Rajiv Gandhi University of Knowledge Technologies(RGUKT),

R.K.Valley, Kadapa, Andhra Pradesh

Project Report

Online Recipe Book

September 2022

Submitted by

M.Vamsi ,R170717

Guided by: B.Lingamurthy

Software Engineer ,

IIIT RkValley

Department of Computer Science and Engineering

CERTIFICATE

This is to certify that the project work titled "**Online Recipe Book**" is a bonafied project work submitted by Mr. **Madugundu Vamsi** ID: **R170717** in the department of **COMPUTER SCIENCE AND ENGINEERING** in partial fulfillment of requirements for the award of degree of Bachelor of Technology in Computer Science and Engineering for the year **2021-2022** carried out the work under the supervision

GUIDE

B.Lingamurthy
Software Engineer,
IIIT Rk Valley

ACKNOWLEDGMENT

At the very beginning, I would like to express my gratitude for being guided and supervised by B.Lingamurthy sir. In fact, his continuous supervision and tracking allows me to work on a regular basis and so I could so far learn a lot about the fascinating world of Web Application Development and Software Engineering.

Abstract

There are a number of websites in online for Recipes Search but none of them support interface for creating, searching, saving, and sharing recipes all at once. Nala Bheema Paakam is an Web Application with image based UI for searching, sharing, creating and saving recipes. This app provides flexibility to user to search variety of recipes from available recipes in the forum. This is very handy application, which every user can search for recipes, save recipe as favorite, share recipe with friends on social media Facebook. This app is time saver providing recipes in few clicks. Through title search, NBP app makes finding recipes easy. With recipes being added daily there will always be something new for user to crave. The project has been implemented using React, Bootstrap, Node ,Express and Mongodb.

Table of Contents

I. Introduction:

Now a days, it is safe to admit that the world has attained the form of global village, where everything is accessible through technology. The advent of internet has shaped the life of many people. It is hard to pass a day by without checking and rechecking your social network accounts. This can merely assert that web applications has already made their ways to our lives. According to Butler (2011),

Whether people love to cook or just love to eat, they have a collection of dishes and recipes they'd like to try. May be they have a bunch handed down from a loved one. In either case, they certainly need a better method to keep them organized for the long haul than a bunch of index cards in a file folder, which old and tedious. Therefore cooking with your phone is a lot tastier when you have the right recipes.

NBP application is a very useful app for people who love to cook and try out new recipes. It provides user flexibility to search, share, save recipes from a database with an additional capability to maintain personal cookbook for creating new recipe, deleting recipe that are no longer required.

This application is a time saver providing recipes in few clicks. The user is given choice to create cookbook, where user can create recipe, view recipe and delete recipe. The interface is clean and simple. The user can search recipes, view added favorite recipe list and access personal cookbook all from home page.

II. Feasibility Study:

The feasibility study, as its name indicates, aims at assessing the practicality of the proposed capstone project. As stated in the Initial Specifications Report, The objective of this capstone is to develop a "Nala Bheema paakam" Web application to be used to elevate user's kitchen skills instead of using index cards in a file folder.

The app development will consist of five parts. The first part will be devoted to data gathering and software requirements specification. Consequently, I will have a look at different web apps which target the same goal. They are plenty of Recipe Organizer apps. Each one has some various features. The second part will be dedicated to the design phase, including the app and the database. Also, in this phase, the software tools to be used will be specified. For example, the IDE, the database Server, the modeling language for the design, and finally the software testing tools. The third part will be the implementation phase, here, the design will be converted to code in order to develop the targeted web app. The fourth step will be devoted to testing the app. In this phase, two testing methods will be used, namely: Black Box testing and White box testing. The last phase will be the deployment phase.

III. User and system requirement document:

1. *1. Project Description:*

The goal of this step is to ensure that the requirements are consistent, precise and complete to ensure that we meet the final outcome expectations. There are two types of requirements: functional and non-functional requirements. The functional requirements are the ones that describes the functions of the app; whereas, the non-functional requirements are the ones that present the app constraints and properties.

A. External Interface Requirements:

✓ **Hardware:**

The Web application will be operating on various browsers.

✓ **Software:**

The Web application will be compatible with the various web browsers.

