**CHAPTER 1**

**INTRODUCTION**

Recipe app displays the list of food recipes on the web page, as requested by the user. To display a list of recipes, we require an Application Program Interface (API) which will fetch all the data from a database which is hosted somewhere else. A programmer writing an application program can make a request to the Operating System using API (using graphical user interface or command interface). It is a set of routines, protocols and tools for building software and applications.

The API used in this application is the Food2Fork’s recipe API. Food2Fork offers an API which exposes its powerful recipe discovery functions for the application to use. The API gives access to their ever expanding recipe database, powerful ingredient search function, and social-media based ranking algorithm. Any request will return a maximum of 30 results.

* 1. **Introduction to recipe app**

Recipe app displays the list of food recipes on the web page, as requested by the user. To display a list of recipes, we require an Application Program Interface (API) which will fetch all the data from a database which is hosted somewhere else. A programmer writing an application program can make a request to the Operating System using API (using graphical user interface or command interface). It is a set of routines, protocols and tools for building software and applications.

**Components**in React are the reusable piece of code that is used to help in building web apps. In simple language, they are more like Lego parts with which we can create different shape toys. Components in ReactJS are used to create smart UI which are reusable piece of code and can be utilize anywhere in the application.

The API used in this application is the Food2Fork’s recipe API. Food2Fork offers an API which exposes its powerful recipe discovery functions for the application to use. The API gives access to their ever expanding recipe database, powerful ingredient search function, and social-media based ranking algorithm. Any request will return a maximum of 30 results. Food2Fork uses a proprietary ranking algorithm based on of how often a recipe is shared across various social-media sites including Facebook, Pinterest, and Twitter.

Node.js is an open source, cross-platform runtime environment for developing server-side and networking applications. Node.js applications are written in JavaScript, and can be run within the Node.js runtime on OS X, Microsoft Windows, and Linux.

* 1. **Overview**

The food listing application displays food recipes based on the search key. The number of recipes which are displayed can be changed based on the requirement.

React components implement a render() method that takes input data and returns what to display. This example uses an XML-like syntax called JSX. Input data that is passed into the component can be accessed by render() via this.props.

In addition to taking input data (accessed via this.props), a component can maintain internal state data (accessed via this.state). When a component’s state data changes, the rendered markup will be updated by re-invoking render().

Using props and state, we can put together a small Todo application. This example uses state to track the current list of items as well as the text that the user has entered. Although event handlers appear to be rendered inline, they will be collected and implemented using event delegation.

React allows you to interface with other libraries and frameworks. This example uses **remarkable,** an external Markdown library, to convert the <textarea>’s value in real time.

* 1. **Cloud computing**

Cloud computing is an information technology (IT) paradigm that enables ubiquitous access to shared pools of configurable system resources and higher-level services that can be rapidly provisioned with minimal management effort, often over the Internet. Cloud computing relies on sharing of resources to achieve coherence and economies of scale, similar to a public utility. Third-party clouds enable organizations to focus on their core businesses instead of expending resources on computer infrastructure and maintenance. Advocates note that cloud computing allows companies to avoid or minimize up-front IT infrastructure costs. Proponents also claim that cloud computing allows enterprises to get their applications up and running faster, with improved manageability and less maintenance, and that it enables IT teams to more rapidly adjust resources to meet fluctuating and unpredictable demand. Cloud providers typically use a “pay-as-you-go” model, which can lead to unexpected operating expenses if administrators are not familiarized with cloud-pricing models.

The goal of cloud computing is to allow users to take benefit from all of these technologies, without the need for deep knowledge about or expertise with each one of them. The cloud aims to cut costs, and helps the users focus on their core business instead of being impeded by IT obstacles. The main enabling technology for cloud computing is virtualization. Virtualization software separates a physical computing device into one or more “virtual” devices, each of which can be easily used and managed to perform computing tasks. With operating system–level virtualization essentially creating a scalable system of multiple independent computing devices, idle computing resources can be allocated and used more efficiently. Virtualization provides the agility required to speed up IT operations, and reduces cost by increasing infrastructure utilization.

.

**CHAPTER 2**

**LITERATURE REVIEW**

# In paper [1], we can infer that in today’s hectic world of work loaded life, there is no time for an individual to give a dedicated schedule for making his daily meals. You just want to relax after returning from work, either by ordering in a takeaway or heading out to a delicious establishment near you. At times it becomes a very tedious work for humans to think on, for what to make at dinner in such limited materials available. The food industry is an underserved market. The possibilities are wide-open, and the competition is scarce. Entrepreneurs can easily make a quick turn around by providing a much needed service. We are hence, developing an Android Application which would make the task of preparing meals a less tensed for an individual. The system is an innovative new recipe search engine which finds recipes you can make with the ingredients you currently have at hand.

In paper [2], In today’s world a person can prepare a dish either by asking another person who has acquired such knowledge of cooking or by reading certain cooking recipe books, and gaining self-knowledge by owning such books. Another way of making delicious recipes is by searching for the recipes over the internet, read the instructions and prepare it. In most of the cases every person visiting a website, already know a name, by which he searches the website for the dish. We have come up with this unique idea of preparing meals just by providing ingredients to the system. Just add your ingredients and Foodie instantly finds matching recipes to give the required meals you want in a jiffy. We're all about good recipes, and about quality home cooking that everyone can enjoy. If you have a little gourmand at home, let him try his hand at making dinner with these simple methods to prepare a meal. It is like having a TV cooking show 'cheftestant' with you in the kitchen who always knows what to make, no matter what random ingredients are in your 'mystery basket'. The study therefore aimed at developing an application to enable user’s search for whatever recipes they are interested in.

In paper [3], the inferences and features adopted for this application is that the recipes would be displayed to the user based on the ingredients provided in form of text, voice or image. With such a system in place, the application would be more competitive and adaptive to all types of users. The goal of this project is to create an Android software application that is accessible to everyone. The app will be easy to use with a variety of features available. These features will include a hierarchy of how different foods are related to one another and a recipe generator. You can just start by typing in the names of the ingredients that you already have in hand. Searching by the ingredients allows you to set your parameters based on what you have. Apart from searching the recipes based on ingredients, this application also gives step by step cooking procedures, nutritional details of the recipes, reviews of the users and also the video directions to prepare that particular food item.

In the paper [4] survey, we have observed following issues with the current available applications for Recipe providers: applications like Cookpad, CookBook, etc. are recipe provider apps which provide the recipes’ directions, ingredients, and reviews. It does not accept ingredients as input in form of voice and image. Most of the available means of recipe apps provide less Indian recipes, while our app is mainly dedicated for Indian users by providing more than ever Indian recipes. The available applications do not provide active user interface with the app. In the conducted survey we learned that, to make the app more user-friendly and efficient, the inputs to the system have to be provided in three ways - text, voice and image. Also it will show different types of dishes and their preparation time.

# 2.1 MERN stack

A stack is the combination of technologies used to create a web application. Any web application will be made using multiple technologies (frameworks, libraries, databases etc).The MERN stack is a JavaScript stack that’s designed to make the development process smoother. MERN includes four open-source components: MongoDB, Express, React, and Node.js. These components provide an end-to-end framework for developers to work in.

**2.1.1 MongoDB**

MongoDB is a cross-platform document database [MongoDB](https://www.mongodb.com/) is a NoSQL (non-relational) document-oriented database. MongoDB is known for being flexible and easy to scale. It is written in C++.

