Software Design Document

For

Online Food Delivery System

Version 1.0 approved

Prepared by Bit Rebels

B Tech
[Computer
Science Engg]

27-Oct-2023

Revision History

Name	Date	Reason For Changes	Version
Bit Rebels	20/10/23	*initial draft*	*1.0 draft 1*
Bit Rebels	27/10/23	*baseline following changes after inspection*	*1.0 approved*

Table of Contents

1. Introduction

1.1 Purpose

The purpose of the SDD is to serve as a comprehensive guide for the development team, stakeholders, and any involved parties, outlining the design and functionality of the online food delivery system. It aims to:

Clearly define the structure and behavior of the system, facilitating a shared understanding among the development team.

Provide a blueprint that can be used as a reference throughout the development and implementation phases.

Ensure that the development process aligns with the business objectives and requirements of the online food delivery service.

Serve as a foundation for future updates, enhancements, and maintenance by providing a clear documentation of the system's architecture, components, and design decisions.

1.2 Scope

The scope of the SDD outlines the boundaries and limitations of the online food delivery system, encompassing the functionalities and features that will be included in the system. This may involve:

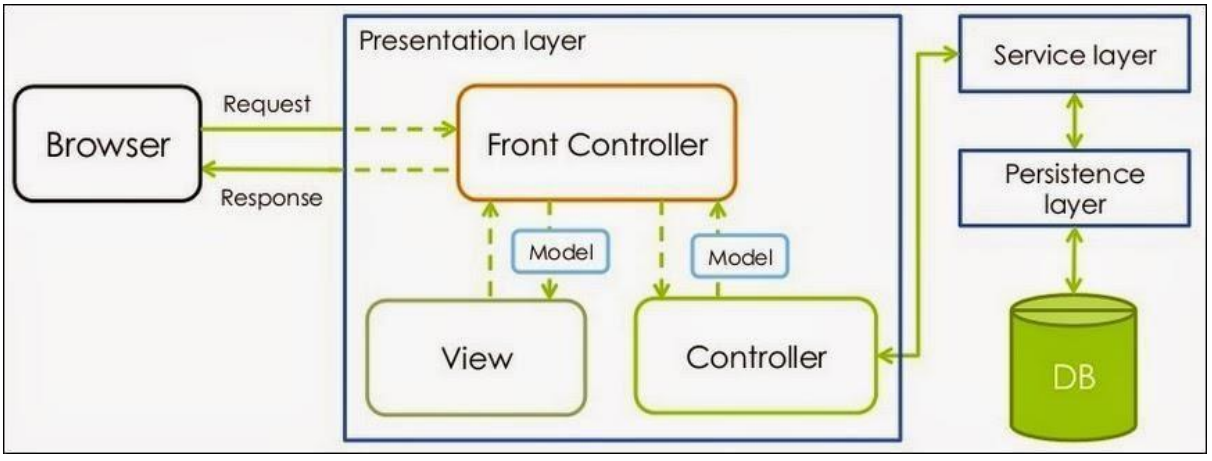
Defining the specific features and capabilities of the online platform, including user registration, restaurant management, menu display, order placement, payment processing, and delivery tracking.

Establishing the target audience and user base for the online food delivery system, such as end-users, restaurants, and delivery personnel.

Outlining any constraints or limitations, such as technology requirements, platform compatibility, or regulatory compliance, that may impact the development and deployment of the system.

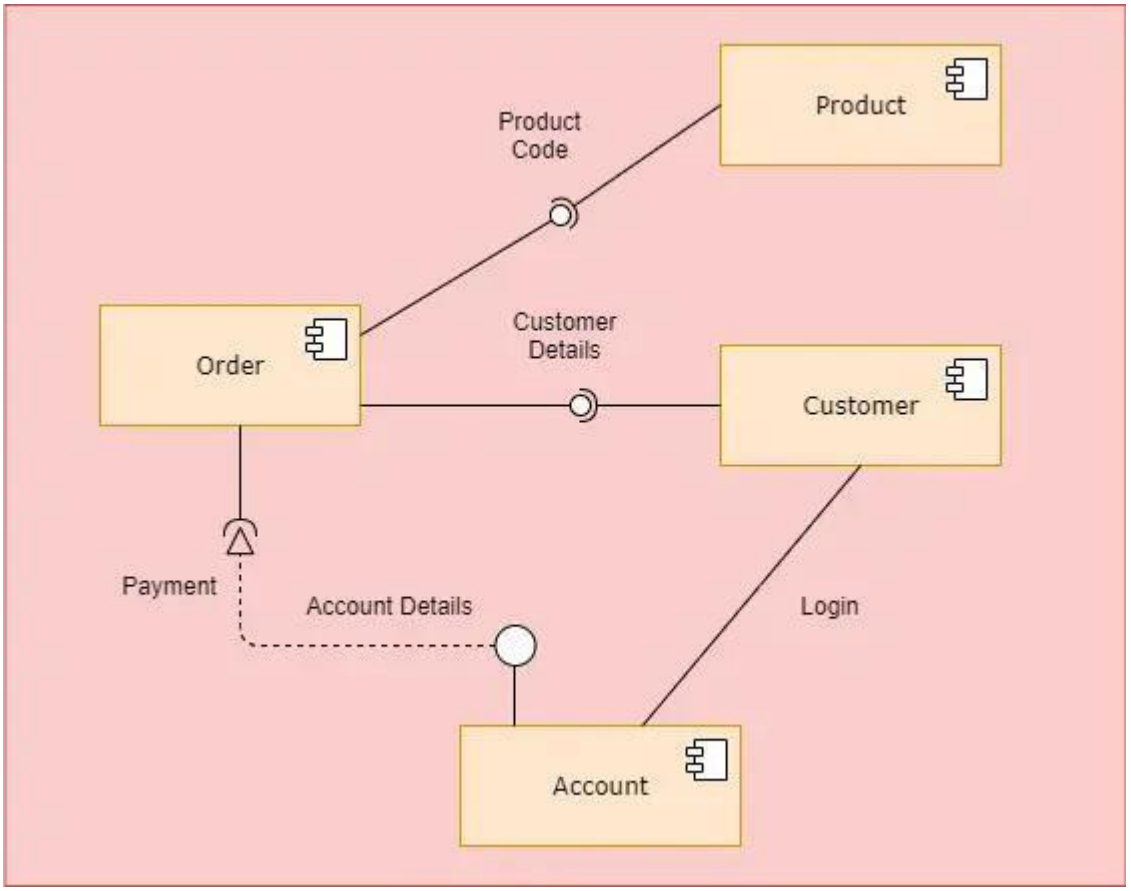
Clarifying the specific deliverables and outcomes that are expected from the development team within the defined timeframe and budget.

