System and Software Architecture Description (SSAD)

SnapValet

Team 03

Name	Role
Brian Vanover	**Project Manager, Feasibility Analyst, Life Cycle Planner.**
Molly Karcher	**IIV & V, Quality Focal Point, Prototyper/Builder.**
Ditong Dong	**System Architect, Prototyper/Builder.**
Ridhima Manjrekar	**Requirements Engineer, Prototyper/Builder.**
Brian Bousman	**IIV&V, Quality Focal Point, Prototyper/Builder**
Patrick Horng	**Feasibility Analyst, Prototyper/Builder**

04/19/2015

Version History

Date	Author	Version	Changes made	Rationale
10/11/2014	Ditong Ding	1.0	• Original for CSCI577 project SnapValet	• Initial draft for use with SnapValet v1.0
10/13/2014	Ditong Ding	1.1	• Change some terms used in system	• For using same words in system document
10/19/2014	Ditong Ding	1.2	• Change all diagrams with Visual Paradigm • Reconstruct use-case diagram	• To update the system design.
11/26/2014	Ditong Ding	2.0	• Complete all sections of SSAD • Modifier use case	• To finish the document • To update the system design
12/04/2014	Ditong Ding	2.1	• Update web server used	• Decided by team
12/07/2014	Ditong Ding	2.2	• Add more technology specific in technology-specific system design section	• Apply feedback of 2nd ARB
02/11/2015	Ditong Ding	2.3	• Make some changes for RDCP, and remove old team members	• For RDCP
02/16/2015	Ditong Ding	2.4	• Redraw sequence diagram • Add ER diagram	• Apply suggestion from RDCR ARB
04/19/2015	Ditong Ding	3.0	• Update ER diagram • Update description of some use cases	• Apply changes from TRR ARB

Table of Contents

Table of Tables

Table of Figures

1. Introduction

1.1 Purpose of the SSAD

The purpose of the SSAD is to record the results of SnapValet system analysis and design. This document is used by developers as reference to the system architecture, and the development of application must follow the statement in the SSAD. Furthermore, the SSAD is used to help the maintainer and clients to understand the architecture of the system once the application is delivered.

1.2 Status of the SSAD

The status of the SSAD is currently at the Foundations phase version number 2.2. This is the version for DCP.

2. System Analysis

2.1 System Analysis Overview

The primary purpose of SnapValet is to improve the experience of valet service. By providing cashless payment, car request feature, SnapValet will highly improve the speed of valet process, give customers more choices when they want to use valet service. For valet and valet company manager, the system also provides some features in employee management and location management, for supporting the cashless valet service.

2.1.1 System Context

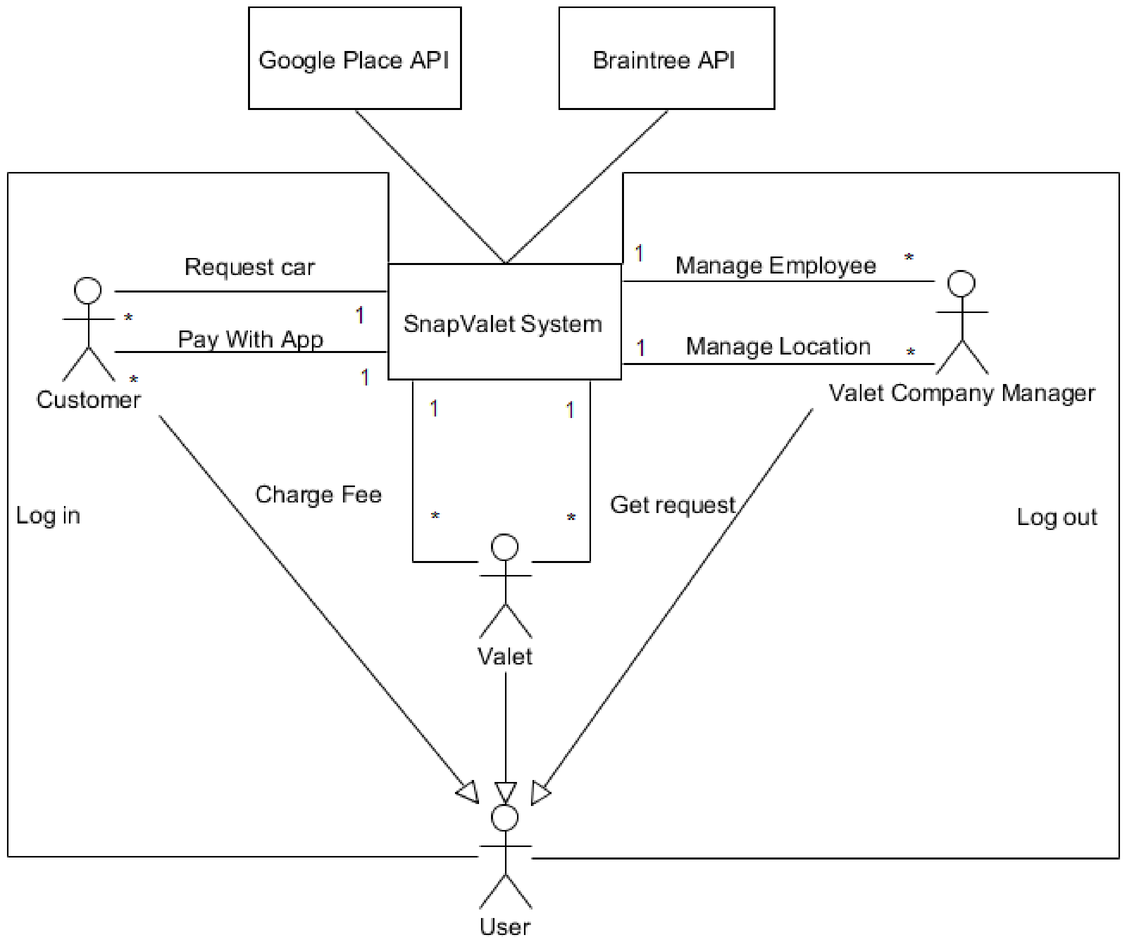


Figure 1: System Context Diagram

Table 1: Actors Summary

Actor	Description	Responsibilities

Actor	Description	Responsibilities
User	People who use SnapValet System	● Connect with system by log in/log out action.
Customer	People who use valet service and use SnapValet app to notify or pay valet service	● Request for valet service ● Request to retrieve cars ● Pay valet service fee and tip
Valet	People who provide valet service	● Provide valet service ● Manage keys and cars ● Charge valet service fee
Valet Company Manager	People which in charge of valet operators	● Manage valet location and service fee ● Manage employee

2.1.2 Artifacts & Information

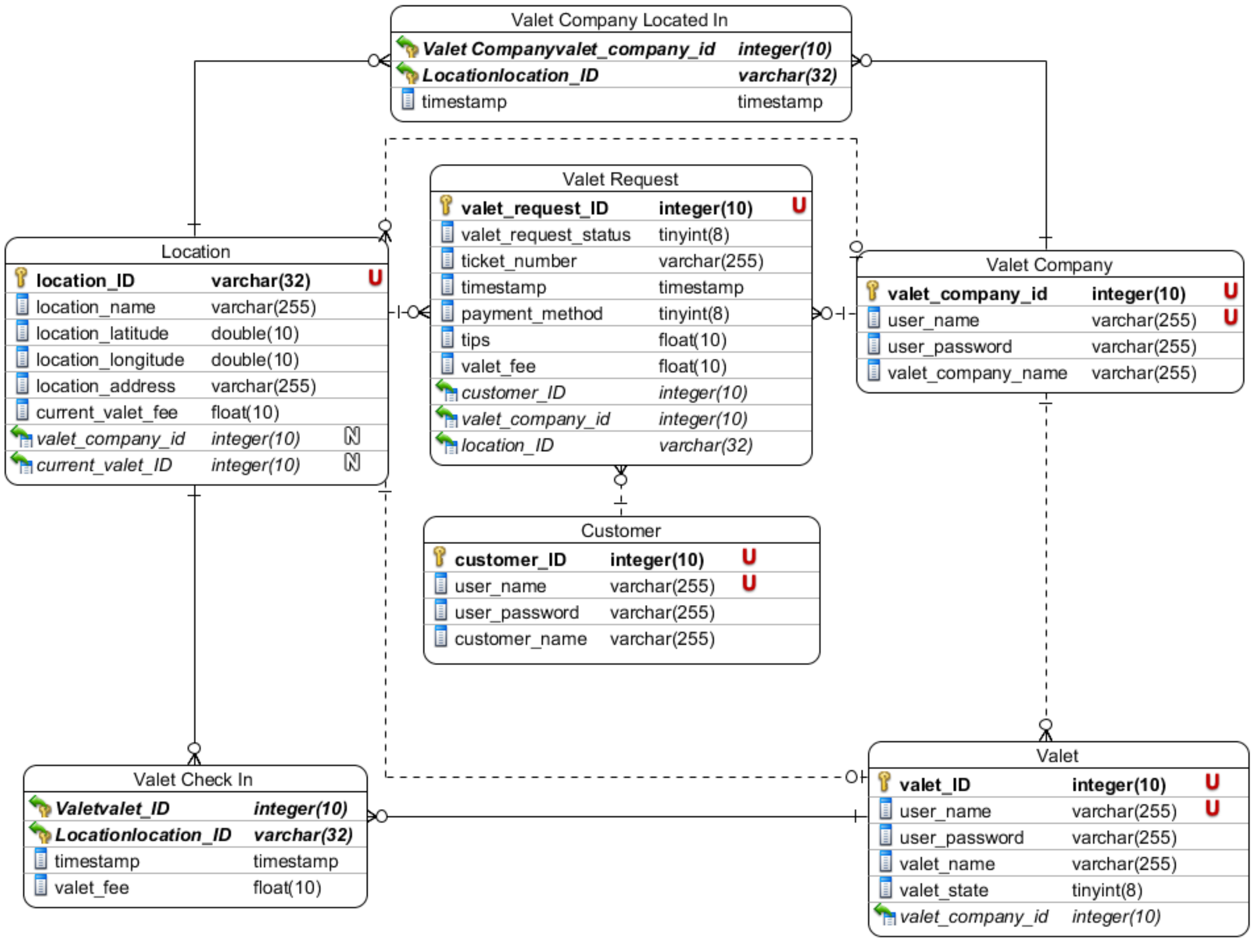

Valet Company Located In
Valet Companyvalet_company_id integer(10)
Locationlocation_ID varchar(32)
timestamp timestamp
Valet Request
valet_request_ID integer(10) U
valet_request_status tinyint(8)
ticket_number varchar(255)
timestamp timestamp
payment_method tinyint(8)
tips float(10)
valet_fee float(10)
customer_ID integer(10)
valet_company_id integer(10)
location_ID varchar(32)
Location
location_ID varchar(32) U
location_name varchar(255)
location_latitude double(10)
location_longitude double(10)
location_address varchar(255)
current_valet_fee float(10)
valet_company_id integer(10) N
current_valet_ID integer(10) N
Valet Company
valet_company_id integer(10) U
user_name varchar(255) U
user_password varchar(255)
valet_company_name varchar(255)
Customer
customer_ID integer(10) U
user_name varchar(255) U
user_password varchar(255)
customer_name varchar(255)
Valet Check In
Valetvalet_ID integer(10)
Locationlocation_ID varchar(32)
timestamp timestamp
valet_fee float(10)
Valet
valet_ID integer(10) U
user_name varchar(255) U
user_password varchar(255)
valet_name varchar(255)
valet_state tinyint(8)
valet_company_id integer(10)

Figure 2: Artifacts and Information Diagram

Table 2: Artifacts and Information Summary

Artifact	Purpose
Valet Company	Contains information of a valet company for cashless valet service.
Valet	Contains information of a valet for cashless valet service.
Customer	Contains information of a customer for valet service.
Valet Request	Contains information of a car retrieval request and payment information from customer.
Location	Contains information of the establishments which provide SnapValet service.
Valet Company Located In	Contains the “belongs to” connection between valet company and location
Valet Check in	Contains the “working for” connection between valet and location

2.1.3 Behavior

Figure 3 illustrates the process diagram of SnapValet. It can be divided into four capabilities: valet service, employee management service, location management service and account service.