✓ **Communication:**

The app will be dynamic. It will interact with the users instantly according to the inputs that it receives.

✓ **Users:**

Users will need to login with Google recorded to access to more elaborated functionalities as Add/Delete/Update a recipe. Also, if they want to print the recipe they can . These information will be kept in our database.

B. Functional Requirements:

- **Login:** A user needs to login in order to access the full app. The user needs to login with his/her Google account. Those information will be stored in the database and will serve as the profile info.

- **View Profile:** After being successfully login. The user can view his/her profile, where information of username and profile picture are displayed. Also, View Profile will enable you to view your created recipes. In addition, View Profile will make you check update,create and delete your recipes.

- **View Recipe:** The registered user shall be able to see the recipes posted by other users. In other words, the user shall be able to access the Recipe Forum, where all the user's recipes are posted and ordered by date of creation.

- **Add Recipe:** Any registered user shall able to create his/her own recipe. The creation of recipe include entering the title, description, and other information that concerns the recipe, such as the steps and the ingredients.

- **Search for a Recipe:**Any user shall able to search for a recipe using the title as a criterion. The search functionality will provide the users with a shortcut to find their targeted recipes if posted before by the other users.

- **View Ingredients:**The user of the app shall be able to view ingredients of a certain recipe. This option will allow him/her to mark those ingredients as needed, and ultimately find them in the shopping list.

- **View My Recipes:** The registered user shall be able to see all the recipes he/she already created.

- **Delete Recipe:** The registered user shall be able to delete his/her recipes, he/she already posted.
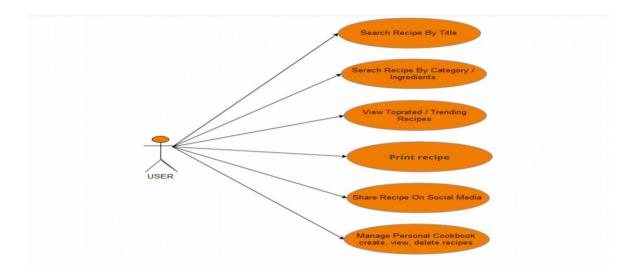
C. <u>Use Case Diagram:</u>

D. <u>Non Functional Requirements:</u>

   ✓ **Product Requirements**

  ▪ **Usability Requirements:**

> -The application shall be easy to use and intuitive.
>
> - The application shall have a user-friendly interface.
> - UI & UX shall be simple and clear.

  ▪ **Performance Requirements:**

- The application shall be fast and robust when loading.

- The program shall not allow more than 10 min/year of failure.

  ▪ **Space Requirements:**

- The application shall have enough memory space in order to store high number of data.

  ▪ **Reliability Requirements:**

- The application shall not produce an incorrect output.

  ▪ **Portability Requirements:**

- The software shall work in all different platforms.

   ✓ **Organizational Requirements:**

  ▪ **Standards Requirements:**

- The application shall conform to ISO standards.

   ✓ **External Requirements:**

  ▪ **Inter-operability Requirements:**

- The application shall allow access to the different department of the application without altering its efficiency and consistency.

  ▪ **Ethical Requirements:**

- *The application shall be license free.*

- **Privacy Requirements:**
- Personal information of the registered user shall only be accessed by the administrator.

- The guest user shall not be able to order a medicine.

- **Safety Requirements:**
- *The application shall be protected from any external danger or attacks.*

Technology Enablers:

The choice of technology enablers that will be used for the development of the

application is essential for its success. The technology enablers should provide a suitable

way to fulfill the requirements stated before. Principles of enterprise class applications

should be kept in mind during the choice of these technologies. The two main ones are

that there is no best technology but instead suitable ones and that we shouldn't reinvent

the wheel which means that we should take advantage on what was already implemented

and offered to the community.

A. Server Side:

The Web application is no longer static, it is dynamic. As the information content

grows, so does the need to make web applications more dynamic . Node js ,Express js and

Mongodb are two ways to make your application dynamic.

- **Node**: is a robust, server-side, open source scripting language that is extremely flexible.
- **MongoDB**: is a source-available cross-platform document-oriented database program. Classified as a NoSQL. MongoDB uses JSON-like documents with optional schema.
- **Express:** is a minimal and flexible Node.js web application framework that provides a robust set of features for web and mobile applications.