2. System Overview



The users will browse and search through the app. They'll scroll through the feed and find restaurants of their choice. They'll browse through the list of cuisine and add their preferences to the cart. Customers can select the orders and proceed to checkout. After finalizing, they'll check out using multiple payment methods. Once the restaurant accepts the order and starts preparing it, customers can track their orders in the app in real-time. The last step is when customers get their food delivered. This is the basic flow of a food delivery app.

2.1 Activity Diagram



1. All requests made from a mobile app or UI will go to different services via the API gateway. API gateway will take care of load balancing and routing requests to services. This will authenticate and authorize the user and send back the token ID. This token is used for further communication

2. Different services like, user registration and management service, order service, payment service will use

   transactional databases. We will use the Amazon Aurora relational database. This is a highly scalable database service to manage users and concurrent orders etc.

3. Information about different restaurants, their menu, price, offers, etc will be stored in JSON document storage in ElasticSearch. We can use a multi-node cluster here. Whenever a customer searches for a menu/cuisines it will be fetched from elastic search. Elastic search provides fast scalable search options

4. Once the user selects the dishes and quantity from the restaurant. He will go to the checkout option and then do payment. Different payment gateways and payment options are integration with the system and upon successful payments, the order is successfully placed

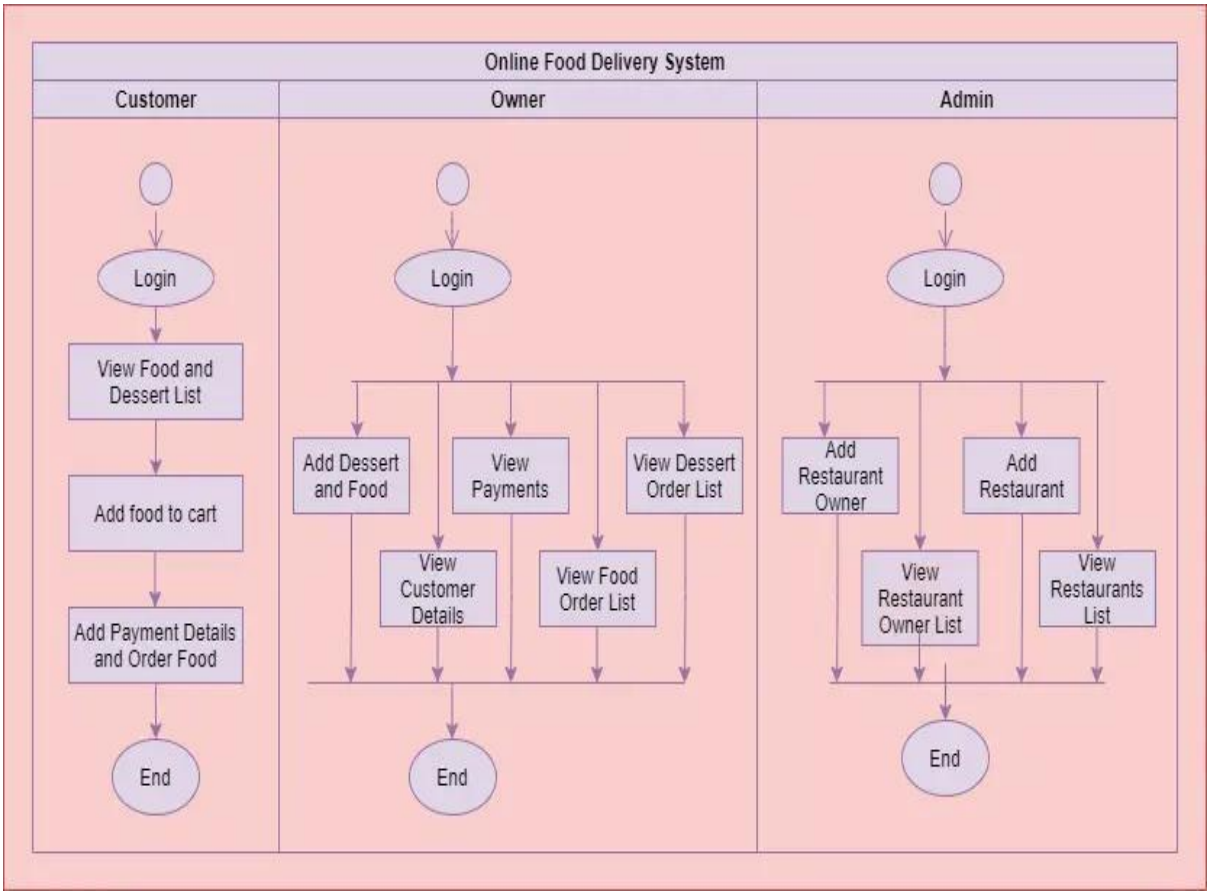
5. Once the order is placed all the information is sent to the central message Queue like Kafka. The order processing unit reads the order info and then notifies the selected restaurant about the order. At the same time, it searches for available delivery partners to nearby locations to pick up the order. It also gets the information like preparation time from the restaurant and estimated pickup time from the delivery partner based on his location and other details. it will select the best available delivery partner and he is notified about order and restaurant details