While conventional relational databases have a typical schema design based on columns and tables, MongoDB is schema-less. Data is stored in flexible documents with a JSON (JavaScript Object Notation)-based query language. The content, size, and number of fields in the documents can differ from one to the next. This means that the data structure to be changed over time.

Database is a physical container for collections. Each database gets its own set of files on the file system. A single MongoDB server typically has multiple databases.

Collection is a group of MongoDB documents. It is the equivalent of an RDBMS table. A collection exists within a single database. Collections do not enforce a schema. Documents within a collection can have different fields. Typically, all documents in a collection are of similar or related purpose.

# ExpressJS

Express is a minimal and flexible Node.js web application framework that provides a robust set of features for web and mobile applications. It is an open source framework developed and maintained by the Node.js foundation.

A back-end web application framework for Node.js, another MERN component. Instead of writing full web server code by hand on Node.js directly, developers use express to simplify the task of writing server code. There’s no need to repeat the same code over and over, as you would with the Node.js HTTP module.

The Express framework is designed for building robust web applications and APIs. It’s known for its fast speed and minimalist structure, with many features available as plugins.

* + 1. **ReactJS**

React was originally created by a software engineer at Facebook, and was later open-sourced. It is maintained by Facebook, as well as a community of development companies and individual developers.

The React library can be used for creating views rendered in HTML. React views are declarative. This means that developers don’t have to worry about managing the effects of changes in the view’s state (the object that determines how components behave) or changes in the data.Instead of relying on templates to automate the creation of repetitive HTML or DOM (Document Object Model) elements, React uses a full-featured programming language (JavaScript) to construct repetitive or conditional DOM elements.

React anchors the MERN stack. In a way, it’s the defining feature of the stack. It’s the one component that differentiates MERN from MEAN, another popular JavaScript stack that uses AngularJS (a front-end web application framework) instead of the React library.

**2.1.4 Node.js**

It is built on Chrome’s V8 JavaScript engine. It’s designed to build scalable network applications, and can execute JavaScript code outside of a browser.

Node.js works without an enclosing HTML page, instead using its own module system based on CommonJS, to put together multiple JavaScript files.

Here are the benefits of using Node.js

1. **Easy Scalability:** Developers prefer to use Node.js because it is easily scale the application in both horizontal and vertical direction. We can also add extra resources during the scalability of application.
2. **Real time web apps:** If you are building a web app you can also use PHP and it will take the same amount of time when you use Node.js, But if I am talking about building chat apps or gaming apps Node.js is much more preferable because of faster synchronization. Also, event loop avoid HTTP overload for Node.js development.
3. **Fast Suite:** NodeJs runs on the V8 engine developed by Google. Event loop in NodeJs handles all asynchronous operation so NodeJs acts like a fast suite and all the operations can be done quickly like reading or writing in the database, network connection or file system
4. **Easy to learn and code:** NodeJs is easy to learn and code because it uses JavaScript. If you are a front-end developer and have a good grasp on JavaScript you can easily learn and build the application on NodeJS
5. **Advantage of Caching:**It provides the caching of single module. Whenever there is any request for the first module, it gets cached in the application memory so you don’t need to re-execute the code.
6. **Data Streaming:** In NodeJs HTTP request and response are considered as two separate events. They are data stream so when you process a file at the time of loading it will reduce the overall time and will make it faster when the data is presented in the form of transmissions. It also allows you to stream audio and video files at lightning speed.

**2.5 References**

* International Journal for Research in Applied Science & Engineering Technology (IJRASET) ©IJRASET 2013
* FOODIE: An Android Application Rahul Dandekar 1 , Sanket Gadkari 2 , Mayuri Sadudia3 , Sayali Kamble4 1,2,3,4
* Under the Guidance of Prof. Shrikant Sanas, Department of Information Technology Padmabhushan Vasantdada Patil Pratishthan’s College of Engineering, Mumbai Colby K. M., 1975
* Artificial Paranoia: A computer program for the study of natural language communication between man and machine, Communications of the ACM, vol. 9, pp. 36-45.

**CHAPTER 3**

**ANALYSIS**

The Recipe App is a fullstack based application.It provides a user-friendly, interactive interface for the users to search for recipes. The End-users can have complete and detailed information regarding of the recipes by an interactive process.

**3.1 Purpose**

Food listing application displays the list of food recipes on the web page, as requested by the user. To display a list of recipes, we require an Application Program Interface (API) which will fetch all the data from a database which is hosted somewhere else. A programmer writing an application program can make a request to the Operating System using API (using graphical user interface or command interface). It is a set of routines, protocols and tools for building software and applications.

**3.2 Scope**

The API used in this application is the Food2Fork’s recipe API. Food2Fork offers an API which exposes its powerful recipe discovery functions for the application to use. The API gives access to their ever expanding recipe database, powerful ingredient search function, and social-media based ranking algorithm.

Any request will return a maximum of 30 results. Food2Fork uses a proprietary ranking algorithm based on of how often a recipe is shared across various social-media sites including Facebook, Pinterest, and Twitter.

**3.3 Motivation**

The main motivation behind the selection of this project was to design, develop and implement a software application which may be useful for people to find recipes sooner and with lesser effort. Further, it was the motivation to make the application user interface interactive and user-friendly and at the same time challenging to design and develop a responsive website.Responsive design can help you solve a lot of problems for your website. It will make your site mobile-friendly, improve the way it looks on devices with both large and small screens, and increase the amount of time that visitors spend on your site.

**3.4 Software Requirement Specifications**

A **Software Requirements Specification** (SRS) is a description of a software system to be developed. It lays out functional and non-functional requirements and may include a set of use cases that describe user interactions that the software must provide.

The proposed system has been designed to support the users to have seem-less, interactive experience. The application is open to all users. Each user can interact with the application.

**3.4.1 Performance Requirement**

The PCs used must be at least be INTEL CORE i3 machines so that they can give optimum performance of the product. In addition to these requirements, the system should also embrace the following requirements: -

* **Reliability:** The system should have little or no downtime.
* **Ease of Use:** The general and administrative views should be easy to use and intuitive.

**Design Constraints**

The designers must design the responses to every request that’s made in way that all the information and security constraints are met.

**3.4.2 Hardware Requirement**

* 64-bit processor
* Internet Connectivity 100 Mbps
* 8 GB memory

**3.4.3 Software Requirement**

1. Download [**npm**](https://www.npmjs.com/get-npm)
2. Install **React**
3. Download and install node.js
4. Install [React CLI](https://www.npmjs.com/package/react-cli)
5. Install **yarn**
6. Install **SASS**
7. Operating System: Windows 10

**CHAPTER 4**

**SYSTEM DESIGN**

**4.1 ReactJS:**

**React** (also known as **React.js** or **ReactJS**) is a [JavaScript library](https://en.wikipedia.org/wiki/JavaScript_library) for building [user interfaces](https://en.wikipedia.org/wiki/User_interfaces). It is maintained by [Facebook](https://en.wikipedia.org/wiki/Facebook) and a community of individual developers and companies.

React can be used as a base in the development of [single-page](https://en.wikipedia.org/wiki/Single-page_application) or mobile applications, as it is optimal for fetching rapidly changing data that needs to be recorded. However, fetching data is only the beginning of what happens on a web page, which is why complex React applications usually require the use of additional libraries for [state management](https://en.wikipedia.org/wiki/State_management), [routing](https://en.wikipedia.org/wiki/Web_framework#URL_mapping), and interaction with an [API](https://en.wikipedia.org/wiki/API):Next.jsand Gatsby.jsare examples of such libraries.