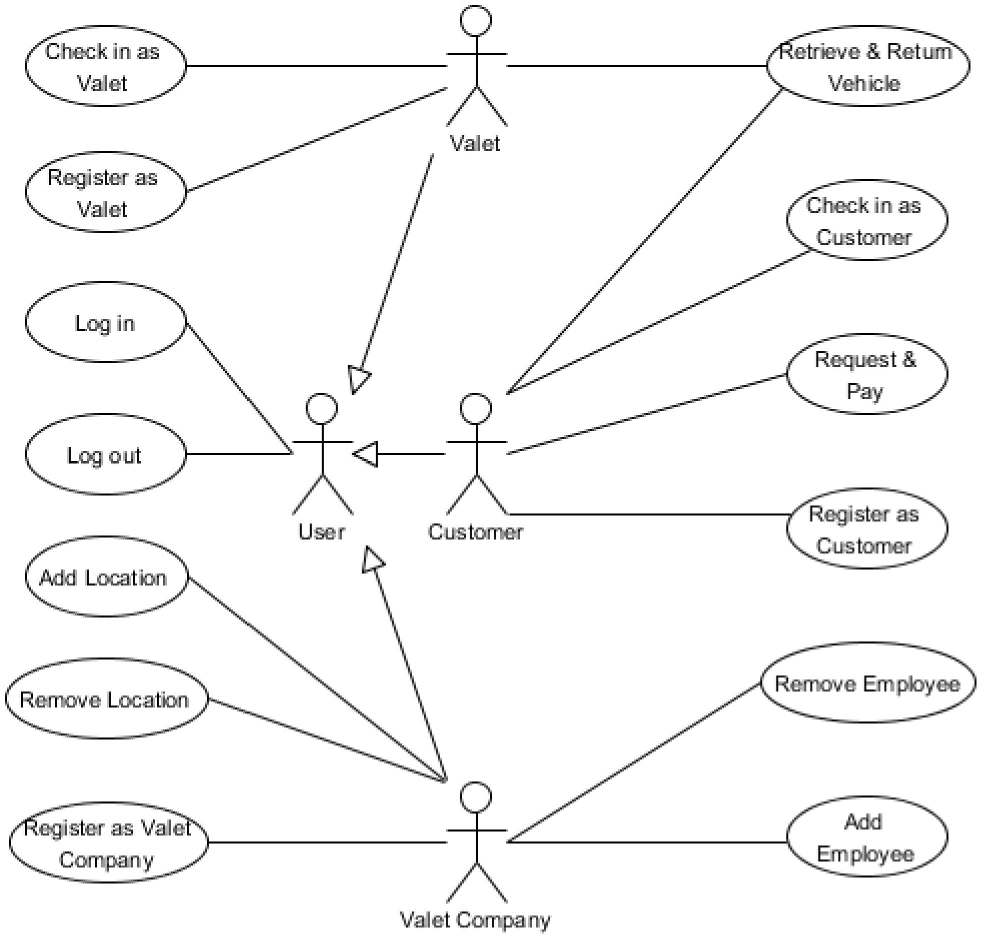


Figure 3: Process Diagram

2.1.3.1 Valet Service

2.1.3.1.1 Check in as Valet

Table 3: Process Description (Check in as Valet)

Identifier	UC-1: Check in as Valet
Purpose	Establishes a temporary one to one relationship between a valet and, effectively, a valet company and a particular venue for a shift
Requirements	WC_3392, WC_3390, WC_3384
Development Risks	Account security problem, device location service problem
Pre-conditions	The valet has logged in with GPS enabled and sees the checkin

	map which displays markers according to venues within a certain distance that have valet services provided by the valet's company. The valet must be within a certain distance of the desired venue to check in to it. Lastly, the valet must be an active employee in the SnapValet database
Post-conditions	The valet is checked in to the selected venue and sees the queue screen

Table 4: Typical Course of Action (Check in as Valet)

Seq#	Actor's Action	System's Response
1	The valet selects the venue according to the location of his/her shift.	
2		The system validates that the venue is not already checked into by another valet and displays a confirmation request.
3	The valet confirms that this is the correct location with valet fee and checks in.	
4		The system navigates to the queue screen.

Table 5: Alternate Course of Action (Check in as Valet)

Seq#	Actor's Action	System's Response
1	The valet selects the venue according to the location of his/her shift.	
2		The system finds that the venue is already checked into by another valet, then send alert back to valet shows that they are unable to check in to this venue because it is being serviced by another valet.

Table 6: Alternate Course of Action (Check in as Valet)

Seq#	Actor's Action	System's Response
1	The valet selects the venue according to the location of his/her shift.	
2		The system validates that the venue is not already checked into by another valet and displays a confirmation

		request.
3	The valet does NOT confirm that the selected venue.	
4		The system returns the valet to the checkin map to select the correct venue.

Table 7: Exceptional Course of Action (Check in as Valet)

Seq#	Actor's Action	System's Response
1	The valet selects the venue according to the location of his/her shift.	
2		The system finds that the valet is not an active employee in the database, then the system will alert the valet, that his/her status is inactive and prevent location checkin.

Table 8: Exceptional Course of Action (Check in as Valet)

Seq#	Actor's Action	System's Response
1	The valet selects the venue according to the location of his/her shift.	
2		Because of server error during checkin submission, the system will alert the valet that something is wrong and to try again later.

2.1.3.1.2 Retrieve & Return Vehicle

Table 9: Process Description (Retrieve & Return Vehicle)

Identifier	UC-4: Retrieve & Return Vehicle
Purpose	The process of retrieving a vehicle from the queue and returning it to the customer.
Requirements	WC_3390, WC_3386, WC_3384
Development Risks	Synchronization problem
Pre-conditions	The customer has requested his/her vehicle via ticket number form submission and the request is displayed in the valet queue
Post-conditions	The requests is no longer in the queue. The customer is closed out of the application and has left the venue. The valet sees the request queue.

Table 10: Typical Course of Action (Retrieve & Return Vehicle)

Seq#	Actor's Action	System's Response
1	The valet selects an unconfirmed request from the queue.	
2		The system displays a dropdown menu containing options: Retrieve and Invalid.
3	The valet confirmed the request by selecting retrieve.	
4		The system processes the previously recorded payment, if it exists, and notifies the customer that his/her vehicle is being retrieved. The system changes the request state to confirmed and paid (showed with green color) if payment is successful or to confirmed (showed with yellow color) if still awaiting payment via cash.
5	The valet retrieves the vehicle and the customer exits the venue. The valet then remove the request in the queue to close out the request.	
6		If mobile payment was processed successfully, the system displays a close out confirmation.
7	The valet confirms the closeout and hands the keys to the customer.	
8		The system returns to the request queue in the valet application. The system also sends an email to the customer recording the transaction. If the email fails, the system will log the failure in the database. The application then closes on the customer application.
9	The customer leaves the venue in his/her car	

Table 11: Exceptional Course of Action (Retrieve & Return Vehicle)

Seq#	Actor's Action	System's Response
1	The valet selects an unconfirmed request (showed with red color) from the queue.	

2		The system displays a dropdown menu containing options: Retrieve and Invalid.
3	The valet invalid this request as there the ticket number is not associated with any existing car key.	
4		The system generate a notification to the requesting customer informing them of the error and prompting them for a valid ticket number. If the customer opens the notification, the system will direct the customer to the request screen for him/her to redo request.

Table 12: Exceptional Course of Action (Retrieve & Return Vehicle)

Seq#	Actor's Action	System's Response
1	The valet selects an unconfirmed request (showed with red color) from the queue.	
2		The system displays a dropdown menu containing options: Retrieve and Invalid.
3	The valet confirmed the request by selecting retrieve.	
4		The system processes the previously recorded payment, and find the transaction fails. Then the system will send a notification to the customer that his/her transaction has failed. When the customer opens this notification, they are directed to the pay screen. The request state is kept "confirmed" until the mobile payment succeed.

2.1.3.1.3 Check in as Customer

Table 13: Process Description (Check in as Customer)

Identifier	UC-5: Check in as Customer
Purpose	Establishes a temporary one to one relationship between a customer and a particular venue.
Requirements	WC_3392, WC_3390, WC_3384, WC_3210
Development	Account security problem, device location service problem

Risks	
Pre-conditions	The customer has logged in with GPS enabled and sees the checkin map which displays markers according to venues within a certain distance that have valet services provided by different valet companies using SnapValet and which have a valet checked in at that location. The customer must be within a certain distance of the venue he/she desires to check in to.
Post-conditions	The customer is checked in to the selected venue and sees the vehicle request screen.

Table 14: Typical Course of Action (Check in as Customer)

Seq#	Actor's Action	System's Response
1	The customer selects a venue according to his/her location.	
2		The system displays a confirmation prompt for checking into this location.
3	The customer confirms the checkin.	
4		The system navigates the customer to the vehicle request screen and push an advertisement about a random venue that provides SnapValet service.

Table 15: Alternate Course of Action (Check in as Customer)

Seq#	Actor's Action	System's Response
1	The customer selects a venue according to his/her location.	
2		The system displays a confirmation prompt for checking into this location
3	The customer does NOT confirm that the selected venue.	
4		The system returns the customer to the checkin map to select the correct venue.

Table 16: Exceptional Course of Action (Check in as Customer)

Seq#	Actor's Action	System's Response
1	The customer selects the venue according to the location of his/her shift.	
2		Because of server error during checkin submission, the system will alert the customer that something is wrong and to try again later.

2.1.3.1.4 Request & Pay

Table 17: Process Description (Request & Pay)

Identifier	UC-6: Request & Pay
Purpose	The customer requests his/her vehicle be retrieved and handles payment.
Requirements	WC_3392, WC_3215, WC_3205
Development Risks	Synchronization problem
Pre-conditions	The customer has logged into SnapValet and has checked into a venue. Customer is on the ticket request screen.
Post-conditions	The request and payment have been recorded. The queue has been updated and the customer sees the request submission confirmation.

Table 18: Typical Course of Action (Request & Pay)

Seq#	Actor's Action	System's Response
1	The customer inputs ticket number and submits form.	
2		The system validates the form then navigates the customer to the payment page which displays the option to pay via mobile or by cash with the default set to mobile.
3	The customer specifies the tip then submits the form.	
4		The system displays a spinner while it RECORDS the payment and updates the request queue of the valet checked into the same venue as the customer. The request is in unconfirmed state (showed with red color) in the queue. The system then alerts the customer of the successful form submission.

Table 19: Exceptional Course of Action (Request & Pay)

Seq#	Actor's Action	System's Response
1	The customer leaves ticket number blank and submits form.	
2		The system will alert the customer that

		a ticket number must be specified and prevents form submission.

Table 20: Exceptional Course of Action (Request & Pay)

Seq#	Actor's Action	System's Response
1	The customer inputs ticket number and submits form.	
2		The system alert the customer that something went wrong when there is a server error during form submission.

2.1.3.2 Employee Management Service

2.1.3.2.1 Remove Employee

Table 21: Process Description (Remove Employee)

Identifier	UC-11: Remove Employee
Purpose	The functionality to remove valet – valet company associations
Requirements	WC_3387
Development Risks	Account security problem
Pre-conditions	The valet company is on the manage employee screen and there exists at least one valet – valet company association.
Post-conditions	The valet-valet company association is removed and the valet company sees the table less the removed association.

Table 22: Typical Course of Action (Remove Employee)

Seq#	Actor's Action	System's Response
1	The valet company clicks the remove valet button.	
2		The system displays a prompt requesting confirmation.
3	The valet company clicks yes.	
4		The system removes the association from the manage employee table and changes the active flag of that valet in the database. It then returns to the manage employee table.

Table 23: Alternate Course of Action (Remove Employee)

Seq#	Actor's Action	System's Response

1	The valet company clicks the remove valet button.	
2		The system displays a prompt requesting confirmation.
3	The valet company clicks cancel.	
4		The system closes the confirmation and returns to the manage employee table.

Table 24: Exceptional Course of Action (Remove Employee)

Seq#	Actor's Action	System's Response
1	The valet company clicks the remove valet button.	
2		The system alerts the valet company to try again later because of system error.

2.1.3.2.2 Add Employee

Table 25: Process Description (Add Employee)

Identifier	UC-13: Add Employee
Purpose	The functionality to add valet associations to the VC. This process will be used for valet registration and login verification
Requirements	WC_3387
Development Risks	Account security problem
Pre-conditions	The valet company is on the manage employee screen.
Post-conditions	The valet company - valet association is created and the VC sees this data in the manage employees table.

Table 26: Typical Course of Action (Add Employee)

Seq#	Actor's Action	System's Response
1	The valet company enters valet name and submits the form.	
2		The system validates the form then generates a random string key as employee id and displays this information in a table to the valet company on the manage employee screen.

Table 27: Exceptional Course of Action (Add Employee)

Seq#	Actor's Action	System's Response

1	The valet company leaves valet name blank and submits the form.	
2		The system alerts the valet company that this field is required and prevents form submission

Table 28: Exceptional Course of Action (Add Employee)

Seq#	Actor's Action	System's Response
1	The valet company enters valet name and submits the form.	
2		The system alerts the valet company to try again later because of system error.

2.1.3.3 Location Management Service

2.1.3.3.1 Remove Location

Table 29: Process Description (Remove Location)

Identifier	UC-8: Remove Location
Purpose	The functionality of a valet company to remove a valet company-location association
Requirements	WC_3215
Development Risks	Account security problem
Pre-conditions	The valet company is on the manage locations page.
Post-conditions	The valet company - location association is removed from the database and table and the valet company sees the location table.

Table 30: Typical Course of Action (Remove Location)

Seq#	Actor's Action	System's Response
1	The valet company clicks the remove valet button.	
2		The system displays a prompt requesting confirmation.
3	The valet company clicks yes.	
4		The system removes the association from the database. It then returns to the manage location table.

Table 31: Alternate Course of Action (Remove Location)

Seq#	Actor's Action	System's Response
1	The valet company clicks the remove valet button.	
2		The system displays a prompt requesting confirmation.
3	The valet company clicks cancel.	
4		The system closes the confirmation and returns to the manage location table.

