ParkHere Sprint 2 Plan

Team 11b

Ryosuke Mitsubayashi	Robert Sieh	Zhicheng(Franklin) Wang	Sally Wei
4239440668	6502483653	6100373109	1145544502

II. Preface

This document includes the backlog for Sprint 2, the documentation for improvements of nonfunctional requirements, and the record of daily Scrums.

III. Sprint Plan

Sprint 1 leftovers

- ~~cannot view past rental spots~~
- ~~cannot cancel reservation (UI)~~
- ~~cannot retrieve past reservation information from server~~

Sprint 2 Backlog

Spot Details Page -> Post page

- ~~When displaying spot price, add 10% booking fee to it.~~
- ~~Show cancellation policy on spot details page~~
- ~~Owners can adjust the price after posting~~
- Confirm expiration of spot so that the owner can get paid.

(when renting)

- ~~Enforce minimum renting slot of 1 hr~~
- When Seeker accepts a recurring rental, they can set the amount of weeks to commit to
- Seeker can accept the rental for up to its whole time slot

Reviews

- Spot rating (written description)

Account

- ~~When displaying spot price (including when filtering by price), add 10% booking fee to it.~~

Search

- ~~Search by both date and time slot~~

Other:

- ~~Some method to “Contact customer service”~~
- Cancellation policies

New requirements:

Functional:

1. An owner can manage (create/edit/delete) her own **parking spots** (not only the posts)

2. When the owner creates a post, she should **choose from the parking spots she already created** rather than entering all the information for the parking spot required when creating the post
3. A seeker may request an **overview of prior bookings for all parking spots available in an area** (e.g., within a mile of a stadium). The visualization would indicate the number of bookings for a spot, **e.g., by the shading**: darker shades indicate that a spot is booked more often, lighter shades that it is not booked as often, and no shading that it has not been booked previously.
4. The rating system changed to:
   a. 1) the **ratings/reviews showed on a post is the rating for the linked parking spot** (i.e., the owner can publish multiple posts about the same parking spot A (PA) with different availabilities. In this case, the ratings/reviews showed on each post should be all the ratings/reviews in all the posts about PA. In other words, the ratings/reviews in all the posts about PA should look the same.);
   b. 2) an owner can **see the ratings/reviews for her parking spots**
5. **"Split" booking** mentioned in the end of Assignment #3
6. All the other "core" functional requirements listed in the Requirement List

Non-functional:

~~7. Improve the performance (speed) of your system~~
   a. Identify at least **two common user behaviors that will potentially cause long response time**, and **provide a way for each case to reduce the response time** in your implementation. E.g., you identify that a seeker always goes to the owner's profile page next when she is on a parking spot information page. In this case, you can improve the performance by requesting owner's profile information when you request the parking spot information, and store the owner's profile information in the cache. Then when the seeker goes to the owner profile page, you can provide the owner profile information from the cache rather than request it from your server.
   b. You need to **show the response time is actually reduced**. E.g., you should record the response time before and after your improvement. Please feel free to use external tools.
   c. Extra points will be considered if
      i. 1) you can simulate your test cases (before and after improvement) under different network conditions;
      ii. 2) you can replay your test cases automatically (e.g., using RERAN)

Optional:

8. Other cool features of your choice! (Particularly nice, non-trivial features will be considered for extra points.)
   a. Added cache for user login (users can login without signing in from the second time onwards)
   b. We can modify the cancellation policy of the spot

IV. NFR Improvement

1.

a. When we go from the search page to the list of spots page, it takes a long time to load.

b. When we go from the search page to the list of spots page, we are passing a list of spot IDs that we retrieve from the server. When we get to the list of spots page, we use the spot IDs to retrieve more information (i.e. the address we display on the buttons) from the server.
Instead, we initially retrieve the spot IDs and all the extra information we need, and we pass that list from the search page to the list of spots page. This way, we do not have the overhead of connecting to the server again to retrieve more information.

