Food Connect
A social networking web application for food lovers

A Project Report
Presented to
The Faculty of the College of
Engineering

San Jose State University
In Partial Fulfillment
Of the Requirements for the Degree
Master of Science in Software Engineering

By
Deesha Desai, Priyanka Devendran, Preeti Parihar, Ananth Upadhya
December 2021

APPROVED

Andrew Bond, Project Advisor

Dan Harkey, Director, MS Software Engineering

Rod Fatoohi, Department Chair

ABSTRACT

Food Connect – A social networking web application for food lovers

By
Deesha Desai, Priyanka Devendran, Preeti Parihar, Ananth Upadhya

With the increase of technological advancement, consumer review has been playing a very important role to drive businesses. Popular consumer review platforms such as Yelp have been a key player in consumer-driven industries such as the restaurant industry. A social networking platform for connecting users of similar interests would help communicate, share, and track each other activities.

People love eating food outside and would like to share their experiences with their friends. Currently, there aren't any popular platforms that help the user to know about reviews of the restaurants from trusted sources like friends, peers, or colleagues. Also, communication between food lovers about a restaurant on a web platform is missing which makes it hard for the users to identify good restaurants from ordinary ones. Keeping track of the restaurants visited and restaurants to visit is also a primary challenge for food lovers.

Food connect is a geosocial networking web application that enables users to check in the restaurants visited and share the restaurant details with friends. The application is used to discover and explore good restaurants nearby and provide valuable feedback about the restaurant visited. It is a platform that helps in connecting with people who love food and enables users to share their experiences on the restaurants visited, which would help users to identify good restaurants, recommended dishes, and services a restaurant has to offer. It enables the user to keep track of all the restaurants visited and provides a recommended list of good restaurants based on the user's interest.

Acknowledgments

The authors are deeply indebted to Prof. Dan Harkey and Prof. Andrew Bond for their invaluable comments and assistance in the preparation of this study.

Table of Contents

Chapter 1. Project Overview

1.1 Literature Search

The main motive of the project is to build a social networking site for food lovers, so we started researching on this idea and looking for papers and articles which can enhance the ideas.

The first paper we reviewed was "A Postcard from Your Food Journey in the Past: Promoting Self-Reflection on Social Food Posting"[1]. This paper talks about creating a food postcard from the posts made by the user on social networking platforms. User posts on food are a common practice on social media platforms, which provides an opportunity to reflect on one's own eating habits as well as indicates their physical and emotional wellness. In current food intake tracking applications, often dietary data must be manually recorded and they do not fully facilitate retrospective of the data. The paper suggests a design that transforms posts about food to "a postcard" and promote information hidden in food postings. The postcards are made from posts of food and may contain nutritional values of the food along with the user's interest. In this paper, the author analyzes the users' food posts with around 20 participants to see how they interpret the information, action, and their underlying aspects. According to the feedback, the design mentioned in the paper motivates people to think about their physical and mental health in a new way than traditional visualization.

The second paper we reviewed was "Modeling Food Popularity Dependencies using Social Media data"[2]. As people can share photographs and textual information about their daily food habits, so much food-related data can be easily obtained. Using these

data the spatiotemporal popularity of various cuisines can be analyzed. Business owners and restaurant investors can benefit from tracking the popularity of different food varieties and retail locations through time and space. In this paper, they show how to use geo-tagged data from social media, Google pictures, and Yelp to detect trends and popularity of cuisine kinds in a given area using off-the-shelf machine learning algorithms. They apply Kernel Density Estimation to get hot areas across the location after correcting for time, and Bayesian Networks to represent the dependencies among culinary cuisines popularity. Their investigations are focused on the Manhattan borough of New York City, but the approach may be applied to any place having social media data and information about retail establishments.

Once we got to know which all machine learning models and public API's were used to identify the trends and popularity of cuisines in the above two papers, we wanted to explore on the impulse on people buying these foods. Hence we referred the paper "Online impulse buying of organic food: moderating role of social appeal and media richness"[3]. In this paper they look at the relationship between the user attitude and the users' online buying behavior by looking at the effect of social media appeal and the application richness on buying organic food. The had gathered information over the course of a couple of months via an online questionnaire hosted on WeChat, which is a popular social network platform. The findings demonstrated that social appeal is moderated by communities and people, which influence user attitudes in both cognitive and effective ways. The findings prompt virtual marketing experts to reconsider strategies for dealing with modern consumers, such as designing websites with social learning mechanisms that are more user-friendly and visually appealing to encourage organic consumption.

At last, we wanted to look into paper which spoke about food awareness and how people can communicate about the food they had and give reviews and ratings for the restaurants. We came across a paper related to this called “The utility of social media as information platforms for public food safety assurance - The perspectives of users” [4]. This paper had information about the consumer knowledge of food safety and mechanism to spread this knowledge across friends. In the experiment they used a social media platform to successfully communicate vital food safety information and expertise to the public. The Taichung Hygiene Bureau established the Food Safety Youth Army (FSYA) to spread critical food safety facts to the public. Food safety facts were distributed to the public via popular social media platforms like Facebook and WeChat. The study's main goal was to learn about users' impressions of the usefulness of various food safety information systems. Users have opinions of these networking platforms, according to the findings. Users did learn about food safety using these networking platforms, according to the findings of this paper. Based on these findings of the study, they propose that government agencies use social media features to communicate crucial information to the public. Hence as social media tends to play a vital factor in delivering information about food and food safety, we decided to build a social networking platform for food lovers.

1.2 Current State of the Art

In the literature search, we saw many techniques and machine learning algorithms that help in identifying and recommending recipes and restaurants. We also got to know the existing 3rd party API’s available and the optimal techniques to build a networking platform.

Our web application enables food lovers to search restaurants, track their eating history, as well as connect with their friends. As we plan to build web application for users across the world we plan to design the application such that it is easily scalable. It's best to deploy the web application on any of the public cloud platforms so that we offer our application in a Software as a Service manner. For the deployment of the web application, we plan to make use of some of the below-mentioned AWS services.

Amazon EKS:- The elastic kubernetes service of amazon provides a cluster management platform for our application.

Amazon S3:- A blob storage service from amazon for storing media content, files, and objects.

Amazon API Gateway:- A gateway service from amazon for secure and efficient traffic filtering and routing.

Amazon Lambda:- A serverless service for handing requests at scale and completely managed by AWS.

Amazon Cloud Formation:- A service for managing infrastructure through code.

CICD pipelines:- Jenkins for continuous integration and deployment.

Tensorflow:- For training and building machine learning models.

Amazon Sagemaker:- A platform for developers to train, create and deploy our machine learning model on the amazon cloud.

1.3 Project Justification

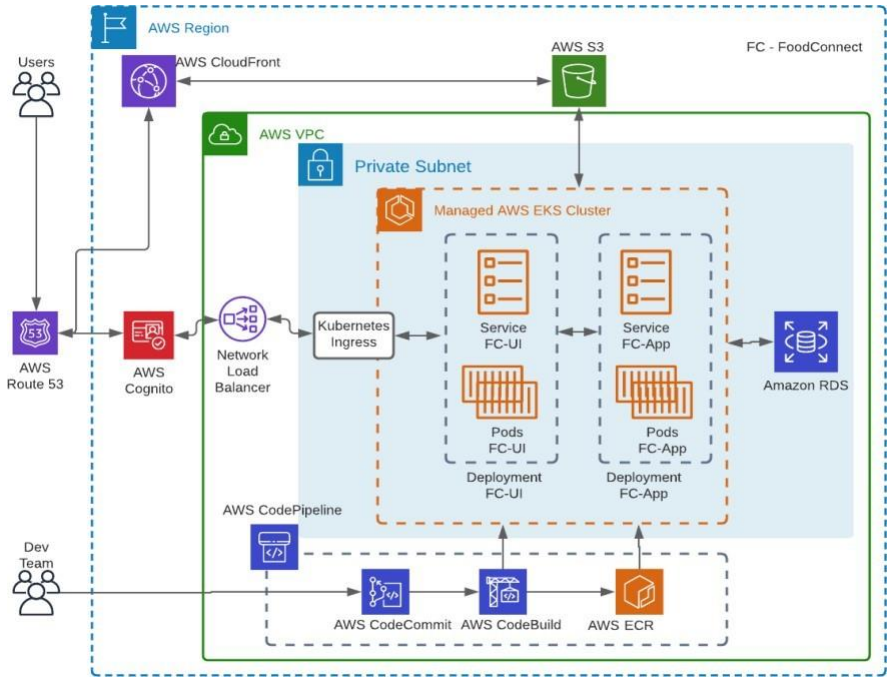
People love to eat and go outside, spend time with their friends and family. Before choosing any restaurants, they do online search, look for reviews, ratings and comments. They want the best experience such as restaurant ambiance before finalizing the restaurants or ordering online. Online research shows that people regularly read online reviews and trust those reviews as much as a personal recommendation. Yelp, which is a very popular online review and rating platform. It provides reviews and ratings for almost all kinds of business, but it doesn't address issues that are particular for businesses such as restaurants. Yelp lack following features for restaurants recommendations:

1. Creating Foodzone (social network for foodies).
2. Enable users to share their experience for a restaurant.
3. Recommend popular dishes for a restaurant.

