System Design Document

Project: Kiwifeeds

By: Kiwitato

# Architecture

*An explanation for the Django framework.*

Django is a high-level web framework written in Python has a powerful inbuild ORM (Object Relational Mapping) that allow us to interact with database using python object instead of SQL Queries.

Django works on MVT (Model-View-Template) Architecture where Models represent the data and database Schema, Views handle the presentation logic and template manage the front-end user.

Django provides a URL routing system that allows developers to map URLs to view functions. This makes it easy to define and organize the application's URLs.

## Web Framework

#### User Interface (responsive for Mobile, Tablet and desktop View):

This is the front end of the application where user can browse search for restaurants.

View menu, location, rating and perform action to save as favourite, order and pay

#### Client-side Components:

1. User Authentication:

These components handle user registration, login and authentication.

1. Search Interface:

Where user can view restaurant listing, search query and filter result.

1. User profile:

User can create and mange their profile and add delivery address, contact detail, payment method, favorite restaurants.

#### Server-side Components:

1. Application Server:

Core application to handle user requests and communicate with database, manages business logic and maintain flow of data between various components.

1. Database:

To store data related to restaurants, user, order, review etc. we will use default SQLite database.

1. Media:

To store all images related to restaurants, user, order, review etc.

1. Order Processing:

Manage order Lifecycle including order placement, payment processing, order Tracking and send notification to user and restaurant.

1. Payment Gateway:

To securely handle payment transaction and validation.

1. Geolocation Services:

Providing geolocation services for search nearby restaurants, location order and provide estimate delivery time.

#### Admin Dashboard:

Provide Restaurant owner and admin to create, edit and delete restaurants listing, menus, reviews, and orders.

# Data Access Design

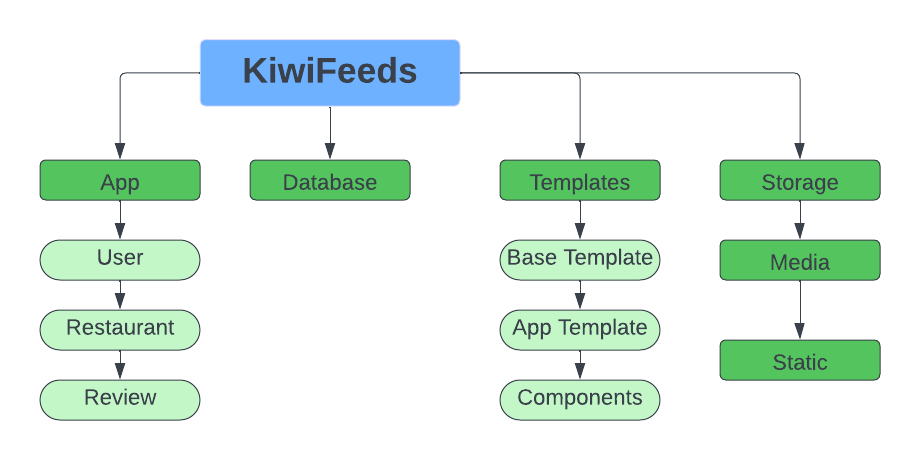
## Model Design (Data Model)

1. User
   1. Attributes:
      1. User ID (PK)
      2. First Name
      3. Last Name
      4. Email
      5. Phone Number
      6. Address
2. Restaurant
   1. Attributes:
      1. Restaurant ID (PK)
      2. Name
      3. Description
      4. Location
      5. User
      6. Menu items
   2. Relationships:
      1. Menu Items one to many
      2. User many to one
3. Menu Item
   1. Attributes
      1. Menu Item ID(PK)
      2. Name
      3. Description
      4. Price
      5. Restaurant
   2. Relationships
      1. Restaurant (many to one)
4. Review
   1. Attributes
      1. Review ID (PK)
      2. User (FK)
      3. Restaurant (FK)
      4. Rating
      5. Comment
   2. Relationships
      1. User many to one
      2. Restaurant many to one
5. Favourite Restaurant
   1. Attributes
      1. User (FK)
      2. Restaurants (FK)

## Functional Decomposition

1. User
   1. Creation of new user i.e., signup
   2. Login
   3. Edit
   4. See favourites
2. Restaurant
   1. Creation(admin)
   2. Calculate rating
   3. Leave review(user)
   4. Add owner(admin)
   5. Add food(owner)
   6. Add to favourites(user)
3. Menu Item
   1. Create(owner)
4. Review
   1. Create(user)

## Folder Structure



# Security Design

## Framework Security

Django comes with many built-in security features.

