

Design Document



Group 1: *Call Of Duty*

Harsh Patel, Jeet Patel, Akshay Patel, Rushi Patel,
Karmit Patel, Brandon Fierros

Table of Contents

1. Project Overview.....	03
1.1 Overview of the design.....	03
1.2 Problem.....	03
1.3 Stakeholders.....	03
1.4 Solution to the Problem.....	04
1.5 Appendix User Stories.....	04
1.6 Visualization of Progress.....	05
 2. Architecture Overview.....	 11
2.1 Subsystem Architecture.....	12
2.2 Alternative Designs.....	13
2.3 Deployment Architecture.....	13
2.4 Persistent Data Storage.....	14
2.5 Global Control Flow.....	15
 3. Detailed System Design.....	 16
3.1 Static View.....	16
3.2 Dynamic View.....	19

1. Project Overview

1.1 Overview of the Design

The overview of our project is to provide a customer with the reviews of the food served at restaurants which not many websites or applications provide. We let the user to Login or Signup with a unique Username and a Password. Once they Login to the website they can read or write the reviews for the specific food for a specific restaurant. Users can also add the new menu item to write a review about. The website also has a search bar where users can find the desired restaurant/menu item and look at the reviews of the food for a specific location. The purpose of this website is to provide customers with reviews of a variety of different food items served at various restaurants.

1.2 Problem

There are so many restaurants people ignore to visit just because they don't know the quality of food they serve. If they decide to go then it gets awkward sometimes when the waiter asks you what can I get you sir/madam. To avoid this kind of situation people also do little research about the restaurants, look for their menu, the ratings of the restaurant, etc. but the internet does not provide detailed information on menus. It is true that there are many different applications as well as websites available to check the ratings of the restaurants like Yelp, Zomato, Zagat, and many more but none of them give the ratings to the food. This creates the short list of restaurants people like to go.

Once the customer gets to the restaurant, they are left with choosing items from a menu that they have no idea about the quality of. Often, customers will be left disappointed by a highly rated restaurant because they order something that is not one of the better items offered by the restaurant. Our application looks to solve this problem by allowing the user to individually see the best menu items offered by each individual restaurant so that the customer can have the optimal dining experience they expect.

1.3 Stakeholders

Restaurant Owners: They are the ones who can benefit the most because the people who were unaware of some restaurants can attract to it after seeing the reviews of their food.

Foodie People: They can have a number of choices to select it from after knowing the kind of food the restaurants serve. People who like to try new restaurants will be benefited the most.

1.4 Solution to the Problem

We have created a web application called HangryBirds which allows people to rate the menu item. People can also add new menu items if missing or add new reviews of the menu items. One of the amazing facts about this application is people can try new restaurants and know what is the most rated food of that restaurant, and don't even have to feel awkward while ordering there. People can search for their favourite food and restaurant on their fingertips. This is the best solution we came up with for people who would like to try new things and new restaurants.

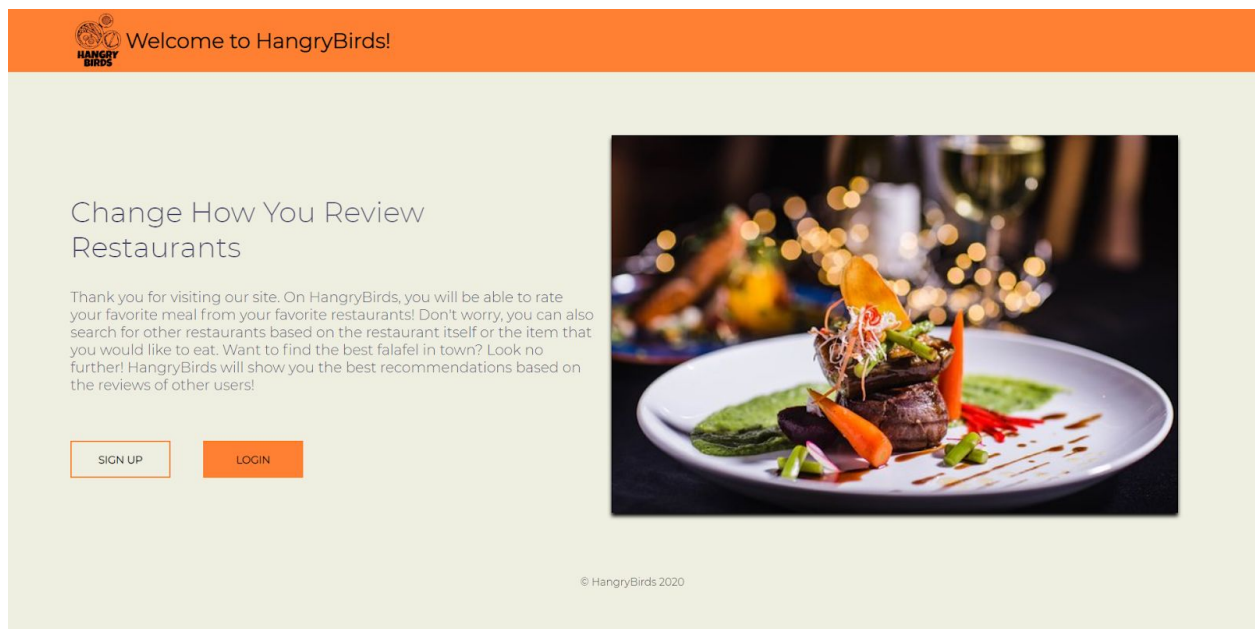
1.5 Appendix User Stories

Story #	Card Front	Card Back	Sprint Number	Priority	Assigned To	Comments
ST-0	As a <user who wants this functionality> I need a <what the user wants> so that <why the user wants it>.	Acceptance Criteria <what the user should be able to do with the functionality>				
ST-1	As a user, I need to be able to add a review for a food item with a star rating so that we can create a community of reviews and have a consensus on the quality of the item	Acceptance Criteria: The user can add a review for any food item	Sprint: 2	High	Harsh	The main concept of the app.....
ST-2	As an admin, I need all the home, login, review, etc. pages to be uniform in design	Acceptance criteria: Uniformity in design	Sprint: 1	Medium	Harsh	
ST-3	As a user, I need to be able to send a request to add a food item that is not currently included in the review database, so that all new items will be available and previously unknown items are available	Acceptance criteria: The user is able to send a request to add a previously non-existent item to the database	Sprint: 2	Medium	Brandon	Might not be possible due to API, May be dependent on the Restaurant not us
ST-4	As a user, I need to be able to search specific food items	Acceptance criteria: The user is able to search for specific food item	Sprint: 1	High	Jeet	Necessary Function
ST-5	As a user, I need to be able to filter search list by restaurant and food item	Acceptance criteria: The user is able to filter the search result	Sprint: 2	High	Jeet	Necessary Function
ST-6	As an admin, I need the design of user-authentication pages (login, logout, home, etc.)	Acceptance criteria: Design principles matches the project idea	Sprint: 1	High	Akshay	A Design feature
ST-7	As an Administrator, I need to be able to remove non-helpful or spam reviews that do not contribute to the integrity of the site.	Acceptance criteria: The moderators are able to remove bad reviews or spam	Sprint: 2	Medium	Rushi	An Administration feature
ST-8	As a user, I need to be able to sign up to write the reviews	Acceptance criteria: The user is able to create an account to unlock extra features of the web application	Sprint: 2	High	Rushi	Necessary Function

