Foodie Connector

Project Backlog

Team 25: Zequan Wu, Shuqi Ma, Shiwen Xu, Xiaonan Shen

Problem Statement

Ordering food delivery is not as easy as it seems. People usually waste money on the food they don’t need to satisfy the minimum delivery fee, or spend much time splitting bills with friends who order with them. Our project aims to build a platform, on which users can create group orders, split bills, invite friends and even find neighbors they don’t know to order together. They will be able to easily meet the order minimums and save money on delivery fees.

Background Information

Users

Food delivery has become more and more popular as a dining choice. What is unique about Food delivery is that it can save tons of time people spend on the road, especially during awful weather or heavy traffic. Also, it is beneficial to people who don’t have a car, in case they prefer some specific restaurants that are not within walking distance. However, order minimums and delivery fees still stop people from ordering.

Restaurants

Today's food delivery platforms are not optimized for the restaurants as well. The restaurants have to set up order minimums and charge delivery fees to make profits in food delivery, which blocks out some customers. Besides, restaurants want to deliver as soon as possible, so they miss many opportunities to deliver the nearby orders together. It means that drivers always go to the same area over and over again, and the restaurants need to hire many drivers. Otherwise, the customers would complain about there delivery time.

Similar Platforms

Plenty of food delivery apps and websites, such as Uber Eats, HungryBoiler, Ricepo, and Mr.Delivery, are already available and do a pretty good job of fulfilling customer needs. Most of them have basic features along with some extra ones such as order tracking and rating. Some of them, like HungryBoiler, allows users to start a group order so that each person has the chance to choose his/her favorite, while the initiator still needs to pay for the whole order.

Limitations

While the existing food delivery platforms are very convenient and plenty of people are using them, they usually have strict order minimum requirements and high delivery fees. Besides, it takes some extra effort if a group of people wants to order together. First of all, for those platforms that have no group ordering or shared ordering features, people need to pass around someone's phone or manually combine everyone's choice into one order. It takes some time, and there could be mistakes. Secondly, people need to split the bill. They have to calculate everyone's cost, which can be difficult when there are taxes, tips, and delivery fees involved. Then cash or some online payment services will be needed.

Our Solution

Our platform helps solve the above problems by allowing users to place group orders while everyone only pays for their own food and fees. Besides, users can find neighbors who are willing to order together directly on our platform by creating public orders or choosing a nearby order to join. Restaurants also benefit from our platform while customers no longer being scared off by the order minimum and delivery fee. Also, neighbors are going to place one group order instead of several individual ones, which help restaurants optimize their delivery and keep a minimum number of drivers.

Requirements

Functional Requirements

1. As a user, I would like to register for an account.
2. As a user, I would like to login and manage my account.
3. As a user, I would like to reset my password if I forget it.
4. As a user, I would like to save my address and payment information so that I don't have to type them in every time.
5. As a user, I would like to have a general address based on my current geolocation so that I don't have to type in my address before I decide.
6. As a user, I would like to see the available restaurants, together with the delivery fee, estimated delivery time, and other related information.
7. As a user, I would like to search for specific restaurants by name.
8. As a user, I would like to filter or sort the restaurant list by different properties such as distance, rating, and estimated delivery time.
9. As a user, I would like to see the menus for each available restaurants.
10. As a user, I would like to see different options (e.g., sauce choice, meat choice) on the menu.
11. As a user, I would like to create group orders.
12. As an order creator, I would like to generate a link of the order.
13. As an order creator, I would like to share the link to others using text messages, emails, or other third party applications.
14. As an order creator, I would like to generate a QR code of the order so that it will be more convenient for face-to-face sharing.
15. As an order creator, I would like to set the order to private so that only people who know the links and QR code can see and join it.
16. As an order creator, I would like to set the order to public so that can be seen and joined by all nearby users.
17. As an order creator, I would like to set a time limitation in which other users can join the order.
18. As an order creator, I would like to be the only user who confirms the order.
19. As an order creator, I would like other users set themselves as ready before I could confirm the order so that we can make sure everyone has selected the items they want.

20. As a user, I would like to see all nearby public orders that I can join.
21. As a user, I would like to split the total cost so that I only pay the price and delivery fee proportional to the food chosen by me.
22. As a user, I would like to add friends.
23. As a user, I would like to invite friends from the friend list to join the order.
24. As a user, I would like to track my order status.
25. As a user, I would like to track my driver's real-time location.
26. As a user, I would like to rate the restaurants from which I've ordered.
27. As a user, I would like to see my order history so I could choose the same restaurant again easily.
28. As an administrator, I would like to login to a dashboard.
29. As an administrator, I would like to manage (add/update/remove) restaurants.
30. As an administrator, I would like to manage restaurants' menus.
31. (If time allows) As a user, I would like to receive notifications when my order status updated.
32. (If time allows) As a restaurant employee, I would like to see the detail of all active orders.
33. (If time allows) As a restaurant employee, I would like to update the status of orders.

Non-Functional Requirements

Architecture and Performance

The website will have separated frontend and backend, communicating using RESTful API. It allows us to divide our work more clearly so that we can develop as effectively as we can. The frontend will be based on React to help accelerate development and implement complicated interacting logics. The backend will be written using Laravel, a popular PHP framework. MySQL is going to be used to store persistent data, and temporary data (e.g., sessions, caches, and flash data) will be stored in Redis, a key-value database that runs very fast.

Usability

The user interface is going to be straightforward and easy to use. The website will be responsive so users can use it on desktops, mobile phones, or tablets. Besides, React support all popular browsers so most of the users should be able to visit our website using their favorite browser.

Response Time

Since in our platform, multiple users can make changes to the same order, so there need to be real-time updates. We will use WebSocket to deal with the two-way communication. The system doesn't have any strict restriction on the response time, but the average should below 500 ms to ensure a good user experience.

Security

Since our platform stores users personal information (e.g., address, phone number) and deals with payment, it is crucial to protect all these data. Laravel has many built-in security features. It uses PHP PDO to prevent SQL injection, and it has CSRF protection using token verification. Besides, we are going to use HTTPS to avoid Man-in-the-middle attack. User passwords will be hashed, and all sensitive information will be encrypted. Privileges will be carefully assigned so that every user can only access to their own data.

Deployment and Scalability

We will use the Bitbucket Pipelines for automatic integration and deployment. Docker and Docker Composer will be used to make deployment easier. By using MySQL, Redis, and Docker, the whole system can be easily scaled and deployed on a distributed system. We will use Nginx as a static file server and a reverse proxy. It can also be used as a load balancer if we have multiple Laravel backends deployed.