Table 32: Exceptional Course of Action (Remove Location)

Seq#	Actor's Action	System's Response
1	The valet company clicks the remove valet button.	
2		The system alerts the valet company to try again later because of system error.

2.1.3.3.2 Add Location

Table 33: Process Description (Add Location)

Identifier	UC-9: Add Location
Purpose	The functionality of a valet company to add a valet company-location association.
Requirements	WC_3387
Development Risks	Account security problem
Pre-conditions	The valet company is on the manage location screen.
Post-conditions	The valet company - location association is stored in the database and table and the valet company sees the location table.

Table 34: Typical Course of Action (Add Location)

Seq#	Actor's Action	System's Response
1	The valet company clicks the add location button.	
2		The system displays a location form prompting for an address string.
3	The valet company enters an address string and submit.	
4		The system validates the form and uses the Google Places API to search for the best match.

5	The valet company selects the result that matches its location.	
6		The system takes the location name, position, and address and stores this information in the database and location table. The system then displays the location table

Table 35: Exceptional Course of Action (Add Location)

Seq#	Actor's Action	System's Response
1	The valet company clicks the add location button.	
2		The system displays a location form prompting for an address string.
3	The valet company enters an address string and submit.	
4		The system validates the form and find no result by using the Google Places API. Then system will advise the VC to try using a more general address

Table 36: Exceptional Course of Action (Add Location)

Seq#	Actor's Action	System's Response
1	The valet company clicks the add location button.	
2		The system displays a location form prompting for an address string.
3	The valet company leaves address field blank and submit.	
4		The system validates the form and alert the valet company that the field is required and prevent form submission

Table 37: Exceptional Course of Action (Add Location)

Seq#	Actor's Action	System's Response
1	The valet company clicks the add location button.	
2		The system alerts the user to please try again later as server error.

2.1.3.4 Account Management

2.1.3.4.1 Register as Valet

Table 38: Process Description (Register as Valet)

Identifier	UC-2: Register as Valet
Purpose	The registration process for valets that work for companies that use SnapValet for mobile requests/payments. This registration is not mandatory for all valets working at a venue. Only one valet need be registered and logged in.
Requirements	WC_3387
Development Risks	Account security problem
Pre-conditions	The valet is on the start screen.
Post-conditions	The valet is registered with an association to his/her company and sees the start screen.

Table 39: Typical Course of Action (Register as Valet)

Seq#	Actor's Action	System's Response
1	The valet presses the register button.	
2		The start screen displays registration page.
3	The valet chooses the valet option.	
4		The system navigates to the valet registration screen.
5	The valet enters an email, password, name, company name, and employee id and submits the form.	
6		The system validates the email and password. The system also validates that there is a match between the company name and employee id. The system then displays a spinner while it stores the information. After process finished, the system displays a confirmation and navigates to the start screen.

Table 40: Exceptional Course of Action (Register as Valet)

Seq#	Actor's Action	System's Response
1	The valet presses the register button.	
2		The start screen displays registration page.
3	The valet chooses the valet option.	
4		The system navigates to the valet registration screen.
5	The valet leaves one of the fields blank/enters an invalid email string/too short password and submits the form.	
6		The system alerts the valet that the form is not fully finished.

Table 41: Exceptional Course of Action (Register as Valet)

Seq#	Actor's Action	System's Response
1	The valet presses the register button.	
2		The start screen displays registration page.
3	The valet chooses the valet option.	
4		The system navigates to the valet registration screen.
5	The valet enters an email, password, name, company name, and employee id and submits the form.	
6		The system find the valet enters a company name/employee id key-value pair NOT contained in the database. Then it will alert the valet that the company name and/or employee id is invalid following submission

Table 42: Exceptional Course of Action (Register as Valet)

Seq#	Actor's Action	System's Response
1	The valet presses the register button.	
2		The start screen displays registration

		page.
3	The valet chooses the valet option.	
4		The system navigates to the valet registration screen.
5	The valet enters an email, password, name, company name, and employee id and submits the form.	
6		The system alerts the valet that something is wrong and to try again later because of server error.

2.1.3.4.2 Register as Customer

Table 43: Process Description (Register as Customer)

Identifier	UC-7: Register as Customer
Purpose	The registration process for customers that utilize valet parking services via SnapValet.
Requirements	WC_3387
Development Risks	Account security problem
Pre-conditions	The customer is on the start screen.
Post-conditions	The customer is now registered with SnapValet and now sees the start page.

Table 44: Typical Course of Action (Register as Customer)

Seq#	Actor's Action	System's Response
1	The customer presses the register button.	
2		The start screen displays registration page.
3	The customer chooses the customer option.	
4		The system navigates to the customer registration screen.
5	The customer enters an email, password, name, and credit card information and submits the form.	
6		The system validates the required information and displays a spinner

		while it stores the information. After process finished, the system displays a confirmation and navigates to the start screen.

Table 45: Exceptional Course of Action (Register as Customer)

Seq#	Actor's Action	System's Response
1	The customer presses the register button.	
2		The start screen displays registration page.
3	The customer chooses the customer option.	
4		The system navigates to the customer registration screen.
5	The customer leaves one of the fields blank/enters an invalid email string/too short password/invalid credit card information and submits the form.	
6		The system alerts the customer that the form is not fully finished.

Table 46: Exceptional Course of Action (Register as Customer)

Seq#	Actor's Action	System's Response
1	The customer presses the register button.	
2		The start screen displays registration page.
3	The customer chooses the customer option.	
4		The system navigates to the customer registration screen.
5	The customer enters an email, password, name, company name, and employee id and submits the form.	
6		The system alerts the customer that something is wrong and to try again later because of server error.

2.1.3.4.3 Register as Valet Company

Table 47: Process Description (Register as Valet Company)

Identifier	UC-12: Register as Valet Company
Purpose	The registration process for valet company that utilize valet parking services via SnapValet.
Requirements	WC_3387
Development Risks	Account security problem
Pre-conditions	The valet company is on the start screen.
Post-conditions	The valet company sees the registration confirmation screen and the valet company account is created.

Table 48: Typical Course of Action (Register as Valet Company)

Seq#	Actor's Action	System's Response
1	The valet company enters their user-id, password, company name, address, email, and account information and submits the form.	
2		The system validates the form fields and displays a spinner while it creates the account. The system removes the spinner after creating the account then displays a registration confirmation with a link to login

Table 49: Exceptional Course of Action (Register as Valet Company)

Seq#	Actor's Action	System's Response
1	The valet company leaves one of the fields blank/enters an invalid string/too short password and submits the form.	
2		The system alerts the valet company that the form is not fully finished.

Table 50: Exceptional Course of Action (Register as Valet Company)

Seq#	Actor's Action	System's Response
1	The valet company enters their user-id, password, company name, address, email, and	

	account information and submits the form.	
2		The system alerts the valet company that something is wrong and to try again later because of server error.

2.1.4 Modes of Operation

The SnapValet, as we envision implementing it, will operate in only one mode, so nothing further need be said of modes of operation.

2.2 System Analysis Rationale

Based on our analysis of how the users interact with the system, we have identified 3 classes of operational stakeholders.

Customer: They are the client of valet service users, and their responsibility for this system is to use request system and cashless payment to improve their valet experience.

Valet operator: They are the performer of valet service. Their responsibility for this system is to perform valet service, and receive salary and tips as return.

Valet Company: They are supervise of valet operator. Their responsibility is to manage the valet operators, and to manage locations which provide SnapValet service.