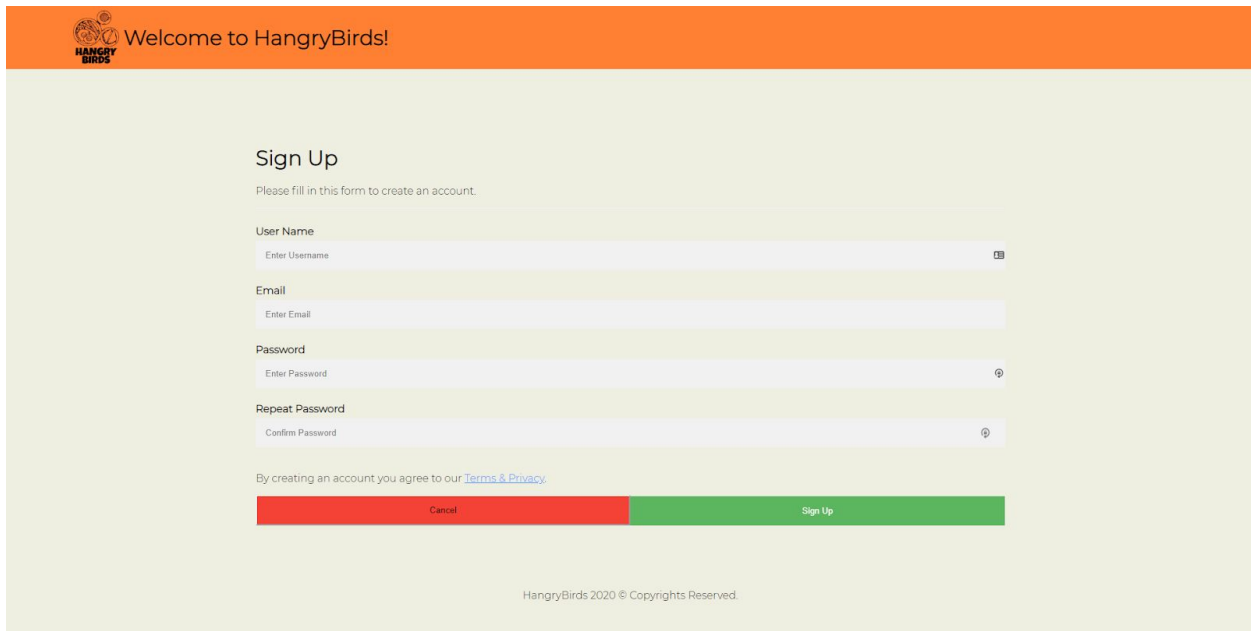
ST-9	As a user, I need to have user accessible review page in order to view or write a review	Acceptance criteria: The user is able to access a page that shows the currently highest reviewed items in the area	Sprint: 2	Medium	Jeet	Priority: medium
ST-10	As a user, I should be able to delete or edit my reviews	Acceptance Criteria: Edit or delete reviews	Sprint: 2	Medium	Akshay	Necessary Function
ST-11	As a user, I need to be able to access the web application on any device	Acceptance Criteria: Web application accessible on any device	Sprint: 2	Low	Akshay	Accessibility Function
ST-12	As a user, I should be able to see a list to 5 latest restaurants added to the website	Acceptance Criteria: Enabling newly added restaurants info on latest page	Sprint: 3	Low	Harsh	Optional
ST-13	As a user, the application should be user friendly making it as simplistic as possible	Acceptance Criteria: Design	Sprint: 1	High	Karmit	Accessibility Function
ST-14	As a user, I should be able to sign-in and write a review	Acceptance Criteria: Sign-in so that user can have more access	Sprint: 2	High	Akshay	Necessary Function
St-15	As a developer, I need all the webpage urls to work precisely and redirected to the correct pages	Acceptance Criteria: Include a option to search by cuisine	Sprint: 1	Low	Karmit	Optional
ST-16	As an admin, I should be able to accept and decline the food item requests posted by the user.	Acceptance Criteria: Accept and decline user requests	Sprint 3	High	Rushi	Necessary Function

1.6 Visualization of Progress

Main Page: On this page, users can a overview of the website along with the two options of proceeding through the application

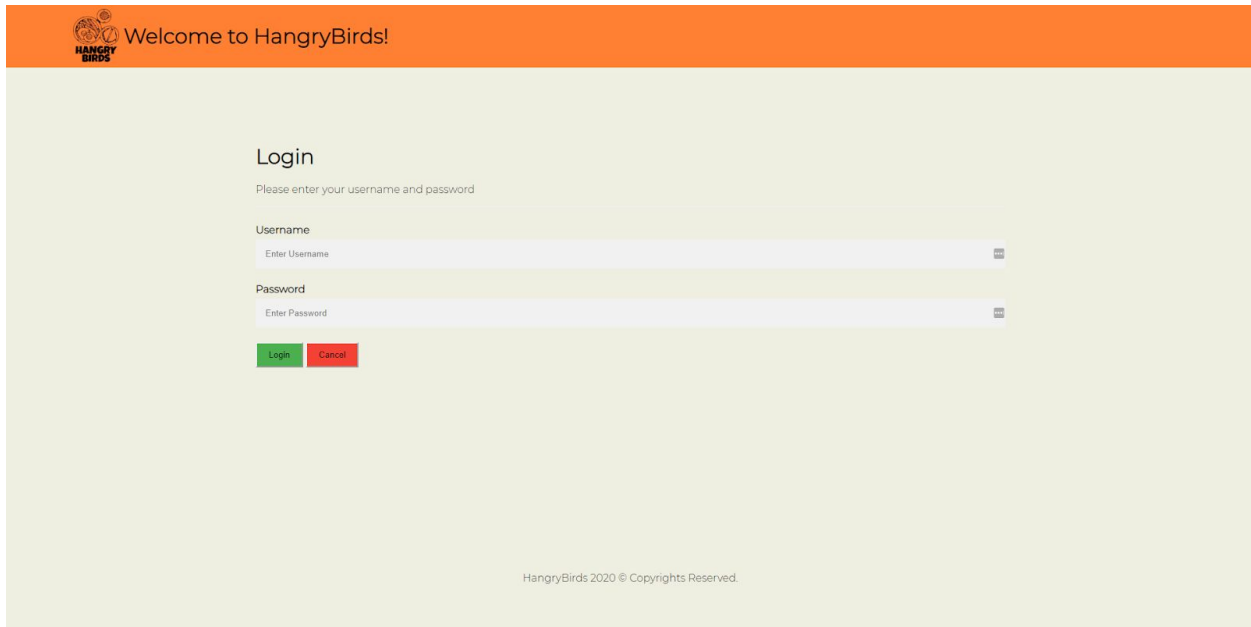


Sign Up Page: On this page, users can enter their credentials to sign up for an account.



