ITCS 6122

SOFTWARE SYSTEM DESIGN AND IMPLEMENTATION

PROJECT REPORT

ON

A RESTAURANT DISH RATING SYSTEM

TEAM MEMBERS

Ajitesh Janaswamy (800987259)

Devireddy Venkata Sai Krishna Reddy (800975328)

Lalit Sandeep Bhavinen (800987686)

Suvajit Chakrabarty (800976080)

Vijayalakshmi Muniyandi (800981131)

CONTENTS

ABSTRACT

We have built a web application that allows users to know information regarding a specific dish in a particular restaurant. Users will be able to search for dishes and know how good a specific dish in a particular restaurant is and also other relevant information of the dish like cuisine and category. A restaurant will be able to add dishes present in their menu and also specific dish information to this application catalog. These dishes can then later be rated and reviewed by foodies. There is a admin user in the backend who can grant specific privileges to specific users and restaurants.

The project makes use of technologies like HTML, CSS, JavaScript, Bootstrap, jQuery, MySql, Ruby on Rails. The User Interface which has been designed with HTML, CSS, JavaScript, jQuery and some Bootstrap has been integrated into the view framework of the Ruby on Rails application. We have used MySql as our database to store all the information of users, dishes and the ratings/reviews of users. The main objective of this application is to make sure that users choose the best dishes when they visit a restaurant.

INTRODUCTION

It is a very rare situation that someone visits a restaurant and is satisfied with all the dishes that they order in the restaurant. It would be beneficial to anyone visiting a restaurant to know ahead of time which dishes are usually good in that restaurant so that they can make the correct choices when choosing dishes from the menu.

A lot of websites provide information about the restaurant like a restaurant rating or whether a restaurant is good or not. We believe that this information only helps users in choosing a restaurant but does not go onto help him further when he is already in a restaurant and has to choose dishes from the menu so that he can order a good dish that he will be satisfied with. This is the problem that we want to solve and so we have created a web application which provides information/reviews about the specific dishes available in a restaurant.

There are primarily two types of users, foodie and restaurant. A user type of foodie will be able to know how good the dish in a particular restaurant is, based on ratings and reviews provided by previous users who have already reviewed the dish. Foodies will also be able to know other details of the dish like cuisine and category. Foodies can also search for a specific dish by providing dish information in the search box like dish name or dish cuisine.

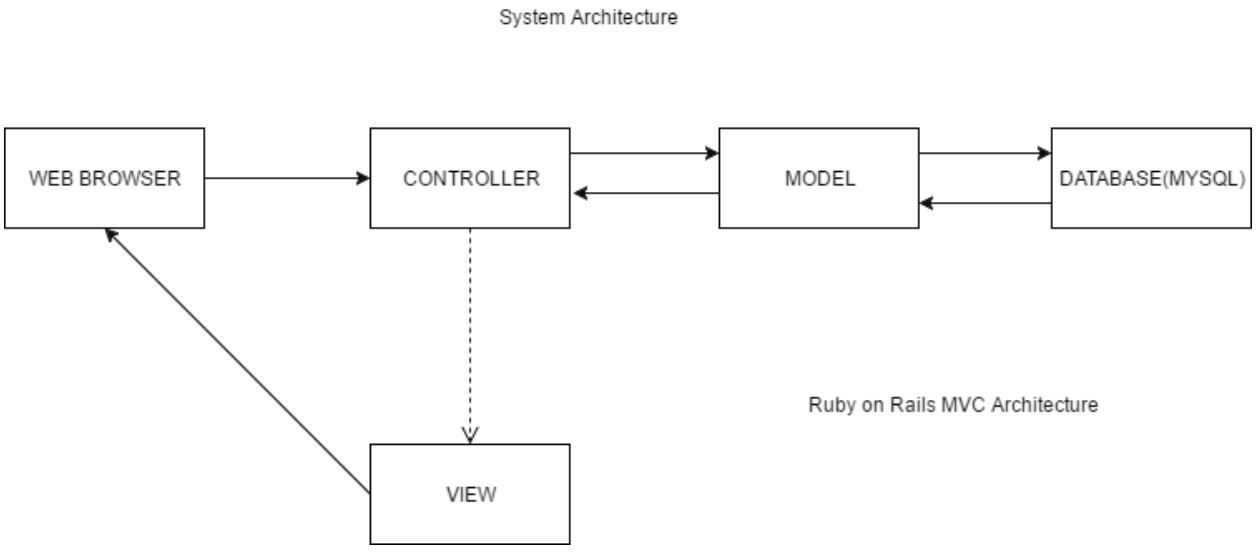
The search results are normally based on average rating, a dish with a higher rating will appear higher in the list. A user type of restaurant will be able to add dishes present in their menu and also specific dish information to this application catalog. These dishes can then later be rated and reviewed by foodies. The web application also provides users a list of trending dishes based on what users are rating and reviewing. There is a admin user in the backend who can grant specific privileges to specific users and restaurants.

One of the best features of our web application is that users have been given the privilege of viewing the ratings/information of a dish without signing in. They will have to create accounts and sign in if they want to rate or add reviews for a dish. Similarly restaurants will have to sign in if they want to add dishes to their catalog.

MATERIAL AND METHODS USED

We have used HTML, CSS, JavaScript, Bootstrap, jQuery, Ruby on Rails and MySQL database to implement our web application. Ruby on Rails is a server-side web application framework written in Ruby. It is a model–view–controller (MVC) framework, providing default structures for a database, web pages and a web service. The User Interface which has been designed with HTML, CSS, JavaScript, jQuery and some Bootstrap has been integrated into the view framework of the Ruby on Rails application. We have used MySql as our database to store all the information of users, dishes and the ratings/reviews of users.

The system architecture(MVC architecture) of a Ruby on Rails application is as follows:



Material Used:

- Ruby on Rails
- HTML
- CSS
- JavaScript
- Bootstrap
- jQuery
- MySql

Methods Used:

- Ruby on Rails has been used for building the functionality of the web application and integrate with the front end and the database.
- HTML has been used for designing the content of web-pages.
- CSS has been used for styling of the web pages.
- JavaScript has been used for controlling the behavior of the web pages.
- We have made use of Bootstrap, which includes the HTML, CSS based templates and also for making the website responsive.
- We have also made use of jQuery, which is a JavaScript library mainly to provide a star based rating system available to users.
- We have MySQL database to store all the information of users, dishes and the ratings/reviews of users.

Software methodology used: Agile

Agile is a software development methodology used to build a software incrementally using short iterations of 1 to 4 weeks so that the development is aligned with the changing business needs. It is a process which values individuals and interactions over processes and tools.

Agile methodology has two important features namely User stories and iterations. User stories provide simpler ways of capturing user requirements in a project instead of having a lengthy documentation model. In the iteration period, development of user stories happen.

As the number of developers in our team is very small, we tried to get a first hand experience with agile methodologies instead of using traditional software development models like waterfall and spiral models. We got a better opportunity in understanding the different aspects of Agile methodologies.

We created a number of user stories for development through Agile approach. We worked on each of these stories considering one at a time. Some of the user stories are as follows- A customer is able to login/signup and search for various cuisines and trending dishes in

a city. Each restaurant can add a new dish, provide description and delete a dish. A customer can rate and review a dish.

Smaller teams always have a problem when the project is heavily dependent on documentation. Agile methodology helps to overcome this problem by providing emphasis on user stories.

Project management tools used:

- Basecamp to manage the project and assign tasks.
- Slack was used to communicate with team mates.

