SPOT FINDER

Easiest way to secure a parking spot

Abstract

This document is the 4th of 4 Milestones of our SE 430 project "SPOT FINDER"

Code N Chips
Group 4

Contents

Overview

The product to be developed is a Parking Spot Reservation System. Parking spots are offered to customers to reserve. The system will be used by Users ranging from individuals, businesses and any other parking lot faculty. Taking into account supply and demand factors. Users would utilize the system to reserve a parking spot at any chosen parking lot to accommodate their parking needs. Driven by location, availability, demand, and of course price.

Document Updates

The below table documents the progression of our work for Milestone 2 and PPT presentation.

01/22	Initial draft created
01/24	Team meeting at DePaul Library Discussed Use Cases Began to distribute responsibilities among team members Further added details to Milestone 2
01/25	Team member communication established via Whatsapp Continued Research among team members Prioritized use cases per our Milestone requirement Teammates continued to add on further details onto Use Cases Revised document where needed
01/26	Team member communication established via Whatsapp Added Class Diagram to document Began to create PPT presentation, suggestions among colleagues
01/27	Team member communication established via Whatsapp Continued revisions prior to turning in Milestone 2 requirements Continued revisions to PPT presentation among team members
1/28	Final huddle call and team member communication established via Whatsapp Final revision for Milestone 2 requirements Final revision to PPT Turn in MS2 and PPT via D2L

2/14	Began reviewing our Milestone 3 responsibilities
2/15	Team members met at the DePaul Library Collaborated among team members Milestone 3 document was created using our Milestone
2/16	Began our PPT slides Milestone 3 continued to be worked on among team members
2/17	Communication continued among team members Discussion cleared/assisted within our milestone Conference call among team members
2/18	Final huddle call and team member communication established via Whatsapp Continued revisions prior to turning in Milestone 3 analysis Continued revisions to PPT presentation among team members
2/19	Final Revision to Milestone 3 Final Revision to PPT Turn in MS3 and PPT via D2L (Zip file submission)
2/28	Began our MS4 analysis and distribution of responsibilities Collaborated among our team members
2/29	Continuous communication via WhatsApp and Google Hangouts Began to review the Design class diagram (the more we dove into the details the more our class diagram grew!)
3/1	Began our PPT slides Milestone 4 continued to be worked on among team members Outlined how we wanted to layout our PPT slides
3/2	Ongoing communication in regards to our current designs. Final huddle call and team members provided visual updates via google hangouts
3/3	Final Revision to Milestone 4 and PPT slides Turn in MS4 and PPT via D2L (Zip file submission)

3/15	Per Professor comments we discussed and began to update diagrams
3/16	Milestone 4 continued to be worked on among team members
3/17	Ongoing communication in regards to our current designs. Final huddle call and team members provided visual updates via google hangouts
3/18	Final Revision to Milestone 4 Turn in MS4 via D2L

Lessons Learned

As intended we have included the lessons each milestone has presented to our team.

Documentation	Lessons Learned
Milestone One	● Lined up administrative tasks, responsibilities, and schedules ● Set on our project - a parking spot a common topic we all could relate would be neat to break down.
Milestone Two	● Peer reviews offered simplification of information ● Communication eliminated bias, overlap and consistency ● Approach prioritization of use cases ● Challenges were incurred ○ Structure diagram - UML ○ Detail sharing -Tone and complexities ○ Different styles - Required unified discussion
Milestone Three	● Peer discussions with our visual diagrams ● Conversations were interactive ● Less is more, made us think to provide minimal information ● Analyze and pushed us to represent data to others ● Identified that certain steps may have not been required or in the wrong place and did not include them in the sequence diagram.
Milestone Four	● Peer discussions with our new/updated diagrams ● Utilized a conference call interface to productively review ● Raising any doubt within our design drove for our team to collaboratively improve our functionality and clarify diagrams ● Identified and simplified our designs
Final Revision	● All lectures came full circle, we were able to now ensure these would flow among each other. ● Communication is key, so if we were to do a soft launch, the Final revision is key to step back and see where our requirements all the way

	to the Design phase are in line. ● This would then open up either a step in the right direction or delays. ● Thus why all of this came into full circle.

Key Users

- Driver
- Owner
- Admin

External Systems

- Location - Google Maps
- External payment processing system - PayPal, Apple Pay, Tap and Pay
- FaceBook (for ease of uploading information & sharing)
- Data Security System

Product Analysis & Design Risk

Lack of a clearly defined target

We do not have a specific demographic/location at the moment. Currently, we are analyzing the holistic view. Our main target audience would be any individual that drives in areas with demand for parking areas. We also need to make sure our system would work in different metropolitan areas or demographical cities.

Verification of User information

Since we are allowing anyone to post parking spots to be rented we may run into issues with verifying the Usership of the area.

Features & Functions

Driver

- Browse/compare available parking spots
- Reserve a spot
- View current reservations

- ➢ Update/cancel current reservations
- ➢ Report issue, refer questions to admin

Owner

- ➢ Post property for lease
- ➢ View current spots posted
- ➢ Update spot information (availability, pricing etc.)
- ➢ Cancel lease
- ➢ Report issue, refer questions to admin

Admin

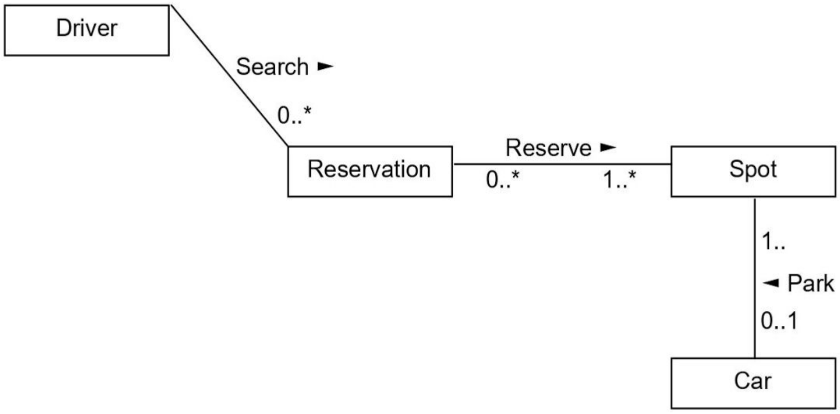
- ➢ Address client or system issues reported and resolve them
  - ○ Client complaints needing support
  - ○ System crash
- ➢ Respond to any inquiry coming from clients

Glossary

Driver	One who wants to reserve parking spot
Owner	Owner of parking lot
Admin	User of the system

1. Domain

The below defines the domain of the Spot Finder system that is being built.



2. Analysis

The below defines the problems we are trying to solve for Spot Finder.