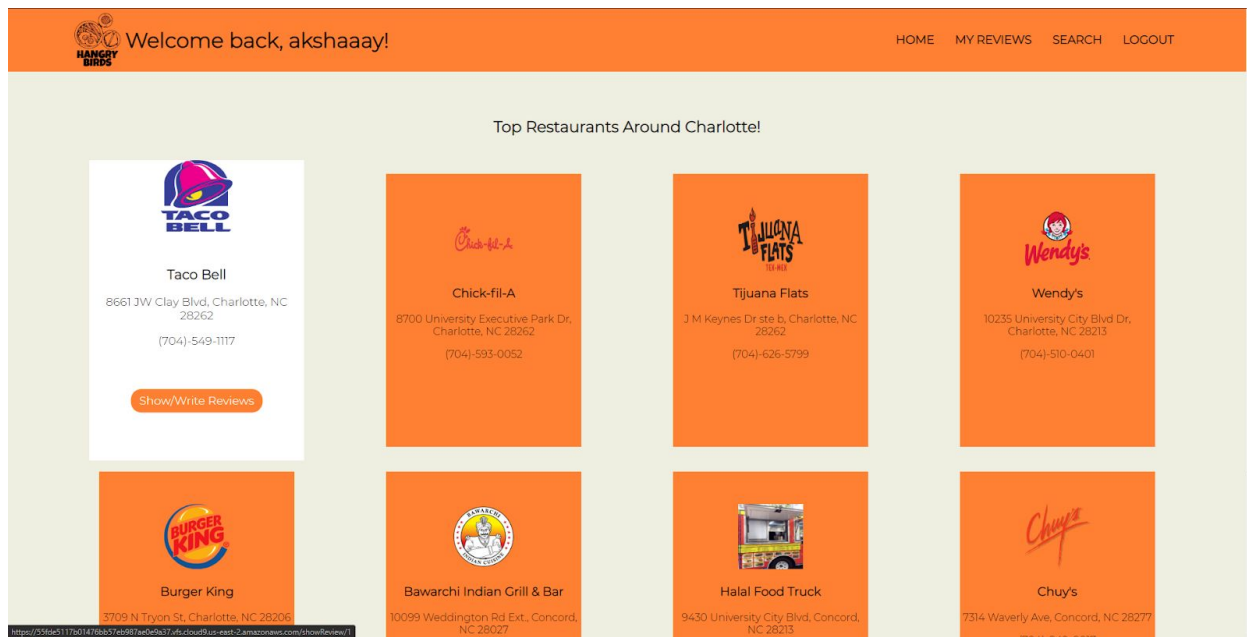
The screenshot shows the 'Sign Up' page for HangryBirds. At the top, there is an orange header with the HangryBirds logo and the text 'Welcome to HangryBirds!'. Below the header, the page has a light green background. The main heading is 'Sign Up', followed by the instruction 'Please fill in this form to create an account.' The form consists of four input fields: 'User Name' (placeholder: 'Enter Username'), 'Email' (placeholder: 'Enter Email'), 'Password' (placeholder: 'Enter Password'), and 'Repeat Password' (placeholder: 'Confirm Password'). Each field has a small icon on the right side. Below the fields, there is a line of text: 'By creating an account you agree to our [Terms & Privacy](#).' At the bottom of the form, there are two buttons: a red 'Cancel' button and a green 'Sign Up' button. At the very bottom of the page, there is a small copyright notice: 'HangryBirds 2020 © Copyrights Reserved.'

Login Page: On this page, users can enter their credentials to login into the application

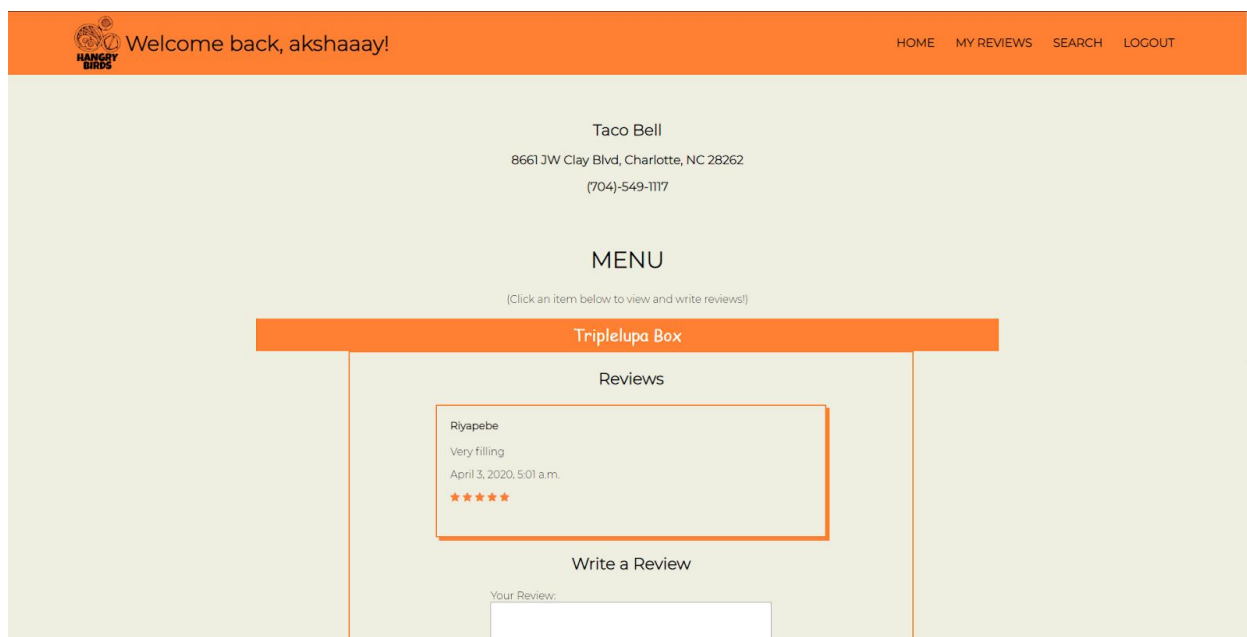


The screenshot shows the 'Login' page for HangryBirds. At the top, there is an orange header with the HangryBirds logo and the text 'Welcome to HangryBirds!'. Below the header, the page has a light green background. The main heading is 'Login', followed by the instruction 'Please enter your username and password.' The form consists of two input fields: 'Username' (placeholder: 'Enter Username') and 'Password' (placeholder: 'Enter Password'). Each field has a small icon on the right side. Below the fields, there are two buttons: a green 'Login' button and a red 'Cancel' button. At the very bottom of the page, there is a small copyright notice: 'HangryBirds 2020 © Copyrights Reserved.'

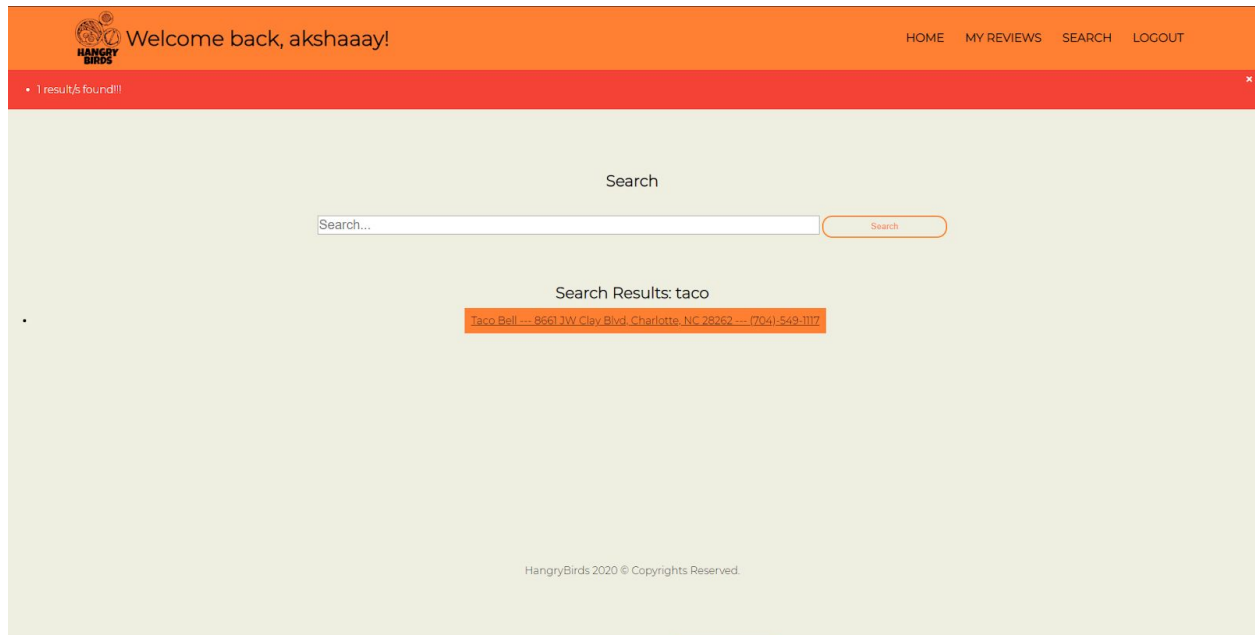
Welcome Page: On this page, users can see the restaurants around Charlotte



