

Garage Automation

Software Engineering – 14:332:452

Group #10: Kendric Postrero, Gao Pan, Vatsal Pandya, Peter Luo, Yunqi Shen, Arthur Rafal, Guy Rubinstein

URL:

<https://vatsal09.github.io/Garage-Automation/>

Submission Date:

February 19, 2017

Effort Breakdown

	Vatsal	Kendric	Gao	Peter	Arthur	Yunqi	Guy
Project management	60%	20%					20%
Customer Statement of Requirements	30%	50%	20%				
System Requirements	40%	10%	10%	10%	10%	10%	10%
Functional Requirements Specification	15%	10%	15%	15%	15%	15%	15%
User Interface Specs				15%		15%	70%
Domain Analysis		50%	30%			20%	
Plan of Work	60%		40%				

Table of Contents

Group Information	1
Effort Breakdown	2
Table of Contents	3
Customer Statement of Requirements	
Statement of Goals	4
Problem Statement	4
Proposed Solution (Essay Version).....	6
Proposed Solution (List Version).....	8
Glossary of Terms	10
System Requirements	
Enumerated Functional Requirements	12
Enumerated Nonfunctional Requirements	13
On-Screen Appearance Requirements	14
Functional Requirement Specification	
Stakeholders.....	18
Actors and Goals.....	19
Use Cases	20
Use Case Diagram	21
Traceability Matrix	22
Full-Dressed Descriptions	23
System Sequence Diagrams	34
User Interface Specification	37
User Effort Estimation	43
Domain Analysis	
Concept Definitions	44
Association Definitions	46
Attributes Definitions	48
Traceability Matrix	52
System Operation Contracts	37

Plan of Work	
Progress to Date Summary	57
Roadmap of Work	58
Product Ownership	58
References	59

Customer Statement of Requirements

Statement of Goals

The objective of this project is to design a software solution to streamline the parking process in commercial parking lots, by offering an automated, ticket-less payment system with suggested parking spots. This new procedure aims to provide an efficient car flow, minimize nearby traffic, and provide a quicker customer turnaround—thereby increasing occupancy and revenue.

Problem Statement

Our customer owns a multi-level, commercial parking garage without an automated system to check for occupancy logistics or a fast-track payment processes. Additionally, the current system relies on manual inspection of occupancy by employees, and this only leads to a slow and primitive way of indicating to customers whether a spot is available. At peak times, the garage could have free spots but no way of instantly checking or displaying the information, discouraging customers from parking at their garage. In another scenario, customers might search for open parking spots only to discover that there are no available spots. The displeasure and frustration felt by the customers could deter future business. The current ticket-based payment system also adds to the congestion inside the parking lot. Each customer has to stand in a line to validate and pay their ticket, then fumble around their car to look for the ticket, and finally stretch their hands all the way out to insert the ticket. This impedes swift exits and adds to the congestion of people trying to find a parking spot. It could be detrimental to the surrounding businesses, as well as for the owner of the parking lot, if customers are discouraged by the inconveniences of finding a parking spot or dealing with the traffic congestion caused inefficiency of the parking lot system.

Proposed Solution

Essay Version

A parking garage's main purpose is to give a potential user convenience and ease of access. The increased efficiency brings in more customers and thus, allows businesses to generate more revenue. In the current system, drivers drive up to a gate, receive a ticket, and the gate unlocks; the driver pays the ticket when he wishes to leave the lot and hands off the paid ticket to leave the exit gate. Moreover, larger parking garages can result in greater inefficiency if managed incorrectly; larger lots require more time to traverse and congestion affects more consumers.

As an owner of the parking lot, the new system will be based around a website or mobile application that will give the user information about any commercial parking lot that has implemented this system. The information displayed to the user will consist of monetary cost of parking, available parking spots, and an overall recommended parking spot. To achieve this, the system uses laser sensors to keep track of available parking spaces. The sensor will determine if a car is occupying the parking spot and mark it in the system as such and when a car exits the spot, the system will know that the spot is vacant. This streamlines the traditional process of manually checking parking spaces to judge occupancy state, and thus reduces labor cost. This also reduces congestion within the parking garage and reduces the time wasted searching for a spot. When customers are faced with choosing a parking lot, they're inclined to select the one that is guaranteed to have an open parking space.

For more specification on the hardware, each parking spot will have a sensor and the entrance and exit will have a camera for analytics purposes. The sensor will give constant updates to the system about the state of the parking spot (occupied or unoccupied). The system will be aware the state of all the parking spots in the parking lot, and will display to the user a visual a map of the parking lot with parking spaces color-coded with green for occupied and red for unoccupied. There will also be a visual on the app to indicate the recommended parking spot for the user.

The system also has the added benefit of automated transactions at the turnstile. A registered user will be able to pay using a linked credit card and license plate. At the entrance, a registered user will be able to drive up to the turnstile, and an entrance license-plate-reader will quickly identify the license plate, understand that they are

registered in the system, and allow entrance. Upon exiting the parking lot, an exit license-plate-reader will identify the car's license plate, calculate expenses, and execute charges on the provided payment method.

We will have to take into account that not all users will register to this system. There are two kinds of guest users to consider. Credit card holding guest users and cash-only guest users. Credit card holding guest users that have not registered online for the system beforehand will have a different experience. Credit card holding guest users will have to approach the turnstile and swipe their credit card to be added into the system as a guest—linking their swiped credit card and analyzed license plate. The benefit that these credit card holding guest users have is that they will not have to go through the ticketing system and have an ease of exiting. For those guest users who prefer cash, have the option to still use the old ticketing system allowing them access to the parking lot without the beneficial features.

We must make some assumptions in this project in order to simplify it. If we did not, this project would be much more complicated, and we would run the risk of failing to complete at the due date. We must assume that the license-plate-reader will work with high accuracy (95%+), the registered user has an email address, a credit/debit card, and a mobile phone with GPS and internet connectivity capabilities.

One of the key benefits of this computerized system is a management-side dashboard. The web-based dashboard serves as a virtual gateway for management to check important metrics: number of parking spots utilized per day, average customer time of stay, daily/weekly/monthly/yearly revenue, and revenue trend compared to historical revenue data. The analytics can help the owner make crucial business decisions which impact the future of the business. For example, if the owner sees that the occupancy rate data suggests that there is high demand for parking in the area, then he/she can use this information to expand his/her parking lot or invest in a nearby parking lot. Likewise, analytics can also allow an owner to understand a loss in demand so that they can divert or allocate resources as they see fit. The manager has the ability to deactivate a parking spot, i.e. marking it red for the app, for repairs, cleanup, maintenance, and etc.

List Version

1. The user shall be able to register for an account online and add user information such as Name and Credit Card Information.
2. The customer shall have the option to delete their account, change information, or add information.
3. The system shall be able to report via sensors which parking spots are occupied and which are unoccupied.
4. The user shall be able to access parking lot information, such as the number of available parking spots and visual occupancy within the parking lot.
5. The system shall be able to recommend a spot to a user, based on current parking lot availabilities.
6. The system shall be able to change the recommended spot to the user if the spot is occupied by another car in transit.
7. The system shall be able to analyze a car's license plate via a license-plate-reader to determine whether or not they are a registered user in the system.
8. The system shall be able to create guest accounts by linking the license-plate-reader information with the swiped credit card by guest user.
9. The system shall be able to recognize a registered user, or guest account (if they swipe a credit card), or cash user (if they take a ticket) and process this information to open the turnstile for customers.
10. The system shall be able to calculate the cost of expense by subtracting exit time from enter time— for the customers using automatic (credit card) payment.
11. The system shall be able to charge the customer automatically upon exiting at the turnstile by analyzing the license plate and execute a charge on an account (registered or guest) linked to the system.
12. If the user is registered, the system shall save transaction history.
13. The system shall collect various user data such as the amount of users in a specific parking lot, and time they have used the parking lot.
14. The system shall have a web-based portal where the owner/management will have access to real-time occupancy data, i.e. visual indication of which spots are available and which ones are occupied.
15. The management portal should also provide an analytics portion where there will be key metric information, including the number of parking spots utilized

per day, average customer time of stay, daily/weekly/monthly/yearly revenue, and revenue trend compared to historical revenue data.

16. The manager can change the price per hour of a parking spot.

17. The manager can disable a parking space.

Glossary of Terms

Automated Transactions	A payment transaction that has been conducted electronically by calculating the time of stay and charging that amount to the credit card associated with license plate, using parking system.
Time of Stay	Duration of time between when the user enters the parking lot and when the user exits the parking lot.
Computer Vision	A software tool that automatically extracts information from an image to be used for other software applications.
Customer	The owner of the vehicle that has been driven onto the parking lot.
User/Registered User	A customer who has a registered account in the system. These users have entries and payment history stored in the database.
Guest User/Guest Account	A non-registered user of the service. These temporary users are not stored in the database entries.
Cash User	A customer who uses the ticketing system and pays for the parking in cash.
Dashboard	A control panel GUI that organizes and displays information in user-friendly manner.
GUI	A user interface that allows the user to interact with the system using images rather than text commands; acronym for graphical user interface.
Manager/management	The owner/management of the parking lot and parking system. They are able to view manager dashboard, fix rates, disable certain spots, and etc.

Mobile Application	An application designed to be used on a mobile device such as a smart phone.
Occupancy Rate	The rate at which the parking spaces are used compared to the total amount of available space.
Register	The action that a customer takes to sign up and send information needed to use the parking system. This action sends the required information to the necessary components.
Payment History	History of locations, durations, and payments for utilizing parking lot.
Sensors	A hardware device used to detect if a parking spot is taken.
Website Application	An application designed to be used on a web browser.
License-Plate-Reader	A combination of hardware (camera) to take pictures of a license plate and computer vision to detect the license plate number.
User Accessibility	Design process to allow all users to interact with the system efficiently and with ease.
Efficiency	The ability to accomplish an action without wasting time or resources. In this context, ability to process payments quickly and manage car flow within the lot.

System Requirements

Enumerated Functional Requirements

Identifier	User Story	Story Point
ST-1	As a user, I will be able to create an account on the app and add my credit card & license plate information.	8
ST-2	As a user, I can also update my information and delete information.	5
ST-3	As a user, the app will recommend me a spot to park	13
ST-4	As a user, the app will display number of spots unoccupied and a map of which spots are open, with red indicating occupied and green indicating unoccupied	5
ST-5	As a user, the app will automatically charge the credit card associated with my account, depending on the duration of the stay	8
ST-6	If I am a guest user, I will still gain the benefits of ST-5 with a guest account, but I will have to swipe my credit card at the entrance turnstile and not enjoy the benefits of ST-9	8
ST-7	As a user, I will be able to see history of my transactions, i.e. how much I paid, which date I paid, and the length of my stay	5
ST-8	As a manager, I will have an online portal where I can find key metric information, including the number of parking spots utilized per day, average customer time of stay, daily/weekly/monthly/yearly revenue, and revenue trend compared to historical revenue data	5
ST-9	As a manager, I will be able to see which spots are taken on a visual parking map	2
ST-10	As a manager, I will be able to disable parking spots (mark red on the map) for repair and maintenance	3
ST-11	As a manager, I will be able set parking prices	2

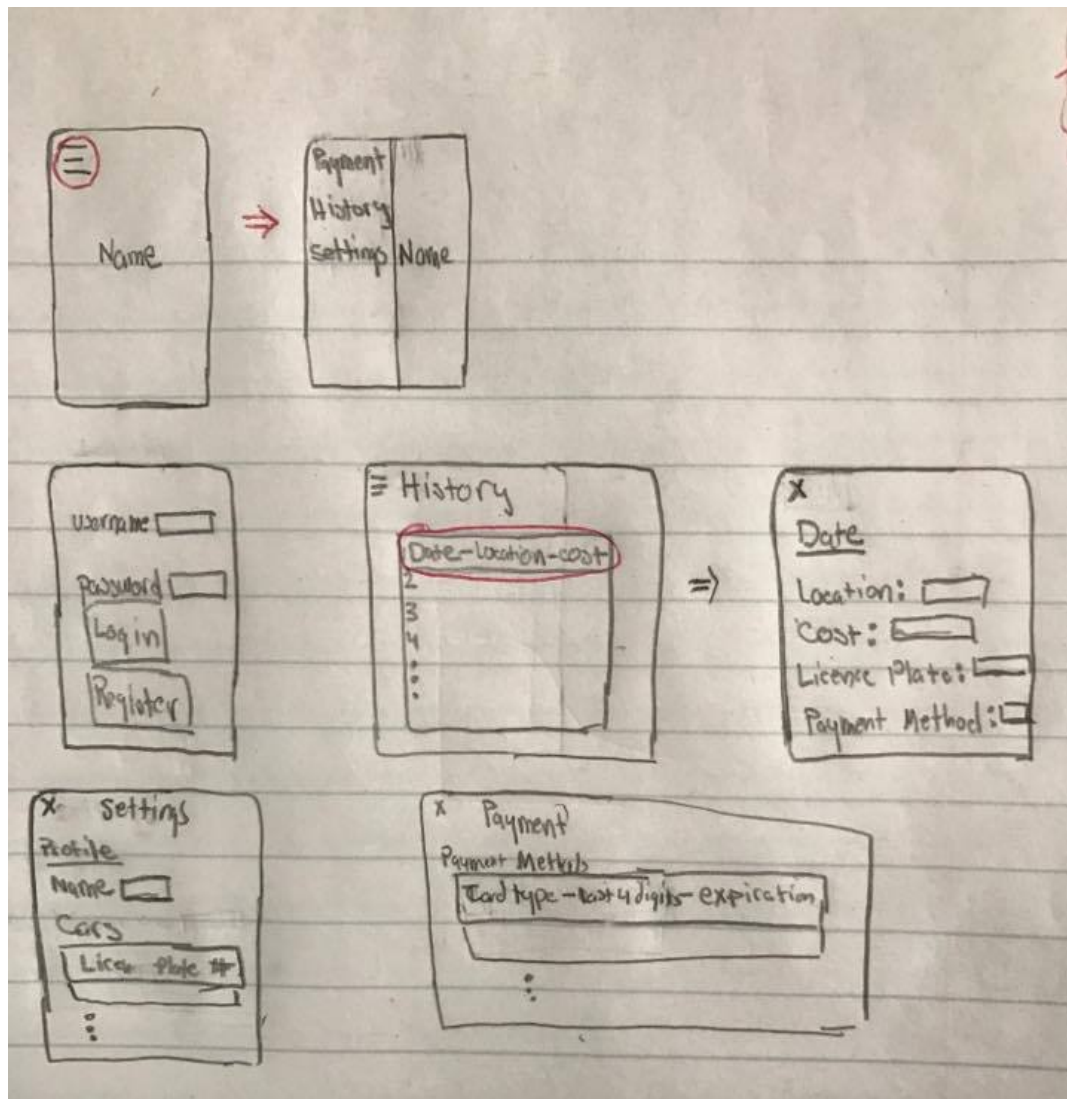
Enumerated Nonfunctional Requirements [6]

