RECIPE SEARCH ENGINE USING YUMMLY API

by

RAJAVARDHAN MALLADI

B.Tech., Kakatiya University, 2014

A REPORT

submitted in partial fulfillment of the requirements for the degree

MASTER OF SCIENCE

Department of Computing and Information Sciences
College of Engineering

KANSAS STATE UNIVERSITY
Manhattan, Kansas

2016

Approved by:

Major Professor
Dr. Daniel Andresen

Abstract

In this project I have built a web application “Recipe Search Engine Using Yummly API”. This application is central information hub for the kitchen—connecting consumers with recipe ideas, ingredient lists, and cooking instructions. It will serve best for the people who uses digital tools to plan their cooking, these days almost everyone does.

The various features available for users in this application are as following. Users can search for their favorite dishes. The search results contain information about ingredients list, total time needed for cooking, user’s rating and cooking directions. Basic search filters are provided to filter out the search results like Breakfast, Lunch and Dinner recipes. The order of displayed results can be sorted according to ratings, total time required to prepare the dish. User can create an account and build their own favorite recipe collection by liking the recipes displayed. The liked recipes are stored into user’s account and user can view, add and delete those recipes anytime from his recipe collection. Users can use their social networking platform Facebook account credentials to log into this application or create a new account in this application.

The application will communicate with the Yummly API to consume data from it. The Yummly API is largest recipe information aggregator with over one million recipes data.

Contents

List of Figures

List of Tables

Acknowledgements

I would like to thank my Major Professor **Dr. Daniel Andresen** for his constant help and guidance throughout the project. I would also like to thank **Dr. Torben Amtoft** and **Dr. Mitchell L. Neilsen** for graciously accepting to serve on my committee.

Chapter 1 - Introduction

1.1 Motivation

The motivation to develop this “Recipe search engine using Yummly API” Web Application comes from my urge to learn technologies like ASP.NET with MVC, jQuery, AJAX and RESTful web services. A Web service is a service offered by an electronic device to another electronic device, communicating with each other via the World Wide Web. REST-compliant (REpresentation of State Transfer) Web services is one of the two identified major classes of Web services, in which the primary purpose of the service is to manipulate representations of Web resources using a uniform set of stateless operations. Also my interest to learn the implementation of OAuth to log users into this application. OAuth is an open standard for authorization, commonly used as a way for Internet users to log into third party websites using their Microsoft, Google, Facebook etc. accounts without exposing their password.

In this web application web pages are created using ASP.NET 4.6 implemented in MVC 5 architecture, jQuery is used for client side scripting and also for creating rich and interactive user interface. Bootstrap 2.0 is used for creating responsive user interface and styling the web pages. This web application will communicate with other web API using RESTful web services to pull data from it. C# and JavaScript languages are used for building the business logic of the application. SQL server 2014 is used for database designing and Entity Framework 6 which is an object relational model (ORM) tool is used to connect the application with database. Advanced technologies are used in building this web application thus making this application more effective. MVC 5 architecture is implemented i.e. model, view and controller layers are separated and independent of each other which increases the application performance in terms of speed, and also changes can be made in any layer without disturbing other layers.

Another motivation is to develop a powerful recipe search engine that will aptly search for the results the users are intending and display the obtained Also provide required filters to filter out the search results and sorting options to sort the results according to users’ interest.

Thus, using the latest technologies to develop a powerful search engine and get familiar with the web services are the main motivation for the project.

1.2 Project Overview

This web application is central information hub for the kitchen—connecting consumers with recipe ideas, ingredient lists, and cooking instructions. Nowadays people rely on web for any kind of information. So by building a web application we can serve many users who are looking for recipe ideas and instructions, list of ingredients for that recipe on the web.

The different functionalities available for the users in this application are as following. Users can search for their favorite dishes. The search results contain information about ingredients required, total time needed for cooking, user's rating and cooking directions. Basic search filters are provided to filter out the search results like Breakfast, Lunch and Dinner recipes. The order of displayed results can be sorted according to ratings, total time required to prepare the dish. User can create an account and build their own favorite recipe collection by liking the recipes displayed. The liked recipes are stored into user's account and user can view, add and delete those recipes anytime from his recipe collection. Users can use their social networking platform Facebook account credentials to log into this application or create a new account in this application.

The application will communicate with the Yummly API[1] to consume the data from it. The Yummly API is largest recipe information aggregator with over one million recipes data in it.

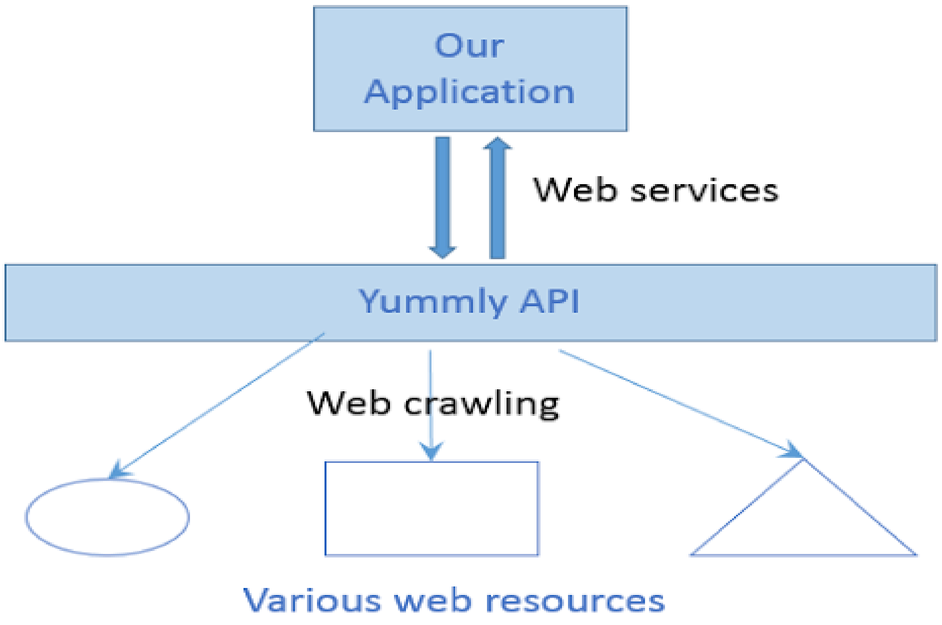


Figure 1.1 Diagram explaining interaction between our Application and Yummly API

The above diagram best explains the relation between the Yummly API and the project. Yummly crawls the web and gets recipes information from various sources on web and store recipes data in their servers. They provide web services for other authenticated applications to communicate with them and pull data from their servers. Our application will first authenticate with that API and then pull the required data from it into our application. These web API calls are completely RESTful in nature.

1.3 Problems with existing systems

The purpose of creating this Web Application is to outcast the discrepancies in hundreds of such existing systems on the World Wide Web. The various existing systems for recipe search and their disadvantages are discussed below.

Internet search engines:

One of the existing systems is Internet search engines like google, bing etc. If user wants to search for their favorite recipe, user will go to web, say google, and search for his recipe in his mind. The search results may redirect user to any other food blog website or any video casting site. But in that website all the information required like list of ingredients, cooking instructions, nutritional data and several other information may or may not be available at one place. Moreover you have to go through too many results to choose which result suits you best. This system involves more time and analysis which is a tedious task thus not ideal for many users.

Food Blogs:

Another existing system is Food blogs and channels on video sharing sites like YouTube. There are many food blogs which provide lot of information about recipes but the disadvantages with this kind of blogs is most of the times they don’t support multi cuisine recipes. Most often they belong to one region which is a disadvantage most of the times. Personally I follow few YouTube channels and one among those is ‘VahChef’ which contains interesting videos on how to cook various Indian dishes but if I want to cook some Mexican food I have to google it and follow some other blogs or video hosting sites. In this case of existing system, food blogs are mostly confined to one or two types of cuisine which is a disadvantage for users looking for multi cuisine.

For better understanding of the disadvantages of existing system consider a regular scenario, I went for shopping groceries and I’m planning to cook some new dish and I don’t know what ingredients are required and of what quantity are required. So I have to search on the web and I might be provided with numerous options throwing me into dilemma which one to rely and if it is video I have to manually jot down the ingredients or remember those list of ingredients which is not at all an ideal thing to do. These all are big drawbacks of existing system and this application is capable of addressing all these issues in an efficient way providing an ideal platform to search for any food recipe and get all the required information such as ingredients, nutritional data, cooking instructions. This application supports all multi cuisine thus serving wide range of people.

1.4 Intended Audience

The intended audiences for this website are the users who rely on web for cooking dishes and also who search for new recipe ideas on web. This web application acts as bridge connecting users with recipe ideas, ingredient lists, and cooking instructions.

Chapter 2 - System Requirements Analysis

2.1 Requirements Gathering

Requirements Gathering is one of the crucial part of application development process. I have identified the requirements by keeping myself in user perspective and thought what features must be included in the application such that it can serve the user in the best way. I have also discussed with my friends to identify requirements and included them in the application. Few assumptions are also made assuming they will be existing in the application using environment. Below discussed are the major requirements of the application and the assumptions made.

2.1.1 Requirements

- User can search for recipe idea in the search box provided and application will list all the results obtained for the search criterion with all the required details.
- User can filter the search results by directly clicking on the breakfast, lunch, dinner and desserts tabs provided in the application.
- User can sort the results on users rating available for each search result.
- User can sort the results on time required to prepare the dish.
- User can register with the application by providing asked details.
- User can also use their social networking platform Facebook to log in to this application.
- User can login and like the recipes and make their favorite collection stored into their account.
- User can view, add and delete recipes from his favorite collection.

2.1.2 Assumptions

- User will have internet connection while using this web application.
- User will have Internet Explorer 5.0 or high IE version or any other browsers with latest versions.
- User has basic knowledge on how to use, navigate through website.
- User has ability to search for recipes and understand the search results in English language.

Chapter 3 – Technologies

This chapter includes the details of the latest technologies and tools used to build this application. Following are the technologies used in the development of the application.

- **ASP .NET MVC 5**[2]**:** It is a framework for building standard and scalable web applications. This framework makes use of the MVC pattern.

- **C#:** It is an object oriented programming language which was developed by Microsoft. It is a general purpose language which has proven to be very efficient to develop web applications.

- **jQuery:** jQuery is a library of JavaScript which is a programming language for the web, used to create dynamic web pages. In this project most of the views are written in jQuery. These scripts are responsible for features such as user input validation (e.g., checking length of password)

- **SQL Server 2014:** Microsoft SQL Server is a relational model database server developed by Microsoft. Its primary query languages are T-SQL and ANSI SQL. SQL server Management Studio is used to create the RDBMS schema and specify various constraints like referential constraints during creation of a schema. To test queries and assertions, the user interface of management studio is very helpful throughout the process.

- **LINQ:** LINQ stands for Language-Integrated Query. Unlike traditional queries, where query output is expressed as simple strings without any type checking at compile time, 59 LINQ queries are written against strongly typed objects which hold the output of the query without any data loss (these objects match the table structure). It is also easier to use these objects in our code.

- **Visual Studio 2015:** Visual studio is an Integrated Development Environment (IDE) developed by Microsoft and it helps to build various ASP.NET web applications. It also helps in building console as well as GUI applications that can be in native code combined with managed code for all platforms that can be supported by Windows, .NET framework etc. There are several other built-in tools which include a forms designer which can also be used to build GUI applications, web/ class/ database schema designers etc.