We are proposing FoodConnect which addresses the above issues. FoodConnect is a geosocial networking web application that enables users to check in the restaurants visited and share the restaurant details with friends. The application is used to discover and explore good restaurants nearby and provide valuable feedback about the restaurant visited. It is a platform that helps in connecting with people who love food and enables users to share their experiences on the restaurants visited, which would help users to identify good restaurants, recommended dishes, and services a restaurant has to offer. It enables the user to keep track of all the restaurants visited and provides a recommended list of good restaurants based on the user’s interests.

Chapter 2. Project Architecture

2.1 Deployment Architecture



FoodConnect - Microservice Architecture using AWS Elastic Kubernetes Services

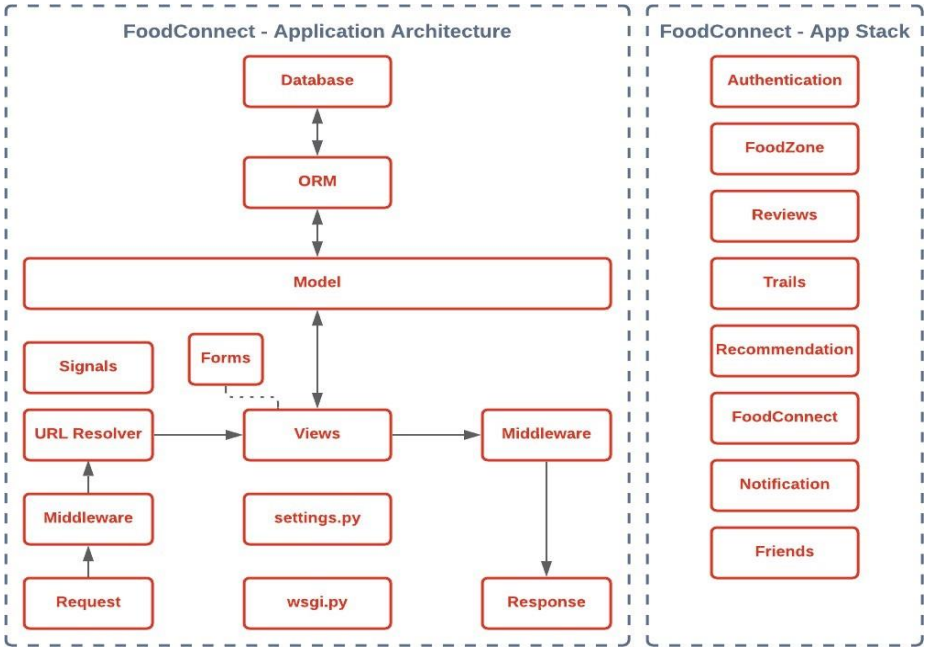
FoodConnect is a microservice based distributed system. To develop microservices we are using Django web application framework for backend, React for frontend and AWS based technology stack to build and deploy our microservices. We have two microservice: FoodConnect-App, FoodConnect-UI. Following are the steps which explains how this architecture works:

We are using AWS Elastic Kubernetes Service to deploy our microservices. It is a fully managed service. The AWS EKS cluster runs nodes which run services, pods for FoodConnect-UI and FoodConnect-App separately. AWS EKS is able to scale horizontally

seamlessly. It runs in a private subnet and is connected to a public network via Kubernetes Ingress via Network Load Balancer.

All incoming requests from AWS Route 53 are authenticated via AWS cognito and forwarded to the Network Load Balancer in the primary AWS Region. Whenever the Dev team pushes a new feature, it pushes via AWS DevOps pipeline which consists of AWS CodeCommit, AWS CodeBuild and AWS ECR. Once AWS ECR builds a new docker image and Kubernetes rolls out a new application in pods.

2.2 Application Architecture



FoodConnect - Application Architecture

FoodConnect has following Django application:

1. Authentication Module - Manages user authentication via AWS Cognito.
2. FoodZone Module - It serves FoodConnect dashboard and other related features.

3. Reviews Module - Provides user rating, reviews and image upload related features.
4. Trails Module - Keeps track of user's restaurant visits.
5. Recommendation Module - ML based restaurant, dish, friends recommendation.
6. FoodConnect Module - Main app which centrally manages other apps.
7. Notification Module -Send event notification such as friend requests, user restaurant visits etc.
8. Friends Module - Manages user's friend network, such sending, accepting or rejecting requests.

Chapter 3. Technology Descriptions

3.1 Client Technologies

3.1.1 React Framework

The front end was implemented using React framework. The application is designed using simple views for each state in the application and react makes sure to update and render the components during any change in data. The components are built in encapsulation, and this helps in managing the own state and composing complex UI. React components are built using JavaScript instead of templates this enabled the application to pass data easily and keep the state out of DOM.

3.2 Middle-Tier Technologies

3.2.1 Middleware

The Food Connect web application is built using python in Django framework. Django middleware is a framework of hooks which assist in request and response processing. The middleware is light in weight and is a low-level plugin system for altering the in input or output response in Django. Each of the component in middleware is responsible for performing a specific task. The web application is built on many in built Django middleware component. The middleware supports either synchronous python or asynchronous python.

3.3 Data-Tier Technologies

3.3.1 AWS RDS

The data for the application is stored using the AWS Relational database Service, which is a distributed relational database, running on cloud which is designed to simplify the operation, setup, and scaling. The Amazon RDS synchronously replicates the data to different availability zone and helps us run the application

in amazon virtual private cloud, this enables to isolate the database instance to the existing infrastructure through an industry standard encrypted IPsec VPN.

3.3.2 AWS S3

The Amazon S3 bucket is used to store the images in the application. The application provides a feature that enables user to update the profile image and to upload the images for the restaurant visited. Amazon S3 through a web service interface provides the object storage for the images. The different storage management offered by S3 enables to manage the cost. The access management provides the bucket auditing feature.

