Software Engineering Spring 2016

Group #15

Parking Garage Automation

<http://www.galuwa.com>

Report 3

5/4/16

|  |  |
| --- | --- |
| Brandon Dunlap | brandon.dunlap@rutgers.edu |
| Vikram Krishna | vk235@rutgers.edu |
| Yufeng Liu | yufeng.liu@rutgers.edu |
| Luke Miller | luke.miller@rutgers.edu |
| Harshil Patel | harshil1029@gmail.com |
| Thomas Walters | twalters1012@gmail.com |
| Xiang Xing | xx52@scarletmail.rutgers.edu |

**Table of Contents:**

1. Summary of Changes…………………………………………………….3
2. Customer Statement of Requirements………………………………....3
3. Glossary of Terms…………………………………………………….......6
4. System Requirements…………………………………………………....7
   1. Functional Requirements……….…………………………..……8
   2. Non-Functional Requirements……….……………………….....11
   3. On-Screen Appearance Requirements……………………...….12
5. Functional Requirements Specification……………………………..….19
   1. Stakeholders…………………………………………………..…..19
   2. Actors and Goals……………………………………………….....19
   3. Use Cases……………………………………………………..…..21
   4. Use Case Diagram………………………………………………..23
   5. Traceability Matrix……………………………………………..….24
   6. Fully Dressed Description…………………………………….….26
   7. System Sequence Diagrams………………………………..…..34
6. Effort Estimation……………………………………………………….....37
7. Domain Analysis……………………………………………………….....38
   1. Domain Model………………………………………………..…...38
   2. System Operational Contracts………………………………….38
   3. Mathematical Model………………………………………….….41
8. Interaction Diagrams………………………………………………….…44
9. Components Class Diagrams & Interface Specification………..…...53
10. System Architecture and System Design………………………….….58
11. Algorithms and Data Structures…………………………………….….58
12. Design of Test…………………………………………………………....67
13. History of Work…………………………………………………………..74
14. Current Status & Future Work……………………………………...….75
15. References………………………………………………………….……75
16. Project Management…………………………………………………....76

**Summary of Changes:**

For the Interaction Diagrams, the diagrams were changed so that the names of all the objects and variables match as used in each diagram. This made sure everything was synchronized properly during the coding part and so we had to make these parts match. Also, the interaction diagrams were given design pattern explanations. The design patterns used were publisher-subscriber, state, command, decorator, and proxy. Lastly, some items were removed from the diagrams as we did not get to the implementation of them and will be noted down in the future work section.

For the system requirements, the pictures were updated with actual screenshots from our website. In addition, anything that was not completed was marked as future work. This includes anything that can be added as additional functions of the website in the future.

For functional requirements, several of the requirements were not implemented before the final demonstration. These requirements have been highlighted and listed under future work.

**Customer Statement of Requirements:**

Existing parking garage systems exhibit a plethora of flaws, including our current method of garage workers manually counting available parking spaces. Our current method makes optimization of garage usage incredibly difficult, while incurring unnecessary costs to the garage operator. Our manual system is also limited in it’s accuracy and resolution. The workers record the available spaces onto a spreadsheet, which is then manually entered into our database. A process which is not only slow, but very prone to human error. Spot identifying workers can only examine the garage so often, and can’t update the system immediately as cars come and go. The information we do have is typically not even available for customer use.

Parking should be a stress free, conveniently provided service, but do to the lack of information, drivers that arrive at the garage are subject to confusion and frustration. Unsure of available parking spaces, customers spend excess time searching for a park. The inconvenience can easily turn into frustration, which can lead to loss of customers. The absence of assigned spaces not only facilitates poor usage optimization, but an aggressive environment within the parking garage as customers are forced to compete for spaces. This issue is common among parking garages, and it forces customers to arrive and claim spaces hours before it should be necessary, for fear they will not find a spot if they do not take initiative. And the customers that don’t come early, and are not lucky, are forced to drive around and congest the garage in search of vacancies.

Current parking garage implementations do not satisfy the needs of those who need regularly scheduled parking spaces. Many garages will sell monthly passes, but without a computerized scheduler they risk overbooking and incorrectly estimating the number of cars present. Most monthly passes do not guarantee the availability of a spot, but instead allow the user to park whenever they’d like since they have already paid. The majority of parking garages also utilize a ticketing system that stamps arrival time and requests appropriate payment on garage exit. This does provide flexibility for the user to stay as long as they need, but can cause long payment lines and makes it impossible for management to monitor how long cars will stay. In the case of a lost ticket, drivers are typically charged for a full 24 hour period. While this benefits the management, customers are unhappy about being responsible for a small slip of paper. Many garages require payment prior to exit which forces the customer to seek out a ticket machine, which is often at a different entrance and has a large line. These machines do not accept PayPal, Android Pay, Apple Pay or any online payment processors.

These issues can be resolved by a centralized computer system that autonomously manages most aspects of user and administrative tasks. Manual spot availability checks can be readily replaced by a series of sensors that identify the presence of a car in every single parking spot. When a customer leaves or arrives, the status of this parking spot will immediately change in the system and be conveyed the customer. This availability can be displayed on a sign outside the garage in addition to the company website. This eliminates the issue of low checking resolution in addition to poor statistics monitoring for the administration. These spot indicators will eliminate the costs incurred by hiring additional spot checking employees and will be a more accurate representation of parking availability within the garage. In addition to actively monitoring occupied location, a centralized computer system would allow the garage to track when vehicles enter and exit, reinforcing availability checks.

The garage needs to be split into two sections. The upper deck will be designated parking for reservation and account holders only. The bottom deck will will behave as a traditional parking garage to facilitate walk-ins who do not wish to create an account with the garage. Availability for this deck is first come, first serve and there will be no guarantee for spot vacancy. The upper deck will be accessible only by a single car elevator and will be departed by a one way exit ramp. The upper deck can be accessed on foot by a stairwell or customer elevator that will require reservation confirmation to enter. The single car elevator will remove congestion inside the garage and restrain any waiting times to outside the parking garage. Wait times will be significantly diminished as a result of the reservation scheduling. There will not be more cars waiting in line than spots available in the garage.

Reservations can eliminate driver frustration within the garage by removing the need to search for a parking spot. Reserving a parking location during a designated period prior to arrival will guarantee the availability of the spot. Parking garage users will be able to create an account online containing payment method, demographic location and license plate information if available. Each account will have the ability to add multiple vehicles, but only business accounts will allow multiple drivers and multiple vehicles. Users of both personal and business accounts will have the ability to log into the company website on their laptop or mobile devices to view/edit their account information and make/manage reservations. Reservations do not require the entry of vehicle license plate information to accommodate those renting or borrowing vehicles. Reservations can be made at any point in advance and require the user to specify the arrival and departure time in increments defined by a specific garage. The user has the opportunity to extend their reservation up until the end of their existing reservation but the extension will be limited to a period defined by the garage. All extensions will be initiated through the user’s account on the company website. If the user exceeds their reservation, they will be charged a premium rate as defined by the garage. When the user exits the garage, their account will be billed. The user will have the ability to cancel a reservation for a garage specified period before their start time. After that period ends, their payment method will be charged whether or not they show up for their reservation. Though the user is encouraged to arrive at their designated time, there will be an arrival grace period as specified by garage management.

This automatic payment system eliminates the need for a driver to carry around any additional information. With the ability to check their active reservation and remaining time in their account, the stress of parking is no longer an issue. Customers who require regular parking will have the ability to purchase contracted reservations. Customers will have the ability to specify a weekly schedule and the parking times needed. This can be modified to occur at biweekly intervals or on set days every month. By default these reservations will be billed in total at the beginning of every month, but the garage management can modify this period in their account settings. Any overages for a contracted reservation will also be billed at the time of departure. At the time of reservation, a user will be notified of any premium or discounted fees during their stay. The user will also be aware of any additional fees through a cost estimator when their reservation is scheduled.

Automation will play a crucial role in enhancing user experience in, and outside the parking garage. Upon arrival, the driver’s licence plate will be scanned in the elevator and the reservation will be confirmed. If the driver was unable to register the vehicle’s license at the time of registration, a confirmation number can be manually entered or a QR code can be scanned to gain access. Users without a reservation will be directed to leave and valid customers will be transported to the correct floor and directed to a particular parking spot. Customers will exit the garage through a separate elevator or stairway and upon their return, will enter their confirmation number or scan their QR code to gain access to the garage’s upper deck. As they exit, a user’s license plate will be scanned and close their active reservation.

An important focus will be the security of the parking garage, the parking garage manager will be able to quickly access any cars within the garage for medical, law enforcement. A problem with current parking garages, is there are no security measures and quick access. Parking spots will be associated with license plates, and model of the car which will allow for quick lookup for any cars suspected for crimes for example. Not to mention, medical responders will be able to quickly find those in need of attention regardless of location in the parking garage, as internal cameras will be linked with alerts to emergency responders.These emergency responders may also alert government officials in accordance with a contractual agreement with our company which will provide benefits to both us and the government.

**Glossary of Terms:**

* **Elevator Camera**: Scans license plate to confirm reservation
* **Garage Elevator**: Used to transport the customer’s vehicle to the designated floor
* **Exit Camera**: Scans license plates as cars leave
* **Elevator Console**: Allows user to access reservation by confirmation number
* **Elevator Display**: Displays registration information, the spot number which assigned to them and issues
* **QRCode Scanner**: Allows user to access reservation with confirmation QRCode
* **Occupancy Photosensor**: Photosensor on each parking spot for detecting car presence
* **Digital Display**: Mounted out front to represent vacancy
* **Closed Circuit Television Cameras**: Monitor crime and activity in garage
* **Confirmed Reservations**: Reservation created online by the customer
* **Guaranteed Reservations**: Subscribed reservations occurring regularly
* **Understays**: Customers who leave before they time has expired
* **Overstays**: Customers who exceed their reserved time
* **Overstay Surcharge**: An increased rate charged to customers for exceeded time
* **Walk-Ins**: Customers who do not reserve a spot prior to entering the parking garage
* **Cancellation**: A request sent in by the customer at least 24 hours prior to the time slot to cancel their reservation
* **Cancellation Fee**: A fee charged to a customer who requests a cancellation within less than 24 hours of their start time
* **Check-In**: When a customer uses their QRCode/confirmation number at the console
* **Web User Account**: An account for the customer used for general account management purposes and to modify payment/vehicle information and reservations/cancellations
* **Occupancy Checks**: Current implementation to manually check for vacant/occupied spots vs the new idea of automated checks by the system
* **Specials**: Price cuts, or discounts, applied during promotion periods
* **Holiday Hours**: Specific hours of operation during holidays
* **Grace Period**: A certain amount of time given to the customer to enter/leave his spot
* **Link Password**: Personal account will use this password to connect to a company account.
* **Business Account**: An account could hold vehicles which are allowed the users connect to this account to use.
* **Recurring Appointment**: An appointment which will be made in the same time for a certain time period.

**System Requirements:**

Actors Available

* Parking Garage Manager; ST-M#
* Vehicle Operator (Customer); ST-C#
* Emergency Response Team; ST-ER# [Future Work]
* Parking Garage Owner; ST-GO# [Future Work]
* Payment Processing Company; ST-P#
* Business Account Manager (Business Customer); ST-BC#

Note: Business customers are a child of customers. All customer requirements apply to business customers but not the other way around.

**Functional Requirements**

|  |  |  |
| --- | --- | --- |
| Identifier | Priority Weight | User Story |
| ST-C1 | 5 | As a customer I want to log into my account and make a single use reservation. |
| ST-C2 | 5 | As a customer I want to cancel reservations during a garage specified period before the reservation start time. |
| ST-C3 | 5 | As a customer I want to add, edit, or delete payment information. |
| ST-C4 | 5 | As a customer I want to enter my confirmation number or scan my QRCode if my license plate is not registered. |
| ST-C5 | 4 | As a customer I want a grace period on reservation arrival. |
| ST-C6 | 3 | As a customer I want the elevator display to show me the location of my spot. |
| ST-C7 | 3 | As a customer I want to be notified of holiday operating hours. |
| ST-C8 | 2 | As a customer I want the website to display promotional offers or discounts, if any are available at the time. |
| ST-C9 | 2 | As a customer I want to receive a reminder by SMS or e-mail when my reservation time is approaching. |
| ST-C10 | 4 | As a customer I want a confirmation after I make a reservation. |
| ST-C11 | 5 | As a customer, I want to seamlessly overstay my reservation at a surcharged rate. |
| ST-C12 | 5 | As a customer, I want to park my car without a reservation, if space is available, in the “walk-in” area. |
| ST-C13 | 3 | As a customer, I want to be notified if I try to make a reservation with an expired payment method. |
| ST-C14 | 5 | As a customer, I want to add, edit, or delete vehicle information from my account settings. |
| ST-C15 | 4 | As a customer, I want there to be step-by-step instructions on what to do as I enter the garage |
| ST-C16 | 5 | As a customer, I want a guaranteed spot or an overbooking compensation if I make a reservation. |
| ST-C17 | 5 | As a customer, I want to log into my account and schedule a recurring parking reservation. |
| ST-C18 | 4 | As a customer, I want the option to park after my reservation grace period if vacancies are available. |
| ST-C19 | 3 | As a customer, I want to be granted garage access 5 minutes prior to my reservation if vacancy exists. |
| ST-C20 | 3 | As a customer, I want to provide my feedback through my account. |
| ST-C21 | 2 | As a customer, I want to make a reservation for my friend or family member. |
| ST-C22 | 5 | As a customer, I want to create a new account if I have not yet registered. |
| ST-C23 | 3 | As a customer, I want to reset my password through email if I cannot log in. |
| ST-C24 | 3 | As a customer, I want to see other users’ feedbacks and have the option to agree or disagree with them. |
| ST-M1 | 2 | As a manager, I want to identify and locate any cars for emergency reasons and make necessary reservation adjustments. |
| ST-M2 | 4 | As a manager, I want to specify the time increment for reservation scheduling. |
| ST-M3 | 5 | As a manager, I want to set the hourly parking rate through my account settings. |
| ST-M4 | 3 | As a manager, I want to associate special rates with specific hours or days. |
| ST-M5 | 3 | As a manager, I want to set an overstay rate through my account settings. |
| ST-M6 | 4 | As a manager, I want to receive emergency reports from an emergency response team if there are any emergencies. |
| ST-M7 | 2 | As a manager, I want to receive an alert if an unauthorized car shows up in the garage. |
| ST-M8 | 3 | As a manager, I want to receive a photo of the car before it enters the garage. |
| ST-M9 | 3 | As a manager, I want to receive a highlighted feedbacks list every week. |
| ST-BC1 | 4 | As a business customer, I want to manage driver identities and login permissions associated with my business. |
| ST-BC2 | 3 | As a business customer, I want to allow multiple drivers to view but not edit or create reservations. |
| ST-BC3 | 3 | As a business customer, I want to make a group reservation through single account. |
| ST-ER1[Future Work] | 1 | As an emergency response team, I want to be immediately notified of a fire, unauthorized access or violence. |
| ST-ER2[Future Work] | 3 | As an emergency response team, I want to be notified of malfunctions of devices and equipments. |
| ST-GO1[Future Work] | 3 | As the parking garage owner, I want to check out the monthly customer flow and total revenue. |
| ST-GO2[Future Work] | 3 | As a parking garage owner, I want to be notified the reasons and results of unexpected cost, for instance, fire, car scratch. |
| ST-P | 4 | As the payment processing agent, I want to get the detail of essential history information, including daily rate, customer information and parking time. |