2.1. Use Case Diagram below for Spot Finder system

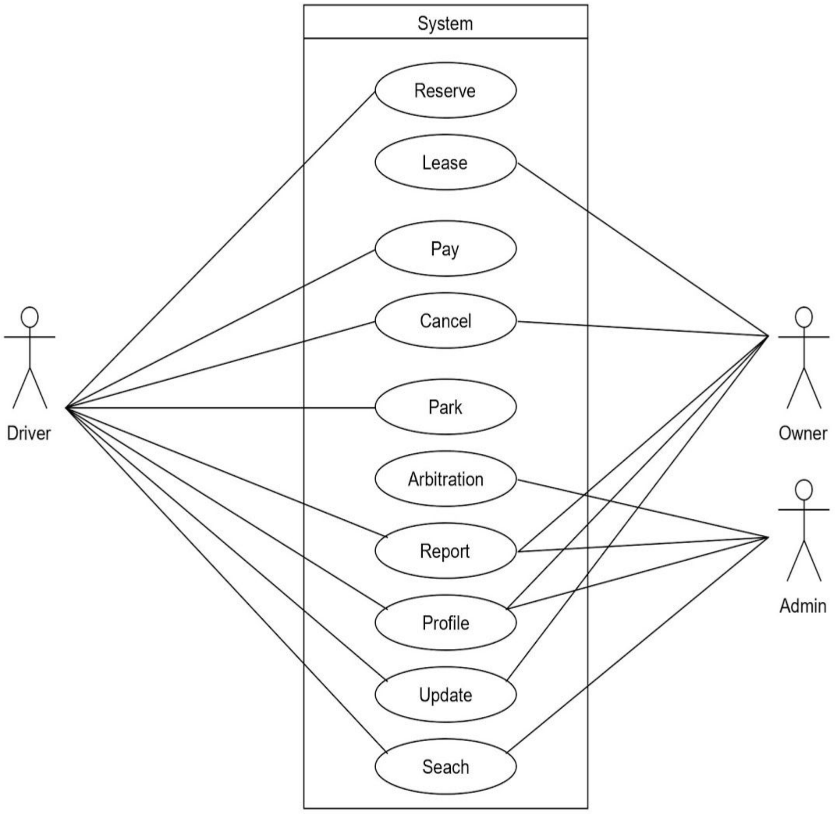


Figure 2.1

2.2. Use Cases

2.2.1. Profile

Actor

Driver/Owner

Description

The Driver and the Owner would be able to create an account so they can use the system to lease, rent, claim/report incidents.

Preconditions

1. The user has a valid space to rent and an active bank account to get paid.
2. The user has a valid bank account/credit card, email, facebook, or/and phone number.
3. Users have access to the internet and can access the system.

Main Steps

1- User:

- Users enter all the information needed from the system to create an account. That include and limited to:
  - Name.
  - User License number.
  - Car information (make/model/license plate)
  - Valid email/facebook account or a phone number.
- System saves all information :
- User enters Payment info
  - Valid Credit card or a bank account.
- System verifies payment and displays confirmation
- User should be able to modify/update profile
  - System should send a confirmation to the User that the account is successfully updates

- system let the User know if the account creation or update has failed.

2 - User enters all the information needed from the system to create an account. That includes and is limited to:

- ○ Name.
- ○ Valid bank account.
- ○ How many spots they own to rent or lease
- ○ Valid email/facebook account or a phone number.
- ○ User should be able to modify/update profile
  - System should send a confirmation to the User that the account is successfully updates
  - System let the User know if account creation or update has failed
- ○ Spot(s) information
  - ■ location
  - ■ upload photos
  - ■ proof of Usership (something with name & address)
  - ■ System verifies info and displays confirmation or a failure

Includes

Lease User Case (if User wants to make spot available right away)

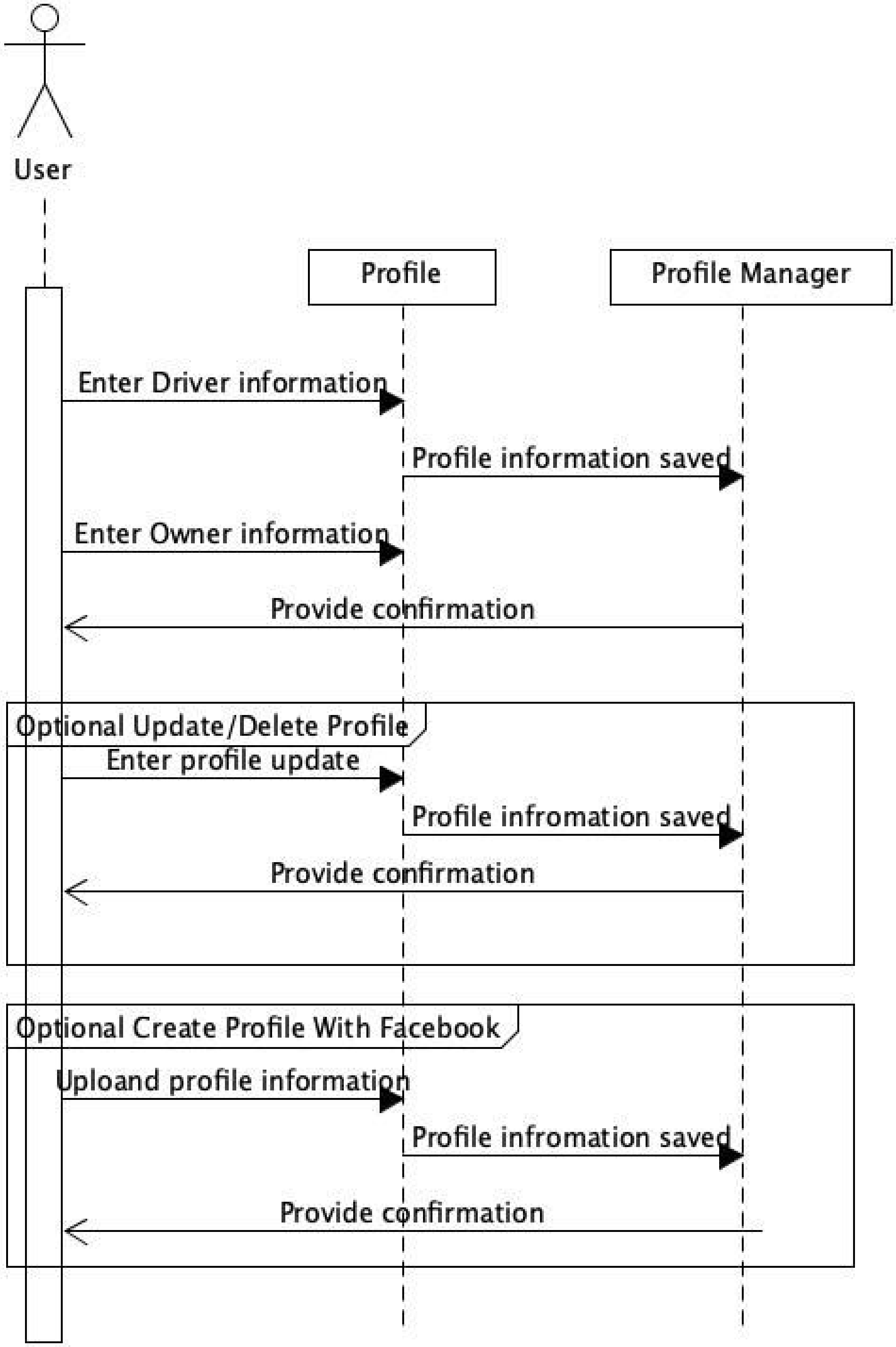
Exceptions

EX.1. Unable to verify payment, insufficient info ect…

Postconditions

Users have a valid profile that can be used and updated.