3. Technology-Independent Model

3.1 Design Overview

3.1.1 System Structure

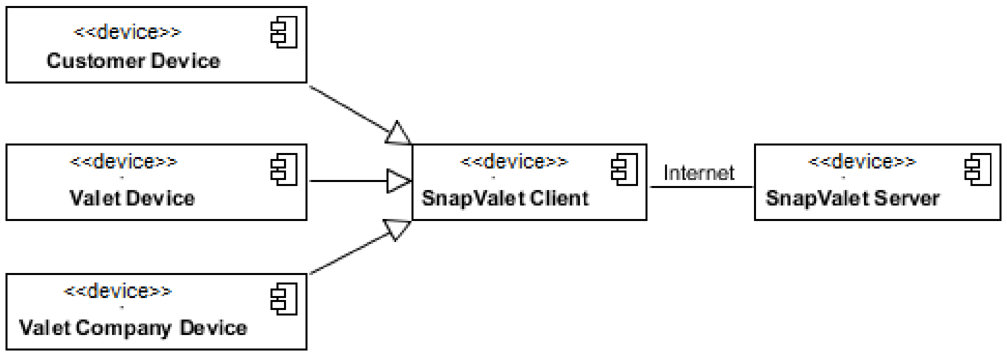


Figure 4: Hardware Component Class Diagram

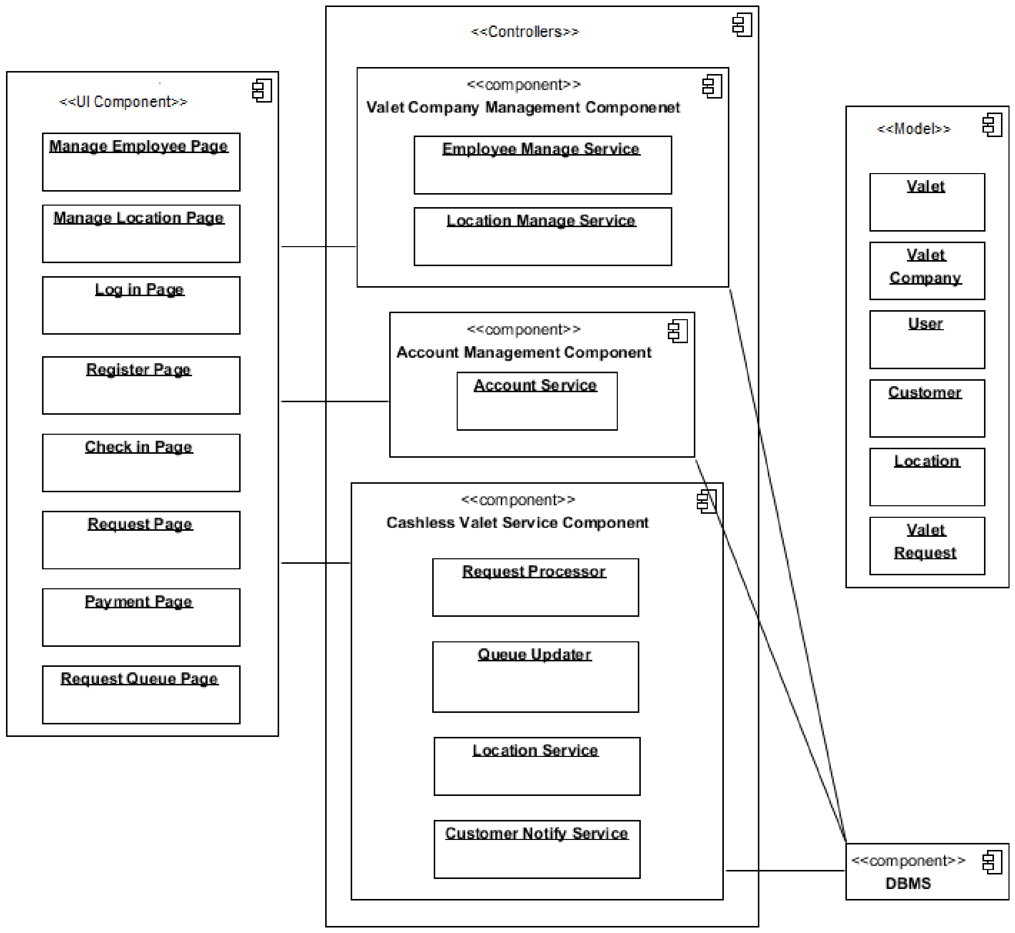


Figure 5: Software Component Class Diagram

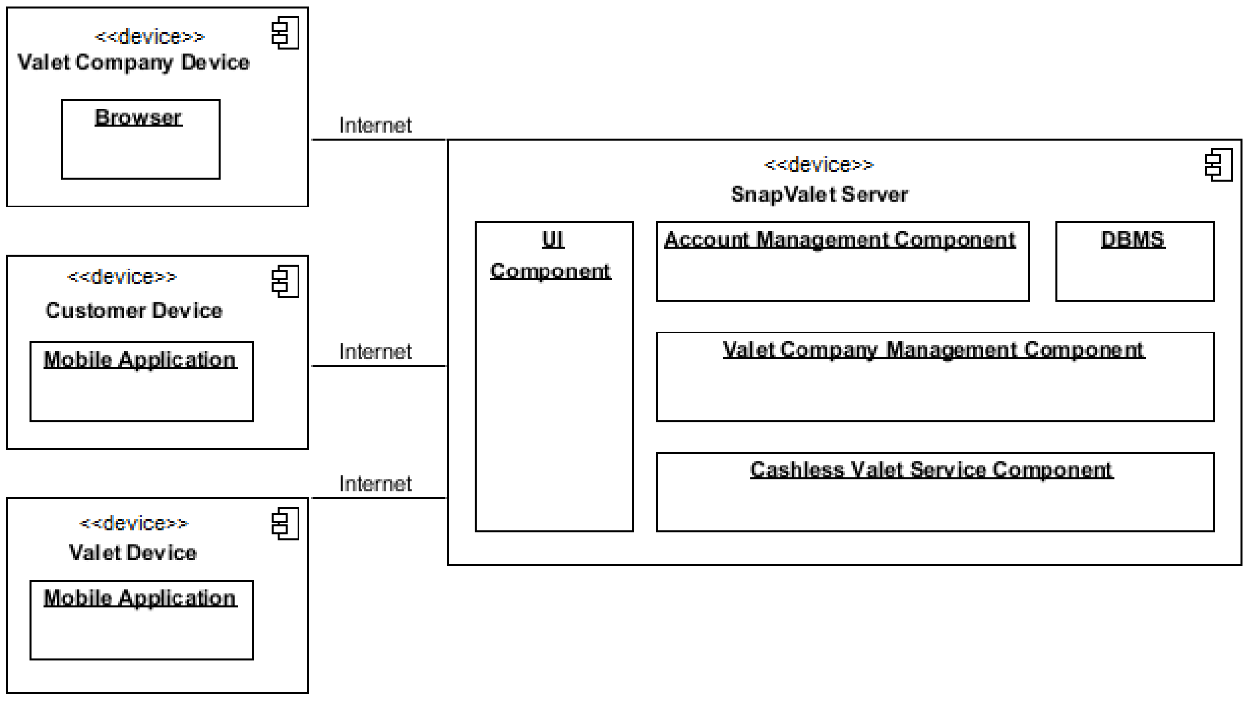


Figure 6: Deployment Diagram

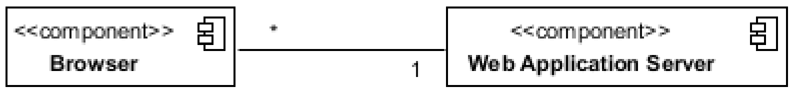


Figure 7: Supporting Software Component Class Diagram

Table 53 and Table 54 contain the descriptions of the hardware and software components in the SnapValet System.

Table 51: Hardware Component Description

Hardware Component	Description
Customer Device	A device that is used by customers to request vehicle retrieval and make mobile payment.
Valet Device	A device that is used by valet to check and accept retrieval request for customers.
Valet Company Device	A device that is used by valet company to manage employee list and location list.
SnapValet Server	A web server that accepts request from valet company, customer and valet device and send response back to specific device. It is responsible for all of business logic.

Table 52: Software Component Description

Software Component	Description
Valet Company Management Component	This component contains employee list management function and location list management function for valet company, and these information will be used for cashless valet service.
Account Management Component	This component contains login and register function for system users to manage their account.
Cashless Valet Service Component	This component contains request car retrieval function, request queue update function and location detect function. All of these functions will be used to support whole valet process.
Model	All data entities used in the system.
UI Component	This component contains all UI classes used by system users.
DBMS	This component support the function of storing and manage data of SnapValet system.

Table 55 contains the descriptions of the web application framework components used in SnapValet system. These components are not implemented by the developers.

Table 53: Supporting Software Component Description

Support Software Component	Description
Browser	An application that is used to connect and display the SnapValet system web pages for valet company.
Web Application Server	A component that accepts request from clients' browser and send response back to clients.

3.1.2 Design Classes

3.1.2.1 UI Classes

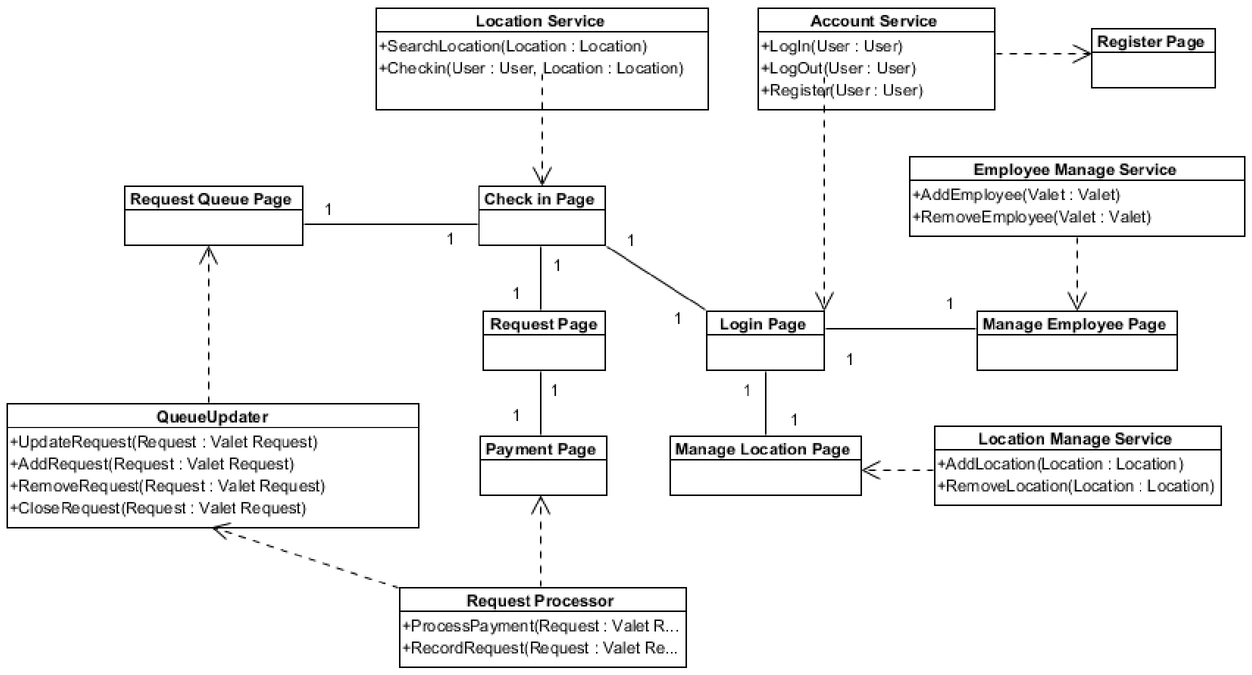

Location Service
+SearchLocation(Location : Location)
+CheckIn(User : User, Location : Location)
Account Service
+LogIn(User : User)
+LogOut(User : User)
+Register(User : User)
Register Page
Employee Manage Service
+AddEmployee(Valet : Valet)
+RemoveEmployee(Valet : Valet)
Request Queue Page
Check in Page
Request Page
Login Page
Manage Employee Page
QueueUpdater
+UpdateRequest(Request : Valet Request)
+AddRequest(Request : Valet Request)
+RemoveRequest(Request : Valet Request)
+CloseRequest(Request : Valet Request)
Payment Page
Manage Location Page
Location Manage Service
+AddLocation(Location : Location)
+RemoveLocation(Location : Location)
Request Processor
+ProcessPayment(Request : Valet R...
+RecordRequest(Request : Valet Re...
1

Figure 8: Design Class Diagram (UI Classes)

Table 56 contains the description for each design class for the set of UI classes, except location service, request processor, queue updater, location manage service, employee manage service and account service, as they come from other packages.

Table 54: Design Class Description (UI Classes)

Class	Type	Description
Manage Employee Page	Boundary	This page displays the employee list and provides basic management function (i.e. add and remove employee)
Manage Location Page	Boundary	This page displays the location list and provides basic management function (i.e. add and remove location)
Register Page	Boundary	This page provides register function for system users.
Login Page	Boundary	This page provides login function for system users.
Check in Page	Boundary	This page displays a map for select location and provides check in function for valet and customer.
Request Page	Boundary	This page displays ticket number input field

		and provides request car retrieval function for customer.
Payment Page	Boundary	This page provides mobile payment function for customer.
Request Queue Page	Boundary	This page support valet to manage request from customers.

3.1.2.2 Account Management Classes

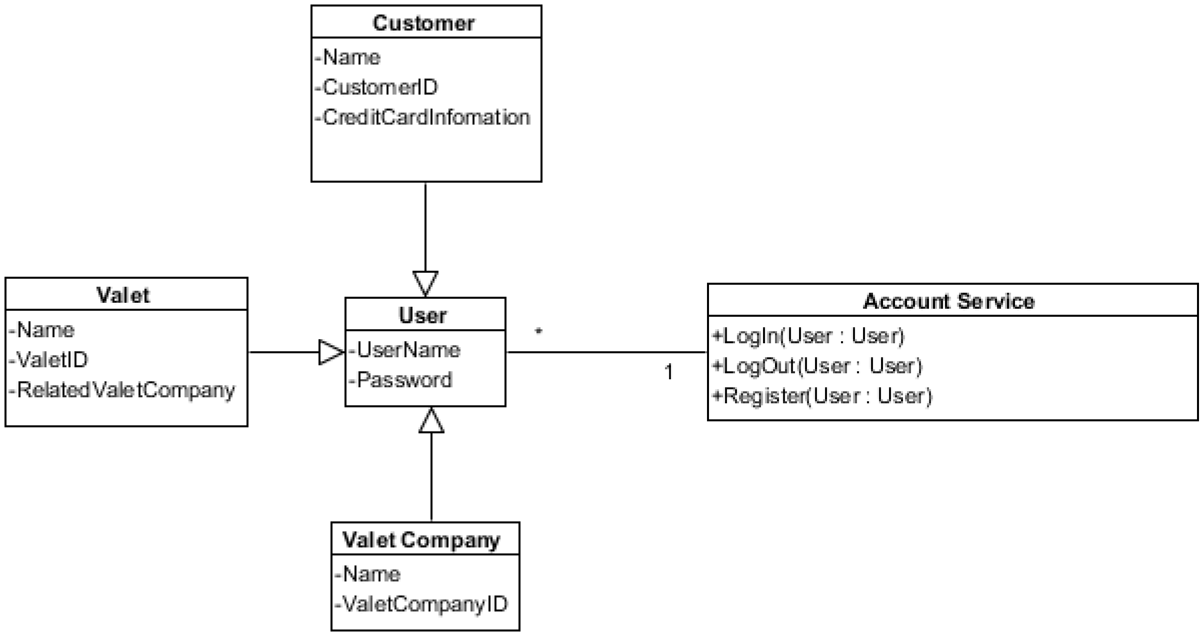


Figure 9: Design Class Diagram (Account Management Classes)

Table 57 contains the description for each design class for the set of account management classes.

Table 55: Design Class Description (Account Management Classes)

Class	Type	Description
User	Entity	This entity contains all basic information of a mobile application user (customer and valet).
Customer	Entity	This entity contains all extra information of a customer.
Valet	Entity	This entity contains all extra information of a valet.
Account Service	Controller	This controller controls business logic of login/logout and register for system users.
Valet Company	Entity	This entity contains all information of a web application user (valet company).

3.1.2.3 Valet Company Management Classes

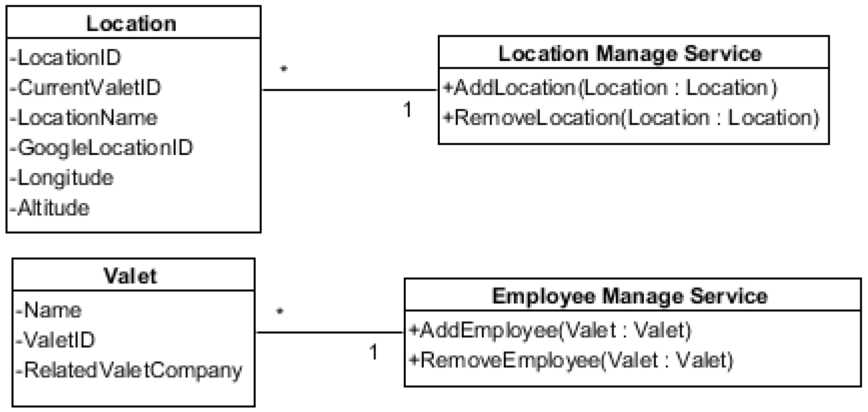


Figure 10: Design Class Diagram (Valet Company Management Classes)

Table 58 contains the description for each design class for the set of valet company management classes, except valet, as it comes from Account Management Classes.

Table 56: Design Class Description (Valet Company Management Classes)

Class	Type	Description
Location	Entity	This entity contains all information for a location.
Location Manage Service	Controller	This controller contains business logic for add/remove location.
Employee Manage Service	Controller	This controller contains business logic for add/remove employee.