B. Client Side:
- **Visual Studio Code**

  A standalone source code editor that runs on Windows, macOS, and Linux. The top pick for JavaScript and web developers, with tons of extensions to support just about any programming language.

- **Live Server** :A Quick Development Live Server with live browser reload. Start or Stop server by a single click from status bar. Open a HTML file to browser from Explorer menu.[Quick Gif Demo]. Support for excluding files for change detection. Hot Key control. Customizable Port Number, Server Root, default browser.

- **Chrome Developer Tools:** The chrome developer tool has made debugging a lot easier. The built-in developer tool allows you to edit the page, debug it and also give them access to work on the internal web browser and application. This one boosts the productivity of the developers, and also it makes the easy diagnosis of the website.

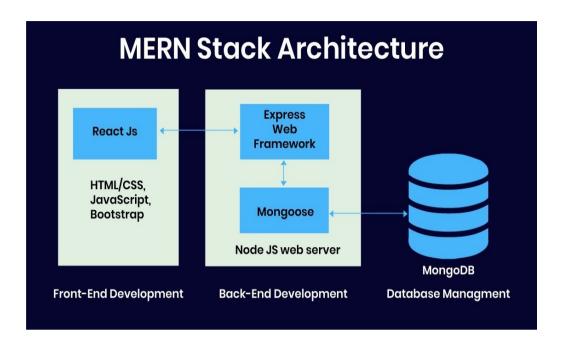## IV. System design and architecture

1.System architecture

Figure:SystemArchitecture

MERN is the acronym for **M**ongoDB, **E**xpress JS, **R**eact JS and **N**ode JS. The MERN stack is a combination of the above technologies, all based on JavaScript, used to build advanced web applications. It is an open source full stack development framework i.e. it provides entire front-end to back-end development components. While MongoDB, Express JS and Node JS are the common components (between MEAN and MERN); Angular JS is replaced by React JS in MERN.

With MongoDB under its hood for higher scalability, Express JS for speed enhancements, JavaScript as its primary language for end-to-end development, MERN is one of the best full stack development suites after MEAN. React JS is the best when it comes to UI layer abstraction. It provides the best-in-class tools for faster code development. While React is only a library, it gives you the freedom to build the application and organize the code the way you want, by providing the necessary tools. Hence, it is better than Angular in terms of UI rendering and performance.

14

```js
JS index.js U ✕
server > JS index.js > 🔶 app.post("/recipes") callback > 🔶 newRecipe.save() callback
 32      });
 33    });
 34
 35    app.post("/recipes", function (req, res) {
 36      const newRecipe = new Recipe({
 37        id: req.body.id,
 38        name: req.body.recipeName,
 39        category: req.body.category,
 40        author: req.body.authorName,
 41        author_id: req.body.author_id,
 42        image: req.body.image,
 43        cookingTime: req.body.cookingTime,
 44        servings: req.body.noOfServings,
 45        about: req.body.about,
 46        ingredients: req.body.ingredients,
 47        process: req.body.procedure,
 48      });
 49      newRecipe.save(function (err) {
 50        if (err) {
 51          console.log(err);
 52        } else {
 53          res.send("Added Successfullty");
 54        }
 55      });
 56
 57      console.log(newRecipe);
 58    });
```

Figure: JSON Architecture

Also, I have integrated my app with Google through the Google OAuth. This integration enables Google login, which authenticate people with their Google credentials, 18and Share and Send Dialogs, which enable sharing content from your app to google. Therefore, it is safe to admit that the server side exposes a RESTFUL API that will be consumed by any REST Client. Any application that uses OAuth 2.0 to access Google APIs must have authorization credentials that identify the application to Google's OAuth 2.0 server. The following steps explain how to create credentials for your project. Your applications can then use the credentials to access APIs that you have enabled for that project. The easiest way to add a Google Sign-In button to your site is to use an automatically rendered sign-in button. With only a few lines of code, you can add a button that automatically configures itself to have the appropriate text, logo, and colors for the sign-in state of the user and the scopes you request.

