A FIELD PROJECT REPORT ON

**KITCHEN ALCHEMY**

Submitted in partial fulfilment of the requirements for the award of the degree

**BACHELOR OF TECHNOLOGY**

in

**COMPUTER SCIENCE & ENGINEERING**

Submitted by

|  |  |
| --- | --- |
| A.Rama Krishna | (231FA04477) |
| Mohammed Rawhan Ramzi | (231FA04516) |
| R.Chandana | (231FA04522) |
| Inavolu Vathsav | (231FA04528) |
|  |  |



Department of Computer Science & Engineering

School of engineering

Vignan’s Foundation for Science, Technology and Research (Deemed to be University) Vadlamudi, Guntur, Andhra Pradesh-522213, India

MARCH – 20**25**



**CERTIFICATE**

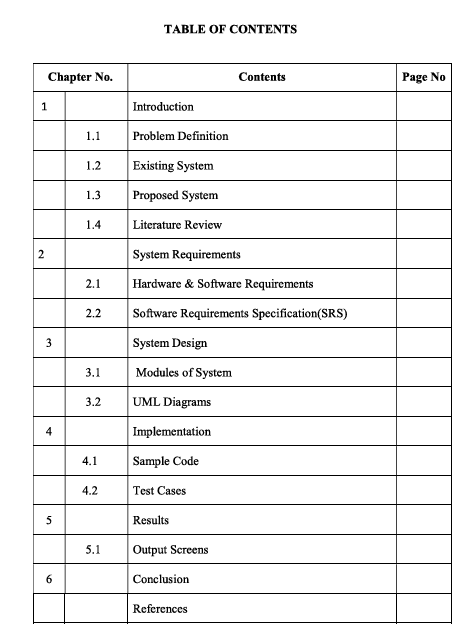
This is to certify that the field project entitled “KITCHEN ALCHEMY” being submitted by (A.Rama Krishna & 231FA04477), (Mohammed Rawhan Ramzi & 231FA04516), (R.Chandana & 231FA04522), and (Inavolu Vathsav & 231FA04528) in partial fulfilment of Bachelor of Technology in the Department of Computer Science & Engineering, Vignan’s Foundation For Science Technology & Research (Deemed to be University), Vadlamudi, Guntur District, Andhra Pradesh, India, is a bonafide work carried out by them under my guidance and supervision.

|  |  |
| --- | --- |
| **Head of the Department** | **Guide** |

**DECLARATION**

We hereby declare that our project work described in the field project titled “KITCHEN ALCHEMY” which is being submitted by us for the partial fulfilment in the department of Computer Science & Engineering, Vignan’s Foundation for Science, Technology and Research (Deemed to be University), Vadlamudi, Guntur, Andhra Pradesh, and the result of investigations are carried out by us under the guidance of (Name of the Guide)

|  |  |  |
| --- | --- | --- |
| A.Rama Krishna | (231FA04477) | Signature |
| Mohammed Rawhan Ramzi | (231FA04516) | Signature |
| R.Chandana | (231FA04522) | Signature |
| Inavolu Vathsav | (231FA04528) | Signature |

****

**I : INTRODUCTION:**

Our project titled as “KITCHEN ALCHEMY” centers on a dynamic recipe finder web page designed to streamline meal planning and promote healthier eating habits. By allowing users to enter available ingredients, the application intelligently curates a list of recipes complete with detailed nutritional information. This innovative tool not only simplifies the process of deciding what to cook but also empowers users to make informed dietary choices based on the nutrient profiles provided. In the following document, we will delve into the project’s design, development process, and the key technologies that underpin its functionality.

* 1. **: Problem Definition**

In today’s fast-paced world, many individuals struggle with deciding what to cook based on the ingredients they have on hand. This often leads to food wastage, unhealthy eating habits, or unnecessary grocery trips. Additionally, people are becoming more conscious of their dietary choices and nutritional intake but lack an easy way to assess the nutritional value of their meals.