Sequence Diagram


User
Profile
Profile Manager
Enter Driver information
Profile information saved
Enter Owner information
Provide confirmation
Optional Update/Delete Profile
Enter profile update
Profile infromation saved
Provide confirmation
Optional Create Profile With Facebook
Uploand profile information
Profile infromation saved
Provide confirmation

2.2.2. Search

Actor

Driver

Description

The Driver would like to find the best rate for a parking spot.

Preconditions

1. The user knows the area in which they would like to reserve a parking spot.
2. User has access to the internet and can access the system.

Main Steps

1. User enters desired geographical location and time.
2. System provides parking options with prices.
3. User selects desired option.
4. System validates the selected parking lot and price and displays total cost.
5. System temporarily puts the spot on hold.

Exceptions

The following describe any anticipated error conditions that could occur during execution of the use case, and defines how the system is to respond to those conditions.

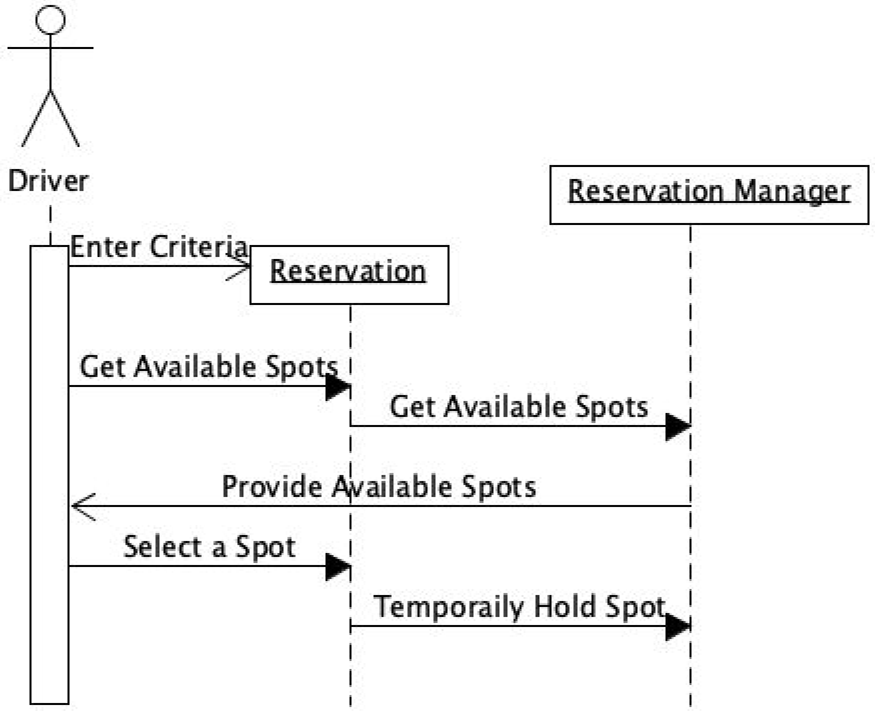
EX.1 GPS does not reflect accurate locations due to high density structures in a downtown area

a. Alternate solution is to allow the User to manually input the address and location of desired area

Postconditions

1. User selects a parking lot that best fits their price point and location

Sequence Diagram



2.2.3. Reserve

Actor

Driver

Description

The objective of this use case is to provide a User the ability to select and retain a desired parking space.

Preconditions

The following activities must take place before the use case can be started.

1. Requires an active User's profile. (Profile Use Case 2.2.1)
2. Requires User's ability to search for comparable and available parking. (Search Use Case 2.2.2)

Main Steps

The following is the relative priority of implementing the functionality required to allow this use case to be executed.

1. Following the search feature.
2. User enters the payment information.
3. System validates the payment information.
4. System confirms that the User wants to book the parking space.
5. System drafts the payment from the payment option.
6. System generates confirmation.

Alternative Courses

The following are other usage scenarios that can take place within this use case independently from the initial reserve case.

AC.1. User would like to update their current reservation.

AC.2. User would like to modify their current reservation.

AC.3. User would like to cancel an existing reservation.

Exceptions

The following describe any anticipated error conditions that could occur during execution of the use case, and defines how the system is to respond to those conditions.

EX.1. An elapsed period of time could pass where a User could go without making a selection but leaving a session active which could potentially block other Users from securing a space.

- a. To address this potential exception, the system would provide a warning within a defined period of inactivity; if no response, then the session would log out releasing the inventory for other Users.

EX.2. Theoretically, although highly unlikely, two Users could attempt to reserve the same space at the same time for the same price.

- a. Alternate solution to this is to have the system identify the split time in which the Users selected the options and grant the earlier submission.
- b. System to automatically refresh the system forcing the Users to resubmit.
- c. System to reflect the error and provide an option to release their reservation at first come first serve with an inconvenience discount for their next selection. Other User obtains the original selection.

EX.3. System crashes in the middle of a reservation.

a. Alternate solution to this is to have the system cache the search criteria and resubmit once recovery is successful.

EX.4. System crashes in the middle of a payment.

a. Alternate solutions have the system cache the payment criteria and resubmit once recovery is successful; before resubmitting for payment, the system will ensure the original payment was not successful to avoid duplicate charge.

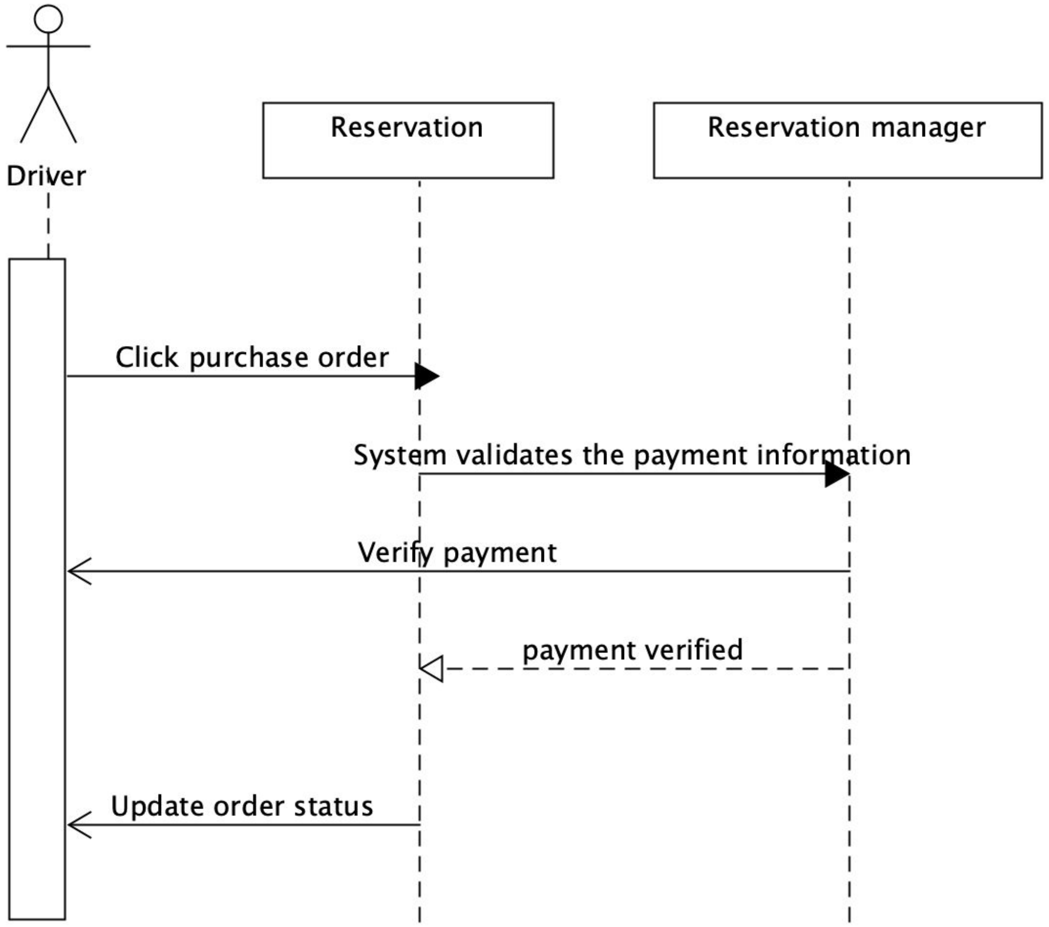
b. System to automatically refresh the system forcing the Users to resubmit.