15

The figure below show the integration of my app with Google's OAuth 2.0.

```
JS Login.js U ×
client > src > components > JS Login.js > ⊙ Login > ⊙ useEffect() callback
   7    function Login({ user, setUser }) {
   8      function handleCallbackResponse(respose) {
   9        let userObject = jwt_decode(respose.credential);
  10        window.localStorage.setItem("Luser", JSON.stringify(userObject));
  11        lUser = JSON.parse(window.localStorage.getItem("Luser"));
  12        setUser(lUser);
  13        console.log("lUser:", lUser);
  14        window.localStorage.setItem("isLoggedIn", true);
  15        document.getElementById("SignInDiv").hidden = true;
  16      }
  17
  18      useEffect(() => {
  19        const google = window.google;
  20        google.accounts.id.initialize({
  21          client_id:
  22            "17925617582-ge35q2hbkmbauk9o9qcqb12cjpd4c4vt.apps.googleusercontent.com",
  23          callback: handleCallbackResponse,
  24        });
  25        google.accounts.id.renderButton(document.getElementById("SignInDiv"), {
  26          size: "large",
  27        });
  28      }, []);
  29      return (
  30        <div id="login">
  31          <h4>Please Login to Add recipe </h4>
  32          <div id="SignInDiv"></div>
  33          {/* {console.log("user.email_verified", lUser.email_verified)}; */}
  34          {window.localStorage.getItem("isLoggedIn") && (
  35            <Navigate to="/main/user" />
  36          )}
  37        </div>
```

Figure:Screen Shot of Google Oauth
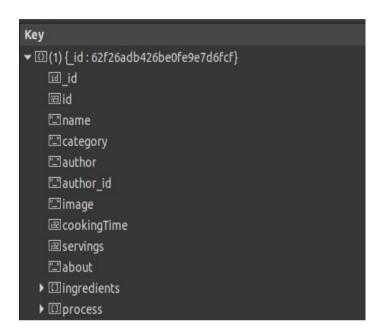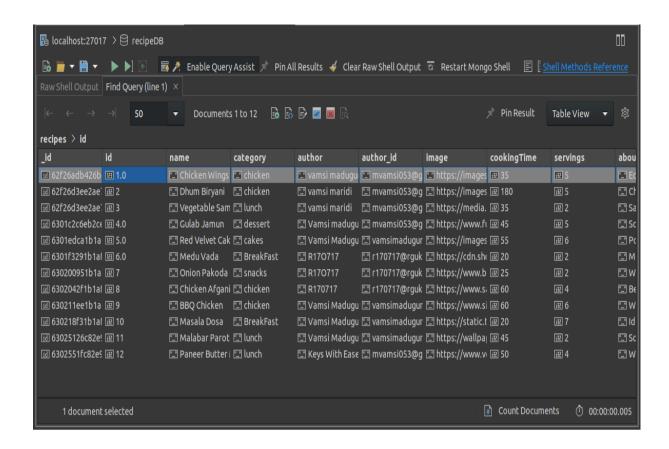
2. Database Design:

Figure: Nala Bheema Paakam Database Design

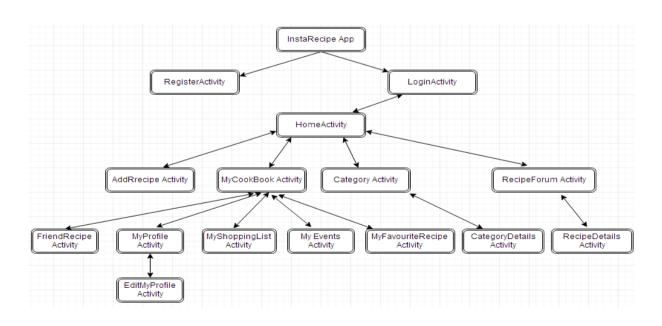*Figure:nala Bheema Paakam data base tables*

Figure:Activity Diagram

3. *Class Diagram:*

Instruction

- cookTimeString
- temperatureString
- narrativeString

+ getCookTime()String
+ getTemperature()
String
+ getNarrative()String

Quantity

- number: double
- unit: String

+ toString() String
+ scale() Quantity

Recipe

- title: String
- ingredients:
ArrayList<Ingredient>
- instruction:Instruction

+ addIngredient(Ingredient)
+ scale(double)Recipe
+ toString() String
+ equals(String)boolean
+ contains(String)boolean
+ display()String
+ getIngredients()ArrayList<Ingredient>