3.1.2.4 Cashless Valet Service Classes

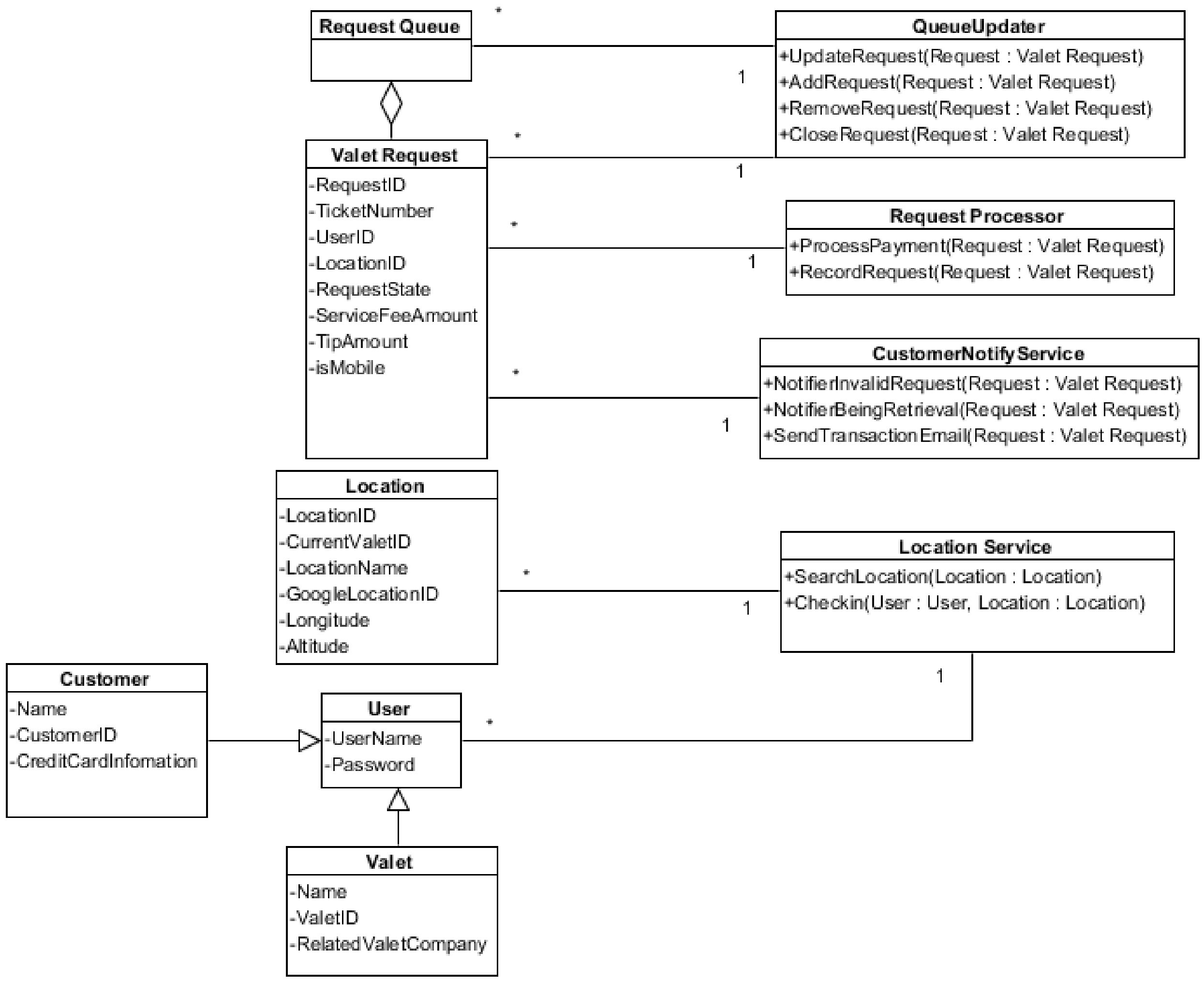

Request Queue
QueueUpdater
+UpdateRequest(Request : Valet Request)
+AddRequest(Request : Valet Request)
+RemoveRequest(Request : Valet Request)
+CloseRequest(Request : Valet Request)
Valet Request
-RequestID
-TicketNumber
-UserID
-LocationID
-RequestState
-ServiceFeeAmount
-TipAmount
-isMobile
Request Processor
+ProcessPayment(Request : Valet Request)
+RecordRequest(Request : Valet Request)
CustomerNotifyService
+NotifierInvalidRequest(Request : Valet Request)
+NotifierBeingRetrieval(Request : Valet Request)
+SendTransactionEmail(Request : Valet Request)
Location
-LocationID
-CurrentValetID
-LocationName
-GoogleLocationID
-Longitude
-Altitude
Location Service
+SearchLocation(Location : Location)
+Checkin(User : User, Location : Location)
Customer
-Name
-CustomerID
-CreditCardInfomation
User
-UserName
-Password
Valet
-Name
-ValetID
-RelatedValetCompany
*
1

Figure 11: Design Class Diagram (Cashless Valet Service Classes)

Table 59 contains the description for each design class for the set of cashless valet service classes, except customer, user, valet, location, as they come from other packages.

Table 57: Design Class Description (Cashless Valet Service Classes)

Class	Type	Description
Valet Request	Entity	This entity contains all information for a car retrieval request from customer.
Request Queue	Entity	This entity contains all requests of a specific location
Queue Updater	Controller	This controller contains all business logic for updating requests showed in a queue.
Request Processor	Controller	This controller contains all business logic for processing request from customer, including

		processing transaction.
Location Service	Controller	This controller contains all business logic for searching location and checking in.
Customer Notify Service	Controller	This controller contains all business logic for notify customer about his/her request.

3.1.3 Process Realization

This section describes how each request be made by customer and how it be processed.

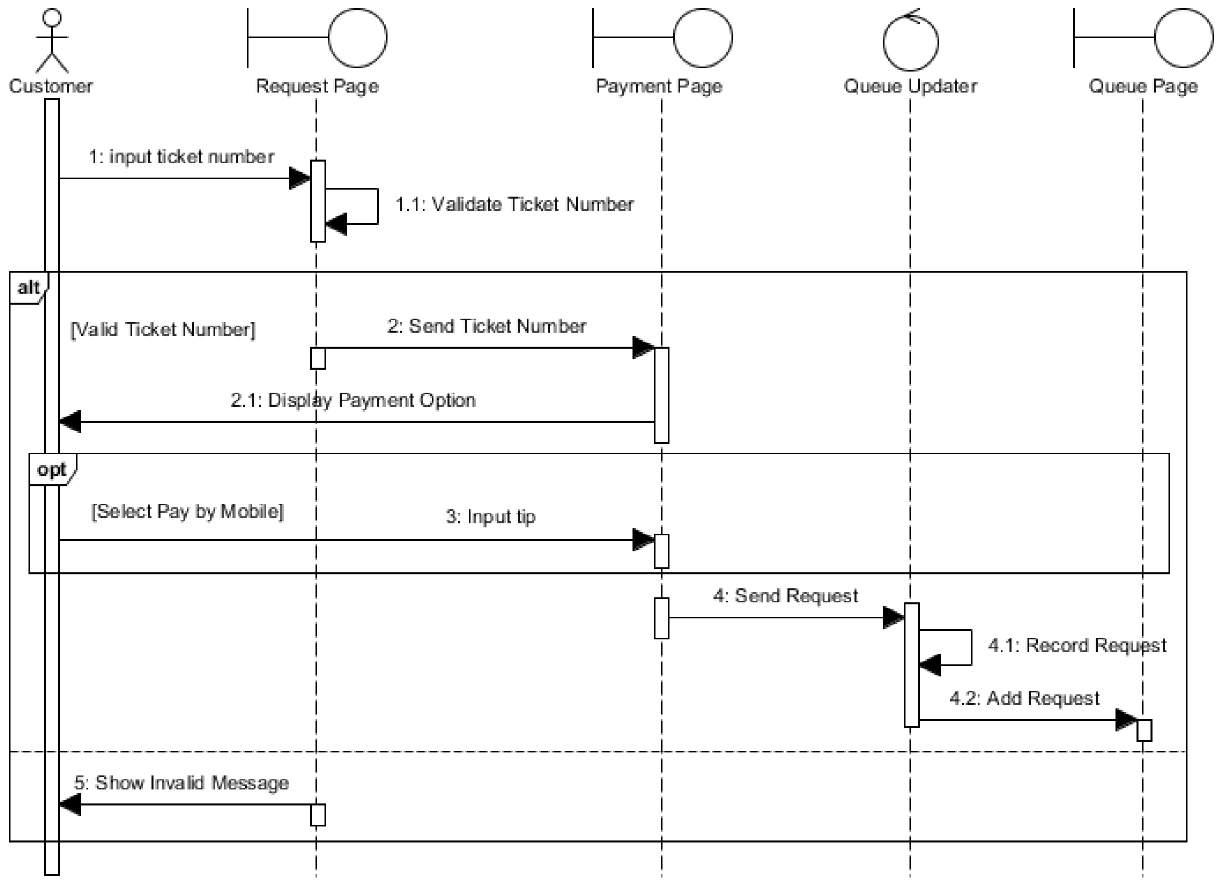


Figure 12: Process Realization Diagram (Request & Pay)

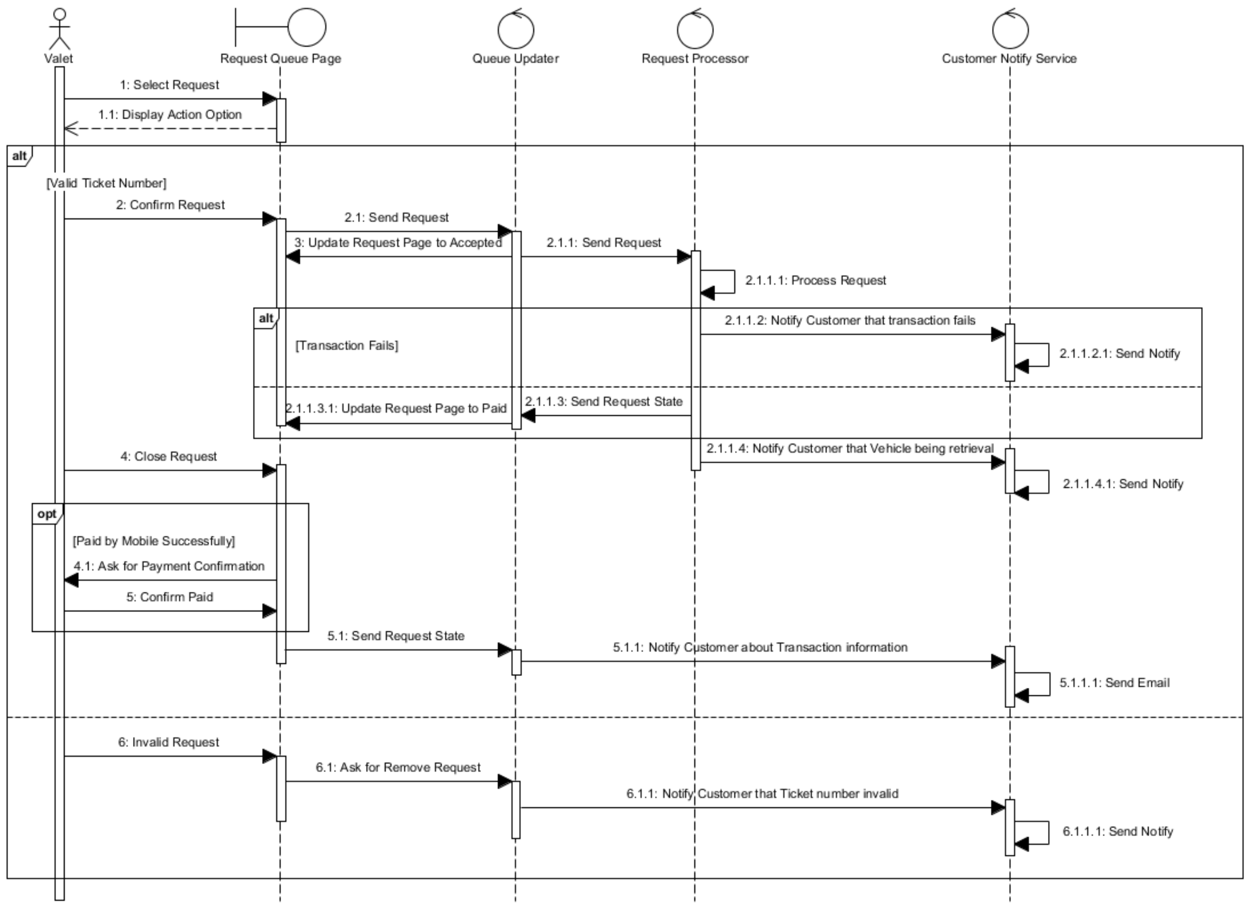

Valet
Request Queue Page
Queue Updater
Request Processor
Customer Notify Service
1: Select Request
1.1: Display Action Option
alt
[Valid Ticket Number]
2: Confirm Request
2.1: Send Request
3: Update Request Page to Accepted
2.1.1: Send Request
2.1.1.1: Process Request
alt
[Transaction Fails]
2.1.1.2: Notify Customer that transaction fails
2.1.1.2.1: Send Notify
2.1.1.3.1: Update Request Page to Paid
2.1.1.3: Send Request State
2.1.1.4: Notify Customer that Vehicle being retrieval
2.1.1.4.1: Send Notify
4: Close Request
opt
[Paid by Mobile Successfully]
4.1: Ask for Payment Confirmation
5: Confirm Paid
5.1: Send Request State
5.1.1: Notify Customer about Transaction information
5.1.1.1: Send Email
6: Invalid Request
6.1: Ask for Remove Request
6.1.1: Notify Customer that Ticket number invalid
6.1.1.1: Send Notify

Figure 13: Process Realization Diagram (Retrieval & Return Vehicle)

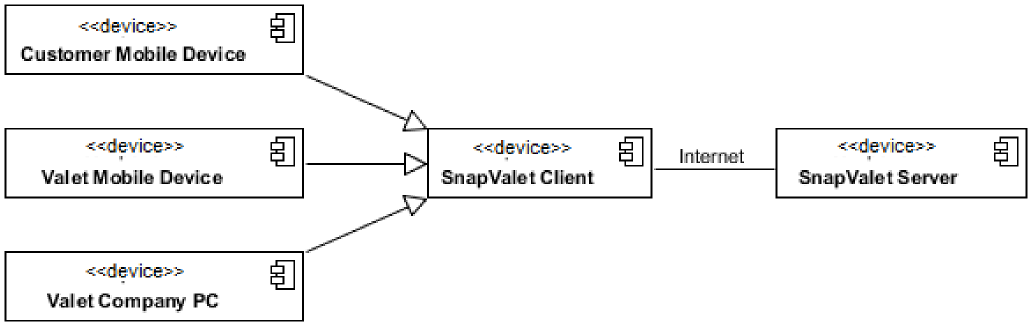
3.2 Design Rationale

We adopted a MVC architecture because it is flexible enough for our system: put data, service and display into different classes, which will make it much easier to make change in any of them. We decided to use a COTS DBMS because it would cost less resources than developing a data management system by ourselves.
We chose to show the sequence diagram for the Request & Pay and Retrieve & Return vehicle use case as they are most important part for SnapValet system, which is aiming to provide a cashless valet service.