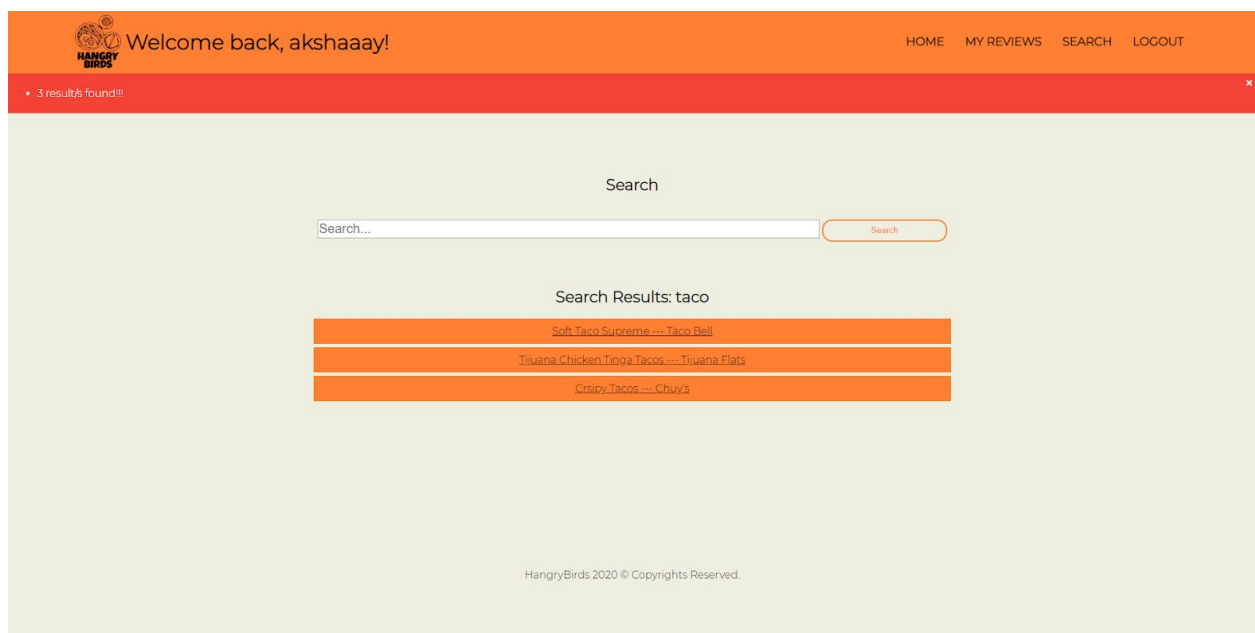
Looking at Reviews/Writing Reviews Page: On this page, users can enter their own reviews or view reviews done by others



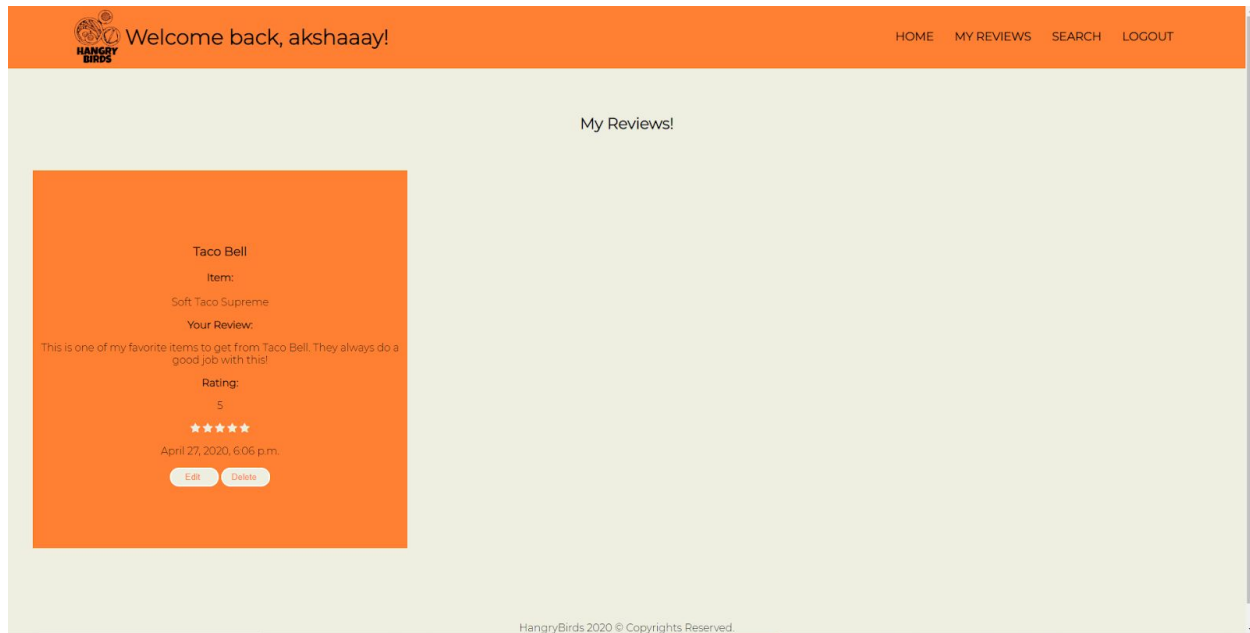
Search By Restaurant: On this page, users can search specially by a restaurant



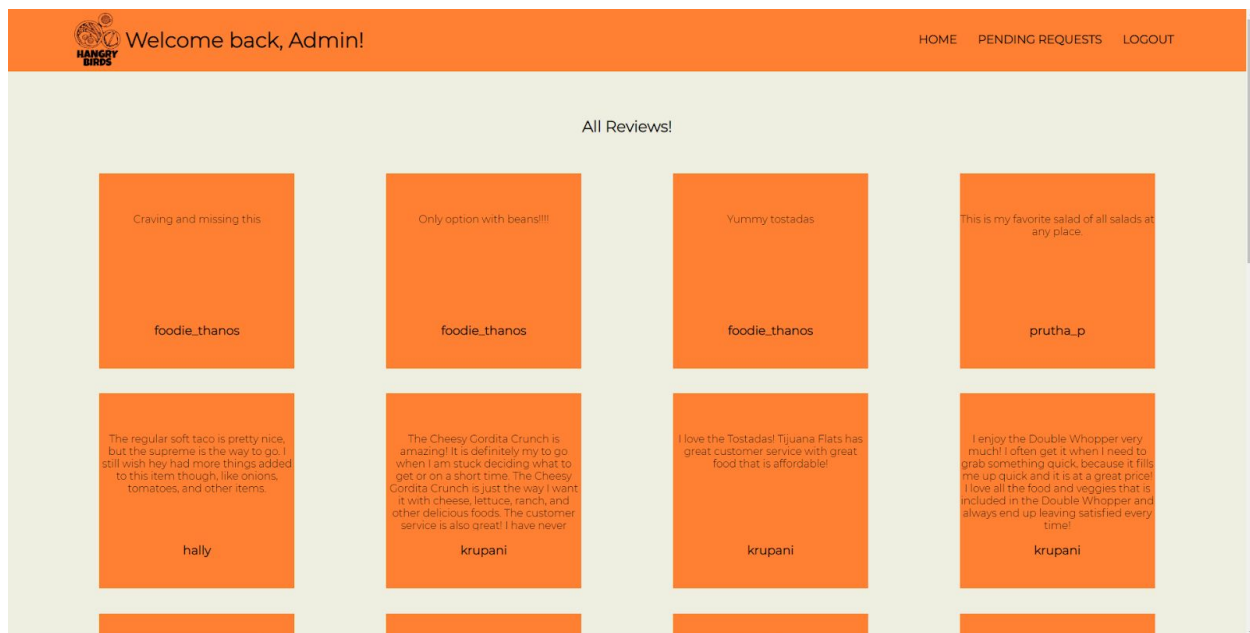
Search By Item: On this page, users can specially search by a food item to see the best restaurants/reviews for that item