* *Cross site scripting (XSS) protection*
* *Cross site request forgery (CSRF) protection*
* *SQL injection protection*
* *Clickjacking protection*
* *SSL/HTTPS*
* *Session security*
* *Validation of User-uploaded content*
* *Authentication and Authorization*
* *Encryption (PBKDF2 algorithm with a SHA256 hash)*

## Security Mechanisms

#### Cross site scripting (XSS) protection

* Django’s template system automatically escapes the output of every variable tag.
* Do not turn auto escape off.

#### Cross site request forgery (CSRF) protection

* Using CSRF tokens in our html forms
* Not turning CSRF off.

#### SQL injection protection

* The use of query parameterization that divides the SQL queries from the query's parameters.

#### Clickjacking protection

* Using X-Frame-Options middleware

#### SSL/HTTPS

* Implement SECURE\_PROXY\_SSL\_HEADER, and SECURE\_SSL\_REDIRECT to True, SESSION\_COOKIE\_SECURE¶ and CSRF\_Cookie\_Secure to True
* Using HTTP Strict Transport Security (HSTS) ensures any requests always directs to HTTPS.

#### Session security

* Providing a notification to user that session will expire when there is inactivity using Warn\_After.
* Automatically logging out user if session has been inactive using Expire\_After

#### Validation of User-uploaded content

* By serving user uploaded content from a distinct top-level or second-level domain ie. a distinct, unique and specific domain name
* By serving static files from a cloud service provider

#### Authentication and Authorization

* Having different user groups with different permissions set

#### Encryption (PBKDF2 algorithm with a SHA256 hash)

* Use the make\_password()function which can be imported from django.contrib.auth.hashers module, which converts a plain text password into a hash that can be stored into a database
* Add the Password Hashers into settings.py file
* Compare the hash password with the plaintext password to check if they are equivalent, using the function check\_password imported from django.contrib.auth.hashers

## User Roles and Permissions

Site Administration

* Remove Reviews
* Delete Restaurant View
* Edit Restaurant and/or Diner Details
* Assign users to groups
* Assign necessary permissions according to group

Restaurant User

* Create Restaurant View
* Edit Own Restaurant View
* Report Review
* Reply to Review
* Request Deletion of Restaurant
* View List of Diners who have purchased from or rated the restaurant

Review User

* Edit User Profile and Password
* Save favourite restaurants
* Search/ View Reviews
* Search/ View Restaurants
* Create and Publish Review
* View Menu

Unauthenticated User

* Search/ View Restaurants
* View Menu
* Search/ View Reviews
* Create a Login account

## Privacy Policy

Our Privacy Policy is as follows:

We collect personal information from you, including information about your:

name

contact information

location

interactions with us

billing or purchase information

We collect your personal information in order to:

allow us to better understand our audience

display relevant information

allow users identifying ways of interacting with the system

Providing some information is optional. If you choose not to enter location or adress, we'll be unable to display targeted local resturants.

If you choose not to enter phone number we'll be unable to allow restaurants to contact customers about reviews with customers permission

We keep your information safe by using CSRF tokens, PBKDF2 algorithm with a SHA256 hash for encryption and only allowing certain staff full access .

You have the right to ask for a copy of any personal information we hold about you, and to ask for it to be corrected if you think it is wrong. If you’d like to ask for a copy of your information, or to have it corrected, please contact us at kiwifeeds@gmail.com.

# User Interface Design

## View Design (UI List)

### User Story 1

I’d like to see a link next to each restaurant entry in the restaurant catalog, so that I can quickly add my review for that establishment.

List of interfaces:

* Homepage
* Restaurant catalog
* Restaurant details page

### User Story 2

I’d like to have a list of restaurants on my account I'd like to go to and be able to remove those after I visit them in person.

List of interfaces:

* Sign Up/Login
* Restaurant details
* Favorites list

## UI Design

UI design consists of Elements, Wireframes and Mockups.

*Link to Figma:*

<https://www.figma.com/file/HpQMIJMDaASN0TKu8NuBXx/Wireframes?type=design&nodeid=1%3A3&mode=design&t=8D28S5J8LCpnRTyu-1>