React code is made of entities called components. Components can be rendered to a particular element in the [DOM](https://en.wikipedia.org/wiki/Document_Object_Model) using the React DOM library. When rendering a component, one can pass in values that are known as "props". The two primary ways of declaring components in React is via functional components and class-based components.

ReactDOM.render(<Greeter greeting = “Hello World!”/>,

Document.getElementById(‘myReactApp’));

### Functional components

React Function Components, also known as React Functional Components are the status quo of writing modern React apps. In the past, there have been various Component Types. But with introduction of Hooks, it's possible to write application with just functions as React components.

**function**Greeting(props){

**return**<div>Hello,{props.name}!</div>;

}

### Class-based components

Class-based components are declared using [ES6](https://en.wikipedia.org/wiki/ECMAScript) classes. They are also known as "stateful" components, because their state can hold values throughout the component and can be passed to child components through props:

**class**ParentComponent**extends**React.Component{

state={color:'green'};

render(){

**return**(

<ChildComponentcolor={**this**.state.color}/>

);

}

}

### Lifecycle methods

Lifecycle methods are [hooks](https://en.wikipedia.org/wiki/Hooking) that allow execution of code at set points during a component's lifetime.

* shouldComponentUpdate allows the developer to prevent unnecessary re-rendering of a component by returning false if a render is not required.
* componentDidMount is called once the component has "mounted" (the component has been created in the user interface, often by associating it with a [DOM](https://en.wikipedia.org/wiki/Document_Object_Model) node). This is commonly used to trigger data loading from a remote source via an [API](https://en.wikipedia.org/wiki/API).
* componentWillUnmount is called immediately before the component is torn down or "unmounted". This is commonly used to clear resource demanding dependencies to the component that will not simply be removed with the unmounting of the component (e.g., removing any setInterval() instances that are related to the component, or an "[eventListener](https://en.wikipedia.org/wiki/Event_(computing))" set on the "document" because of the presence of the component)
* render is the most important lifecycle method and the only required one in any component. It is usually called every time the component's state is updated, which should be reflected in the user interface.

**4.2 NodeJS**

**Node.js** is an [open-source](https://en.wikipedia.org/wiki/Open-source_software), [cross-platform](https://en.wikipedia.org/wiki/Cross-platform), [JavaScript](https://en.wikipedia.org/wiki/JavaScript) [run-time environment](https://en.wikipedia.org/wiki/Runtime_system) that executes JavaScript code outside of a browser. Node.js lets developers use JavaScript to write command line tools and for [server-side scripting](https://en.wikipedia.org/wiki/Server-side_scripting)running scripts server-side to produce [dynamic web page](https://en.wikipedia.org/wiki/Dynamic_web_page) content before the page is sent to the user's web browser. Consequently, Node.js represents a "JavaScript everywhere" paradigm,[[6]](https://en.wikipedia.org/wiki/Node.js#cite_note-6) unifying [web application](https://en.wikipedia.org/wiki/Web_application) development around a single programming language, rather than different languages for server- and client-side scripts.

Though .js is the standard [filename extension](https://en.wikipedia.org/wiki/Filename_extension) for JavaScript code, the name "Node.js" does not refer to a particular file in this context and is merely the name of the product. Node.js has an [event-driven architecture](https://en.wikipedia.org/wiki/Event-driven_architecture) capable of [asynchronous I/O](https://en.wikipedia.org/wiki/Asynchronous_I/O). These design choices aim to optimize [throughput](https://en.wikipedia.org/wiki/Throughput) and [scalability](https://en.wikipedia.org/wiki/Scalability) in web applications with many input/output operations, as well as for [real-time Web](https://en.wikipedia.org/wiki/Real-time_Web) applications (e.g., [real-time communication](https://en.wikipedia.org/wiki/Real-time_communication) programs and [browser games](https://en.wikipedia.org/wiki/Browser_game)).

The Node.js [distributed development](https://en.wikipedia.org/wiki/Distributed_development) project, governed by the Node.js Foundation,[[8]](https://en.wikipedia.org/wiki/Node.js#cite_note-8) is facilitated by the [Linux Foundation](https://en.wikipedia.org/wiki/Linux_Foundation)'s Collaborative Projects program.

Corporate users of Node.js software include [GoDaddy](https://en.wikipedia.org/wiki/GoDaddy), [Groupon](https://en.wikipedia.org/wiki/Groupon), [IBM](https://en.wikipedia.org/wiki/IBM), [LinkedIn](https://en.wikipedia.org/wiki/LinkedIn), [Microsoft](https://en.wikipedia.org/wiki/Microsoft), [Netflix](https://en.wikipedia.org/wiki/Netflix), [PayPal](https://en.wikipedia.org/wiki/PayPal), [Rakuten](https://en.wikipedia.org/wiki/Rakuten), [SAP](https://en.wikipedia.org/wiki/SAP_SE), [Voxer](https://en.wikipedia.org/wiki/Voxer), [Walmart](https://en.wikipedia.org/wiki/Walmart), and [Yahoo!](https://en.wikipedia.org/wiki/Yahoo!)

Node.js allows the creation of [Web servers](https://en.wikipedia.org/wiki/Web_server) and networking tools using [JavaScript](https://en.wikipedia.org/wiki/JavaScript) and a collection of "modules" that handle various core functionalities.Modules are provided for [file system](https://en.wikipedia.org/wiki/File_system) I/O, networking ([DNS](https://en.wikipedia.org/wiki/Domain_Name_System), [HTTP](https://en.wikipedia.org/wiki/HTTP), [TCP](https://en.wikipedia.org/wiki/Transmission_Control_Protocol), [TLS/SSL](https://en.wikipedia.org/wiki/Transport_Layer_Security), or [UDP](https://en.wikipedia.org/wiki/User_Datagram_Protocol)), [binary](https://en.wikipedia.org/wiki/Binary_file) data (buffers), [cryptography](https://en.wikipedia.org/wiki/Cryptography) functions, [data streams](https://en.wikipedia.org/wiki/Stream_(computing)), and other core functions. Node.js's modules use an API designed to reduce the complexity of writing server applications

Though initially the module system was based on commonjs module pattern, the recent introduction of modules in the [ECMAScript specification](https://en.wikipedia.org/wiki/ECMAScript) has shifted the direction of using ECMAScript Modules in Node.js by default instead.

Node.js is officially supported on [Linux](https://en.wikipedia.org/wiki/Linux), [macOS](https://en.wikipedia.org/wiki/MacOS) and [Microsoft Windows](https://en.wikipedia.org/wiki/Microsoft_Windows) 7 and Server 2008 (and later), with tier 2 support for [SmartOS](https://en.wikipedia.org/wiki/SmartOS) and [IBM AIX](https://en.wikipedia.org/wiki/IBM_AIX) and experimental support for [FreeBSD](https://en.wikipedia.org/wiki/FreeBSD). [OpenBSD](https://en.wikipedia.org/wiki/OpenBSD) also works, and LTS versions available for [IBM i](https://en.wikipedia.org/wiki/IBM_i) (AS/400). The provided source code may also be built on similar operating systems to those officially supported or be modified by third parties to support others such as [NonStop OS](https://en.wikipedia.org/wiki/NonStop_OS)and [Unix](https://en.wikipedia.org/wiki/Unix) servers. Alternatively, it can be written with [CoffeeScript](https://en.wikipedia.org/wiki/CoffeeScript) (a JavaScript alternative), [Dart](https://en.wikipedia.org/wiki/Dart_(programming_language)) or [TypeScript](https://en.wikipedia.org/wiki/TypeScript) ([strongly typed](https://en.wikipedia.org/wiki/Strongly_typed) forms of JavaScript), or any other language that [can compile to JavaScript](https://en.wikipedia.org/wiki/Source-to-source_compiler).