**Non-Functional Requirements**

|  |  |  |
| --- | --- | --- |
| Identifier | Priority Weight | Requirement |
| ST-NF1 | 5 | The active number of cars in the garage will be monitored by entry and exit plate scanners. |
| ST-NF2 | 5 | The presence of a vehicle in each spot will be monitored by photo-sensors in each parking spot. |
| ST-NF3 | 5 | The online scheduling software will only display available parking spots in addition to an allowed tolerance of overbooked spots. |
| ST-NF4 | 4 | The online scheduling software will automatically adjust the reservation cost based upon any premiums and discounts at that time. |
| ST-NF5 | 3 | The online scheduling software will automatically reduce the reservation limitation during holidays. |
| ST-NF6 | 5 | Upon departure, a user’s payment method will be billed accordingly. |
| ST-NF7 | 3 | If requested by the customer, notifications will be sent a set period prior to reservation start and end time. |
| ST-NF8 | 5 | Management specification rules including hourly rate, overstay premium, special pricing and grace period will seamlessly factor into the customer’s cost. |
| ST-NF9 | 4 | The information of the vehicle should be checked, including owner’s name, make, model, color, and car accident history |
| ST-NF10 | 3 | If requested by the customers, notifications about changing of operating hours will be sent to them by SMS or e-mail. |
| ST-NF11 | 3 | The license plate numbers of the cars parked in the garage will be automatically matched with their parking spot numbers. |
| ST-NF12 | 2 | If customers choose to use QR code, the system will automatically recognize the QR code they provide. |
| ST-NF13 | 5 | The garage will have internal cameras linked with servers to provide information about the security of the garage |

**On-Screen Appearance Requirements**

|  |  |  |
| --- | --- | --- |
| Identifier | Priority Weight | User Story |
| ST-AR1 | 5 | As a customer, I want a secure login screen protecting my online account.  LoginPage.PNG |
| ST-AR2 | 4 | As a customer, I want to access my active reservations in a single click once logged in.  homepage.PNG |
| ST-AR3 | 5 | As a customer, I want to view available reservation slots and an estimated cost before making a single-time reservation.  newreservation.PNG |
| ST-AR4  [Future Work] | 4 | As a customer, I want to access my current reservations with a single click after loggin in.  IMG_0564.JPG |
| ST-AR5  [Future Work] | 4 | As a customer, I want to cancel my reservations with three clicks after logging in.  IMG_0565.JPG |
| ST-AR6  [Future Work] | 3 | As a customer, I want to receive a notification several hours before predetermined time.IMG_0566.JPG |
| ST-AR7  [Future Work] | 4 | As a customer, I want to check my reservations history after log in.  sCapture.PNG |
| ST-AR8  [Future Work] | 3 | As a customer, I want to receive notifications for discount and check it out on the calendar.1.PNG2.PNG |
| ST-AR9 | 5 | As a customer, I want to add/edit or remove payment information within two clicks once logged in.  payment.PNG |
| ST-AR10 | 3 | As a manager, I want to view garage information within 1 click of logging in.Screen Shot 2016-05-01 at 5.28.50 PM.png |
| ST-AR11  [Future Work] | 4 | As a manager, I want to view active reservations within 2 clicks of logging in.  IMG_0570.JPG |
| ST-AR12  [Future Work] |  | As a customer, I want to confirm my parking detail as soon as leaving the garage.IMG_2211.JPG |
| ST-AR13 | 4 | As a customer, I want to be allowed to reset my password.  passwordreset.PNG |
| ST-AR14  [Future Work] | 4 | As a manager, I want to check the emergency report within 1 click of logging in.  20160207_165632.jpg |
| ST-AR15 | 5 | As a customer, I want to create a new account if I have not registered.  newuser.PNG |
| ST-AR16 | 2 | As a customer, I am allowed to edit my personal preferences in order to get better services.  profile.PNG |
| ST-AR17  [Future Work] | 3 | As a customer, I am allowed to leave my feedback through website or app when I complete my parking.  20160207_183757.jpg |

**Functional Requirements Specification:**

**Stakeholders**

* Parking Garage Manager; ST-M#
* Vehicle Operator (Customer); ST-C#
* Emergency Response Team; ST-ER# [This has not been implemented: Future Work]
* Parking Garage Owner; ST-GO# [This has not been implemented: Future Work]
* Payment Processing Company; ST-P#
* Business Account Manager (Business Customer); ST-BC#

**Actors and Goals**

*Parking Garage Manager:* (Initiating Actor) The role of the parking garage manager is to implement policies utilized by the customers. These responsibilities include but aren’t limited to: setting prices, establishing days & hours of operation and setting a reservation grace period. The goal of the parking garage manager is to satisfy all customers and optimize garage usage and profits.

*Vehicle Operator- Customer (Initiating Actor)* The role of the vehicle operator is to utilize available features of the garage to occupy a parking space during a set period of time. The goal of the Customer is to access convenient and headache free parking.

*Emergency Response Team (Participating Actor)* The role of the emergency response team is to respond to emergencies such as a fire, electrical outages, safety issues with the users, and watch over security camera feedback. The goal of the Emergency Response Team is to ensure that no users of the system experience danger or injuries while using the garage. [Future Work]

*Parking Garage Owner (*Initiating Actor) The role of the parking garage owner is to oversee the manager and make high-level administrative decisions for his investment. The goal of the parking garage owner is to maximize profits.

*Payment Processing Company (Participating Actor)* The role of the payment processing company is to process the payments made by users through credit cards and other “virtual” payment methods. They will be given a fee for this service. The goal of the Payment Processing Company is to maximize profits and ensure the safe transfer of funds between the user and the garage. [Future Work]

*Business Account Manager (Initiating Actor)* The business account manager is essentially a business customer. This person’s role is to use the account given by his business to make multiple vehicle/person reservations for parking spaces when needed. The goal of the business account manager is to efficiently manage multiple drivers and vehicles within their business account.

*Plate Scanner (Participating Actor):* The role of the plate scanner is to scan the user’s license plate number and transfer it to the elevator console. The goal of the plate scanner is to correctly interpret the license plate number and relay the information to the centralized server for efficient processing.

*Elevator Console (Participating Actor):* The role of the elevator console is to check if the information given by customer matches the data saving in their account. This information will come in the form of a reservation confirmation number when the customer’s licence plate is not on file. The goal of the Elevator Console is to provide a mechanism through which a customer can still access the garage without having registered their license plate previously.

*Occupancy Photosensor (Participating Actor):* The role of the occupancy photosensor is to check if the parking spot is occupied or vacancy. The goal of the Occupancy Photosensor is to relay the status of every parking spot to the centralized computer system.

*Garage Website (Participating Actor) :* The role of the website is to provide a platform which allows customers to make the reservations and edit their personal information. In addition, the garage managers could check the garage statics and receive important messages about garage within their accounts. The goal of the website is to tie together all aspects of the parking garage into a cohesive user experience.

*Garage Console (Participating Actor) :* The role of the garage console is to compute and charge the parking fee when customer leaves the garage. The goal of the Garage Console is to fetch current pricing schemes and elapsed time from the centralized server when a user leaves the garage. The user will be billed appropriately.

**Use Cases**

*Casual Descriptions:*

UC-1:Registration - User A wishes to create a user account through the garage website. User A provides his/her personal information on the login page of [www.galuwa.com](http://www.galuwa.com). After server confirmation, he/she can log in using their email and password. The user can then configure vehicle and payment information from the quick links available on their account home page.

[C1, C3, C7, C9, C14, C20, C22, C23, C24, BC1]

UC-2:Reservation - User A wishes to make a single or contracted reservation through the garage website. User A can log in to their user account and create a reservation through the quick links on their home page. User A will receive a confirmation from the website.

[C1, C2, C10, C13, C16, C17, C21, BC2, BC3]

UC-3:Parking - User A has arrived the parking garage and wishes to parks his vehicle into the garage. User A has to provide a QR code or confirmation number to get the access to elevator. [Future Work]

[C4, C5, C6, C12, C15, C19, ER1, ER2]

UC-4:Overstay - User A overstays his reservation and did not make an extension. User A have to pays a surcharged rate due to the parking policy when he leaves. User A will receive a confirmation email later.

[C11, C18, M5]

UC-5:Understay - User A has made his reservation online and arrived on time. User A now has to leave earlier and the spot is left to be vacant. User A will be charged at full price and has no more access to the spot.

[C1, C5, C6, C11]

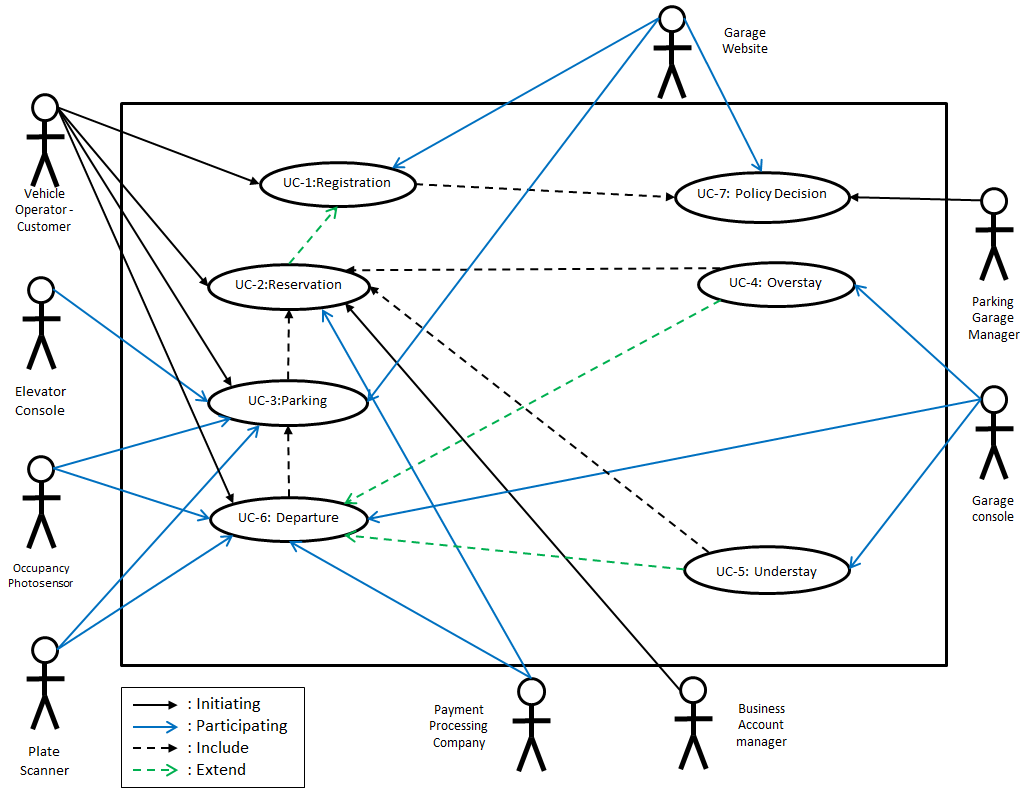
UC-6:Departure - User A exits the garage on time.User A ‘s payment method will be automatically charged when the exit camera detects his leaving. User A will receive his receipt through email later.