- **Facebook SDK for JavaScript:** Facebook provides SDK in JavaScript[3] for connecting to Facebook and access graph API, request data from third party applications.

- **Apache JMeter:** Apache JMeter[4] is used as a load testing tool for analyzing and measuring the performance of a variety of services, with a focus on web applications.

Chapter 4 - System Design

Requirements gathering followed by careful analysis leads to a systematic Design i.e. Object Oriented Design (OOD).

4.1 Use-Case Diagram

A use case diagram depicts the application from an external observer perspective. Use case diagram identifies the agents of the system and the functions that these agents perform with the system. In this application only one agent or actor is identified i.e. user who interacts with application and performs various tasks. These various actions are depicted in the below use case diagram.

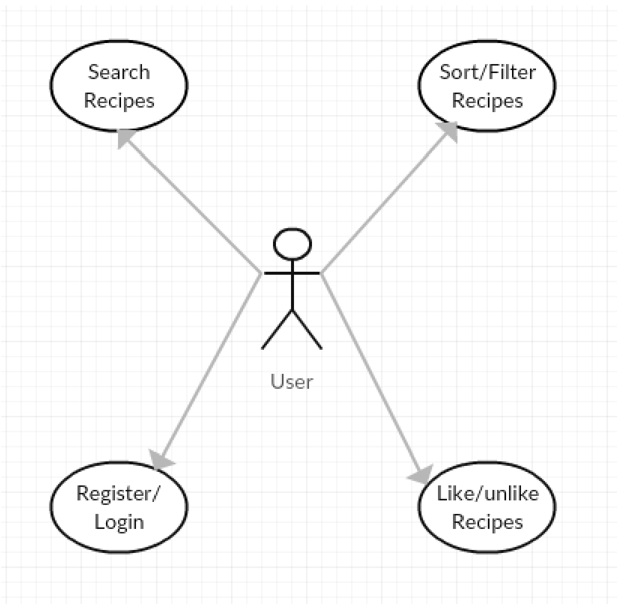


Figure 4.1 Use case diagram

The different actions that actor i.e. user can perform with application are register with application, login to application, search for recipes, sort the results, create and manage favorite collection, logout of application.

4.2 Class Diagram

In UML, class diagram is a static structure diagram which describes the structure of a system by showing the classes, attributes and their relationships. It is the main building block in object oriented modeling. The classes represent the structure or framework for the main objects and interactions in the application. The class diagram consists of classes represented in boxes which contain three parts. The name of the class is contained in the upper part, with the attributes of classes in the middle part and the bottom part contains the methods or operations that the classes undertake. The following figure shows the class diagram of the application.

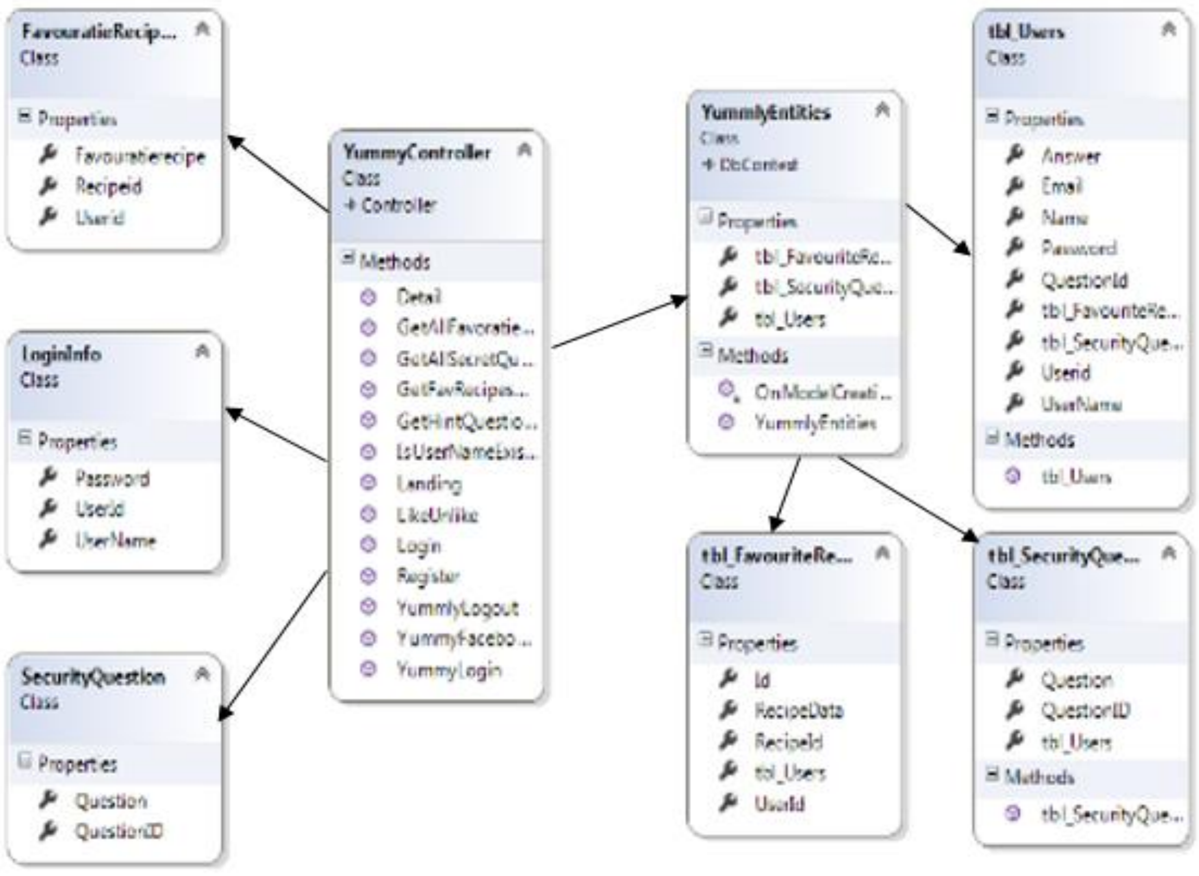


Figure 4.2 Class diagram

Controller class:

This class is the most important class of this application. It has the business logic to control the whole application. The different methods in this application are discussed briefly as below.

Landing: This is invoked when homepage of application is accessed. In this method, session is checked for user login details, if contains then user username is returned to the view. If session is empty username with null is returned which means no logged in user.

YummyLogin: This method is invoked when user clicks on login button. This method will validate credentials provided by user and if valid starts a new session else returns an error message.

GetHintQuestionByUsername: If user forgets password, to retrieve password user has to answer the hint question and this method will verify the answer provided by user matches the answer provided by user during registration. If valid then returns password else it returns error message.

IsUserNameExists: During registration, user will select a username and this has to be unique. This method is invoked to check whether the username is available or already exists.

GetAllSecretQuestions: During registration user has to select and answer a secret question which will be helpful to retrieve password if user forgets password.

Register: It is invoked when user clicks on register button during registration. This method will validate all the details given by user and stores them into the database.

YummyLogout: It will just remove the session containing logged in user details.

Detail(): This method will simply return the detail view. Detail view has the html code to display content and jQuery code for the business logic.

LikeUnlike: This method is invoked when user clicks on like button for a recipe. It will take FavouratieRecipeInfo class object and connects to database and checks whether this recipe is liked by user or not, if liked it will unlike and if not like it will mark as liked and store its details into database. This will handle the business logic for both like and unliking a recipe.

GetFavRecipesByUserId: When user tries to access favorite collection, this method is called. This method will connect to database and pulls information of all the liked recipes and returns them to the view which will be displayed to user.

LoginInfo class:

This class has only attributes to store login information of a user. The different attributes in this class are UserId, Password and UserName.

FavoriteRecipeInfo class:

This class has only attributes to store login information of a user. The different attributes in this class are Userid, FavouriteRecipeInfo, Recipeid.

SecurityQuestion class:

This class is has attributes QuestionID and Question.

All the last three class mentioned doesn't have any methods in them, they are classes in the model with attributes to hold data.

The classes **tbl_FavouriteRecipes, tbl_Users, tbl_SecurityQuestions** are used to map the objects of these classes to the database. The tables in the database are created basin on these class objects by Entity framework which is an ORM tool.

4.3 Design Goals

The goals behind this design are listed as follows.

- Crucial design goal of this application is to implement good search engine to get relevant search results and short list the obtained results.
- Build an interactive front end, making users to navigate from one web page to another webpage seamlessly with no effort.
- MVC design always optimizes the size of the code and keeps the code clean and simple.
- It also involves designing fast responsive web pages through the usage of session and view states thereby considerably reducing the cost of post backs to the server using AJAX.
- Follow Object-relational mapping technique (a.k.a. design pattern) for accessing a relational database from the programming language.
- Providing different options for users who are logged in and not logged in. Only logged users can like the recipes and create favorite collection. Use sessions to identify the logged in users.

4.4 System Architecture:

This application is implemented in MVC architecture. MVC is a framework methodology that divides an application's implementation into three component roles: models, views, and controllers.

- "Models" in a MVC based application are the components of the application that are responsible for maintaining state. Often this state is persisted inside a database (for example: Login class with fields username, password, email id that is used to represent user data from the users table inside SQL).
- "Views" in a MVC based application are the components responsible for displaying the application's user interface. Typically this UI is created of the model data (for example: Login page is a view that is shown when user clicks on “login” tab on homepage. It will have textboxes and buttons).
- "Controllers" in a MVC based application are the components responsible for handling end user interaction, manipulating the model, and ultimately choosing a view to render it to display UI. All the business logic of the application is written in the controllers. In a MVC application the view is only about displaying information - it is the controller that handles and responds to user input and interaction.

The benefits of using a MVC methodology is that it helps enforce a clean separation of concerns between the models, views and controllers within an application. Maintaining a clean separation of concerns makes the testing of applications much easier, since the contract between different application components are more clearly defined and articulated.

ASP.NET MVC Framework

ASP.NET MVC is a very good platform for creating unit-testable applications that are maintainable in the long run, as well as for achieving a clearer separation between the UI and the UI-handling logic. The ASP.NET MVC framework was released by Microsoft as an alternative approach to web forms when creating ASP.NET based web applications. The ASP.NET MVC framework is not a replacement or upgrade of web forms, but merely another way of programming your web applications so that we can get the benefits of an MVC design with much less effort.

How the code of this application is organized according to MVC pattern using the ASP.NET template is explained in detail in below.

4.4.1 Model:

The Model is the lowest level of the pattern and is responsible for maintaining data. The different class declared in the model are discussed below.

LoginInfo: This class holds the data related to login information of user such as UserId, UserName, and Password.
FavourtieRecipeInfo: The different attributes to hold information about recipe declared in this class are Userid, Favouriterecipe, Recipeid. Object of this class is created when a user likes a recipe.
SecurityQuestion: This class is used to store data about security question such as QuestionID and Question, used when user tries to access forgot password option in login page.
The objects of *tbl_Users*, *tbl_FavouriteRecipes*, *tbl_SecurityQuestions* classes are mapped to the database using entity framework. About this is discussed in detail in the next section.