Node.js is primarily used to build network programs such as Web servers. The most significant difference between Node.js and [PHP](https://en.wikipedia.org/wiki/PHP) is that most functions in PHP [block](https://en.wikipedia.org/wiki/Asynchronous_I/O) until completion (commands only execute after previous commands finish), while Node.js functions are [non-blocking](https://en.wikipedia.org/wiki/Asynchronous_I/O) (commands execute [concurrently](https://en.wikipedia.org/wiki/Concurrent_computing) or even in [parallel](https://en.wikipedia.org/wiki/Parallel_computing), and use [callbacks](https://en.wikipedia.org/wiki/Callback_(computer_programming)) to signal completion or failure).

### Platform architecture

Node.js brings [event-driven programming](https://en.wikipedia.org/wiki/Event-driven_programming) to [web servers](https://en.wikipedia.org/wiki/Web_server), enabling development of fast web servers in JavaScript. Developers can create scalable servers without using [threading](https://en.wikipedia.org/wiki/Thread_(computing)), by using a simplified model of [event-driven programming](https://en.wikipedia.org/wiki/Event-driven_programming) that uses callbacks to signal the completion of a task. Node.js connects the ease of a scripting language (JavaScript) with the power of Unix network programming.

Node.js was built on the [Google](https://en.wikipedia.org/wiki/Google) [V8 JavaScript engine](https://en.wikipedia.org/wiki/V8_JavaScript_engine) since it was open-sourced under the [BSD license](https://en.wikipedia.org/wiki/BSD_license). It is proficient with internet fundamentals such as [HTTP](https://en.wikipedia.org/wiki/HTTP), [DNS](https://en.wikipedia.org/wiki/DNS), [TCP](https://en.wikipedia.org/wiki/Transmission_Control_Protocol). [JavaScript](https://en.wikipedia.org/wiki/JavaScript) was also a well-known language, making Node.js accessible to the [web development community](https://en.wikipedia.org/wiki/Web_developer).

### Threading

Node.js operates on a [single thread](https://en.wikipedia.org/wiki/Single_threading) event loop, using [non-blocking I/O](https://en.wikipedia.org/wiki/Non-blocking_I/O) calls, allowing it to support tens of thousands of concurrent connections without incurring the cost of thread [context switching](https://en.wikipedia.org/wiki/Context_switch). The design of sharing a single thread among all the requests that use the [observer pattern](https://en.wikipedia.org/wiki/Observer_pattern) is intended for building highly concurrent applications, where any function performing I/O must use a [callback](https://en.wikipedia.org/wiki/Callback_(computer_programming)). To accommodate the single-threaded event loop, Node.js uses the [libuv](https://en.wikipedia.org/wiki/Libuv) library—which, in turn, uses a fixed-sized thread pool that handles some of the non-blocking asynchronous I/O operations.

A thread pool handles the execution of parallel tasks in Node.js. The main thread function call posts tasks to the shared task queue, which threads in the thread pool pull and execute. Inherently non-blocking system functions such as networking translate to kernel-side non-blocking sockets, while inherently blocking system functions such as file I/O run in a blocking way on their own threads. When a thread in the thread pool completes a task, it informs the main thread of this, which in turn, wakes up and executes the registered callback.

A downside of this single-threaded approach is that Node.js doesn't allow [vertical scaling](https://en.wikipedia.org/wiki/Vertical_scaling) by increasing the number of [CPU cores](https://en.wikipedia.org/wiki/CPU_core) of the machine it is running on without using an additional module, such as cluster, StrongLoop Process Manager or pm2. However, developers can increase the default number of threads in the libuv thread pool. The server [operating system (OS)](https://en.wikipedia.org/wiki/Operating_system) is likely to distribute these threads across multiple cores. Another problem is that long-lasting computations and other CPU-bound tasks freeze the entire event-loop until completion.

**4.3 Food2Fork API**

Food2Fork offers an API which exposes its powerful recipe discovery functions for your app to use.

The API gives you access to our ever expanding recipe database, powerful ingredient search function, and social-media based ranking algorithm.

##### **Submitting a Query**

All search requests should be made to the search API URL.

https://www.food2fork.com/api/search

All recipe requests should be made to the recipe details API URL.

https://www.food2fork.com/api/get

* **Parameters**

All parameters can be encoded as either HTTP GET or POST parameters.

###### **Search Parameters**

key: API Key

q: (optional) Search Query (Ingredients should be separated by commas). If this is omitted top rated recipes will be returned.

sort: (optional) How the results should be sorted. See Below for details.

page: (optional) Used to get additional results

Replace YOUR\_API\_KEY with your api key, and copy/paste the below url into your browser for an immediate example, searching recipes with "chicken breast" as an ingredient.

https://www.food2fork.com/api/search?key=YOUR\_API\_KEY&q=chicken%20breast&page=2

###### **Recipe Details Parameters**

key: API Key

rId: Id of desired recipe as returned by Search Query

Replace YOUR\_API\_KEY with your api key, and copy/paste the below url into your browser for an immediate example.

https://www.food2fork.com/api/get?key=YOUR\_API\_KEY&rId=35382

##### **Search Sorting**

The Food2Fork API offers two kinds of sorting for queries. The first is by rating. This rating is based off of social media scores to determine the best recipes.

sort=r  
The second is by trendingness. The most recent recipes from our publishers have a trend score based on how quickly they are gaining popularity.

sort=t

##### **Pages (Search Only)**

Any request will return a maximum of 30 results. To get the next set of results send the same request again but with page = 2

The default if omitted is page = 1

##### **Response Parameters**

The response is json encoded.

**CHAPTER 5**

**DETAILED DESIGN**

**5.1 Pre-Requisities**

1. Download [npm](https://www.npmjs.com/get-npm): We use Node Package Manager (NPM) to download and manage packages in our application.
2. Install **React** and the [React CLI](https://www.npmjs.com/package/react-cli)

*npm install -g react-cli react.*

We want to install React and the React Command Line Interface so we can use them to create React applications anywhere in our file system.

1. Install **yarn**

*npm install homebrew.*

*brew install yarn.*

Homebrew is another package manager that we need to install Yarn. We use Yarn to install package dependencies.

1. Install **sass** with

*npm install -g sass.*

We’re using Syntactically Awesome Stylesheeets as our CSS pre-processor to make our styles modular and to get features which CSS doesn’t provide.

**Step 1: Project Architecture**

1. Using the command-line, CD into your project folder. I cloned my repository to my desktop so my command looks like this:

*cd Desktop/react-recipes*

* Once inside the project directory run the following command.

*create-react-app*

This creates a React application in your already established directory. To create a new application without a project folder, run:create-react-app *name-of-app*

1. Once create-react-app has finished, install the dependencies with:yarn
2. Now, we’ll open the app in your IDE of choice (I enjoy [VS Code](https://code.visualstudio.com/)) and start the the application.Once you open your project in your IDE, this is what your src/ folder will look like.Run yarn start in the command line to see the default create-react-app landing page.
3. Next, delete the files we won’t be using:App.css, App.test.js, index.css, and logo.svg.
4. I like my components to be modularized, so let’s go ahead and add some folders to structure our app.
5. Create a components folder in src/ and move App.js inside.
6. Update index.js to point to the new path ofApp.js and remove old imports for files we just deleted.
7. Inside App.js remove old imports and the default code you get with create-react-app.