[C11, P]

UC-7:Policy Decision - Manager B logs into the garage administrative panel to set hourly rates and this month’s specials

[C7, C8, C16, C17, C18, M1, M2, M3, M4, M5, M6, M7, M8, M9, GO1, GO2, P]

**Use Case Diagram**

****

**Traceability Matrix**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Requirements** | **Priority Weight** | **UC-1** | **UC-2** | **UC-3** | **UC-4** | **UC-5** | **UC-6** | **UC-7** |
| **ST-C1** | **5** | **x** | **x** |  |  | **x** |  |  |
| **ST-C2** | **5** |  | **x** |  |  |  |  |  |
| **ST-C3** | **5** | **x** |  |  |  |  |  |  |
| **ST-C4** | **5** |  |  | **x** |  |  |  |  |
| **ST-C5** | **4** |  |  | **x** |  | **x** |  |  |
| **ST-C6** | **3** |  |  | **x** |  | **x** |  |  |
| **ST-C7** | **3** | **x** |  |  |  |  |  | **x** |
| **ST-C8** | **2** |  |  |  |  |  |  | **x** |
| **ST-C9** | **2** | **x** |  |  |  |  |  |  |
| **ST-C10** | **4** |  | **x** |  |  |  |  |  |
| **ST-C11** | **5** |  |  |  | **x** | **x** | **x** |  |
| **ST-C12** | **5** |  |  | **x** |  |  |  |  |
| **ST-C13** | **3** |  | **x** |  |  |  |  |  |
| **ST-C14** | **5** | **x** |  |  |  |  |  |  |
| **ST-C15** | **4** |  |  | **x** |  |  |  |  |
| **ST-C16** | **5** |  | **x** |  |  |  |  | **x** |
| **ST-C17** | **5** |  | **x** |  |  |  |  | **x** |
| **ST-C18** | **4** |  |  |  | **x** |  |  | **x** |
| **ST-C19** | **3** |  |  | **x** |  |  |  |  |
| **ST-C20** | **3** | **x** |  |  |  |  |  |  |
| **ST-C21** | **2** |  | **x** |  |  |  |  |  |
| **ST-C22** | **5** | **x** |  |  |  |  |  |  |
| **ST-C23** | **3** | **x** |  |  |  |  |  |  |
| **ST-C24** | **3** | **x** |  |  |  |  |  |  |
| **ST-M1** | **2** |  |  |  |  |  |  | **x** |
| **ST-M2** | **4** |  |  |  |  |  |  | **x** |
| **ST-M3** | **5** |  |  |  |  |  |  | **x** |
| **ST-M4** | **3** |  |  |  |  |  |  | **x** |
| **ST-M5** | **3** |  |  |  | **x** |  |  | **x** |
| **ST-M6** | **4** |  |  |  |  |  |  | **x** |
| **ST-M7** | **2** |  |  |  |  |  |  | **x** |
| **ST-M8** | **3** |  |  |  |  |  |  | **x** |
| **ST-M9** | **3** |  |  |  |  |  |  | **x** |
| **ST-BC1** | **4** | **x** |  |  |  |  |  |  |
| **ST-BC2** | **3** |  | **x** |  |  |  |  |  |
| **ST-BC3** | **3** |  | **x** |  |  |  |  |  |
| **ST-ER1** | **1** |  |  | **x** |  |  |  |  |
| **ST-ER2** | **3** |  |  | **x** |  |  |  |  |
| **ST-GO1** | **3** |  |  |  |  |  |  | **x** |
| **ST-GO2** | **3** |  |  |  |  |  |  | **x** |
| **ST-P** | **4** |  |  |  |  |  | **x** | **x** |

**Fully Dressed Description**

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case UC-1: | | | Registration |
| Related Requirements: | | | ST-C1, ST-C3, ST-C7, ST-C9, ST-C14, ST-C20, ST-C22  ST-C23, ST-C24, ST-BC1 |
| Initiating Actor: | | | Vehicle Operator - Customer, Business Account Manager |
| Actor’s Goal | | | To create a account on [www.galuwa.com](http://www.galuwa.com) which allows customer to make reservations and modify the personal and vehicle information. |
| Participating Actors: | | | Website Server |
| Preconditions: | | | Customers have to provide the authenticated information required by the garage website. |
| Postconditions: | | | Customer account information will be stored in the website database and can be modified by the customer at any time. |
| Flow of Events for main Success Scenario: | | | |
| → | 1 | The customer presses the “Don’t have an account” button on garage user login page. | |
| ← | 2 | The garage website displays a form for the customer to fill out. This form contains the First Name, Last Name, Email, Password, Password Confirmation, Zip-Code and Account Type fields. | |
| → | 3 | The customer fills out the required information and presses “Submit.” | |
| ← | 4 | The garage website verifies the information. This check ensures that the email does not already exist within the database of the users. The zip-code is also used to lookup local garages for the customer to use. | |
| ← | 5 | The garage website stores the customer’s information into a database and allows the customer to log in with the username and password. | |
| Flow of Events for Extensions (Alternate Scenarios): | | | |
| →  ← | 4A | If the email already exists in the database, the garage website highlights the invalid information and prompts the user to change it.  The customer can change the unverified information and resubmit the form. Proceed to Step 4. | |
| →← | 3A | If the password and password confirmation fields do not match, the user will be asked to retype both fields. | |

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case UC-2: | | | Reservation |
| Related Requirements: | | | ST-C1, ST-C2, ST-C3, ST-C7, ST-C8, ST-C10, ST-C16, ST-C17, ST-C21, ST-BC2, ST-BC3 |
| Initiating Actor: | | | Vehicle Operator - Customer, Business Account Manager |
| Actor’s Goal | | | To make a reservation for a parking spot during an available time period. |
| Participating Actors: | | | Payment Processing Company, Garage Website |
| Preconditions: | | | Customers must have had to make an account on the website to be logged in. |
| Postconditions: | | | There will be an active reservation schedule accessible through the customer’s account |
| Flow of Events for main Success Scenario: | | | |
| → | 1 | The customer presses “New Reservation” link in his/her account home page. | |
| ← | 2 | The website displays the metadata associated with making a reservation such as the time and date. | |
| → | 3 | The customer chooses the time, reviews expected pricing, and presses “Submit.” | |
| ← | 4 | Garage website verifies the reservation information and uploads it to the server. | |
| → | 5 | The confirmation of the time and date are displayed to the user along with a QR code and confirmation number. | |

|  |
| --- |
| FlowChart for Alternative Scenarios |

|  |  |  |
| --- | --- | --- |
| →  ← | 4a | The customer tries to create a reservation that conflicts with their existing reservation. They will be shown an error and asked to try again. Proceed to step 2. |

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case UC-3: | | | Parking |
| Related Requirements: | | | ST-C1, ST-C4, ST-C5, ST-C6, ST-C9 |
| Initiating Actor: | | | Vehicle Operator - Customer, Business Account Manager |
| Actor’s Goal | | | To park their vehicles in the garage, either by using their reservations or via the walk-in lots. |
| Participating Actors: | | | Plate Scanner, Elevator Console, Occupancy Photosensor. |
| Preconditions: | | | At least one spot must be available in either the reserved or walk-in lots. |
| Postconditions: | | | Photo-Sensors will report the presence of a vehicle in the spot assigned to the user. |
| Flow of Events for main Success Scenario: | | | |
| → | 1 | The customer drives into the elevator after arriving at the garage. | |
| ← | 2 | The plate scanner scans the license plate of the customer’s vehicle and transfers it to elevator console. | |
| ← | 3 | The elevator console confirms the customer’s information and selects a vacant parking spot for the user. | |
| ← | 4 | The elevator lifts the customer’s vehicle to the upper level and displays the user’s assigned parking space. | |
| → | 5 | Customer parks vehicle into assigned spot. | |
| ← | 6 | Occupancy photosensor detects customer vehicle has parked into the spot. | |
| Flow of Events for Extensions (Alternate Scenarios): | | | |
| →  ← | 2B | Elevator console can’t find the license plate in the database and asks the customer to provide the confirmation number/QR code.  Customer provides his confirmation number/QR code (If it’s right, move to 3. If not move to 2C). | |
| →  ← | 2C | Elevator console can’t find the confirmation number/QR code in the database and asks customer to leave or drive to the walk-in area.  Customer chooses leave or drives to the walk-in area. | |
| →← | 3A | Despite customer reservation, the garage is overbooked and no vacant spots are available for the customer. The customer will be given a refund and an additional discount coupon decided upon by management. | |

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case UC-4: | | | Overstay |
| Related Requirements: | | | ST-C11, ST-C18, ST-M5 |
| Initiating Actor: | | | Garage Console. |
| Actor’s Goal | | | To seamlessly overstay their allotted reservation time. |
| Participating Actors: | | | Occupancy Photosensor, Parking Garage Manager,Vehicle Operator - Customer |
| Preconditions: | | | The customer’s parking exceeds the reserved time. |
| Postconditions: | | | The customer pays a surcharged rate for additional stay when leaving the garage. |
| Flow of Events for main Success Scenario: | | | |
| → | 1 | The garage console flags the overstayed vehicle and marks it within the database. | |
| ← | 2 | The system will automatically adjust and recalculate all vacant spaces. | |
| ← | 3 | The customer is notified that his vehicle is an overstay and the details about the overstay policy including pricing. | |
| Flow of Events for Extensions (Alternate Scenarios): | | | |
| → | 4 | If the customer’s vehicle is not retrieved in a set amount of time specified by the administration, a towing company will be called. | |

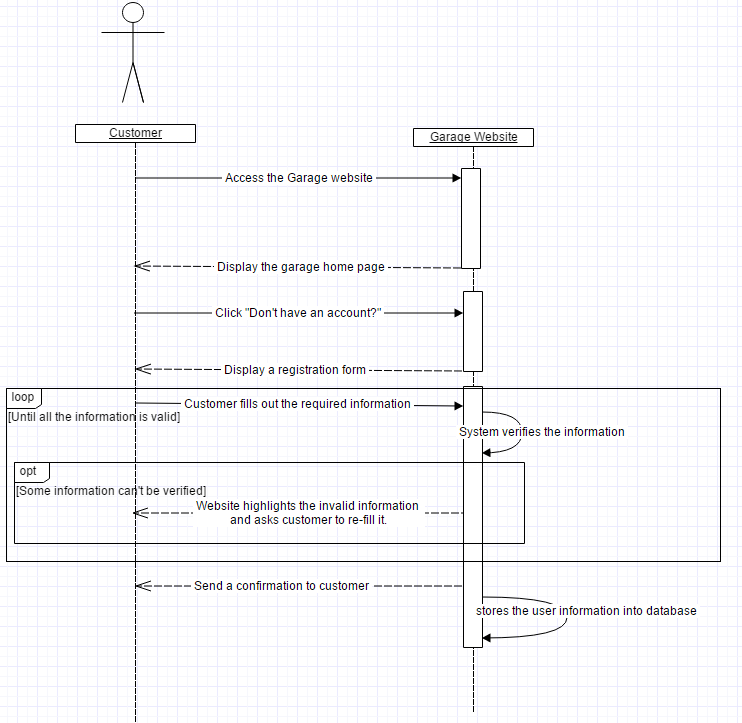
|  |  |  |  |
| --- | --- | --- | --- |
| Use Case UC-5: | | | Understay (Normal Usage) |
| Related Requirements: | | | N/A |
| Initiating Actor: | | | Vehicle Operator - Customer |
| Actor’s Goal | | | This is normal operating procedure for a vehicle operator. |
| Participating Actors: | | | Payment Processing Company, Plate Scanner, Garage Console. |
| Preconditions: | | | The customer leaves on time or prior to the end of their reservation. This frees a parking spot. |
| Postconditions: | | | Once the customer leaves the parking garage their reservation is complete and will have to make a new reservation if they wish to re-enter the garage |
| Flow of Events for main Success Scenario: | | | |
| → | 1 | Customer leaves the garage on time or earlier than his/her scheduled departure. | |
| ← | 2 | The Occupancy Photosensor detects the vacant parking spot and signals the system and a customer is leaving. | |
| ← | 3 | The Exit Plate scanner detects the departure of the customer and send notifies the central system. | |
| ← | 4 | The Garage console marks this reservation as “completed” and updates the database. | |
| → | 5 | The customer receives an email/SMS thanking them for their service and requesting feedback. | |
| Flow of Events for Extensions (Alternate Scenarios): | | | |
|  |  |  | |

|  |  |  |  |
| --- | --- | --- | --- |
| Use Case UC-6: | | | Departure |
| Related Requirements: | | | ST-C11 |
| Initiating Actor: | | | Vehicle Operator - Customer |
| Actor’s Goal | | | To leave the garage and pay the parking fee. |
| Participating Actors: | | | Payment Processing Company, Plate Scanner, Garage Console, Occupancy Photosensor. |
| Preconditions: | | | Customers is at the exit of the garage. |
| Postconditions: | | | The system is updated to reflect the change in active reservations. |
| Flow of Events for main Success Scenario: | | | |
| → | 1 | The customer’s vehicle arrives to the garage exit. | |
| ← | 2 | The plate scanner scans the license plate number of the arriving vehicle and identifies the user. | |
| ← | 3 | The garage console charges the appropriate fee based on the parking time and rate set by the garage manager. Overstays will be factored into the price. | |
| → | 4 | Customer’s selected payment method will be charged automatically with the Payment Processing Company’s assistance. | |
| → | 5 | A email notification will be sent to both customer. | |
| Flow of Events for Extensions (Alternate Scenarios): | | | |
| →  ← | 1 | The customer overstays his reservation. The system will adjust for vacant spaces and flag the user’s reservation with overstay pricing. Proceed to step 4. | |

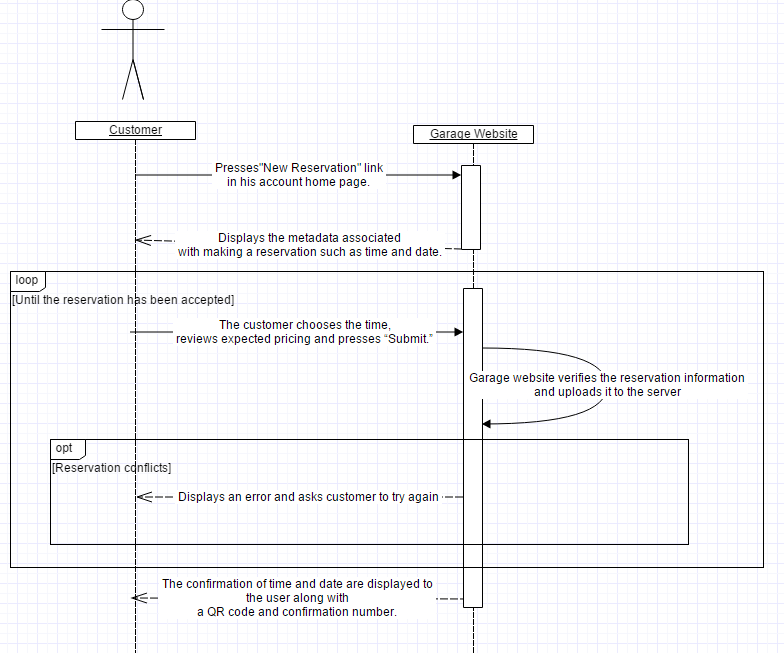
|  |  |  |  |
| --- | --- | --- | --- |
| Use Case UC-7: | | | Policy Decision |
| Related Requirements: | | | ST-C7, ST-C8, ST-C16, ST-C17, ST-C18, ST-M1, ST-M2, ST-M3, ST-M4, ST-M5, ST-M6, ST-M7, ST-M8, ST-M9, ST-P, ST-GO1, ST-GO2 |
| Initiating Actor: | | | Parking Garage Manager |
| Actor’s Goal | | | To set parking rates, promotional offers or any policies which are related to garage. |
| Participating Actors: | | | Garage Website |
| Preconditions: | | | The parking garage manager is logged in to their account. |
| Postconditions: | | | The system reflects the manager’s changes. |
| Flow of Events for main Success Scenario: | | | |
| → | 1 | The parking garage manager logs into his/her account and accesses the “Rates” or “Settings” panel. | |
| → | 2 | The garage manager is able to view existing policies in addition to editing, deleting, and creating new ones. | |
| → | 3 | The new policies are verified by the server and are uploaded. All changes are reflected on the website. Existing reservations will not be affected by price changes. | |
| → | 4 | Any contracted reservation customers are notified by email or SMS of the new rates and will be subject to them starting the next billing period. | |
| Flow of Events for Extensions (Alternate Scenarios): | | | |
| →← | 1 | New policy conflicts with an existing policy. The manager will be notified of the conflicting rules and asked if the old rule should be overwritten. Proceed to step 3. | |