6. The user gets push notification about the order. The order processing and tracking service will work together and the user can track their order status, live location of the delivery person, etc

7. Delivery person pickup order and deliver to customers. Customer is real-time notified with ETA for the order

3. System Components

3.1 Decomposition Description



From the Admin panel, the admin can add the restaurant. Admin adds details like restaurant name, city, address, postal code, cuisine type, operational hours, owner details, payment shares, etc. All this information is stored in a relational database. We use Amazon Aurora here.

Once the restaurant is added we will generate a Unique ID for the restaurant. This unique ID will be used in Elastic search to store information like different menus, their price, preparation time, etc.

Restaurants have access to add /update/delete menu, price, preparation time, etc

So when a customer searches food options by dish name, restaurant name, location then ElasticSearch is queried. Elasticsearch is a highly available, scalable open-source full-text search and analytics engine. With elastic search, you can store, analyze, search large volumes of data quickly and in near realtime.

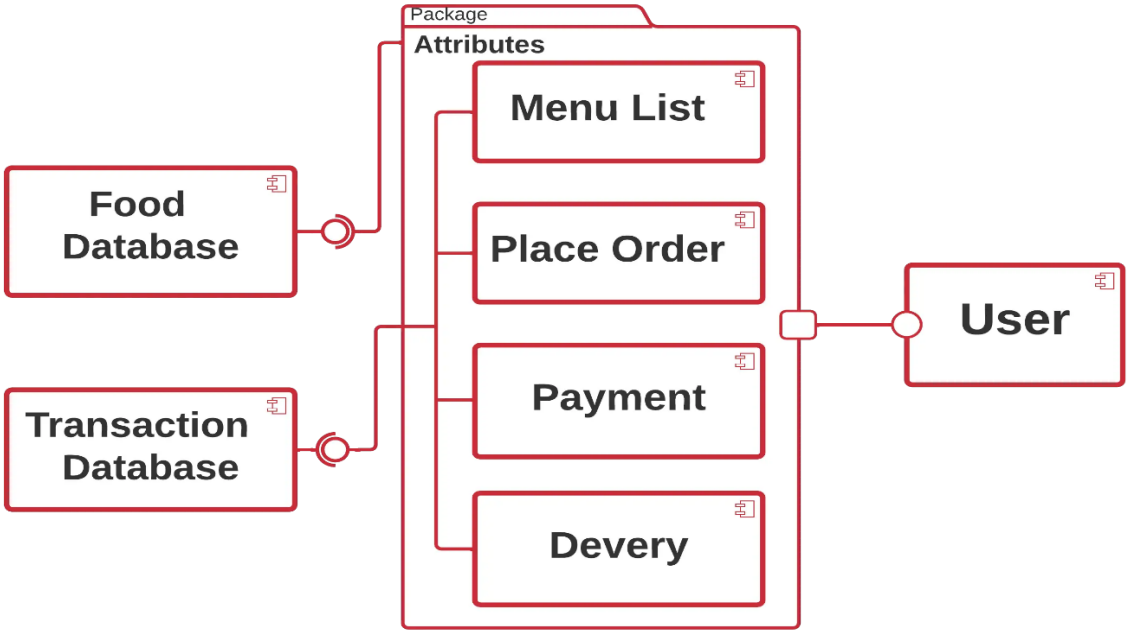
When a customer opens the app, the first call is made to the Inventory/Menu system to figure out:

1. Nearby serviceable restaurants to customer locations. This can be done by customer location and restaurant location.

2. The restaurants that are actually serviceable to the user/customer, those restaurants which can deliver food within some stipulated time (say 45 minutes)

