Dispatchr: Project Requirements Document
By: Team Savage

Members:
Alok Gupta (Lead)
Jordan Nguyen (Scribe)
Sal Olivares
Spencer Prescott
Brian Yan

Mentors:
Lu Jin
Daniel Vicory
Instructors:
Chandra Krintz
Ankita Singh

Intro & Goal

For many college students who do not have cars, it can be challenging to go the store to get groceries, supplies, toiletries, etc. For others, it can be a struggle to make time to go shopping. Dispatchr is the solution to both these problems more. Dispatchr is a community-center platform that allows users to help each other when it comes to shopping. Users can post requests for what items they want to buy from what store, and other users can accept these requests and buy those items. Users can also post which store they will be shopping at, and other users can request products from that store they'd like picked up.

Innovation

While there are various applications and services already existent that provide similar functionality to Dispatchr, there are several components that set it apart from the competition. Dispatchr's first competitive advantage is the ability to get any item picked up, regardless of whether it's groceries or school supplies. In contrast, other services only provide items of a single category, most commonly food from local restaurants, to be picked up.

The second competitive advantage of Dispatchr is the community aspect, particularly within college communities. In other services, products are picked up by random "contractors." In Dispatchr, items are picked up by local students that reside in the same community as those requesting items for pick up. Through this, students are able to help serve others within their community and do social good.

Project Specifics

This project is for primarily college students. In particular, it's a major pain point today for many to retrieve groceries or other items from local stores, especially since many college students don't have cars to do so. On the other hand, other college students that have a car and are able to visit stores regularly might be willing to pick up a few extra items for their peers, especially if they are compensated to do so. Being college students who can empathize with this dire issue in our community, we want to build a solution.

Dispatchr provides a two-sided platform for college students. The students who want items can post the item(s) they need, as well as any further information such as when they would like it by, if they have a store preference, as well as distance to actually pick up the item. On the other side, the "dispatcher" can view items from stores they are considering visiting and can opt to pick it up for their peer. In addition, when they visit stores that others are requesting items from, they will receive a push notification alerting them that they can pick up certain Items for their peers since they are there, and that they will be compensated accordingly as well. We will also include gamification in the form of trophies that are awarded to users as they continue to help their community peers, further incentivizing them to pick up items for others when they are shopping.

Team Goals

By the end of this project we plan to learn the scrum and software engineering process by building and deploying our application from scratch. Our main goal is to continuously iterate on the product we are building so we can solve a major problem for college students around the world. More specifically, we want to better learn and understand the software development lifecycle. We will be learning how to build an application from scratch, which includes building the client side mobile applications in React-Native and an API with Ruby on Rails using PostgreSQL for the database as well as Redis for caching requests. Throughout our software development lifecycle, we will be practicing test-driven development in order to build a robust and polished application. In addition, we will learn deployment processes. We will be deploying Dispatchr in the cloud using the Heroku Platform as a Service. As we build, we will continuously be iterating, making our product more refined and robust with every proceeding release.

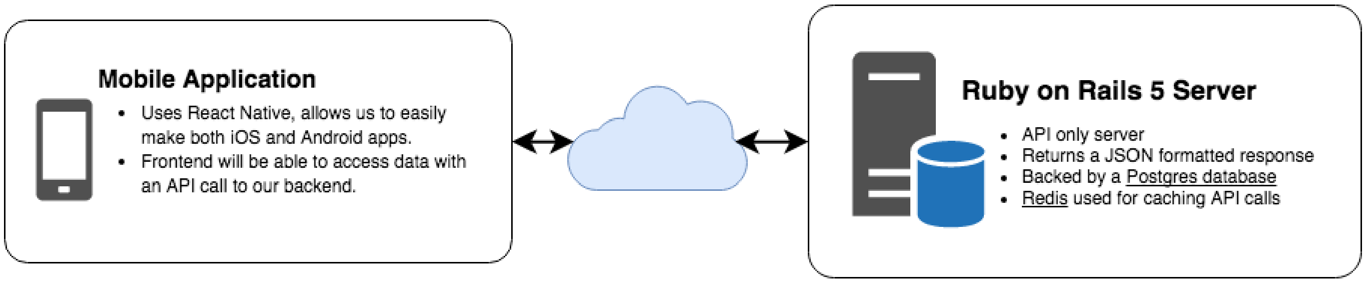
Background

Dispatchr is an iOS and Android application geared primarily for college students. In particular, it's an inconvenience today for many to retrieve groceries or other items from local stores, especially since many college students don't have cars to do so. On the other hand, other college students that have a car and are able to visit stores regularly might be willing to pick up a few extra items for their peers, especially if they are compensated to do so.