FURPS	
Functionality	<ul style="list-style-type: none">• The laser-based sensors will be able to determine whether a parking spot is occupied• Includes a camera-based system that is able to detect license plate numbers at the entrance and the exit point• For guest users, there will be a system where they can swipe a credit card at the entrance
Usability	<ul style="list-style-type: none">• There is a help dialog box on every page to provide the user with an idea of the steps they need to take for certain actions.• There is consistency as all the web pages follow the same template. To keep it simple, the site was kept to roughly five pages.• To accomplish simplicity, each page has a navigation bar to access the individual pages
Reliability	<ul style="list-style-type: none">• Database is stored in the cloud to take advantage of upgrading storage at anytime, little to no downtime, and continued lowering of cloud storage prices
Performance	<ul style="list-style-type: none">• To be efficient, the repeat customers will not have to go through the payment process themselves. Their linked account will charge the credit card associated with the license plate.• Map of the parking lot and its spaces is available to customers so that they know where open spaces are ahead of time.
Supportability	<ul style="list-style-type: none">• This design is very adaptable as far as the garage size is concerned.<ul style="list-style-type: none">○ If there are ever future plans to expand the garage additional floors or any other renovation, the adjustment would be as simple as making an update in the database table.• Maintainability is also fairly straightforward as there isn't too much customer information to deal with.<ul style="list-style-type: none">○ If there was ever a need to remove or update a customer from the database, the action would be carried out to their credit card, vehicle, and reservation information automatically

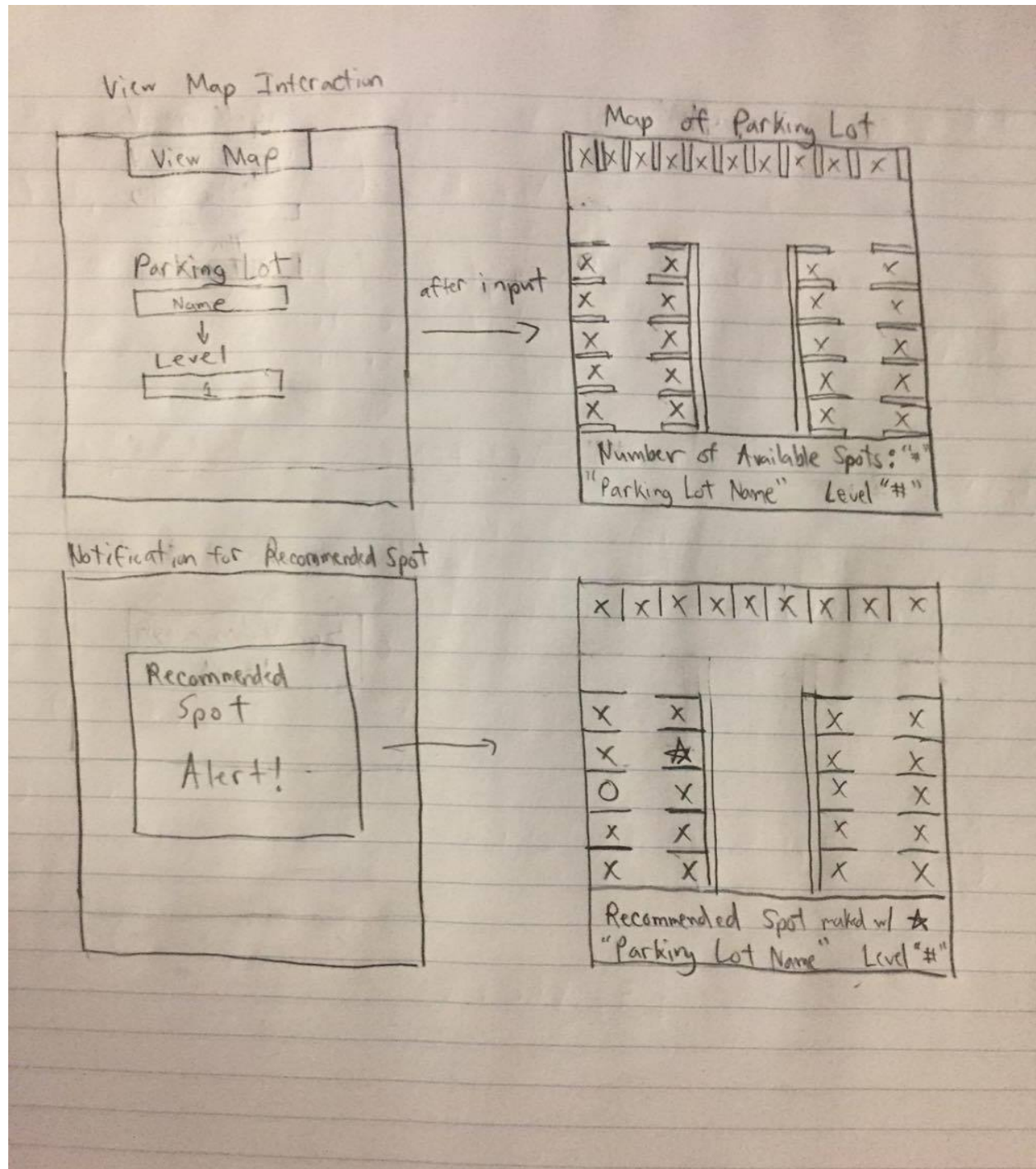
On-Screen Appearance Requirements

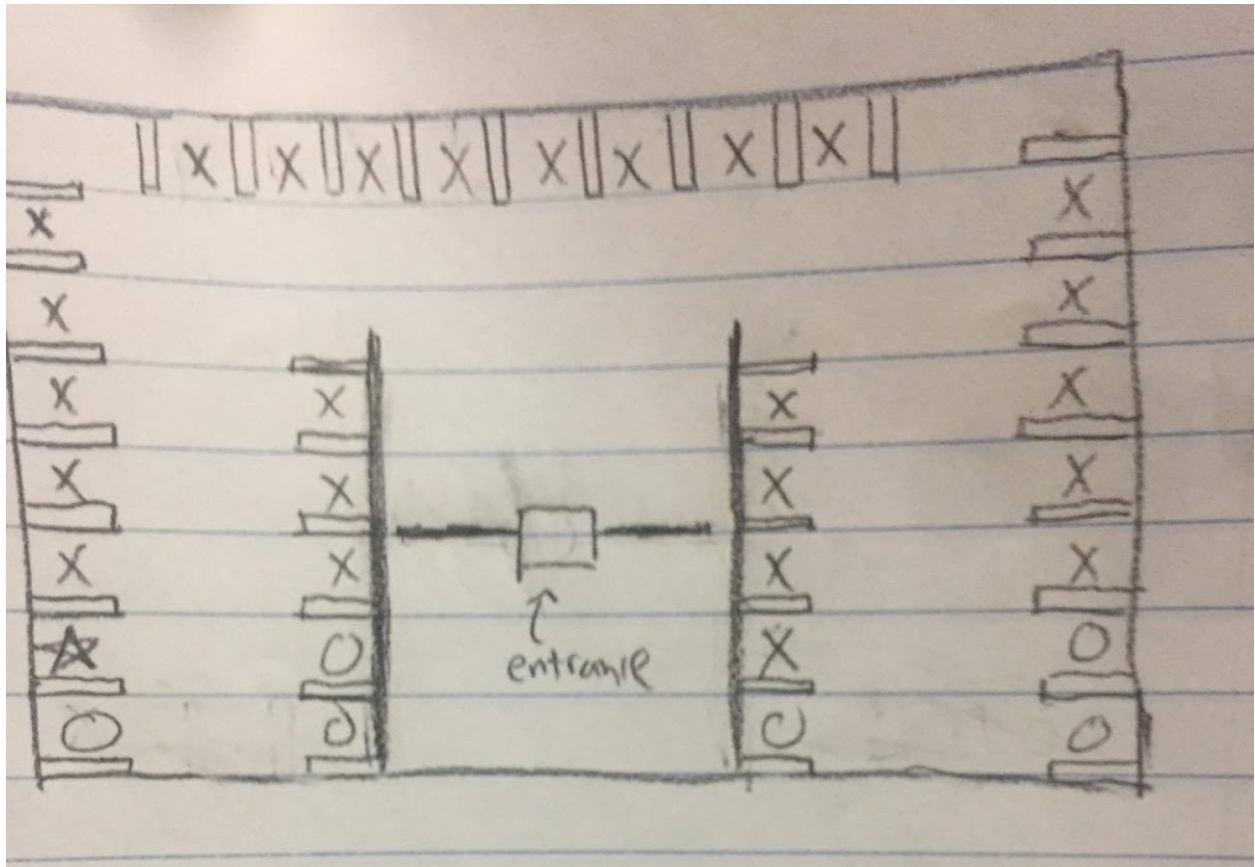
Our team is not building a product that relies on heavy graphics. Nonetheless, during the weekly meetings, we decided to create a mockup to truly understand what the user might see and how he/she will interact with the system. This gave us insight on which functions need to be added and how the team would have to subdivide the tasks. In addition, we were able better understand the design requirements of the shared API and the structural nature of the data.

Website Terminal for Registering, Adding User Information, and Viewing Transaction History



Map of the Parking Lot for the User displaying certain parking lot information





The user sees a map of the parking lot. He will see all the parking spots and information about the state of occupancy. A red X will indicate a spot is occupied. A green O will show that the space is available, and a star would show where the recommended spot is located.

Manager Dashboard

Analytics Login

Username:

Password:

Login

Past Customers & weeks Avg Stay

week 1:

week 2:

week 3:

Calendar

Heat Graph

Total profit today —

total customers —

Average length of stay —

% change month vs last month —

Past 10 Weeks Customers

Average stay of customer past weeks

Total profit ~~rest~~ & Total Customers

Calendar with Profit & customer at the day

% change in profit last month vs this month

Heat graph of amount of traffic

The manager first has to log in with the first web page. After successful authorization, the second page is shown. This page displays analytical information on the parking lot. It also provides tools for the business manager to track the transactions and build analytical reports concerning the revenue, customer behaviors, pathway traffics, so that the manager can figure out where to improve.