**System Sequence Diagrams**

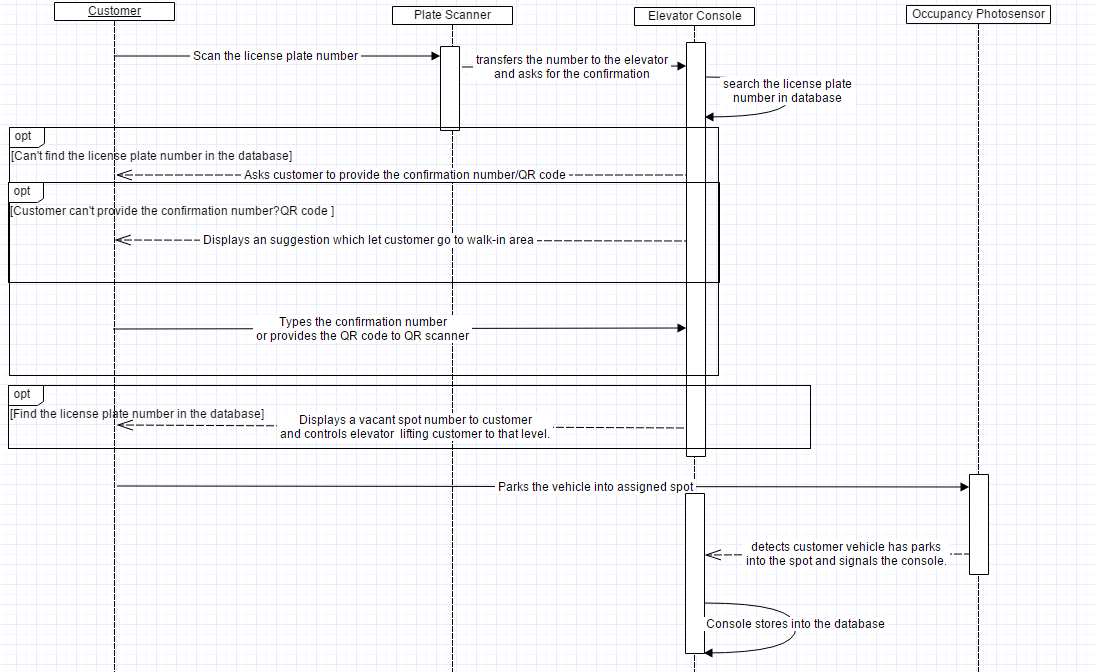
**UC-1 Registration:**

****

**UC-2 Reservation:**

****

**UC-3 Parking:**

****

**Effort Estimation:**

UCP- Use Case Points

PF- Productivity Factor

UUCP- Unadjusted Use Case Point

TCF- Technical Complexity Factor

ECF- Environmental Complexity Factor

UAW- Unadjusted Actor Weight

UUCW- Unadjusted Use Case Weight

Duration = UCP x PF

PF=28 hours per use case

UCP = UUCP x TCF x ECF

UUCP=UAW+UUCW

UAW= 2 x Simple + 2 x Average + 2 x Complex= 2 x 1 + 2 x 2 + 2 x 3 = 12

UUCW = 3 x Simple + 2 x Average + 2 x Complex = 3 x 5 + 2 x 10 + 2 x 15 = 65

UUCP = 12 + 65 = 77

TCF = 0.6 + (TF/100)

TF = 35

TCF = 0.6 + (35/100) = 0.95

ECF = 1.4 + (-0.03 x EF)

EF = 8

ECF = 1.4 + (-0.03 x 8) = 1.16

UCP = 77 x .95 x 1.16 = 84.85

Duration = 84.85 x 28 = 2,375.8

**Domain Analysis:**

Domain Model

Untitled Diagram (2).png

This Domain Model shows the interaction of System Concepts. This Diagram serves as a general flowchart through the computer portion of our system.

System Operational Contracts

|  |  |
| --- | --- |
| **Operation** | Registration |
| **Class Invariants** | The user requires access to an account on www.galuwa.com. |
| **Preconditions** | * The user must not yet have a registered account on galuwa.com. * The user must fill in the necessary fields on the registration form. * Must enter a valid e-mail address. * Must enter a password between 8-12 characters(no punctuation/symbols). |
| **Postconditions** | * An account is created for the user and this information is stored in the database. * The user is directed to the home page of his account with a welcome message. |

|  |  |
| --- | --- |
| **Operation** | Reservation |
| **Class Invariants** | The user must be registered and logged in. |
| **Preconditions** | * There must be at least one reservation slot open. * The user must select an open slot. * A valid payment must be made. |
| **Postconditions** | * A reservation is made, the slot is booked. * The user is e-mailed a confirmation number. * The user is e-mailed a QR code. |

|  |  |
| --- | --- |
| **Operation** | Parking |
| **Class Invariants** | The system detects the vehicle through the various devices. |
| **Preconditions** | * The user’s vehicle has been authenticated by the plate scanner or the user has employed his confirmation number/QR code. * The user’s vehicle is granted access to the lift. * The lift brings the user up to the allocated floor. |
| **Postconditions** | * The occupancy sensor keeps track of the vehicle’s status. * The lone user exits the garage |

|  |  |
| --- | --- |
| **Operation** | Overstay |
| **Class Invariants** | The user must have made a reservation. |
| **Preconditions** | * The user’s reservation period must be over. * The occupancy sensor must send the vehicle’s “present” status to the system. |
| **Postconditions** | * The user is charged with a surcharged rate. * The user exits the garage with his vehicle within specific time else is towed and pays a towing fee. * The occupancy photosensor must have marked the space as empty. |

|  |  |
| --- | --- |
| **Operation** | Understay |
| **Class Invariants** | The user must have made a reservation. |
| **Preconditions** | * The user leaves the garage with his vehicle before the end of his reservation period. |
| **Postconditions** | * To re-enter the garage, even before the actual end of his reservation period, the customer must make a new reservation. * The occupancy photosensor must have marked the space as empty. |

|  |  |
| --- | --- |
| **Operation** | Departure |
| **Class Invariants** | The user acquires a new destination. |
| **Preconditions** | * The user requests an exit status. * The user has paid any additional fees his account has acquired for any reason in our terms of use. |
| **Postconditions** | * The occupancy sensor acquires and transfers exit information to the system * The database marks the vehicle as “departed” and updates a free space. |

|  |  |
| --- | --- |
| **Operation** | Policy Decision |
| **Class Invariants** | The manager maintains/enforces the policies. |
| **Preconditions** | * The manager must log into his super user account. * The manager must consult on which policies to enlist. |
| **Postconditions** | * The user’s must follow the policies. * The database and website are updated to reflect the policy changes. |

Mathematical Model

This software package will feature the ability to simulate the reservations, arrivals, and departures going through the garage. This simulation of artificial customers will be done using 2 Poisson processes. One of the Poisson processes is for the number of reservations being made, and the other process is for the number of walk-in customers that arrive. For a Poisson process with an average arrival rate , the probability of seeing arrivals in time interval t is:

****

The interarrival times are defined by the exponential distribution:



This simulation will assume that a customer will not make a reservation more than 24 hours in advance of their desired arrival time, and no less than 1 hour before. If reserved parking is booked the customer will attempt to make a reservation for 5 minutes later, and will continue until they get a reservation. We will assume that any customers that made a reservation will arrive to the garage on time. Reserve customers that made a reservation for the same time will be randomly entered into a queue to enter garage. Upon arrival, both walk-in and reserve customers will be assumed to enter the garage and park in optimal time. So the time it will take for a customer at the front of the queue to park T, will be:



Where c is the time to check in at the front console (it should take slightly longer for walk-in customers), e is the time for the elevator to lift the customer, this will be 0 for walk-in customers, and p is the time to park. When the customer checks in, they are assigned the available parking space closest to the exit. We are assuming the entrance and exit are as far apart as possible, this way cars parking at the same time should never cross paths or get in each other’s way . The simulation will know the shortest route to any park for a distance of *d*, and will assume the customers will always move at the garage’s speed limit *s*. When dealing with a queue to enter the walk-in floor, the next car begins their check in process as soon as the car in front finishes theirs. So if a walk-in customer is ith in line, the time it will take for them to park is:



This simulation will assume that a customer will not enter the line if it will not be possible to park by the time they make it to the front of the queue. For the reserve customers they will start their check in process after the car in front finishes their check-in process, takes the elevator up, and the elevator comes down. So if a reserve customer that is ith in line will take:



Departure from the garage will be based on the reservation the customer made. This simulation will assume that customers will make a reservation of at least 10 minutes, and at most 9 hours, and will set the end of their reservation to be at a 5 minute interval of real time. In other words a customer would set the end for their reservation at 12:00 or 12:05, but would not set it at 12:02 or 12:07.

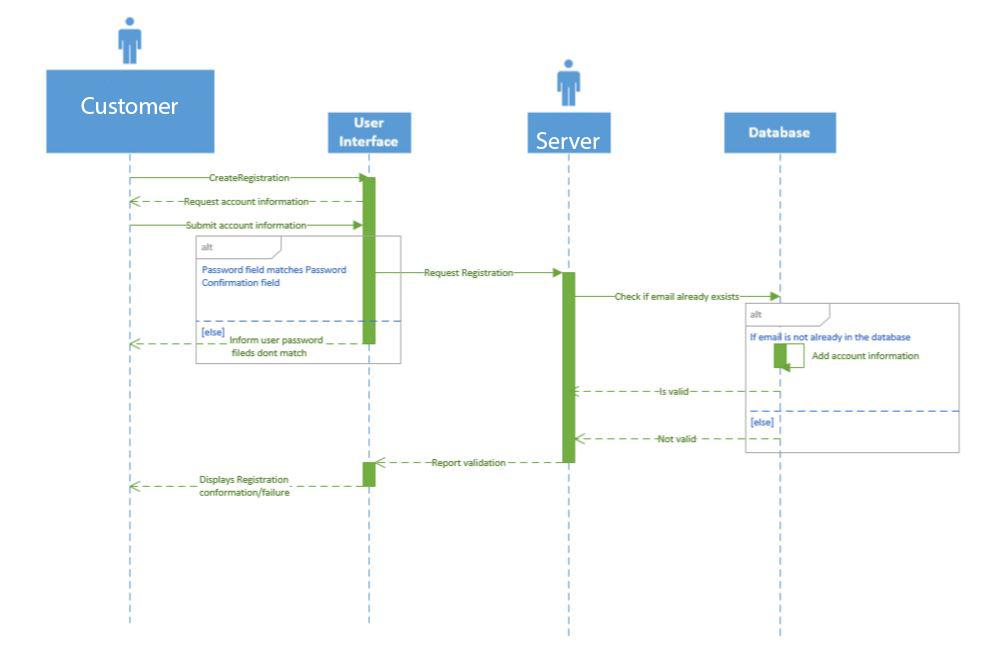
The simulation will act on the assumption that the closer it is time for a reservation will end, the more likely the customer is to depart. However this simulation will assume that customers will not depart for at least 5 minutes after parking. The probability that a given customer will depart at any given second is:



Where is the total time of the reservation in seconds, and is the seconds since the customer parked.

**Interaction Diagrams:**

UC-1 Registration



Description: This UML diagram for UC-1 Registration demonstrates the process of a user/customer creating an account on the Garage Automated system website using a method that involves the client making a request of the server, the server making a request of the database, and the result cascading back to the client.

This Interaction Diagram uses the Publisher-Subscriber Pattern as it check for a valid key and if so the subscriber receives and continues the process. This is done during checking if registration exists already.

Step 1

Customer→ User Interface: Customer presses button to begin registration process.

Step 2

User Interface→ Customer: Ask customer to fill out account information form.

Step 3

Customer→ User Interface: User fills out form and presses the “Submit” button.

Step 4

User Interface→ Website Server: Interface submits form to the Controller.

Step 4 alt

User Interface→ Customer: Informs user that both the password and password confirmation fields must match.

Step 5

Website Server→ Database: Check if customer’s email is already in use.

Step 6

Database → Website Server: Confirm customer account’s submission to database.

Step 6 alt

Database → Website Server: Customer email is already in use, registration not valid.

Step 7

Website server → User Interface: relay registration results.

Step 8

User Interface → Customer: show registration results to user.

UC-2 Reservation

[UC2_Reservation_SQ_DIAG.png](https://drive.draw.io/#G0B1vkj8y_n1eFbkZHWnpnVXJNUFU)

Description: This UML Diagram for Reservation details the interaction between the User, Client and Server as a User attempts to create a new reservation online.

This Interaction Diagram uses the Command pattern. The customer’s inputs are redirected to the server through an interface which provides uniform method signatures. Even if the Server style changes this still works.

Step 1:

Customer → Client: Customer accesses website and types in username and password to log in to his account.

Client→ Server: The clients asks the server to get the account information with the given credentials.

