.

Software Requirements Specification

Version 5.0

for

Vehicle Renting System

Prepared by

1.Bhavana Bomma	- 40071320 - bomma08@gmail.com
2.Divyaprabha Rajendran	- 40089282 - divyanajma@gmail.com
3.Harish Jayasankar	- 40105791 - harishjayasankar28@gmail.com
4.Ishpreet Singh	- 40093666 - ishpreetsingh923@gmail.com
5.Mahy Salama	- 40074737 - mahyhanysalama@gmail.com
6.Swetha Chenna	- 40092019 - swethachenna2018@gmail.com [Team Leader]

Instructor:	Dr. C. Constantinides
Course:	SOEN-6461
Date:	17/11/2019

Document history

Date	Version	Description	Author
17/11/2019	V.5	Updating document	All the team members
31/10/2019	V.4	Updated team leader info	Swetha Chenna
27/10/2019	V.3.2	Adding administrator functionality	All team members
06/10/2019	V.3.1	Revising, adding missing items and format	All the team members
05/10/2019	V.2.4	Adding sequence diagrams	All the team members
04/10/2019	V.2.3	Adding glossary and references	Mahy
02/10/2019	V.2.2	Adding use cases scenarios and contracts	All the team members
20/09/2019	V.2.1	Use Case Diagram	Harish
18/09/2019	V1	Introduction and purpose and scope	Swetha

Table of contents

1. Introduction 5

Purpose 5

Scope 5

Definitions, acronyms, and abbreviations 5

References 5

2. Overall description 6

Product perspective 6

Product functions 6

User characteristics 6

Constraints 6

Assumptions and dependencies 7

3. Specific requirements 8

3.1 External interfaces 8

3.2 Functional requirements 9

Actor goal list 9

3.3 Non-functional requirements 11

3. Analysis Models 13

List of figures

Figure 1. A Model View Controller Architecture. 6
Figure 2. An example user interface.. 8
Figure 3. Use case model. 10

1. Introduction

Vehicle Rental System is a web application which includes two functionalities namely clerk and admin .Clerk functionality involves reserving ,renting and returning vehicles,managing client record and view the catalog for details. Whereas administrator functionality involves searching catalogues for vehicle details, viewing and managing vehicles records,transaction details.

Purpose

In this software requirement specification document we specify the process of Vehicle Rental System in different aspects starting from identifying requirements from potential users and analysing the requirements to design in broader perspectives using use case modeling,domain model and sequential representations . Implementing the functionality using patterns and to provide good and efficient web application we define non-functional requirements in this document.

Scope

The Software Requirements Specification (SRS) is a communication tool between users and software designers. Scope of this software requirement specification applied mainly to the stakeholders and according to our system we have two users, the clerk who is a primary actor in the scenario handles complete client records and manages all the functionalities, and also the administrator handles vehicle record management.

Definitions, acronyms, and abbreviations

Provides the definitions of all terms, acronyms, and abbreviations required to properly interpret the Software Requirements Specifications Document.

project’s Glossary

Abbreviation	Definition
SRS	Software Requirements Specification, is a description of a software system to be developed. It lays out functional and non-functional requirements and lists sufficient and necessary requirements for the project development.
UC	Use Case, is a list of actions or event steps typically defining the interactions between a role (an actor in UML) and a system to achieve a goal. The actor can be a human, machine or other external systems.
CO	Contract, is a document that describes system behaviour for system operations that the system (as a black box) offers in its public interface to handle incoming system events.
UML	Unified Modeling Language, is a standardized modeling language enables developers to specify, visualize, construct and document the artifacts of a software system.

References

[1] Cockburn, A. (2000). *Writing effective use cases*. Addison-Wesley Professional.

[2] Ladd, S., Davison, D., Devijver, S., Yates, C., Harrop, R., & Donald, K. (2006). *Expert Spring MVC and Web Flow* (Vol. 1). Berkeley, CA: Apress.

[3] Larman, C. (2012). *Applying UML and patterns: an introduction to object oriented analysis and design and iterative development*. Pearson Education India.

2. Overall description

This section describes the background to the requirements: The general factors that affect the product, such as constraints, assumptions and dependencies.