Functional Requirements Specification

Stakeholders

A stakeholder is someone who has interest or concern with the system-to-be. Due to the umbrella description, the system-to-be will have several types of stakeholders— all with varying degrees of stakes in the system-to-be. Individuals, teams, and organization with a stake include:

- Parking Garage Owners - sponsor of the system-to-be
- Maintenance - on-site individual whose job is inspect garage wear, monitor manager-side dashboard, and ensure everything is going smoothly
- End Users - individuals with and without accounts who will be parking their cars
- Database Manager - ensuring data is securely being updated, processed, and backed up
- Business Analyst - monitoring the manager-side dashboard to see if the revenue is aligned with projected revenue and looking for growth opportunities
- Camera and Sensor Companies - engineering the hardware components for the system-to-be
- Local business - business owners in the near proximity of the parking lot
- Payment service companies - credit card companies and banks involved with verifying, charging, and processing payment transactions

Actors and Goals

Actors	Goals
Management	To set the parking prices, monitor the management-side dashboard, disable spots when needed, and managing employees
Customer	To park the car
System	To exchange information with the application such as license plates and parking vacancy data.
Application	To allow customers to create an account, display transaction history, display recommended parking spot, display number of vacancies, and display occupied/unoccupied parking spaces
Database	To store all of the customer account information, duration of stays, and transaction history. To store parking occupancy data
Internet Site	To allow management to view occupancy statistics, revenue, and display current occupied/unoccupied parking spaces
License Plate Reader	To extract license plate number
Vacancy Display	To display the number of vacant spots within the garage
Turnstile	To stop the car
Sensors	To find out whether a spot is vacant or not
Credit Card Machine	Terminal to accept credit card payments

Use Cases

Casual Description

Use Case	Name	Description
UC-1	Register	Customer registers for an account online, providing information including a payment method
UC-2	Arrive	Customer pulls up to the turnstile
UC-3	Remove Car	Customer removes a car from his/her account
UC-4	Delete Account	Customer deletes account
UC-5	Add/Update User Information	Customer adds/updates his address, credit card, or car information
UC-6	Recommended Spot	Customer is given a recommended parking space where the customer should park.
UC-7	Exit	Customer exits the garage via exit gate
UC-8	History	Customer views their payment transaction history
UC-9	Check Occupancy	Customer checks the a map of the parking lot for which spaces are occupied/unoccupied and how many spots are open
UC-10	Change Price	Manager can adjust prices for parking
UC-11	Analytics	Manager can view data statistics about parking lot revenue and logistics, and detailed payment history
UC-12	Remove Payment Method	Customer deletes a payment method
UC-13	Change Spot Availability	Manager can manually change the availability of parking spots

Use Case Diagram



Traceability Matrix

Use Cases	Register	Arrive	Remove Car	Delete Account	Add/Update User Information	Recommend Spot	Exit	History	Check Occupancy	Change Price	Analytics	Remove Payment Method
Requirements												
Scan license plate		X					X					
Recognize registered customers		X										
Display number of available spots						X			X			
Display a map of which spots are occupied/unoccupied									X			
Create user account	X	X										
View past transactions								X				
Update user information					X							
Delete user information			X	X								X
Create temporary account		X		X								
Automatic payment		X					X					
Manager Analytics									X	X	X	
Total PW	5	23	5	13	5	10	15	5	17	3	7	5

Fully-Dressed Descriptions

Use Case UC-1: Register

Related Requirements: 1, 6, 9, 11

Initiating Actor: Customer

Actor's Goal: Create an Account that will be stored in the database and allow for faster entry.

Participating Actors: Application, Database

Preconditions: The application will request all required information.

Postconditions: The user's account will be stored in the database.

Flow of events for Main Success Scenario:

- 1. Customer accesses the website and chooses the "Register" option.
- ← 2. The application returns the display that states the required information (Username, Password, Name, Phone Number, Payment Method).
- 3. The customer fills out the required data fields (listed in step 2).
- ← 4. The application verifies the information provided.
- 5. Information is stored into the database and the customer is redirected to the login page.

Flow of Events for Extensions (Alternate Scenarios):

- 5a. The information provided was not valid
- ← Same as in Step 2 above.

Use Case UC-2: Arrive

Related Requirements: 7, 8, 11, 12, 13

Initiating Actor: Customer

Actor's Goal: Get entrance to the parking lot

Participating Actors: System, Application, Database

Preconditions: Customer arrives in a vehicle with a licence plate and a valid credit/debit card registered to an account.

Postconditions: Customer's vehicle will be registered to his account and the turnstile will rise.

Flow of events for Main Success Scenario:

- 1. Customer approaches the turnstile.
- ← 2. The system will scan the license plate
- ← 3. The system will request the a payment method
- 4. The customer will swipe a credit/debit card
- ← 5. If the provided credit/debit card is registered with a valid account, the car will be registered to that account in the database and System raises the turnstile.

Flow of Events for Extensions (Alternate Scenarios):

3a. The license plate is registered to an account

- ← 1. Same as in Step 5 above.

4a. The provided credit/debit card is invalid

- ← 1. Application detects invalid payment and signals System, System signals Customer.
- 2. Customer will swipe a valid credit/debit card
- ← 3. Same as in Step 5 above.

4b. The customer only has cash

- 1. Customer presses button
- ← 2. System signals Application to mark license plate as cash user
- ← 3. System prints ticket
- 4. Customer takes ticket
- ← 5. System raises turnstile

5a. The credit/debit card is not registered with a valid account

- ← 5. Vehicle data and credit/debit card are temporarily stored together in the database and System raises the turnstile.

Use Case UC-3: Remove Car

Related Requirements: 16

Initiating Actor: Customer

Actor's Goal: Remove a car/license plate from the account.

Participating Actors: Application, Database

Preconditions: The user has an account with a car listed.

Postconditions: The user's account will have one or more cars removed.

Flow of events for Main Success Scenario:

- 1. Customer accesses the website and goes to Settings Page
- ← 2. The application returns the customers account information, including the cars registered to their account.
- 3. The customer presses the x button next to the car they want to remove.
- ← 4. The application requests the customer to confirm this action
- 5. Customer confirms decision
- ← 6. Car information is deleted from database.

Flow of Events for Extensions (Alternate Scenarios):

- 5a. Customer declines decision
- ← 6. The application closes the confirmation window and does nothing.

Use Case UC-4: Delete

Related Requirements: 16

Initiating Actor: Customer

Actor's Goal: Remove account data from the database.

Participating Actors: Application, Database

Preconditions: The application will request all required information.

Postconditions: The user's account will be removed from the Database.

Flow of events for Main Success Scenario:

- 1. Customer accesses the website and logs in.
- ← 2. The system returns the display that states their account information.
- 3. The customer presses the "Delete Account" button in the Settings page.
- ← 4. The Application presents a warning that this action is irreversible,
- 5. Customer confirms decision
- ← 6. The Application logs the user out and requests the data removed from the Database.

Flow of Events for Extensions (Alternate Scenarios):

- 5a. Customer declines decision
- ← 6. The application closes the confirmation window and does nothing.

Use Case UC-5: Add/Update User Info

Related Requirements: 14, 15

Initiating Actor: Customer

Actor's Goal: Add a new payment option or update personal info (name, email, car plates).

Participating Actors: Application, Database

Preconditions: The user has an account.

Postconditions: The user's account will have updated info or a new payment option.

Flow of events for Main Success Scenario:

- 1. Customer accesses the website and chooses the "Add Payment" option.
- ← 2. The system returns a form for the user to fill in.
- 3. The customer enters payment info.
- ← 4. The system validates the payment info, if invalid return to step 2 if valid, proceed.
- 5. Payment information is added to database.

Flow of Events for Extensions (Alternate Scenarios):

- 1. Customer accesses the website and chooses the "Settings" option.
- ← 2. The system returns current information.
- 3. The customer presses edit and fills in the fields.
- ← 4. The system validates new info, if invalid return to step 2 if valid, proceed.
- 5. Updated information is added to database.

Use Case UC-6: Given a recommended a Parking Space

Related Requirements: 4, 5

Initiating Actor: Customer

Actor's Goal: Finds a suitable parking space to recommend to a Customer by searching for the closest available parking spot to the Ground Level.

Participating Actors: System, Database, Customer

Preconditions: The parking lot has enough parking spaces to give to the user even with cars in transit possibly taking the user's recommended spot.

Postconditions: The user will be in their recommended parking spot, and the system will recognize the changes in occupancy state.

Flow of events for Main Success Scenario:

- 1. The customer successfully goes through UC-2 Arrive.
- ← 2. The system gives the customer a recommended parking spot via a notification on the application.
- 3. The customer accesses the application to view the recommended parking spot.
- 4. The customer goes to the recommended parking spot.
- ← 5. The sensor indicates that the parking spot is taken and the system updates that information.

Flow of Events for Extensions (Alternate Scenarios):

- 4a. The customer attempts to go the recommended parking spot, but someone took it instead.
- ← 5. The system researches the parking lot and gives you another parking spot.
- 6. The customer goes to the new recommended parking spot.
- ← 7. The sensor indicates that the parking spot is taken and the system updates that information.

Use Case UC-7: Exit

Related Requirements: 6, 7, 8, 9, 10, 11

Initiating Actor: Customer

Actor's Goal: To exit a parking lot.

Participating Actors: System, Database

Preconditions: The customer has a valid credit card and debit card that was verified on the Arrive UC-2.

Postconditions: The customer has successfully left the parking lot and was charged for their stay.

Flow of events for Main Success Scenario:

- 1. The customer approaches the turnstile.
- ← 2. The system will scan the license plate
- ← 3. The system will find a user account linked to license plate
- ← 4. The system will calculate and execute payment with credit card/debit card linked to user based on current parking lot rate, and save the transaction to the user's history.
- ← 5. System raises the turnstile.

Flow of Events for Extensions (Alternate Scenarios):

3a. The license plate is registered to as a guest account

- ← 4. The system will calculate and execute payment with credit card/debit card linked to guest user based on current parking lot rate.
- ← 5. The system will delete guest user information from database.
- ← 6. System raises the turnstile.

3b. The license plate is registered as a cash user

- ← 4. System requests ticket receipt
- 5. Customer inserts ticket receipt
- ← 6. System raises the turnstile

Use Case UC-8: View History of Transactions

Related Requirements: 11

Initiating Actor: Customer

Actor's Goal: To view a history of transactions for the user.

Participating Actors: System, Database, Application

Preconditions: The system has saved a list of previous transactions for the user.

Postconditions: The user will have viewed information of the transaction history.

Flow of events for Main Success Scenario:

- 1. The customer successfully logs in their account on the website.
- 2. The customer goes on in the view transaction history tab.
- ← 3. The system accesses the database for the collected history.
- ← 4. The system displays the data of the collected history
- 5. The customer views the transaction history data.

Use Case UC-9: Check Occupancy of Parking Lot

Related Requirements: 2, 3

Initiating Actor: Customer

Actor's Goal: View display of a map to show available parking spots.

Participating Actors: System, Database, Application, Sensor

Preconditions: The user has an account.

Postconditions: The user will have viewed any necessary parking lot information

Flow of events for Main Success Scenario:

- 1. The customer successfully logs in their account on the app.
- 2. The customer goes in the view parking lot map tab.
- 3. The customer inputs the specific parking lot and level they want to view
- ← 4. The system accesses the database for that parking lot and outputs a display of the parking lot.
- 5. The customer views the map of the specific parking lot level where they are shown the state of all the parking spaces at the level.

Use Case UC-10: Change Price

Related Requirements: 13,14,15

Initiating Actor: Management

Actor's Goal: Change the Price/Hour of the parking lot

Participating Actors: App,Credit Card Machine

Preconditions: The correct credentials have been provided to prove the user has authority to change the price

Postconditions: The price of the parking lot will be updated

Flow of events for Main Success Scenario:

- 1.User is presented the login page and he/she enters the account credentials
- ← 2.The credentials are accepted and the analytic page is displayed
- 3.User clicks on the edit price button and enters the new price that he/she wants to decide.

Flow of Events for Alternate Scenario:

- 1.User is presented the login page and he/she enters the account credentials
- ← 2.The credentials are incorrect and the analytics page is not displayed.

Use Case UC-11: Analytics

Related Requirements: 13,14,15

Initiating Actor: Management

Actor's Goal: Analyze the parking lot's statistics

Participating Actors: Database, Internet Site, Management

Preconditions: The required information has been attained by the parking lot system and stored in the database.

Postconditions: The pertinent information will be displayed in an easy to understand format.

Flow of events for Main Success Scenario:

- 1. User is presented the login page and he/she enters the account credentials
- ← 2. The credentials are accepted and the analytic page is displayed
- 3. User takes in the information displayed and changes the range of information timeline where allowed. This includes sections such as the revenue, average customers, and a map of the parking lot with current availability status.

Flow of Events for Alternate Scenario:

- 1. User is presented the login page and he/she enters the account credentials
- ← 2. The credentials are incorrect and the analytics page is not displayed.

Use Case UC-12: Remove Payment Method

Related Requirements: 17

Initiating Actor: Customer

Actor's Goal: Remove a payment method from the account.

Participating Actors: Application, Database

Preconditions: The user has an account with more than one payment method.

Postconditions: The user's account will have one payment method removed.

Flow of events for Main Success Scenario:

- 1. Customer accesses the website and chooses the "Remove Payment" option.
- ← 2. The system returns the list of available payment options. If only one payment option displays "Add new payment option before deleting."
- 3. The customer checks which payment option to remove.
- ← 4. The system deletes the payment option, if none selected returns same page(step 2), if selected go to step 5.
- 5. Payment information is deleted from database.

Use Case UC-13: Change Availability of Spots

Related Requirements: 17

Initiating Actor: Manager

Actor's Goal: Mark certain spots as unavailable/available in case of emergency of special conditions.

Participating Actors: Application, Database

Preconditions: There are spots marked as available in the database.

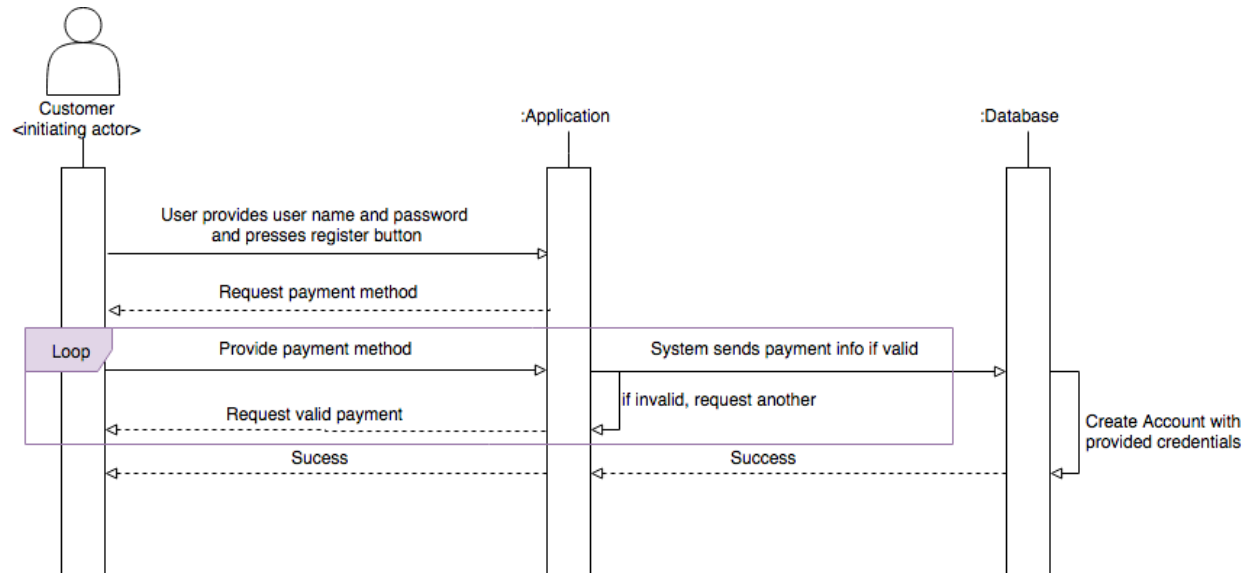
Postconditions: The desired spots are marked as unavailable (or back to available) to users.