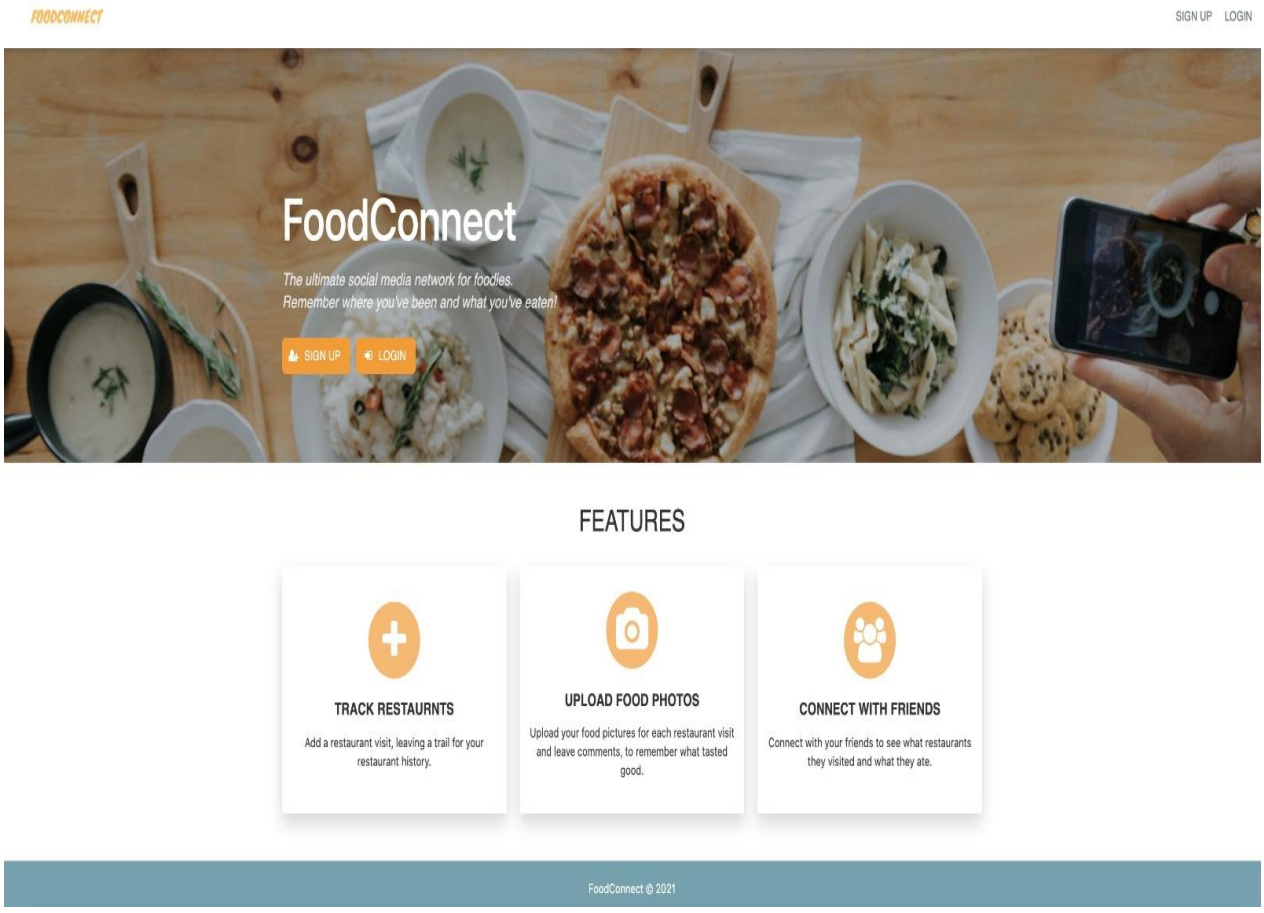
Chapter 4. Project Design

Add additional chapters if necessary to keep chapters at a reasonable length. This chapter should describe the important design elements of your project. Describe elements that are key to project and that are innovative. The topics below are for a typical MS Software Engineering project. Adjust the topics in this chapter to meet the needs of your project.

4.1 Client Design

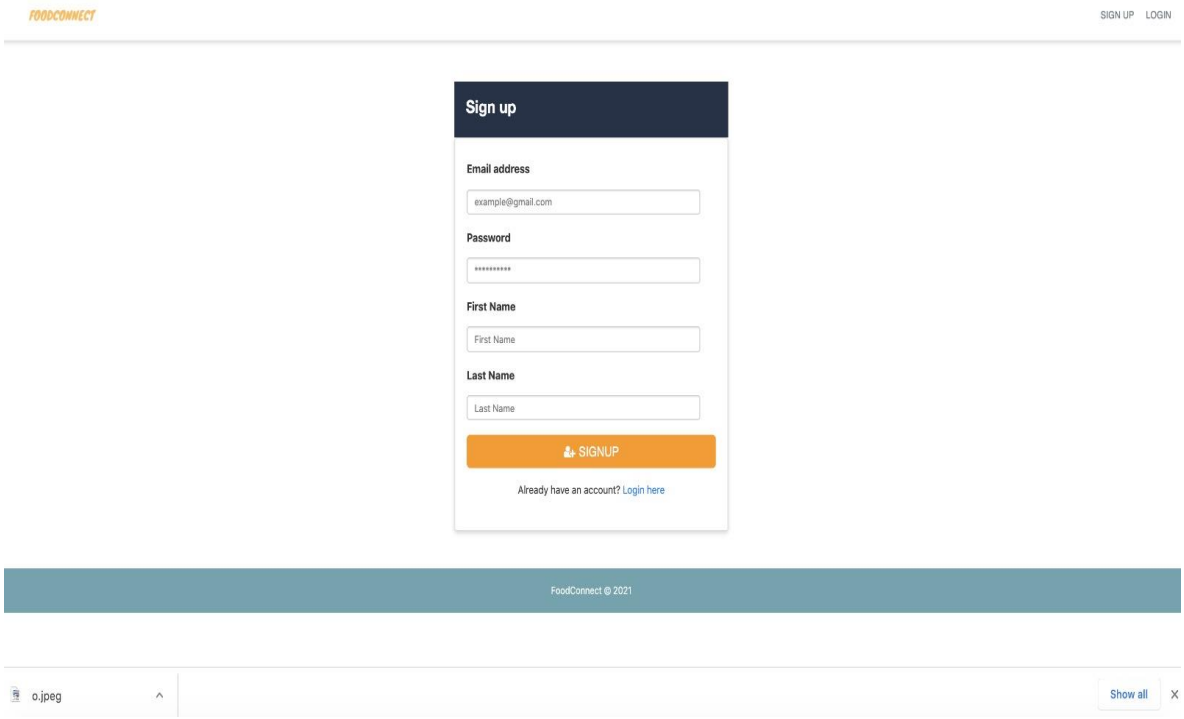
4.1.1 UI Wireframes

Home Page:



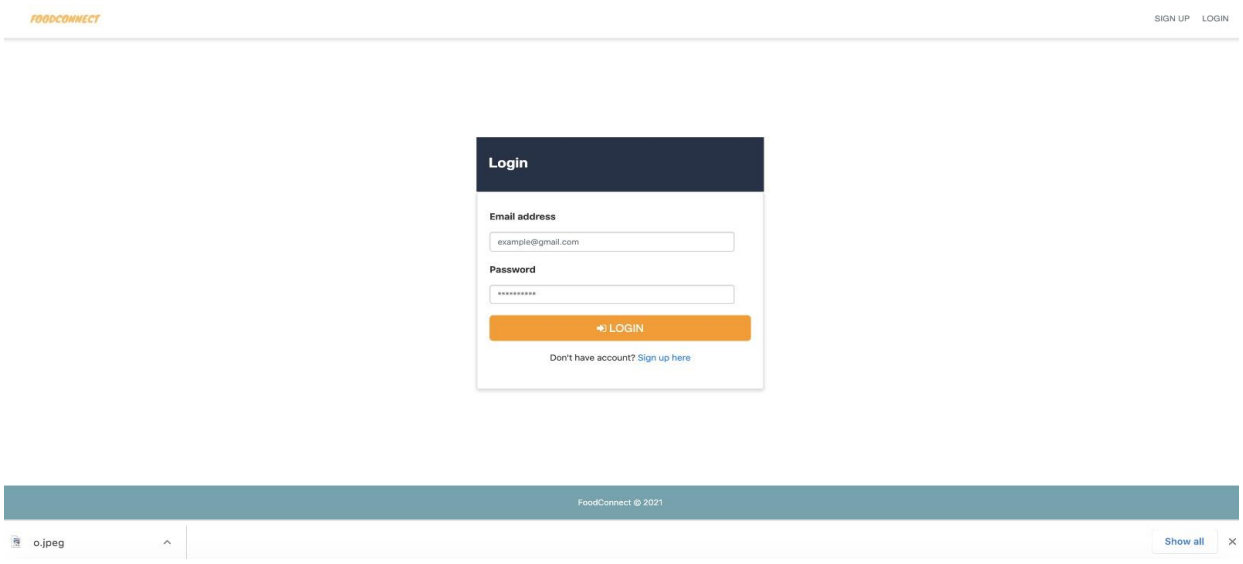
This is the home page of the application where the user can signup or login to the application.