SOFTWARE, HARDWARE AND TECHNICAL REQUIREMENTS

Software Requirements:

- Atom/Sublime text
- Bootstrap
- MySql
- Ruby
- Ruby on Rails
- jQuery

Hardware Requirements:

- RAM: 4 GB
- Processor: i5
- 256 GB Hard Disk

Hardware Requirements to view the web application: Any user with a computer or a smartphone can view the web application.

Technical Documentation:

A) **Programming Languages:**
    - HTML
    - CSS
    - JavaScript
    - BootStrap
    - jQuery
    - Ruby on Rails

B) Reused Algorithms and Programs:

- We have used 'Devise' ruby gem(library) to manage user login authentication. This gem takes care of user sessions and allows multiple models to login at the same time.
- We have used 'Paperclip' ruby gem to act as an image processor and an easy image upload library for the images being uploaded for each dish.
- We have used 'Searchkick' ruby gem to implement the search system. It can return correct search results even if the user makes a spelling mistake when searching.

C) Tools And Environments:

The source code for the complete Ruby on Rails application has been implemented in Atom text editor. The Ruby on Rails application is executed on inbuilt Ruby on Rails server called **Puma**.

To set up the appropriate environment for building a Ruby on Rails application, few things have to be installed and configured.

- Homebrew is a tool to install dependencies of ruby libraries from the command line.
- Bundler is a tool to make sure all ruby gems(libraries) have the correct versions installed on the system. It also installs any missing libraries which the application might need.
- The latest version of ruby needs to be installed.
- The latest version of ruby on rails needs to be installed.
- Mysql database needs to be installed and the correct username and password configuration for the user setting up the database has to be configured with the correct privileges.
- Some libraries like Searchkick require Java to be installed on the system.
- Draw.io has been used to create the UML diagrams for the application.
- Basecamp has been used for project management.
- Slack has been used for project communication between team mates.

D) Database System: We have used MySQL as our database. The name of the database is Savour_development.

E) Text Files:

All files have been placed in a separate path called Text files. This section contains the list of text files and their format.

Functionality for the Ruby on Rails application has been achieved using Ruby files

which are under the controller part of the MVC architecture.

Ruby files

- application_controller.rb
- dishes_controller.rb
- reviews_controller.rb

Files containing associations between tables of the database and the features of the Model part of the of the MVC architecture is achieved using Ruby files.

- application_record.rb
- dish.rb
- foodie.rb
- restaurant.rb
- review.rb
- user.rb

Files containing the front end stack which is part of the View of the of the MVC architecture is achieved using html.erb files. erb stands for "Embedded RuBy". A .html.erb file is HTML with Ruby code embedded in it.

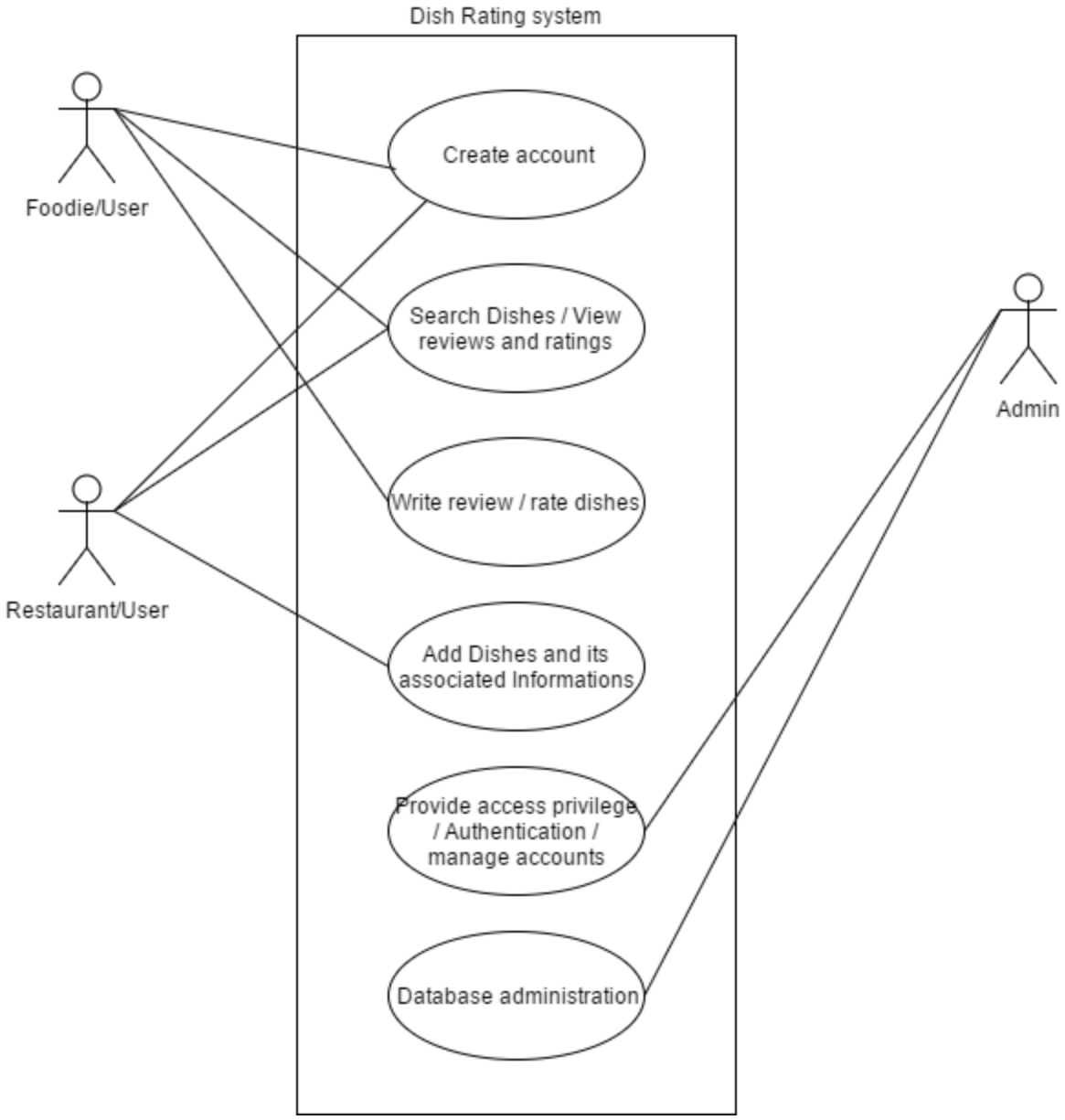
- form.html.erb
- edit.html.erb
- index.html.erb
- new.html.erb
- rate.html.erb
- search.html.erb
- show.html.erb
- application.html.erb

There are also config files for the Ruby on Rails application.