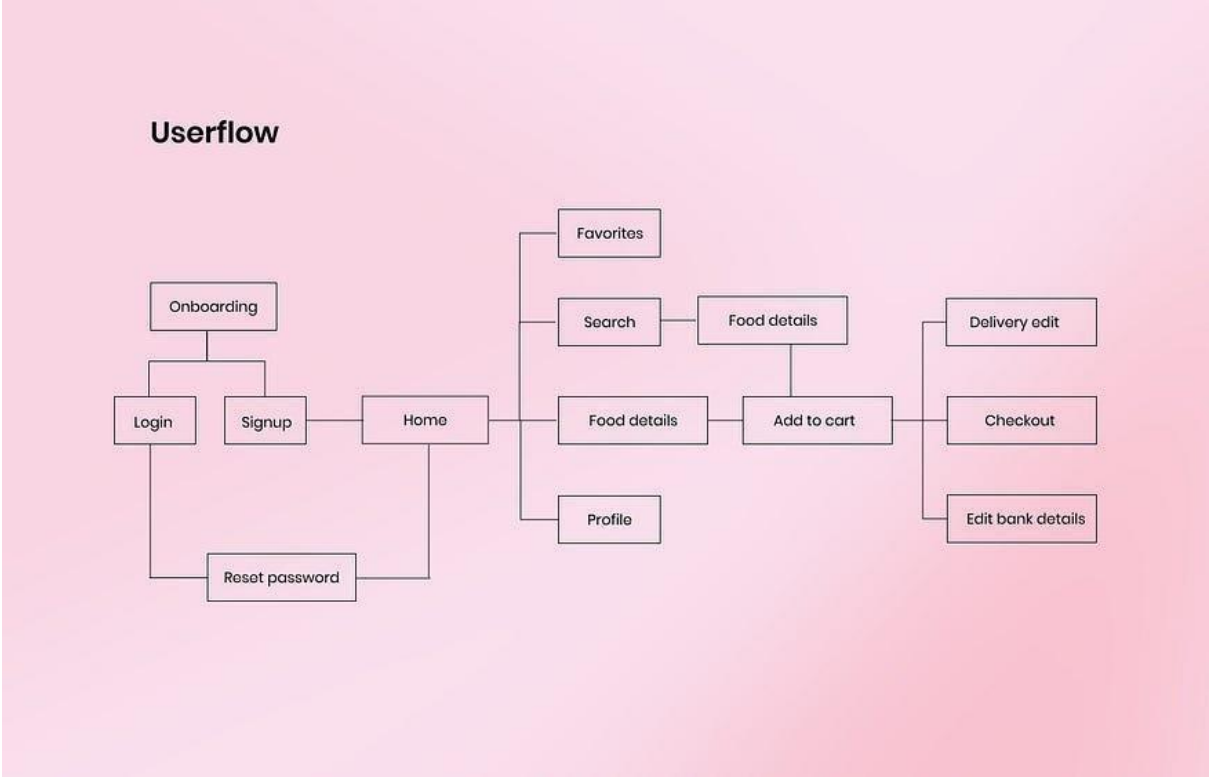
3. The expected delivery time for your food order from a potential restaurant.