4. Technology-Specific System Design

4.1 Design Overview

4.1.1 System Structure



<<device>> Customer Mobile Device
<<device>> Valet Mobile Device
<<device>> Valet Company PC
<<device>> SnapValet Client
Internet
<<device>> SnapValet Server

Figure 14: Hardware Component Class Diagram

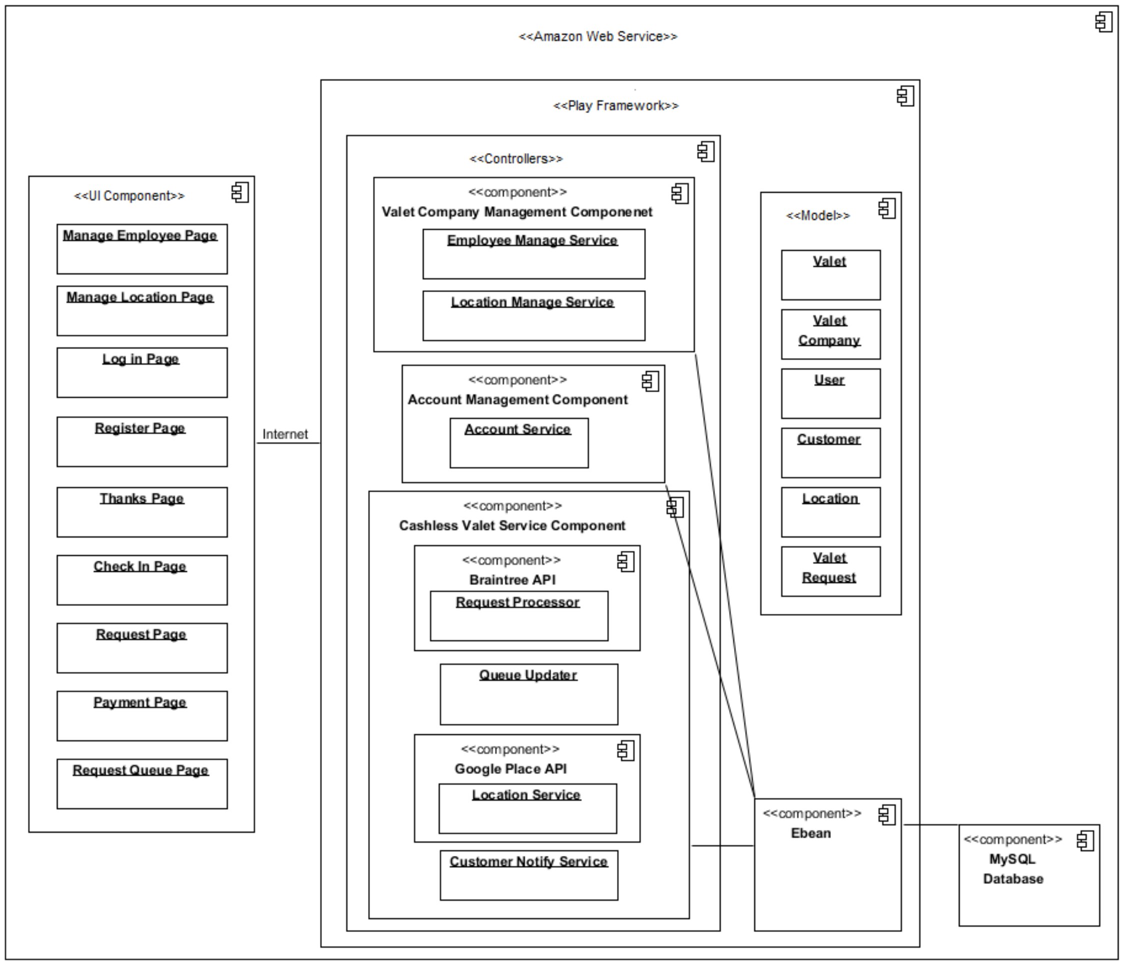


Figure 15: Software Component Class Diagram

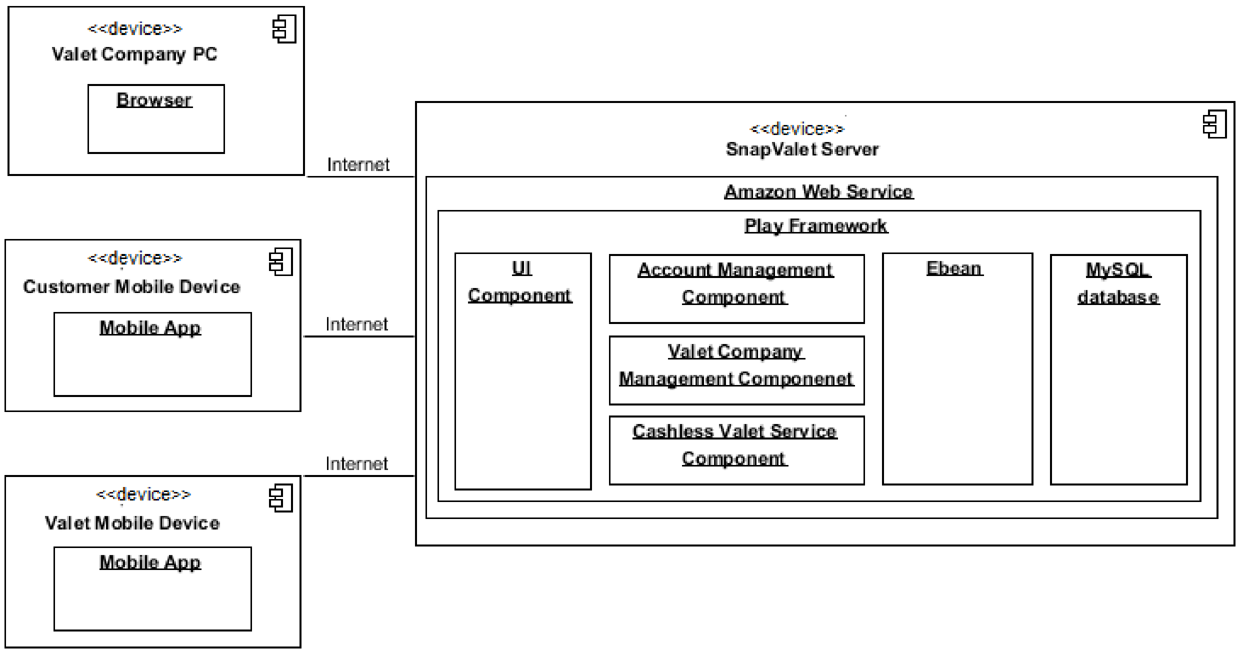


Figure 16: Deployment Diagram

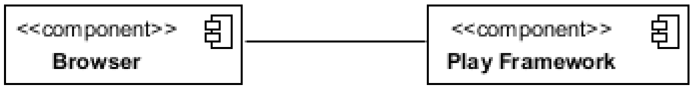


Figure 17: Supporting Software Component Class Diagram

Table 60 and Table 61 contain the technology-specific descriptions of the hardware and software components in the SnapValet System.

Table 58: Hardware Component Description

Hardware Component	Description
Customer Mobile Device	A mobile device used by customer to send car retrieval request and make mobile payment
Valet Mobile Device	A mobile device used by valet to accept car retrieval requests from customers.
Valet Company PC	A computer used by valet company to manage valet list and location list via browser.
SnapValet Server	A server device that accepts request from valet company, customer and valet device and send response back to specific device. It is responsible for all of business logic.

Table 59: Software Component Description

Software Component	Description
Valet Company Management Component	This component contains employee list management function and location list management function for valet company, and these information will be used for cashless valet service. Implemented by Java.
Account Management Component	This component contains login and register function for system users to manage their account. Implemented by Java.
Cashless Valet Service Component	This component contains request car retrieval function, request queue update function and location detect function. All of these functions will be used to support whole valet process. And the location service will talk with Google Map API, the request processor will talk with Brian Tree API to make mobile payment. Implemented by Java.
Google Place API	A location service provided by Google. It helps to detect the surrounding venues in SnapValet system
Braintree API	An online transaction service. It helps system to process online payment.
Play Framework	A web application framework. It helps to organize the structure of the system.
UI Component	This component contains all web pages used by system users.
Model	All data entities used in the system.
Amazon Web Service	A web server application which is responsible to hold all backend code.
Ebean	This component contains all function to access data storing in database. Implemented by Java.
MySQL	This component support the function of storing and manage data of SnapValet system.

Table 62 contains the technology-specific descriptions of the web application framework components used in SnapValet system. These components are not implemented by the developers.

Table 60: Supporting Software Component Description

Support Software Component	Description
Browser	An application (Chrome/Safari/Firefox/…) that is used to connect and display the SnapValet system web pages for valet company.
Amazon Web Service	A component that accepts request from clients’ browser and send response back to clients.

4.1.2 Design Classes

4.1.2.1 UI Classes

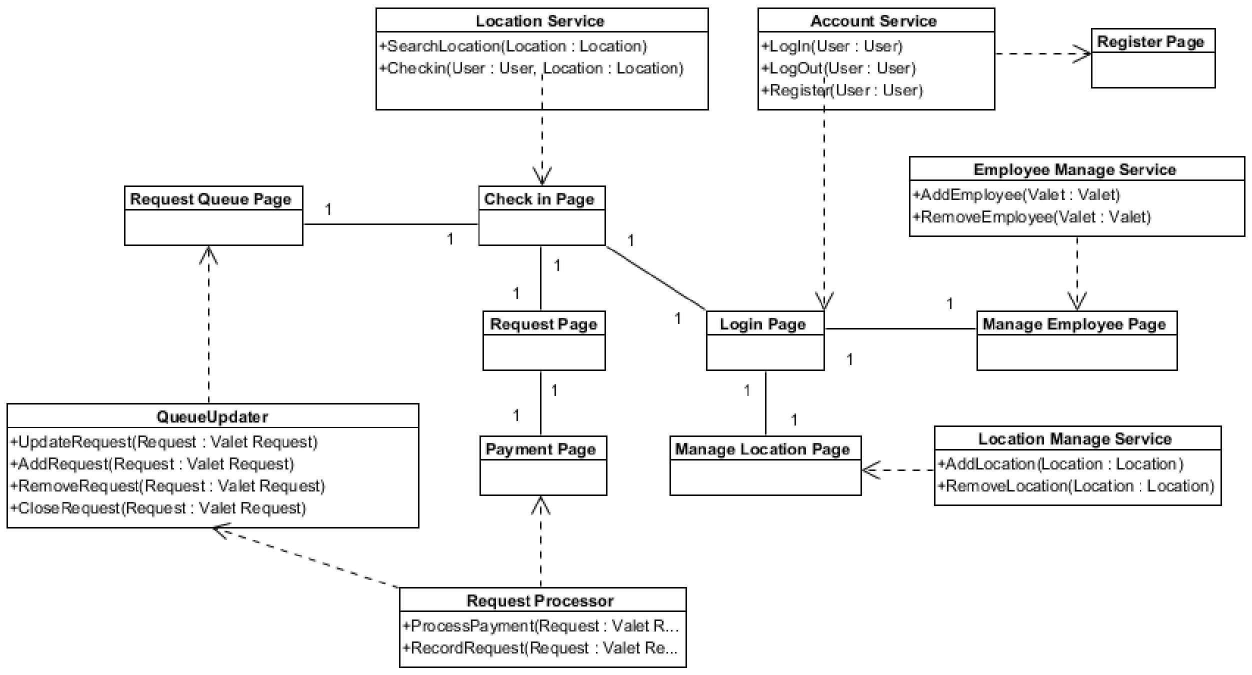

Location Service
+SearchLocation(Location : Location)
+Checkin(User : User, Location : Location)
Account Service
+LogIn(User : User)
+LogOut(User : User)
+Register(User : User)
Register Page
Employee Manage Service
+AddEmployee(Valet : Valet)
+RemoveEmployee(Valet : Valet)
Request Queue Page
Check in Page
Request Page
Login Page
Manage Employee Page
QueueUpdater
+UpdateRequest(Request : Valet Request)
+AddRequest(Request : Valet Request)
+RemoveRequest(Request : Valet Request)
+CloseRequest(Request : Valet Request)
Payment Page
Manage Location Page
Location Manage Service
+AddLocation(Location : Location)
+RemoveLocation(Location : Location)
Request Processor
+ProcessPayment(Request : Valet R...
+RecordRequest(Request : Valet Re...
1

Figure 18: Design Class Diagram (UI Classes)

Table 63 contains the technology-specific description for each design class for the set of UI classes, except location service, request processor, queue updater, location manage service, employee manage service and account service, as they come from other packages. All HTTP request sent by web page will be parsed by HTTP Request Parser Layer server and it will resend it to specific controller.