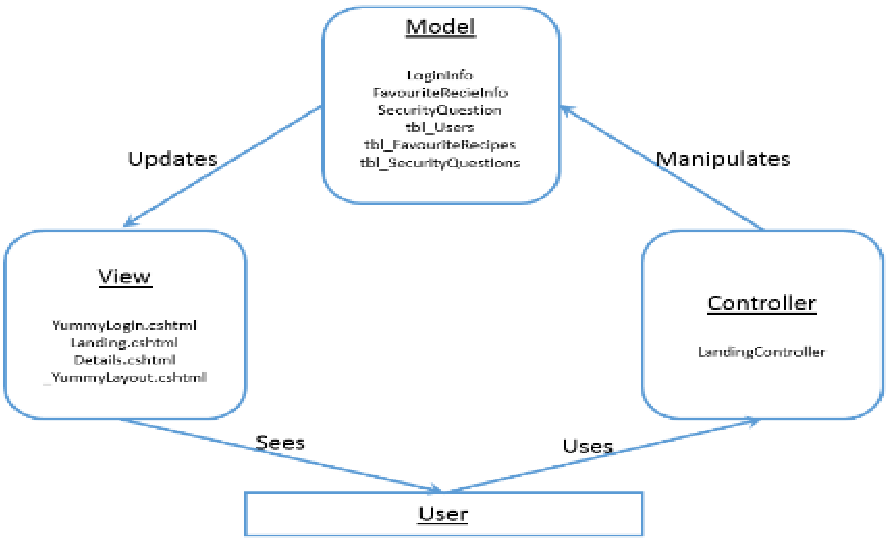


Figure 4.3 MVC architecture of this application

4.4.2 View:

The different views in the application are:

YummyLogin.cshtml: This view is the cshtml file which contains code of HTML, CSS and jQuery logic to make AJAX call to the controller. This view is responsible for displaying the login page of application and looking after the required logic to be performed behind the HTML input controls.

Landing.cshtml: This view is responsible for displaying and handling the main page of application. HTML and CSS is used to display the web page. "Landing.js" file has the jQuery logic to generate and display the partial views in the main page. For example when user clicks on "Lunch" tab, respective partial view is generated and displayed. From users point of view it looks like the application has many web pages but actually it's just one main page acting as different webpages.

Detail.cshtml: When user clicks on one of the recipe from the search results displayed, a new web page is created dynamically and HTML code for that webpage is in this view. The logic for generating this view dynamically is written in "Detail.js".

YummyLayout.cshtml: This view contains the HTML code for displaying layout i.e. headers and footers for the main page. This Layout is shared and used for all partial views in the main page.

4.4.3 Controller

All the business logic of the application is present in the controller. I have used only one controller which has different methods to implement different business logic. The various methods in the controller and there utilities is discussed in section 4.2.1.

This application will communicate with the Yummly web API to get recipe data in form of JSON (JavaScript Object Notation) from it using web services. The web services can be implemented in various ways. This application follows RESTful web services.

4.4.4 Web services

Web services [5] are client and server applications that communicate over the World Wide Web's (WWW) HyperText Transfer Protocol (HTTP). As described by the World Wide Web Consortium (W3C), web services provide a standard means of interoperating between software applications running on a variety of platforms and frameworks. Web services are characterized by their great interoperability and extensibility, as well as their machine-processable descriptions, thanks to the use of XML. Web services can be combined in a loosely coupled way to achieve complex operations. Programs providing simple services can interact with each other to deliver sophisticated added-value services.

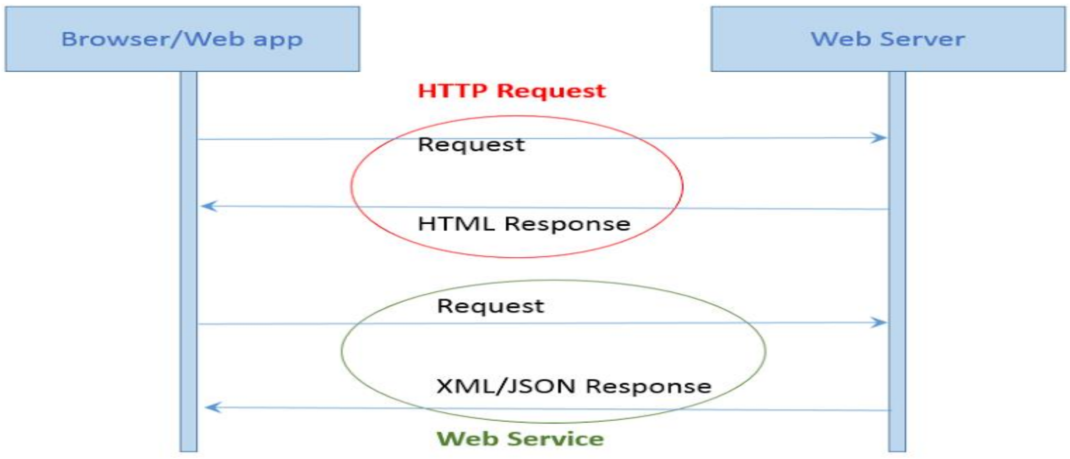


Figure 4.4 Diagram explaining web service request

REST: Representation State Transfer REST means Representational State Transfer, an architectural pattern used to identify and fetch resources from networked systems such as the World Wide Web (WWW). The REST architecture was the foundation of World Wide Web. But the term itself came into being around the year 2000, and is quite a buzzword these days. The core principle of REST is to facilitate the sharing of resources via unique identifiers, just as we use Uniform Resource Identifiers (URIs) while accessing resources on the Web. In simple terms, REST specifies how resources should be addressed, including URI formats, and protocols such as HTTP. The term resources include files such as ASPX pages, HTML files, images, videos, and so on.

A RESTful design may be appropriate when the following conditions are met [6].

- The web services are completely stateless. A good test is to consider whether the interaction can survive a restart of the server.
- A caching infrastructure can be leveraged for performance. If the data that the web service returns is not dynamically generated and can be cached, the caching infrastructure that web servers and other intermediaries inherently provide can be leveraged to improve performance. However, the developer must take care because such caches are limited to the HTTP GET method for most servers.
- The service producer and service consumer have a mutual understanding of the context and content being passed along. Because there is no formal way to describe the web services interface, both parties must agree out of band on the schemas that describe the data being exchanged and on ways to process it meaningfully. In the real world, most commercial applications that expose services as RESTful implementations also distribute so-called value-added toolkits that describe the interfaces to developers in popular programming languages.
- Bandwidth is particularly important and needs to be limited. REST is particularly useful for limited-profile devices, such as PDAs and mobile phones, for which the overhead of headers and additional layers of SOAP elements on the XML payload must be restricted.

4.5 Database Design

This web application is supported by the SQL Server 2014 as database engine to store and maintain data of the application. Entity Framework 6.0 is used to connect database and application.

Entity framework:

The Microsoft ADO.NET Entity Framework is an Object/Relational Mapping (ORM) framework that enables developers to work with relational data as domain-specific objects, eliminating the need for most of the data access plumbing code that developers usually need to write. Using the Entity Framework, developers issue queries using LINQ, then retrieve and manipulate data as strongly typed objects. The Entity Framework's ORM implementation provides services like change tracking, identity resolution, lazy loading, and query translation so that developers can focus on their application-specific business logic rather than the data access fundamentals [7].

In the below diagram tbl_Users, tbl_FavouriteRecipes, tbl_SecurityQuestions are the model class defined to hold data of the application. Entity framework will create the tables in the database engine based on these classes.

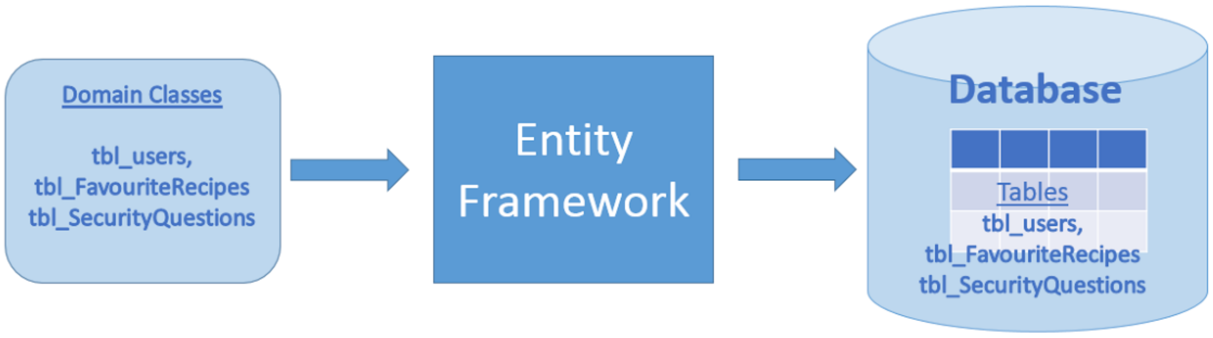


Figure 4.5 Diagram showing how entity framework maps objects to tables in Database.

4.5.1 Database Diagram

The database diagram for the application is generated using Microsoft SQL Server management studio. The diagram is displayed below.

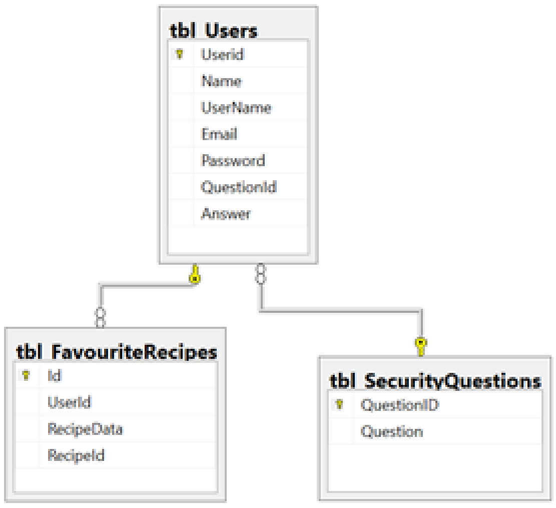


Figure 4.6 Database Diagram of application.

The database diagram has three tables which are generated from the entity classes by entity framework. The table ‘tbl_Users’ is used to store the information of users like Userid, Name, UserName, Email, Password, QuestionId, Answer. ‘UserId’ is primary key in this table. ‘tbl_FavouriteRecipes’ is used to store information about recipes liked by users. Different fields in this table are Id, UserId, RecipeData, RecipeId. These two tables have UserId as a foreign key. Another table is ‘tbl_SecurityQuestions’ with fields QuestionId and Question to store information about security questions used at user registration. Primary key in this table is ‘QuestionId’. This table and ‘tbl_Users’ have QuestionId field as a foreign key.

4.5.2 Database connection strings

The connection string is used to connect the application and database. These connection strings are declared in the web.config file so that whole application can access these connection strings. Below is the code snippet from web.config file containing connection strings section written in xml format.

```
<connectionStrings>
        <add name="yummlyconn" connectionString="Data Source= LAPTOP-RCALEOII\
                SQLEXPRESS; Initial Catalog=Yummly;" />

          <add name="YummlyEntities" connectionString="metadata=   r
                es://*/YummlyModel.csdl|res://*/YummlyModel.ssdl|res://*/YummlyModel.m
        sl;provider=System.Data.SqlClient;provider connection string=&quot;data
        source=LAPTOP-RCALEOII\SQLEXPRESS;initial catalog=Yummly;
        MultipleActiveResultSets=True;App=EntityFramework&quot;"
        providerName="System.Data.EntityClient" />
</connectionStrings>
```

Chapter 5 - Implementation

The functionalities of the application can be broadly divided into three modules. First, implementation of recipe search engine, filtering and sorting the results. Second, registering and logging in the user. Third, Logged in user can create, view, add and delete favorite collection.