Flow of events for Main Success Scenario:

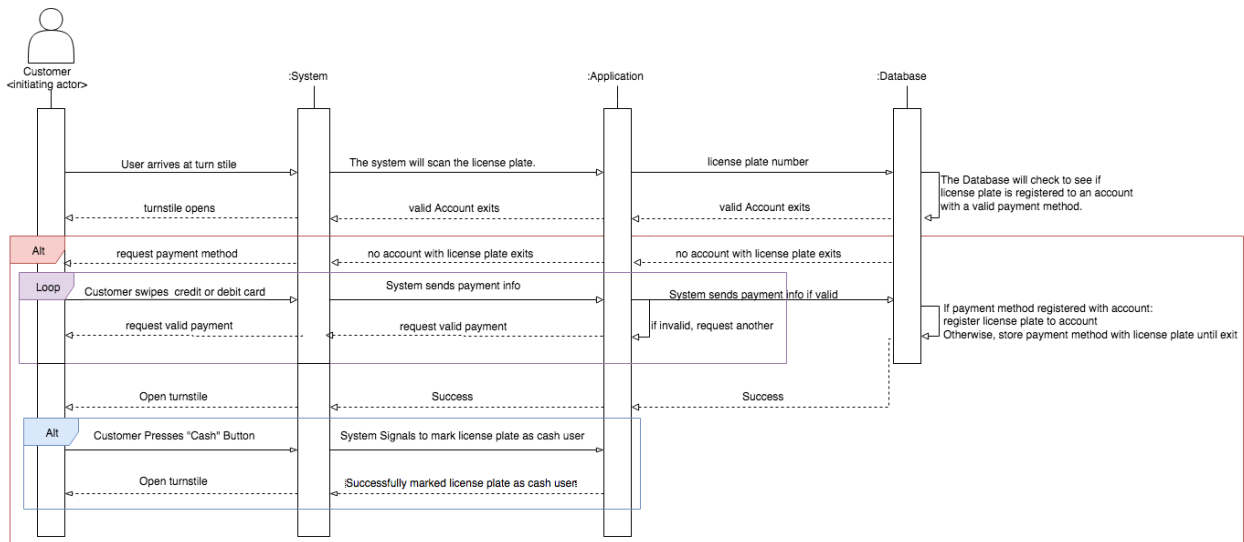
- 1. The manager (Admin) accesses the website and chooses the "Change Spot Availability" option.
- ← 2. The system returns the map of all spots and the availability of each.
- 3. The manager marks desired spots as "unavailable" (or "available"), with an option to add a remark displayed to users.
- ← 4. The system marks those spots as unavailable/available in database.

System Sequence Diagram

Use Case: UC-1

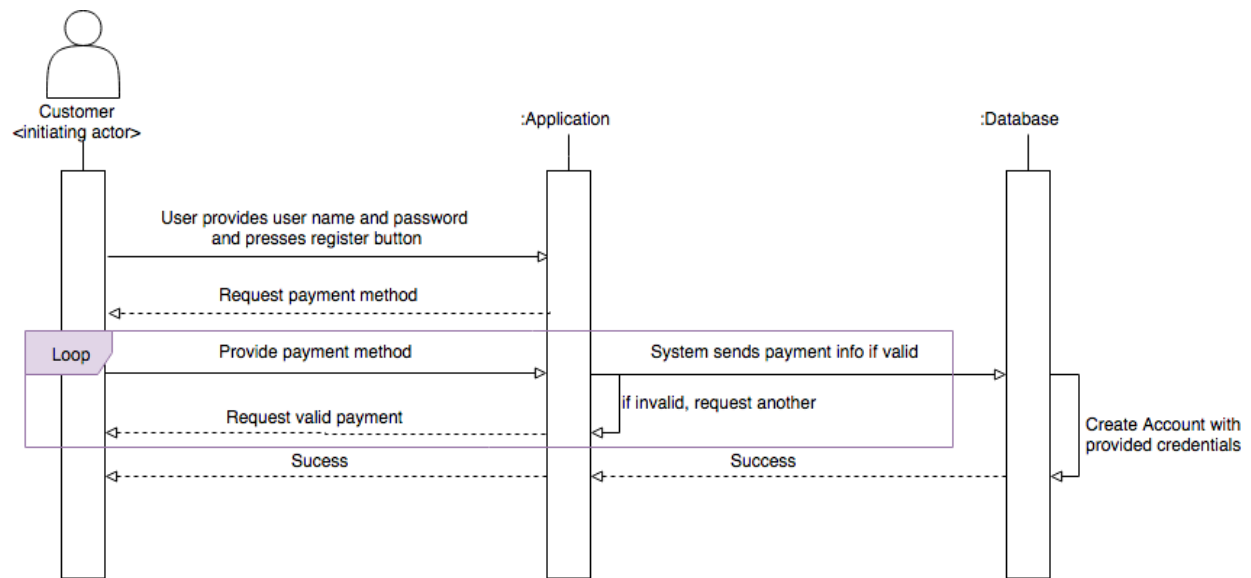


Use Case: UC-2

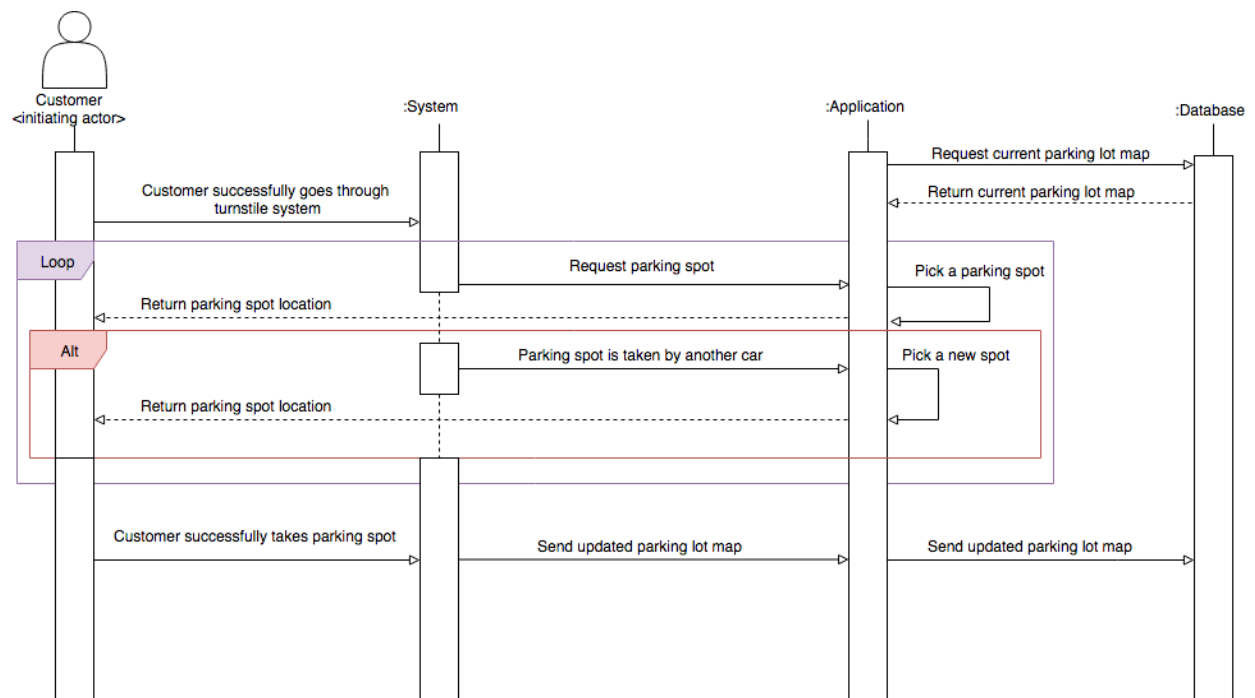


Please note that the diagrams are difficult to scale into the pdf. Please look in the "Diagram" folder in the GitHub repository for a high resolution image.