Server→ Client: The server(database) verifies the info and gives the client the user info. Client→ Customer: The client displays the end-user with a welcome page and his account information.

Step 2:

Customer→ Client: Customer clicks link to create a new reservation.

Client→ Server: The client asks the server for the schedule of reservations.

Server→ Client: The server sees the activation and sends the client information of the schedule. Client→ Customer: The end-user is greeted to a schedule page with dates, times, and booked and unbooked slots.

Step 3:

Customer→ Client: The customer sends information through client to book a slot with a new date and new time.

Client→ Server: The client forwards this information to the server.

Server→ Client: The server(database), after verifying the slot is available, books the slot and tells the client.

Client→ Server: Once receiving the booking, the client asks for confirmation.

Server→ Client: The server then generates a confirmation number, after making sure the request is valid, and sends it to the client.

Client→ Customer: The client finally sends this confirmation information to the end-user who can then use it as he needs when parking.

UC-3 Parking

UC3-PARKING.png

Description: This UML Diagram for Parking details the interaction between the Customer, Elevator Console, Plate Scanner, Occupancy Photosensor and parking database as a Customer arrives at the garage.

This Interaction Diagram uses the Publisher-Subscriber Pattern as well. The valid key is the automatic confirmation number. Once received by the subscriber from the publisher, the spot assignment is done and the spot number is calculated.

Step 1

Customer -> Plate Scanner: Customer arrives at the garage and has their license plate scanned to confirm an existing reservation. This reservation is verified and a spot number is displayed to the user.

Step 1 Alternate

Customer -> Elevator Console: Customer arrives at the garage with an existing reservation, but the license plate scanner is unable to link license plate to a reservation. Customer will manually input a confirmation number, the information will be verified and the spot number is displayed to the user.

Step 2

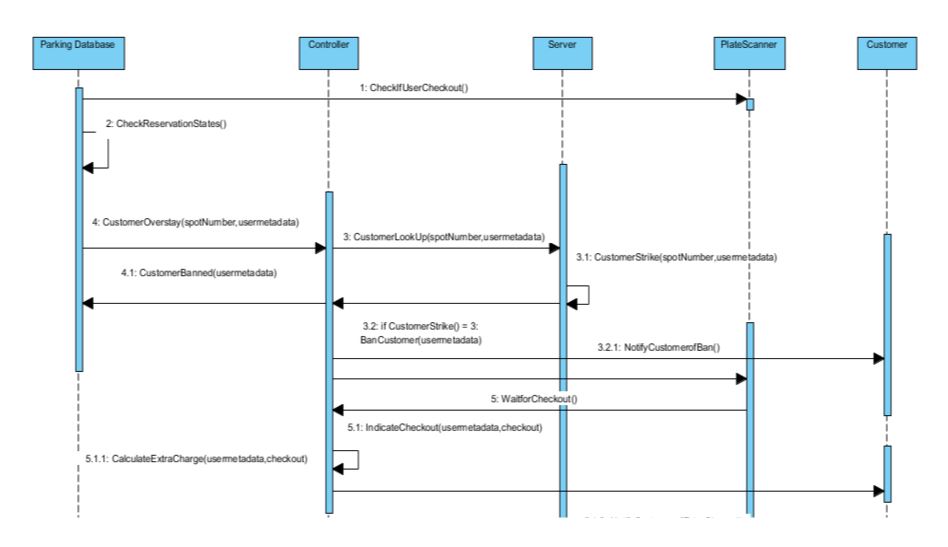
License Plate Scanner -> Occupancy Photosensor: The License Plate Scanner will listen through the Occupancy Photosensor at an assigned spot to determine that a user has parked in their spot within a timeout period.

Step 3 Alternate

Elevator Console -> Occupancy Photosensor: The Elevator Console will listen through the Occupancy Photosensor at an assigned spot to determine that a user has parked in their spot within a timeout period.

Step 4

Occupancy Photosensor -> Parking Database: Once a car has arrived, the Occupancy Photosensor will relay to the parking database that an additional vehicle has been parked in the garage.

UC-4 Overstay

Description: This UML Diagram for Overstay details the interaction between the Parking Database, Controller, Server, Plate Scanner and Customer as a Customer leaves the garage laste.

This Interaction Diagram uses the State Pattern. The controller is given the State and from there can determine the next state of the object. This is shown if the customer is waiting for checkout or has checked out or needs to be charged extra. These are all different states.

Step 1: Parking Database <-> PlateScanner Parking Database retrieves information from the PlateScanner as to whether the cars in timeperiod T have checked out

Step 2: Parking Database<-> Parking Database

Scans database for spots thats are overstaying in timeperiod T

Step 3: ParkingDataBase <-> Controller

Indicates that a customer or customers have indeed overstayed while passing their spotnumber and metadata(car #, plate number, model, etc.)

Step 4: Controller<->Server

Finds the customer with metadata and spotnumber provided

Step 5: Server <-> Server

Issues a strike for the customer for overstaying, and if the user has had 3 strikes(including current), then proceed with alternative path

Step 6: Controller<->PlateScanner

Controller waits for plate scanner to register user has left the garage

Step 7: PlateScanner <-> Controller

PlateScanner sends confirmation to Controller indicating the user has left, along with the usermetadata and checkout time

Step 8: Controller <-> Controller

Calculates the extra charge based upon the surcharge rate and how long the overstay was

Step 9: Controller <-> User

Send an email ,text, or user preference of communication to indicate that he has overstayed, gained a strike, and is paying this surcharge

Alternative Path:

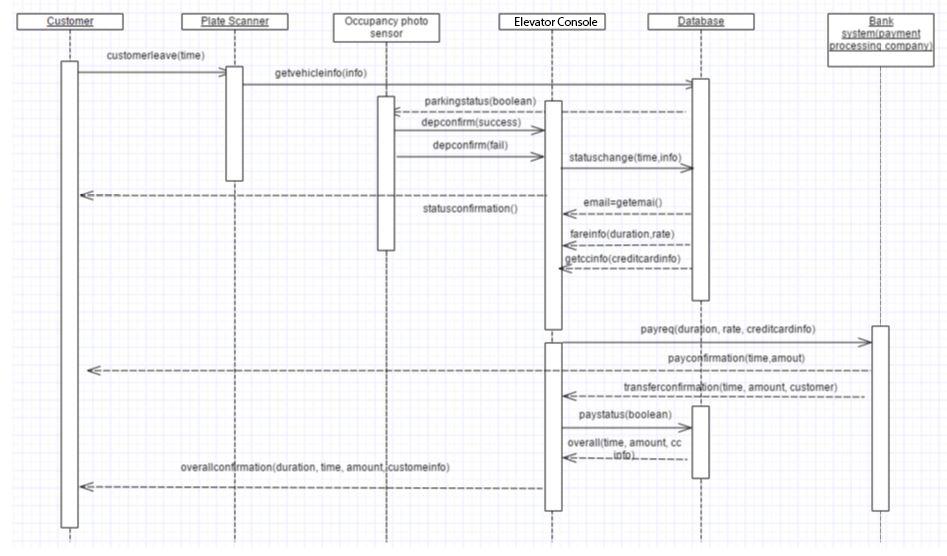
5.1: Controller<-> ParkingDatabase

Update the ParkingDatabase to indicate the user's ban from the parking garage along with his information(car, model, license plate)

5.2: Controller <-> User

Indicate to the user via email, that he has been banned from the parking garage for his overstay strikes

UC 5 - Departure



Description: This UML Diagram for Departure details the interaction between the Customer, Plate Scanner, Occupancy Photosensor, Garage Console, Database and Payment processing system.

This Interaction Diagram makes use of the Decorator pattern. It essentially uses a “linked-list,” not as in the data structure. This is seen when the customer leaves, it is linked to the plate scanner which is linked to the photosensor and database. Each part does its job not really knowing what the other object does. This is exactly how the Decorator pattern works.

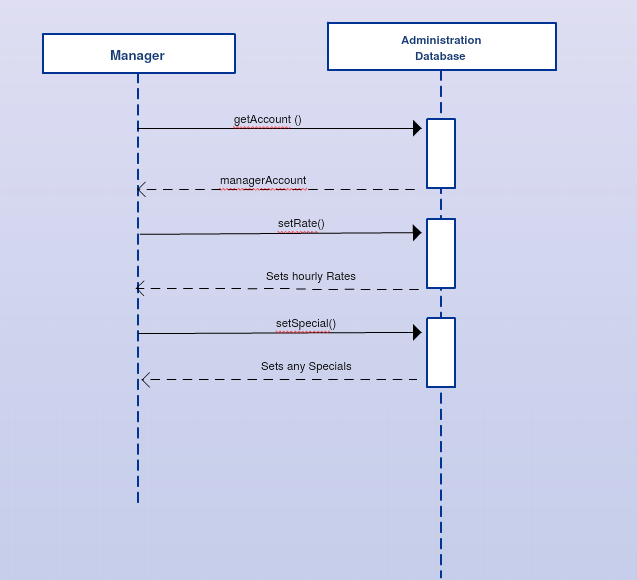
Step1. Customer->plate Scanner. Customer leave the garage and plate Scanner detect the leaving

Step 2. Plate Scanner->Occupancy Photo sensor. Plate Scanner send a message to Occupancy Photo sensor to make sure the leave

Step 3. Garage console->database. After confirmation, the Garage console ask for customer information from database

Step 4. Garage console->Payment processing company. Garage console post a payment requirement to bank system and wait for reply

Step 5. Garage console-> Customer and database. Garage console send a confirm e-mail to customer and a history log to database

UC-6 Policy Decision

Description:

This last Interaction Diagram can make use of the Proxy pattern through initiating a Protection Proxy. For example, when getting the account by the Manager he gets full access to the manager account and can perform administrative privileges such as setting rates and specials. On the other hand, when customers try to log into their system using their credentials, they do not get such high level of access.

Step 1

Manager→Administration Database: The manager first will login to his account on the administration database.

Manager→Administration Database: The manager can set hourly rates for parking in the garage.

Manager→Administration Database: The manager can set any specials for the upcoming week or month for the parking garage.

**Components Class Diagrams & Interface Specification**

In order to represent interacting components of the system, a single class diagram was broken into parts representing functional components of the whole system. Each component can be viewed as its own system or as a subsystem of the entire operation.

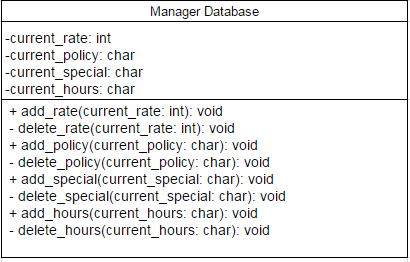
**Account Management:**

**account management.png**

**Vehicle Departure:**

**departure.png**

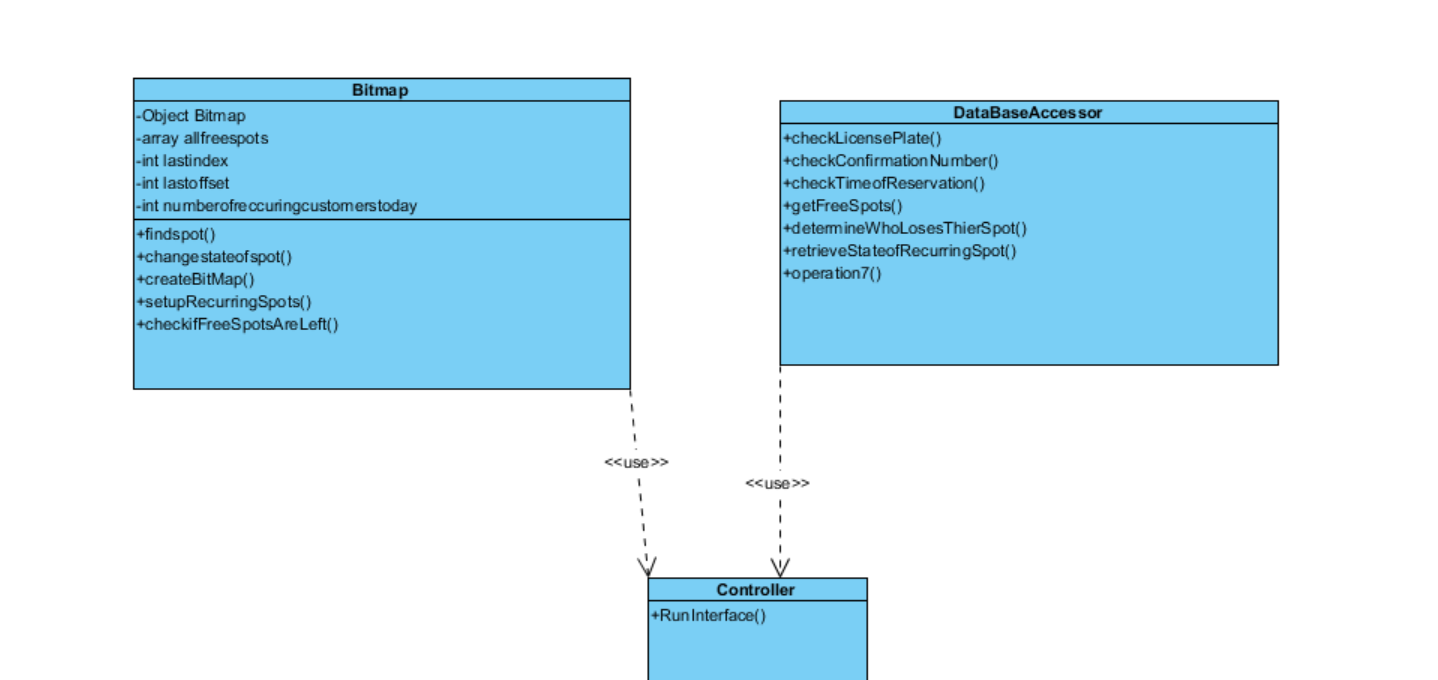
**Policy Decision:**

****

**Parking Garage Simulation:**

**Simulation.png**

**Car Arrival:**



**Traceability Matrix**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Domain Concepts | Website | Parking Interface | Outdoor Sign | Parking Database |
| Classes |  |  |  |  |
| Controller | X | X | X | X |
| Customer Database | X |  |  | X |
| Elevator Console | X | X |  | X |
| Manager account database | X |  |  |  |
| OccupancyPhotosensor |  |  |  | X |
| Outdoor Display |  | X | X | X |
| ParkingDatabaseInterface |  |  |  | X |
| Payment Processing | X |  |  |  |
| Plate Scanner | X |  |  | X |
| System manager | X |  |  |  |
| System User | X |  |  |  |
| Web Server | X | X |  |  |
| Reservation Database | X | X | X | X |
| Software Simulation | X | X |  | X |
| User interface | X |  |  | X |

**Domain Concepts:**

**OCL:**

|  |
| --- |
| **Simulation** |
| Precondition:  Status=1(Random reservation created) |
| Postcondition:  Status=0(Reservation is added to the database) |
| Invariant: User Generation, Vehicle Generation |

|  |
| --- |
| **Departure** |
| Precondition:  Status=1(vehicle in the garage) |
| Postcondition:  Status=0(vehicle leave the garage) |
| Invariant: None |

**System Architecture and System Design:**

Similar to many other software engineering based projects, our project uses an architecture revolved around databases acting as communication tools. We employ a server client model in which the server holds all of the customers information, reservations, type of membership, etc. in a series of databases. These databases then convey the necessary information when a car approaches the parking garage and exits from it. This is due to a fact that an object oriented programming language is used with the databases to properly give out spots and charge customers based upon how long they were in the spot. We currently have 4 databases, membership, reservation, and recurring and parking spots(only for departure and arrival). Including the hardware, we would also have a new system of signals being generated by hardware, and a controller hardware to sort and send the signals to the correct database/software.