Postconditions:

Describes the state of the system at the conclusion of the use case execution.

1. User will be able to locate the parking space of choice
2. Accessibility to price and availability of parking inventory
3. Option to pay on demand with discount or place a hold on a parking space for future payment
4. Option to update or cancel an existing reservation

Sequence Diagram



2.2.4. Update

Actor

Driver

Description

The Driver would like to extend the duration of their current parking spot reservation.

Preconditions

1. The User has an existing valid reservation for a spot that is currently in effect.

2. User has access to the internet and can access the system.

3. User knows or has access to all the information needed to reference current reservation.

4. The reservation information includes payment information for the method used to make the initial reservation.

Main Steps

1. User references their current reservation.

2. System presents available extension time options and costs.

3. User selects the desired option.

4. System validates the payment information, generates and presents confirmation.

Exceptions

EX.1. The spot is not available for the desired time - User can search for other spots but removes the vehicle from the parking spot by the originally agreed time.

EX.2. Users are unable to access their current reservation - they can create a new reservation.

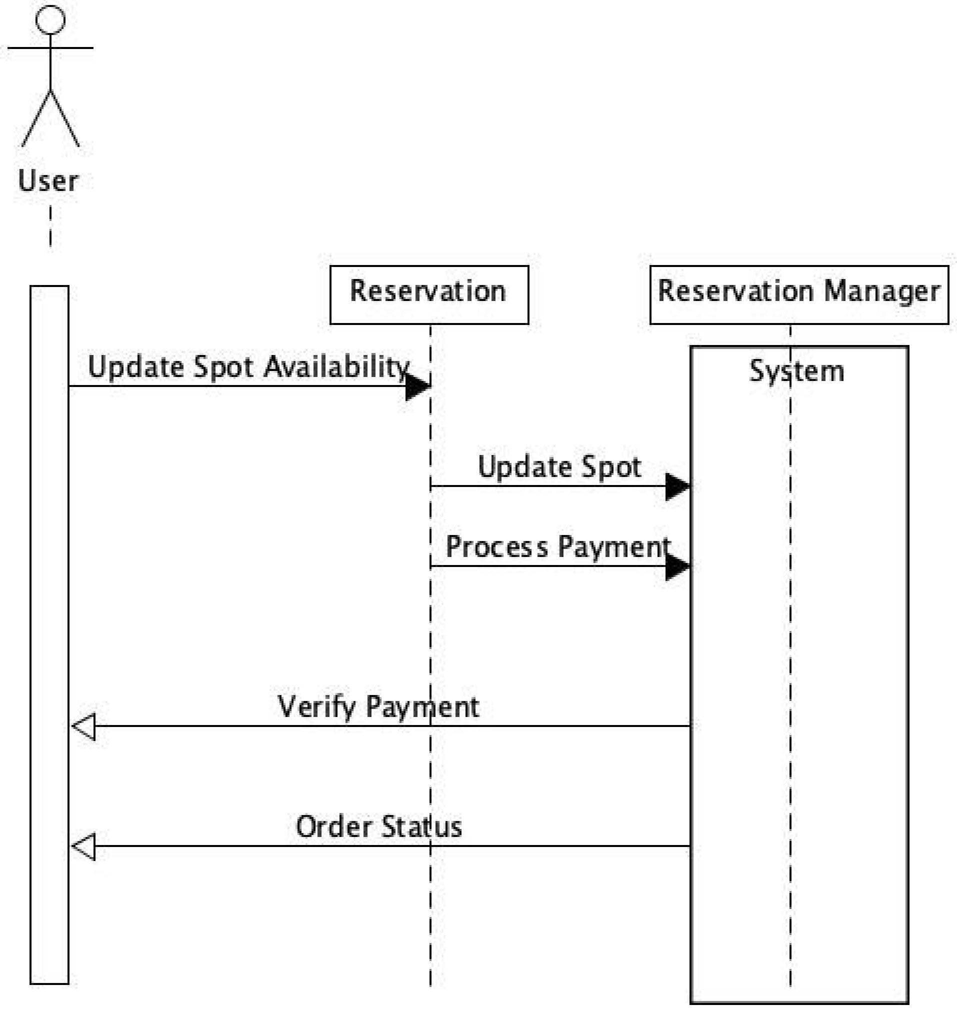
EX.3. System fails to authorize payment - User may reenter information.

Postconditions

1. User’s current reservation now has an extended end time.

2. Payment has been made for the extension.

Sequence Diagram



2.2.5. Parking

Actors

Driver

Description

Driver is going to physically utilize the desired spot they reserved and paid for.

Preconditions

1. User has already created a profile.
2. User has searched for desired parking spot.

3. User has reserved paid parking spot.
4. User has valid confirmation for reserved space.

Main Steps

1. Find parking location.
2. The User checks their reservation to confirm the spot matches their current physical spot.
3. Reservation reminds User of possible towing penalties incurred for parking under incorrect spot.
4. The User can reference their current reservation to keep track of their time left.
5. System offers support if any questions in regards to the parking location.

Alternative Courses

1. The parking does not exist.
2. The Parking is full.
3. User is not able to fulfill the reservation.
4. User proceeded to call for support.
5. User’s reserved spot has been taken by mistake by another car.