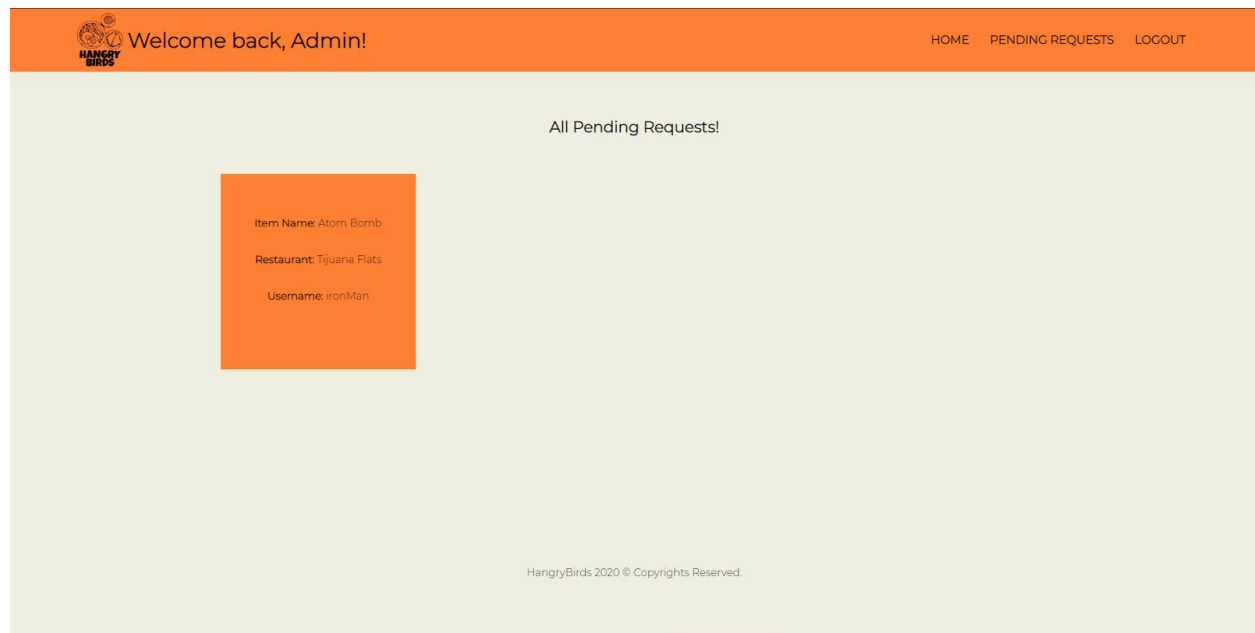
My Reviews Page: On this page, users can see/edit/delete the reviews that they have already submitted



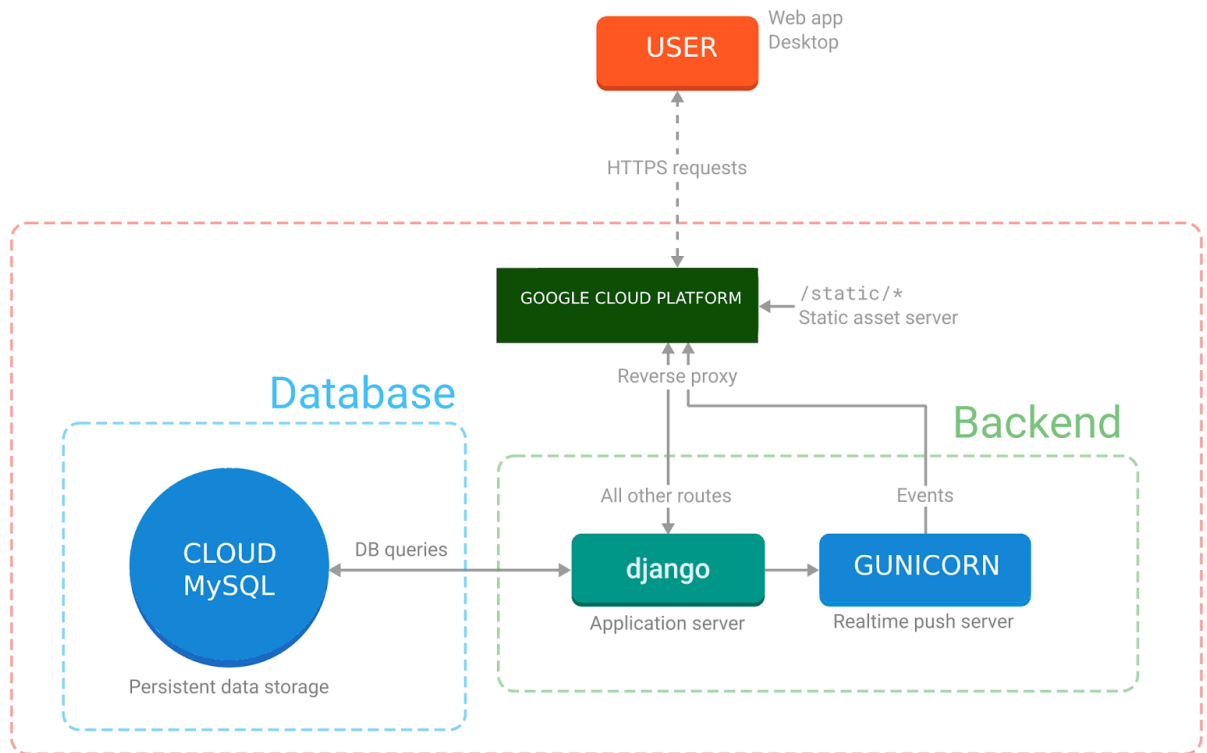
Admin Page: On this page, Admins can see internal information to the application and delete the scan or irrelevant reviews



Pending Item Approval Page: On this page, admins can see which items are still pending to be approved through the application

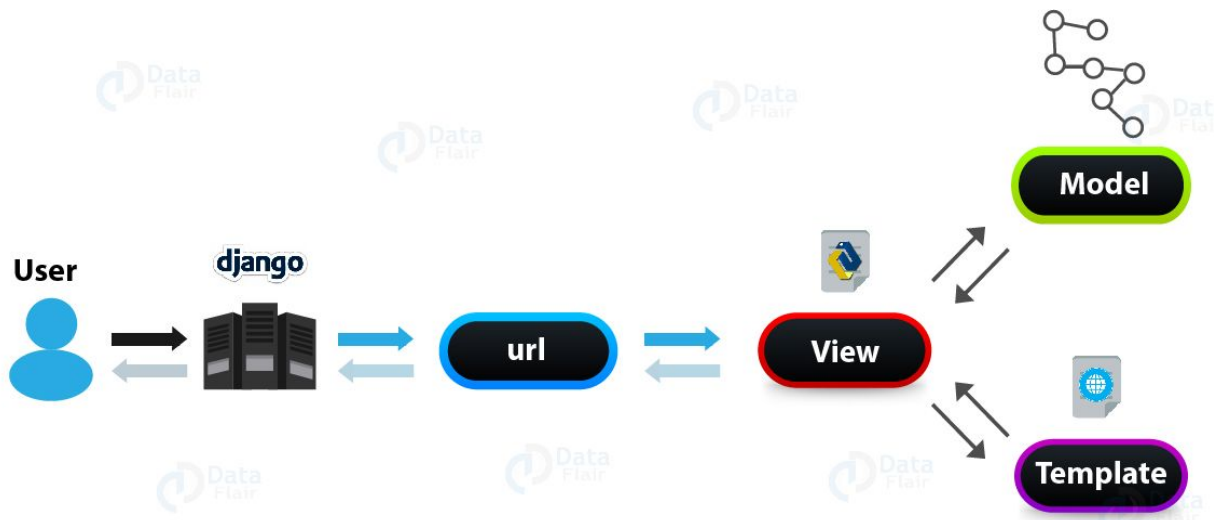


2. Architecture Overview



The architecture of the HangryBirds web application was chosen after conducting face-face team meetings, understanding the strength of each team members, and thorough research on different web application frameworks. After a lot of brainstorming, we finalized with a layered architectural design pattern. To support the backend of our application, we used the Django framework supported by Python. For our database needs, we utilized Cloud MySQL server. For the frontend of our application, we made the application using HTML, CSS, JavaScript and jQuery.