Assumptions

There are various assumptions that we are making while building this product. First, we're assuming that college students are willing to pay $5-$10 to have their items picked up, and that the students who are picking the items up are willing to accept this amount as well. We need to conduct further validation from students in Isla Vista to determine the final price for this by potentially creating a supply and demand curve. In addition, we are assuming that students will be willing to pick up items for others in the first place, although we're hoping that the compensation will incentivize them to do so for their peers, as well as the gamification.

Architecture



Mobile Application
Uses React Native, allows us to easily make both iOS and Android apps.
Frontend will be able to access data with an API call to our backend.
Ruby on Rails 5 Server
API only server
Returns a JSON formatted response
Backed by a Postgres database
Redis used for caching API calls

This is a high-level overview of our system architecture. We will be building a client applications for both iOS and Android, using React-Native. These clients will be powered with our Ruby on Rails API which will serve to persist data and conduct our backend logic. This RESTful API will use PostgreSQL for persisting the data, as well as Redis for caching particular requests, and return responses in JSON format that our client applications will interface with for use by our end users.

Glossary

1. Post - a submission from a user that comes in two forms
    a. Request - users post what items they want picked up from which store
    b. Buying - users post they are at the store and are available to buy and deliver items
2. User - anyone using the application
    a. Requester - a user who posts requests
    b. Buyer - a user who buys and delivers items for requesters
3. Item - something that can be bought and delivered at a store
4. Feed - a list of posts in reverse chronological order that can be filtered. There is a feed for Requesters and Buyers
5. Profile - user profile linked to basic information
6. Rating - user rating based on previous user interactions

Requirements (User Stories)

1. As a requester, I can create a list of items that I want picked up so other users can fulfill my request.
    a. Precondition: Requestor has an idea of what they need to get from the grocery store.
    b. Event: Requestor can enter all the items the need on an easy to use form in the app.
    c. Postcondition: Requestor is shown their list with all the items they added before saving it.
2. As a requester, I can select where I want my items to be delivered so that I can pick up my request.
    a. Precondition: Requestor has a list of items that he or she needs to be delivered.
    b. Event: Requestor enters a location where they will be to pick up their items.
    c. Postcondition: The requester's items have a delivery location so buyers know where to drop of the items.
3. As a requester, I can specify a price range for my request so a buyer does not buy an item I cannot afford.
    a. Precondition: Requester has a list of items they need to be picked up.
    b. Event: Requester specifies for each item, the price range they are willing to pay for that item.
    c. Postcondition: Each item has a max price so the buyer does not buy an item the requester cannot afford.
4. As a buyer, I want to be able to easily filter the posts, to be able to more easily purchase items for others.
    a. Precondition: Buyer is at the grocery store looking for items to buy.
    b. Event: Buyer filters for items he or she is alright with buying.
    c. Postcondition: Buyer is shown items that requesters have added for pickup that also match the filters supplied.
5. As a requester, I can select when I want to pick up my items for convenience.
    a. Precondition: Requester has a list of items they need to be picked up.
    b. Event: Requester specifies when they can pick up their items.
    c. Postcondition: Each item has a delivery location so the buyer knows where to drop of the item.
6. As a requester, I will be able to cancel my request before someone accepts it so that I do not receive items I no longer need.
    a. Precondition: Requester decides they no longer need the items they requested.
    b. Event: Requester goes to their posting and deletes it.
    c. Postcondition: The requester's post has been deleted from the feed and is no longer accessible.

7. As a user, I want to give a rating to those I interact with so that others will know the quality of people they are dealing with.
    a. Precondition: Two users have finished their interaction with each other, and now have the option to give each other ratings through the application.
    b. Event: Either user gives a rating for the other user.
    c. Postcondition: User ratings are updated on their profile and can be viewed by anyone using the application.
8. As a user, I want to easily switch feeds between requested items and available drivers so that I can choose what role to participate as.
    a. Precondition: User wants to change which view they are looking at.
    b. Event: User selects the feed they wish to see at the top of the screen.
    c. Postcondition: The feed the user selected will be displayed on their screen.
9. As a buyer, I will be able to receive money before fulfilling a request so that I will not be scammed.
    a. Precondition: Buyer agrees to pick up items for a requester.
    b. Event: The requester is presented with a list of payment options available by the buyer, and selects one.
    c. Postcondition: Buyer receives a notification and confirmation of payment.
10. As a user, I can upload a picture of myself so others know what I look like when I deliver or pick up items.
    a. Precondition: User is on their profile settings and wants to upload their picture.
    b. Event: User selects to upload a picture via camera roll or through taking a new picture.
    c. Postcondition: User’s profile has a new picture associated with it so other users can see it.

Appendices

Our system architecture uses Ruby on Rails for our API server, React Native for our mobile app, Postgresql for our datastore, and Redis for caching. We are using Github for our version control system, Google Drive for storing our documents, and Pivotal Tracker to keep track of our process.