5.1 Recipe Search Engine Module:

In this module the functional requirement covered is mainly the implementation of the recipe search engine. The implementation of the search engine can be breakdown into following things. Providing a search box for the user to click on it and search for any recipe. Then listing the search results using a grid format. Basic filters like Breakfast, Lunch, Dinner, and Dessert are provided as tabs in the main page of the website so that when user clicks on one of these tabs the results related to that tabs are listed to user. When user clicks on a recipe, a new web page will be opened in another tab of browser with complete details of the recipe.

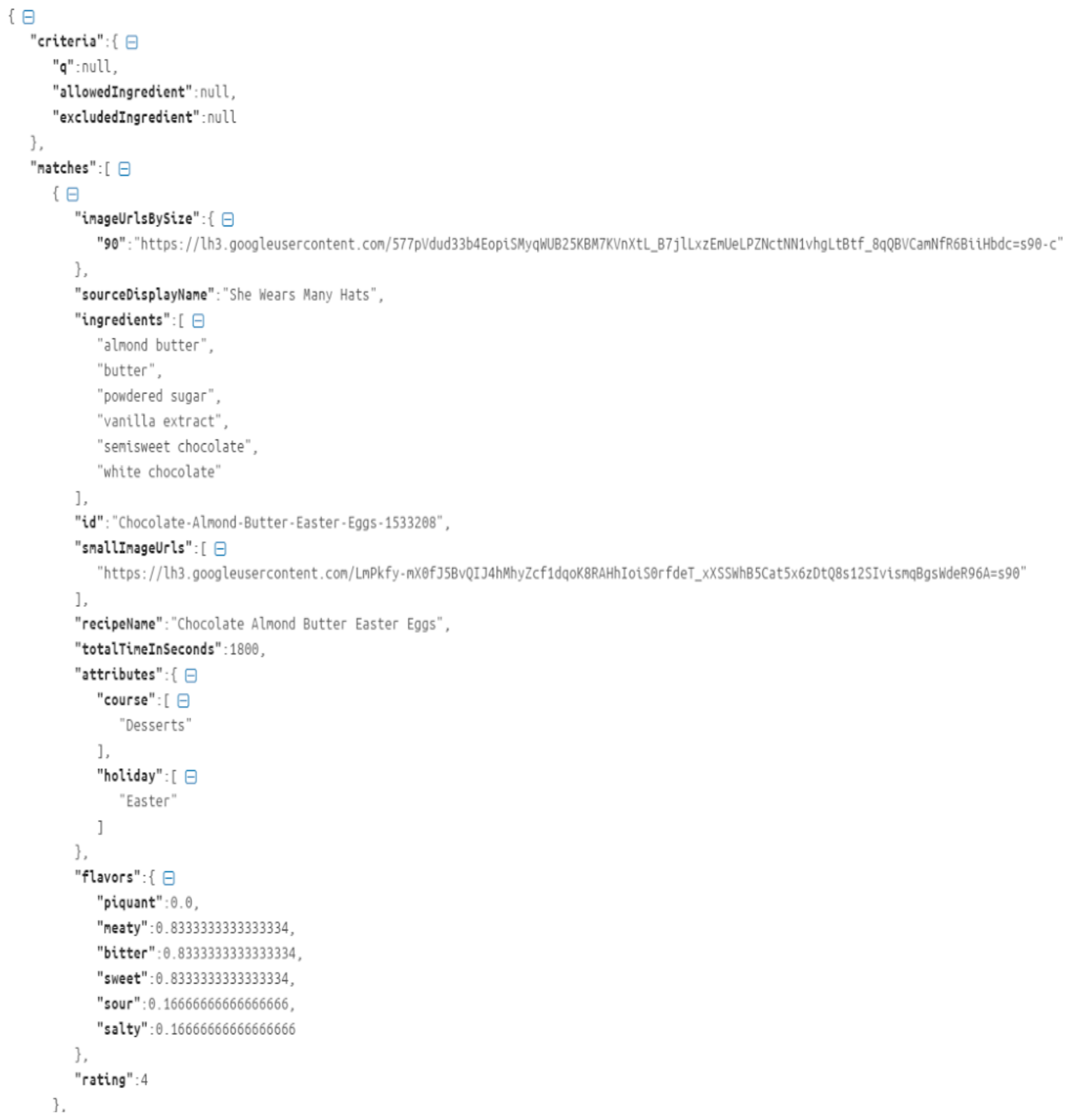
This application will get recipe data from the Yummly API. The Yummly API is largest recipe information aggregator with over one million recipes data in it. Yummly API will crawl the web and get all the recipe data found all over web and store in their servers. They have created web services and helping other authenticated applications to consume their web services. For that first we need to register as a developer in their website and request for an app-id and app-key. These both are used as credentials to authenticate our application. As discussed earlier RESTful API calls are made to the API to get data. Application will make a get request whenever we need data to the url specified by the Yummly team on their website called as endpoint. We must also send app-id and app-key for every call to the web API to authenticate the calls. Yummly web API will respond to our get request by checking the credentials we submitted to the API and if valid it will send out the requested data in form of JSON.

Below a sample get request call to API written in AJAX is explained.

```
App-id = “*****20f7”
App-Key = “*******************aa9663a968f9”

$.ajax({
        url: 'http://api.yummly.com/v1/api/recipes?_app_id=' + appid + '&_app_key=' + 
appkey,
        dataType: "json",
        data: {
          term: request.term,
        q: $("#searchInput").val() //Search Parameter
    },
```

Yummly API will respond to the above get request with recipes data in form of JSON. That response is of below type and its different parameters are as shown in figure below.



```
{
   "criteria":{
      "q":null,
      "allowedIngredient":null,
      "excludedIngredient":null
   },
   "matches":[
      {
         "imageUrlsBySize":{
            "90":"https://lh3.googleusercontent.com/577pVdud33b4EopiSMyqWUB25KBM7KVnXtL_B7jlLxzEmUeLPZNctNN1vhgLtBtf_8qQBVCamNfR6BiiHbdc=s90-c"
         },
         "sourceDisplayName":"She Wears Many Hats",
         "ingredients":[
            "almond butter",
            "butter",
            "powdered sugar",
            "vanilla extract",
            "semisweet chocolate",
            "white chocolate"
         ],
         "id":"Chocolate-Almond-Butter-Easter-Eggs-1533208",
         "smallImageUrls":[
            "https://lh3.googleusercontent.com/LmPkfy-mX0fJ5BvQIJ4hMhyZcf1dqoK8RAHhIoiS0rfdeT_xXSSWhB5Cat5x6zDtQ8s12SIvismqBgsWdeR96A=s90"
         ],
         "recipeName":"Chocolate Almond Butter Easter Eggs",
         "totalTimeInSeconds":1800,
         "attributes":{
            "course":[
               "Desserts"
            ],
            "holiday":[
               "Easter"
            ]
         },
         "flavors":{
            "piquant":0.0,
            "meaty":0.8333333333333334,
            "bitter":0.8333333333333334,
            "sweet":0.8333333333333334,
            "sour":0.16666666666666666,
            "salty":0.16666666666666666
         },
         "rating":4
      },
```

Figure 5.1 JSON data reply for the get request made to Yummly API

The criteria section has information of parameters of the request sent to API, ‘q’ is for the search text, ‘alloweIngredient’ and ‘excludeIngredient’ tells information about the ingredients to be included and excluded from search query. In this application we are not handling those so we can ignore these details.

The next section contains the information of recipe matched for the search request. For each recipe the different information available are ‘imageUrlBySize’- web address from where image is obtained, ‘sourceDisplayName’-from where the recipe data is obtained by the API, ‘ingredients’-list of ingredients used in the recipe, ‘id’ – id of the recipe to uniquely identify it, ‘smallImageUrls:’- the shortened version of image url, ‘recipeName’- name of the recipe, ‘totaltimeInSeconds’- time taken to prepare the dish, ‘attributes’- different attributes such as ‘course’, ‘cuisine’ and ‘holiday’ are provided which can be used to categorize the search results, ‘flavors’- different taste flavors are shown in percentages and ‘rating’- user rating for the recipes. I have used most of the data obtained as JSON response excluding few parameters in this application.

When user searches for a recipe in the search box, application will make an appropriate get request call to web API by providing required parameters. And then API will respond to request and send data in form of JSON. Business logic to parse the JSON is written and parsed data is sent to View to display the results to user in a grid format.

Filters:

Basic filters are provided on the website in form of tabs. When user clicks one of the tabs say “Lunch”, application when invoking a get request to the web API it will add “Lunch” keyword to the url. Web API will read the url and understands that application is asking for recipes related to lunch.

Below is sample get request call which pulls Lunch recipes data.

```
$.ajax({
        url: 'http://api.yummly.com/v1/api/recipes?_app_id=' + appid + '&_app_key=' +
appkey,
        dataType: "json",

        data: {
           term: request.term,
           q: $("#searchInput").val() //Search Parameter
        },
```

The ‘q’ parameter in the request will be the filter applied for example if user clicks on ‘Lunch’ tab in main page of application in the background when get request is being send to API this ‘q’ parameter is added. API will respond to this request by sending recipes related to Lunch.

Sorting:

The search results can be sorted according to the rating of the recipe. Each recipe has user rating and this will range from 1-5 stars. When user clicks on the sort tab the results are sorted recipes with high user rating are displayed first and low rating recipes are displayed last. This sorting order can be reversed by double clicking the sort tab. User can sort the recipes according to time required to cook the dish. User searches for recipes and web API will return all the available results in JSON format. The data is array of JavaScript objects. ‘SortByRating’ JavaScript function will parse these objects and sort the objects in array according to rating and ‘SortByTime’ is used to sort results by time taken. After sorting, sorted data is sent to view which will display the results in the grid format.

5.2 Register and Login Module

Users can create their own favorite collection by liking their favorite recipes. To create favorite collection user has to register and login into the application. Liking the recipes and creating favorite collection option is only provided to logged in, these features are hidden for users not logged in. There are two ways for user to login to application, one is application’s login and another is social networking platform Facebook credentials. They are discussed in detail below.

5.2.1 Application’s credentials

There is a login page in the application where user has to provide username and password and login to application. If user is a new user, one has to fill the registration form and register with the application. The details asked at registration are Name, Username, Email, Hint question and answer. Username, email id must be unique for each user. Necessary validation are written to validate the details provided by user. Suppose if user enters a username which is already taken by some other user then application will warn the user to choose another username. Application will connect to the database and checks the availability of username. Application will create a new session for each user whenever user signs into application. This session contains user’s information such as username, favorite collection which will be displayed on webpage and status to check whether s logged in or not. There will be a ‘Like’ button on each search results displayed in a grid format. This ‘Like’ button is disabled and hidden when user is not logged in.

5.2.2 Facebook’s credentials

Users has option to choose to login into the application using social networking platform ‘Facebook’ credentials. This option is very useful for both the users and application point of view. With this option users can provide their ‘Facebook’ login credentials to login to application. User need not have to register with this application and remember another username and password which is a big burden on users to remember credentials. From application point of view using Facebook’s credentials of users to login to application increases the count of users who are registering into application. To implement this other platform authentication, OAuth protocol is used. OAuth [8] is an authentication protocol that allows users to approve application to act on their behalf without sharing their password. To connect with the ‘Facebook’ API, Facebook provides a Facebook SDK for JavaScript which implements OAuth. This ‘Facebook’ SDK for JavaScript has all implemented functions and can be easily used and get required results. First, this application is registered with the ‘Facebook’ and ‘Facebook’ provides an app-id and app-secret to authenticate this application with the ‘Facebook’. Facebook provides basic data of users such as email id, username, user id. For other user details, a request has to made to ‘Facebook’ and seek permissions from ‘Facebook’.