SignUp Page:



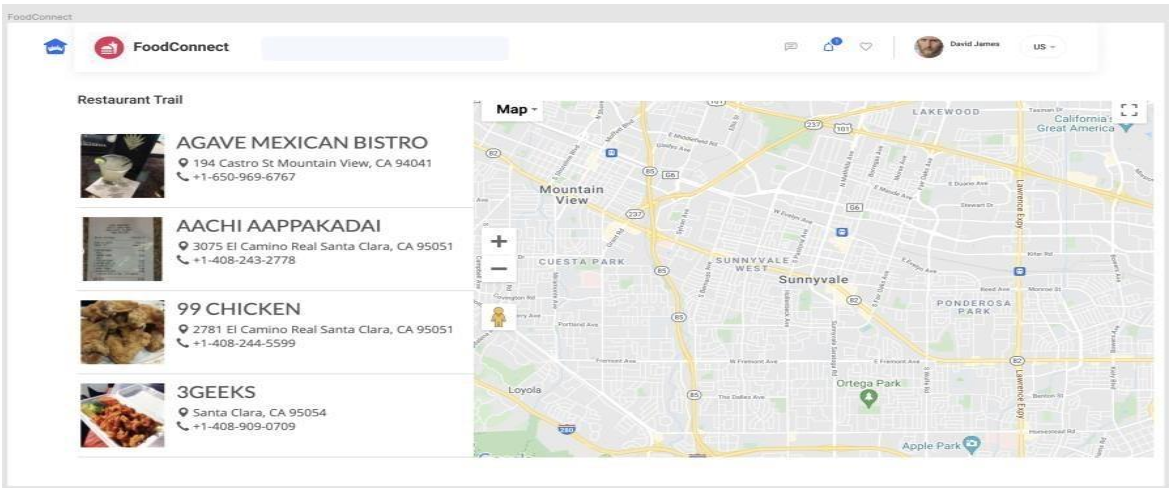
The registration page allows the user to register a new account with name, email, and password. The registration is made to collect the restaurant trail details and store it in the database for the respective user.

SignIn Page:



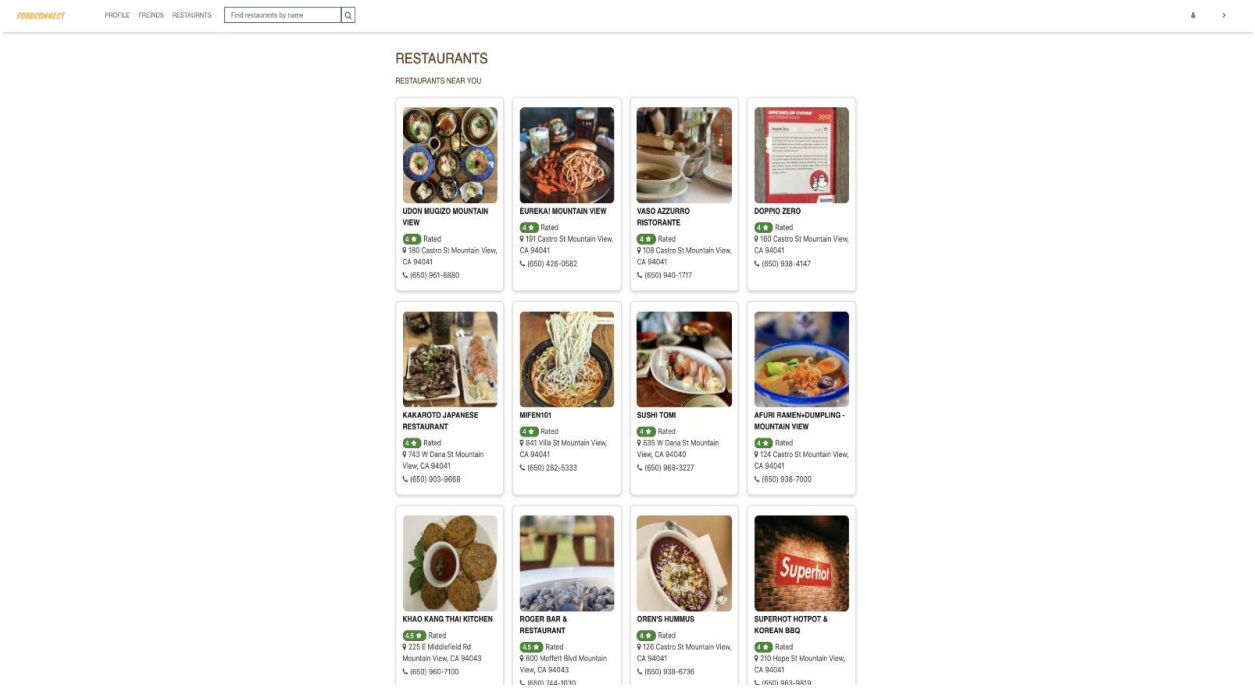
The login page enables registered users to access the application with email and password.

Restaurant Trail Page:



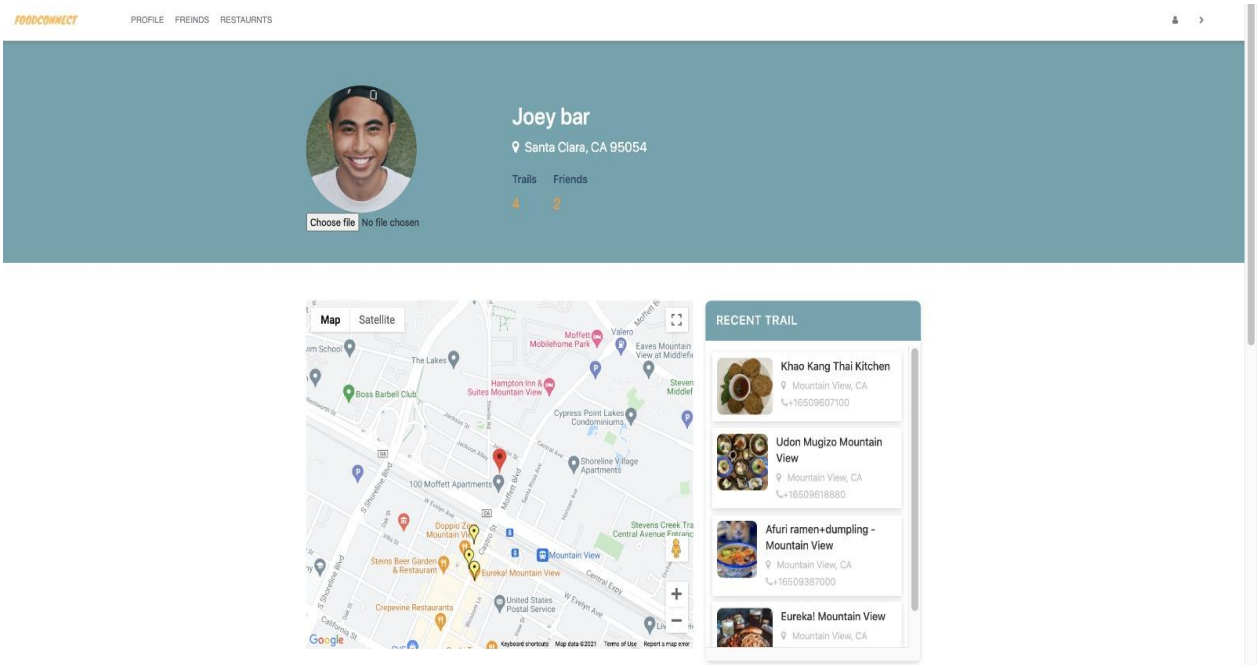
The first page that comes up after the user logins is the restaurant trail page. The page provides the user's trail history which lists out the restaurants visited nearby.