Exceptions

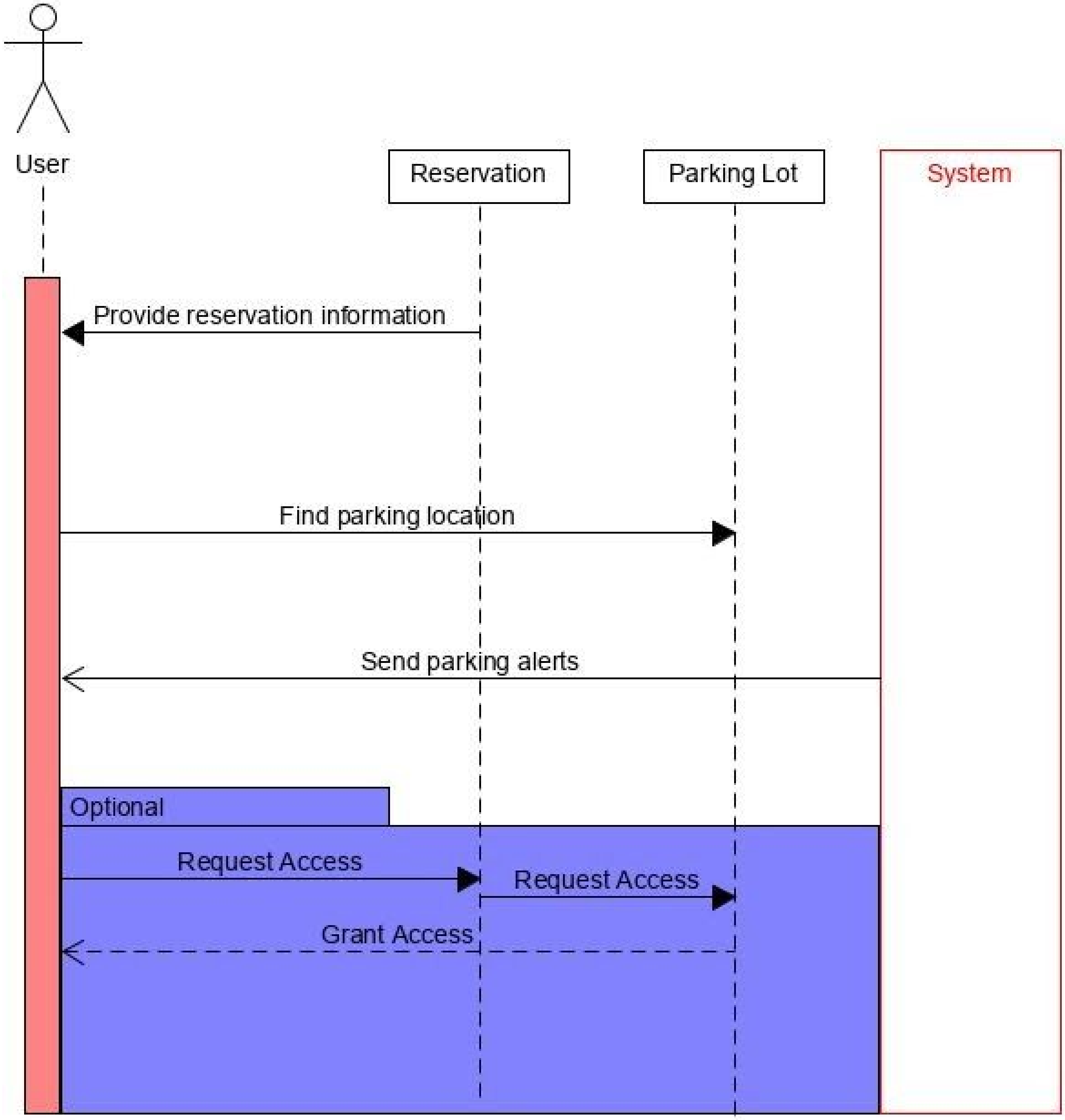
EX.1. AC.1User is notified that their current reservation is about to expire. User either extend time as needed, or conclude reservation.

EX.2. User is reminded of Tow Zoning policies and possible penalties incurred under reservation location.

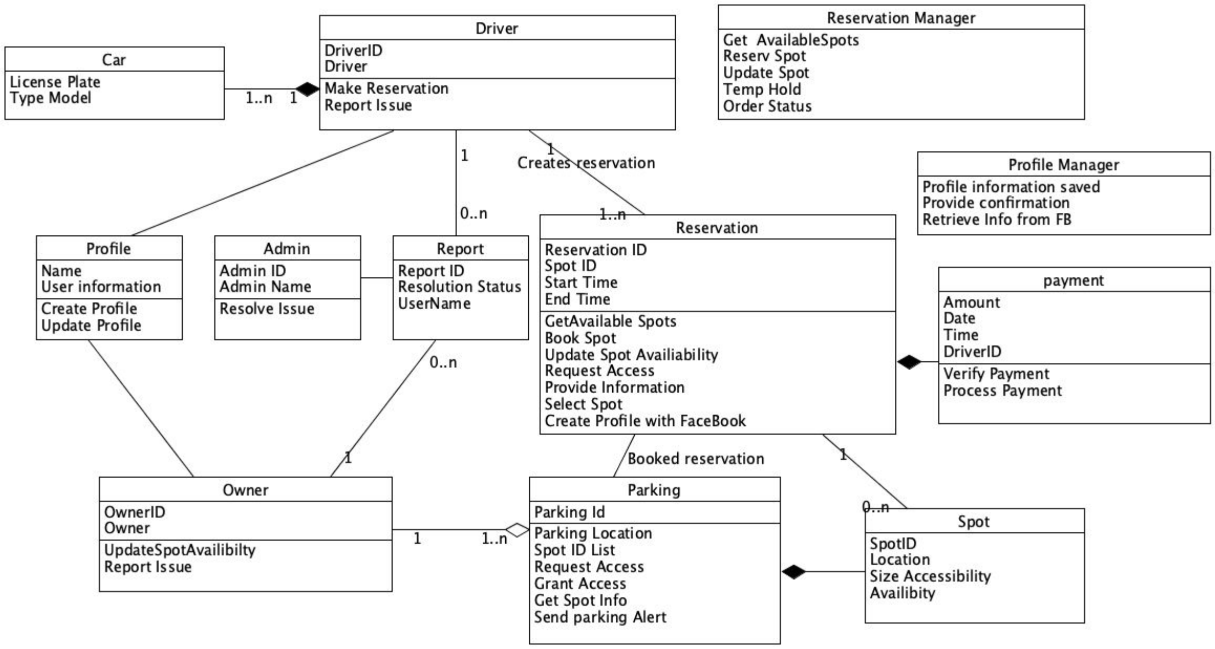
Postconditions

Reservation has concluded. Clients are parked per their reservation detailed location.