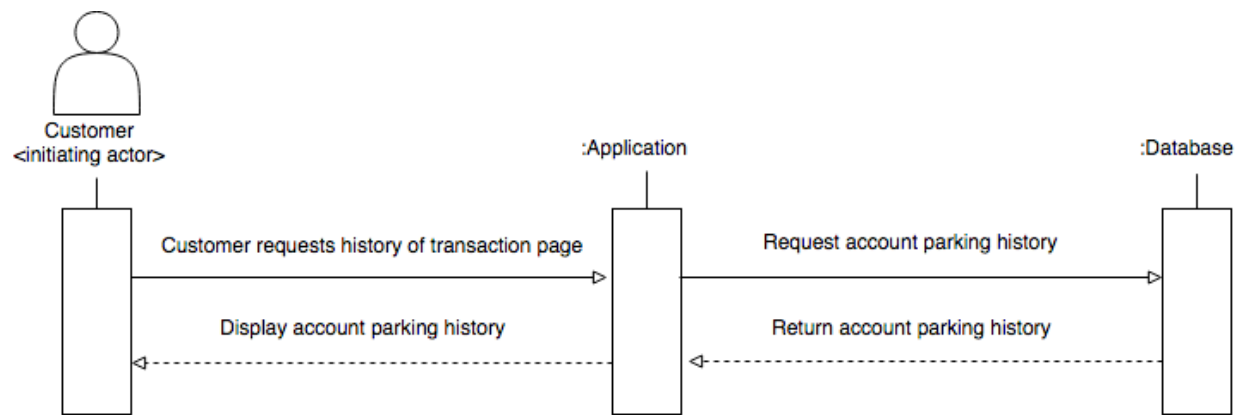
Use Case: UC-3



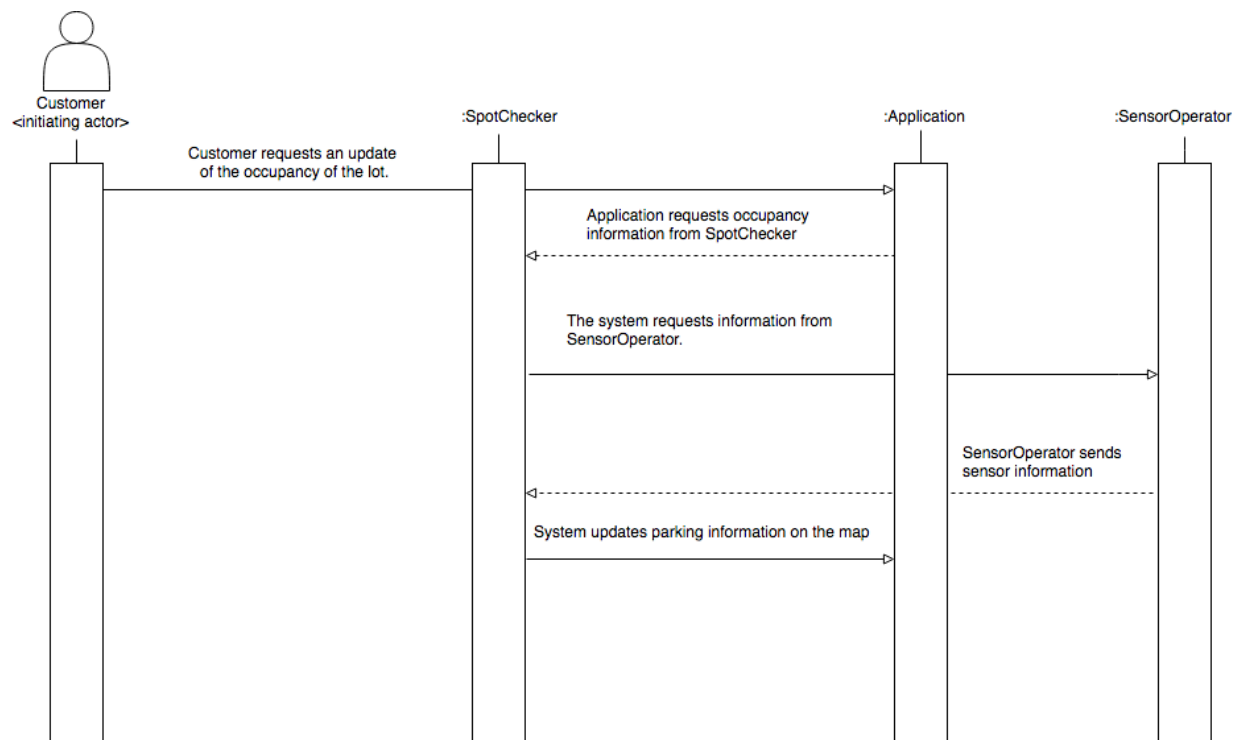
Use Case: UC-6



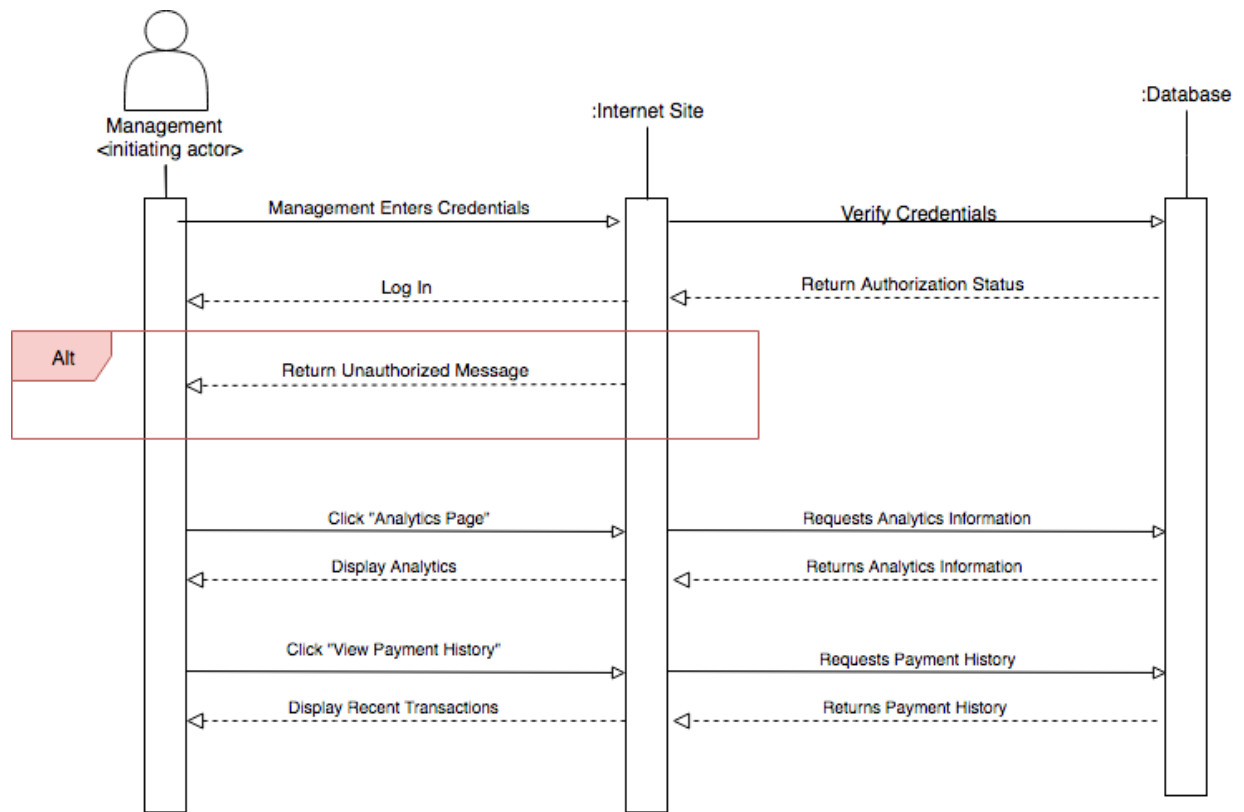
Use Case: UC-8



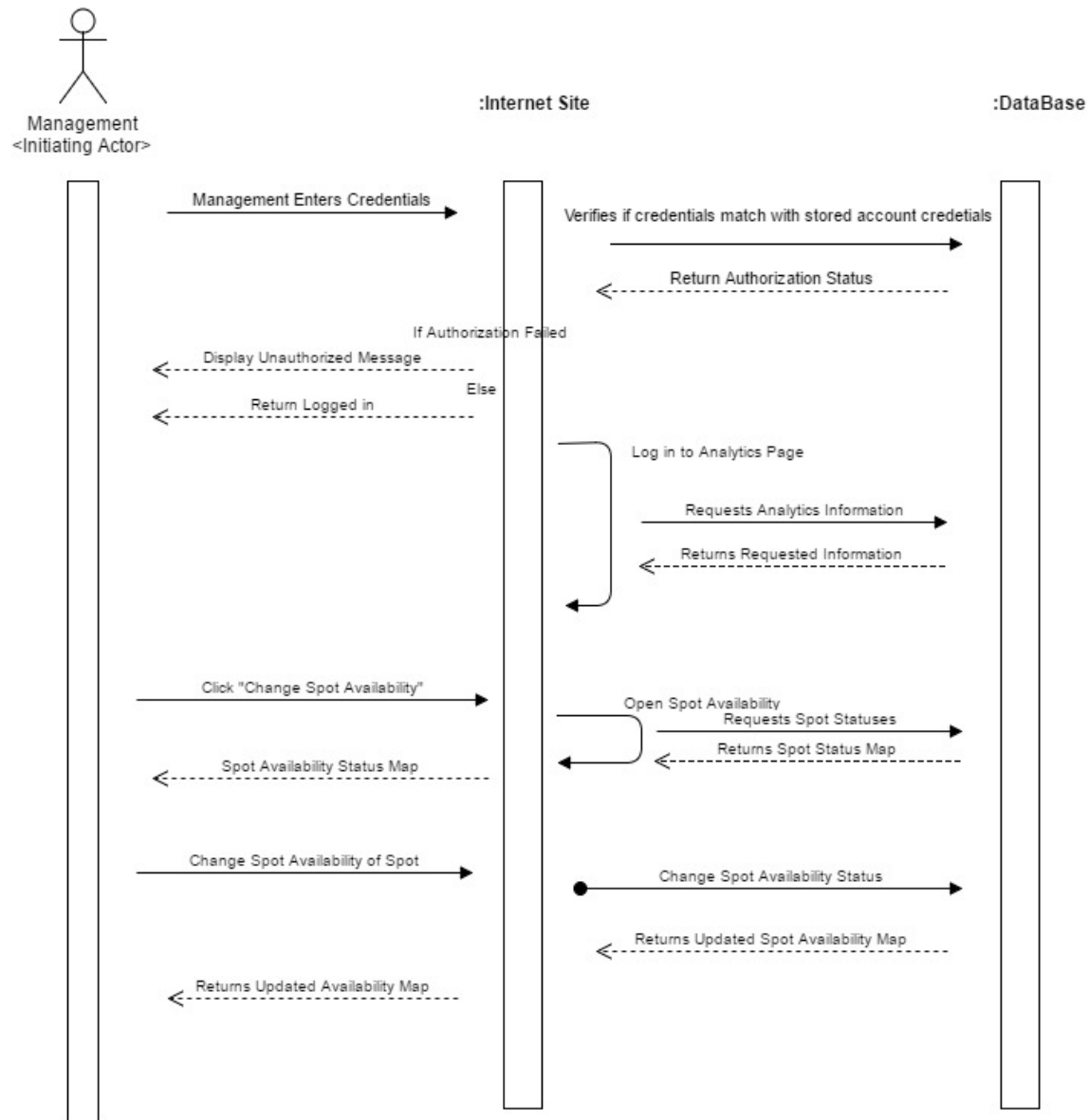
Use Case: UC-9



Use Case: UC-11



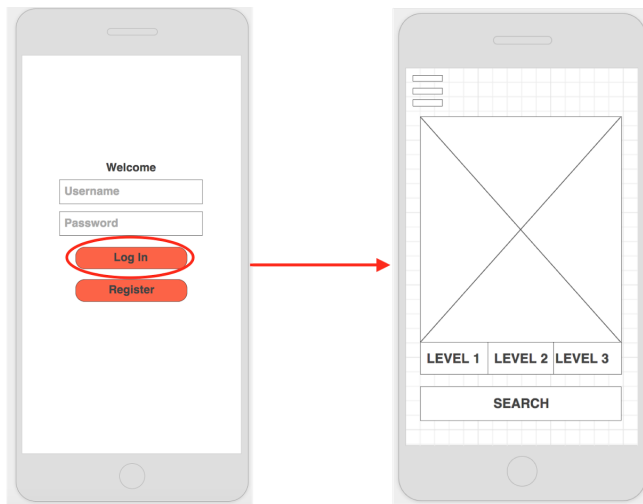
Use Case: UC-13



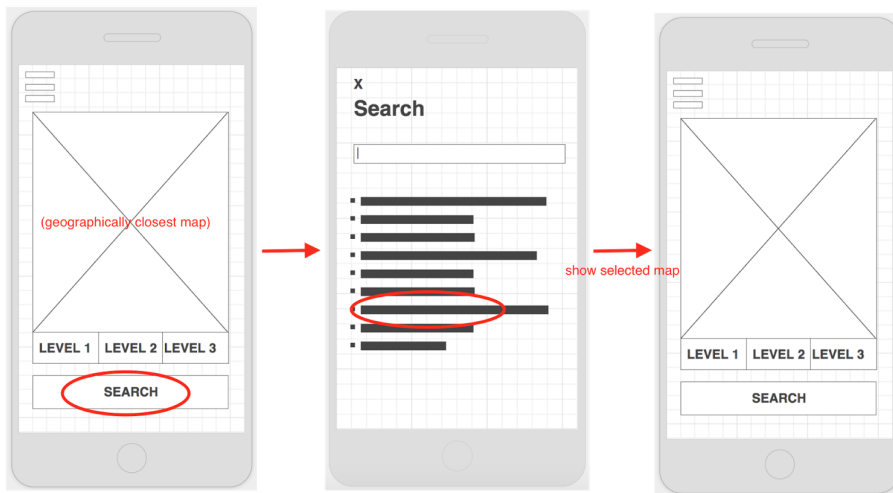
User Interface Specification

User Preliminary Design

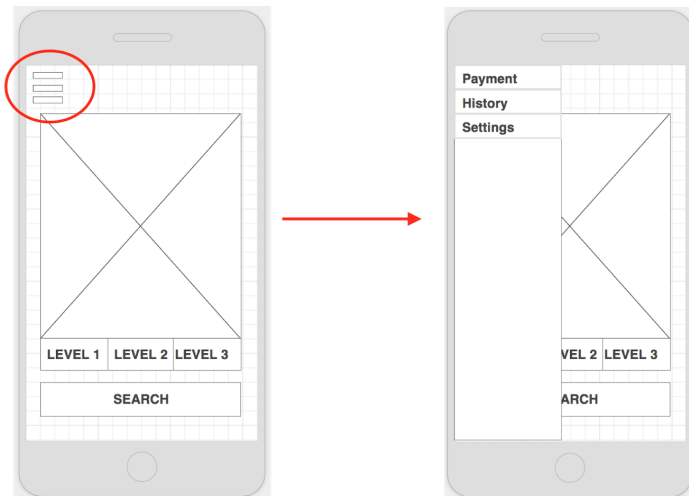
Application Functionality



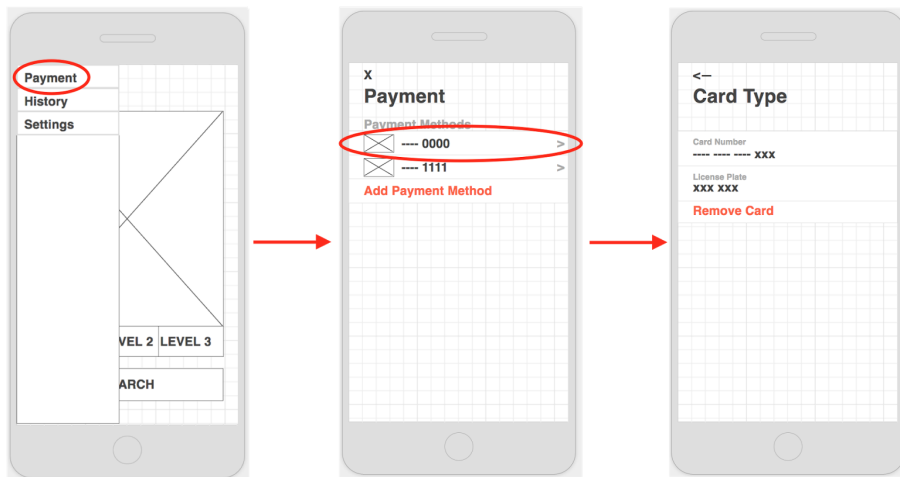
The user can log in or register from the Welcome page. When registering, the user will be required to provide a payment method. Then, will be directed to the home page. The home page displays the map of the geographically closest parking lot.