List of Restaurants Page:



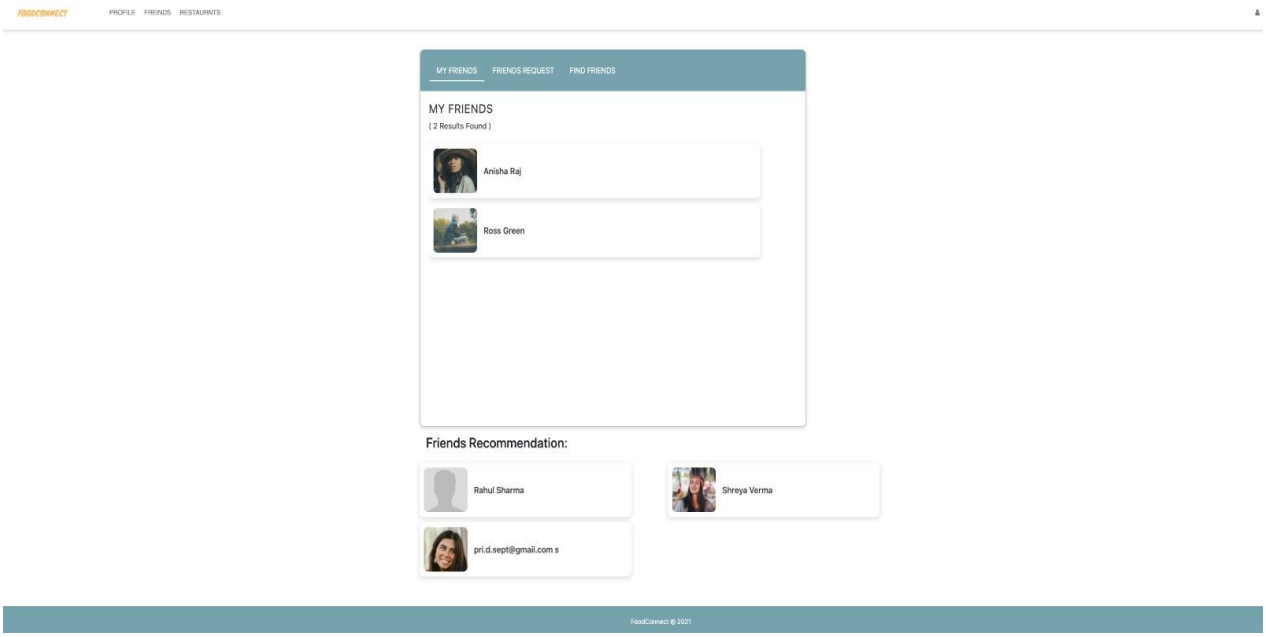
Displays the list of restaurants, rating, address and phone numbers of the restaurants present in the user's location.

Profile page:



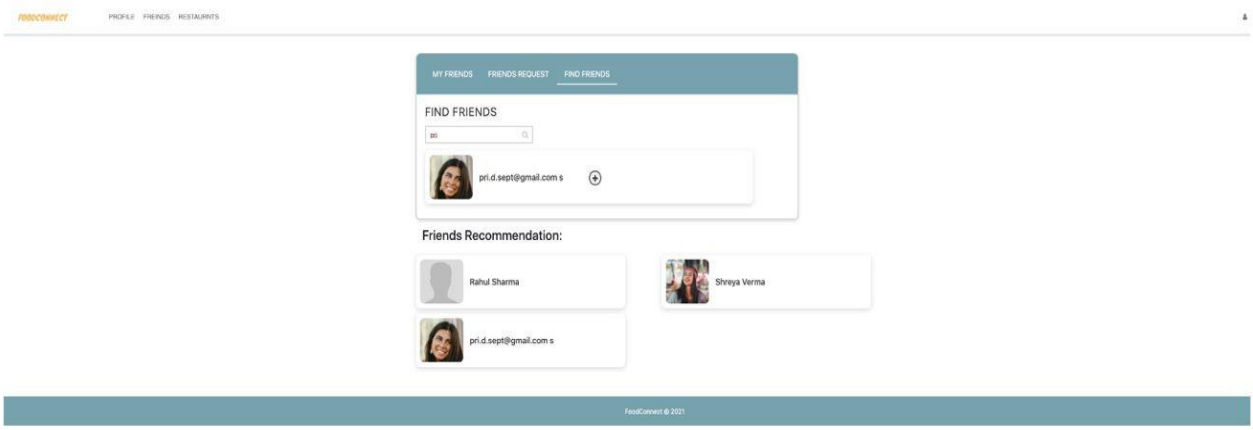
Displays the user's profile and the list of recently visited restaurants and their corresponding locations as marking pointers on Google Maps. Also, the page shows the current user location.

My Friends Page:



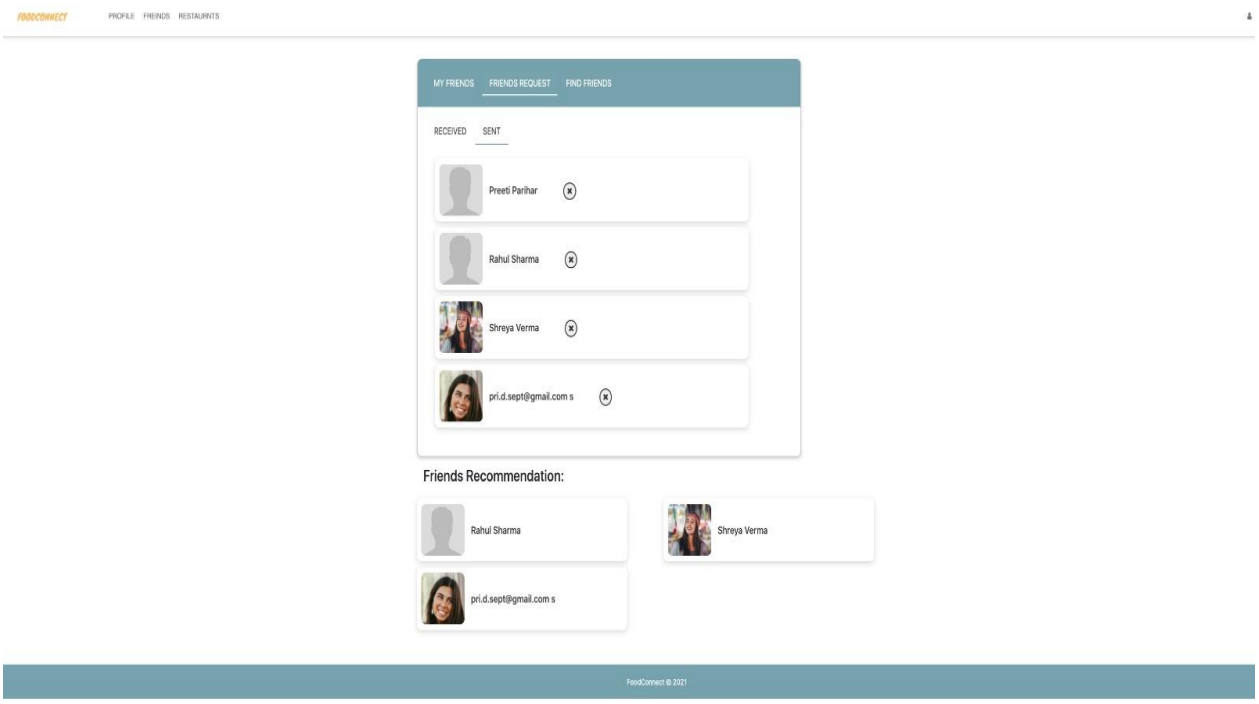
My Friends page displays the list of friends along with their profile picture and also displays the recommended list of Friends.