To address these challenges, **Kitchen Alchemy** provides an intuitive and efficient solution. Developed using **HTML, CSS, JavaScript, and various APIs**, this web-based application enables users to input available ingredients and receive a curated list of recipes they can prepare. Additionally, it provides detailed **nutritional information** for each recipe, helping users make informed dietary decisions. By leveraging technology, Kitchen Alchemy simplifies meal planning, reduces food wastage, and promotes healthier eating habits.

* 1. **: Existing Systems**

In the current scenario, individuals often rely on **manual recipe searches** through cookbooks, food blogs, or generic search engines to find meals they can prepare with available ingredients. This process can be **time-consuming and inefficient**, as users have to sift through multiple sources to find relevant recipes. Additionally, most traditional methods **do not provide nutritional insights**, leaving users unaware of the health aspects of their meals.

Some existing digital solutions, such as general recipe websites and cooking apps, allow users to browse recipes but **lack ingredient-based filtering** or require manual input for each ingredient, making the experience cumbersome. Furthermore, these platforms often do not integrate **real-time nutritional analysis**, making it difficult for health-conscious users to track their dietary intake.

Due to these limitations, there is a clear need for a **more streamlined, intelligent, and user-friendly solution** that not only suggests recipes based on available ingredients but also provides **accurate nutritional information**—a gap that **Kitchen Alchemy** aims to bridge.

* 1. **Proposed System**

**Kitchen Alchemy** offers an intelligent and user-friendly solution to simplify meal planning by providing **recipe suggestions based on available ingredients** while also offering **nutritional insights**. Unlike traditional methods and existing digital solutions, our system integrates **modern web technologies (HTML, CSS, JavaScript) and APIs** to enhance user experience and efficiency.

With **Kitchen Alchemy**, users can:

* **Enter available ingredients**, and the system will generate a list of recipes they can prepare.
* **Access detailed nutritional information** for each recipe, helping them make informed dietary choices.
* **Reduce food waste** by utilizing ingredients they already have instead of letting them spoil.
* **Save time and effort** by eliminating the need for extensive online searches or cookbook references.
  1. **Literature Review**

The concept of **recipe recommendation systems** has gained significant attention in recent years, particularly with the increasing reliance on technology for meal planning and dietary management. Various studies and existing applications have explored different approaches to recipe suggestions, ingredient-based filtering, and nutritional analysis.

**1. Nutritional Analysis and Healthy Eating**

With the increasing focus on **health and nutrition**, many individuals seek ways to track their dietary intake. Research suggests that **nutritional awareness** plays a crucial role in maintaining a balanced diet and preventing health issues such as obesity and diabetes. Some applications provide **calorie tracking and nutritional breakdowns**, but they often require separate tools or additional user effort. **Kitchen Alchemy integrates nutritional data within the recipe finder**, offering users a **comprehensive solution** for both meal planning and health tracking.

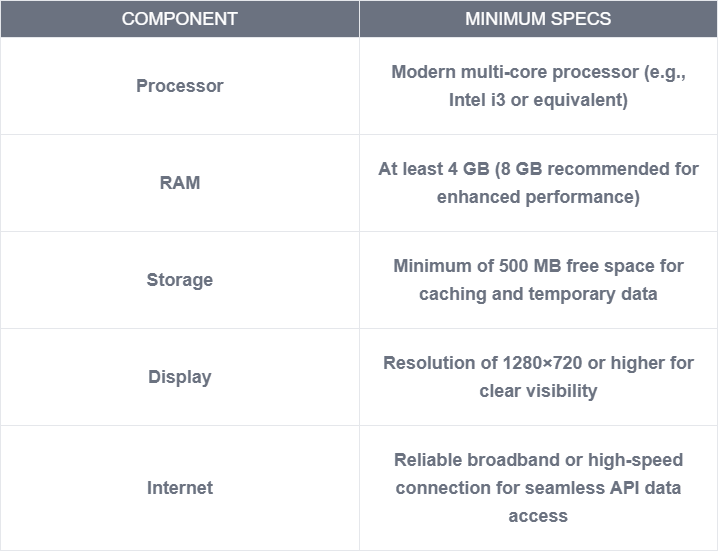
**2. Web Technologies for Recipe Applications**