**Setting Up Sass**

Here is a quick and easy way to use Sass in your project without ejecting!

1. Create a styles/ folder in src/ and add two folders inside of that: scss/ and css/. Inside of the scss/ folder, add two more folders: base/ and components/ . Also add an index.scss file inside of the scss/ folder.
2. Update index.js to include a path to index.css (which we haven’t generated yet). It will live inside the src/styles/css folder.
3. Now, let’s add a few styles to see the transpilation of Sass (SCSS) to CSS.
   * Inside styles/base/, create \_variables.scss and \_elements.scss. The elements partial will contain any styling to base HTML elements such as <body>, <h1>, etc. and the variables partial will contain our variables for fonts, colors, etc.
   * [Partials](https://sass-lang.com/guide) in Sass are snippets of Sass code which you can include in other Sass files. We’re creating a partial for variables and base elements so we can keep each type of styling separate. We will import each partial into the index.scss file in order to use it within our HTML file. Partials are denoted by an underscore in the file name (although you can omit this during import).
   * Once including the custom font link in the index.html file, create two variables.
4. Inside styles/scss/base/\_elements.scss, set the <h1> background-color and font-family.
5. Inside styles/scss/index.scss, import the partials.
6. In package.json, create a new script:

*"sass"* : "sass --watch src/styles/scss:src/styles/css" .This will take any Sass (SCSS) file found in src/styles/scss and transpile it to CSS code in src/styles/css.

1. In the terminal, run yarn run sass.
2. Run yarn start to see your newly created App!

Let’s start with the Recipe component.

What kind of information do we need to keep track of in our state for each recipe?

* **ingredients**: string []
* **steps**: string []
* **title**: string
* **id**: string

Create this.state in the constructor and add two items to our recipes array. The id for each recipe ideally should go through a function which ensures its uniqueness, but for simplicity’s sake we’ll just set it explicitly.

In order to check the type of the data passed through to the Recipe component, we are going to use PropTypes. We want to ensure the data we receive is the type of data we expect.

npm install prop-types --save

* Import PropTypes to Recipe.js.
* At the bottom of the component, before the export, add a propTypes key with the corresponding properties we’re going to be passing in: **ingredients**, **steps**, **title**, and **id**. I have set the isRequired flag on these because we don’t want to show a recipe without any of them.
* Create two constants, ingredients and steps, before the return() statement in the render function. These will create new list items for each item in the

ingredients and steps array. You can do this in the JSX, but I like to extract my logic into variables.

* Add HTML elements for the title, the Ingredients and Steps headers, and display the list of ingredients and steps we just created above with const.

**CHAPTER 6**

**IMPLEMENTATION**

Implementation is the process of: defining how the system should be built, ensuring that it is operational and meets quality standards. It is a systematic and structured approach for effectively integrating a software-based service or component into the requirements of end users.

**6.1 Pseudocode**

**6.1.1 Importing libraries**

importReactfrom'react';

importReactDOMfrom'react-dom';

import'bootstrap/dist/css/bootstrap.min.css';

importRouterfrom'./components/Router';

import\*asserviceWorkerfrom'./serviceWorker';

**6.1.2 Recipe details**

import React from 'react';

import { Link } from 'react-router-dom'

function Recipe(props) {

    return (

        <divclassName="col-md-4"style={{ marginBottom: "2rem" }} >

        <div className="recipes\_\_box">

        <imgclassName="recipe\_\_boximg" src={props.data.image\_url}

alt={props.data.recipe\_i} />

          <div className="recipe\_\_text">

           <h5className="recipes\_\_title{props.data.title}</h5>

            <pclassName="recipes\_\_subtitle">

Publisher:

                        <span>{props.data.publisher}</span>

              </p>

                </div>

                <button className="recipe\_buttons">

                    <Linkto={{pathname:`/recipe/${props.data.recipe\_id}`state: props.datA}}>

                        View Recipe

                    </Link>

                </button>

            </div>

        </div>

    );

}

export default Recipe;

**6.1.3 Recipe Search**

function SearchBar(props) {

  return (

    <formstyle={{ marginBottom:"2rem" }}

        onSubmit={(e)=>{

          e.preventDefault();

          props.onSubmit(e.target.recipeName.value);}}>

    <input className="form\_\_input"type="textname="recipeName"/>

    <inputclassName="form\_\_button"  type="submit" />

      </form>

  );

}

**6.1.4 Router**

function Router() {

    return (

        <BrowserRouter>

            <Switch>

                <Route exact  path="/" component={App}   />

                  <Route     path="/recipe/:id"  component={RecipeDetail}   />

   </Switch>

        </BrowserRouter>

    );

}

**6.1.5 Driver**

class App extends React.Component{

  constructor(props){

     super(props);

     this.searchRecipe = this.searchRecipe.bind(this);

     this.state = {  recipes: [] };

         }

 searchRecipe(recipeName){

    fetch(`https://www.food2fork.com/api/search?key=${YOUR\_API\_KEY}&q=${recipeName}`)

    .then(response=>response.json())

    .then(data=>

      this.setState({

      recipes: data.recipes

    }));

  }

  render(){

    return (

      <div className="App">

        <header className="App-header">

          <h2 className="App-title">Recipe Search</h2>

        </header>

        <SearchBar onSubmit={this.searchRecipe}/>

        <Recipes recipes={this.state.recipes} />

      </div>

    );

  }

}

**6.2 Programming Coding Guidelines**

Following Code guidelines are important to programmers for a number of reasons:

* 80% of the lifetime cost of a piece of software goes to maintenance.
* Hardly any software is maintained for its whole life by the original author.
* Code conventions improve the readability of the software, allowing engineers to understand new code more quickly and thoroughly.
* If you ship your source code as a product, you need to make sure it is as well packaged and clean as any other product you create.

**CHAPTER 7**

**TESTING**

### Software Testing is theprocess used to help identify the correctness, completeness, security and quality of the developed computer software. Testing is the process of technical investigation and includes the process of executing a program with the intent of finding errors.

**7.1 Unit Testing**

1. Record valid input is checked to see if it is of characters:

* Valid set of characters.
* Characters uppercase or lowercase.

Table 7.1 Unit test case for Record valid input check

|  |  |
| --- | --- |
| Sl No. of test case | 1 |
| Name of test | Check test |
| Feature being tested | Valid input |
| Sample Input | apple |
| Expected output | Prompt of list of all images of apple recipes along with view recipe button at bottom corner of the image. |
| Actual output | List of all images of apple recipes along with view recipe button. |
| Remarks | Test succeeded |

### appleresult.JPG

Figure 7.1 Unit testing result of search

The above figure shows the unit testing result for input apple.

**7.2 Integration Testing**

1. The submit is checked to see whether it accepts a proper input and gives a proper result to the user. Here validation is made by checking the set of characters typed by the user.