**Algorithms and Data Structures:**

Software Simulation

**Algorithms:** To simulate realistic customer reservation and arrival, the software will employ two poisson processes. The arrivals will be governed by the following:

1. The database will be queried to verify availability. If spots are available, assign a spot randomly and mark it as occupied. If no parking spots are available, relay to the centralized server that an overbooked event has been registered.

2. To determine the next vehicle arrival time, generate a random number using the following equation when lambda is the average arrival rate and U is a random number between (0,1). This output will need to be scaled to obtain the next arrival time. For example: if rx = 0.3, then t(rx) = 0.3 60 minutes = 18 minutes.

**Data Structures:** The simulation will rely heavily on several objects that house data collected from various databases. The User object has a ArrayLists housing registered vehicles and active reservations.

The Garage object has ArrayLists for all currently parked vehicles, reservations and registered users.

Every program cycle, this information will be fetched from the respective database.

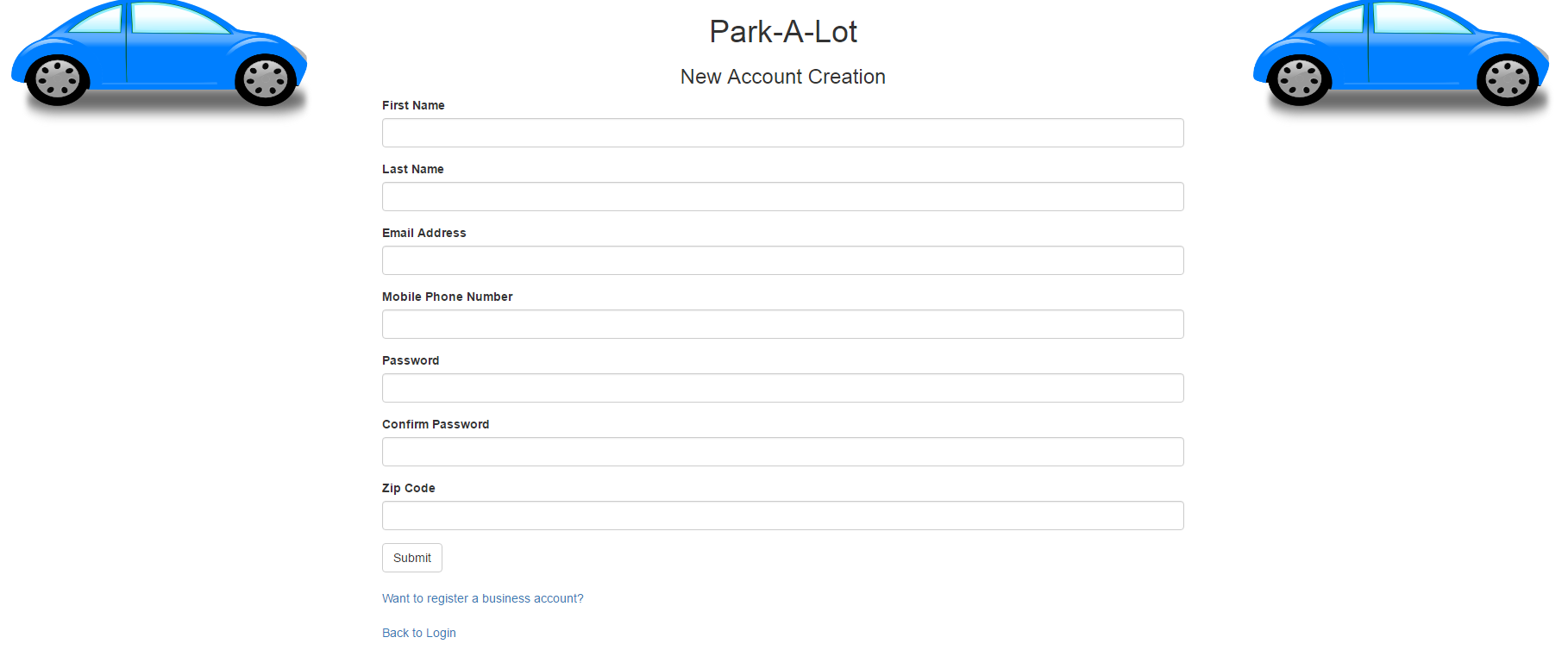
User Interface Design

**AccountManagement:**

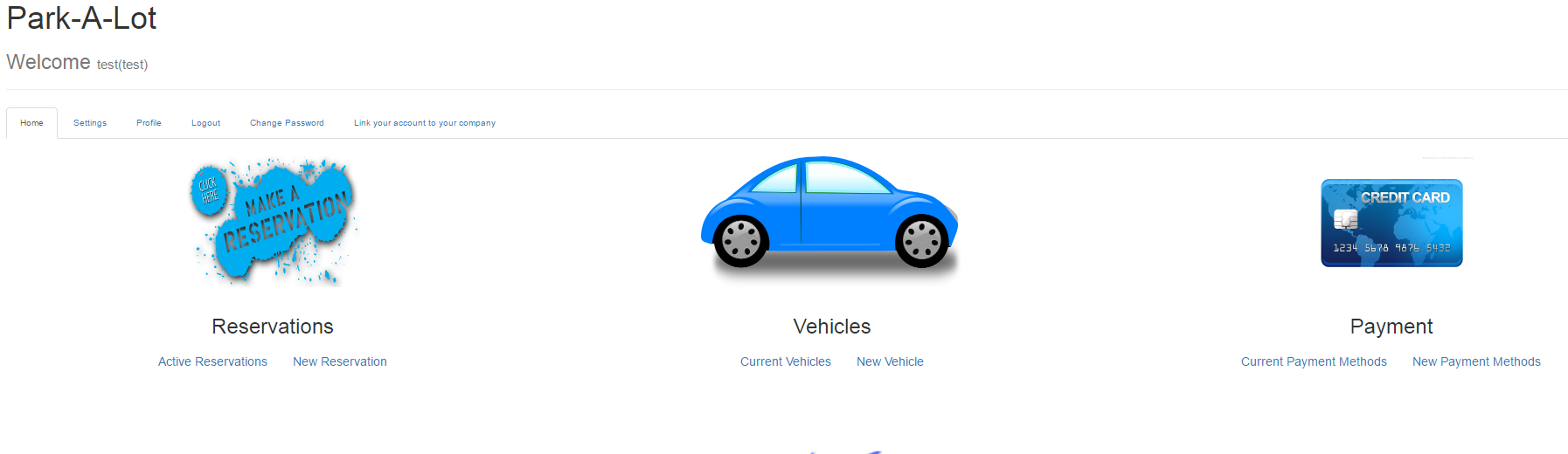
Login page: This webpage will check whether the combination of username and password the user types in is exist in the database. In addition, since we have two different types of account: Business and personal. It detects the account type automatically and directs users to their profile pages base on their account type.

****

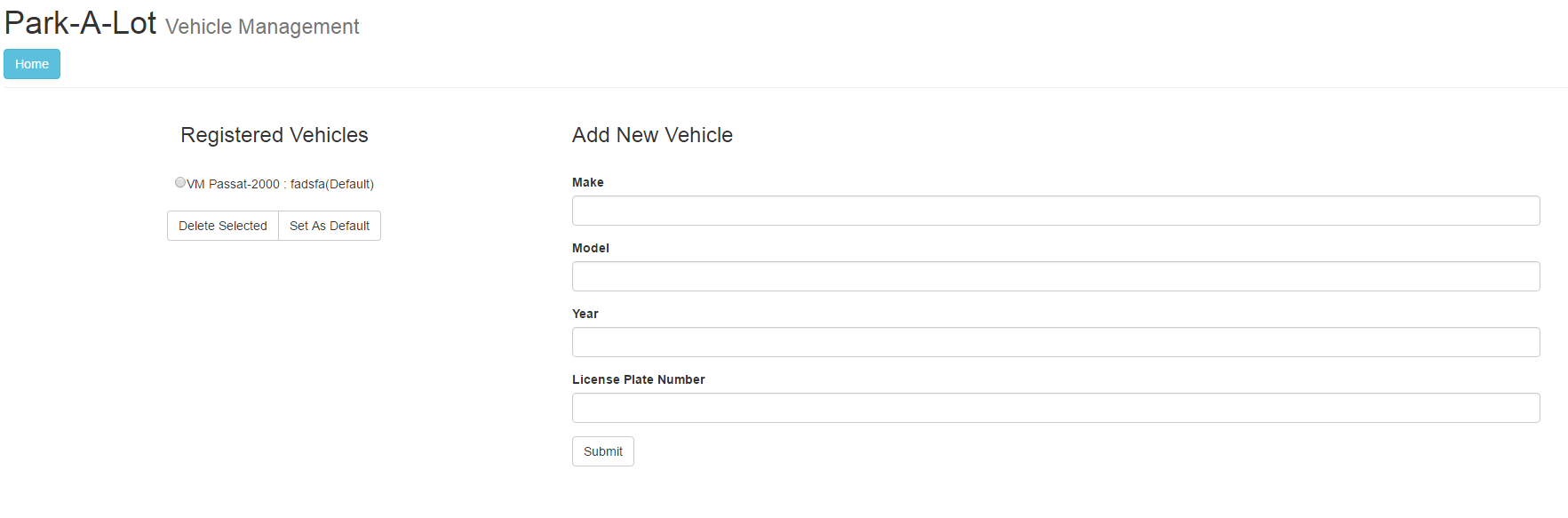
Registration Page: This webpage allows new users to register their own accounts by typing the required information. Since i use the email address as the primary key in the database, it will avoid the situation which different users have the same username. I don’t show the business account registration page here, since the algorithm of the those two registration page is same.

****

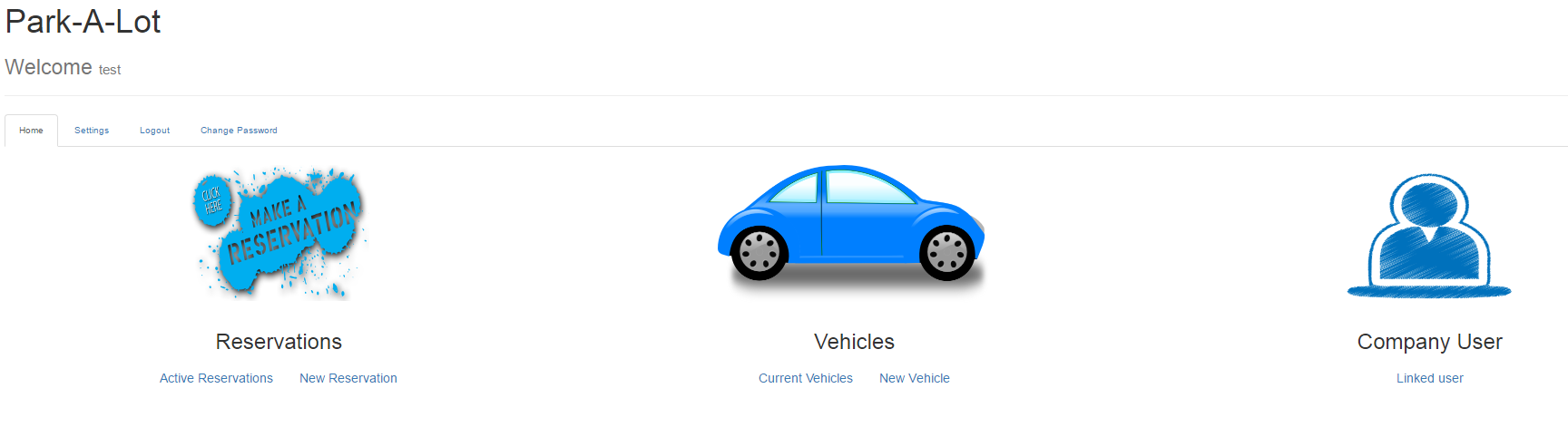
Personal Profile page: This webpage will allow user to access other utility page. The user’s first name will be shown on the up-left corner and the company’s name will be shown in the bracket(if they have).