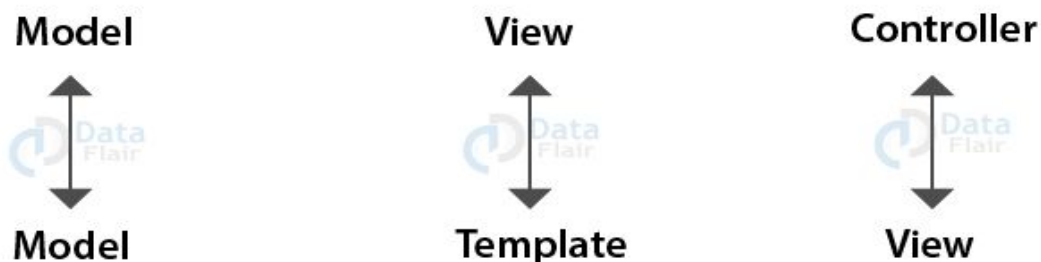
2.1 Subsystem Architecture



Our system heavily relies on MVC architectural patterns. All of the restaurant information which includes the restaurant name, location, menu, and contact information is stored in Google Cloud MySQL server. The HTML pages fetch all the information needed for the specific page from the server. For the view of the design we have HTML, CSS and JavaScript, for the backend we have Python acting as a controller, performing actions such as login check for the user, signing up user, passing data to and from HTML to successful store populate the database. Lastly we have database tables acting as Models for creating tables to store data.

Django is mainly an MTV (Model-Template-View) framework which is very similar to MVC. It uses the terminology Templates for Views and Views for Controller.

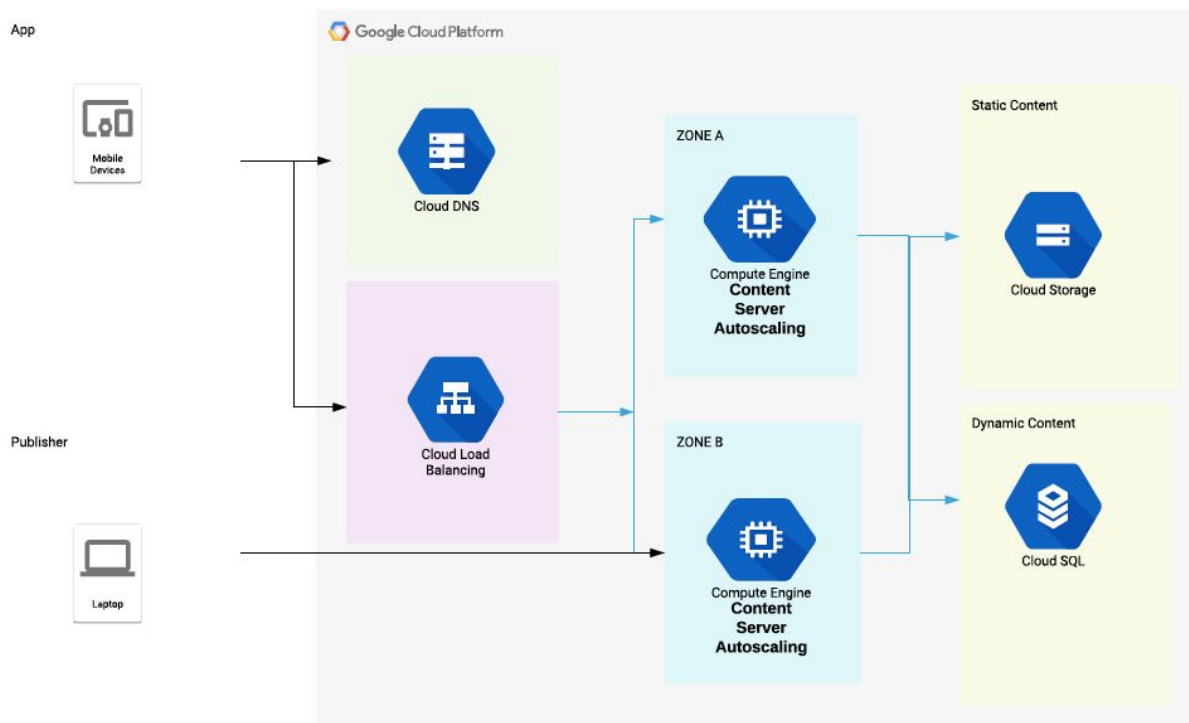
MTV instead of MVC django



2.2 Alternative Designs

At start we decided to move forward with C# language and Firebase as our deployment platform. After doing some research and doing a test run to deploy a simple program on the internet, we came to learn that firebase does not support C#. We were back to square one and started looking for other deployment platforms and language that is supported by the platform. After a few tries we were successful in deploying a simple program using the Django framework on Google Cloud Platform.

2.3 Deployment Architecture



For the deployment, as mentioned earlier, we used the Google Cloud Platform as it offered cloud storage for static contents, Cloud SQL for database support, and Gunicorn server for hosting the web application on the Google App Engine instance. All devices ranging from smartphones to laptops can easily access the website hosted through the google cloud services.

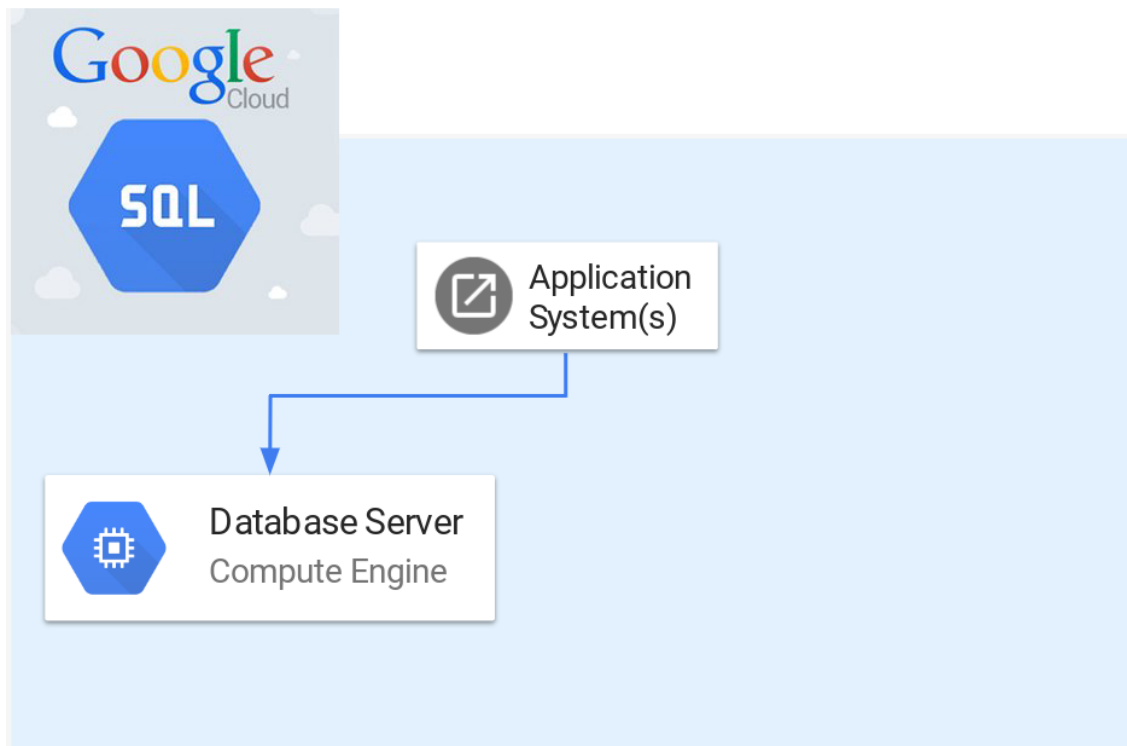
2.4 Persistent Data Storage

- User Data

To access the write and view reviews feature of our web application user is asked to sign up which asks for email, username, and password. It is required for us to store user details from the signup page so that the user doesn't have to sign up everytime he/she visits our website. Passwords are stored in encrypted form for user privacy.