3.2 Interface Description

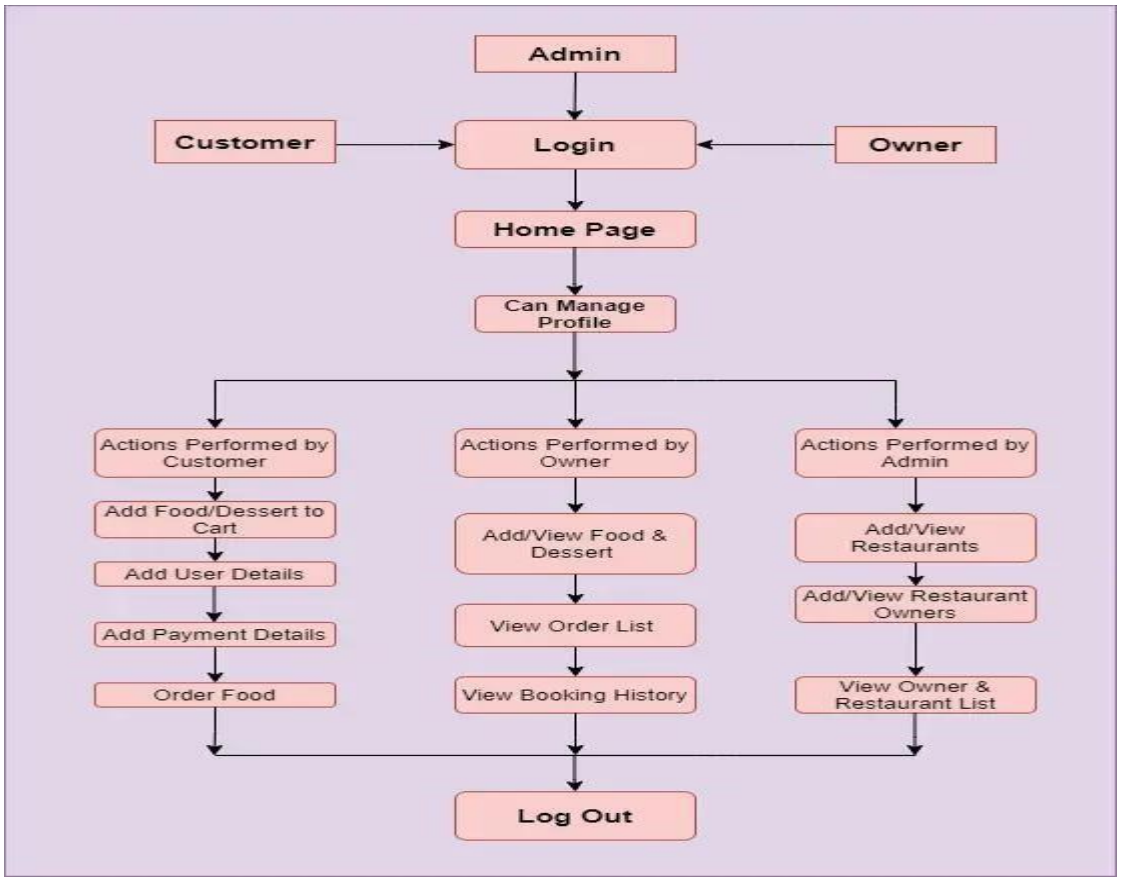


3.3 User flow

Here is a portion of the whole user flow showing the decisions that the user and system can make in the process of browsing through dishes.

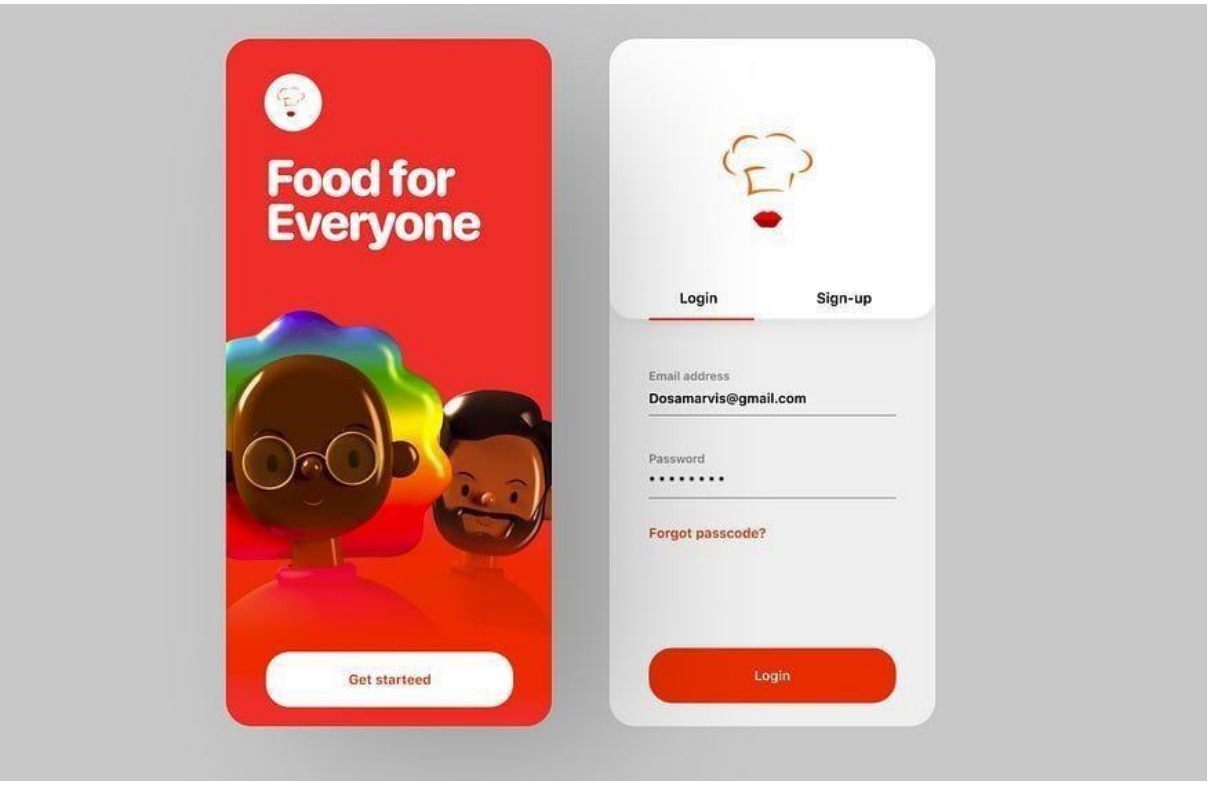

Userflow
Onboarding
Login
Signup
Home
Favorites
Search
Food details
Profile
Food details
Add to cart
Delivery edit
Checkout
Edit bank details
Reset password

