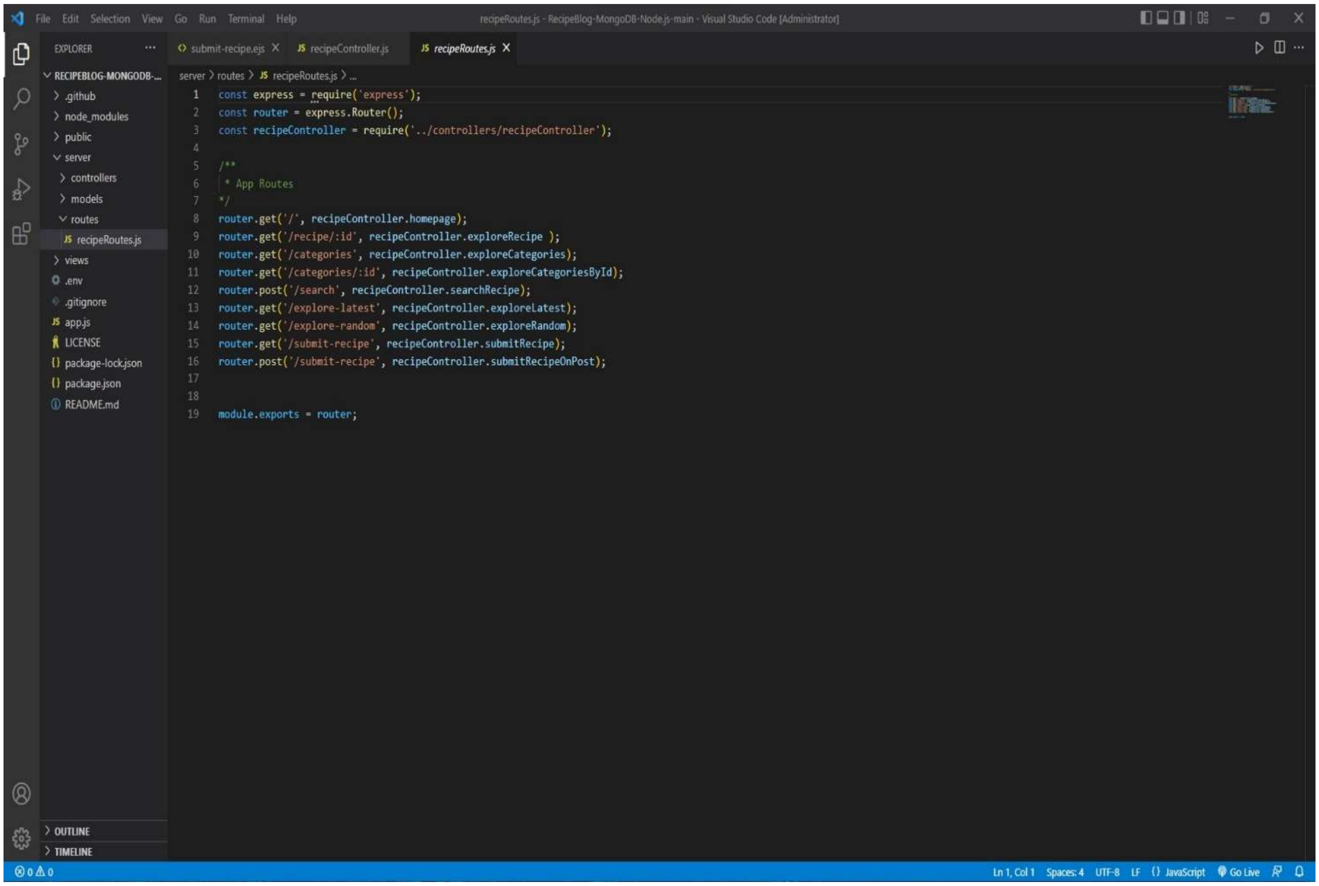
```javascript
const mongoose = require('mongoose');

const categorySchema = new mongoose.Schema({
  name: {
    type: String,
    required: 'This field is required.'
  },
  image: {
    type: String,
    required: 'This field is required.'
  },
});

module.exports = mongoose.model('Category', categorySchema);
```