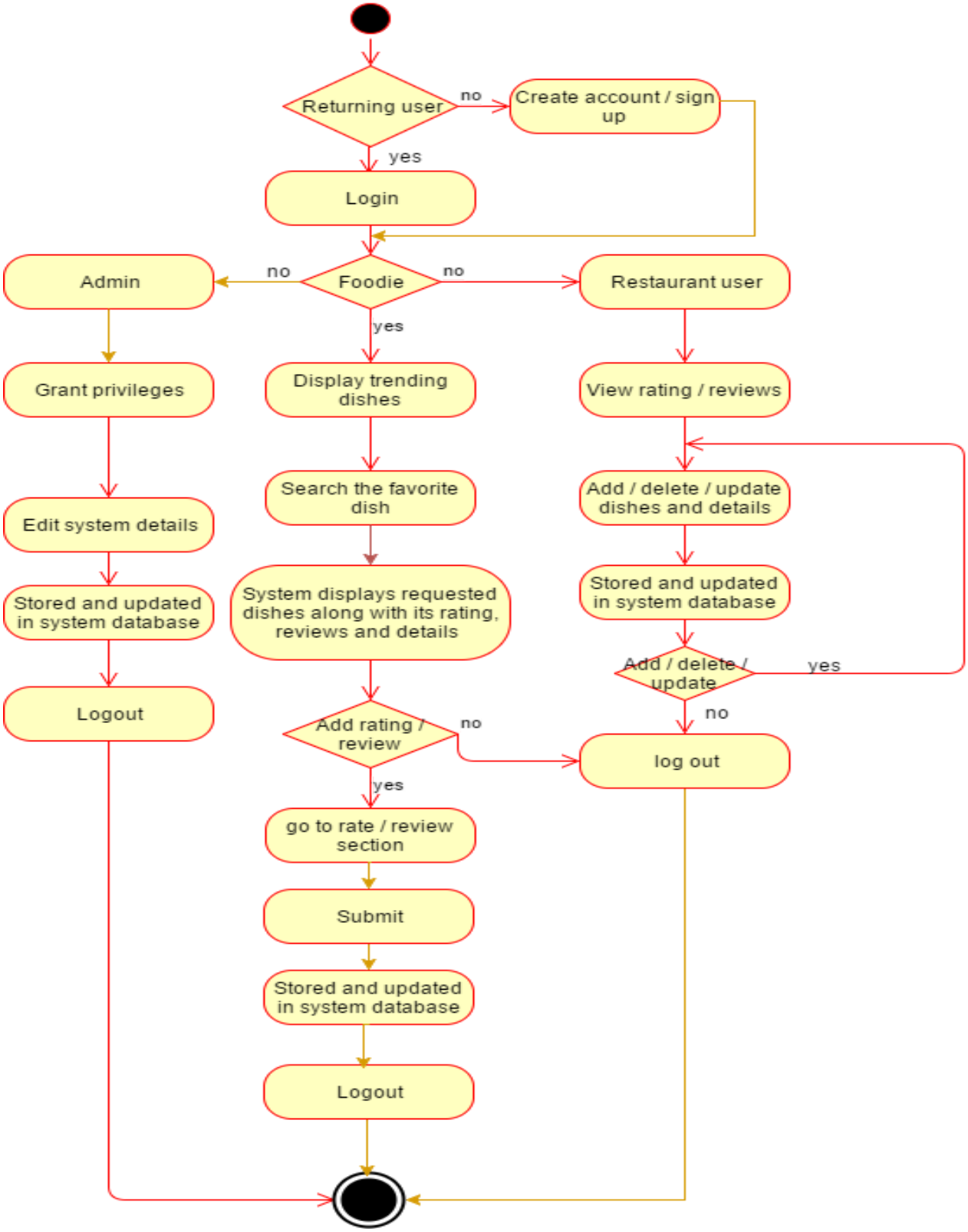
- routes.rb
- database.yml

UML DIAGRAMS:

1. Use Case Diagram

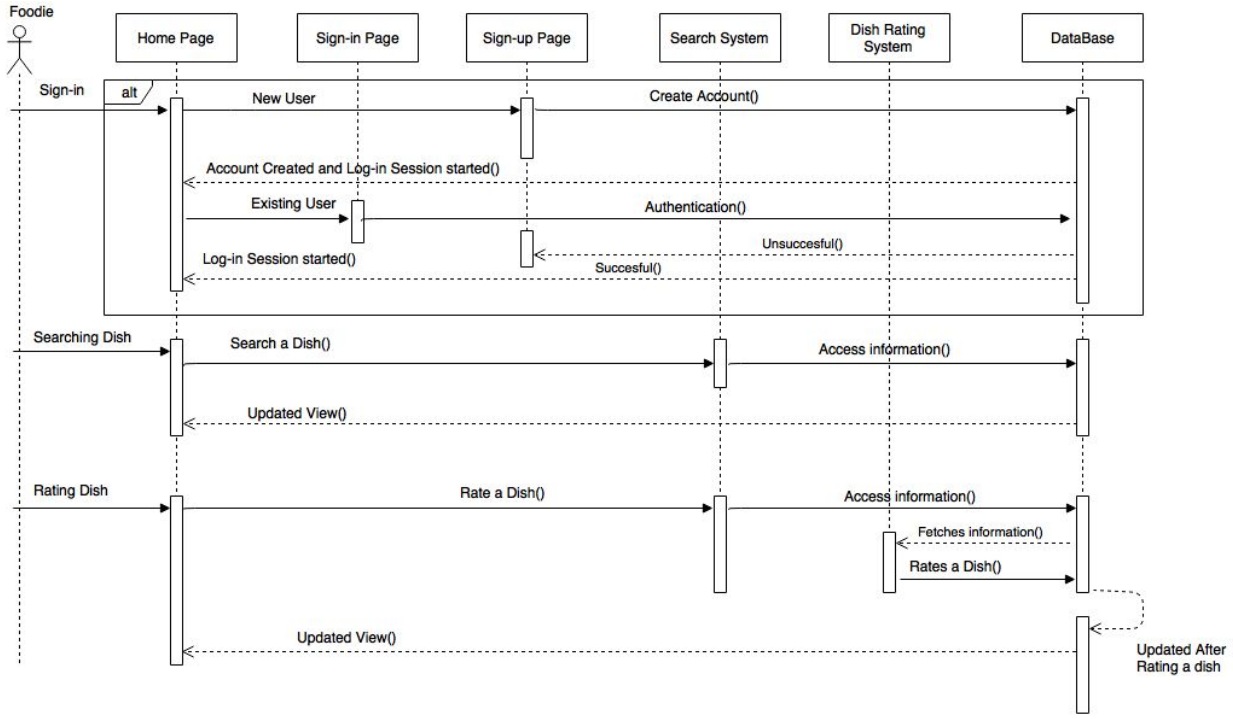


2. Activity Diagram

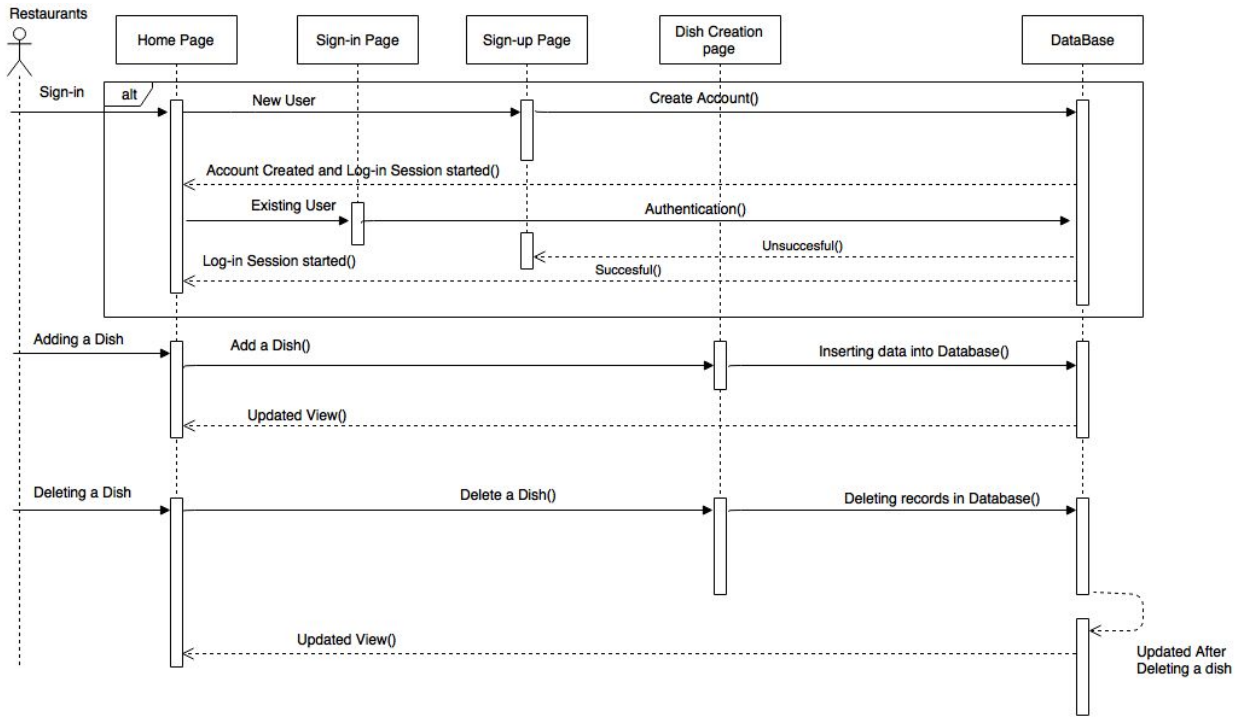


3. Sequence Diagram

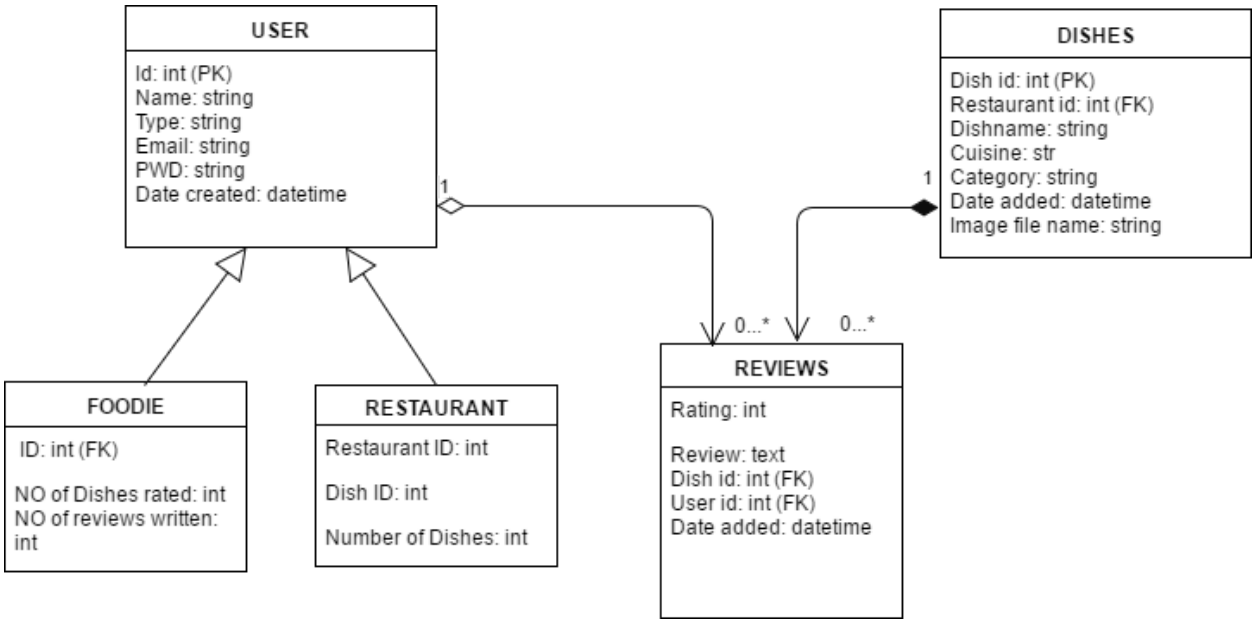
a) Sequence Diagram for a User model of type Foodie



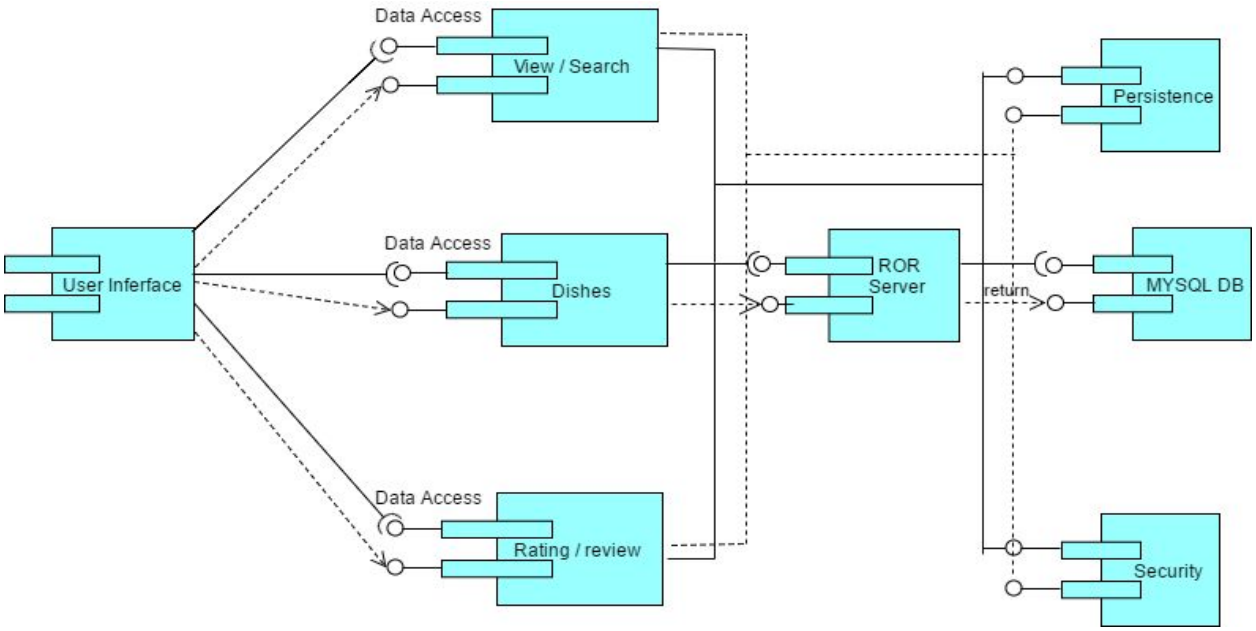
b) Sequence Diagram for a User model of type Restaurant



4 .Class Diagram



5. Component Diagram



CODING

All our source code has been placed in a separate path. We are providing some basic functionality source code methods in this document.