DFD Diagram



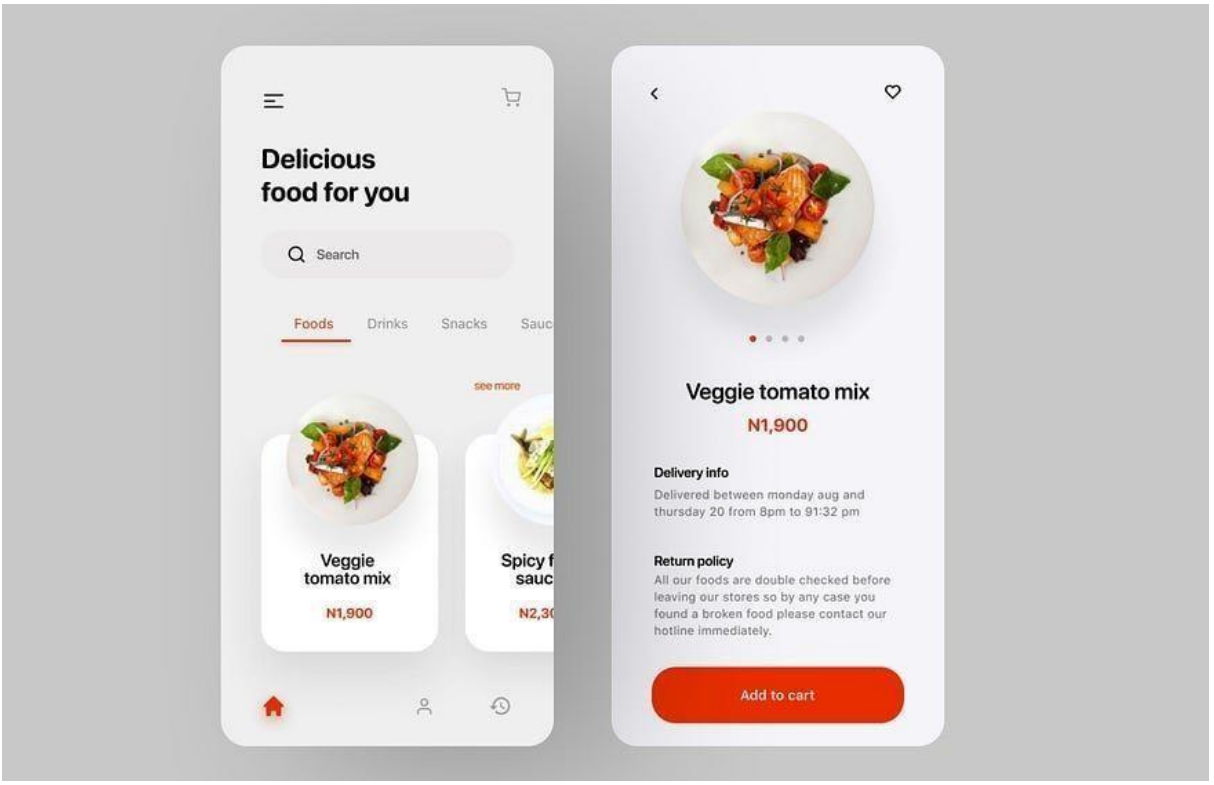
4. Design and implementation

Splash screen and Login screen making the login, sign up screen quick to use and less clickable.