Find Friends Page:



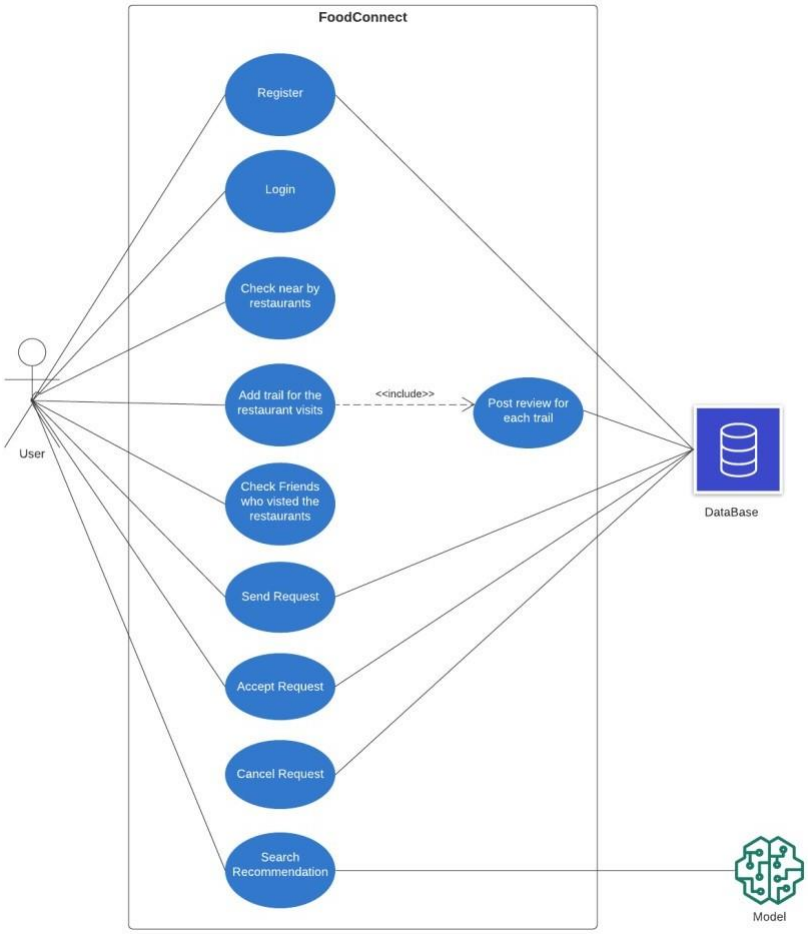
Find friends page displays the list of potential friends based on the users search.

Friend Request Accept/Reject/Send Page:



Friend Request Accept/Reject/Send Page helps the user to send a friend request, accept/reject a friend request.

4.1.2 Use Case Diagram

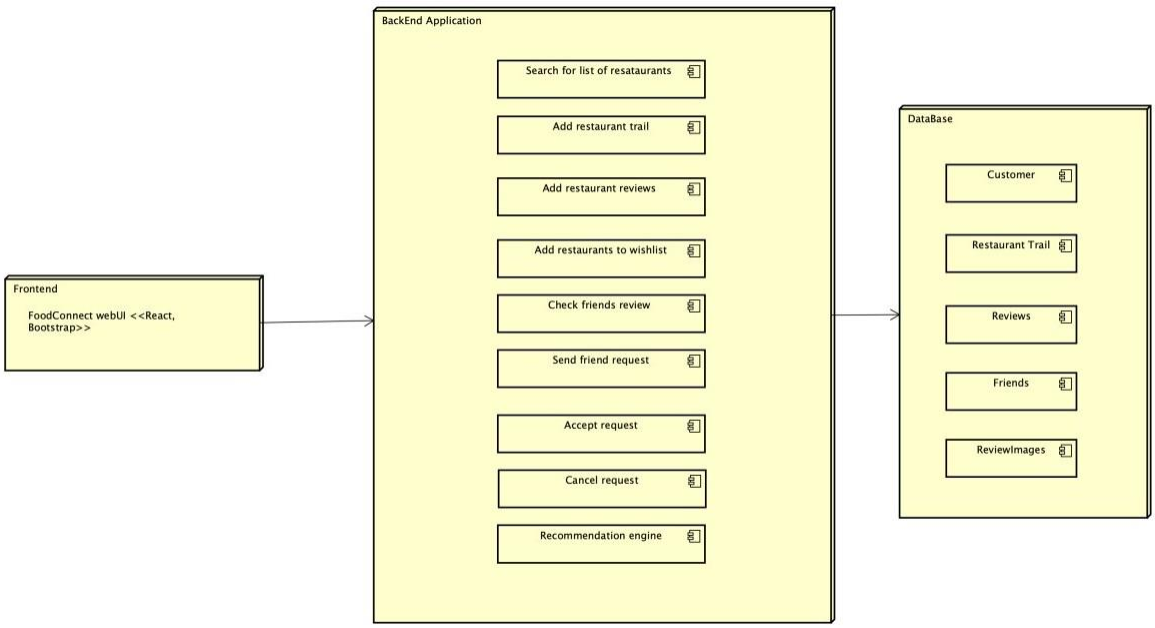


Above diagram shows that there is mainly one single actor involved in the project, that is the end user. The initial use cases are to register and login, after successful registration the user's details will be stored in database. When the user logins, the user can check for nearby restaurants using the location using the google API. The list of restaurants for the provided location is fetched using the 3rd party yelp API, the user can add trail to any of the restaurants

visited and share the restaurant experience with friends. The user can also send, accept, or cancel any requests received.

4.2 Data-Tier Design

4.2.1 Deployment Diagram



The above Deployment diagram conveys that, the frontend for Food Connect web application is built in JavaScript using React framework and Bootstrap for styling. The backend consists of set of APIs as shown in the diagram that structures the functionality of the application. The Database consists of five different table to store the customers, restaurant trial, reviews, friends, and review images details.

Chapter 5. Project Implementation

5.1 Client Implementation

5.1.1 React Components

1. **Restaurants Search Component**: Users will have the ability to search nearby restaurants based on the restaurant name or address. Users can see the restaurants rating, contact number, price and distance.
2. **Restaurant Trail Component**: Users can add the restaurants in the list of restaurants visited and these would be marked on the google maps which is visible to user
3. **Upload Photo component**: Users can upload the images of the food taken by them in the restaurant.
4. **Review Component** :- Users can drop review comments about the food they had, their best recipe and can comment about the restaurant.
5. **Send/Receive friend requests component**: Users would be able to search their friends and send friend requests. The User can either accept or reject friend requests.
6. **Friends component** :- Users would be able to see the list of his friends.
7. **View trail Component**:- Users would be able to see the list of visited restaurant trails of their friends.

5.2.1 Two factor Authentication

The application is enabled with an extra layer of security with two-factor authentication. Once the user registers an authentication confirmation is sent to the user's mail id with a passcode that needs to be entered to be successfully registered with Food Connect. When the

registered users login the authorization server issues a access token and refresh token to make an artifact ensure that the client uses a secure call to the API server.

5.2.2 Google Map Integration

The web application trail component utilizes map to locate the trails visited by the users. The map is integrated with the client using google map which helps us to customize the add a marker for each trail visit, which enhances the user experience.

5.2 Middle-Tier Implementation

5.2.1 Middleware Components

Django middleware uses framework of hooks which assist in request and response processing. The various middleware components used in the application are:

- CorsMiddleware
- SecurityMiddleware
- SessionMiddleware
- CommonMiddleware
- CsrfViewMiddleware
- AuthenticationMiddleware
- MessageMiddleware
- XFrameOptionsMiddleware
- AuthenticationMiddleware
- RemoteUserMiddleware
- AwsDjangoMiddleware

5.3 Data-Tier Implementation

5.3.1 Data Model

The data for the application is stored using the AWS Relational database Service, which is a distributed relational database, running on cloud which is designed to simplify the operation, setup, and scaling. The Amazon RDS synchronously replicates the data to different

availability zone and helps us run the application in amazon virtual private cloud, this enables to isolate the database instance to the existing infrastructure through an industry standard encrypted IPsec VPN.

5.3.2 S3 Bucket

The images created is stored under standard S3 bucket. The bucket policy is configured in order to optimize the cost, where it is configured to move the images greater than 75 days old to S3 IA and images which are greater than 365 days old are archived by moving it to the Glacier bucket. To address the load and to improve the latency the images are cached and are made highly available facilitating targeting service using the content delivery network, using Amazon CloudFront.

5.4 Restaurant Recommendation System:

Food Connect is the platform where food lovers collectively share food experiences and can find new and exciting taste. For users to get personalized restaurant recommendations we have tried to build the recommendation engine for the restaurants. Using this Machine Learning Model, we will be recommending restaurants based on the restaurants that user has previously visited. We tried to provide recommendation based on collaborative filtering. We experimented User-Item based collaborative filtering and Item-Item Based collaborative filtering Methods.