Ingredient

- nameString
- quantity:Quantity

+ getName()String
+ contains()boolean
+ display()String
+ toString() String
+ scale(double)Ingredient

RecipeList

- recipes:
ArrayList<Recipe>
+ add(Recipe)
+ remove(Recipe)
+ filterRecipeString)RecipeList
+ filterIngredient(Ingredient)RecipeList
+ sort()
+ display()String
+ toList(): String[]
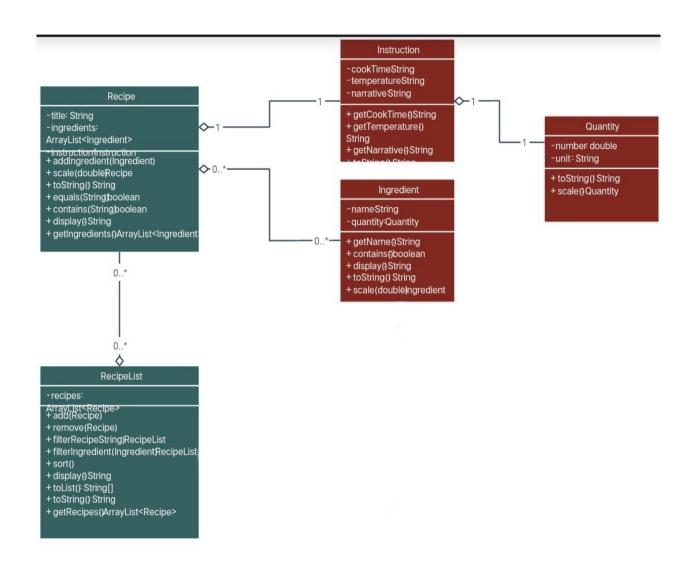+ toString() String
+ getRecipes()ArrayList<Recipe>

Figure:Class Diagram

V. Implementation Phase:

The implementation took roughly three months, and involved coding every single day to finally arrive at the final product which will be presented during the capstone defense. The implementation was about designing activities, which are similar to pages in web applications, creating the database inside my project and linking the graphical user interface with the functionalities and the database. Every time I implemented a functionality, I tested it directly on the web Browser for responsiveness, so as to get a real life representation of what the app would look like on our clients' browsers. I also managed to make the web app compatible with phones supporting different versions. Further, the web application has many features:

- **User Session Management:** Session are useful when you want to store user data globally throughout the application. This can be done in two ways: one is storing them in a global variables and second is storing the data in shared preferences(Local Storage). The problem with storing data in global variable is the fact that the data will be lost once the user closes the application, but storing data in shared preferences will be persistent even though user closes the application. Therefore, I have used Local Storage.

```
7   function Login({ user, setUser }) {
8     function handleCallbackResponse(respose) {
9       let userObject = jwt_decode(respose.credential);
10      window.localStorage.setItem("Luser", JSON.stringify(userObject));
11      lUser = JSON.parse(window.localStorage.getItem("Luser"));
12      setUser(lUser);
13      console.log("lUser:", lUser);
14      window.localStorage.setItem("isLoggedIn", true);
15      document.getElementById("SignInDiv").hidden = true;
16    }
17
```

```
30      <div id="login">
31        <h4>Please Login to Add recipe </h4>
32        <div id="SignInDiv"></div>
33        {/* {console.log("user.email_verified", lUser.email_verified)}; */}
34        {window.localStorage.getItem("isLoggedIn") && (
35          <Navigate to="/main/user" />
36        )}
37      </div>
38    );
39  }
```