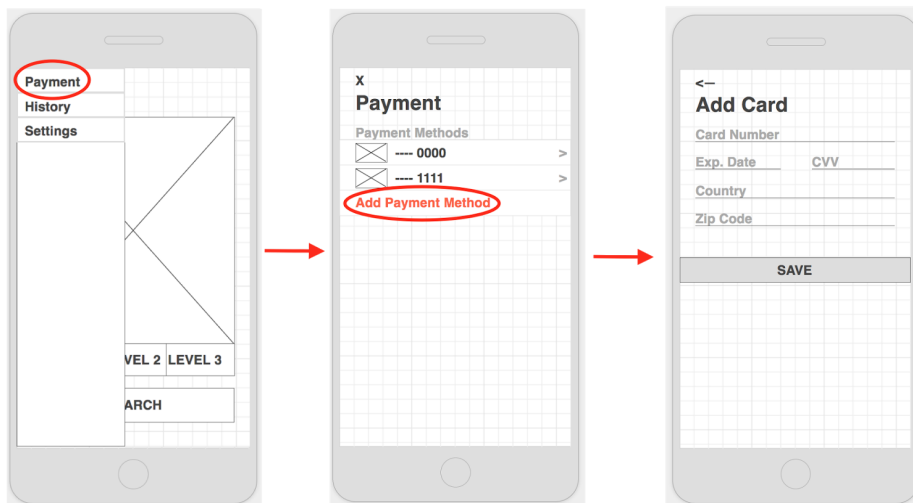
Pressing the Search Button brings up a search window. Here the user can search for a specific parking lot and select it from a list. Then they will be returned to the home page, where the selected parking lot's map will be displayed.



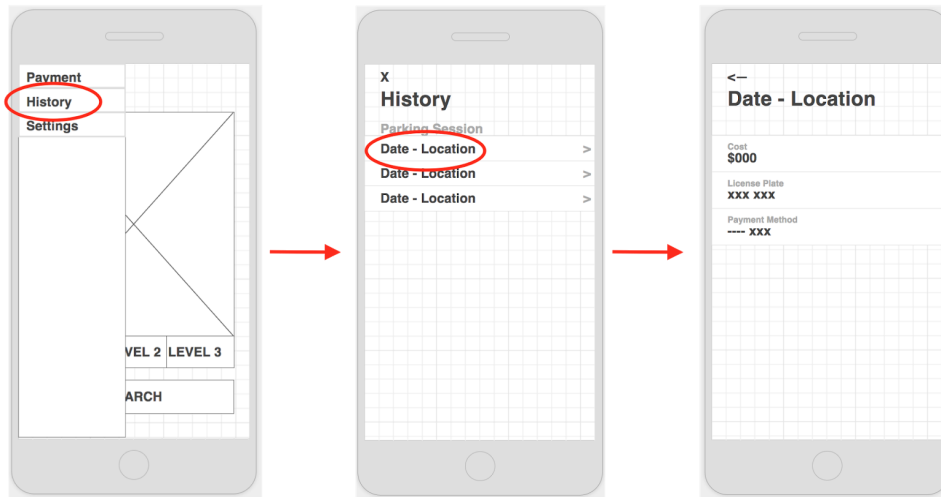
Pressing the menu button will bring up Payment, History, and Settings buttons



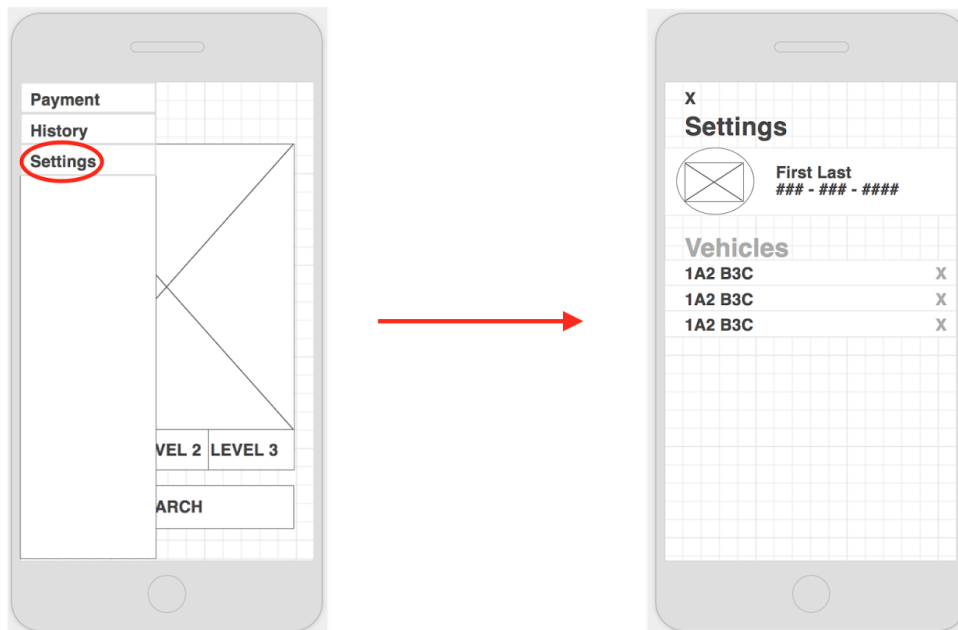
Pressing the payment button brings up the payment screen. Here the user has the option to review and remove payment methods. However, they must always have at least one valid payment method on their account.



To add a payment method, the user simply presses the Add Payment Method button and fills in the necessary credentials.



From the menu, the user can also access their history. The History page displays all of their previous parking sessions. Pressing on a specific session will bring up additional information, such as the cost, the licence plate they drove in with, and the payment method that was charged.



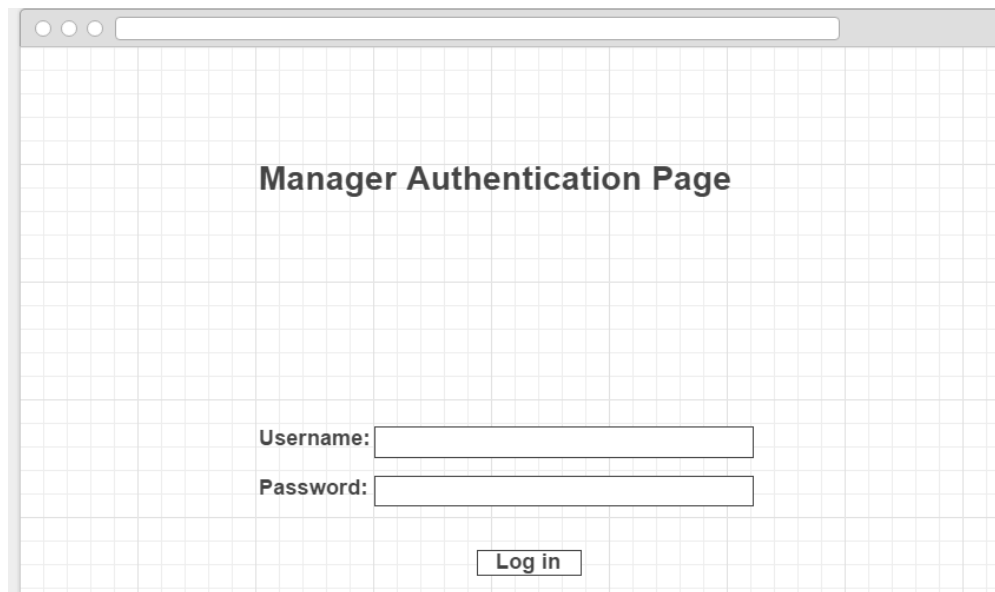
The settings page contains information regarding the user. This includes their name, phone number, and the license plates registered to their account. To remove a license plate from an account, the user presses the X next to the license plate.

If the user no longer wishes to pay for a vehicle's parking, they must remove it from their account. A vehicle cannot be removed if it is currently checked into a parking lot.

A license plate can only be registered to a single account, this is to avoid any confusion regarding who should be charged. To avoid any malicious abuse of this restriction, the user cannot manually add vehicles to their account.

The system will handle registration of a vehicle upon arrival. When a customer arrives with an unregistered license plate, they will be asked to provide a credit or debit card. If this payment method is registered to an account, the license plate will be added to that account for future visits.

Manager Dashboard



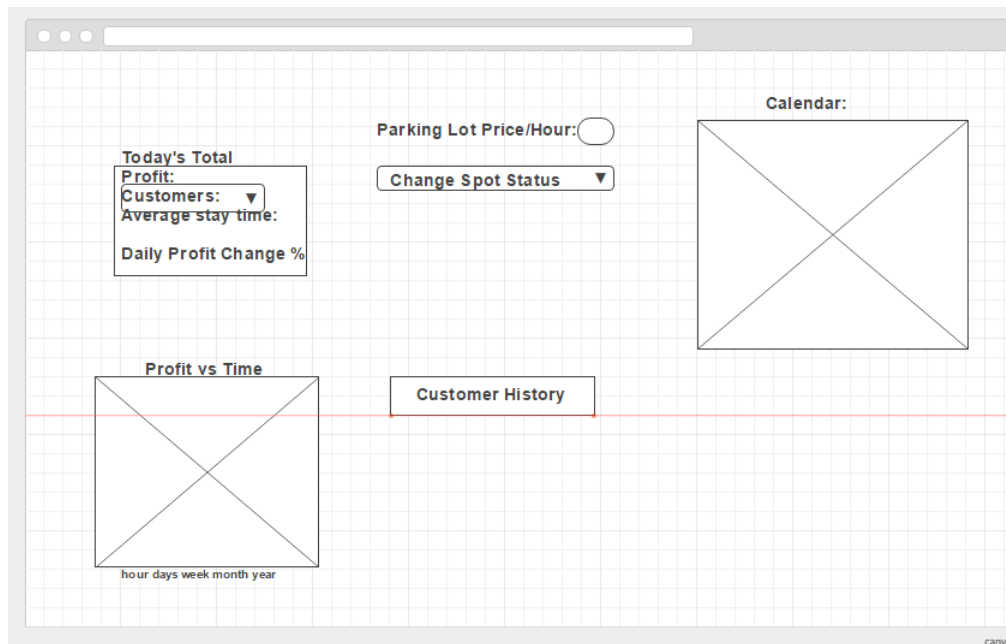
The image shows a web browser window displaying a login page. The page has a light gray grid background. At the top, there is a title bar with three window control buttons (minimize, maximize, close) and a search bar. The main content area is titled "Manager Authentication Page" in a bold, black font. Below the title, there are two input fields: "Username:" and "Password:". Each field has a corresponding text label to its left. Below the password field, there is a "Log in" button. The browser window has a small "c" icon in the bottom right corner.

Manager Authentication Page

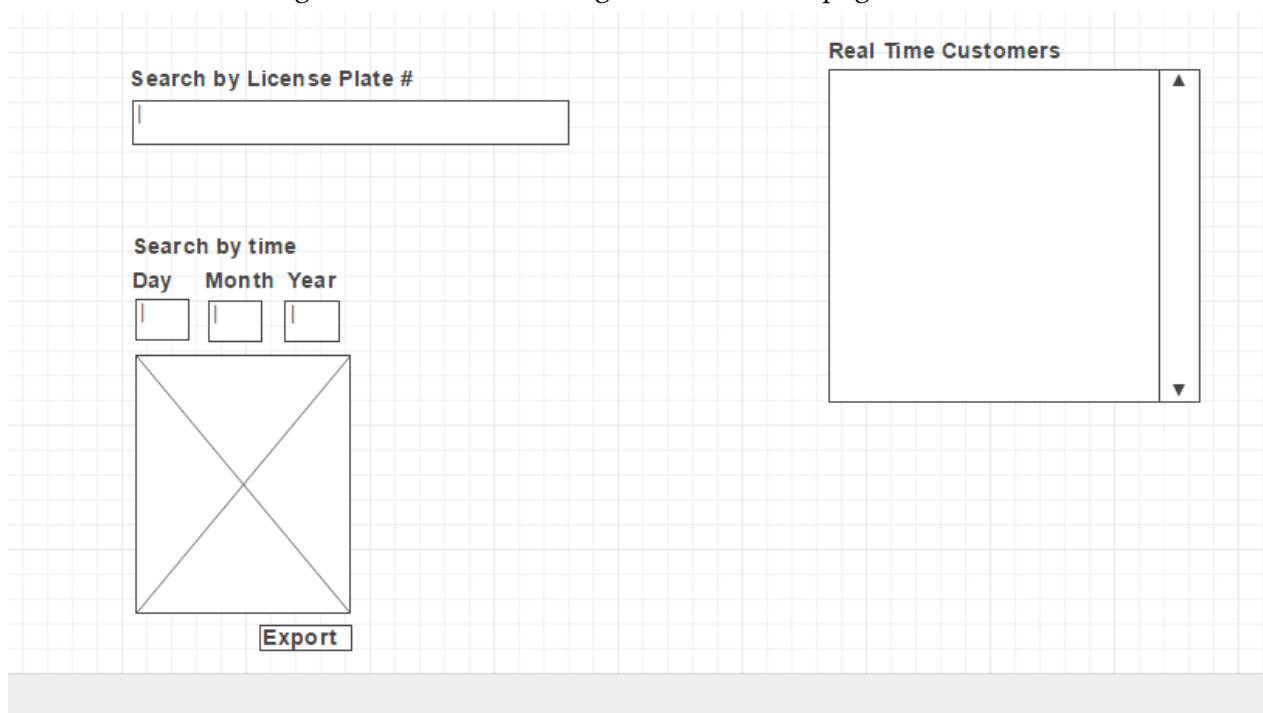
Username:

Password:

The first page prompts the manager to input the confidential and verify the identity. After successful login, the second page is shown.



In the second page, the manager is able to view the statistics of revenue earned and summary of customer behaviors (e.g. average stay time) for the current day on the top left box. The manager can set price on the top middle of page, where followed by a option bar to change spot status. On the right shows the calendar. Graphs of data will be displayed on the left bottom. There is also a button for manager to click, which navigates to the third page.