Table 7.2 Integration test case for submit

|  |  |
| --- | --- |
| Sl No. of test case | 1 |
| Name of test: | Integration Test Check |
| Feature being tested | Submit button |
| Sample Input | Cheese |
| Expected output | On click it must validate the input and produce a page with all recipes with images. |
| Actual output | A page with recipes along with their images are displayed. |
| Remarks | Test succeeded |

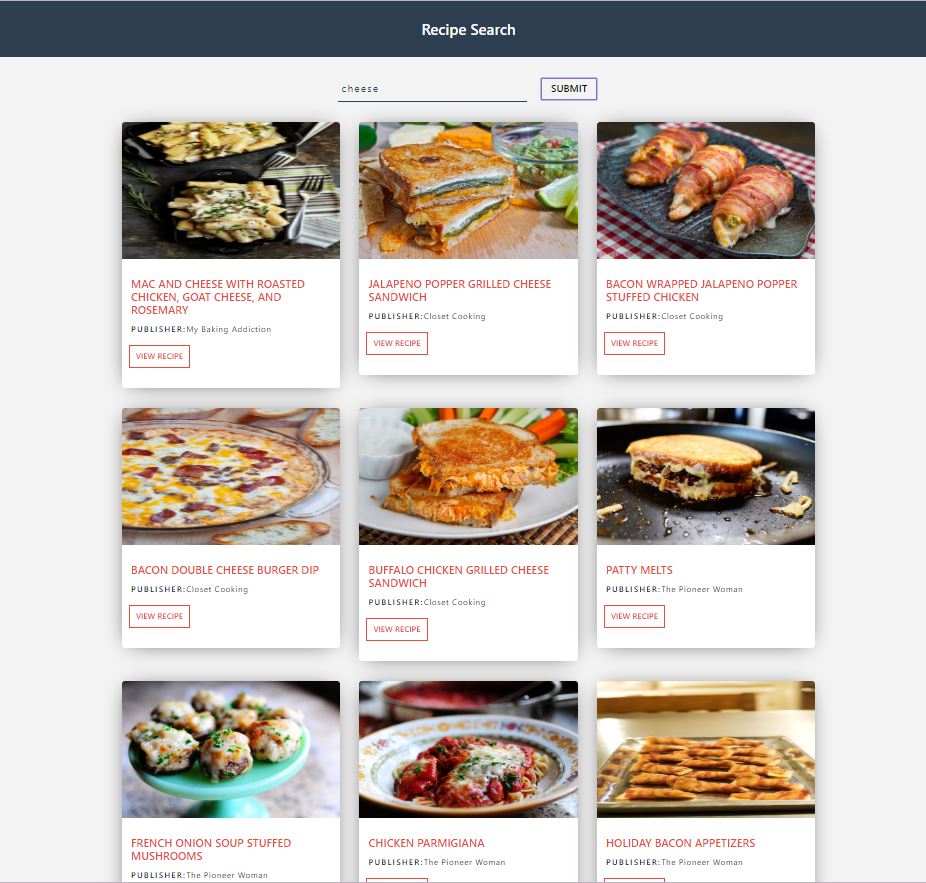
****

Figure 7.2 Integration test result for submit button

The above figure shows the integration testing when pressed on submit button.

2. The view button is checked to see whether on click displays the page with detailed recipe of food to prepare.

Table 7.3 Integration test case for view recipe

|  |  |
| --- | --- |
| Sl No. of test case | 2 |
| Name of test: | Integration Test Check1 |
| Feature being tested | VIEW RECIPE |
| Sample Input | pineapple |
| Expected output | On click it must display the page of detailed recipe of food when user clicks at bottom corner of image. |
| Actual output | A page with detailed recipe along with their images are displayed. |
| Remarks | Test succeeded |

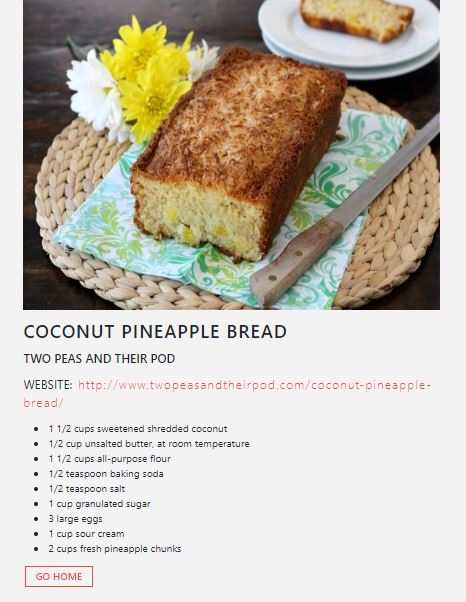
****

Figure 7.3 Integration test for view recipe

3.Integration testing for go home which displays the home page when clicked.

Table 7.4 Integration Testing for GO HOME

|  |  |
| --- | --- |
| Sl No. of test case | 3 |
| Name of test: | Integration Test Check2 |
| Feature being tested | GO HOME |
| Sample Input | mango |
| Expected output | On click it must display the home page |
| Actual output | Home page is displayed |
| Remarks | Test succeeded |

The below figure shows the integration test results for go home

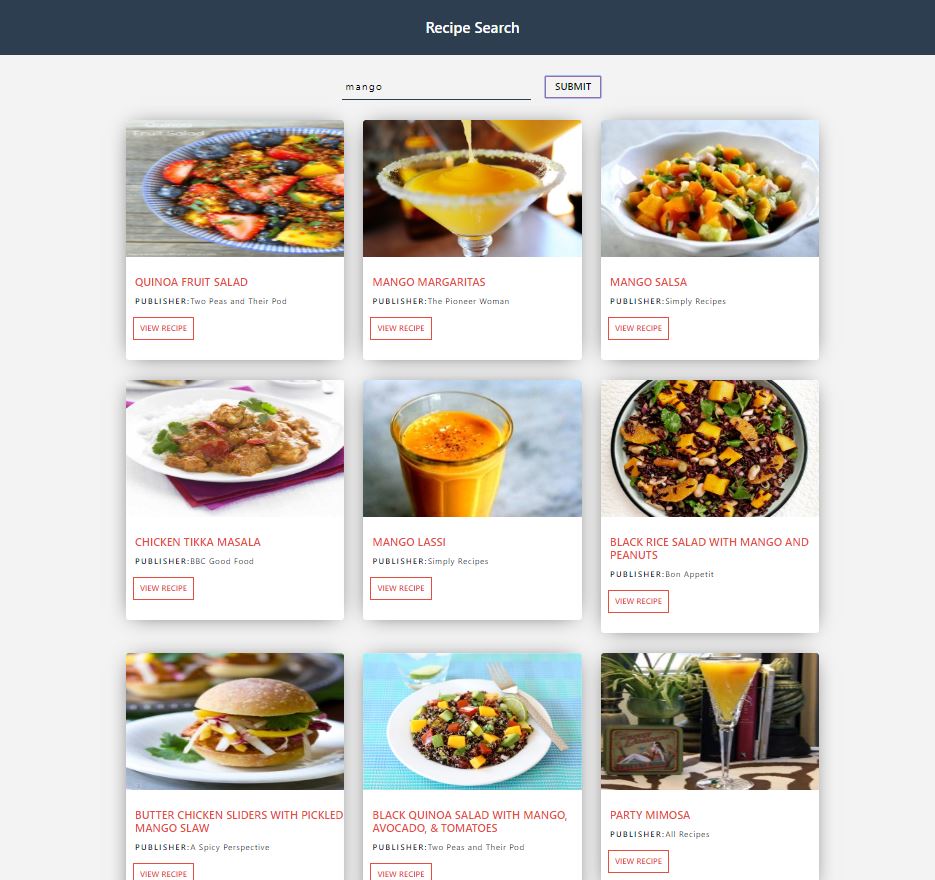


Figure 7.4 Integration test for go home

.