5.4.1 Data Collection:

The Data that we used was an open-source version of Yelp_Review_Dataset provided by yelp.com. This dataset comprises reviews posted by all users for restaurants.

5.4.2 Data Pre-Processing and Feature Extraction:

The Yelp_Review_Dataset was a very huge dataset of 6 GB and it was very difficult to process it on our computer or even Google Collab Pro. For JSON to be used by the application the JSON data is supposed to be loaded onto memory. As we have 6GB of data to load, we were running out of memory.

To access this dataset and make it consumable we had to perform multiple pre-processing so that we don't load the entire dataset into the memory. We divided the dataset into Chunks of data using Pandas Library. Using this we extracted user information only. We extracted User_ID, Restaurant_Id, and review stars from the dataset. We removed missing values and to make the data consumable by machine learning models we converted user_ids into numbers using LabelEncoding from the SKlearn Library.

5.4.3 Model Training and Evaluation:

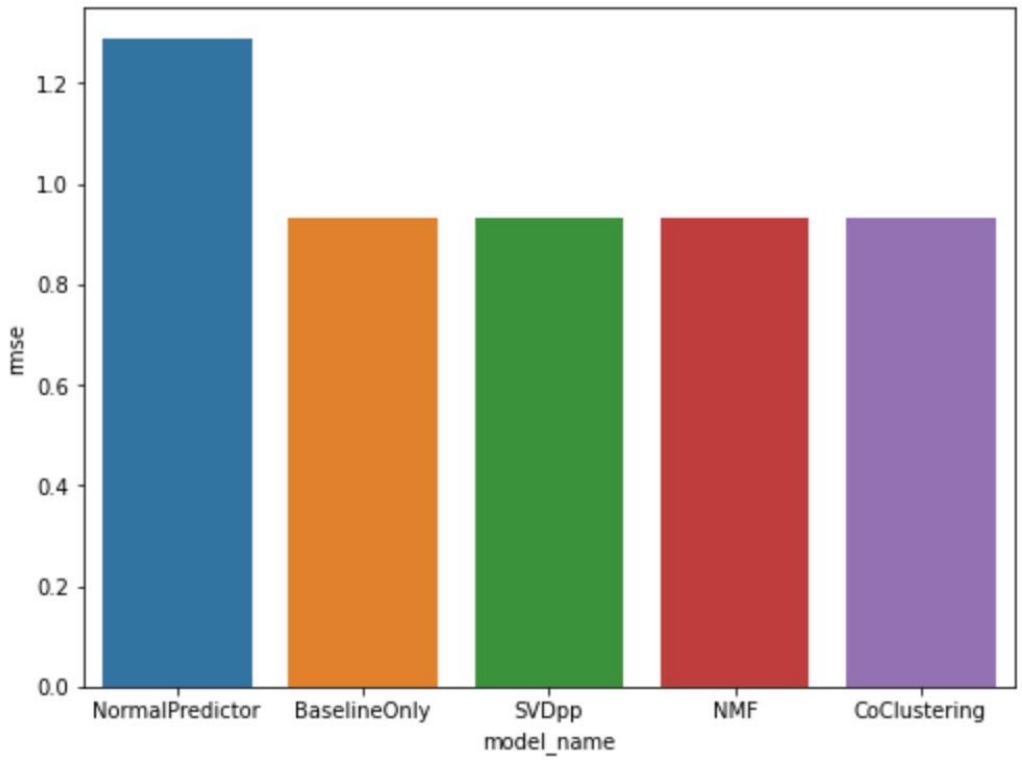
- **User-Item Collaborative Filtering:**

The Data was finally reduced to 17,00,000 reviews User-Item Collaborative Filtering. As the user that we have in our application is not going to be a part of the current dataset, we decided to take only the mean of the reviews of restaurants. Subsequently making the data reduction to 32000 restaurants ratings.

Using this we have used Surprise Library which is used to build a recommendation engine. We Split the input data into 80% Train and 20% Testing datasets.

Then using the surprise libraries Reader module converted the data into Surprise trainset.

Later tried training on K-Nearest Neighbor algorithms, Normal Predictor, Baseline only Algorithms, Singular value decomposition, Co Clustering algorithms, etc. Following is the Root Mean Square Error of the algorithms, Lesser the error better the model. So Model Performed better.



- **Item-Item Collaborative Filtering**

In this type of recommendation system input is a restaurant (ITEM) and recommendation is a List of recommended restaurants(ITEM).

For this, we need to have pivoted table. Rather than having user-restaurant-rating we need to have a matrix structure, in which all the restaurants are rows, all the users are columns, and value is the rating as below

user_id	business_id	review_stars
ak0TdVmGKo4pwqdJSTLwWw	buF9druCkbuXLX526sGELQ	4
YoVfDbnISIW0f7abNQACIg	RA4V8pr014UyUbDvI-LW2A	4
eC5evKn1TWDyHCyQAwguUw	_sS2LBIGNT5NQb6PD1Vtjw	5
SFQ1jcnGguO0LYWnbbftAA	0AzLzHfOJgL7ROwhdww2ew	2
0kA0PAJ8QFMeveQWHFqz2A	8zehGz9jnxPqXtOc7KaJxA	4
...	...	...
DCqK3QO1HiO_HMMAjGQWXg	gFwIUy6xBvG1M0VTfgZZFw	5
eQhK6dajPnWo4eQj6-Oy4g	Ag89Kt5lObftTZG7yPCMsQ	4
ae5PRyJfV6wJ8tD1IsR8fA	xLEyXF9LCaoUxi_MgW3Nrw	1
3bJ2zvYx2HbpxvedNmp8Gw	IFt7w7weA2pwp45Ulvaevg	1

For this, we need to have pivoted table. Rather than having user-restaurant-rating we need to have a matrix structure, in which all the restaurants are rows, all the users are columns, and value is the rating as below

business_id	-164t1nclzzmca7eDiJMw	--zLxanRCkHHl6l6Mlqobg	-0A60UZl9nbdq2WWySJ_tQ	-(
user_id				
--Cq0V6vgQQDU8oQEa0NGQ	0	0	0	
--QEBVe2PKMXI2i9332HjQ	0	0	0	
--eaFajuio8YSUV4Wk8PQg	0	0	0	
--kzKp2AUaQHQsVWhinTYQ	0	0	0	
-0EI05gI9jF6qdcJxWBbaw	0	0	0	
...	...	...	...	
zyqyrDoFh7hBzACSSfj0_Q	0	0	0	
zyvkEj0aiaW0ZP2h18lsRw	0	0	0	
zzHvKkalUfONRV5hK7GJLA	0	0	0	
zzYF64kl9es4c16fv68OIQ	0	0	0	
zzl-SGlkzcs65g1bFq3cpA	0	0	0	

16285 rows × 8688 columns

As it is evident that many users might not have visited many restaurants, we will be having a majority of 0's which don't carry useful information. Hence this matrix is condensed into a sparse matrix. We have utilized the SciPy library to get the sparse matrix.

After this using the KNN algorithm we can suggest the N nearest restaurants for the supplied restaurants. The KNN algorithm is as on need basis.

Using the Sparse Matrix, and KNN Machine learning model restaurants can be recommended. This is usually the best way to recommend in classical machine learning when we have new users coming up in the system and hence this is the process we have used in our application.

Chapter 6. Testing and Verification

6.1 Server-Side Testing

This will involve testing of all the Django APIs, to check if they are giving desired output for the given input. We have used "TestCase" class of Django REST framework, to create and execute test cases for the custom APIs.