MongoDB Connection



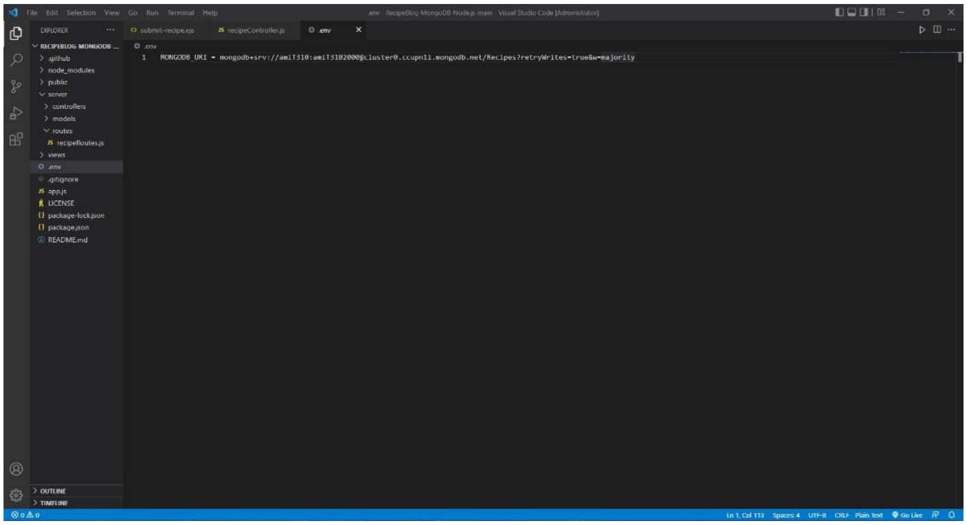
```
const mongoose = require('mongoose');

const recipeSchema = new mongoose.Schema({
  name: {
    type: String,
    required: 'This field is required.'
  },
  description: {
    type: String,
    required: 'This field is required.'
  },
  email: {
    type: String,
    required: 'This field is required.'
  },
  ingredients: {
    type: Array,
    required: 'This field is required.'
  },
  category: {
    type: String,
    enum: ['Thai', 'American', 'Chinese', 'Mexican', 'Indian'],
    required: 'This field is required.'
  },
  image: {
    type: String,
    required: 'This field is required.'
  },
});

recipeSchema.index({ name: 'text', description: 'text' });
// WildCard Indexing
//recipeSchema.index({ "$**" : 'text' });

module.exports = mongoose.model('Recipe', recipeSchema);
```

Recipe-Submission Form Validation



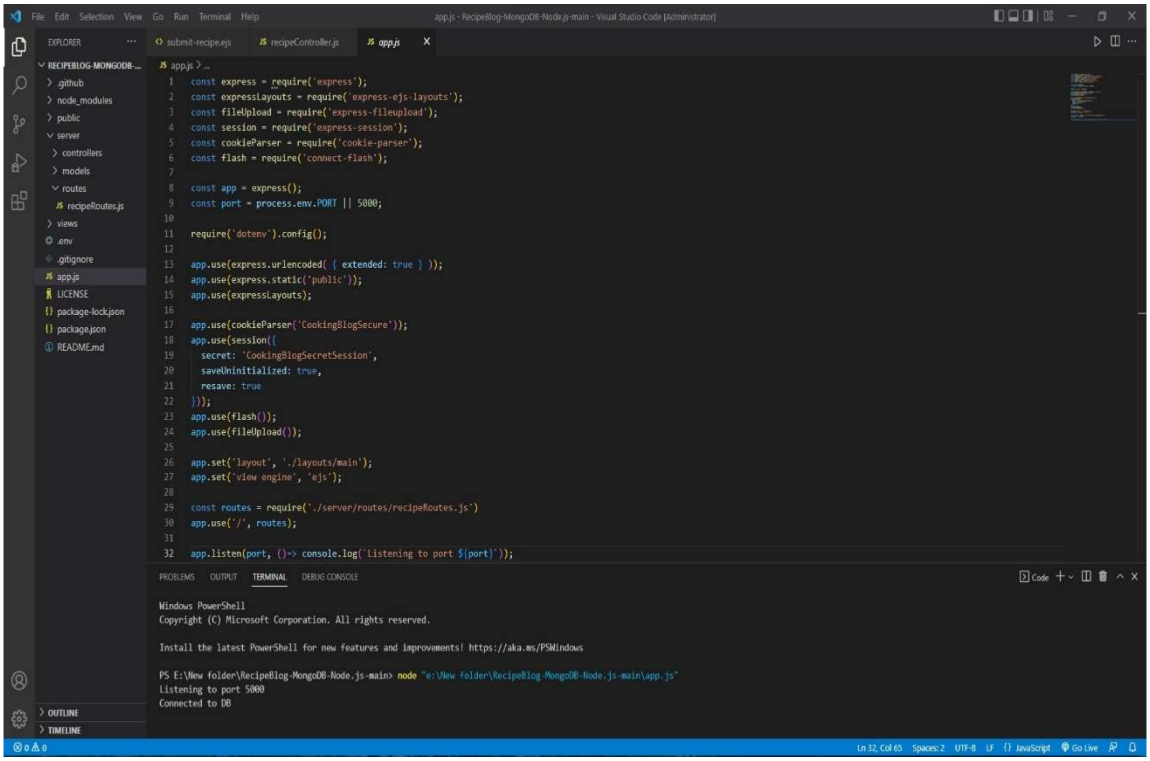
```javascript
const express = require('express');
const router = express.Router();
const recipeController = require('../controllers/recipeController');

/**
 * App Routes
*/
router.get('/', recipeController.homepage);
router.get('/recipe/:id', recipeController.exploreRecipe );
router.get('/categories', recipeController.exploreCategories);
router.get('/categories/:id', recipeController.exploreCategoriesById);
router.post('/search', recipeController.searchRecipe);
router.get('/explore-latest', recipeController.exploreLatest);
router.get('/explore-random', recipeController.exploreRandom);
router.get('/submit-recipe', recipeController.submitRecipe);
router.post('/submit-recipe', recipeController.submitRecipeOnPost);


module.exports = router;
```