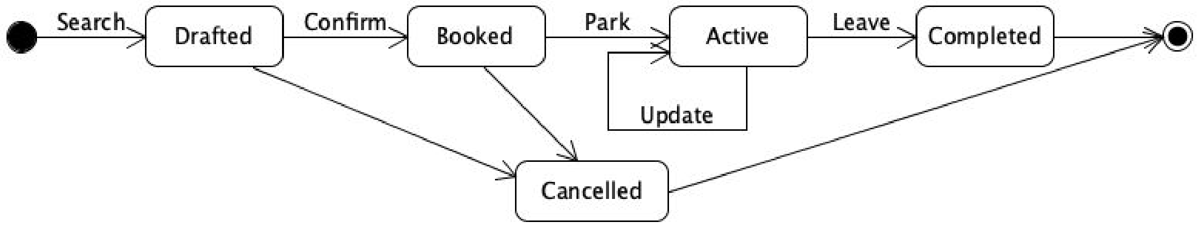
Sequence Diagram



Class Analysis Diagram



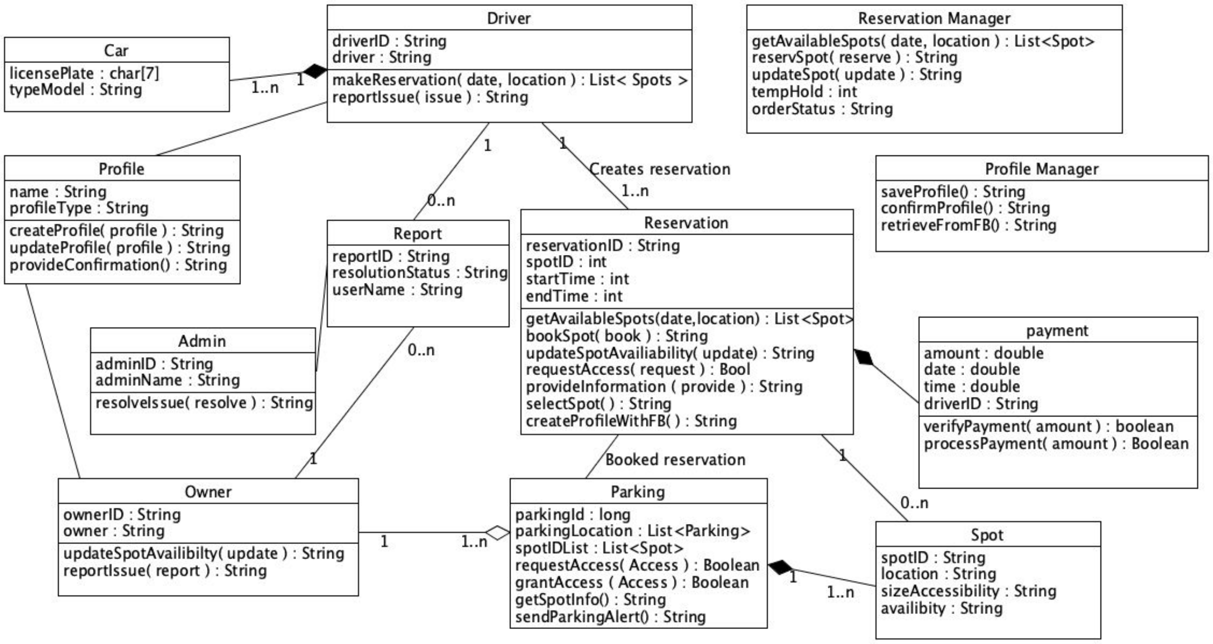
State Diagram



3. Design

This defines insights on the implementation.

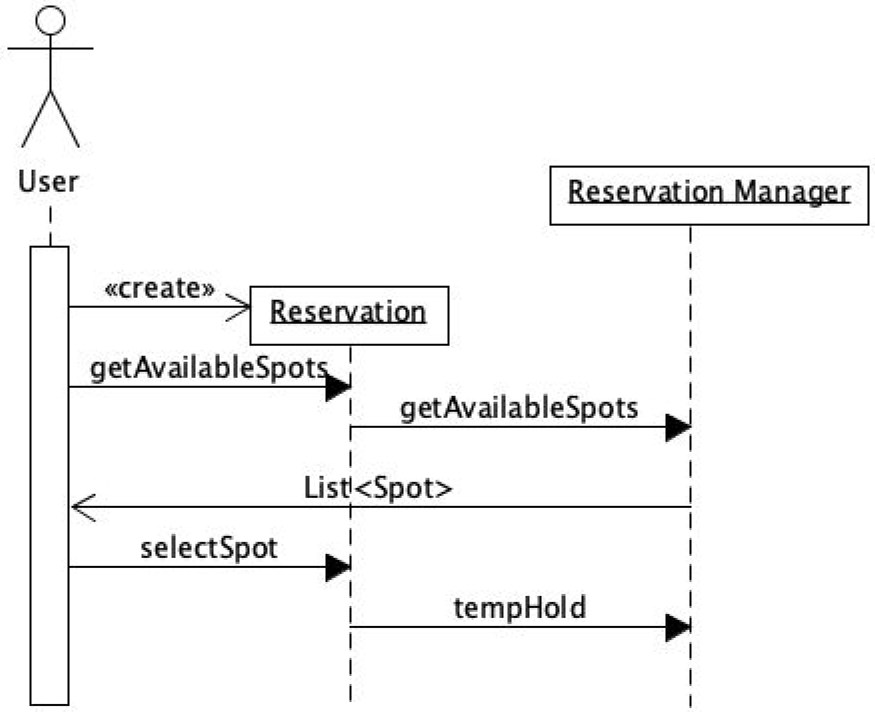
3.1. Class Design Diagram


Car
licensePlate : char[7]
typeModel : String
1..n
1
Driver
driverID : String
driver : String
makeReservation(date, location) : List< Spots >
reportIssue(issue) : String
Reservation Manager
getAvailableSpots(date, location) : List<Spot>
reservSpot(reserve) : String
updateSpot(update) : String
tempHold : int
orderStatus : String
Profile
name : String
profileType : String
createProfile(profile) : String
updateProfile(profile) : String
provideConfirmation() : String
1
0..n
1
Creates reservation
1..n
Profile Manager
saveProfile() : String
confirmProfile() : String
retrieveFromFB() : String
Report
reportID : String
resolutionStatus : String
userName : String
Reservation
reservationID : String
spotID : int
startTime : int
endTime : int
getAvailableSpots(date,location) : List<Spot>
bookSpot(book) : String
updateSpotAvailiability(update) : String
requestAccess(request) : Bool
provideInformation (provide) : String
selectSpot( ) : String
createProfileWithFB( ) : String
Admin
adminID : String
adminName : String
resolveIssue(resolve) : String
0..n
payment
amount : double
date : double
time : double
driverID : String
verifyPayment(amount) : boolean
processPayment(amount) : Boolean
1
Booked reservation
1
0..n
Owner
ownerID : String
owner : String
updateSpotAvailibilty(update) : String
reportIssue(report) : String
1
1..n
Parking
parkingId : long
parkingLocation : List<Parking>
spotIDList : List<Spot>
requestAccess(Access) : Boolean
grantAccess (Access) : Boolean
getSpotInfo() : String
sendParkingAlert() : String
1
1..n
Spot
spotID : String
location : String
sizeAccessibility : String
availibity : String