On the third page, the manager can input license plate # (or user account #) to view payment history of that specific user. An alternate way of searching is to type date and time on the left bottom to export all payment records at that time. On the right displays the real-time information of spots and customers.

User Effort Estimation

Accessing Management Analytics: 6 Clicks

Navigation: 1 Click

1. Click on the web page icon to open to login page.

Data Entry: 4 Clicks

1. User enters their username.
2. User presses "tab" key or click password data field.
3. User enters their password.
4. User presses "enter" key or click "Log in".

Analytics Navigation: 1 Click

1. User can click to view more information on License Plate information.

Accessing and Editing customer information: 8 Clicks

Log in: 4 Clicks

1. User enters their username.
2. User presses "tab" key or click password data field.
3. User enters their password.
4. User presses "enter" key or click "Log in".

Navigating to desired information: 2 Clicks

1. User clicks the top left menu icon.
2. User clicks "Payment" or "Settings" based on what they want to change.

Editing desired information: 2 Clicks

1. User selects the field they would like to edit.
2. User saves the information.

Accessing Parking Lot Information: 8 Clicks

Giving a recommended Parking Space: 1 Click

1. Clicking on the app will display the recommended parking spot.

Exiting the parking Lot: 0 Clicks

1. Nothing is needed to be done by the user

View History of Transactions: 4 Clicks

1. Click on the phone app/web app to open the login page.
2. Enter username and password.
3. Press enter/click login.
4. Navigate to history.

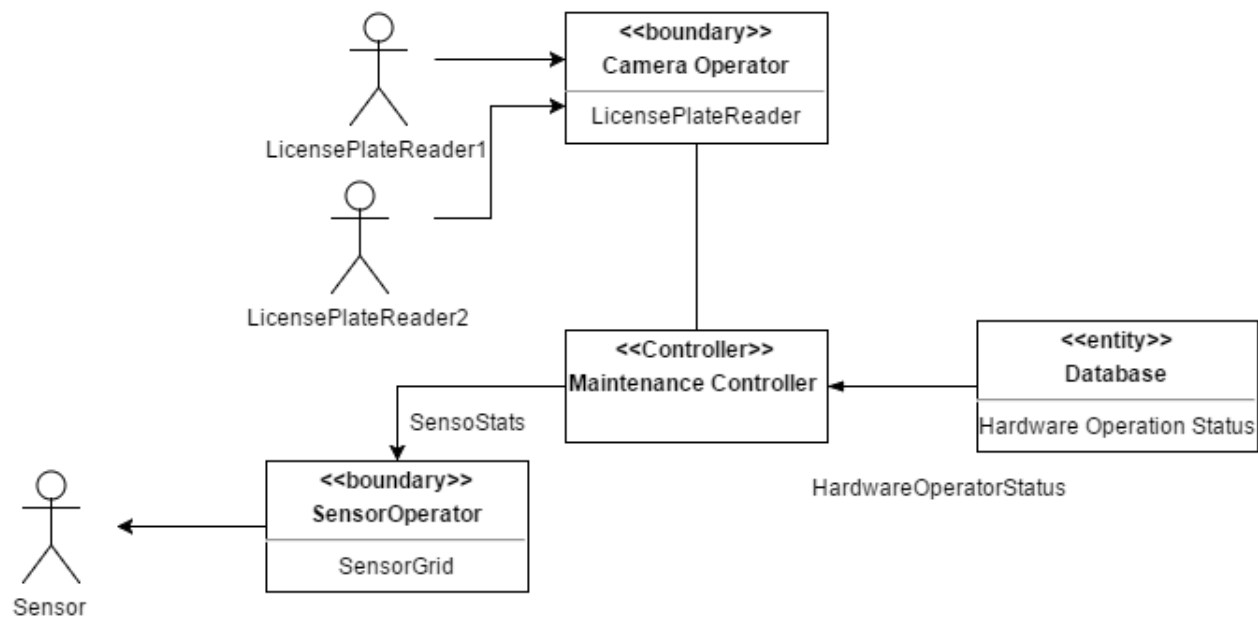
Check Occupancy of Parking Lot: 3 Clicks

1. Click on the phone app/web app to open the login page.
2. Enter username and password.
3. Press enter/click login.
4. Occupancy will automatically display at the right.

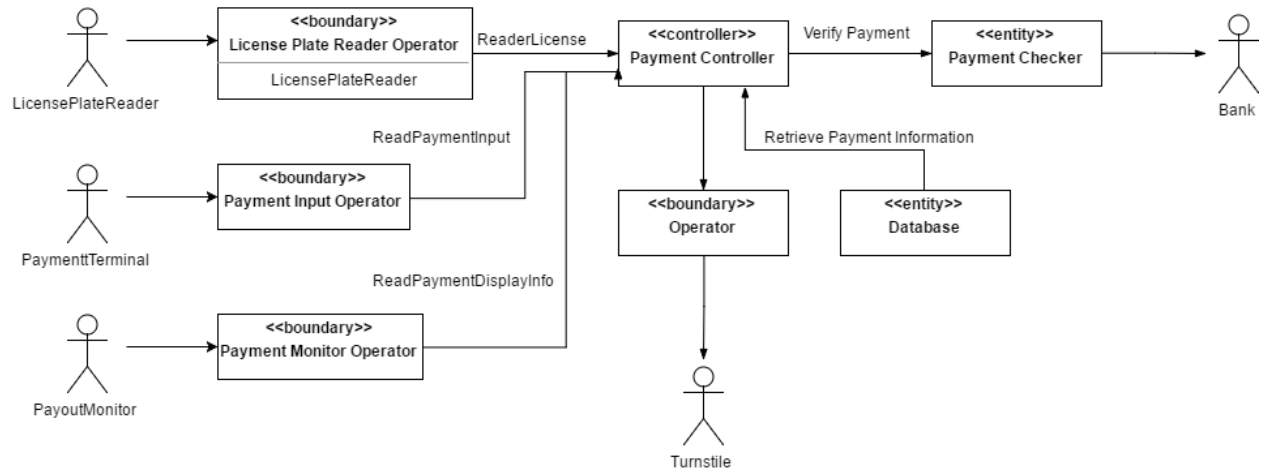
Domain Analysis

Domain Analysis UML Diagrams

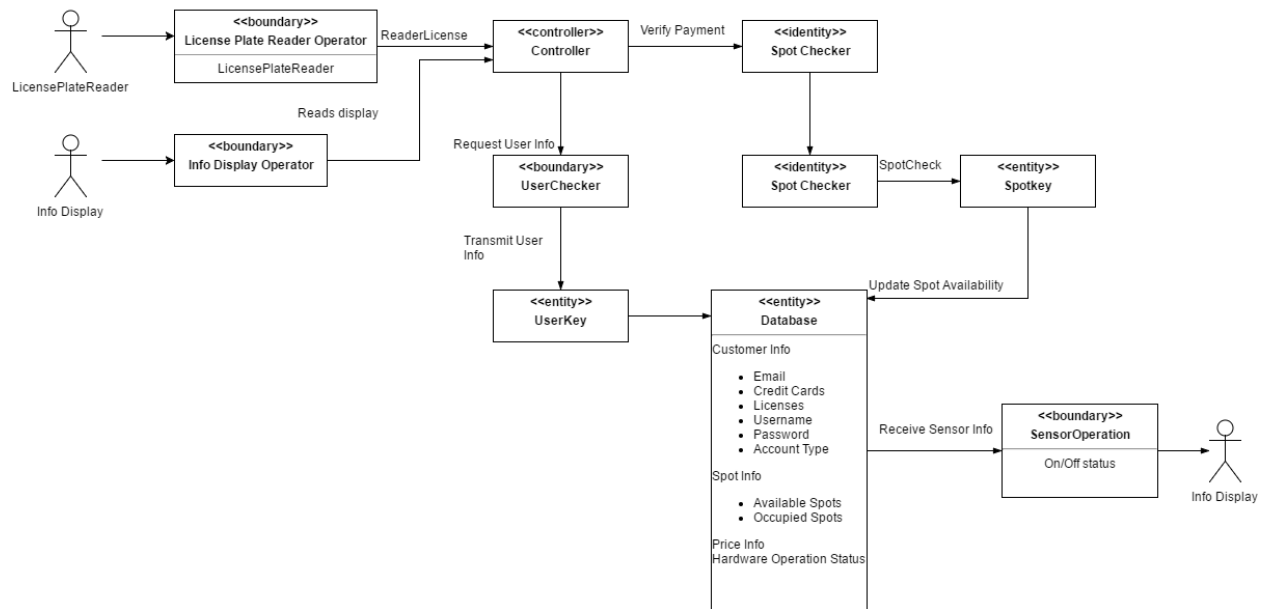
Maintenance Domain View



Exit Domain View



Park Domain View



I. Concept definitions

Responsibility Descriptions	Type	Concept Name
Reads inputs from the sensors to determine occupancy of each spot and informs the SensorChecker. If contradicting inputs are read in from particular spots dual sensors, a signal will be sent to the MaintenanceLogger to record the problem.	K	SensorOperator
Reads input from SensorOperator to update the FreeSpotKey.	D	SensorChecker
Reads input from LicensePlateReaderOperator and UserChecker. It determines whether or not to admit entry to the User. It uses the input from UserChecker and works with RecommendController to give a registered user a recommended parking spot.	D	TurnstileController
Outputs commands to the Turnstile	D	TurnstileOperator
Reads input from the License Plate Readers and communicates with the TurnstileController for arrivals and the PaymentController for exits.	D	LicensePlateReaderOperator
Serves to interact between the PaymentController to confirm session info has been logged and payment has been appropriated so the user can exit.	D	PaymentChecker
Receives request from the TurnstileController to obtain the type of customer that has arrived. It then accesses UserKey to determine the type of customer from the profiles where it will start a log for the session. It will also create a guest profile in the UserKey if the customer is not registered for a spot.	D	UserChecker
Receives notifications from hardware in the event of an abnormality, sends notifications to local service center and logs information in the Database	D	MaintenanceLogger
Receives input from LicensePlateReader of user approaching to exit. This plate number is then sent to the PaymentChecker to determine amount owed. Once payment criteria is confirmed through the PaymentChecker, the PaymentController will tell the GateOperator to open the gate.	D	PaymentController
Serves to interact between the PaymentController and the UserKey to confirm session info has been logged and payment has been appropriated so the user can exit.	D	PaymentChecker

Reads the payment received from the Payment terminal in the event a customer must pay before being allowed to exit and sends it to the Payment Controller.	D	PaymentOperator
Intermediary between Interface Page and Pagemaker	D	WebPageController
Gathers information to be displayed on the interface	D	PageMaker
Acts as the GUI between the system and the User	K	InterfacePage
Stores all pertinent variable information of the system	K	Database
Used to temporarily store all user info that it retrieved from the database for the UserChecker until the User is processed.	K/D	UserKey
Receives input from FreeSpotKey and determines a vacant parking spot to recommend to the user who has passed through the turnstile. It can also change the spot if FreeSpotkey gives a notification that the spot has been taken by someone.	D	RecommendController
Container for current number of parking spots available on each floor and where their subsequent location is, collaborates with the Database.	K	FreeSpotKey
It will output necessary payment information through the Payment Monitor	D	PaymentMonitorOperator
Reads input from Sensor Operator to determine the state of all the parking spots. Reports to FreeSpotKey all the spots that are open.	D	SpotChecker

ii. Association definitions

Concept Pair	Association Description	AssociationName
SensorChecker - SensorOperator	SensorChecker obtains sensor readings from SensorOperator in order to update spot availability (FreeSpotKey)	ReadSensor
TurnstileController - SpotChecker	TurnstileController obtains data from SpotChecker to check for spot availability (from FreeSpotKey)	RequestSpotInfo
TurnstileController - TurnstileOperator	TurnstileController sends command to TurnstileOperator to open turnstile and waits for execution confirmation	DecideAvailabilty
LicensePlateOperator - TurnstileController	TurnstileController reads information from LicensePlateOperator as a User pulls up to park	ReadsPlateInfo
PaymentController - LicensePlateOperator	PaymentController reads license number from LicensePlateOperator as Users pull up to attempt to exit.	ReadExitLicense
PaymentController - PaymentOperator	If it is a guest customer arrives and swipes their credit card at the turnstile, the PaymentOperator will send the payment info that it receives to the PaymentController.	ReadsPaymentInput
PaymentController - PaymentMonitorOperator	PaymentController will send the amount due to the PaymentMonitorOperator	ReadPaymentDisplay
UserChecker - UserKey	UserChecker asks the UserKey to retrieve any User info that is saved based on its information.	TransmitUserInfoRequest
UserKey - Database	UserKey loads any info pertaining to the request that is found in the Database	RetrieveUserInfo
PaymentController - PaymentChecker	PaymentController sends the license info of who is attempting to exit to the PaymentChecker. The amount owed that is	VerifyPayment

	returned is then sent to be displayed on the PaymentMonitor. Any payment that is received from the Payment Operator is verified by the PaymentChecker before the User can exit.	
MaintenanceLogger - SensorOperator	MaintenanceController receives notifications from SensorOperator in the event of any malfunction	SensorStatus
MaintenanceLogger - LicensePlateOperator	MaintenanceController receives notifications from LicensePlateOperator in the event of any malfunction	LicensePlateReaderStatus
MaintenanceLogger - ElevatorOperator	MaintenanceController receives notifications from ElevatorOperator in the event of any malfunction	ElevatorStatus
MaintenanceLogger - Database	Updates hardware statuses in the database	HardwareOperationStatus
PaymentChecker - PaymentKey	PaymentChecker will receive the amount due from PaymentKey and will update it if payment is received	RequestPaymentInfo
PaymentKey - Database	PaymentKey accesses info based on the info it receives from PaymentChecker from the Database	GetAmountDue
SpotChecker - FreeSpotKey	SpotChecker consults the FreeSpotKey to check for available spots	OpenSpotCheck
SensorChecker - FreeSpotKey	FreeSpotKey receives the input from the sensors through the SensorChecker	UpdateSpotKey
FreeSpotKey - Database	FreeSpotKey and Database keep each other synchronized	UpdateSpotAvailability
PaymentController TurnstileOperator	PaymentController will send the signal to the TurnstileOperator to lift the gate once payment is confirmed to have been appropriated.	ExitSuccess