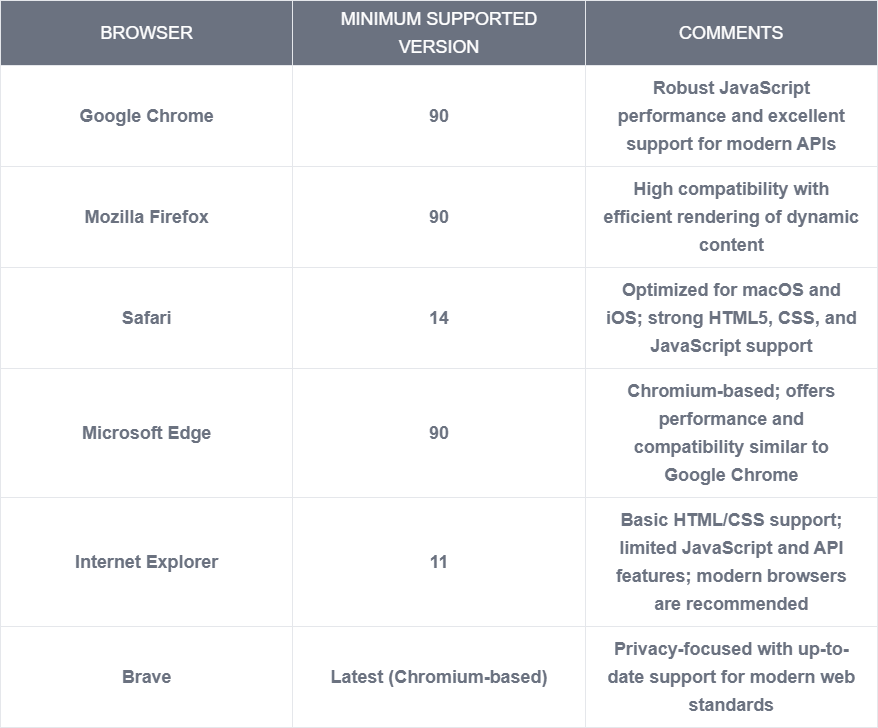
The use of **HTML, CSS, and JavaScript** in web development has allowed for the creation of **interactive and user-friendly interfaces**. Additionally, **API integration** has enabled access to vast recipe databases and nutritional information, making web-based solutions more powerful and efficient. Research on API-based systems indicates that **real-time data fetching and dynamic content generation** improve user experience, which is a key focus of **Kitchen Alchemy**.

**Conclusion:** The literature review indicates that while various solutions exist for **recipe recommendations, ingredient-based search, and nutritional analysis**, they often lack a **seamless and integrated approach**.

**2 : SYSTEM REQUIREMENTS**

**2.1 : Hardware & Software Requirements**

****



**2.2 : Software Requirements Specification ( SRS )**

This recipe finder web application, built with HTML5, CSS, and JavaScript, aims to deliver a seamless recipe search and display experience. It must function reliably across modern web browsers, with a minimum support baseline of Chrome 80, Firefox 75, Safari 13, and Edge 80. The application's core feature, recipe search, shall allow users to find recipes by ingredient keywords, recipe titles, and cuisine types. Upon search initiation, JavaScript will asynchronously fetch data from an external recipe API, presenting results in a structured format: recipe names, ingredient lists, step-by-step instructions, and images where available. The user interface must be responsive, adapting to various screen sizes across desktop, tablet, and mobile devices, ensuring consistent usability. JavaScript will handle all data fetching and dynamic content rendering, with a focus on minimizing API calls and optimizing page load times. The application must gracefully handle API errors and no-result searches, providing clear user feedback. Security considerations mandate that any API keys or sensitive data remain strictly client-side. The front-end development, constrained to HTML5, CSS, and vanilla JavaScript, will prioritize clean, semantic markup and efficient CSS styling. The website must maintain a user-friendly and intuitive layout. The data fetched from the API must be presented in a way that is easily readable, and the application should be able to handle a large amount of recipe data, and display it in a timely manner within the specified browser versions.

