ParkHere Implementation Document

October 21, 2016

Team Name: LazeeBear

Team 11

Henry	Ryosuke	Jason	Robert	Zhicheng	Sally
Levy	Mitsubayashi	Roodman	Sieh	(Franklin) Wang	Wei
6761013619	4239440668	3920867056	6502483653	6100373109	1145544502

1. Preface

This document outlines the changes made to the planned architecture of the program during implementation of the ParkHere system.

2. Introduction

ParkHere is a Peer to Peer Resource Sharing Platform (PRSP) for private parking spaces. Especially in large cities and near popular venues, parking can be hard to find and can be very expensive. ParkHere aims to alleviate this by allowing owners of nearby private parking spots to rent their parking spots out, possibly for cheaper prices than available public parking such as metered street parking. This would also provide drivers a way to ensure and determine their parking spot ahead of time.

3. Architectural Change

1. The classes previously named “pages” (ex SignInPage) were renamed to “activities” (ex SignInActivity)
    a. To keep consistency with the naming conventions of Android
2. The class previously named “ParkingSpotPage” was renamed to “SpotDetailsActivity”
    a. The change from ParkingSpot to SpotDetails was to help differentiate it from the CreateSpotPage/Activity
3. Added an extra class called “SpotDetailFragment”
    a. Separates the layout of spot detail’s information and the display of spot detail
4. Several data structures were renamed from their names on the Architectural Design
    a. To reduce the length of strings.
5. Removed the option to review a spot from the account page (as detailed in the Class Interaction Diagram)
    a. Makes more sense to only be able to review a spot from the spot that was rented.
    b. To review from the account page is difficult to implement than reviewing from the spot page. Reviewing from a spot already includes username data in it, whereas reviewing from an account page requires fetching of the username and the relevant spot’s data.
6. Replaced the original PaymentPage with an implementation of the Braintree API (see 4.2.a)
7. Changed the path from the login page. Now after login, user (both seeker and owner) will go to account page, and then user can go to search page or create spot page.

8. We do not have filter search right now, but we will implement it in the following assignments.
9. No view for deleting spots

4. Detailed Design Change

1. Excluded latitude/longitude for searches
    a. (@Jason please explain the rationale)
2. Using Braintree API to implement payment transactions (through credit card, Venmo, and PayPal)
    a. Relatively simple to implement, and Braintree handles everything, including switching applications and making payments.
3. Added a GetClientTokenHandler to get a Braintree Client Token
4. Added description field to spot
5. Added several low-level methods to the DatabaseManagers and User Objects(to pass data along to the Handlers)
6. Added a AddSpotReview page to add review of spot
7. Search server connector (getting search results) does not work, so we did not get to test any functionality beyond that.
8. Add rating server connector (both spots and users) is not implemented so we can't post or see ratings
9. Images not implemented but we will implement it in the future
10. In spot detail page, we only have one button for rating the user, and we will add one button for rating the spot
11.

5. Requirement Change

This changes our design because currently we specify for each spot a start time, an end time, and a boolean isBooked combined with a renterEmail to specify a single renter. Thus, our design only allows for one renter for spot and does not support partial bookings. To make this change in our design documents, we need to create a new Bookings table. renterEmail and isBooked are no longer included in the Spot table. The Spot table only holds the necessary posting information posted by the owner. Bookings holds bookings, which contain spotID, renterEmail, start, and end times. The BookSpotHandler checks to make sure a proposed booking does not conflict with any of the bookings containing the same spotIDs in the Bookings Table. ViewSpotHandler returns information about the spot and a list of all the current bookings times searched through spotID in the Bookings Table.

This doesn't change very much on the front end, as time slot bookings are already being handled with options to spit hours into minutes and days into hours. However, there might be a need to duplicate posts on the front end that will correlate with the new ability to split a single post between multiple renters.