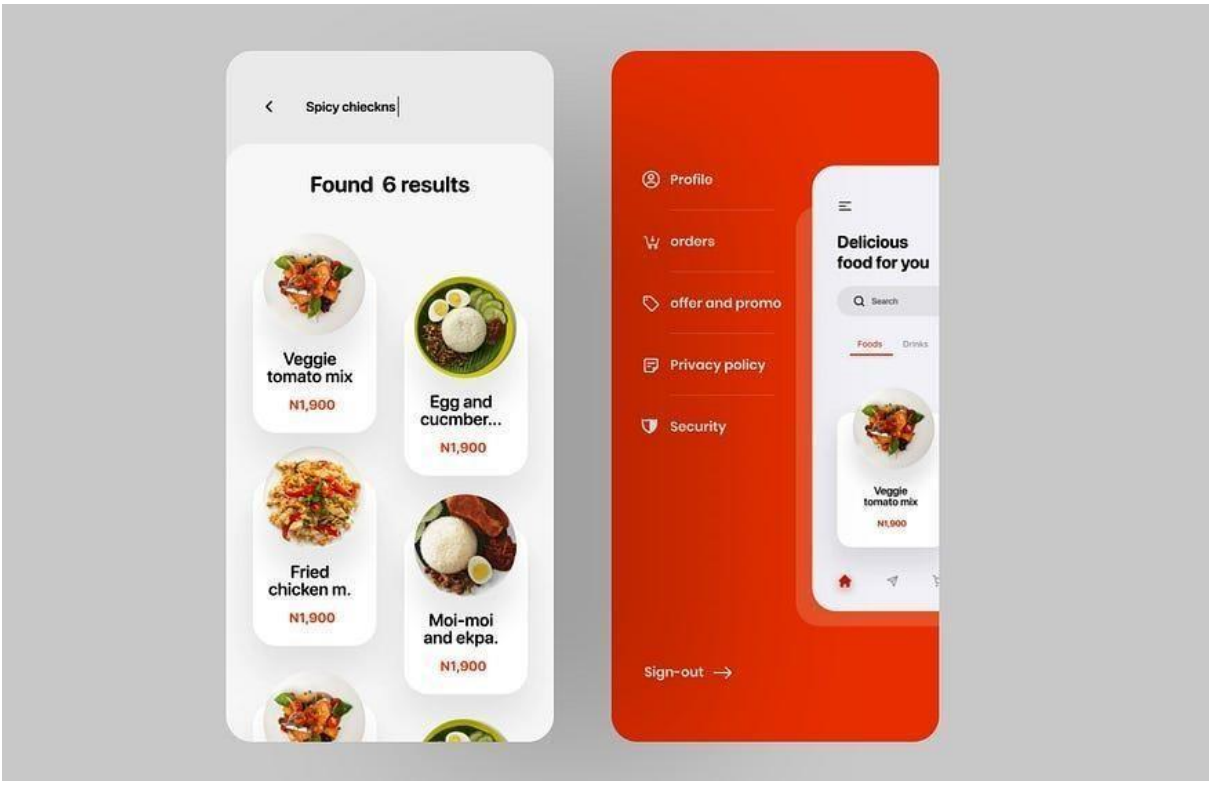


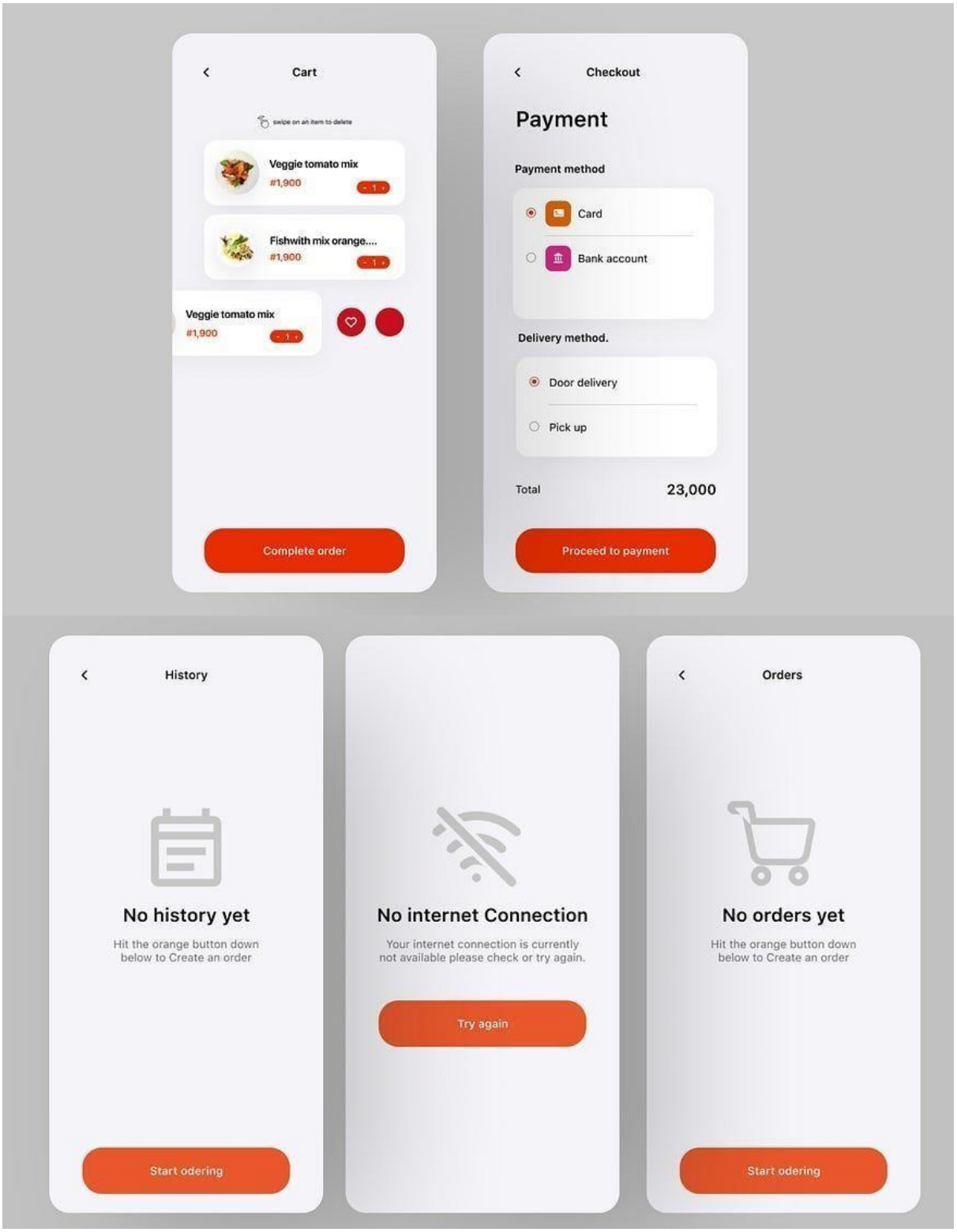
Home Screens

Shop, categories, and search: colors, font sizes, and product listing layout were determined by running multiple A/B tests. For the listing, it turned out that a grid arrangement slightly increased the conversions, thanks to its professionally shot pictures.