I have created app on ‘Facebook’ to access users’ details. While creating app ‘Facebook’ asks for various details and they are the type of app say food, games etc. , platform like web, android etc. from which app will connect to ‘Facebook’, from which url does the app sends request to ‘Facebook’ graph API and upon successful validation of user the application has to be redirected to where i.e. callback url of application. Upon on successful creation of app, Facebook will generate app-id and app-secret which are to be passed with each call to Facebook graph API. Below are the details I mentioned which I have provided to ‘Facebook’ while creating ‘Facebook’ app.

App-name: yummlywebapp
Platform: web
Site url: http://localhost:56849/Yummy/YummyLogin
callback url: http://localhost:56849/Yummy/YummyLogin

App-id:***********62582
App-secret: *****************957301be8ce13

5.3 Manage favorite collection

Logged in users can create and manage their favorite collection of recipes. When user clicks on the ‘Like’ button recipe is added to user’s favorites. User can view all liked recipes in favorite tab on the webpage. When a user likes a recipe user will store the recipe information by converting JSON data to string format in the database by assigning a unique id to recipe. So when user clicks on the ‘Favorites’ tab, application will connect to the database and pulls the recipe information convert back to JSON data and then it is parsed and feed to the view which will display this data in grid format. Users can delete the recipe from his favorite collection by unliking the recipe i.e. just clicking on the ‘Like’ button again. Application will delete the recipe record from the table in database.

Chapter 6 - Testing

Software testing is the process of executing a program to find the bugs in the application. It is nothing but validating and verifying that a software program is working as expected and is meeting the technical as well as business requirements. This can be implemented at any time in the development process. There are many types of testing software such as unit testing, black box testing, performance testing, stress testing, regression testing etc. Unit testing, White box testing and load testing are performed for this application.

6.1 White Box Testing

Functional testing by validating the appropriate algorithms and data structures is done by white box testing. This testing ensures that

- All loops are exiting appropriately; i.e., loops in various parts are implemented appropriately.
- All conditional statements handle both the true and false statements.
- All the sections of the code are reachable by the control.
- All the function calls are possible.

Each and every module is tested manually and rigorously with various inputs to observe the behavior of each and every module in the code base.

6.2 Unit Testing

Unit testing refers to those tests that verify the functionality of a specific section of the code, usually at the function level. In an object oriented environment this is usually at the class level and the unit tests include the constructors and the destructors. The primary goal is to take the smallest piece of testable software, isolate it from the rest of the code and check if it is behaving exactly as expected. Each unit is tested separately before integrating them into modules to test the interfaces between modules.

Test cases and Results:

Each test case tests the functionality of a particular function. Following are the test cases that were designed and executed to check the functional requirements of the application.

	Test Module	**Test Case description**	**Expected Result**	**Test Result**
1	Login page	Enter valid login credentials	User logged in successfully and redirected to Main page.	Pass

2	Login page	Enter invalid login credentials	Error message displayed.	Pass
3	Registration Page	Enter valid details asked for registering new user.	User is registered successfully.	Pass
4	Registration Page	Enter invalid details asked for registering new user.	Error message displayed.	Pass
5	Main page	Search for valid recipe in search box	Valid search results are displayed in grid format.	Pass
6	Main page	Search for invalid recipe in search box	No results are displayed.	Pass
7	Main page	Click on ‘Lunch’ tab on Main page. (Similarly for all tabs on Main page such as ‘Breakfast’, ‘Desserts’ etc.)	Lunch recipes are displayed.	Pass
8	Main page	Check for like button on recipe data in grid format while user is not logged in. (similarly Favorites tab on Main page)	Like button is not visible (Favorites tab is not visible).	Pass
9	Main page	Check for like button on recipe data in grid format while user logged in. (similarly Favorites tab on Main page)	Like button is visible. (Favorites tab is visible)	Pass
10	Main page	Like recipes and access them from Favorites collection.	Liked recipes are available in favorite recipe.	Pass
11	Main page	Unlike recipes in favorites collection.	Recipes unliked are not available in favorites collection	Pass
12	Main page	Choose “Sort by rating ” / “Sort by Time”	Recipes are displayed in sorted according to rating/ Time.	Pass
13	Main and login page	Check for responses of ajax calls to controller	Getting expected responses	Pass

Table 6.1 Test cases and results

6.3 Performance Testing

A web application has to have the features of sustaining huge volume of requests and provide high throughputs at all the times and under all kinds of loads. To test this feature we need to perform stress and load testing for the application. In order to test this we need to simulate situations where in we have multiple users requesting pages on a website and have to check whether there is any performance degradation because of this.

6.3.1 Tool for performance testing

The website has been tested for its performance by the tool Apache JMeter, which are open source software and a pure java based application. JMeter has been designed to load the test functional behavior and measure performance. This can be used to test performance on static and dynamic resources such as files, databases and queries and more. It can also be used to simulate heavy load on a server, network or to test the strength or to analyze the overall performance under different load types.

6.3.2 System Configuration

The details of the configuration of the system which has been used for testing the application are listed below:

Operating System	Windows 8 – 64 bit
RAM	8 GB
Processor	Intel core i7
Processor Speed	3.40 GHz

Table 6.2 System configuration for testing

The below listed tests are conducted on the system with the above mentioned configuration.

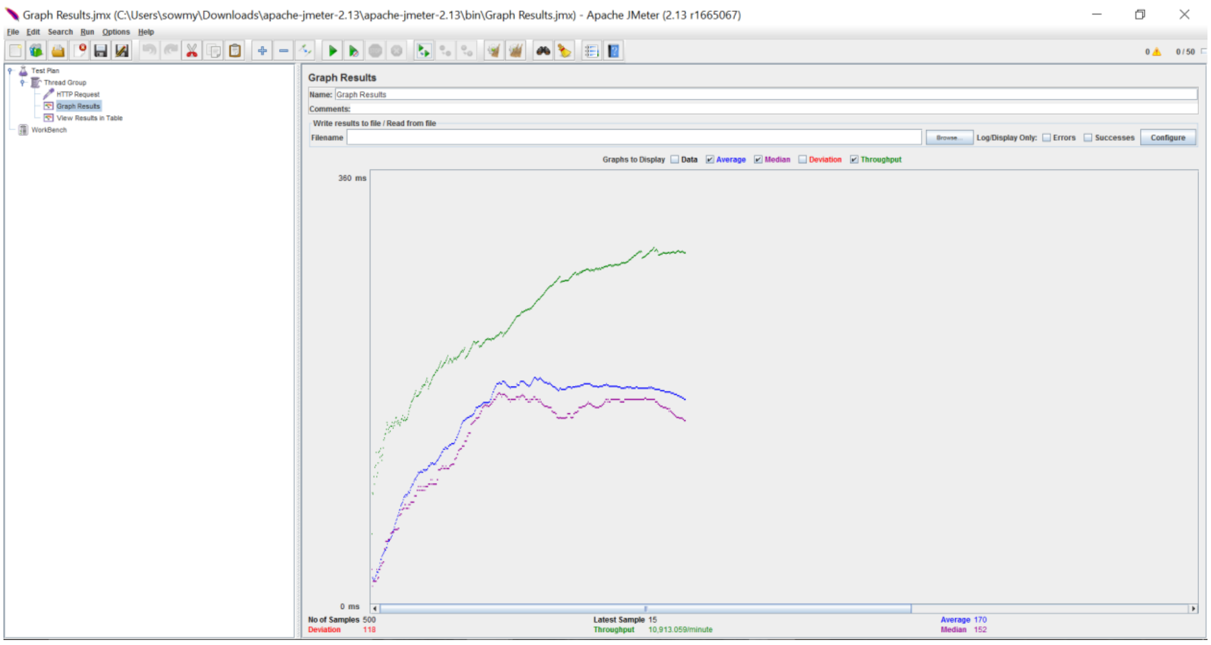
6.3.3 Test plans Results and Evaluation

The results would show the response times as well as throughput, when increasing the number of users accessing the application at the same time.

Test 1: JMeter test for Main page:

The main page which contains all the tabs is tested to analyze its behavior and to get the idea about the performance of the application. The analysis would show the throughput as well as response times for the web page when number of users is accessing it. This test plan is also used to get an approximate estimation of maximum number of users who can access it at the same time.

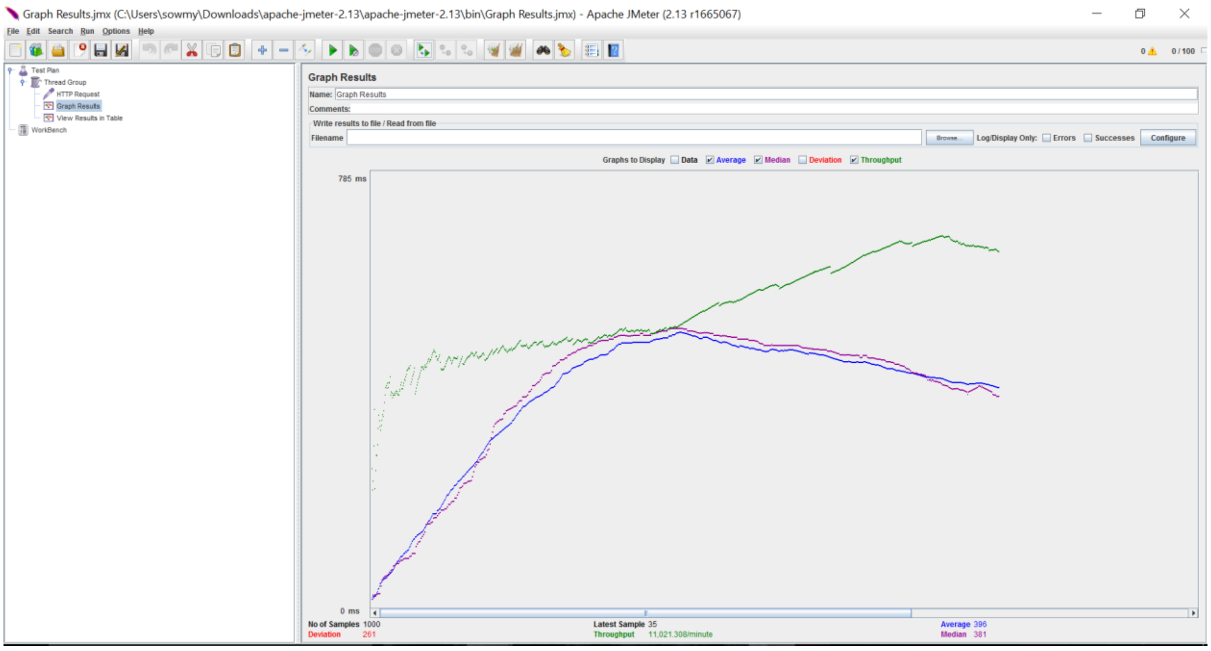
First the test case is executed with a total of 50 threads, simulating 50 simultaneous users, looping 10 times, with a ramp period of 1 second, thus a total of 500 requests depicted as 500 samples in below graph in 1sec.



No of Samples 500
Deviation 118
Latest Sample 15
Throughput 10,913.059/minute
Average 170
Median 152

Figure 6.1 Screenshot of graph results of test case 1 in jmeter for 500 samples.