**7.3 System Testing**

System testing is a level of the software testing where a complete and integrated software is tested. The purpose of this test is to evaluate the system’s compliance with the specified requirements. The program is run to check if all the modules can be executed concurrently, if each return correct results of the operations performed by them.

Table 7.5 System test for Recipe Lists

|  |  |
| --- | --- |
| Sl No. of test case | 1 |
| Name of test: | System Test Check |
| Feature being tested | Recipe Lists |
| Sample Input | Choice of selective buttons provided like submit, view recipe, go home. |
| Expected output | Pages are to be displayed based on the click performed. |
| Actual output | Pages are displayed for all responsive buttons. |
| Remarks | Test succeeded |

The below figure shows the system testing results for recipe lists.

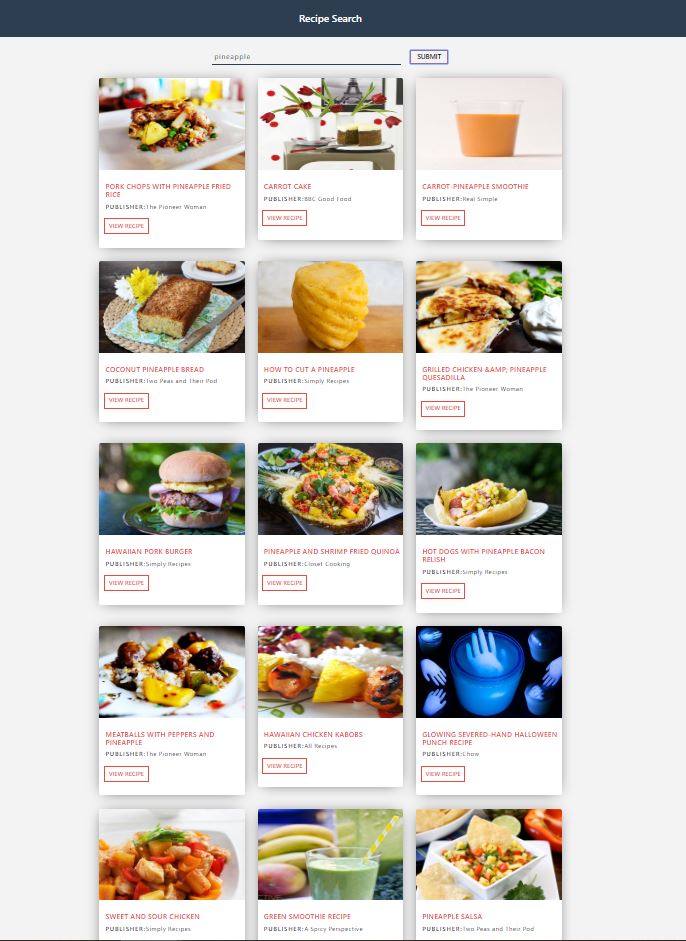


Figure 7.5 System test for Recipe Lists

**7.4 Acceptance Testing**

Acceptance testing, a testing technique performed to determined whether or not the software system has met the requirement specifications. The main purpose of this test is to evaluate the system’s compliance with the business requirements and verify if it has met the required criteria for delivery to end users.

There are three important forms of acceptance testing they are, Alpha testing, Beta testing and User acceptance testing.

**Alpha testing:** is a type of acceptance performed to identify all possible issues/bugs before releasing the product to everyday users or the public.

**Beta testing:** it is released to a limited number of end users of the product to obtain feedback on the product quality.

The below figure shows the acceptance testing made by the user.

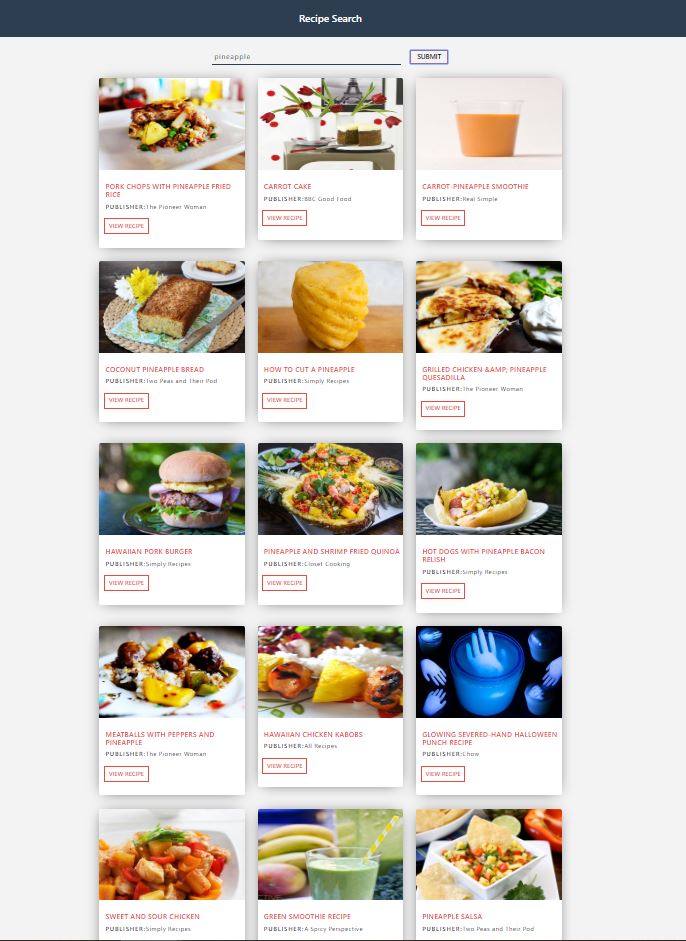


Figure 7.6 Acceptance Testing done by the users

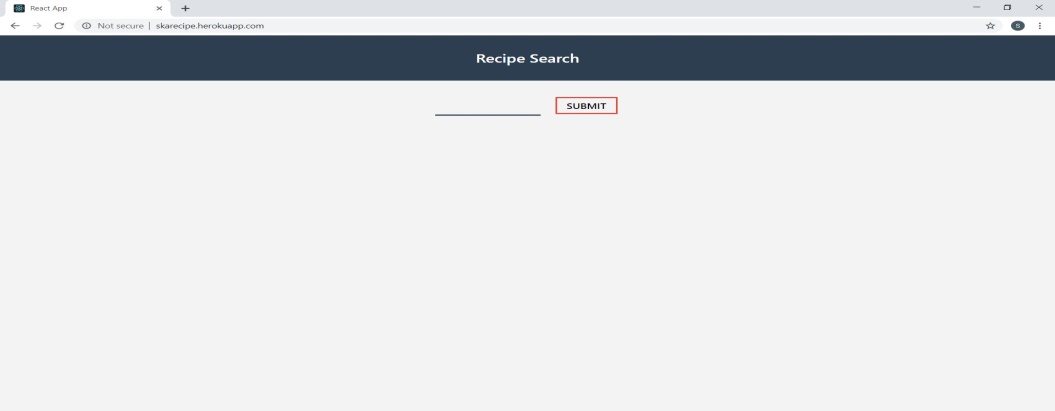
**CHAPTER 8**

**DISCUSSION OF RESULTS**

The outcomes of test results for a variety of input images and user interactions with the application are discussed in the following sections of the chapter

### Menu

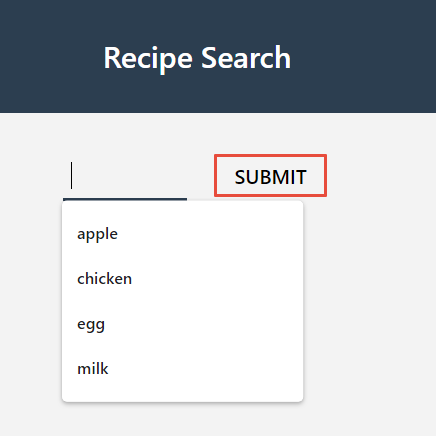
The menu page shown below is the first page that appears when searched for web application which gives user to type the food item for which recipe is to be searched.