Figure:Session Management using Local Storage

- **Single Sign-on:Google o Auth**   The easiest way to add a Google Sign-In button to your site is to use an automatically rendered sign-in button. With only a few lines of code, you can add a button that automatically configures itself to have the appropriate text, logo, and colors for the sign-in state of the user and the scopes you request.
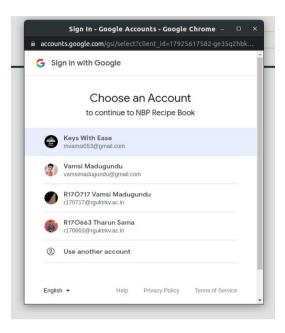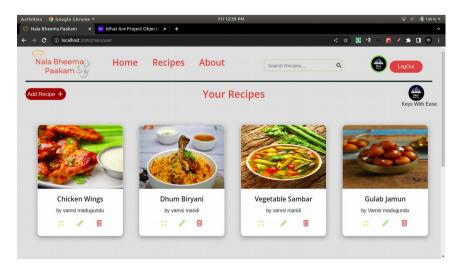

Figure:Google Integration

- **Friendly and flexible UI:** My app has a friendly and flexible User Interface. I have used Picasso as an image loader. Also, I made sure that every single layout is scrollable. I have used "Canva" as a software to design the UI , layouts and backgrounds.

VI. Testing:

I have used two types of testing:

- **Black-Box testing**: is a functional testing that targets only the interface to ensure that they work as expected. I make sure that warnings are shown if an input is missing using alerts.

- **White-Box testing:** is a clear-box one where the tester has access to the code. Since I have been the developer of my app, I have been practicing this type of testing on a daily basis.
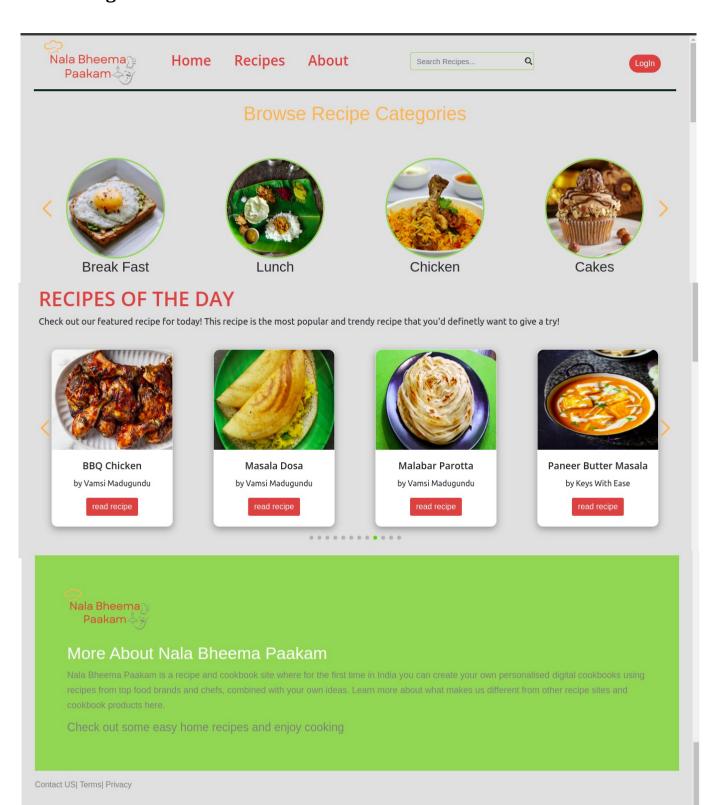
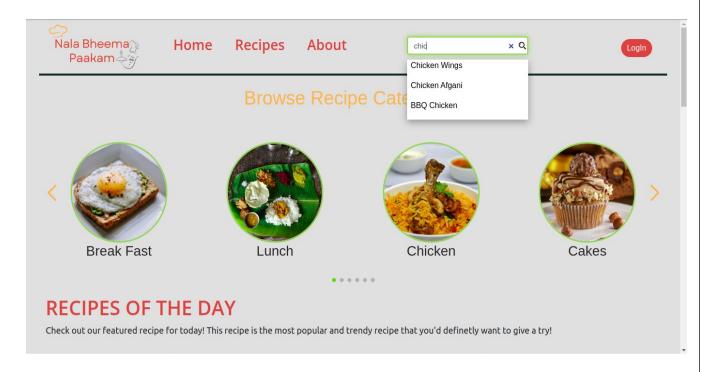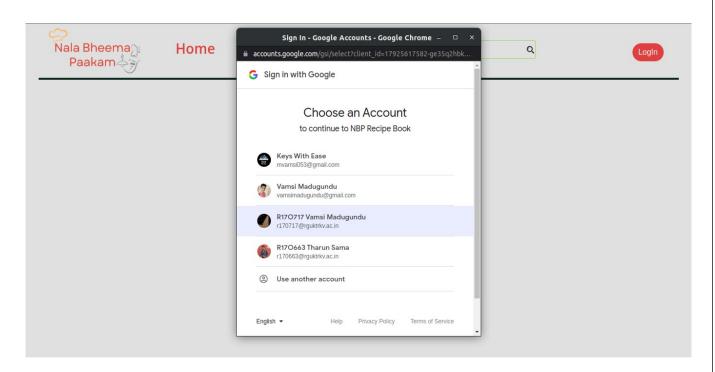## Evaluation:

Project URL: http://localhost:3000/
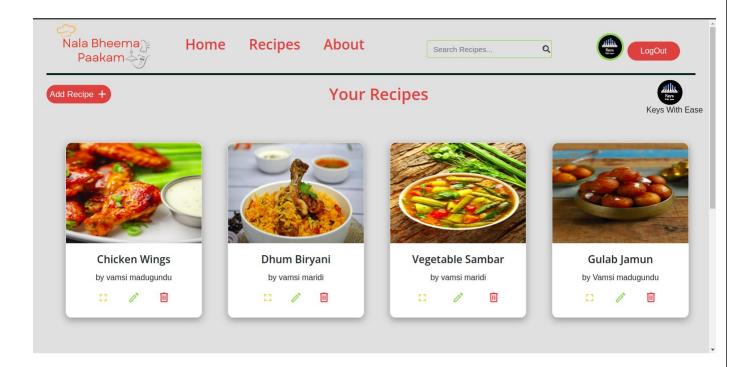
Home Page:

Main Page:

Nala Bheema Paakam

Home   **Recipes**   **About**

Search Recipes...   🔍

LogIn

Browse Recipe Categories

‹

Break Fast

Lunch

Chicken

Cakes

›

RECIPES OF THE DAY

Check out our featured recipe for today! This recipe is the most popular and trendy recipe that you'd definetly want to give a try!

‹

BBQ Chicken
by Vamsi Madugundu
read recipe

Masala Dosa
by Vamsi Madugundu
read recipe

Malabar Parotta
by Vamsi Madugundu
read recipe

Paneer Butter Masala
by Keys With Ease
read recipe

›

Nala Bheema Paakam

More About Nala Bheema Paakam

Nala Bheema Paakam is a recipe and cookbook site where for the first time in India you can create your own personalised digital cookbooks using recipes from top food brands and chefs, combined with your own ideas. Learn more about what makes us different from other recipe sites and cookbook products here.

Check out some easy home recipes and enjoy cooking

Contact US| Terms| Privacy

Copyright Nala Bheema Paakam 2022

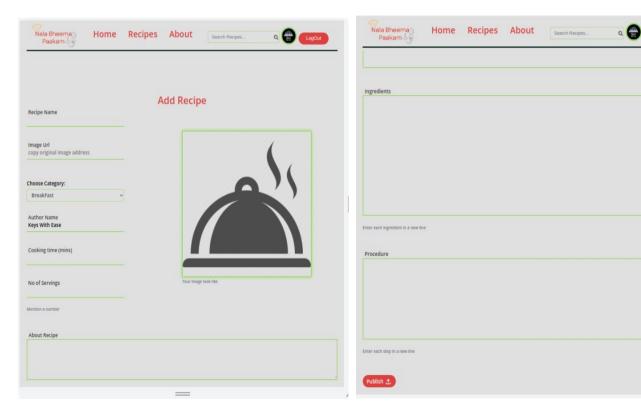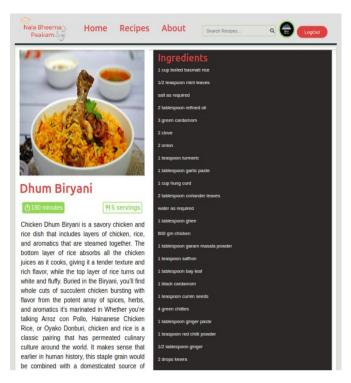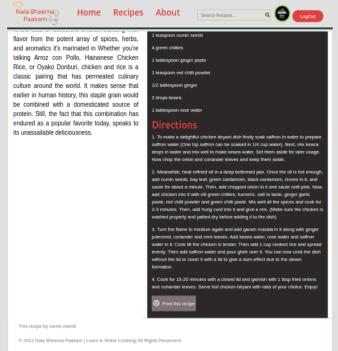## Searching:

Login Page:

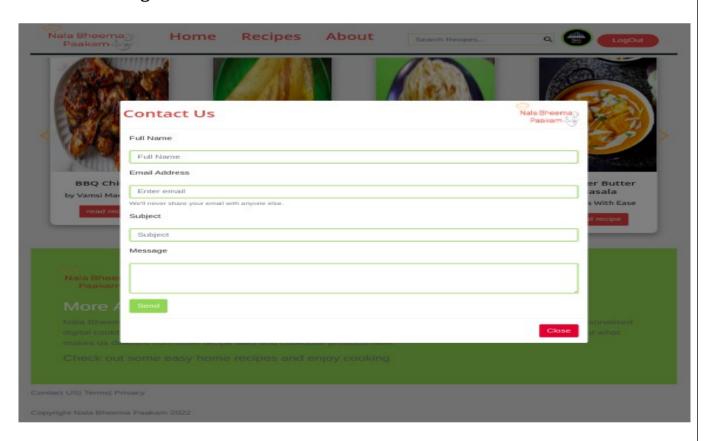## User Page:

Add Recipe Page:

Recipe Page:

Contact Us Page:

VII. STEEPLE analysis

- *Social*

- Increase social interaction and solidarity
- Meet new people and make new friends through the Recipe Forum

  - Technology

- web usage is increasing day after day
- The application is accessible from anywhere using a browser.
- Real time communication between actors

  - Environmental

- Increase of awareness regarding healthier foods and recipes.

  - Economical

- Recipe Organizer will enable users to try good and less costly recipes

  - Political

- Recipe Organizer may help the government in sharing good and useful information

  through ads.

  - Legal

- Insurances of users.

  - Ethical

- Client confidentiality should be kept: all information related to users should only be

  communicated to their respective users.

VIII. Future Work:

As a future work, I am planning to persist in developing more web apps and entering
deeply the world of web development. Nala Bheema Paakam has helped me to gain a lot of
development skills and enrich my background, as I spent the previous 4 months searching for
every tiny detail that concerns the development of web application. Thankfully, I have built
a good knowledge. Therefore, any upcoming project of web application development will
undoubtedly be within my reach. In addition, as to the future of Nala Bheema Paakam, I will deploy
it in **Netlify** and **heroku** and update the app from time to time if necessary. I have already started
the deployment phase . I will export my database soon, and then buying an account in
AWS to publish the app. Also, I will monetize my application .

IX. Conclusion:

The NBP application meets with the enterprise class application principles. It is designed to be performing, scalable , extensible, and highly available. It also ensures the privacy of the users' data and secures its access. Given that it may be improved in many ways, the application is also easily maintainable.

This documents summarizes the work that has been done since the beginning of this semester. Indeed, it starts by giving an overview about the project specification and requirements. The document also states the methodology followed and which consists of 5 main parts: The first part will be devoted to data gathering and software requirements specification. Consequently, I will have a look at different web apps which target the same goal. They are plenty of Recipe Organizer apps. Each one has some various features. The second part will be dedicated to the design phase, including the app and the database. Also, in this phase, the software tools to be used will be specified. For example, the IDE, the database Server, the modelling language for the design, and finally the software testing tools. The third part will be the implementation phase, here, the design will be converted to code in order to develop the targeted app. The fourth step will be devoted to testing the app. In this phase, two testing methods will be used, namely: Black Box testing and White box testing. The last phase will be the deployment phase.

X. References:

- Food book: https://myfoodbook.com.au/
- Bootstrap: https://getbootstrap.com/docs/4.3/getting-started/introduction/
- Stack Over Flow:https://stackoverflow.com/
- Canva Design: canva design