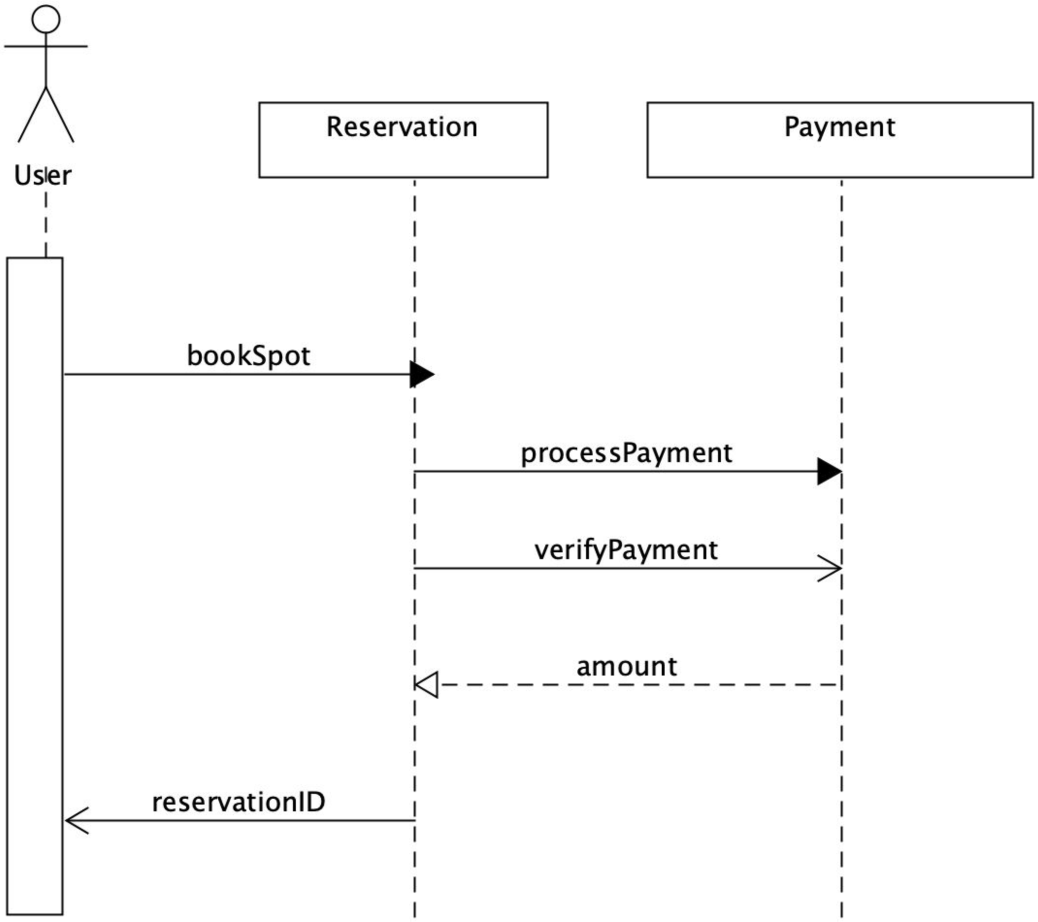
3.2. Sequence Design Diagrams

3.2.1. Profile

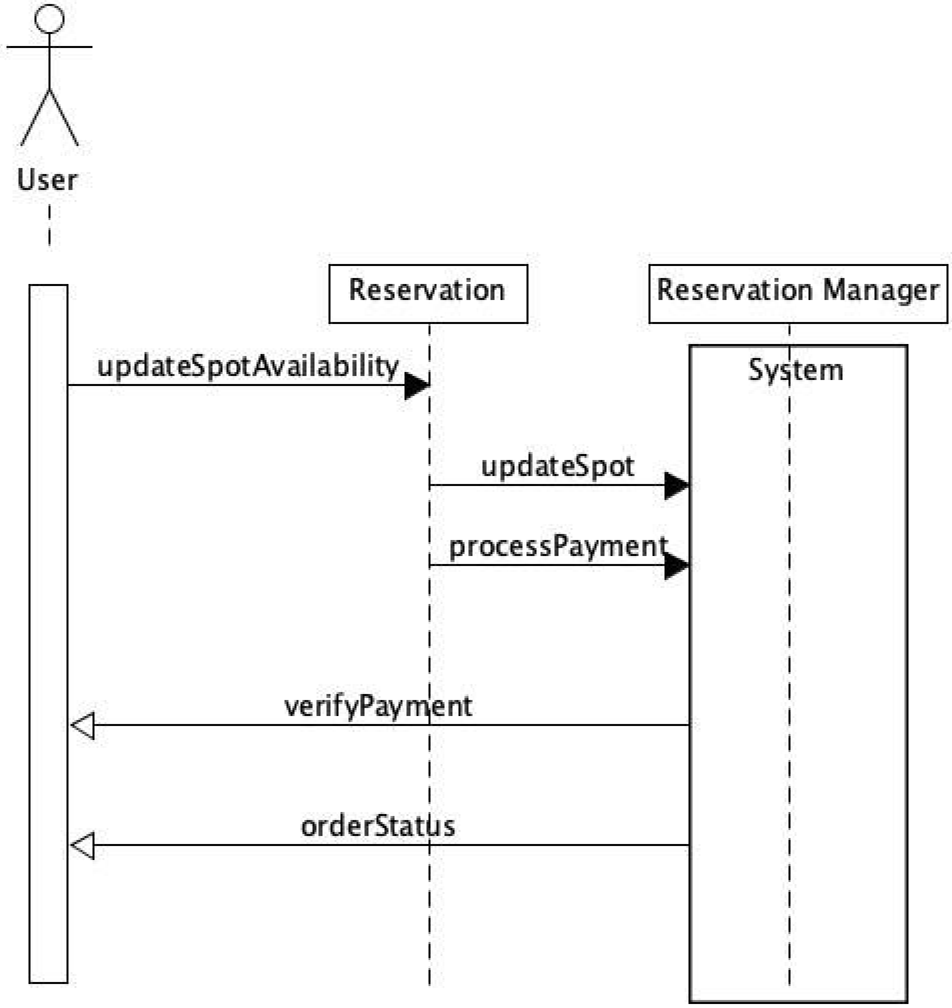
3.2.2. Search


User
Reservation Manager
«create»
Reservation
getAvailableSpots
getAvailableSpots
List<Spot>
selectSpot
tempHold