**Figure 8.1Menu Page**

* 1. **Suggestion for Search**

The suggestions page shown below are shown on menu page while doing a search for recipe for a particular ingredient.



**Figure 8.2 Suggestions**

### Search

### The above figure shows a page where search for recipe is done.

### 

### Figure 8.3 Search Page

### Result of search

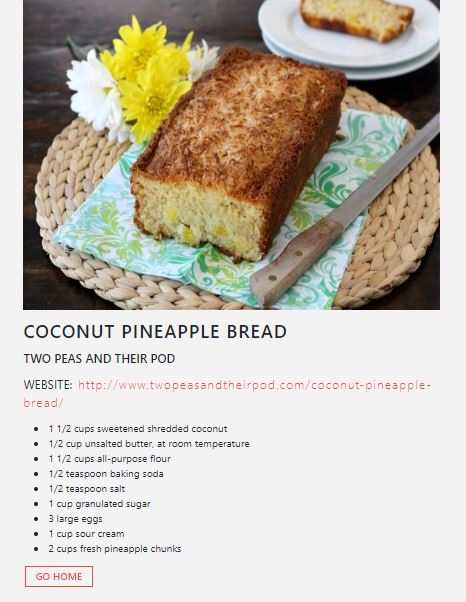
The page shown below shows the result for the particular item for which recipe is searched.

### 

Figure 8.2 Search results for item

### Particular Page for Search

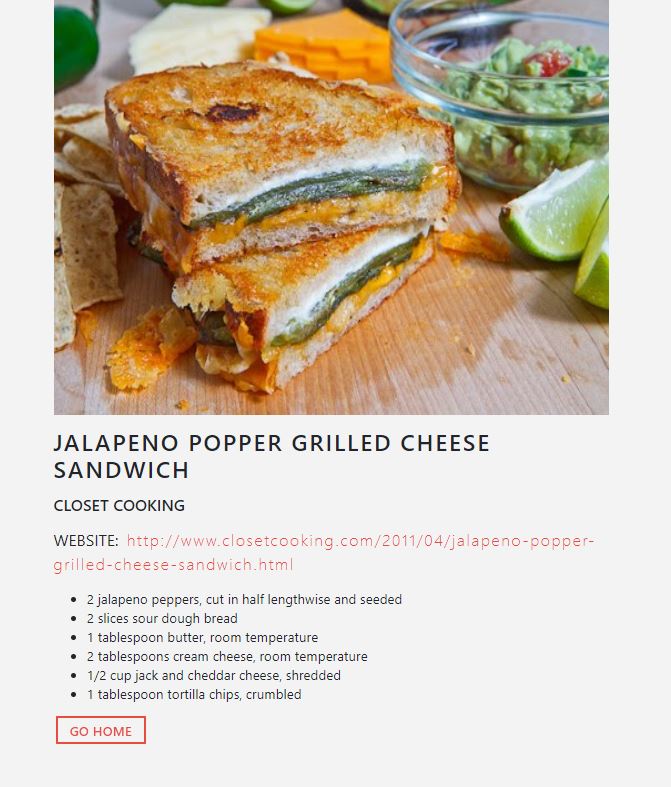
The below figure shows the result for a particular page of recipe and ingredients for a particular item.



**Figure 8.4 Particular Page**

* 1. **Go Home Page**

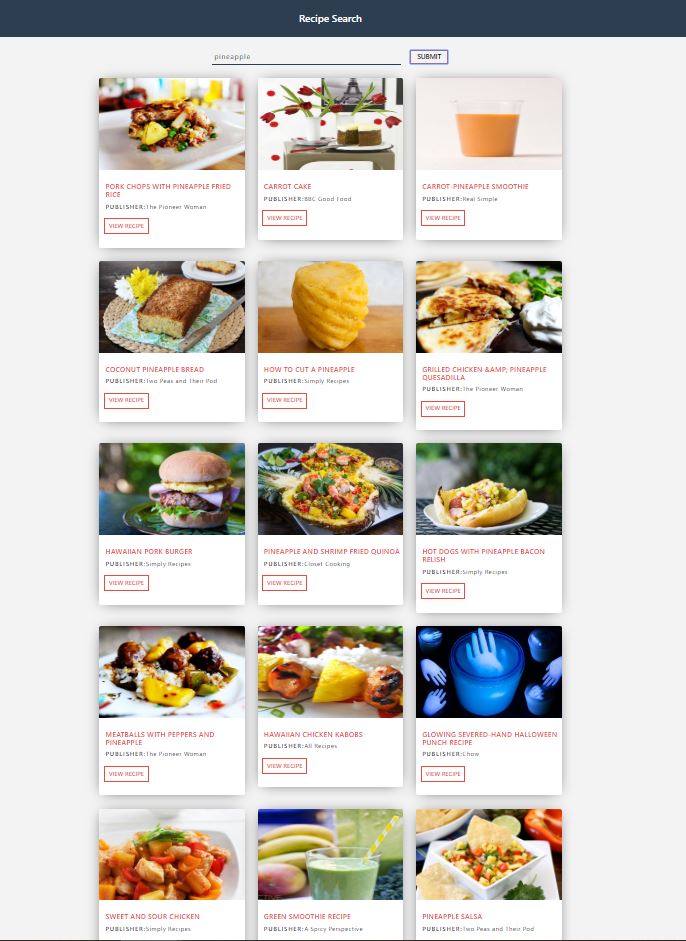
The figure shown below has a button GO HOME at the end of each recipe page which redirects to home page i.e. Menu Page.

****

**Figure 8.5 Go Home Page**

**8.7 Webpage**

The below figure is webpage shown to the user when search is made.

****

**Figure 8.7 Webpage**

**CHAPTER 9**

# CONCLUSION AND FUTURE ENHANCEMENTS

The recipe web application allows users to search for an item that can be used in a food recipe. This web application redirects the user to the page where the results of food recipes are shown. Based on requirements of food to be prepared the user can view the recipe page by clicking on VIEW RECIPE button present at the bottom corner of each image.

This web application is helpful to the user where it gives the user some of the delicious recipes for item searched. It is improvement of technology which replaces olden days recipe books and very easy to use.

* This web application can be further enhanced by allowing users to update their recipes in the web application so that other users can see that.
* User can be given an option to rate the recipe so it will be helpful for other users to select the recipe.

**REFERENCES**

1. https://www.freecodecamp.org
2. https://www.w3schools.com
3. https://www.stackoverflow.com
4. https://www.reactjs.org
5. https://www.tutorialspoint.com
6. Fundamentals of Web Development by Randy Connolly, Ricardo Hoar
7. The Little MongoDB Book by Karl Seguin
8. https://www.javatpoint.com