**System Design: Client-Side Recipe Finder Application**

1. Overview:

This system design outlines the architecture for a recipe finder web application that utilizes HTML, CSS, and JavaScript for the front-end, with all data fetching handled directly by the client-side JavaScript from an external recipe API.

The application's core functionality revolves around a search interface that allows users to input recipe names, ingredients, or cuisine types. Upon initiating a search, the client-side JavaScript constructs a query and sends an asynchronous request directly to the external recipe API. The API, in turn, responds with recipe data formatted in JSON, which the JavaScript then parses and utilizes to dynamically generate the recipe display within the user's browser. This display includes essential recipe information such as the recipe name, ingredient list, preparation instructions, and accompanying images, ensuring a comprehensive and visually appealing presentation

**Architecture Diagram:**

A diagram of a user flow

AI-generated content may be incorrect.

**System Components:**

* **Client (User):**
  + **Technologies:** HTML5, CSS3, JavaScript.
  + **Responsibilities:**
    - Presents the user interface (UI) for searching and displaying recipes.
    - Handles user interactions (search queries).
    - Makes direct API requests to the external recipe API.
    - Handles API responses, including error handling.
    - Formats and renders recipe data in the browser.
    - Handles the user interaction in an asynchronous manner.
* **3.2 External Recipe API:**
  + **Responsibilities:**
    - Provides access to a database of recipe information.
    - Responds to API requests with recipe data in JSON format.

**4. Data Flow:**

1. **User Input:** The user enters a search query in the client-side UI.
2. **Client Request:** The client-side JavaScript constructs an API request to the external recipe API, including the user's search query.
3. **API Request:** The client-side JavaScript sends the API request to the external recipe API.
4. **API Response:** The external recipe API responds with recipe data in JSON format.
5. **Client Rendering:** The client-side JavaScript receives the JSON response, formats the data as needed, and dynamically updates the UI to display the recipe results.

**5. Technologies:**

* **Front-end:** HTML5, CSS3, JavaScript (fetch).
* **Data Format:** JSON.

**3.1 Modules of System:**

**User Interface (UI) Module:**

* Responsibilities:
  + Handles all user interactions, including search input and result display.
  + Constructs and manages the HTML structure of the application.
  + Applies CSS styling for visual presentation and responsiveness.
  + Provides a clear and intuitive interface for users to interact with the application.
  + Handles user inputs and actions.
  + Displays loading indicators and error messages.

**API Communication Module**:

* Responsibilities:
  + Constructs and sends API requests to the external recipe API.
  + Parses and processes JSON responses from the API.
  + Manages asynchronous API calls using fetch or axios.
  + Handles API rate limiting and error responses.
  + Sanitizes and validates the data that is received from the API.

**Data Processing and Rendering Module:**

* Responsibilities:
  + Transforms the JSON data received from the API into a format suitable for display.
  + Dynamically generates HTML elements to display recipe information.
  + Populates the UI with recipe data, including names, ingredients, instructions, and images.
  + Optimizes data rendering for performance.
  + Handles edge cases where data is missing or invalid.

**Search Logic Module:**

* Responsibilities:
  + Handles the logic of the search.
  + Manages the search queries.
  + Triggers the API communication module.
  + Handles empty search results

3.2 **Uml Diagram**

A diagram of a flowchart

AI-generated content may be incorrect.