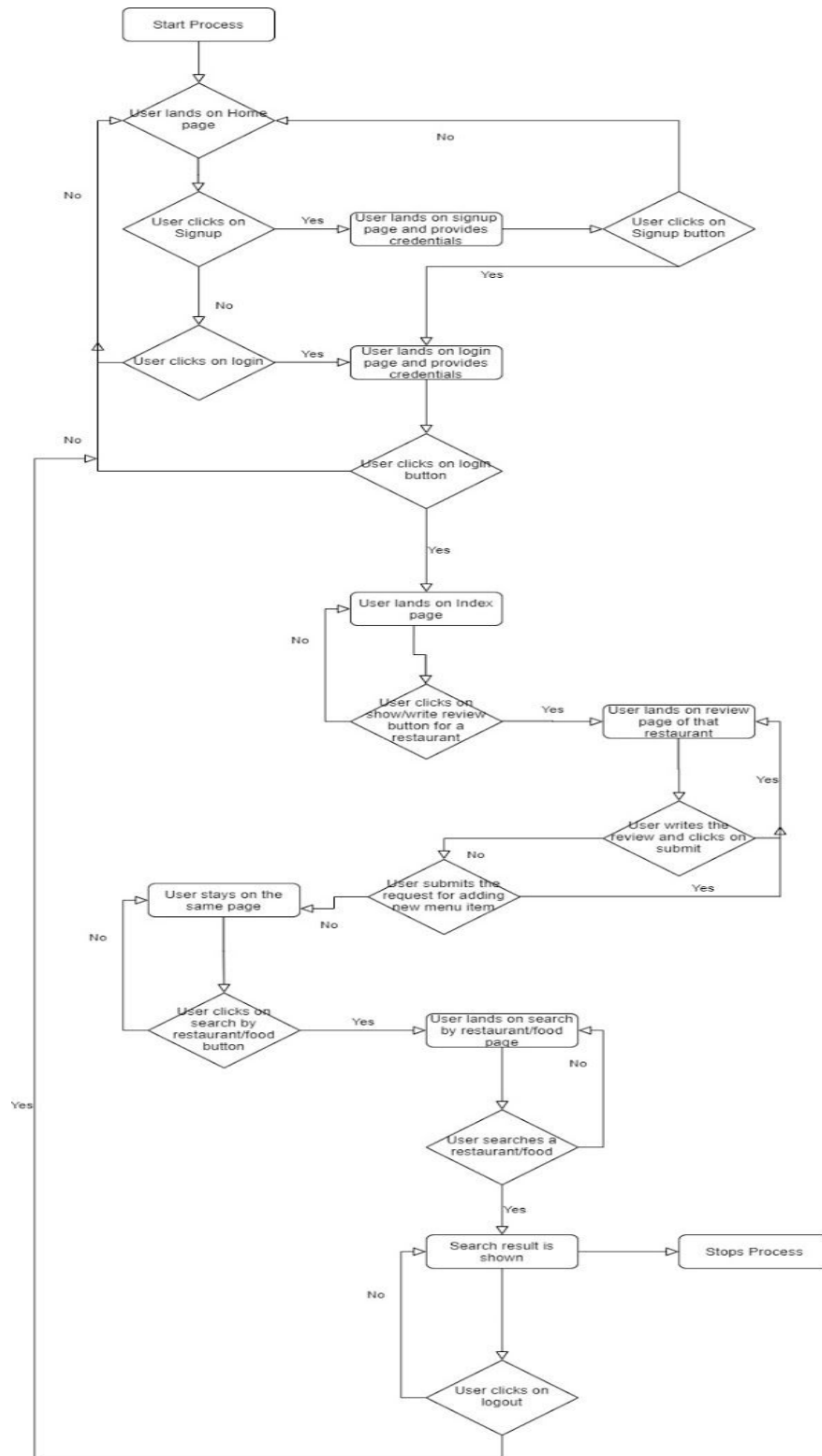
- Website Infrastructure Data

Currently the website is fully functional and is successfully deployed on Google Cloud Platform. All the data related to user authentication, reviews, restaurants, menus, requests, etc. are stored and regularly updated on the Google Cloud MySQL database.



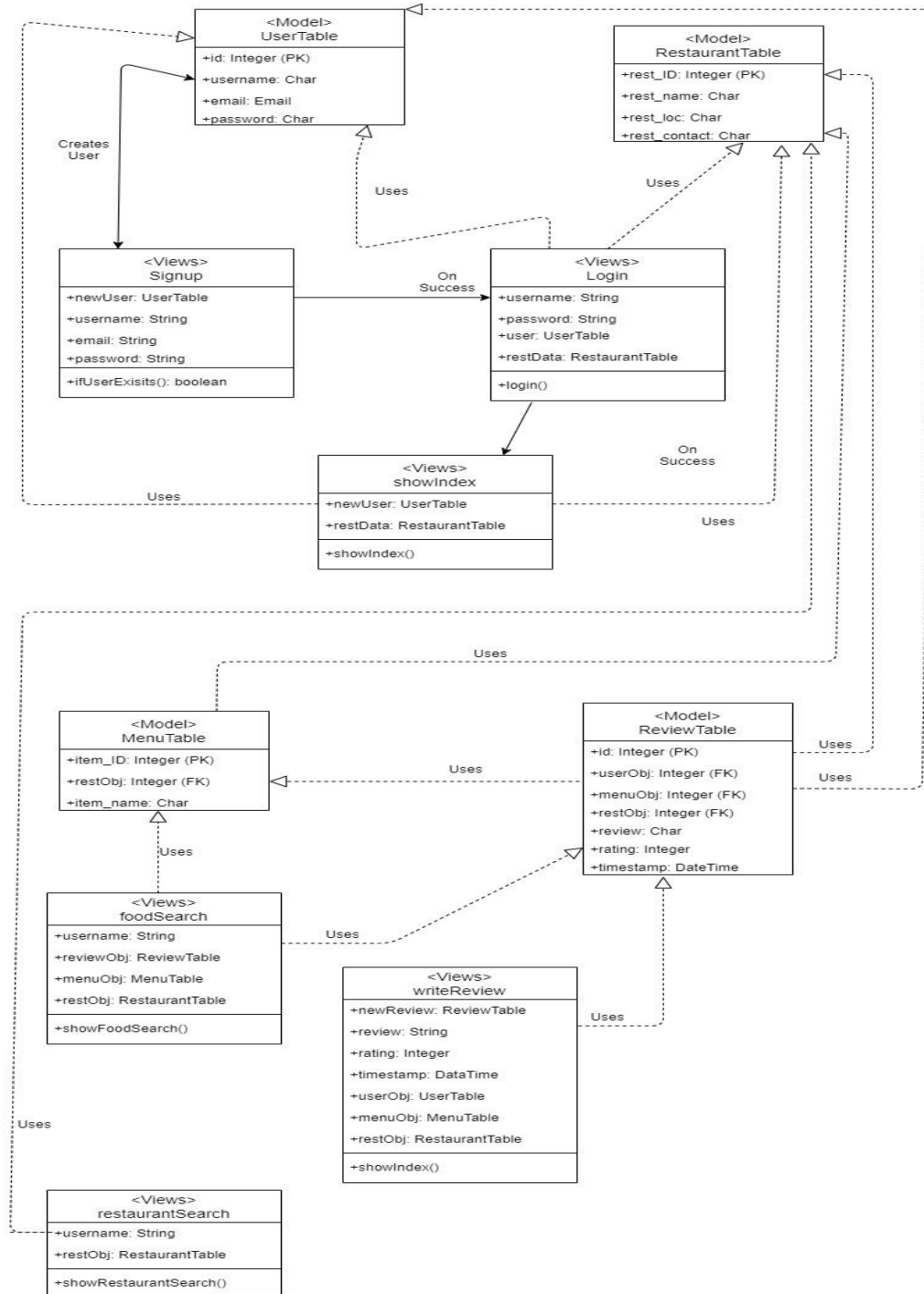
2.5 Global Control Flow

The Control Flow Diagram is shown below along with the different processes involved during the user requests sent to the web application server:-