WebpageController InterfacePage	WebpageController receives info from the InterfacePage and sends request to the PageMaker to build a response	WebPosts
InterfacePage PageMaker	InterfacePage reads the information collected by the PageMaker	PreparesWebPage
PageMaker - Database	PageMaker queries database for pertinent information	providesWebData
PageMaker WebPageController	PageMaker receives the request from the WebPageController	ConveysWebRequests
FreeSpotKey - RecommendController	Gives User a recommended from FreeSpotKey	GivesRecommendedSpot

iii. Attribute definitions

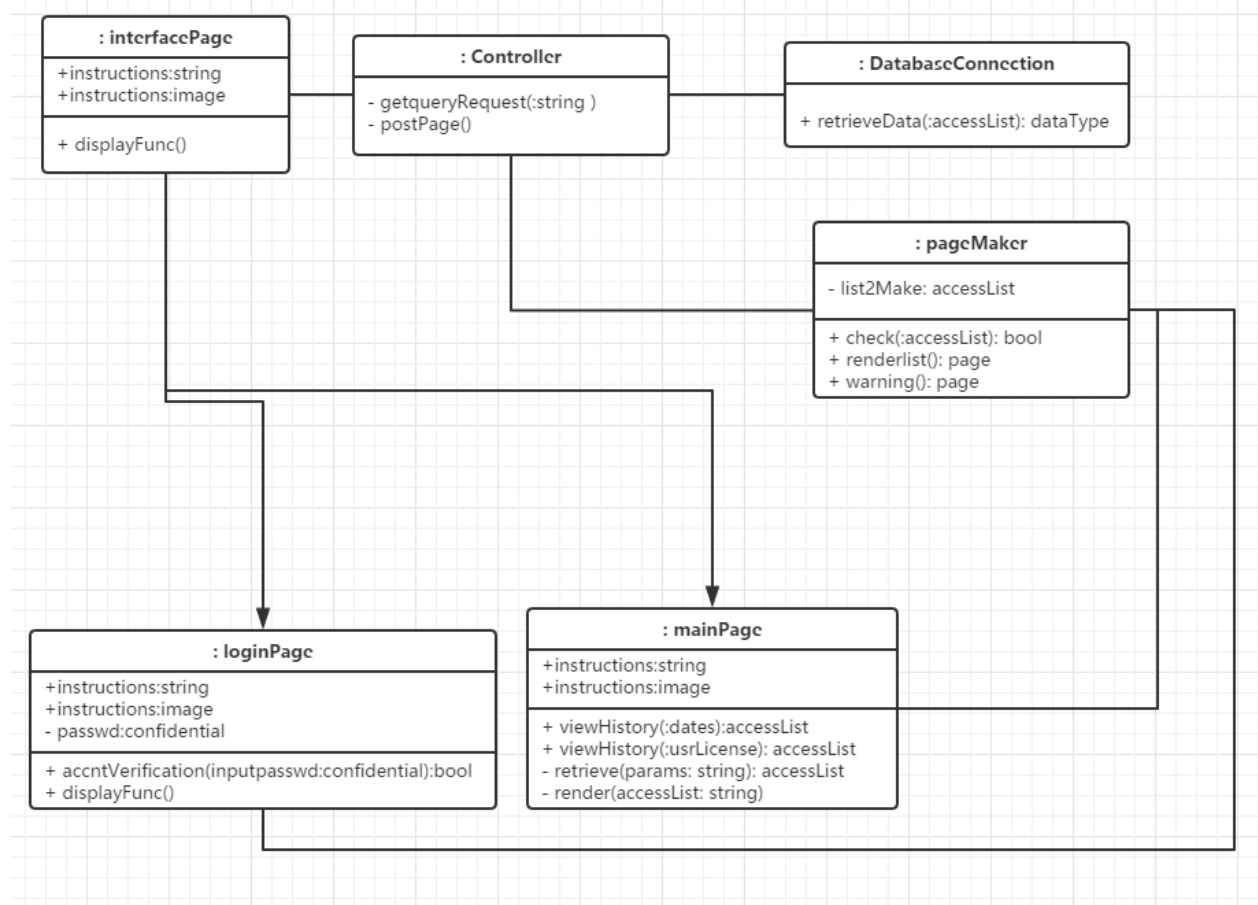
deviceName	Name of the hardware with the error
LicensePlateNumber	String of characters that represent the license plate information taken from the License Plate Readers
errorMessage	String pertaining to the type of error with the hardware
password	...
userID	...
SessionOwed	Amount owed that is taken every parking session unless they are a guaranteed customer during the extent of their contract time
TotalOwed	Total of all SessionOwed-SessionPaid
SessionPaid	Total amount paid per session
EntranceTurnstileStatus	present status of entrance turnstile
ExitTurnstileStatus	present status of exit turnstile
MaintenanceIssue	String to log into MaintenanceKey
FreeSpots	Bitmap of spot availability
CreditData	Record of credit card information for guest users who swiped at the entrance and will later get charged at the exit
OpenSession	Time entered and pending duration of current stay

iv. Traceability Matrix

[illegible]

Manager's Domain

I. UML Class Diagram



II. Concept Definition

Responsibility Description	Type	Concept Name
Coordinate actions of all concepts associated with Manager's analytics.	D	ManagerController
Contain login screen and verification of identity.	P	Login page
Contain responsive buttons by which the manager views the data analytic options.	K	Main page
Coordinate and display graphs and analytical information on the web page.	P	Pagemaker

Coordinate the connection to server database for fetching requested data.	P	Database Connector
---	---	--------------------

III. Association Definition

Concept pair	Association description	Association name
Controller - Pagemaker	Controller passes requests to pagemaker and receives back pages containing processed pages for display	Conveys requests
Pagemaker - Database Connection	Database Connection passes retrieved data to pagemaker for display	Provides data
Pagemaker - Main page	Pagemaker prepares the main page.	Prepares
Controller - Database Connection	Controller passes search requests to Database Connection.	Conveys requests

IV. Attribute Definition

Concept Pair	Attribute	Attribute Description
Controller	Price	Used to change the price per hour of the parking lot
	Search Customer Parameters	Used to search through history of customers. Includes date and time range.
	Date Parameters	The range of time that the analytics displayed should stretch. This can be range from a few hours up to years.
Log in Page	Username	Which account is attempting to log into the manager page.

	Password	Authentication that the account attempting to log in is the correct person.
PageMaker	Analytics	Used to convey the statistics on the parking lot business for managers to use.
MainPage	Graph Parameters	Changes the graph's date display range. Options include hours, days, weeks, months, and years.
Database Connector	Data Requested	The data that is requested from the Controller.

V. Traceability Matrix

Use Case	1	2	3	4	5	6	7	8	9	10	11	12	13
Controller								x	x	x	x		
Log in Page													
Page Maker									x	x	x		
Main Page													
Database Connector										x			x
PW								5	13	20	14		5

VI. System Operation Contracts

Operation	Perform Analytics
Precondition	The required information has been attained by the parking lot system and stored in the database.
Postcondition	The pertinent information will be displayed in an easy to understand

	format.
--	---------

Operation	Authenticate Manager
Precondition	There is a database storing the username and passwords of managers so that the entered authentication parameters can be checked.
Postcondition	The manager attempting to log will know if the credentials he/she entered was correct.

VII. Mathematical Models

Regression may be used to study the customers' behaviors and predict the number of customers in a particular time period, based on real data collected in actual payments.

Plan of Work

Progress to Date Summary

Since the proposal, the team has taken great strides in designing the overall system. We first started out with fragmented ideas about who the user was and what types of requirements were needed for each unique type of user. However, as seen from our meeting notes, we started to slowly digest the magnitude of the problem, as we started to break the larger problems into smaller subsets. During meetings three and four, we started to develop User Stories and the UI Specification for the system. Once we started using the whiteboard to draw all different use cases and how the user would interact with the system, everyone could see a clear vision of what steps needed to take place for the system to come to fruition. In addition, we realized that we did not gain anything from meetings unless we had clear objectives for the meeting and definitive actionables for each person. This is why we started converting our meeting note takeaways into TODOs that we posted on the GitHub page for everyone to see.

Week	Description of Planned Events (on 1.29.2017)	Actual Occurrence of Events
1	The team will finalize the goal and scope of the overall project, as well as the sub-projects. Sub-groups will work to create a schedule to accommodate personal schedule, while meeting discussed deadlines. The “shared infrastructure” of the project, i.e user and data models, must be agreed upon, and a schedule must be set for its imminent completion.	Team spent the first week thinking of different solutions for the proposed problem. We set times for weekly meeting schedule and sub-groups set their own schedules for meeting online. We determined the technologies we wanted to use and started learning Django framework.
2	All groups will communicate and coordinate to finish the shared API. Individual sub-groups will work on sub-group specific design details.	We did not finish making the shared API because we did not have a clear vision for the solution design. We coordinated heavily to complete the report.

3	All groups will continue working on their set responsibility while maintaining communication between the subgroups. This includes describing their work to the other groups and coordinating with each other.	We finally had a clear vision of what we were building and what it would look like. Sub-groups started finishing their designs, use cases, domain models, and etc. Successfully implemented the license plate reading and started building the Django foundation for the implementation stage. Further discussion about class diagrams commenced among sub groups.
---	---	--

Roadmap of Work

<https://app.smartsheet.com/b/publish?EQBCT=6e550bb75170435685b17b56898a4db5>

Due to the extensive nature of the Gantt Chart, please view it on your browser.

Product Ownership

Teams

1. Arthur Rafal and Guy Rubinstein
2. Kendric Postrero, Vatsal Pandya, and Gao Pan
3. Peter Luo, Yunqi Shen

Responsibility Breakdown:

Team 1: User registration, turnstile implementation, user data verification, payment verification

Team 2: Parking lot system implementation, parking space recommendation, parking lot testability, data collection for the parking lot

Team 3: Manager access of setting price and enabling/disabling spots, Web portal dashboard for managers to view data, implement useful data analytics algorithms

References

1. Marsic, Ivan. Software Engineering. 2012.
 - a. Referenced the tic-tac-toe example as a basis for how to go about problem breakdown
 - b. In addition, we utilized the online resources (website) associated with the textbook
2. Bruegge, Bernd, and Allen H. Dutoit. Object-oriented Software Engineering: Using UML, Patterns, and Java. Boston: Prentice Hall, 2010
 - a. Consulted to get a better understanding of UML and view examples
3. Free flowchart maker and diagrams online." RSS. N.p., n.d. Web.
 - a. <<https://www.draw.io/>>
 - b. We wanted to view more examples of sequence diagrams before we started building ours
 - c. The tools on the website were used to construct our sequence diagrams
4. "Minimal Wireframe Tool" Web.
 - a. <<https://wireframe.cc/>>
 - b. Used examples and the provided tool to create wireframes for the UI specification section
5. User Story Points
 - a. <https://www.atlassian.com/agile/estimation>
 - b. The website was consulted to determine how story points were awarded and the number awarded to each problem size
6. License Plate OCR
 - a. <https://github.com/openalpr/openalpr>
 - b. We were unsure about the current industry procedure for recognizing license plates. Furthermore, we were looking for an open source tool, from which we could better understand the recognition procedure—through viewing their code. We were able gain a strong understanding through their documentation, and we successfully integrated that into our code base.
 - c. This allowed us to recognize car licence plate, make, model, color, and etc.
 - d. We were also able to confirm that an image can be taken from various angles, not just straight.
7. Non-Functional Requirements

- a. <<http://www.ece.rutgers.edu/~marsic/books/SE/projects/ParkingLot/2012-g3-ProjectFiles.zip>>
 - b. After much discourse, we were struggling to determine how to structure Non-Functional Requirements because most examples listed them as REQ list but we were using user stories. We consulted 2012 Group 3's documentation.
 - c. Despite some years, non-functional requirements are similar to 2012 iteration of garage the automation track
8. Gantt Charts
- a. <http://en.wikipedia.org/wiki/Gantt_charts>
 - b. We were familiar with the term, but we needed to understand why product managers used this tool frequently.
 - c. Gained a step-by-step understanding of how to construct a Gantt chart
9. UML Domain Analysis Model
- a. <<http://www.ece.rutgers.edu/~marsic/books/SE/projects/ParkingLot/2012-g4-report3.pdf>>
 - b. <<http://eceweb1.rutgers.edu/~marsic/books/SE/projects/ParkingLot/2013-g5-report3.pdf>>
 - c. Used 2012 Group 4 and 2013 Group 5's domain analysis models and edited it to take into account our new features.