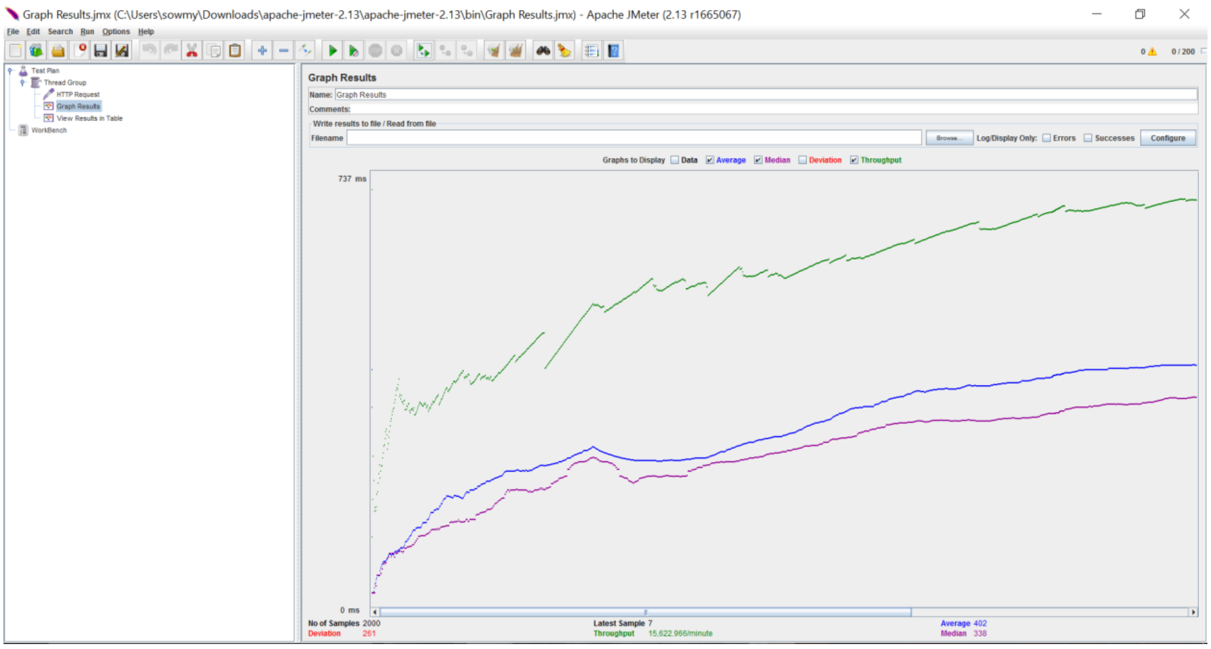
Generating the above test plan for 100 users and loop count 10, thus 1000 samples:



No of Samples 1000
Deviation 261
Latest Sample 35
Throughput 11,021.308/minute
Average 396
Median 381

Figure 6.2 Screenshot of graph results of test case 1 in jmeter for 1000 samples.

Generating the above test plan for 200 users and loop count 10, thus 2000 samples:



No of Samples 2000
Deviation 261
Latest Sample 7
Throughput 15,622.966/minute
Average 402
Median 338

Figure 6.3 Screenshot of graph results of test case 1in jmeter for 2000 samples.

	Users	Ramp up	Loop count	Samples	Avg Response time	Throughput
1	50	1	10	500	170	10913
2	100	1	10	1000	390	11921
3	200	1	10	2000	402	15322

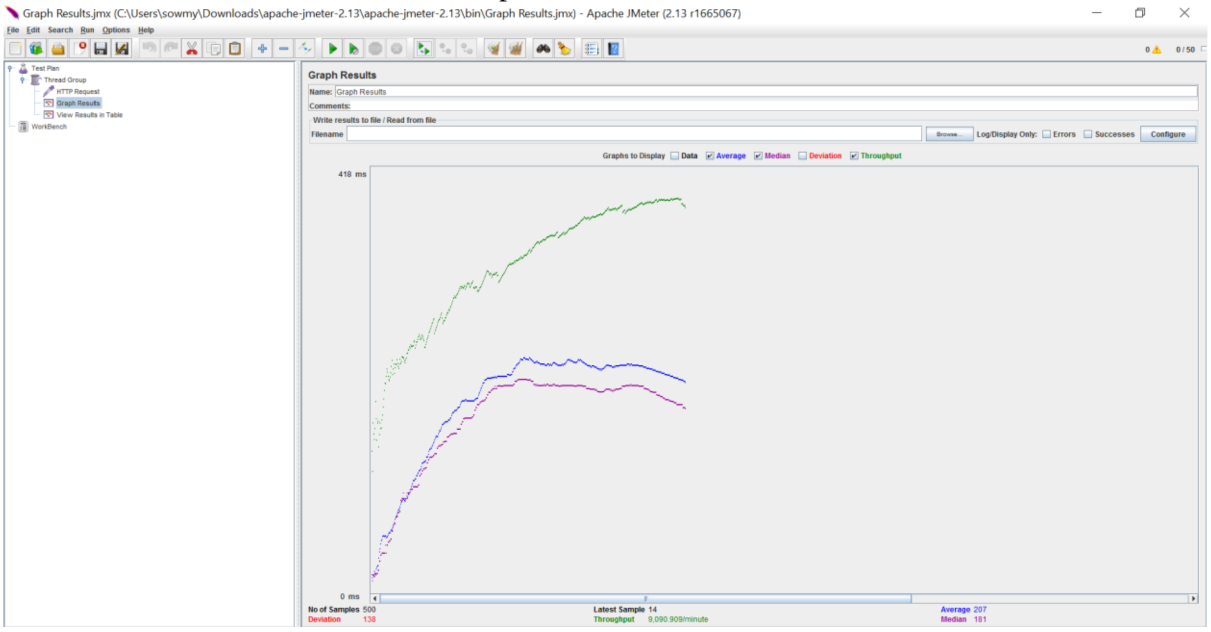
Table 6.3 Summary of results for test case 1

A graph is plotted with users and their response times for main page. The average response time of the application for 2000 samples accessing application page at a time is 402ms and average throughput is around 12000/minute but when samples are above 2000 then throughput is decreasing that means the application works good for up to 2000 requests.

Test 2: JMeter test for Login page:

In order to test the login page where recipes are displayed, this test uses a total of 500 threads, simulating 500 simultaneous users, looping single time, with a ramp period of 1 sec, with a total of 500 requests.

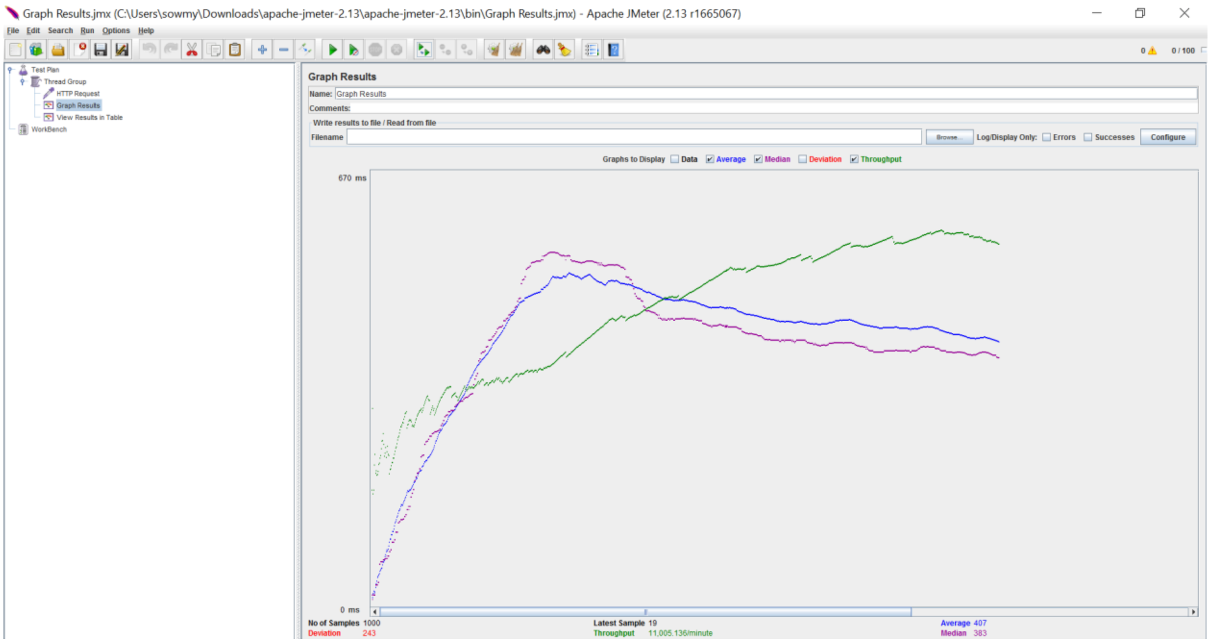
Below are the results of the above test plan.


Graph Results.jmx (C:\Users\sowmy\Downloads\apache-jmeter-2.13\apache-jmeter-2.13\bin\Graph Results.jmx) - Apache JMeter (2.13 r1665067)
File Edit Search Run Options Help
Test Plan
Thread Group
HTTP Request
Graph Results
View Results in Table
WorkBench
Graph Results
Name: Graph Results
Comments:
Write results to file / Read from file
Filename
Browse...
Log/Display Only: Errors Successes
Configure
Graphs to Display Data Average Median Deviation Throughput
418 ms
0 ms

No of Samples 500
Deviation 138
Latest Sample 14
Throughput 9,090.909/minute
Average 207
Median 181

Figure 6.4 Screenshot of graph results of test case2 in jmeter for 500 samples.

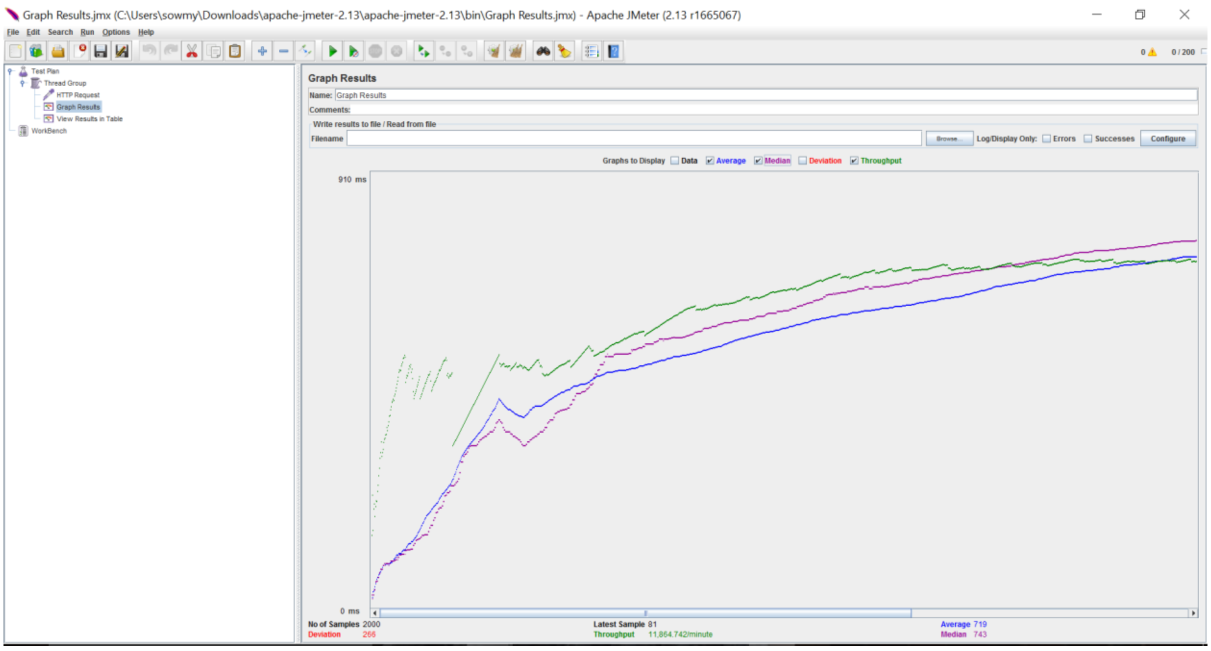
Generating the above test plan for 5000 users:


Graph Results.jmx (C:\Users\sowmy\Downloads\apache-jmeter-2.13\apache-jmeter-2.13\bin\Graph Results.jmx) - Apache JMeter (2.13 r1665067)
File Edit Search Run Options Help
Test Plan
Thread Group
HTTP Request
Graph Results
View Results in Table
WorkBench
Graph Results
Name: Graph Results
Comments:
Write results to file / Read from file
Filename
Browse...
Log/Display Only: Errors Successes
Configure
Graphs to Display Data Average Median Deviation Throughput
670 ms
0 ms

No of Samples 1000
Deviation 243
Latest Sample 19
Throughput 11,005.136/minute
Average 407
Median 383

Figure 6.5 Screenshot of graph results of test case 2 in jmeter for 1000 samples.