## Elements

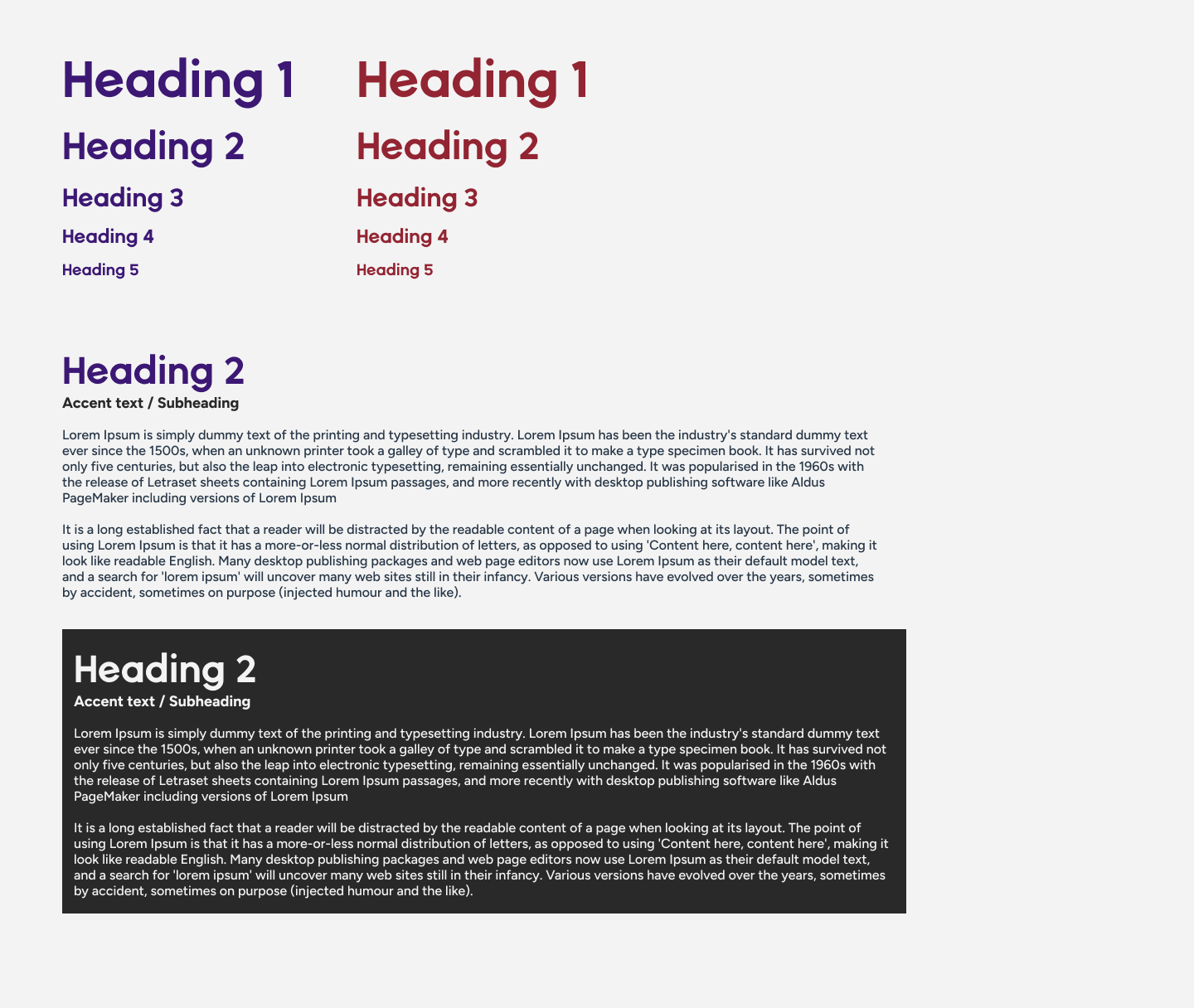
A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