Method to checking whether a user is of a particular foodie or restaurant type.

```
def confirm_user_type(user_type)
   unless current_user.is_a?(user_type)
     sign_out(current_user)
     redirect_to new_user_session_path
   end
end
```

Algorithm to query for list of trending dishes according to their average rating with the higher rated dish appearing earlier.

```
def index
  @dishes = Dish.joins('LEFT JOIN reviews ON dishes.id = reviews.dish_id').
  select("dishes.image_file_name,dishes.id,dishes.dish,dishes.user_id,
  avg(ifnull(reviews.rating,0)) as average_rating, count(reviews.id) as number_of_reviews").
  group("dishes.id").order("average_rating DESC, number_of_reviews DESC")
end
```

Method to Search for dishes

```
def search
  if params[:search].present?
    @dishes = Dish.search(params[:search])
  else
    @dishes = Dish.all
  end
end
```

Method to create a dish

```
def create
  @dish = current_user.dishes.build(dish_params)

  if @dish.save
    redirect_to @dish
  else
    render 'new'
  end
end
```

Method to show reviews and average rating in a specific dish page

```
def show
  @reviews = Review.where(dish_id: @dish.id).order("created_at DESC")

  if @reviews.blank?
    @avg_rating = 0
  else
    @avg_rating = @reviews.average(:rating).round(2)
  end
end
```

Method to update and delete dishes

```
  def update
    if @dish.update(dish_params)
      redirect_to @dish
    else
      render 'edit'
    end
  end

  def destroy
    @dish.destroy
    redirect_to root_path
  end
```

Method to create a new review

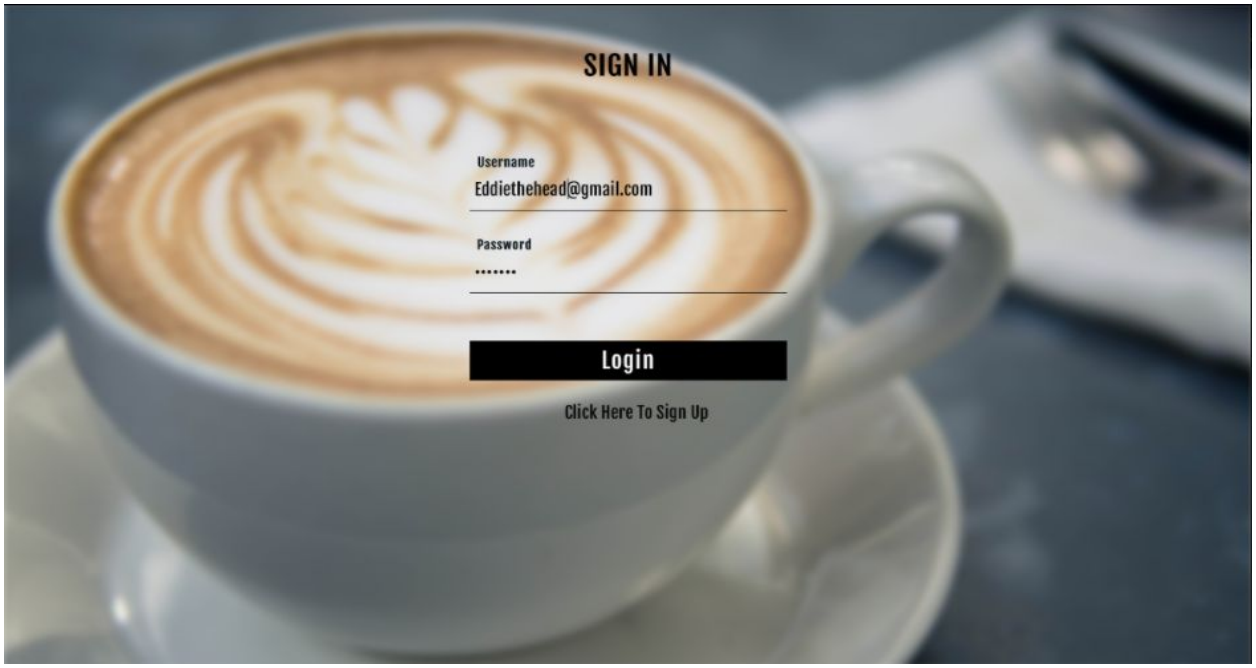
```
def create
  @review = current_user.reviews.build(review_params)
  @review.dish_id = @dish.id

  if @review.save
    redirect_to @dish
  else
    render 'new'
  end
end
```

Method to update and delete reviews.

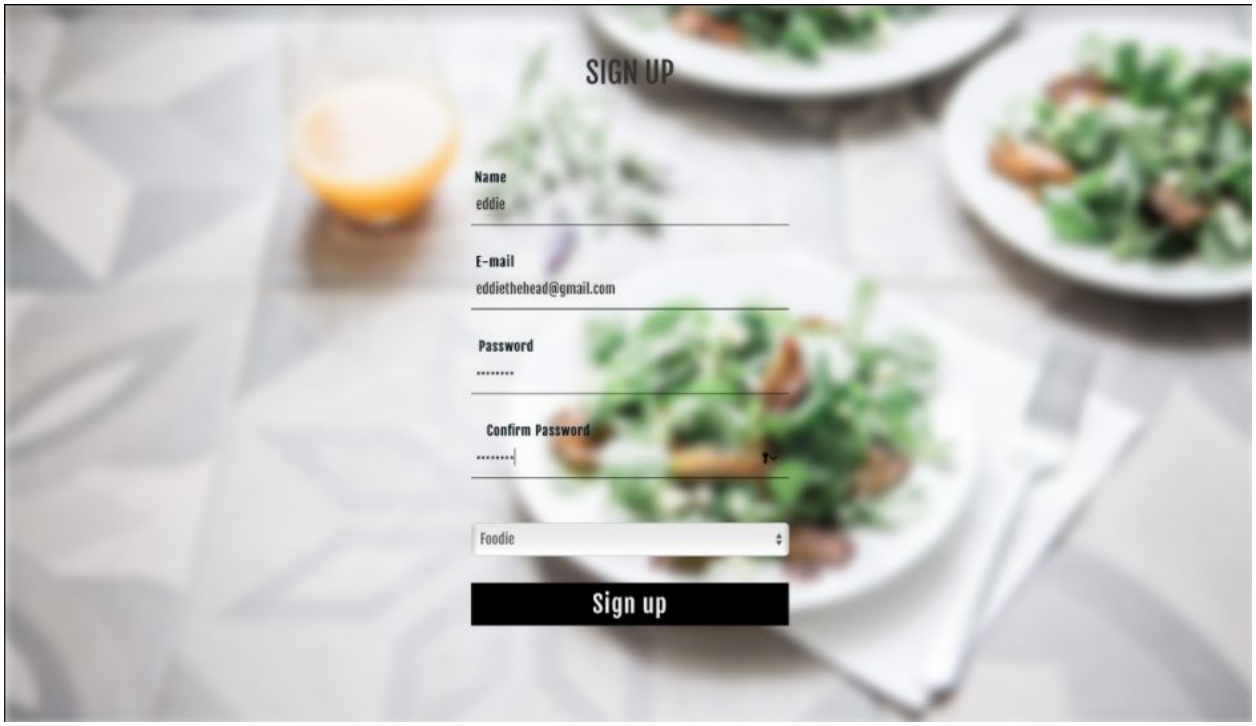
```
def update
  if @review.update(review_params)
    redirect_to @dish
  else
    render 'edit'
  end
end

def destroy
  @review.destroy
  redirect_to @dish
end
```

Sample Results and Output:

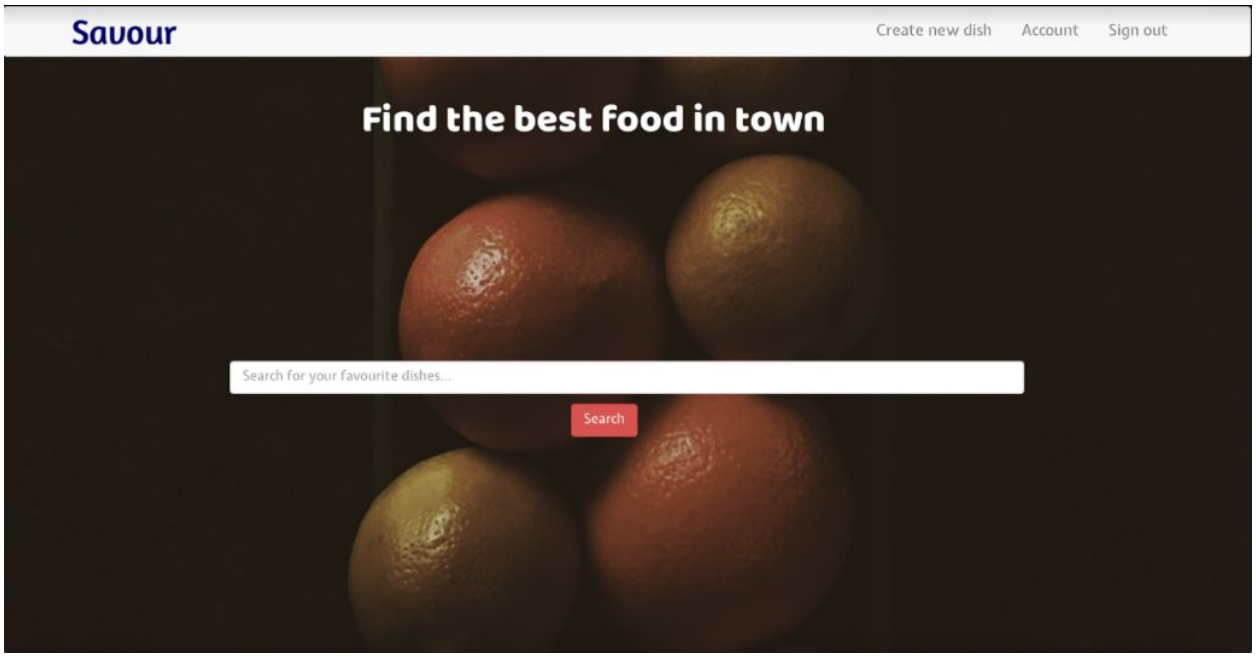
- **Sign-in Page**



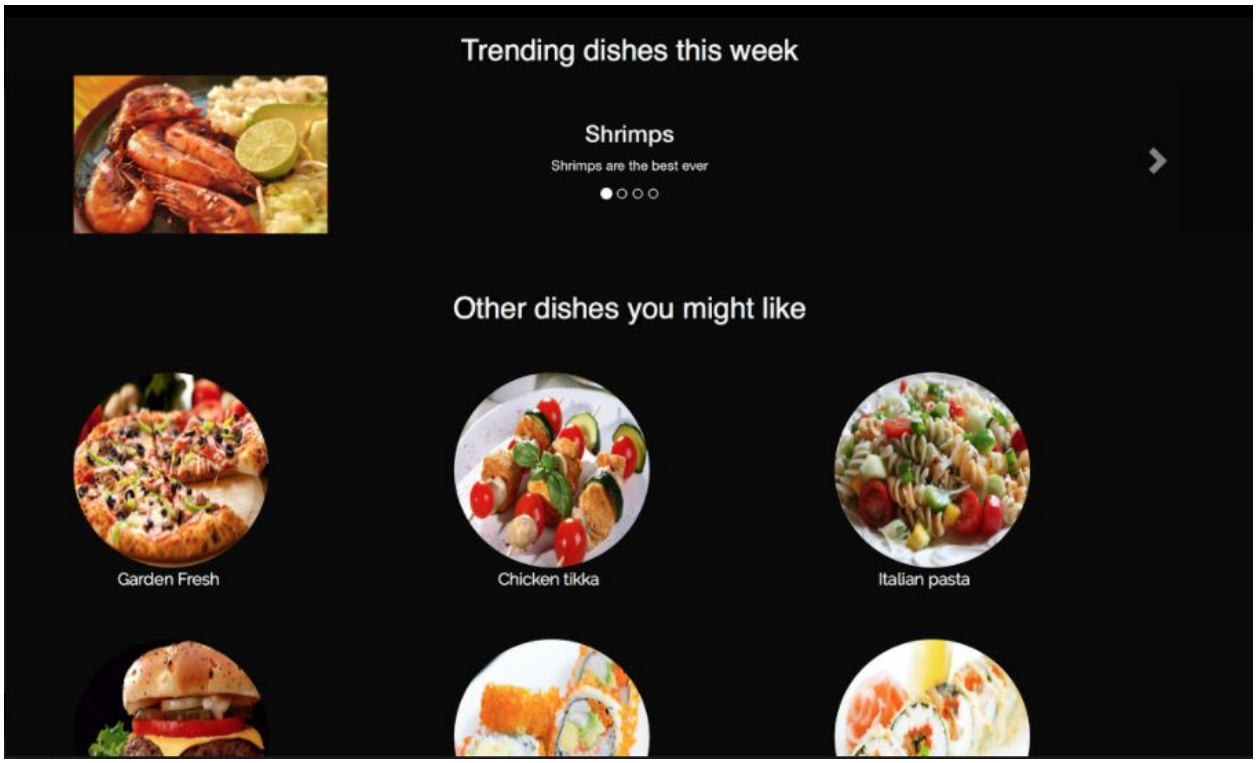
- **Sign-Up Page**



- **Home Page 1**

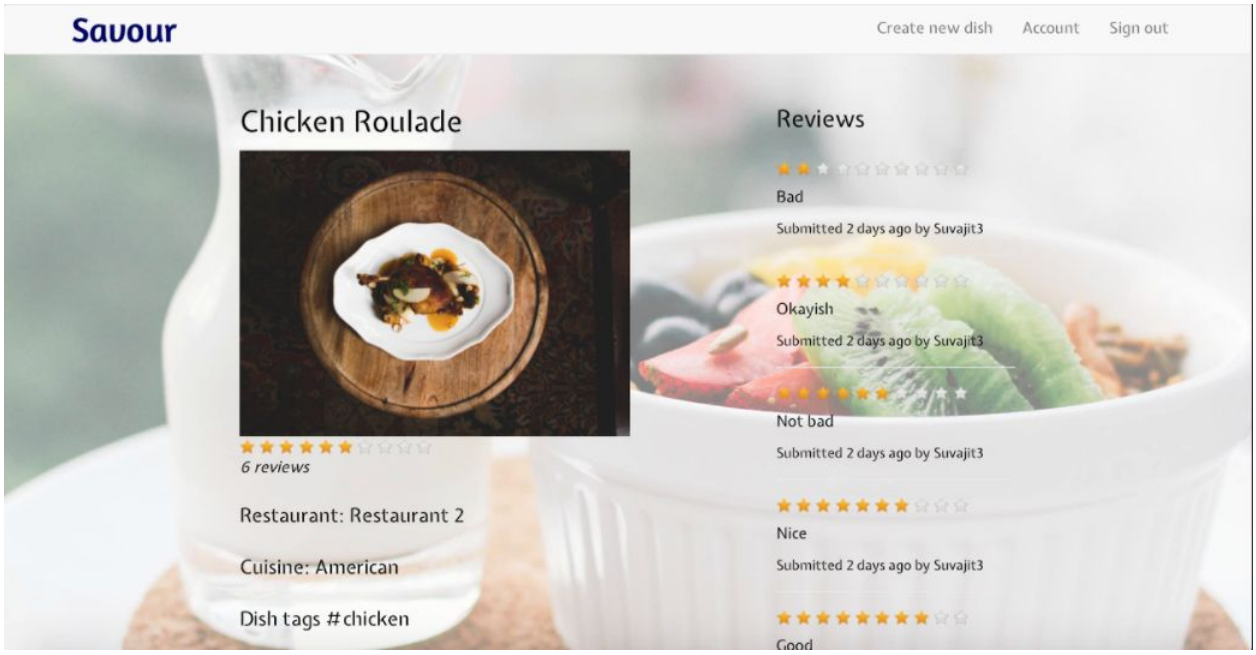


- **Home Page 2**

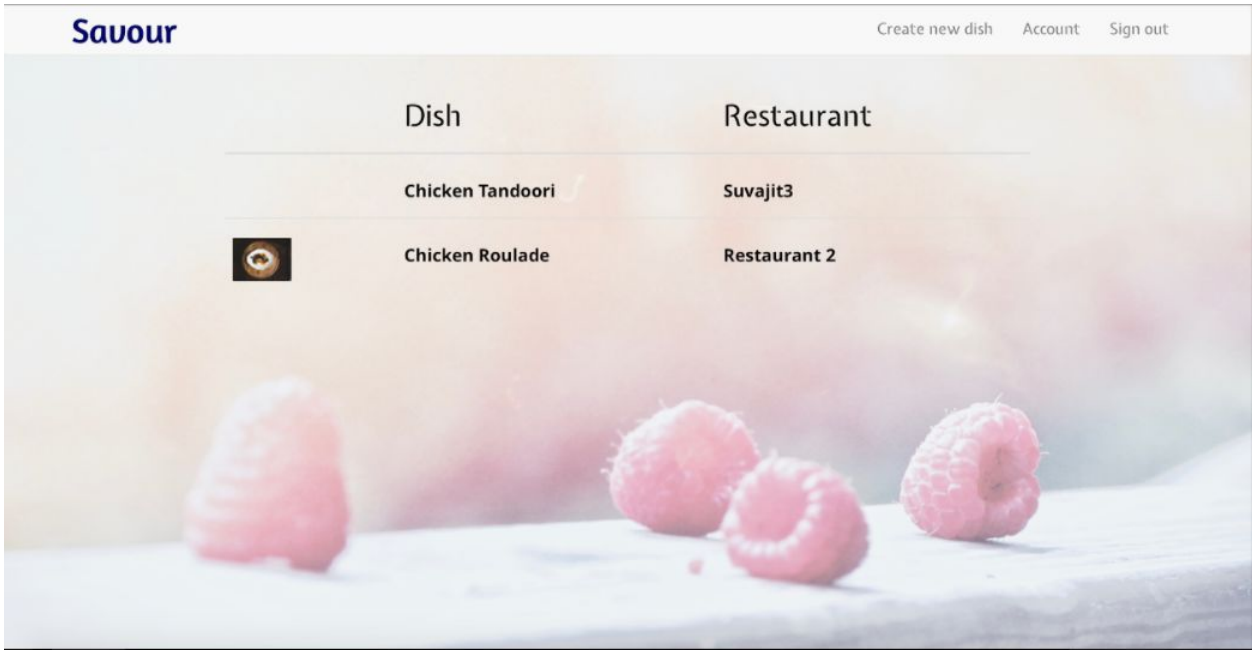


- **Home Page 3**

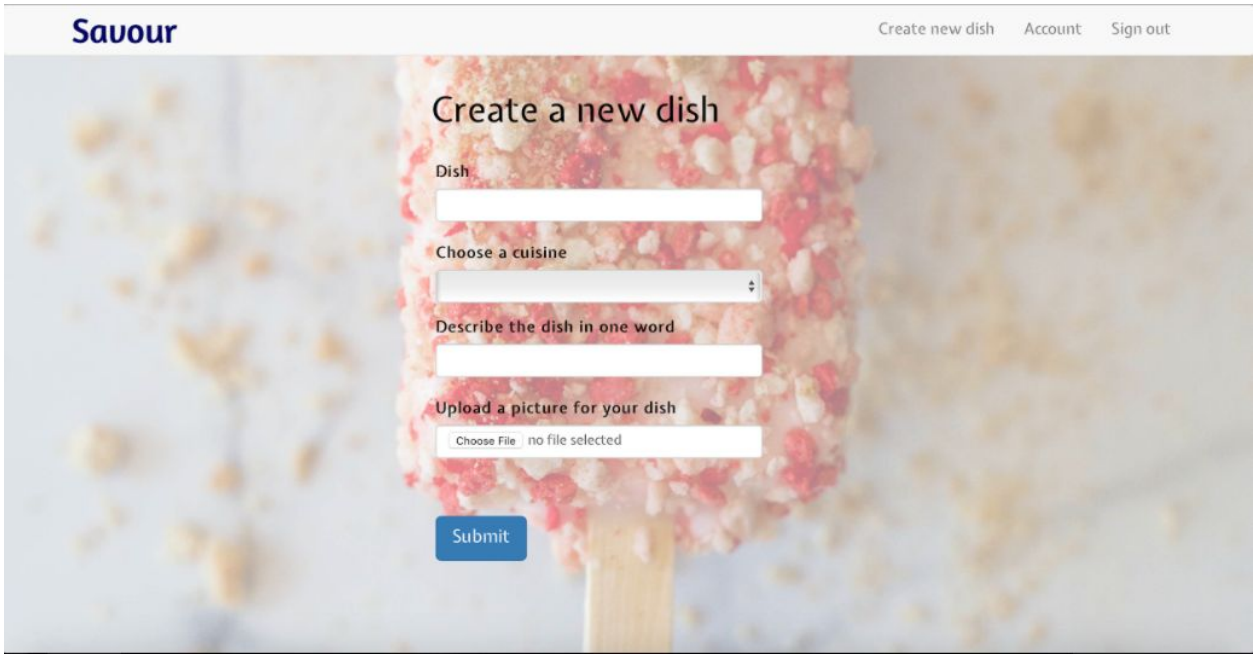
- **Dish Page**



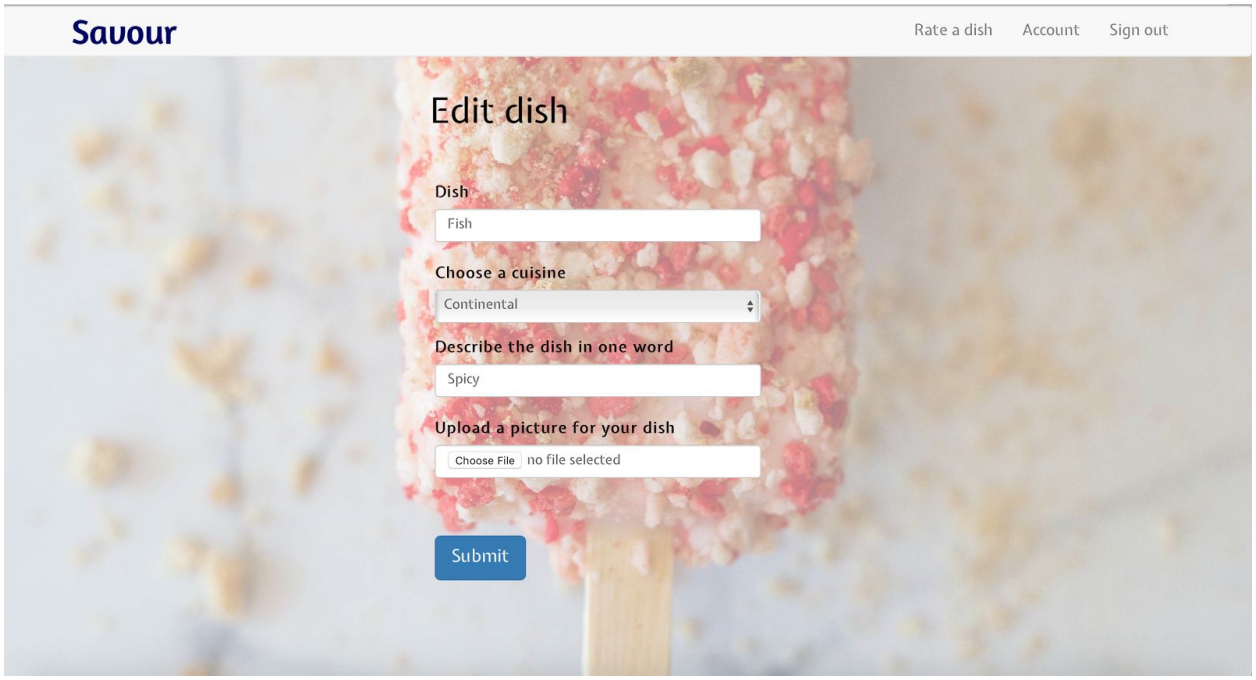
- **Search for a dish Page**



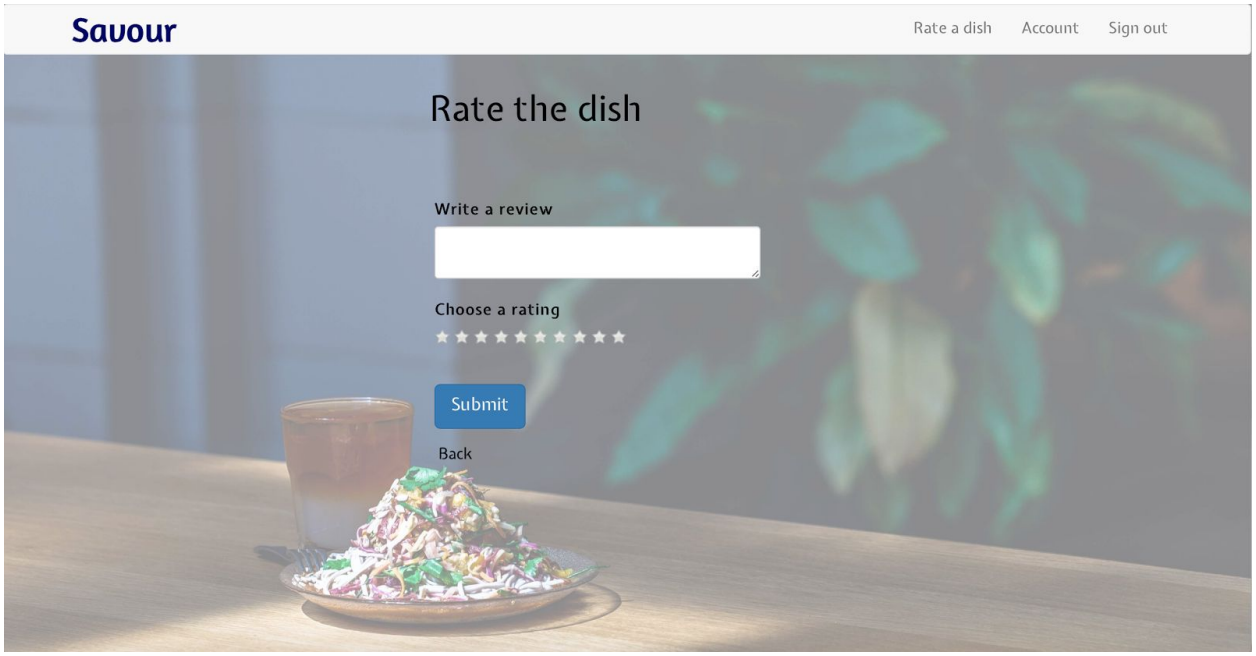
- **Create a new dish**



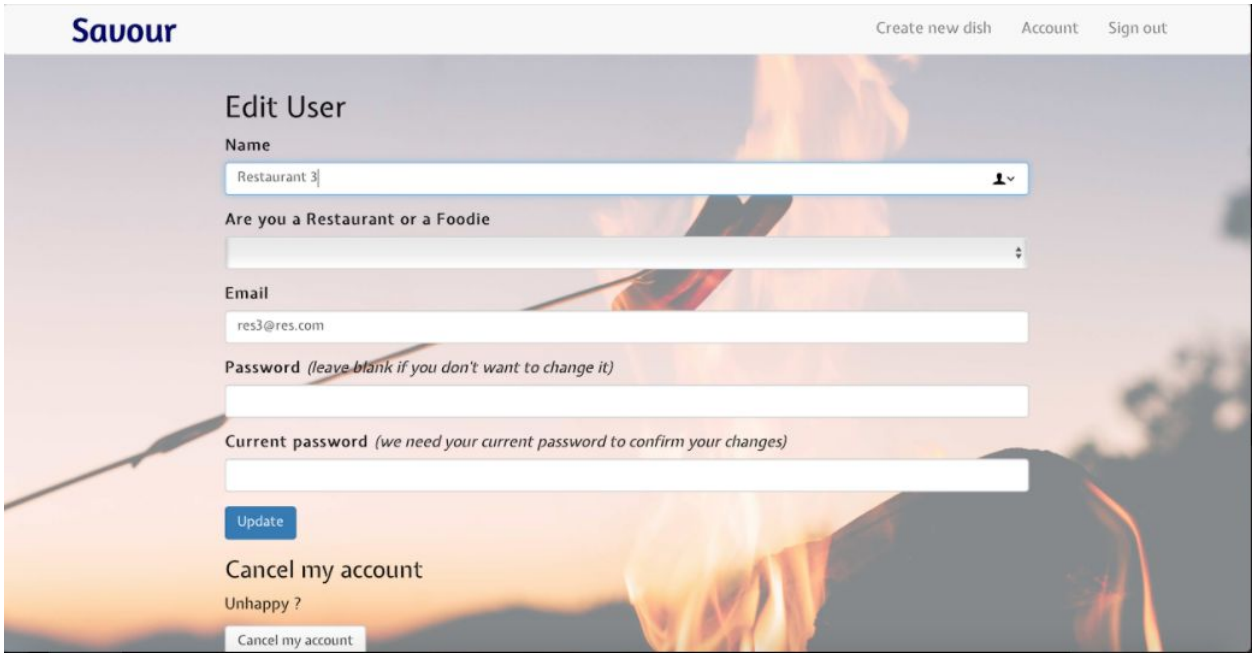
- **Edit dish**



- **Rate the Dish**



- **User Account Page**



CONCLUSION:

This web application allows the users to choose the best dishes in a restaurant. Users can now make a more educated decision about what to have in the restaurant. We guarantee that users will get full value for money when they go to a restaurant from now on.

FUTURE ENHANCEMENTS:

- In the later versions users will be able to make more specific searches for a dish. For example, an user might be wanting to eat a spicy crispy chicken dish. He will be able to make such a specific search for a dish with these adjectives of spicy, crispy and crunchy. The best dishes with these adjectives will appear in the search results.
- Location services will be integrated into the application. A dish will be belonging to a particular restaurant and how far that dish/restaurant is away from him will be informed to him in the application using GPS and Location services.
- Other dish specifics will be provided in the future on the application like how filling the dish is, how popular the dish has been in the last few days, how many people found the dish average rating overrated or underrated.

- Restaurants will be able to add more custom dishes into their menu catalog. They can add a festival specific list of dishes which might only be offered at the restaurant only for a certain amount of time.

REFERENCES:

[1] https://www.w3schools.com

[2] http://stackoverflow.com

[3] http://getbootstrap.com

[4] https://jquery.com

[5] https://thenewboston.com

[6] https://www.tutorialspoint.com

[7] JavaScript & jQuery: Interactive Front-End Web Development Hardcover / Edition 1 By Jon Duckett

[8] Ruby on Rails Tutorial: Learn Web Development with Rails - Book by Michael Hartl

[9] https://unsplash.com - Source for all images used in the application