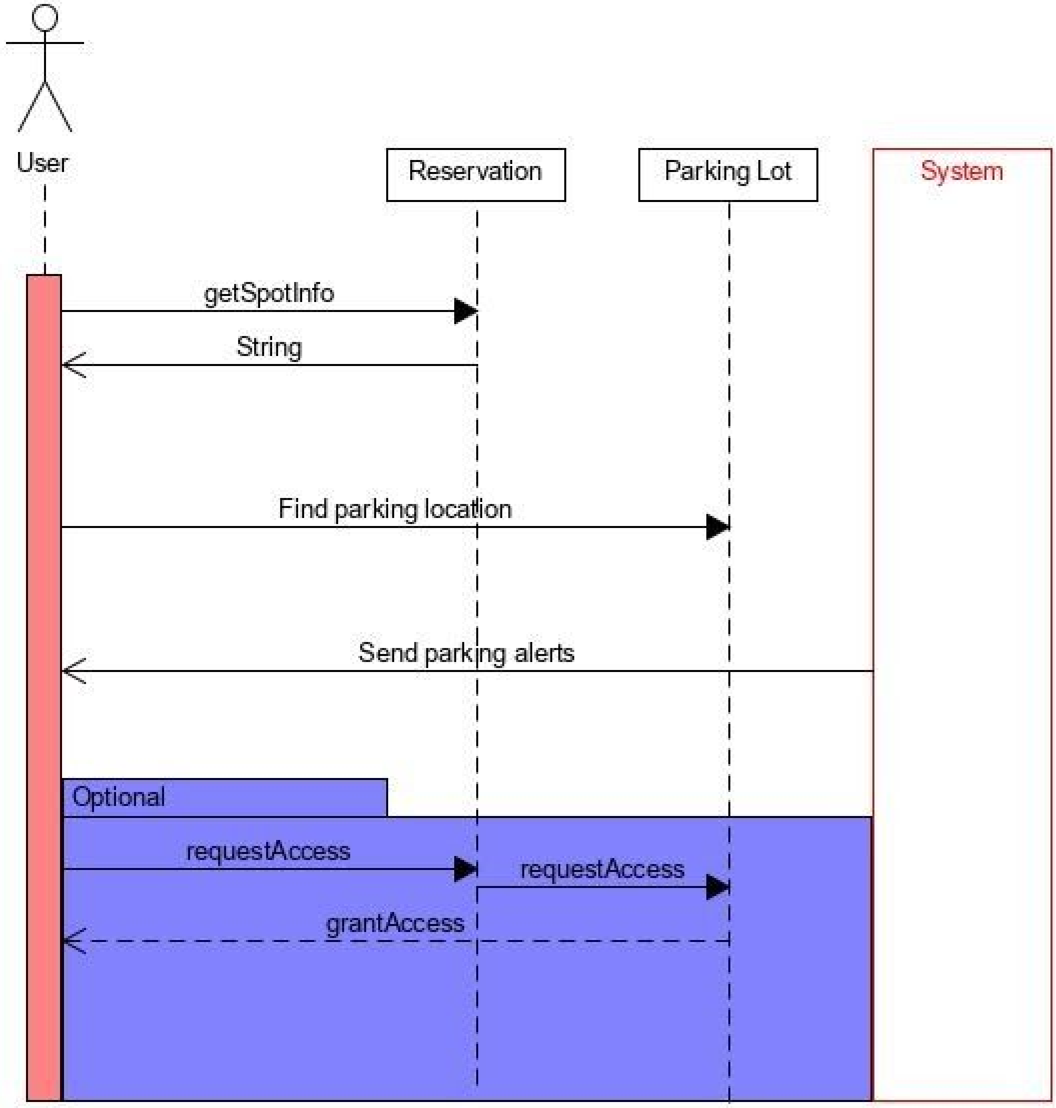
3.2.3. Reserve


User
Reservation
Payment
bookSpot
processPayment
verifyPayment
amount
reservationID

3.2.4. Update


User
Reservation
Reservation Manager
System
updateSpotAvailability
updateSpot
processPayment
verifyPayment
orderStatus

3.2.5. Parking


User
Reservation
Parking Lot
System
getSpotInfo
String
Find parking location
Send parking alerts
Optional
requestAccess
requestAccess
grantAccess

Team Collaboration

Contribution Identification

Team Member	Use case ID	Use case	Sequence diagram
Said L.	2.2.1	Create Profile	>> Reserve
Giries H.	2.2.2	Search	>> Park
Gabe P.	2.2.3	Reserve	>> Create Profile
Agnes K.	2.2.4	Update	>> Search
Bernie O.	2.2.5	Park	>> Update

Discussions

Milestone 1

Since we are fortunate enough to all be in class students we decided to meet up in person over the weekend for each milestone if possible. We all work full-time but our schedules are closely aligned, making it easy to set up preferred dates and times. If we have more members join the group this set up may need to be revisited and adjusted to fit everyone's needs.

For our project we wanted to choose something everyone in our group can relate to and has had some experience with. We considered working on the gym equipment rental system, food or food delivery system. We bounced off ideas until we reached out we all liked, which was a parking spot reservation system.

Milestone 2

Use Cases

To select the use cases we will be working on we started out very broad and tried to narrow it down. We wrote out on a white board anything we could think of that our system would be used for. We then tried to eliminate the trivial cases and combine the ones that overlap the most.

Even with what we thought was clearly defined use cases we still ran into issues with narratives overlapping each other and to make sure they are consistent with each other we needed to revisit our narratives and adjust. We did one in person meeting and a couple conference calls throughout the process.

Milestone 3

As a group we decided that a state diagram would be helpful to show the different states of a reservation. Throughout the process of a parking spot being booked, the reservation objects changes its state the most on a fairly consistent basis. First, it is drafted as a user browses and compares parking options. Once it is booked it becomes confirmed. The user redeems it by parking in the spot at the agreed time and completes the reservation. There was some discussion on how to include possible updates to an existing confirmed reservation but in the end we decided to leave it off of the state diagram because even if time is extended(if available) it remains confirmed since the user is guaranteed the availability of the spot for at least the booked time.

There was also a lot of discussion about how our class diagram should be constructed. We struggled to narrow down what each class should be responsible for and relationships between classes. In the end, drawing it out on a white board helped us agree on the best solution. As a group we listed all potential classes, the external actions and reactions of the object and narrowed it down to a structure that we could agree would make the most sense.

We learned that even this far along we still were not all on the same page. For example, we had to narrow down the definition of what constitutes a reservation. If we set a confirmation# as an attribute for reservation than we cannot have a reservation that has not yet been confirmed, such as when a user is browsing through options and has not yet booked and paid for it.

Milestone 4

The Design Phase forced us to narrow down even further roles of our classes and the interactions between objects of each class. We found quite a few inconsistencies in our diagrams. As we worked through our use cases we needed to update them accordingly. We did this by iterating through all of them as a team and addressing issues as they come up. After a few rounds of review we were able to work out most of the kinks.

After having our first drafts of design sequence diagrams, we as a team began to discuss further detail the importance of consistency. Whether this is via visual representations and if our sequence diagrams were communicating the proper events and proper details under each design.

For example, one of the use cases (Create Profile) drove our design class diagram. We saw the need to go back and continue to update the class diagrams per the demand of the designs. As a team and functionality we saw that there was a need to detail the simple but clear context from our class diagrams onto our design sequence diagrams.

Final review

Per the feedback. Discussed to improve the create profile use case. Driven by our design. See external factors. For example, added an optional block for social media.

We also discussed the multiplicity among our team members. This aspect definitely helped understand the concept a lot better.

Object Diagram

We discussed whether a state diagram would be helpful but in the end decided to not include one. A state diagram would be used to show examples of how objects are connected based on specific multiplicity numbers. Since in our system each reservation process is a separate instance of a reservation it did not seem to make much sense. Even though a user can book multiple reservations and a parking spot can be reserved multiple times (as long as the times don’t overlap) these relationships seem trivial to the system.