****

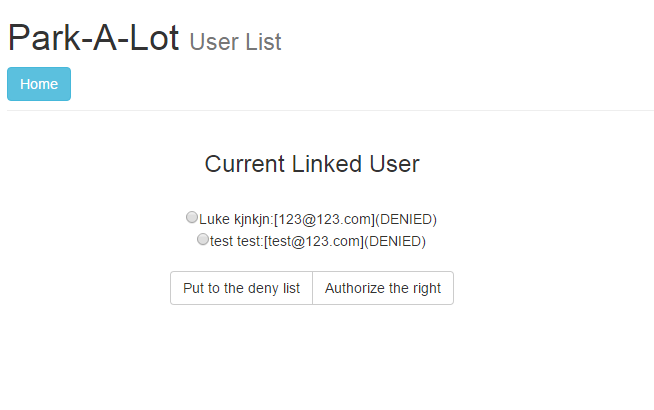
Vehicle Editting page: This webpage allows user to edit their vehicle information. If there is no vehicles links to user account, it will show “You haven’t added any cars yet!” They can use right side of the webpage to add vehicle to the database. And the left side will show the vehicle information automatically. And if user sets one of them as a default car, there will be a “default” mark which locates at the right of the vehicle information and this will be the default car when they make the reservation. The payment page is quite similar as the vehicle one, so i don’t show it here.

****

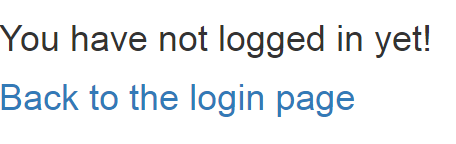
Business profile page: This webpage will help business account to go to other utility pages. And its structure is quite same as the personal one’s, besides the user list. The vehicle page is quite same as the personal one, so i won’t show it here.

****

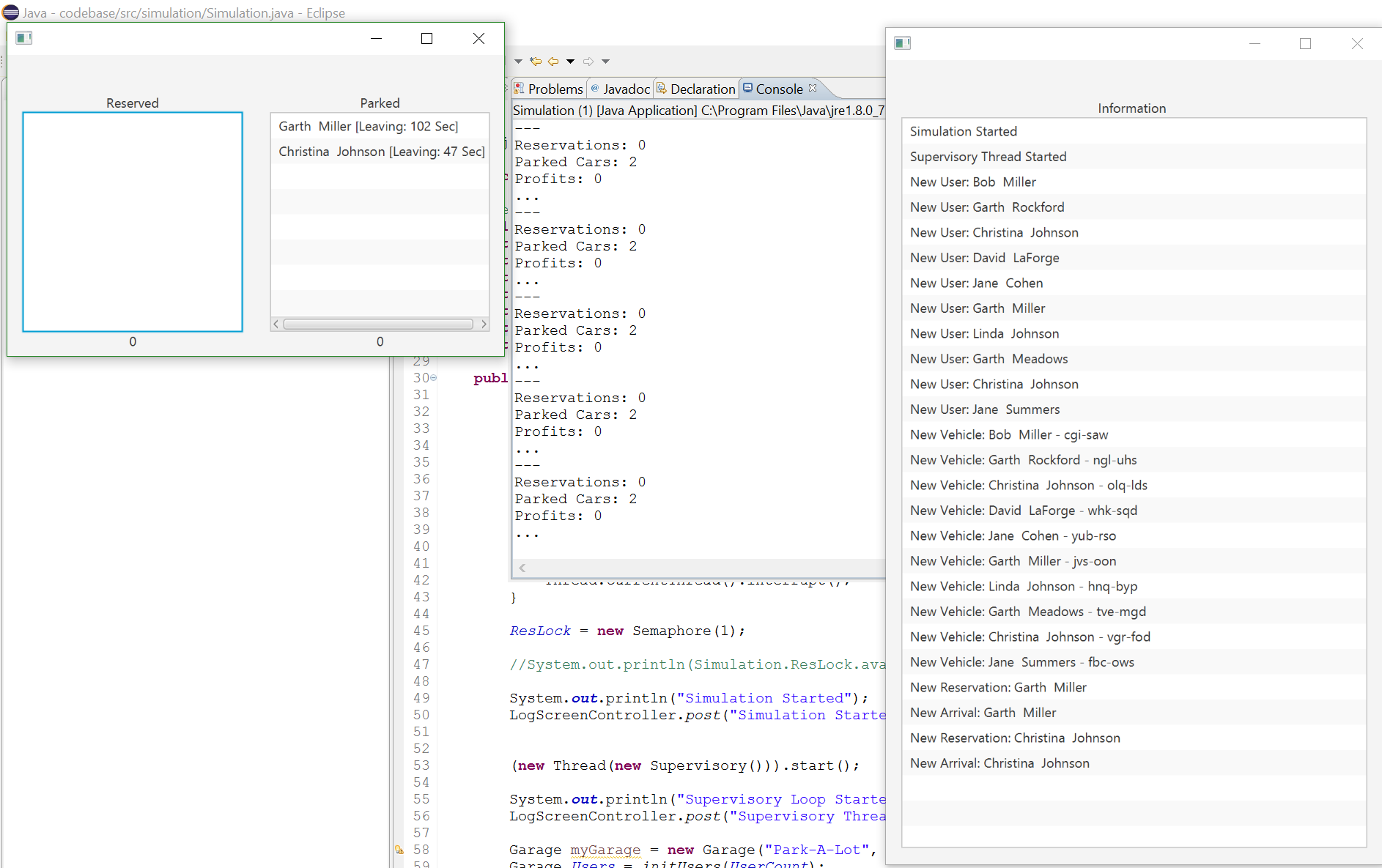
Business account user list: The business account could see the personal accounts which links to them. If business account authorizes the right to the user, the user will see the business account vehicle in their personal vehicle list and they can use that car to make the reservation. And i plan to implement the occupied status in the database table. So it can avoid the race conditions which multiple users use the same vehicle in the same time.

****

Error page: If user try to access the profile page, vehicle or payment page without login. The mechanism i implement(check if session variable is null) into those pages will direct user to this page and remind them to login.

****

**Software Simulation:**



The simulation user interface was designed in a manner that allows the operator to gain immediate access to relevant system information. The simulation interface is broken into 3 windows: Reservation Status, Garage Log and Console Window. The Reservation Status window includes two boxes that house reservations of a particular status. The first box, Reservations, will display all active reservations that have not arrived and the number of seconds until they arrive at the garage. This will update dynamically for late arrivals. The second box shows vehicles that are parked within the garage the number of seconds until the depart the garage. This will update dynamically for overstay reservations. The second window, Garage Log is a status log that display all important garage events such as arrival, departures and profits. The Console Window is the background log that display all unparsed simulation information. This will display every happening in the garage at all times. A combination of these three windows, provides the user with a high level understanding of the garage status at all times.

**Policy Decisions:**

The manager portal interface was designed strictly for the manager to use and no one else. It contained the login page where the manager would specify his login credentials. This would take him/her to the webpage with different links for the specified rates, specials, policies and hours. The manager could then click on any of the links which would allow him to add/delete new or specified rates, policies, specials, and hours. The portal is user friendly, which allowed easy access and allows the manager to easily edit any of the specified items.

**Statistics:**

The Statistics function of this Automated garage system will accessible via a java app linked with a MySQL database, downloadable from the website. It will only be accessible through a manager’s account, both as website download, and on the app’s login. The charts will be created using JavaFX.

**Arrival:**

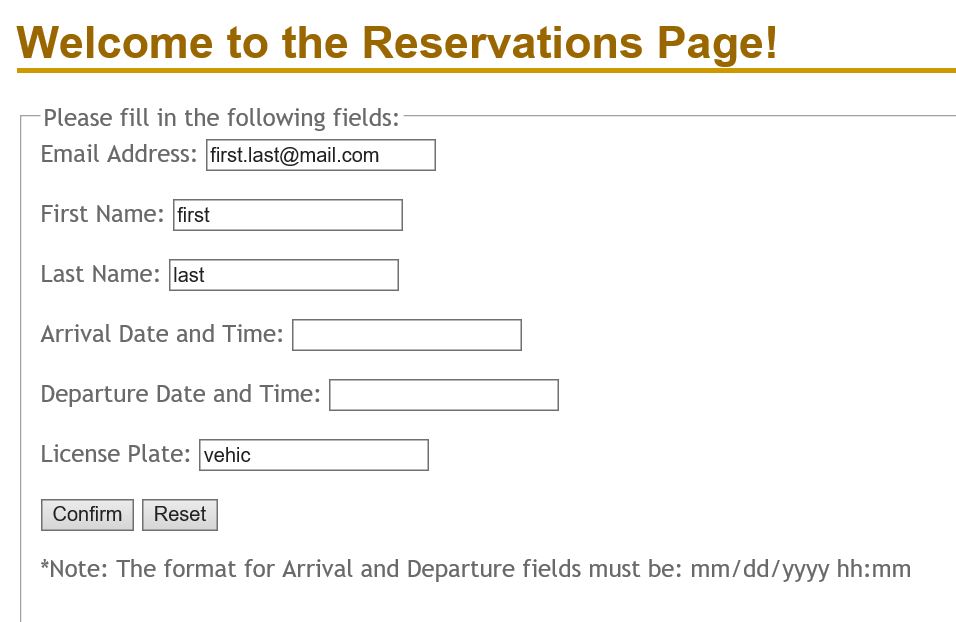
User interface implantation actually has changed, although a device pad hooked up to a monitor, with a number-pad, keyboard, and possible other security measures is needed, rather than having to create a GUI for people who want to do a walk in reservation, we could simply reuse the code written in Reservation for this, by having a console run an application(however the walk in reservations is more of a business decision, depending on the manager). Therefore, minimal interface is really needed for Arrival as we want to automate the process as much as possible, after a customer books a reservation, and arrives at the parking garage

**Departure**:

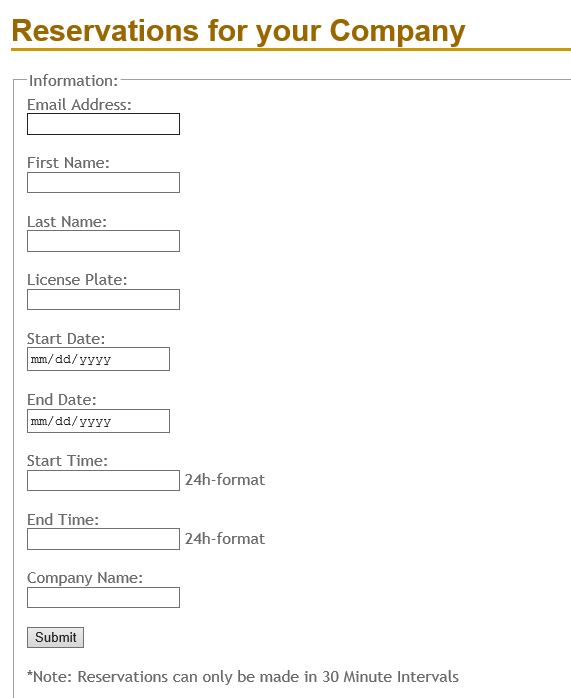
There will not be too many interfaces interact with customer in this part. While the occupancy sensor detects a vehicle leaving, the departure resolver will automatically access the database to fetch the customer information and payment method. After that, customer will receive a confirmation notification of their payment detail.

**Reservations:**

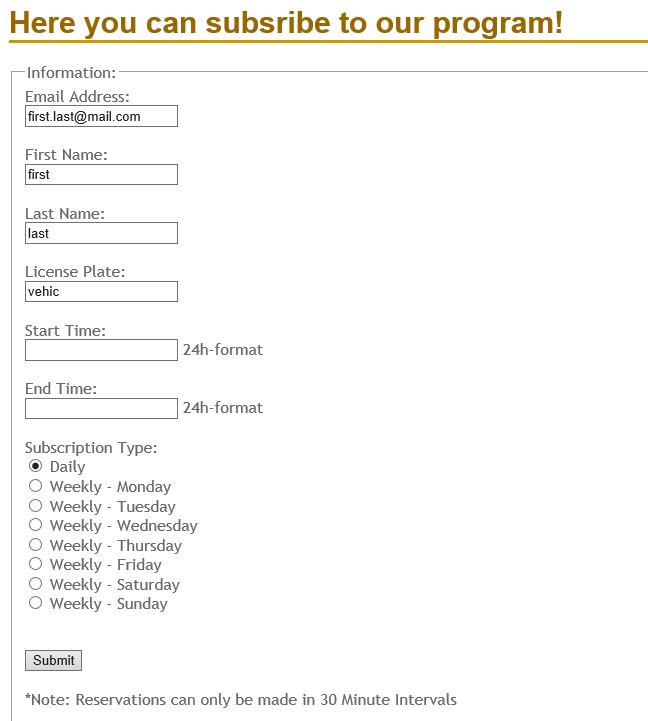
The the User Interface of the system has a few sections dealing with reservations. Mainly these are a part of the website where customers can make reservations. Namely, they are the Normal Reservations page, the Subscribing Reservations page, and the Business Reservations page. A cancellations page is also included. These are all very user-friendly as all the customer has to do is go to the website, log in with their credentials and go the reservations tab. Once here they can make regular reservations, those associated with their company, or subscribe to our program simply by entering the fields in the form they link to and then clicking on a submit button. All the UI designs related to the reservations system are shown below.