Code Fixes
<u>app/src/main/java/lazeebear/parkhere/Search.java</u>
Added

```
ArrayList<String> spotAddresses = new ArrayList<>();
spotAddresses.add(spots.get(i).getAddress());
intent.putExtra("addresses",spotAddresses);
```

<u>app/src/main/java/lazeebear/parkhere/SpotListActivity.java</u>
Added

```
ArrayList<String> spotAddresses = intent.getStringArrayListExtra("addresses");
populateList(spotIDs.get(i), spotAddresses.get(i));
```

Removed
//connecting to the server
SpotDetails DAO spot = ServerConnector.spotDetails(id)

Changed
populateList(int id){} to populateList(int id, String address){}
createSpotButton(spot.getAddress()) to createSpotButton(String address)

c.

i. We did this by hand using a stopwatch

ii. The time is from when we click the button to when the next page appears. The search query is the same for both test cases and it returns 4 spots.

iii. Results:
Before -> approximately 5 seconds (4.8, 5.0)
After -> approximately 2 seconds. (2.3, 1.8)

2.

a. On the account page, grabbing information about spot history, spots owned, and current reservations takes time (3 fixes?)

b. On the account page, we save the spot IDs of the spot history, spots owned, and current reservations. When we click on the button to display the list of spots, we connect to the server for each spot and grab the address.

app/src/main/java/lazeebear/parkhere/Account.java

```
private Button createSpotButton(int id) {
            String address = “” ;
            final int finalID = id;

            try {
                System.out.println(”Getting spot DAO from spot ID...”);
                SpotDetailsDAO spot = ServerConnector.spotDetails(id);
                System.out.println(”Successfully got spot DAO!”);
                address = spot.getAddress();
            } catch (Exception e){
                Log.i(”ERROR”, ”Exception while getting spot info on account page”);
            }

            System.    .println(”Creating button in createSpotButton()”);
            Button button = new Button(this);
            button.setText(address);
            button.setOnClickListener( new View.OnClickListener(){
                public void onClick(View view){
                    addIntentDetailedSpotButton(finalID);
                }
            });
            System.    .println(”Returning button in createSpotButton()”);
            return button;
}
```

Instead, we grab the spot id and spot address from the server initially (saved in SpotButtonDAO) and parse that through the same method

```
private Button createSpotButton(SpotButtonDAO dao) {
   String address = dao.getAddress();
   final int finalID = dao.getID();
   System.    .println(”Creating button in createSpotButton()”);
   Button button = new Button(this);
   button.setText(address);
   button.setOnClickListener( new View.OnClickListener(){
       public void onClick(View view){
           addIntentDetailedSpotButton(finalID);
       }
   });
   System.    .println(”Returning button in createSpotButton()”);
   return button;
}
```

C. i. Stopwatch

Ii. The time is from when we click the spot history/owned spots/current reservations to when the list shows up

iii.

Method	# of spots returned	Before (s)	After (s)

Current reservation	1	0.5	0.1
Spot history	2	1	0.1
Spots owned	4	2	0.1

V. Daily Scrums

1. Nov 23, 2016 Wed. 6pm through Slack

Robert

1. Worked on date time search spot code
2. Finish date time search code
3. None

Ryo

1. Worked on solving the issue with spot history not displaying (unfinished)
2. Will attempt to fix it tomorrow as well
3. Can’t login?

2. Nov 24, 2016 Thurs. 5pm through Slack

Robert

1. Finished date time search code
2. Test date time search code
3. none

Ryo

1. Fixed spot history, did the same fix for current reservations. Also finished user rating (front end + server connector)
2. Work on performance improvements
3. Too many spam messages that are suppressing other output to the console

Franklin

1. Read through the backend code, tried to understand handlers
2. Understanding the serverconnector and the url code in it
3. none

3. Nov 25, 2016 Fri. 8pm through Slack

Ryo

1. Fixed spamming problem. Also improved the performance of displaying search results
2. Work on latitude/longitude
3. No visible progress on other features we need to implement.

Sally

1. Looked over server. Planned how to add the spots/posts differentiation.
2. Work on the server.
3. Don’t know how to access the database.