Table 61: Design Class Description (UI Classes)

Class	Type	Description
Manage Employee Page	Boundary	This page displays the employee list and provides basic management function (i.e. add and remove employee). And in this web page we will use html/css/javascript.
Manage Location Page	Boundary	This page displays the location list and provides basic management function (i.e. add and remove location). And in this web page we will use html/css/javascript.
Register Page	Boundary	This page provides register function for system users. And in this web page we will use html/css/javascript.
Login Page	Boundary	This page provides login function for system

		users. And in this web page we will use html/css/javascript.
Check in Page	Boundary	This page displays a map for select location and provides check in function for valet and customer. And in this web page we will use html/css/javascript.
Request Page	Boundary	This page displays ticket number input field and provides request car retrieval function for customer. And in this web page we will use html/css/javascript.
Payment Page	Boundary	This page provides mobile payment function for customer. And in this web page we will use html/css/javascript.

4.1.2.2 Account Management Classes

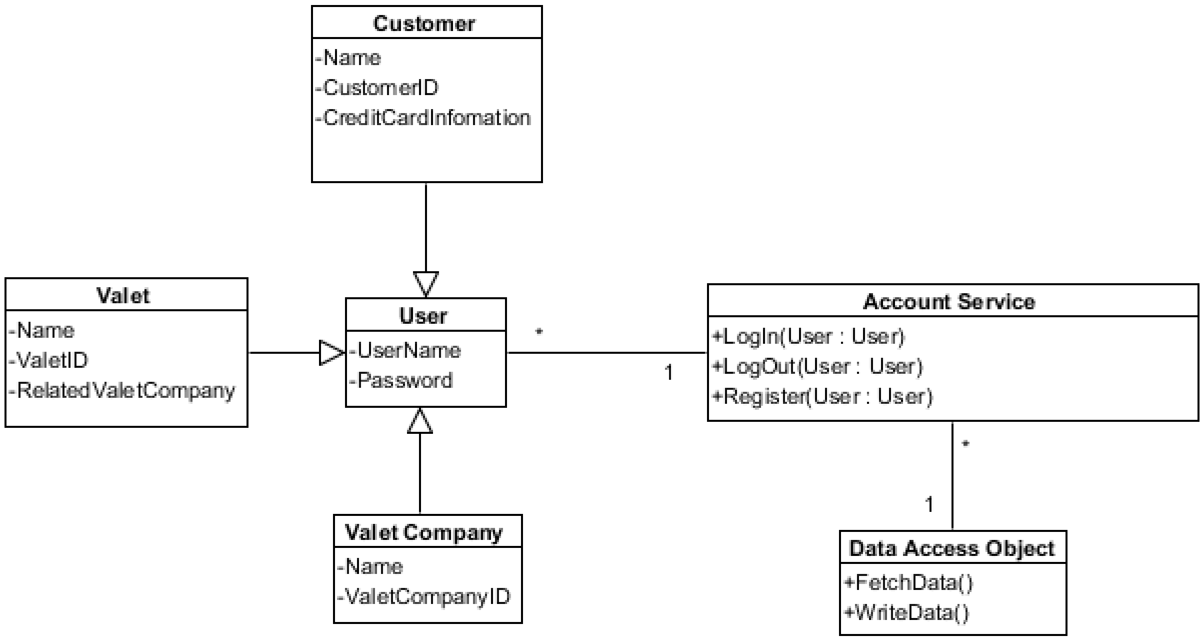


Figure 19: Design Class Diagram (Account Management Classes)

Table 64 contains the technology-specific description for each design class for the set of account management classes, except Data Access Object, as it comes from other package.

Table 62: Design Class Description (Account Management Classes)

Class	Type	Description
User	Entity	This entity contains all basic information of a mobile application user (customer and valet).
Customer	Entity	This entity contains all extra information of a customer.
Valet	Entity	This entity contains all extra information of a

		valet.
Account Service	Controller	This controller controls business logic of login/logout and register for system users.
Valet Company	Entity	This entity contains all information of a web application user (valet company).

4.1.2.3 Valet Company Management Classes

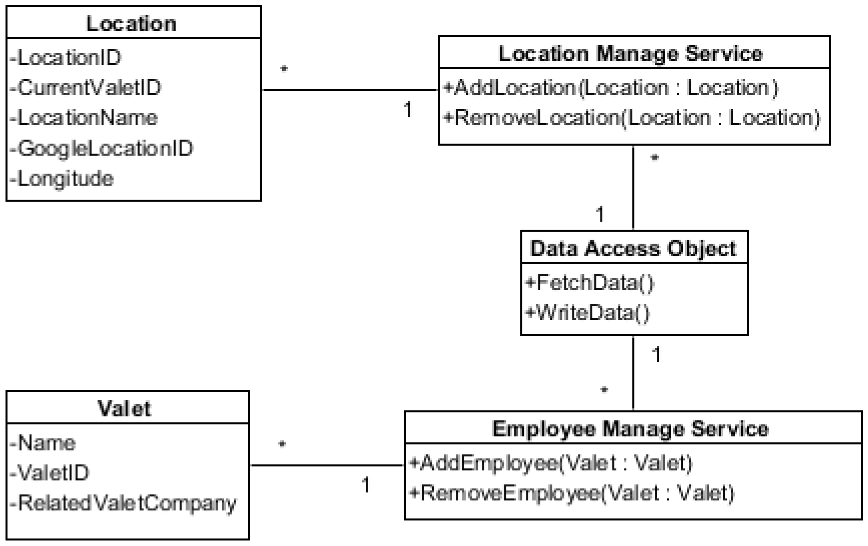


Figure 20: Design Class Diagram (Valet Company Management Classes)

Table 65 contains the technology-specific description for each design class for the set of valet company management classes, except valet and Data Access Object, as they come from other package.

Table 63: Design Class Description (Valet Company Management Classes)

Class	Type	Description
Location	Entity	This entity contains all information for a location.
Location Manage Service	Controller	This controller contains business logic for add/remove location.
Employee Manage Service	Controller	This controller contains business logic for add/remove employee.

4.1.2.4 Cashless Valet Service Classes

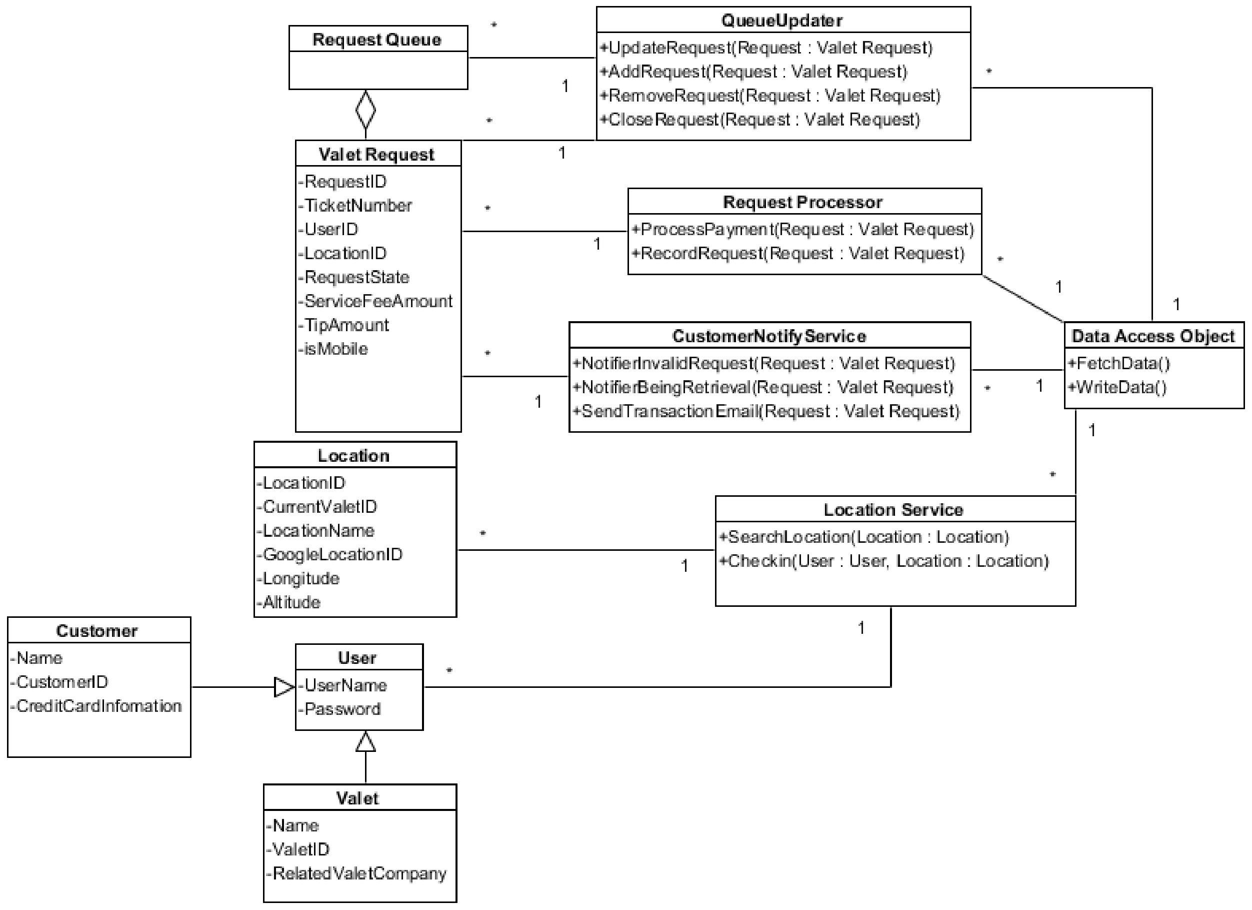

Request Queue
QueueUpdater
+UpdateRequest(Request : Valet Request)
+AddRequest(Request : Valet Request)
+RemoveRequest(Request : Valet Request)
+CloseRequest(Request : Valet Request)
Valet Request
-RequestID
-TicketNumber
-UserID
-LocationID
-RequestState
-ServiceFeeAmount
-TipAmount
-isMobile
Request Processor
+ProcessPayment(Request : Valet Request)
+RecordRequest(Request : Valet Request)
CustomerNotifyService
+NotifierInvalidRequest(Request : Valet Request)
+NotifierBeingRetrieval(Request : Valet Request)
+SendTransactionEmail(Request : Valet Request)
Data Access Object
+FetchData()
+WriteData()
Location
-LocationID
-CurrentValetID
-LocationName
-GoogleLocationID
-Longitude
-Altitude
Location Service
+SearchLocation(Location : Location)
+Checkin(User : User, Location : Location)
Customer
-Name
-CustomerID
-CreditCardInfomation
User
-UserName
-Password
Valet
-Name
-ValetID
-RelatedValetCompany

Figure 21: Design Class Diagram (Cashless Valet Service Classes)

Table 66 contains the technology-specific description for each design class for the set of cashless valet service classes, except customer, user, valet, location and Data Access Object, as they come from other packages.

Table 64: Design Class Description (Cashless Valet Service Classes)

Class	Type	Description
Payment	Entity	This entity contains amount information for a transaction.
Request	Entity	This entity contains all information for a car retrieval request from customer.
Queue	Entity	This entity contains all requests of a specific location
Queue Updater	Controller	This controller contains all business logic for updating requests showed in a queue. And it will update the Request Queue Activity in valet android device.

Request Processor	Controller	This controller contains all business logic for processing request from customer, including processing transaction. The mobile payment part will use Braintree API.
Location Service	Controller	This controller contains all business logic for searching location and checking in. This controller will use Google Map API and location service of Android device.
Customer Notify Service	Controller	This controller contains all business logic for notify customer about his/her request.