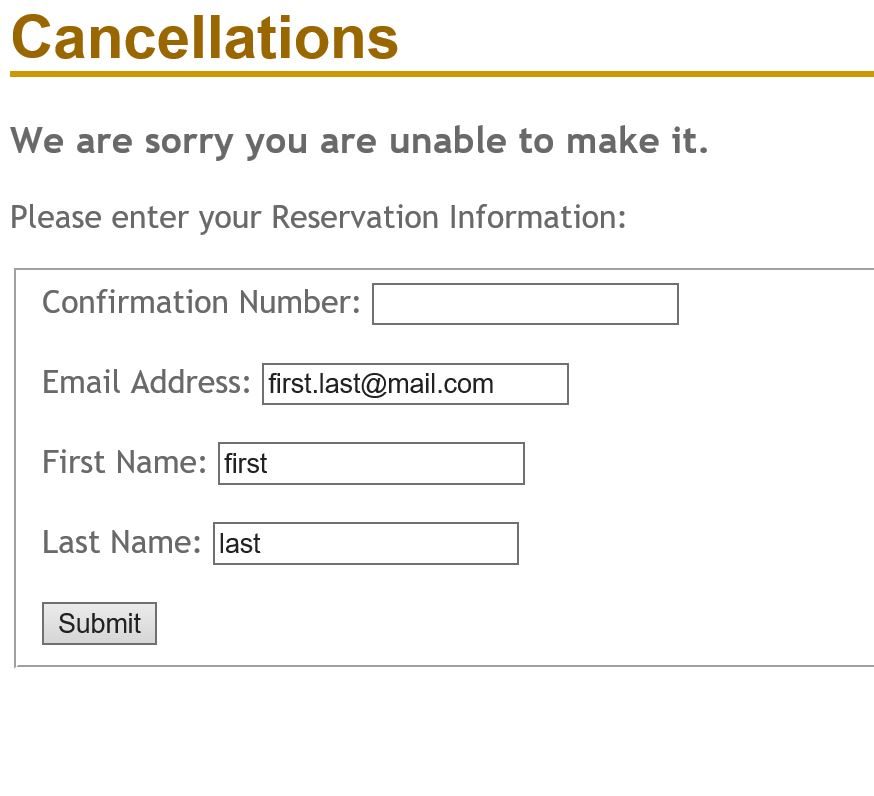
This is the UI section for the regular reservations.



This the UI section for the business reservations.



This is the UI section for the subscribing reservations.



This is the UI section for cancelling reservations.

As you can see, in all the pages the fields unique to the user are pre-filled to ensure added security and proper functionality and reliability.

Design of Tests

**Software Simulation:**

The simulation was designed with the purpose of integration testing in mind. As the simulation runs, many different components of the website and Park-A-Lot system are invoked dynamically in order to demonstrate functionality of a system. For every software object generated within the simulation, the *AccessWebsite* class is used to automate the Park-A-Lot website which adds appropriate elements to the database. When the next object is generated, it often relies on the successful production of the previous item. This can be seen in the case of Users, Vehicles and Reservations. In order to automate the addition of a new vehicle to a user’s account, each user account must have been successfully created. The same applies to reservations. A new reservation creation will fail if either the user or vehicle creation did not complete successfully. For this reason, the simulation serves as integration testing for the rest of the system.

Unit testing of the simulation was performed on a class by class basis. Each class was populated with an appropriate number of print statements according to expected data structure values are particular point during execution. The simulation was run repeatedly while observing these print statements for nominal behavior. For iteration of execution, important simulation parameters were modified that tested the elasticity of the software in off-nominal situations. Only when a single class had been put through its paces, would an interface be written that allowed it communicate with other classes. This interface was then tested as well to verify that information spread between multiple classes remained intact.

An example of this testing paradigm is the simulation User class. Users are randomly generated at program execution and assigned personal information. For this class, each function was tested individually and the pertaining personal information was closely monitored to verify inaccuracies. Once it was guaranteed that a single user could be generated without issue and placed in the appropriate location, the next testing phase involved generating many more users to verify that the system was able to distinguish between individual objects.

**Statistics:**

1. Test Cases
   1. Can the program identify the best and worst hour, in terms of profit, on a given day
   2. Can the program identify the best and worst days, in terms of profit, a given week(s)/month(s)/year
   3. Can the program identify the best and worst hour, in terms of number of reservations, on a given day
   4. Can the program identify the best and worst days, in terms of number of reservations, a given week(s)/month(s)/year
2. Test Coverage: The above test cases will cover all code within this portion of this section of the Garage Automation System.

**Arrival:**

For arrival, the testing was broken up into 2 parts, the base architecture testing and the added features testing, here is the base architecture testing:

Testing the software will be composed of 3 main sets of tests. First set will involve, cars that have registration and booked a reservation for this time x, and date t. Second set will involve cars that and need to enter a confirmation number or show a QR scan, for time x and date t. 3rd will involve cars that need to go to walk in, hence all free walk in spots need to be properly displayed. 4th will be registered members who are attempting walk-in reservation, and are indeed registered members(either license plate or membership number). 5th will be same as 4th except, they aren't registered. Each of these sets, will involve testing to see expected vs actual, this will be done through 10-20 trials to properly stress test the algorithms, and data structures. For the integrated testing, without question, departure, reservations, and arrival would have to work together and be closely integrated due to the massive overlaps.

Added Features:

For the next set of design tests after the first demo, we need to apply the same rhetoric as was applied for the first set of design tests, except now for customers who are recurring, and those who make or don't make the grace period. The first set of tests are designed to see if customers who are recurring are guaranteed a spot until the grace period is up. The second set of tests are designed to see if customers who miss the grace period get a spot, if spots are open, or don't. The third set of tests are designed to block customers who miss the grace period, from affecting other customers who don't

**Departure:**

(a) Test if the customer will be charged automatically as soon as leaving the garage

(b)Test if a vehicle leave the garage, the database give out proper customer information

(c) Test if understay will be fully charged, overstay will be charged more due to parking policy

(d) Test if there will be a confirmation email sent as soon as possible the customer leaves the garage.

(e) Test if the history log has been correctly stored in the database

**AccountManagement:**

Since my part is mainly about webpage design and implementing functions in the webpages. I didn't write the code to test my part. But i write a list which is the functions my website could realize.  
  
1. User can't access the profile.php, vehicleInfo.php and PaymentMethod.php without login.  
  
2. User could create an account under the registration.html. Since i use emailaddress as a primary key under the logininfo and user, so they can't create another account by using the same email address.  
  
3. User could login by providing the password and username which created before. And website will automatically direct to the profile.  
  
4. I use session to recognize the user identity.  
  
5. Under payment method and vehicle page, user could add multiple cars and payment methods and set them as default or delete them.  
  
6. Password reset page could help user to reset their their password when they provide correct old password.  
  
7. If user doesn't provide the information asked, the website will inform them and let them fill in.

8.Login page will identify the account type automatically and direct user to different profile page.

9.Personal account could link to a company account by the link password and the company name will be reflected in the upper-left corner of user profile page.

10. If business account authorize the right to the account which links to them, the personal account will see the company vehicle in their own vehicle list.

**Reservations:**

During the process of creating the Reservations subsystem, many testing cases were performed to ensure proper functionality and reliability.

After the user logs in, test to see if they can access the Reservations tab. Once accessible they should have three options displayed:

Normal Reservations, Subscribing Reservations, and if a Business Account,

Business Reservations

The user should be able to access the normal reservations page.

Testing was done to make sure that fields unique to the customer such as their name and email address were pre-filled so they can only make a reservation for themselves. In addition for business reservations, the company name field should be pre-filled.

Testing to make sure a user can only make a reservation for themselves, and they had to log in with proper credentials to access the reservations page.

In addition a few test cases were performed to ensure users can only enter valid text in the fields and submit the form and make sure the form could not be submitted for processing if invalid entries were made in the forms.

Once submitted, testing to check that a successful entry in the database was entered and properly formatted.

A testing case was also performed for the confirmation number and made sure that it was displayed properly with proper formatting based on the type of reservation made, for example, with no letter, an R, or a B, for normal, recurring, and business reservations respectively.

Lastly, testing was done for cancellations so that once the reservation confirmation number is inputted, the proper entry was deleted from the database.

Account Management

**Algorithms:** There is no mathematical model included in this part.

**Data Structures:** The user information is stored in an Arraylist. The payment method and vehicle information are child classes of user information and they are saved in Arraylists as well.

Arrivals

One of the data structures used will be an array used to keep track of the state of every parking spot, specifically a bitmap vector. Each index in the bitmap, represents a spot in use or not in use. This is done for several reasons, mainly simplicity, easy reconstruction, and ease of use for the programmer. The biggest benefit is bitmap is an extremely efficient way to save space, not to mention it can be further broken up. However, if a garage is extremely large, then even the bitmap will fail however most garages are under 2000-3000 spots so overall, most garages this can work without further updates from programmers. Specifically, the algorithm assigned with this structure will be a quick and robust searching algorithm. It is linear search, and despite its efficiency; the number of times a spot will need to be found will always be at most few times per minute. Also, the algorithm is easy to code and will be less prone to bugs, errors, etc.

Garage Policies and Scheduling

**Algorithms:** There are no mathematical models in this section

**Data Structures:** In this section the manager will have access to the website and a database with all of the garage policies and scheduling information. This information will be stored in an Arraylist and the manager will be able to change the information.

Departure

**Algorithms:** Since the mathematical model of arrival for this system is a poisson distribution as mentioned in Report one. However, I am not apply and mathematical model for departure since we suppose that most of the customer will leave on time, that is to say , the schedule for departure is almost fixed by reservation. So we are not using any other algorithms for departure.

**Data Structures:** the information will be stored Arraylist since all previous part applied arraylist to store information and there will be not other complicated structure.

Statistics

**Algorithms:** The Reservation Statistics portion of the Parking Lot Automation system does not implement mathematical models from the Mathematical Model. Rather, the algorithms that govern are based on statistical analysis; requesting information from the garage database and producing models based on that information. At the user’s request, this system will generate a table, or graph displaying information including; the number of reservations, reservation frequency, concentration, account type (personal or business), and or length in any given time frame. The feature will also feature capability to determine most and least profitable day, and or the busiest and least busy days in a given time frame. The system will also be able to display statistical information regarding revenue from reservations, including; profits within a given time frame, and money from personal accounts and or business accounts, or overstay charges.

**Data Structures:** This Reservation Statistic portion of the Parking Lot Automation will organize relevant information into an ArrayList, which will be used in creating graphs.

Reservations

**Algorithms**: The part of the system that deals with reservations does not have anything to do with the mathematical models from report 1. Also, it does not use any complex algorithms. It will mainly extract information the user puts in the website and test and place it in another data structure alongside information from databases. The methods are as follows:

Normal Reservations - Based on the user’s login, the name will be extracted and placed in an array. Once the user clicks on the make reservations link, a schedule with empty and filled slots which will be held in a container will be displayed. Once the blocks are selected, that portion of the array’s contents will be extracted and made reserved, the block will be filled in. This will also be used to display the user’s reservation. His payment method will be extracted from the payment database and displayed alongside this. From the account database, if a business account is detected he will be discounted based on management discretion, a variable. A confirmation code of 6 numbers will be randomly generated using a random number generator (like the mersenne twister) and checked against a list of already existing numbers placed in a list. If it exists a new one will be generated until it is not repeated and then outputted to the user online. A QR code will be generated using a generator in the same manner.

Cancellations - Once the user logs into their account and clicks cancel reservation, the reservations made will be in a list and extracted for output to the user. The user can only select from these reservations to cancel. Once selected they may be charged a cancellation fee based on management discretion, a variable or fully refunded. The amount, calculated and extracted from the payment database will be displayed as a refund minus any cancellation fees.

Subscriptions - A subscription field will also display a similar schedule of blocks stored in an array as did the normal reservations part. Once the user clicks make a subscription they will be greeted to this. A list of frequencies (daily, weekly, or monthly) will be displayed. For these subscribers, one slot in the list of available slots will be filled in for the time available. The user will be displayed a monthly, weekly, or daily charge extracted from the payment database based on his selection. This depends on the management database as well and will be discounted if a business account is detected by an account type variable.

**Data Structures**: We will extract this information from the fields and place them in a *NESTED LIST*. This is useful because to make a reservations part of the data needed will require the above three variables. With a nested list, we can have the a list of a sets of lists which include these three variables. PHP has limited data structures available, namely arrays and lists, and I believe a nested list is the best way to go. SQL will be used as it goes hand in hand with PHP to get information from databases as needed.

**History of Work:**

The project progressed as meet on a weekly basis after class. The basis for the entire project, what coding languages we would use, as well as what database solutions, was decided early in the semester. Milestones and deadlines mostly revolved around when reports were due, until tangible work began several weeks before the first demo. Over the course of the semester we mostly worked individually, as we had modularized our project into individual processes for each of us. During our weekly meetings we would discuss our process, hard deadlines for code were not set within the group until we started integrated our code before spring break.

After the first demo, our process became much less modularized as integration became a bigger focus. Individual progress needed to translate into results frequently as tests on the whole system were made.The overall process was within the bounds of our expectations for the project; to work individually, and come together towards the end.

Key Accomplishments:

* Multi threaded concurrent data modification
* Proper database communication
* Formatting database information for specific functions
* Reacting to user arrival and departure
* Unifying the system through the database
* Implementing policy decisions in real time

**Current Status & Future Work:**

The project is nearly complete and requires several modifications before the submission of the final project archive. The simulation has been mostly completed but will need to interface with any additional features added to the website. The statistics function for the system still needs to be completed. Some system requirements were marked as future work to show that we can always add in these new functions to improve the project and website. Many of these are being worked on currently.

**References:**

"1. Project Description." *Software Engineering ProjectParking Garage/Lot*. Web. 01 May 2016.

"Book: Software Engineering - Textbook by Ivan Marsic." *Book: Software Engineering - Textbook by Ivan Marsic*. Web. 01 May 2016.

Chemuturi, Murali. *Software Estimation Best Practices, Tools & Techniques: A Complete Guide for Software Project Estimators*. Fort Lauderdale, FL: J. Ross Pub., 2009. Print.

**Project Management:**

Breakdown of Responsibility

|  |  |  |
| --- | --- | --- |
| Group member name | Responsibility | Content |
| Brandon Dunlap | Statistics | Manager statistics model |
| Yufeng Liu | User account management | Website for creating and managing user accounts. The interaction between business accounts and personal ones. |
| Luke Miller | Simulation, Integration Testing | Dynamic software simulation to test individual components of the parking garage functioning together |
| Harshil Patel | Reservations System | Web development dealing with the creation and management of different types of reservations |
| Thomas Walters | Policy management | Manager portal to add new/delete any policies, rates, hours, and specials |
| Xiang Xing | Vehicle Departure | Reading license plates and calculating cost |
| Vikram Krishna | Vehicle Arrival | Vehicle arrival and assigning spots |

Although we have completed our final demonstration, we plan on polishing the functionality of the system before the submission of the final electronic archive. There are several additional features that we would like to integrate and we are going to setup the system to run on a live external server to be accessed to through a public domain.

The future work section discusses these improvements.