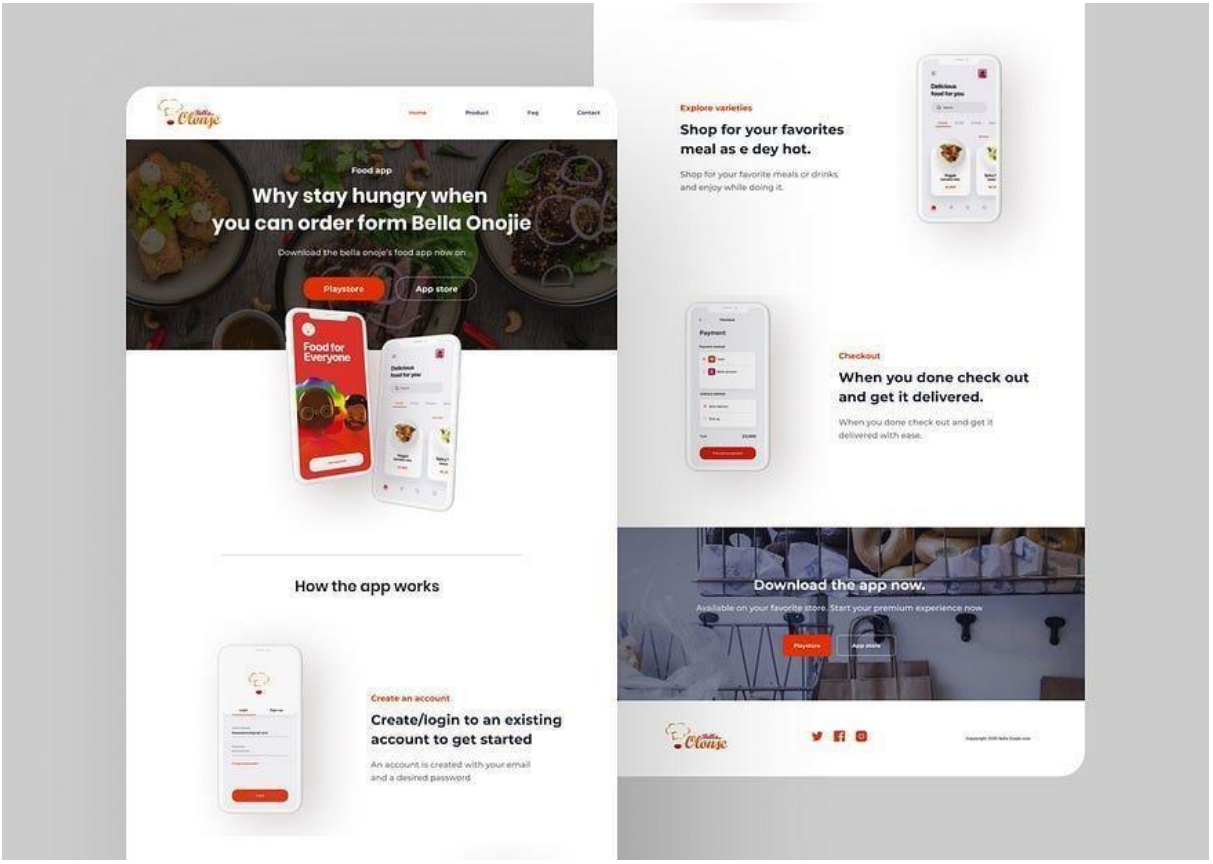


Other Screens



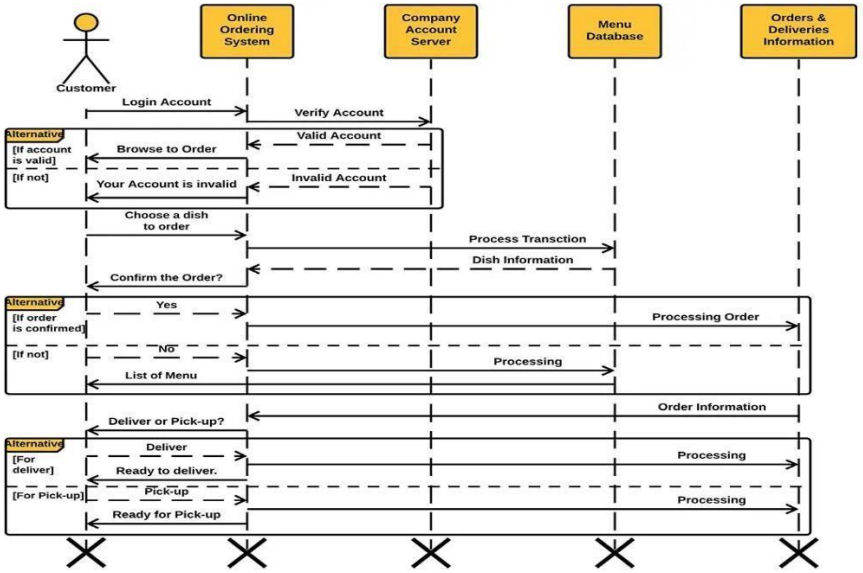

Cart
swipe on an item to delete
Veggie tomato mix
#1,900
- 1 +
Fishwith mix orange....
#1,900
- 1 +
Veggie tomato mix
#1,900
- 1 +
Complete order
Checkout
Payment
Payment method
Card
Bank account
Delivery method.
Door delivery
Pick up
Total
23,000
Proceed to payment
History
No history yet
Hit the orange button down below to Create an order
Start odering
No internet Connection
Your internet connection is currently not available please check or try again.
Try again
Orders
No orders yet
Hit the orange button down below to Create an order
Start odering

Website Design



4.1 Detailed Design

4.2 Sequence diagram


Customer
Online Ordering System
Company Account Server
Menu Database
Orders & Deliveries Information
Login Account
Verify Account
Alternative
[If account is valid]
Valid Account
Browse to Order
[If not]
Invalid Account
Your Account is invalid
Choose a dish to order
Process Transction
Dish Information
Confirm the Order?
Alternative
[If order is confirmed]
Yes
Processing Order
[If not]
No
Processing
List of Menu
Order Information
Deliver or Pick-up?
Alternative
[For deliver]
Deliver
Processing
Ready to deliver.
[For Pick-up]
Pick-up
Processing
Ready for Pick-up