Delivery and Order Management System(DOMS)

Team 15 Project Backlog

Hemil Desai, Shivank Tibrewal, Dinesh Gajwani and Kunal Goyal

Problem Statement:

In our current society, delivering customers right to their doorstep significantly raises the value of any store. Although local stores would like to provide delivery, they often do not have the means or capital to do so. Our solution will help local store owners to deliver their orders through an automated delivery and ordering system. Not only will this give the local stores an online identity, but it will also help them compete against big box online retailers.

Background:

Audience

Currently there are many local stores who do not have the resources to provide delivery around their neighbourhood. Also many customers exist who prefer delivery to any other form of commute to meet their transactional requirements. There are also many people willing to be drivers to provide delivery. With our platform, we plan to bridge the gap between all three parties and provide a unified way to meet the requirements of all through a convenient UX and a platform that allows the local markets to administer their deliveries.

Similar Applications

There is hardly any system that administers the capabilities we are planning to offer. There are many other close platforms that offer delivery but only for restaurants. Platforms like Hungry Boiler offer only food deliveries through their own or local restaurant drivers, Grubhub offers a

unique way to offer food but only through local drivers hired by the restaurant.

We are planning to provide a way to crowdsource drivers by creating an algorithm to accomplish multiple deliveries through different stores at the same time in a neighbourhood, and this is not just restricted to restaurants. Based on the urgency of the deliveries, our algorithm plans to combine deliveries making the delivery efficient for all three parties (Local stores, drivers, customers). This platform solves a huge platform and there does not exist a similar application with these functions.

Limitations

The limitations of delivery and ordering platform exist due to their limited scope. There are ordering platforms that use local store drivers and there are delivery platforms that lack efficiency due to improper delivery scheduling. Our platform tends to create a platform efficient for the local stores, that can be used by any ordering service to form a unified platform for accepting orders and scheduling deliveries through our algorithm that groups deliveries based on neighbourhoods.

Functional Requirements

1. As a administrator, I should be able to:
    a. To add store owners
    b. To manage users
    c. To manage drivers
    d. Manage all kinds of accounts
    e. Locate active drivers on map
    f. Provide a way for different ordering platforms to place orders for all local stores
2. As a Store owner, I should be able to:
    a. Add orders to the system
    b. Add my drivers
    c. Assign orders to available drivers - automatically and manually.
    d. Locate working drivers

    e. Integrate with existing ordering systems.
3. As a Driver, I should be able to:
    a. Accept and reject delivery
    b. Accept and reject group deliveries based on neighbourhoods
    c. See the delivery address on the map
    d. Get an ETA
    e. See the best route to get to the address
    f. Complete a delivery

4. As a customer (TBD), I should be able to:
    a. See my driver information.
    b. Estimated time of arrival
    c. Provide feedback about my service
    d. Pay tip.
5. As a normal developer, I should be able to:
    a. Use the tools of the platform through an open API
    b. Use simple HTTP protocols to place orders via the API
    c. Integrate seamless payments
    d. Get feedback based on the orders

Non-Functional Requirements

1. The application should be available as an iOS application
2. The application should be available as an android application (If time allows)
3. The application will have a web-based delivery platform as well built using nodejs and front-end frameworks like HTML, CSS, Javascript.
4. The application will have a web-based admin platform for stores built using nodejs and front-end frameworks like HTML, CSS, Javascript.
5. The application should be as user friendly as possible. The application should not have too many buttons on one screen as that can overwhelm the user. Instead, we will should make it simple and understandable.

6. The system should have an online response time of no more than 10 seconds.
7. The application must be integrated with Google Maps.
8. The application will use apple's swift programming language to develop the iOS application.
9. The application will use the java programming language to develop the android or iOS application.
10. The application must be able to integrate with a phone's location services in order to analyze someone's position.
11. The server must be able to handle at least 100 concurrent users.
12. The application must be available 24 hours a day and 7 days a week. The allowable downtime is an hour for one month.