Top left, clockwise: Logo, Buttons Dark, Buttons Light, Colour Palette

A screenshot of a computer

Description automatically generated

From left: Text, Elements

## Wireframes

A white background with pink lines

Description automatically generated

From left: Homepage, Restaurants

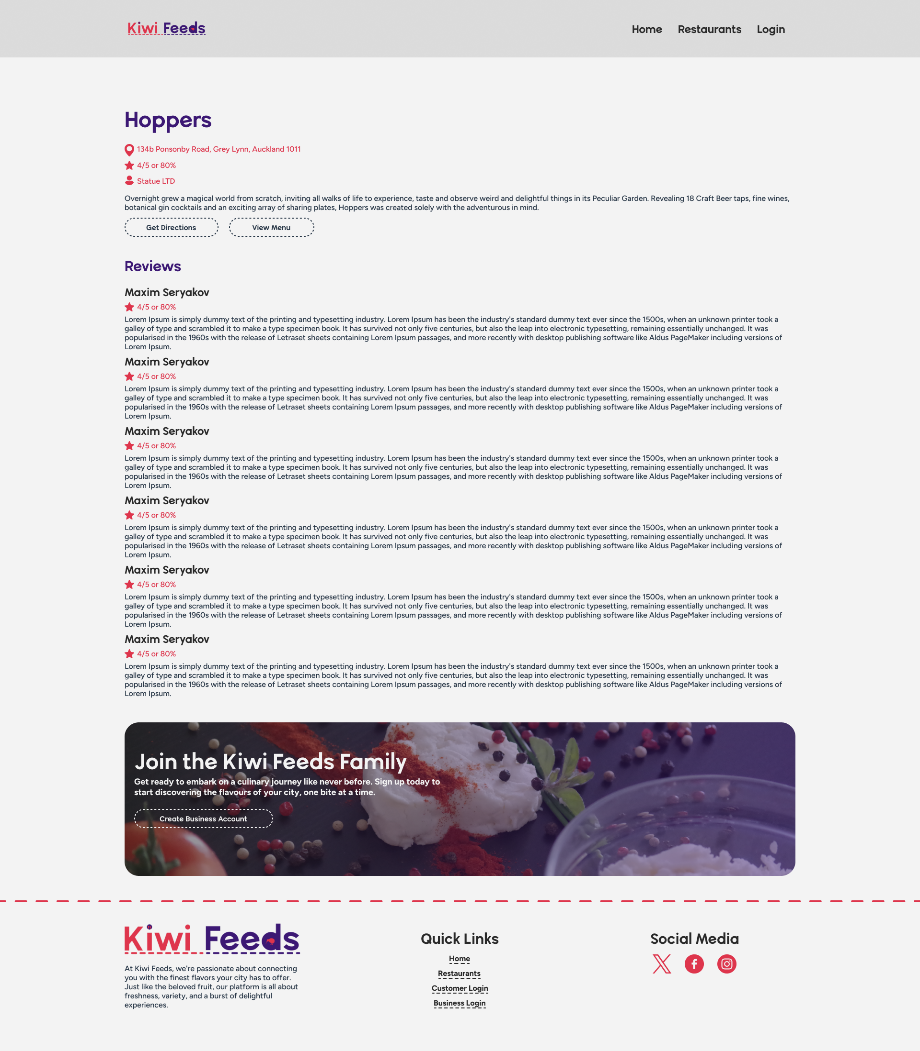
A screenshot of a computer

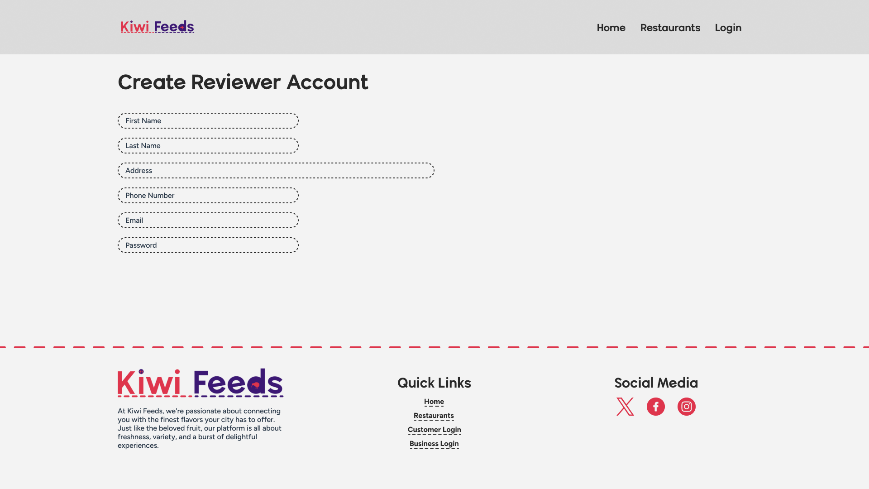
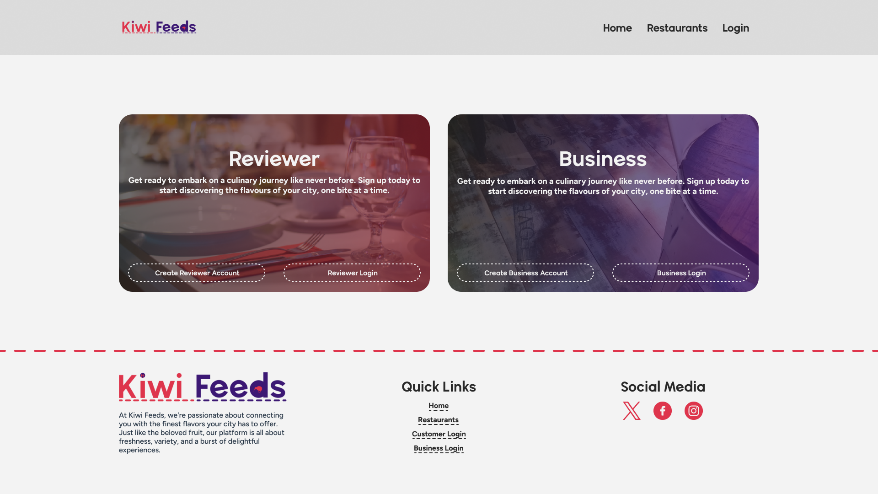
Description automatically generated

From left: Resturant Page, Sign in Choice Screen

## Mockups

From left: Homepage, Restaurant List





Clockwise from left: Restaurant Page, Sign Up page, Sign up account type choice page.