Generating the above test plan for 10000 users:


Graph Results.jmx (C:\Users\sowmy\Downloads\apache-jmeter-2.13\apache-jmeter-2.13\bin\Graph Results.jmx) - Apache JMeter (2.13 r1665067)
File Edit Search Run Options Help
Test Plan
Thread Group
HTTP Request
Graph Results
View Results in Table
WorkBench
Graph Results
Name: Graph Results
Comments:
Write results to file / Read from file
Filename
Browse...
Log/Display Only: Errors Successes
Configure
Graphs to Display Data Average Median Deviation Throughput
910 ms
0 ms

No of Samples 2000
Deviation 266
Latest Sample 81
Throughput 11,864.742/minute
Average 719
Median 743

Figure 6.6 Screenshot of graph results of test case 2 in jmeter for 2000 samples.

	Users	Ramp up	Loop count	Samples	Avg Response time	Throughput
1	50	1	10	500	203	9200
2	100	1	10	1000	407	11005
3	200	1	10	2000	719	11854

Table 6.4 Summary of results for test case 2

A graph is plotted with users and their response times for main page. The average response time of the application for 2000 samples accessing application page at a time is 719ms and average throughput is around 11000/minute but when samples are above 2000 then throughput is decreasing that means the application works good for up to 2000 requests.

6.4 Screenshots of application

Screenshots of different web pages in application are provided below.

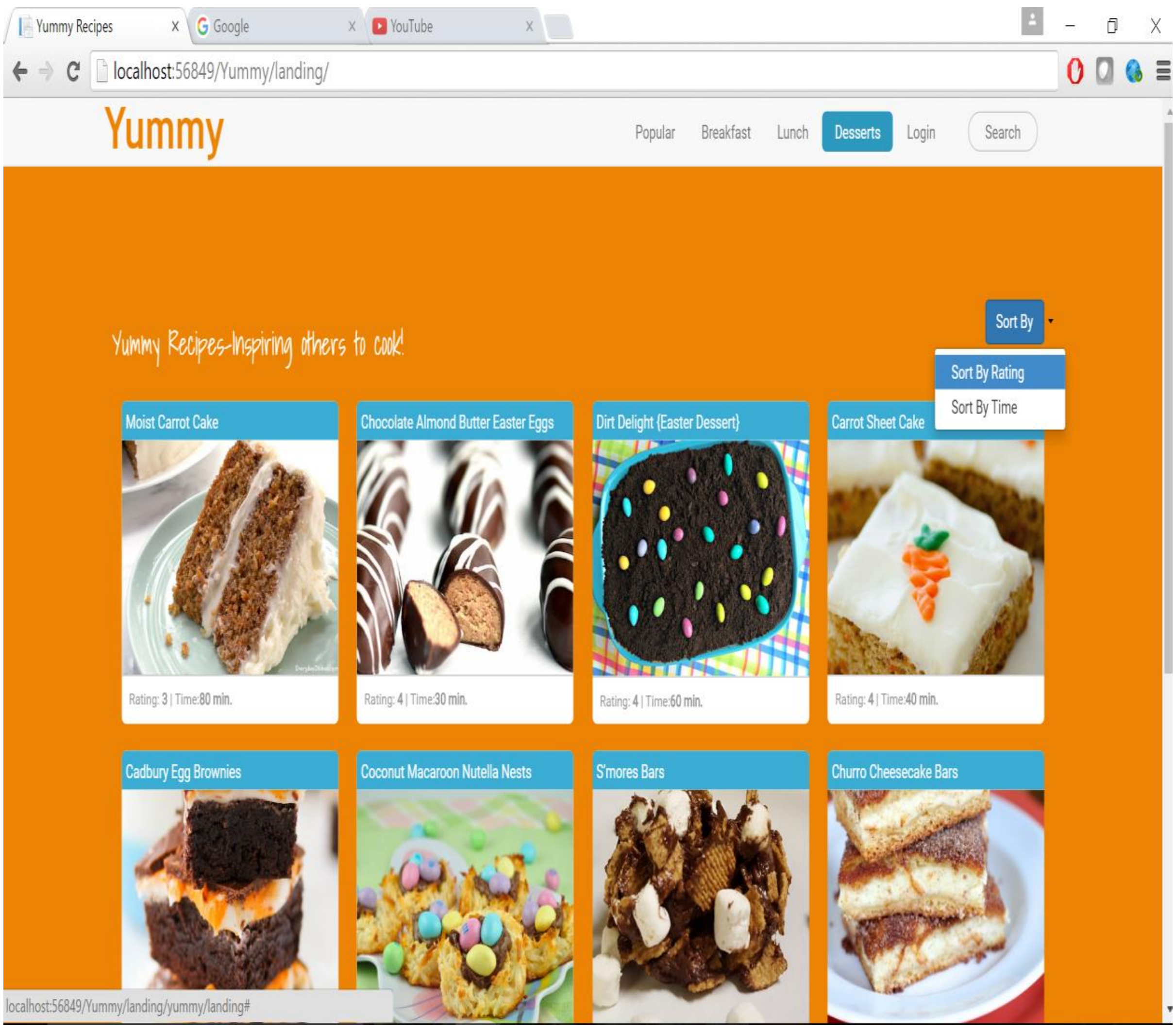


Figure 6.7 Screenshot of main page

The above screenshot is of the main page of application. The different tabs available at the top of webpage underlined by red line are the direct search filters for example when user clicks on desserts tab, recipes only related to desserts are displayed. In this screenshot no user is logged in so ‘Like’ button to like recipes is hidden. Dropdown menu with two sorting options is provided to sort the display order of recipes.

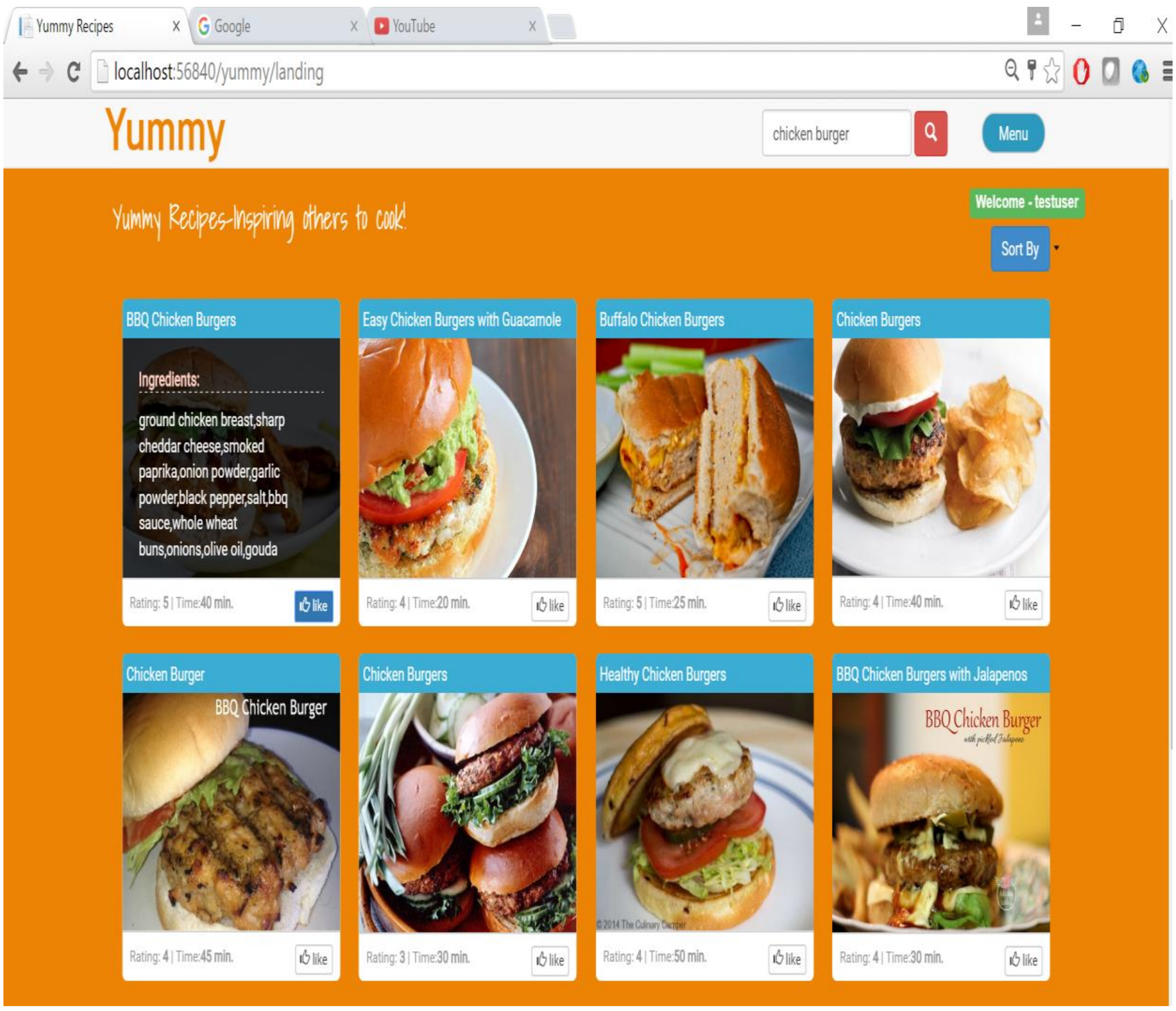


Figure 6.8 Screenshot of main page when user logged in

The above screenshot shows user has searched for ‘chicken burger’ and results are displayed. The “testuser” has logged into application hence like button is enabled and user can like a recipe and add it to his favorites which can be accessed by clicking on favorites tab.