4.1.2.5 Data Access Layer Classes

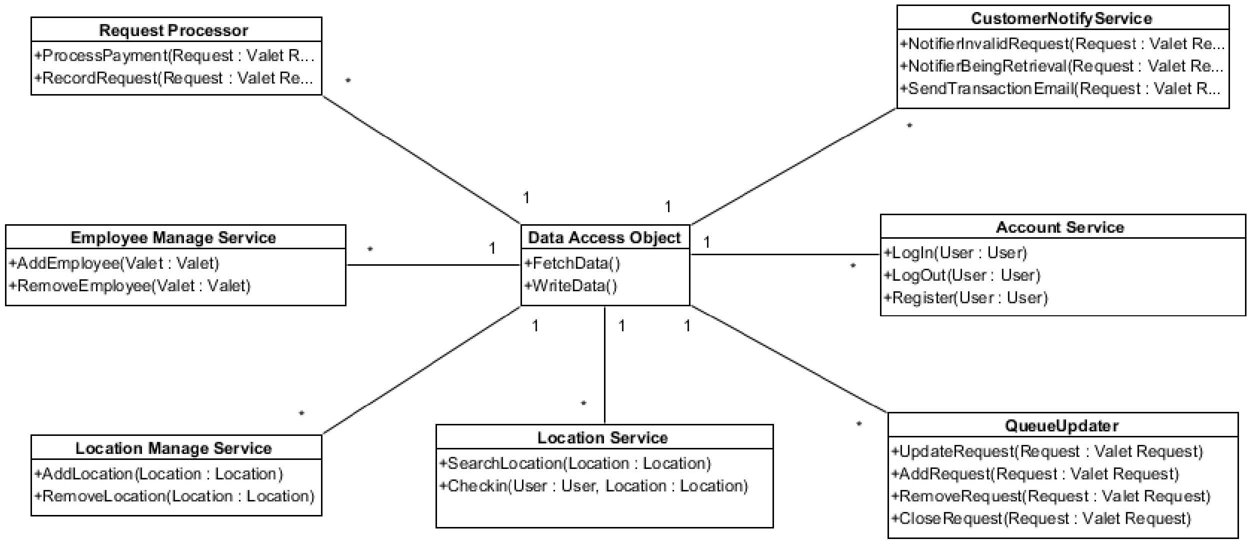

Request Processor
+ProcessPayment(Request : Valet R...
+RecordRequest(Request : Valet Re...
CustomerNotifyService
+NotifierInvalidRequest(Request : Valet Re...
+NotifierBeingRetrieval(Request : Valet Re...
+SendTransactionEmail(Request : Valet R...
Employee Manage Service
+AddEmployee(Valet : Valet)
+RemoveEmployee(Valet : Valet)
Data Access Object
+FetchData()
+WriteData()
Account Service
+LogIn(User : User)
+LogOut(User : User)
+Register(User : User)
Location Manage Service
+AddLocation(Location : Location)
+RemoveLocation(Location : Location)
Location Service
+SearchLocation(Location : Location)
+Checkin(User : User, Location : Location)
QueueUpdater
+UpdateRequest(Request : Valet Request)
+AddRequest(Request : Valet Request)
+RemoveRequest(Request : Valet Request)
+CloseRequest(Request : Valet Request)
1
*

Figure 22: Design Class Diagram (Data Access Layer Classes)

Table 67 contains the description for Data Access Object.

Table 65: Design Class Description (Data Access Layer Classes)

Class	Type	Description
Data Access Object	Controller	This controller contains function to read/write data from database. In this project we will use Ebean to deal with connection with database.

4.1.3 Process Realization

This section describes how each request be made by customer and how it be processed.

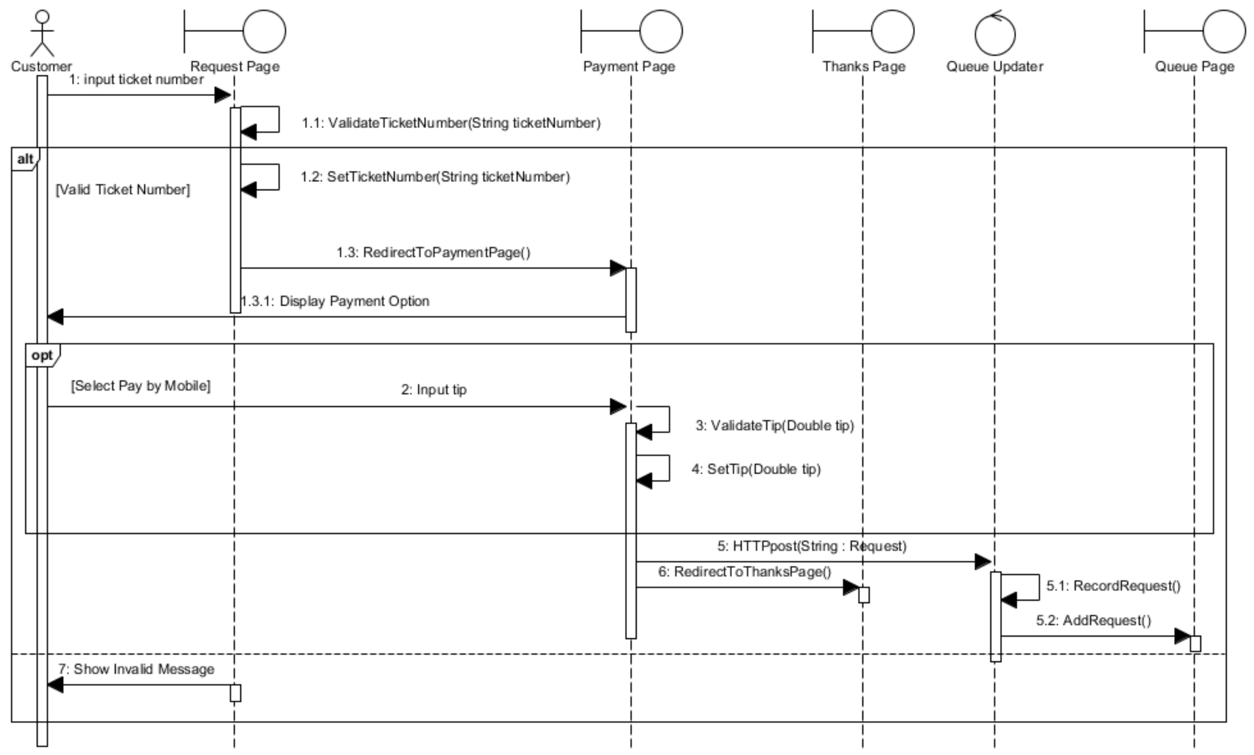


Figure 23: Process Realization Diagram (Request & Pay)

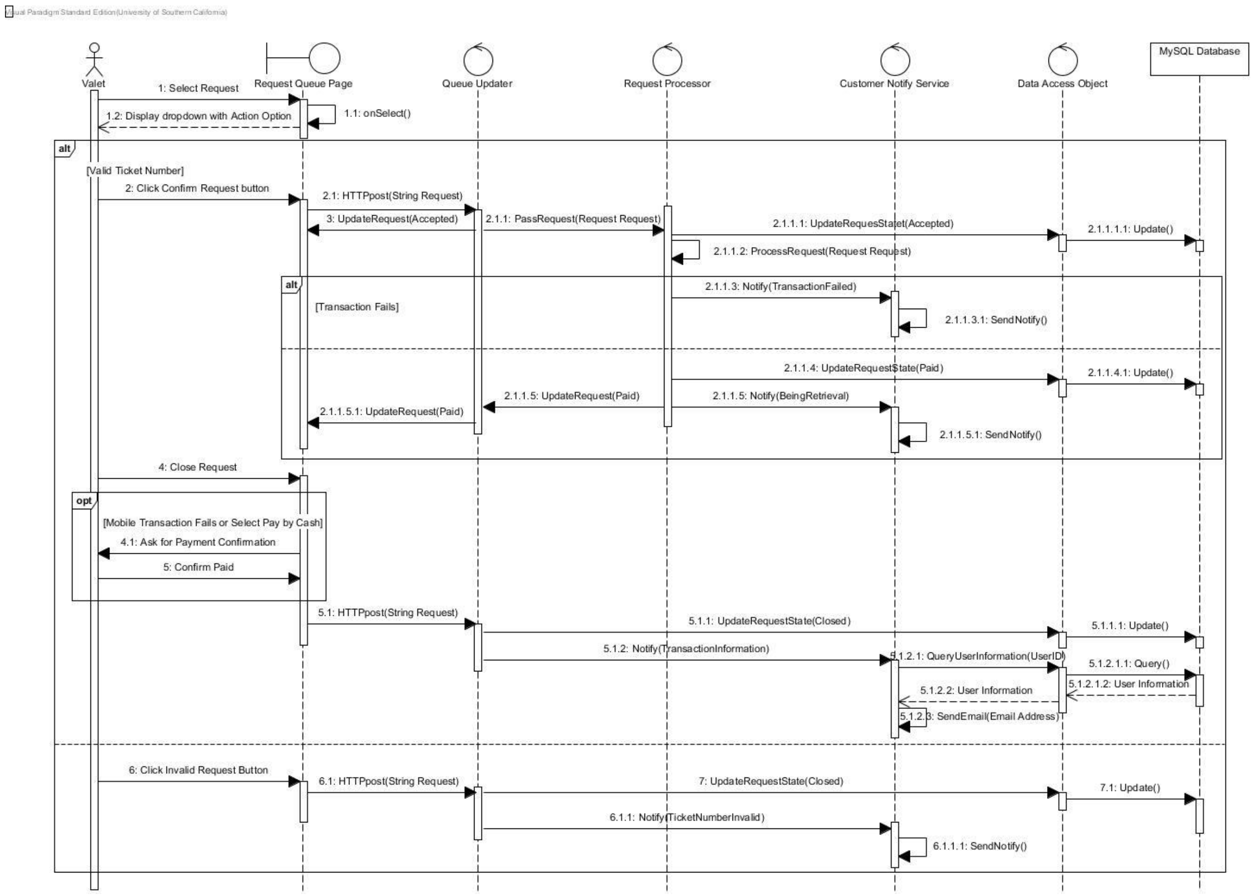

Visual Paradigm Standard Edition(University of Southern California)
Valet
Request Queue Page
Queue Updater
Request Processor
Customer Notify Service
Data Access Object
MySQL Database
1: Select Request
1.1: onSelect()
1.2: Display dropdown with Action Option
alt
[Valid Ticket Number]
2: Click Confirm Request button
2.1: HTTPpost(String Request)
3: UpdateRequest(Accepted)
2.1.1: PassRequest(Request Request)
2.1.1.1: UpdateRequesState(Accepted)
2.1.1.1.1: Update()
2.1.1.2: ProcessRequest(Request Request)
alt
[Transaction Fails]
2.1.1.3: Notify(TransactionFailed)
2.1.1.3.1: SendNotify()
2.1.1.4: UpdateRequestState(Paid)
2.1.1.4.1: Update()
2.1.1.5: UpdateRequest(Paid)
2.1.1.5: Notify(BeingRetrieval)
2.1.1.5.1: UpdateRequest(Paid)
2.1.1.5.1: SendNotify()
4: Close Request
opt
[Mobile Transaction Fails or Select Pay by Cash]
4.1: Ask for Payment Confirmation
5: Confirm Paid
5.1: HTTPpost(String Request)
5.1.1: UpdateRequestState(Closed)
5.1.1.1: Update()
5.1.2: Notify(TransactionInformation)
5.1.2.1: QueryUserInformation(UserID)
5.1.2.1.1: Query()
5.1.2.1.2: User Information
5.1.2.2: User Information
5.1.2.3: SendEmail(Email Address)
6: Click Invalid Request Button
6.1: HTTPpost(String Request)
7: UpdateRequestState(Closed)
7.1: Update()
6.1.1: Notify(TicketNumberInvalid)
6.1.1.1: SendNotify()

Figure 24: Process Realization Diagram (Retrieval & Return Vehicle)

4.2 Design Rationale

We adopted a MVC architecture because it is flexible enough for our system: put data, service and display into different classes, which will make it much easier to make change in any of them. We decided to use a MySQL, Amazon Web Service, and Play Framework to act as main framework of our application, as it will be time consuming to manage data and parse HTTP request by developers. And we choose to use Google Map API, Braintree API to develop cashless valet service.

We chose to show the sequence diagram for the Request & Pay and Retrieve & Return vehicle use case as they are most important part for SnapValet system, which is aiming to provide a cashless valet service.

And another thing is, we decide to use Phone Gap to develop application for different platforms, which will highly reduce work for developers to learn specific platform language.

5. Architectural Styles, Patterns and Frameworks

Table 66: Architectural Styles, Patterns, and Frameworks

Name	Description	Benefits, Costs, and Limitations
MVC	Model-view-controller (MVC) is a software architecture pattern, and it put user data, user interaction, and user interface into different classes.	Benefits: reduce cost to make modification, and help developer team to divide tasks to different developers. Cost: no specific cost to use MVC. Limitations: make the system design more complex.
Play Framework	A third party web application framework.	Benefits: highly improve the productivity for developing, and make it easier for following developers to get familiar with this project Cost: no specific cost. Limitations: limit the option for different system structures.
Google Map API	A third party API used to search location on a map, and provide a lot of functions to design map.	Benefits: help developer to easily develop a location based application. Cost: no specific cost. Limitations: only can access data provided by Google.
Braintree API	A third party API used to make online payment.	Benefits: help developer to easily develop an application with online payment. Cost: cost per transaction.