Route for Recipes



```
MONGODB_URI = mongodb+srv://amiT310:amiT3102000@cluster0.ccupn1i.mongodb.net/Recipes?retryWrites=true&w=majority
```

MongoDB User ID & Password for connecting database



```javascript
const express = require('express');
const expressLayouts = require('express-ejs-layouts');
const fileUpload = require('express-fileupload');
const session = require('express-session');
const cookieParser = require('cookie-parser');
const flash = require('connect-flash');

const app = express();
const port = process.env.PORT || 5000;

require('dotenv').config();

app.use(express.urlencoded( { extended: true } ));
app.use(express.static('public'));
app.use(expressLayouts);

app.use(cookieParser('CookingBlogSecure'));
app.use(session({
  secret: 'CookingBlogSecretSession',
  saveUninitialized: true,
  resave: true
}));
app.use(flash());
app.use(fileUpload());

app.set('layout', './layouts/main');
app.set('view engine', 'ejs');

const routes = require('./server/routes/recipeRoutes.js')
app.use('/', routes);

app.listen(port, ()=> console.log(`Listening to port ${port}`));
```

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS E:\New folder\RecipeBlog-MongoDB-Node.js-main> node "e:\New folder\RecipeBlog-MongoDB-Node.js-main\app.js"
Listening to port 5000
Connected to DB
```

Successful Connection to port & Database

Conclusion

It has been a great pleasure for me to work on this exciting and challenging project. This project proved good for me as it provided practical knowledge of JavaScript (for functionality), Express (to set up middleware's to respond to HTTP Requests) and MongoDB (for storing data), CSS (for style) and HTML web-based application. It also provides knowledge about the latest technology used in developing web enabled application will be great demand in future. This will provide better opportunities and guidance in future in developing projects independently.

BENEFITS:

The project is identified by the merits of the system offered to the user. The merits of this project are as follows: -

- It's a web-enabled project.
- This project offers user to enter the data through simple and interactive forms. This is very helpful for the user to enter the desired information through so much simplicity.
- There are checks on every stage of any form's submission, so that the user cannot enter the invalid data, which can create problems at later date.
- Sometimes the user finds in the later stages of using project that he needs to update some of the information that he entered earlier. There are options for him by which he can update the records. Moreover, there is restriction for his that he cannot change the primary data field. This keeps the validity of the data to longer extent.
- User is provided the option of monitoring the records he entered earlier. He can see the desired records with the variety of options provided by him.
- Data storage and retrieval will become faster and easier to maintain because data is stored in a systematic manner and in a single database.
- Easier and faster data transfer through latest technology associated with the computer and communication.
- Through these features it will increase the efficiency, accuracy and transparency,

REFERENCES

- Stack Overflow
- Geeks For Geeks
- Coding Ninjas for JavaScript
- MongoDB Documentations
- Documentations of Express
- Documentations of Node.js
- W3schools for HTML & CSS