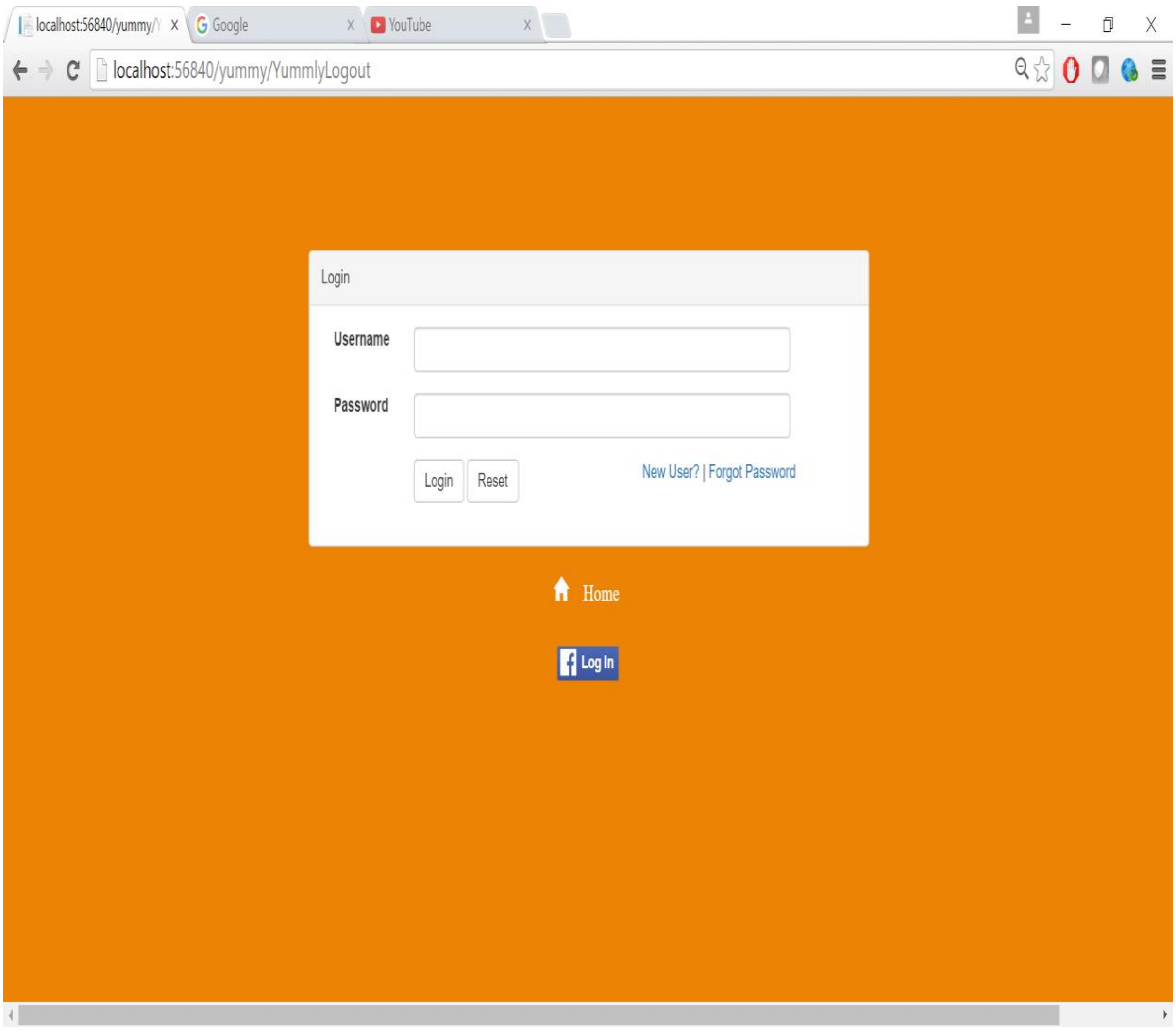
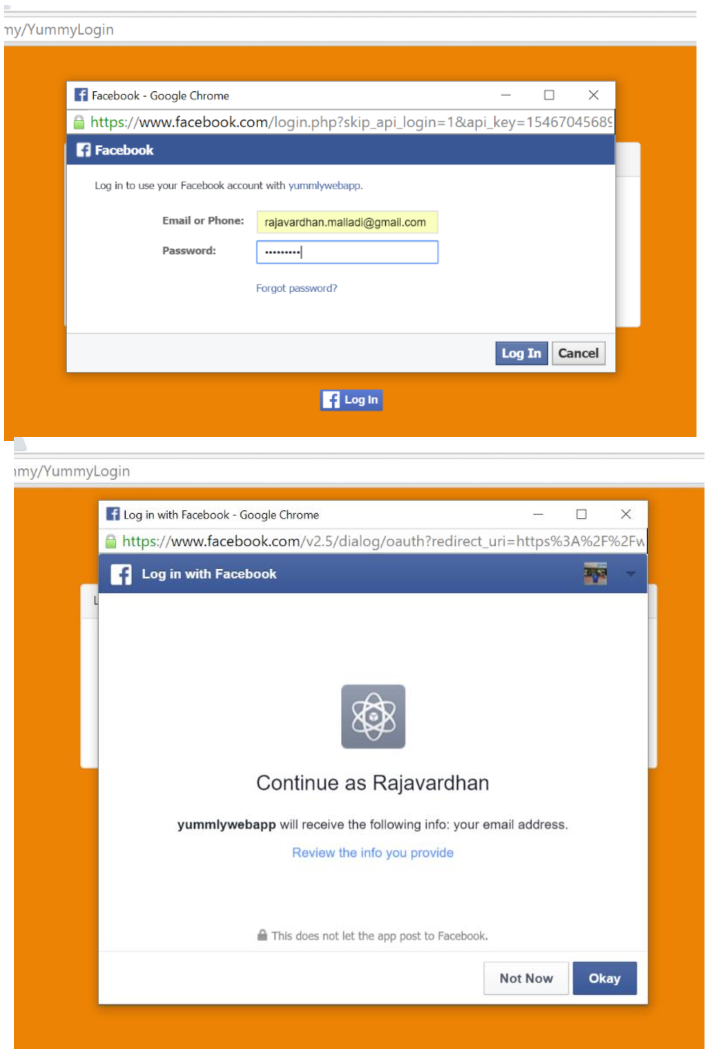


Figure 6.9 Screenshot of Login page

The login page of application shows two options for user to login, either use applications credentials created at registration time or use Facebook login credentials. If user has not registered yet he can click on “new user?” button provide asked details and register.


my/YummyLogin

Facebook - Google Chrome

https://www.facebook.com/login.php?skip_api_login=1&api_key=15467045689

Facebook

Log in to use your Facebook account with yummlywebapp.

Email or Phone: rajavardhan.malladi@gmail.com

Password: ••••••••

Forgot password?

my/YummyLogin

Log in with Facebook - Google Chrome

https://www.facebook.com/v2.5/dialog/oauth?redirect_uri=https%3A%2F%2Fw

Log in with Facebook

Continue as Rajavardhan

yummlywebapp will receive the following info: your email address.

Review the info you provide

This does not let the app post to Facebook.

Figure 6.10 Screenshot of Facebook login page

When user clicks on “Facebook login” button, application will access Facebook login form and display it to the user. The above screenshot is taken after user has entered Facebook credentials and Facebook is reviewing the user’s permission to allow access his details for this application. “yummlywebapp” is the app name created by me in Facebook.

Chapter 7 - Project Metrics and Experience

This chapter presents the project metrics showing the number of hours spent completing each phase of the project. It also summarizes the challenges I came across and how I tackled them and my overall learning experience during the entire lifecycle of the project.

7.1 Project Metrics

Project Metrics are the indicators that track the ongoing project progress. The Project Metrics discussed in this document are source lines of code and the amount of time spent during the entire project span. Table 1 and Table 2 represent the project metrics: source lines of code and project phases and their duration respectively.

Below table gives details amount of lines of code written in different languages. Auto generated code is excluded from the count.

ASP.NET & C# Server Side code	810
jQuery, html, css	1784

Table 7.1 Project Lines of Code

Learning Project Technologies	4 weeks
Requirement Gathering and Design	1 week
Implementation	4 weeks
Testing	1 week
Documentation	2 weeks

Table 7.2 Project Planning Phase

7.2 Experience

Overall development process is a great learning process and here I present some challenges and learnings from it.

- The biggest challenge is learning and working with many new technologies such as Entity framework, jQuery, Bootstrap which I'm not familiar with before this project.
- I have experience of working with ASP.NET web forms but with this project I learnt the implementation of MVC architecture using ASP.NET MVC. Creation of dynamic webpages was much easier and effective following MVC.
- One of the important learning is consumption of web services created by another applications on the web. The ability of connecting to web applications and pulling data from their servers to our servers by using RESTful API calls was really exciting thing to learn.
- Designing a rich looking and interactive user interface. Applying some cool transition in the user interface. Displaying the recipe details in a grid format.
- During the development of application I came across an interesting challenge. When creating user's favorite collection, application has to store recipe details in database. Initially I stored recipe name in database so that whenever users tries to access their favorites collection, application will make get request calls to web API for each recipe. Disadvantage with this approach is application has to frequently call the API which increases the traffic making a delay in loading the webpage which is not we want. Later on I came up with another approach where I converted the JSON object of recipe data into a string and stored in database. So when user tries to access favorite collection, application will communicate with the local database and loads the data to user. The later approach is far efficient than earlier as it doesn't create any unnecessary traffic.

  This project gave me an opportunity to learn a lot of new concepts in programming and web development which would help me throughout my career. The overall experience was like crash course, learning many new things and implementing them on the go.

Chapter 8 - Conclusion and Future Work

8.1 Conclusion

This Recipe Search Engine web application is developed to work as a central information hub for the kitchen—connecting consumers with recipe ideas, ingredient lists, and cooking instructions. The requirements are collected by thinking what all features the user expects to present in a food blog like application and also discussing ideas with my friends. ASP.NET MVC is used to design web pages and implement MVC architecture. Bootstrap, jQuery are used for creating interactive user interface. RESTful web services is used to communicate with the Yummly web API and pull data from it. SQL server 2014 is used as database engine and entity framework is used to connect application with the database. Functionality wise in this application, users can search for their favorite recipes. They can filter and sort the display order of search results. By liking the recipes they can create their own favorite collection which will be stored in their account and also they can view and modify it anytime.
The performance of this application is evaluated by rigorously testing it against various test scenarios. Efficiency and correctness of the application is evaluated with the help of various test cases.

8.2 Future Work

This project is developed as a master's project and still gives lot of scope for its extension if it is going to be developed as commercial product. Below discussed are some of the features that can be added to the application.

- More filter options can be provided for the users to filter the search results such as filtering the results according to cuisine type, ingredients type.
- More sorting options can also be added like sorting results according to nutritional elements in recipe.
- Search options according to taste type can also be added, for example search recipes only with more salty taste.
- Users can share their favorite collection to others users on social networking platforms.
- Users can add recipes to the web application.
- Adding admin side module to validate the recipes added by users.
- Providing personalized recommendations to users by tracking their search history and liked recipes.

References

[1] Yummly API documentation. (03/15/2016)
https://developer.yummly.com/documentation
[2] ASP.NET MVC 5 Tutorial. (03/01/2016)
http://www.asp.net/mvc/overview/getting-started/introduction/getting-started
[3] Facebook SDK for JavaScript (04/05/2016)
https://developers.facebook.com/docs/javascript
[4] Apache Jmeter and load testing tutorial (04/10/2016)
http://jmeter.apache.org/
https://www.youtube.com/watch?v=4mfFSrxpl0Y
[5] Introduction to Web services(04/12/2016)
https://docs.oracle.com/cd/E19798-01/821-1841/gijti/index.html
[6] RESTful web services design guidelines(04/12/2016)
https://docs.oracle.com/cd/E19226-01/820-7627/gijti/index.html
[7] Entity framework (03/10/2016)
http://www.entityframeworktutorial.net/what-is-entityframework.aspx
[8] OAuthentication(04/13/2016)
http://oauth.net/2/