Following are the server-side test cases:

Description	Request type	URL	Parameters	Verification
Login API	POST	/login	Authorization headers, with username and password in base64	{success: boolean, accessKey: string}
SignUp API	POST	/register	Authorization headers, with username and password in base64. Body: {firstname: string, lastname: string, phoneNumber: number}	{success: boolean}
Get Restaurants API	GET	/getRestaurants	{longitude: string, latitude: string}	{success: boolean, restaurants: [{Restaurant data}]}
Add visit details by the user	POST	/addVisit	{rating: number, review: string, images: [blob], topDishes: string}	{success: boolean}
Get user details	GET	/getUser	emailId	{visits: number, friends: number, firstname: string, lastname: string: phoneNumber: string,}

Add Friends	POST	/addFriends	{userId: string, friendId: string}	{success: boolean}
Get Friends	GET	/getFriends	{emailId:string, status: string}	{friends:[{name: string, email:string}]}
Get restaurant by ID	GET	/getRestaurantDetails	restaurantId	{listofVisitedFriends: [{firstname: string, email:string}], visitedDetails:{}}

Table 2: API

6.2 Front End testing

This will involve testing of the all the React Components, to check if they are working correctly.

We have used JEST framework, to write the test cases for UI components.

Following are the UI test cases:

Description/Action	**Verification**
Login Page: User enters Email and password and clicks on "Login" Button	Email should be in appropriate format. If user has entered right credentials, user should be navigated to Home page. Otherwise display an error message
SignUp Page: User enters details and clicks on "Register" Button	Email should be in appropriate format. The user should be navigated to login page and requested to Login, to proceed. Otherwise display an error message
Home Page	List of restaurants based on user location are displayed. Restaurant name, location, Review, Price and image is displayed. Header will have a user icon.
Click on a restaurant in Home page	User is navigated to restaurant details page. All the images of restaurant are displayed. User's friend list who have visited the restaurant are displayed. Review given by the user for the restaurant is displayed. If the user has not visited the restaurant, "Add Visit" button will be displayed.
Click on "Add Visit" Button	A pop-up will be opened, requesting the user to enter Rating, Review, top 5 liked dishes and pictures.

Click on "My Profile"	User is navigated to user profile page, where user details are displayed. An edit icon is also displayed, clicking on which, user details will be editable. "Save" Button when clicked, will save the details.
Click on "My Visits"	User is navigated to "My Visits" page. A map will be displayed to show the User's restaurant trail. Beside the map, list of restaurant details visited by the user will be displayed. If user has not visited any restaurant, Map will show user's current location is displayed.
Click on “Logout”	User is logged out from the application and is navigated to “Login”
Click on "Friends"	User is navigated to Friends page, where folllowing tabs are displayed. "My Friends", "Sent Requests", "Received Requests" and "Find Friends"
Click on "My Friends"	Details of user's friends will be displayed. Clicking on each friend, will navigate to that user's "My visits" page.
Click on "Find Friends"	User is navigated to "Find friends: page. This will have a search bar, where user can search friends by name and send requests.
Click on "Sent Requests"	A page with user's sent requests will be displayed. Every request will have a "Cancel" button, clicking on which will cancel the request.
Click on "Received Requests"	A page with the pending friend requests are displayed. "Accept" and "Deny" buttons are displayed along with every request.

Table 1: UI test cases

6.3 Integration Testing

After integration of front end and backend components, we have done integration testing, to ensure that everything is working as required. Integration tests are automated and the end to end functionality of the deliverable is tested. Integration tests are mainly checking the functionality of each components of the system. Validation of these tests are performed based on the expected behavior of the web application. A test result suit is generated which helps in analyzing the results of these tests and helps in easier debugging/testing of various features of the product.

6.4 Recommendation Engine Testing:

The various Steps involved are:

- Testing the data: Making sure that pre-processed data to be used for recommendation engine is in proper format.
- Testing the features: Checking if the features generated adhere to feature schema before feeding it to the model for inference.
- Testing the model performance: It involves evaluating the performance of the model using metrices such as Accuracy, AUC score, etc.

Chapter 7. Performance and Benchmarks

7.1 Application Performance

Our application is a social networking website for food. Hence, it can be expected to have many users using our application at the same time. Hence, we will make sure that our system can handle multiple users I at the same time. And effectively handle requests from them. For this we will be using State of the art services from Amazon Web Services like CloudFront, Route 53, EKS etc. We will test this by doing Stress testing on our application by opening concurrent connections using JMeter.

7.2 Benchmarks

Benchmarking were done both on the server side and front end side. A single node server was able to handle around 1200 concurrent requests. The memory consumption of the entire application is less than 65 MB. And the CPU utilization was lesser than 30 percent on a 4 core cpu. All the responses from the apis were coming in less than 200ms.

Chapter 8. Deployment, Operations, Maintenance

8.1 Cloud Deployment

We have deployed the complete application on AWS by:

1. Configuring Amazon RDS as the Database for the Application

2. Configuring S3 bucket and CloudFront to store images uploaded by the user in the application.

3. Configuring Amazon Cognito for Authentication and Authorization

4. Configuring Amazon CodeCommit as a source control system.

5. Configuring Elastic Kubernetes Service (EKS) cluster to deploy microservices for frontend and backend

6. Configuring AWS CodePipeline to trigger AWS CodeBuild which gets latest code from AWS CodeCommit and creates a docker image which is stored in AWS ECR

7. Configuring Network Load Balancer to generate external IPs for frontend service and to horizontal scale the frontend service using AutoScaling

8. Configuring a hosted zone in AWS Route 53 to host services the on the created domain.

8.2 Operations and Maintenance

Operations and Maintenance of our application is light weight as we have deployed our application on a fully managed Amazon Kubernetes Service. We have leveraged AWS services wherever required thereby the operations and maintenance required for our application is minimal. The application is a SaaS based web portal and users can easily operate it. The Kubernetes cluster manages the services, deployments, pods and takes cares things related to load balancing, auto scaling, pods creation etc which makes it very easier to manage the application.

Chapter 9. Summary, Conclusions, and Recommendations

9.1 Summary

Food connect is a geosocial networking web application that enables users to check in the restaurants visited and share the restaurant details with friends. The application is used to discover and explore good restaurants nearby and provide valuable feedback about the restaurant visited. It is a platform that helps in connecting with people who love food and enables users to share their experiences on the restaurants visited, which would help users to identify good restaurants, recommended dishes, and services a restaurant has to offer. It enables the user to keep track of all the restaurants visited and provides a recommended list of good restaurants based on the user's interests.

9.2 Conclusions

Food connect application would bring a lot of impact on people who would like to explore nearby restaurants, make friends, connect with them, and get to know about their friends food tastes and interests. The application is very unique as we can see the list of friends and their reviews on the restaurants visited by them. It helps in connecting with people who have similar food interests and also recommends restaurants which are nearby based on the reviews of the friends who have visited them. Users can see the reviews and ratings given by their friends on the restaurants visited and can make a record of all the restaurants visited.

9.3 Recommendations for Further Research

The application can be enhanced by including features where users would be able to chat with their friends in real time. This helps in planning for a nearby restaurant visit along

with their friends and also helps in getting to know the cuisines which they have tried out in these restaurants. The application can be further enhanced to recommend the best restaurant nearby to the user for a given recipe which the user would like to try. The recommendation engine for the restaurants can be improved by accurately predicting which restaurant the user would prefer based on the data collected by the previous visits and interests.

References

1. Sun, Zhida, Wang, Sitong, Yang, Wenjie, Yürüten, Onur, Shi, Chuhan, & Ma, Xiaojuan. (2020). Proceedings of the 2020 ACM Designing Interactive Systems Conference (DIS '20: Designing Interactive Systems Conference 2020). ACM.

2. Khulbe, D. (2019). Modeling Food Popularity Dependencies using Social Media data [arXiv]. In arXiv (pp. 5–5).

3. Tariq, Anum, Wang, Changfeng, Akram, Umair, Tanveer, Yasir, Sohaib, Muhammad, Xu, Jiuping, International Conference on Management Science Engineering Management author. (2020). Proceedings of the Thirteenth International Conference on Management Science and Engineering Management Volume 2 / (1st ed. 2020. ed., Vol.1002, Proceedings of the Thirteenth International Conference on Management Science and Engineering Management). Cham :: Springer International Publishing.

4. Tao, Kuan-Chuan, Lee, Yi-Ling, He, Bo-Jun, & Liu, Li-Wei. (2019). Proceedings of the 2019 3rd International Conference on E-Society, E-Education and E-Technology(ICSET 2019: 2019 The 3rd International Conference on E-Society, E-Education and E- Technology).ACM