Robert

1. Testing search by date and time

2. Work on recurring rentals structure, solve SQL bug in search by date and time
3. None

Franklin

1. Read through the fixed code, understanding the fix
2. Work on the server connector
3. None

4. Nov 26, 2016 Sat. Slack

Ryo

1. Added spot rating.
2. Work on actually letting users change their password
3. Still no visible progress on other features we need to implement.

Sally

1. Looked over database. Added server functionality for spots.
2. Work on server connector and frontend for spots.
3. Don't know how the server URLs work.

Franklin

1. Understanding the table of address
2. Helping with creating address
3. Haven't tested the Address.py yet

5. Nov 27, 2016 Sun. Fertitta

Ryo

1. Enforced 1 hour minimum parking spot, thought about another aspect of speed we can improve
2. Improve another aspect of speed (creating a cache so the user automatically logs in the second time around?)
3. Nothing really. Just need to work a lot today and tomorrow

Sally

1. Worked on frontend for spots. When creating a post, the owner is sent to select an address or create an address first.
2. Frontend contact customer service.
3. Not sure about the backend for post management.

Robert

1. Tested search by date and time (complete)
2. Contact customer service, handle recurring rentals, get Sally's code for address vs spot distinction functional. Finish front end modify and delete address.
3. Need Front end for all this done as I am working on it

Franklin

1. Working on the address of spots. Helping Sally debug the front end
2. Tested searching by date and fixed the code
3. Not too much

6. Nov 28, 2016 Mon. 6:25 pm - 6:34 pm Fertitta

Robert

1. Worked on Backend Address code, finished customer customer service on the backend,
2. Finish address Code. Create address server connectors. Add modify price. If I have time, work on split and recurring bookings on the backend.
3. None

Ryo

1. Implemented cache for user log-in. Working on spot history speedup
2. Working on spot history speedup and RERAN. Braintree if there is extra time(refund handling)
3. None

Franklin

1. Finished testing search by date and time
2. Transfer date time information from search page to spot details page. Create search by date/time AND location
3. None so far

Sally

1. Finished contact customer service frontend.
2. Finish up create spot page, added modify spot functionality.
3. Address ID is missing from the returned address information.

7. Nov 29, 2016 Tues. 4:22 pm - @ LVL

Robert

1. Finished address code. Created address server connectors. Created modify price handler and server connector
2. Add send email inside book spot to notify owners to contact owners to receive payment, handle refunds on the backend
3. Front end for refunds

Franklin

1. Creating and testing search by both address and date
2. Finish searching by both address and date
3. Having server problem.

Sally

1. Finished adding modify spot functionality to the create spot page, connected it. Finished making owned spots go to address list before spot list.
2. Test GUI.
3. Having server connection errors (timing out when logging in).

Ryo

1. Finished improving speed of spot history, current reservations, and owned spots. Fixed logout bug.
2. Help out Sally with server-side knowledge, fix server bugs as they occur
3. None

VI. Sprint Review - Nov 29, 2016 Tues. 11pm Leavey Library

What has been done?

-Everything on the Sprint 1 Backlog

-Everything on the Sprint 2 Backlog except for recurring/split bookings and spot reviews

-New reqs 1,2,6,7

-Adding cache for user log in

-Extra function for modifying the cancellation policy on the spot

What was planned but wasn't completed? Why?

-Recurring/split bookings(Requirement 5) was not completed because we felt it required a large structural change that we would not have time to finish

-Spot reviews was not completed because we felt it required a new table and large modifications to much of our existing code and we would have ran out of time

-Requirement 3 (map overview with shading) was half completed (backend side) but we ran out of time to create a map-based interface on the front end side

-Requirement 4 (consolidated ratings) was not completed for the same reasoning behind Spot Reviews

What went well during the Sprint?

- Searching by the date worked very well, so we combine the date search and address search together to meet the seeker's requirement.
- Everyone was able to become familiar with the whole code

What could be improved in the next Sprint?

-Everyone starting earlier

-Having a discussion (not a scrum) to plan out what needs to be done and who should do it

-Notifying people of any commitments during long breaks (Thanksgiving)

-Letting people know immediately when working on a major change