Delivery and Order Management System(DOMS) 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

Presently, there are many local stores who do not have the technological resources to provide delivery in their neighbourhood. Most customers prefer that their items be delivered to their doorstep over pickup. Similarly, there are many people willing to be drivers to earn some quick money. With our platform, we plan to bridge the gap between the three parties and provide a unified way to meet the requirements of all through a convenient platform that is not only user friendly but which allows local dealers to provide deliveries.

Similar Applications

At this point, we are not aware of a similar service that offers exactly what we plan to do. There are many platforms that offer delivery, but are only limited to the food industry. A good example of such a platform is HungryBoiler.

For the platform to succeed, it needs to be efficient and reliable, so that it is feasible for both the drivers and store owners. Due to time constraints, we plan to implement the First-Come-First-Serve algorithm, and if time permits, we would like to optimize it further. An optimized algorithm would follow the travelling salesman approach to provide drivers with best routes to take several deliveries on the same route. We believe that this platform has potential to succeed and is the need of the hour.

Existing Limitations
The biggest drawback of the current system is that they are limited to the food industry. Also, drivers in many restaurants are forced to remain idle during non peak hours. This is a huge waste of man hours. Since our system is very generic, restaurants may choose to send their drivers for smaller deliveries in the neighbourhood, in return for a small pay.

Functional Requirements

1. As a administrator, I should be able to:
    a. To add store owners
        i. Add store owners.
        ii. Delete store owners.
        iii. Modify store owners.
    b. To manage users
        i. Add users.
        ii. Delete users
        iii. Modify user data.
    c. To manage drivers
        i. Add drivers.
        ii. Delete drivers
        iii. Modify user drivers.
    d. Locate active drivers on map
    e. View feedback by users

2. As a Store owner, I should be able to:
    a. Add orders to the system
    b. To manage drivers
        i. Add drivers.
        ii. Delete drivers
        iii. Modify driver details.
    c. Assign orders to available drivers - automatically and manually.
    d. Locate working drivers
    e. Easily use the system with existing ordering mechanisms.
    f. View feedback by users

3. As a Driver, I should be able to:
    a. Accept and reject delivery
    b. See the delivery address on the map
    c. Get an ETA
    d. See the best route to get to the address
    e. 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 mobile application should be available as an iOS application.
2. The application should be available as an android application (If time permits).
3. The application will have use a web application to host delivery and admin platforms. The app will be built using nodejs for the backend and will use languages like HTML, CSS, Javascript for the frontend.
4. The application should focus on being user friendly. The application should not have too many buttons on one screen as that can overwhelm the user. Instead, we should make it simple and understandable.
5. The system should have fast response times and efficient latency and bandwidth. The system should have an online response time of no more than 10 seconds.
6. The application must be integrated with Google Maps.
7. The application must be able to integrate with a phone's location services in order to analyze the user's(driver) position.
8. The server must be able to handle concurrent users.
9. The app should aim to have minimum downtime.
10. Driver Scheduling algorithm will work on a first-come first serve basis.
11. The system will be reliable to not to misplace any orders and keep a constant track of them based on their status.
12. The API and database should be scalable to handle occasional overloads and several users.
13. The server should be efficient. If deployed on cloud, it should integrate well with services like Amazon AWS or Microsoft Azure.
14. The platform should have a testing mode and a production mode.