**Sample Code:**

<!DOCTYPE html>

<html>

<head>

<title>Recipe Finder</title>

<style>

body { font-family: sans-serif; }

#results { margin-top: 20px; }

.recipe { border: 1px solid #ccc; padding: 10px; margin-bottom: 10px; }

</style>

</head>

<body>

<input type="text" id="searchInput" placeholder="Enter ingredients or recipe name">

<select id="searchType">

<option value="ingredients">Ingredients</option>

<option value="name">Recipe Name</option>

</select>

<button id="searchButton">Search</button>

<div id="results"></div>

<script>

const searchInput = document.getElementById('searchInput');

const searchButton = document.getElementById('searchButton');

const resultsDiv = document.getElementById('results');

const searchType = document.getElementById('searchType');

searchButton.addEventListener('click', () => {

const query = searchInput.value;

const type = searchType.value;

resultsDiv.innerHTML = 'Loading...';

if (type === 'ingredients') {

fetch(`https://api.spoonacular.com/recipes/findByIngredients?ingredients=${query}&number=5&apiKey=YOUR\_SPOONACULAR\_API\_KEY`)

.then(response => response.json())

.then(data => {

if (data && data.length > 0) {

resultsDiv.innerHTML = '';

data.forEach(recipe => {

resultsDiv.innerHTML += `

<div class="recipe">

<h3>${recipe.title}</h3>

<img src="${recipe.image}" alt="${recipe.title}" style="max-width: 200px;">

</div>

`;

});

} else {

resultsDiv.innerHTML = 'No recipes found for those ingredients.';

}

})

.catch(error => {

console.error('Error fetching data:', error);

resultsDiv.innerHTML = 'An error occurred while fetching data.';

});

} else if (type === 'name') {

fetch(`https://api.api-ninjas.com/v1/recipe?query=${query}`,{

headers: {

'X-Api-Key': 'YOUR\_NINJA\_API\_KEY'

}

})

.then(response => response.json())

.then(data => {

if (data && data.length > 0) {

resultsDiv.innerHTML = '';

data.forEach(recipe => {

resultsDiv.innerHTML += `

<div class="recipe">

<h3>${recipe.title}</h3>

<p>Instructions: ${recipe.instructions}</p>

</div>

`;

});

} else {

resultsDiv.innerHTML = 'No recipes found for that name.';

}

})

.catch(error => {

console.error('Error fetching data:', error);

resultsDiv.innerHTML = 'An error occurred while fetching data.';

});

}

});

</script>

</body>

</html>

**Test Cases**:

**TC\_1.1: Valid Ingredients Search**

* **Description:** Verify that the application returns recipes when valid ingredients are entered.
* **Steps:**
  1. Open the recipe finder application.
  2. Enter "chicken, rice" into the search input field.
  3. Select "Ingredients" from the search type dropdown.
  4. Click the "Search" button.

**Expected Result:** A list of recipes containing chicken and rice is displayed.

**TC\_1.2: No Matching Ingredients**

* **Description:** Verify that the application displays an appropriate message when no recipes match the entered ingredients.
* **Steps:**
  1. Open the recipe finder application.
  2. Enter "unicorn tears" into the search input field.
  3. Select "Ingredients" from the search type dropdown.
  4. Click the "Search" button.
* **Expected Result:** The message "No recipes found for those ingredients." is displayed.

**TC\_1.3: Empty Ingredients Search**

* **Description:** Verify how the application handles an empty ingredient search.
* **Steps:**
  1. Open the recipe finder application.
  2. Leave the search input field empty.
  3. Select "Ingredients" from the search type dropdown.
  4. Click the "Search" button.
* **Expected Result:** Either an error message is displayed, or the application handles this gracefully (e.g., does nothing).

**RESULTS:**

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

**CONCLUSION:**

This project created a client-side recipe finder using HTML, CSS, and JavaScript, pulling data from Spoonacular and Recipe Ninja APIs. It handles ingredient and name-based searches, displaying results dynamically. Basic error handling and UI feedback are implemented. Testing primarily involves manual browser verification. Security concerns, especially API key exposure, highlight the need for server-side proxying in production. While functional, the application requires further development for production use, including enhanced UI/UX, robust testing, and secure API key management.

**References:**

**Books:**

* HTML & CSS: Design and Build Websites
* Eloquent JavaScript
* You Don't Know JS (series)
* JavaScript: The Definitive Guide
* CSS Secrets

**Websites:**

* MDN Web Docs
* W3Schools
* GeeksforGeeks
* FreeCodeCamp