3. Detailed System Design

3.1 Static View



The above UML diagram describes the way in which user signup, login, and write review occurs in the web application. For the signup process, the user fills the signup form and all that data is passed into the views where that data is saved in the UserTable of the models. Then comes the user login where again the user provides his/her credentials and that data is passed in the views where that information is matched against the existing entries of the UserTable and if a user exists, then he/she is allowed to login into the web application.

Finally, after logging in the web application (i.e. showIndex view), where a user writes the review and on submitting the review, the data is passed into the views where the information is stored with the ids of MenuTable, RestaurantTable, and UserTable into the ReviewTable, and the user is notified about the successful submission of the review.

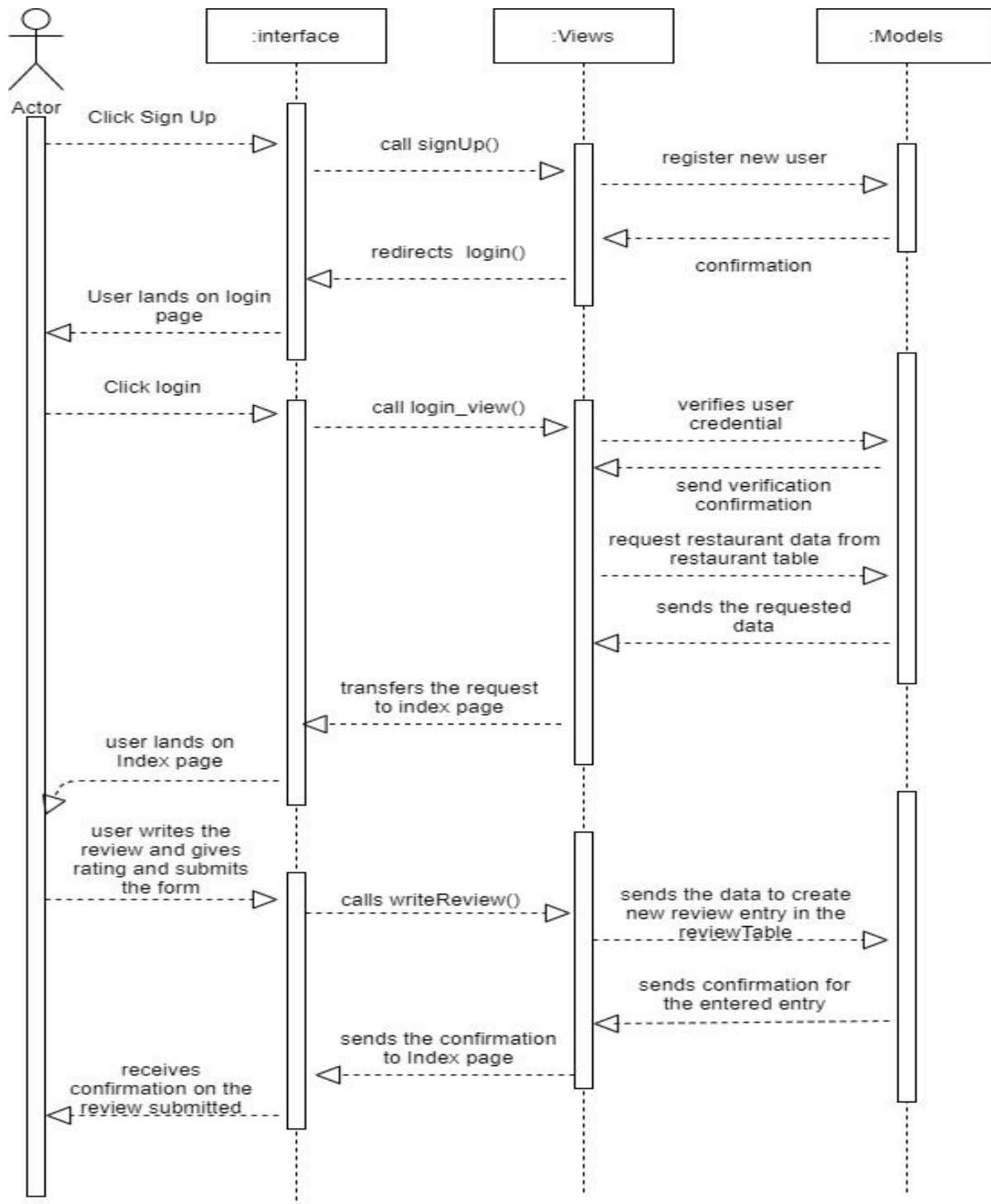
A brief description of some of the key view methods is described below:-

- **signup():** This method aims at the user creation through filling out the form which requires credentials such as unique username, email address, and password. Then after form submission the data is stored in the database for future login by that user.
- **login():** This method asks the user for login credentials and then checks if the user input data exists in the database. If the data exists the user is guided to the index page and the data does not match a user is asked to check and re-enter the credential.
- **showIndex():** This function assists the user towards navigating to the index page of the web application which is achieved only after successful login by the user. On the index page, the user can see different restaurants and then choose one of them so as to check out the menu and write/read a review.
- **writeReview():** This method gets the data that the user inputs which is a review of the item and rating out of 5 and stores into the database.
- **foodSearch():** It aims at searching the user input of the food item from the search bar against the database to find the closely related food items and

then display them on the page. This in turn helps the user to navigate the item across different restaurants and judge them based on the reviews by the other users.

- **restaurantSearch():** This method is called when a user does a search by restaurant to find a specific restaurant. This method displays all the restaurants that match with the user input.

3.2 Dynamic View



The above sequence diagrams provide the architectural flow of the logic and algorithm of the web application. Let's understand from the signup call which

is triggered when a user clicks on the signup button on the homepage of the website. That call is transferred to the views to call the signup function which checks for validation and then saves the data to the UserTable of the models. Then the confirmation is sent back to views that redirects the user to the login page.

Now comes, the login function which is triggered when the user provides the credentials and tries to sign-in on the login page. At that point, the data containing the credentials is passed into the views to check whether a user with such credentials exists in the UserTable or not. If present, then the models send a confirmation to the views to allow the user to login to the web application. Upon receiving the confirmation, the user is redirected to the index page through the views.

At last, comes the part of writing a review of the food item which was also the User Stories assigned to me in Sprint 2. After the user has successfully logged in, he/she can write a review about a particular food item served at a restaurant by clicking write a review section of that food item. On clicking the submit button, the review data which consists of review, rating, user-info, and a timestamp is transferred to the views where the writeReview() function saves that data in the ReviewTable of the models. A confirmation of the successful entry of the review is sent from models to the views which in turn informs the user about the successful submission of the review. This is the working mechanism of the web application.