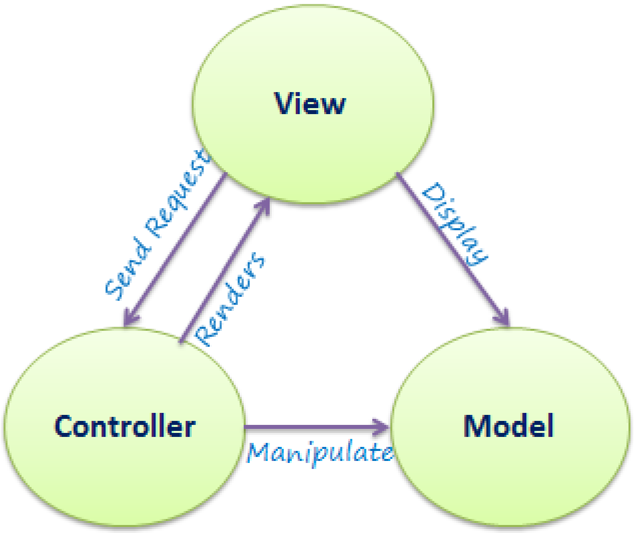


Figure 1. A Model View Controller architecture.

Product perspective

"Vehicle Rental System" web application, which consists of server containing the database using MySQL, the administrator and the clerk can access the application that is connected to the server through the cloud, and the clients that deal with the clerk to get benefit from renting vehicles.

Product functions

In this version of the SRS, major functions of the Clerk are:

- Viewing the catalog of vehicles listed or for a detailed item, he can see either the vehicle is available or not (i.e. rented or reserved).
- Updating the catalog, cancel registration or handle a return for a vehicle.
- Viewing the clients records, create new client, modify or delete a client.
- Creating a reservation or renting for a vehicle to a client.

And major functions of the Administrator are:

- View and search the history of transactions.

- Manage a vehicle record (create new, modify, delete).
- View contents of vehicle catalog and making searches.

Persistence

We provided persistence by a relational database and designed and implemented object-relational mapping from scratch.

Concurrency

The system allows at most one administrator being logged in at any time. The system allows possibly multiple clerks being logged in performing their tasks.

User characteristics

The intended users who can get benefits from the system are anyone that he or she has a valid driving license to drive a vehicle.

Constraints

He/She must have valid driving License.

The License must not be expired.

Assumptions and dependencies

Software used is Java 1.8, Tomcat 7, both operating systems Windows and Mac OS are compatible, and an internet connection.

3. Specific requirements

This section contains all requirements in detail: Functional as well as non-functional requirements (quality attributes and constraints). The quality attributes are listed according to the *ISO/IEC 25010* standard that classifies software quality in a structured set of characteristics and sub-characteristics.

3.1 External interfaces

A detailed description of all inputs into the system and all outputs from it (in terms of content and form).

View Catalog form:

List of all vehicles details showing their availability.

Inputs: press on vehicle id link or back to previous page or logout the system.

Outputs: transfer to the selected vehicle id to view in details or previous page or logging out.

Vehicle details form:

Inputs: press Back or Next.

Outputs: transfer to the previous or next page in the list.

Client Management System form:

List of all clients details.

Inputs: press Create new client link or View specific client link or Back

Outputs: transfer to Create new client form or View client form or to the previous page.

Create new client form:

Inputs: Client details (First Name, Last Name, License Number, Expiration Date, Phone Number) then press Submit or press on Back.

Outputs: new client is added or go to the previous page.

View client form:

Inputs: update to any of the client details (First Name, Last Name, License Number, Expiration Date, Phone Number) then press Update Record or press Delete or press on Back.

Outputs: client details are updated or client is deleted or go to the previous page.

Reserve or Rent form:

Inputs: press on Reserve/Rent for reserving or renting a car for a client.

Outputs: the Reserve/Rent form will be opened.

Reserve form:

Inputs: press on Reserve to reserve the selected vehicle for the particular client or press Back.

Outputs: the vehicle is reserved or go to the previous page**Sort and Filter form:**

Inputs: select to sort the vehicles by year, or select the desired type and press on filter.

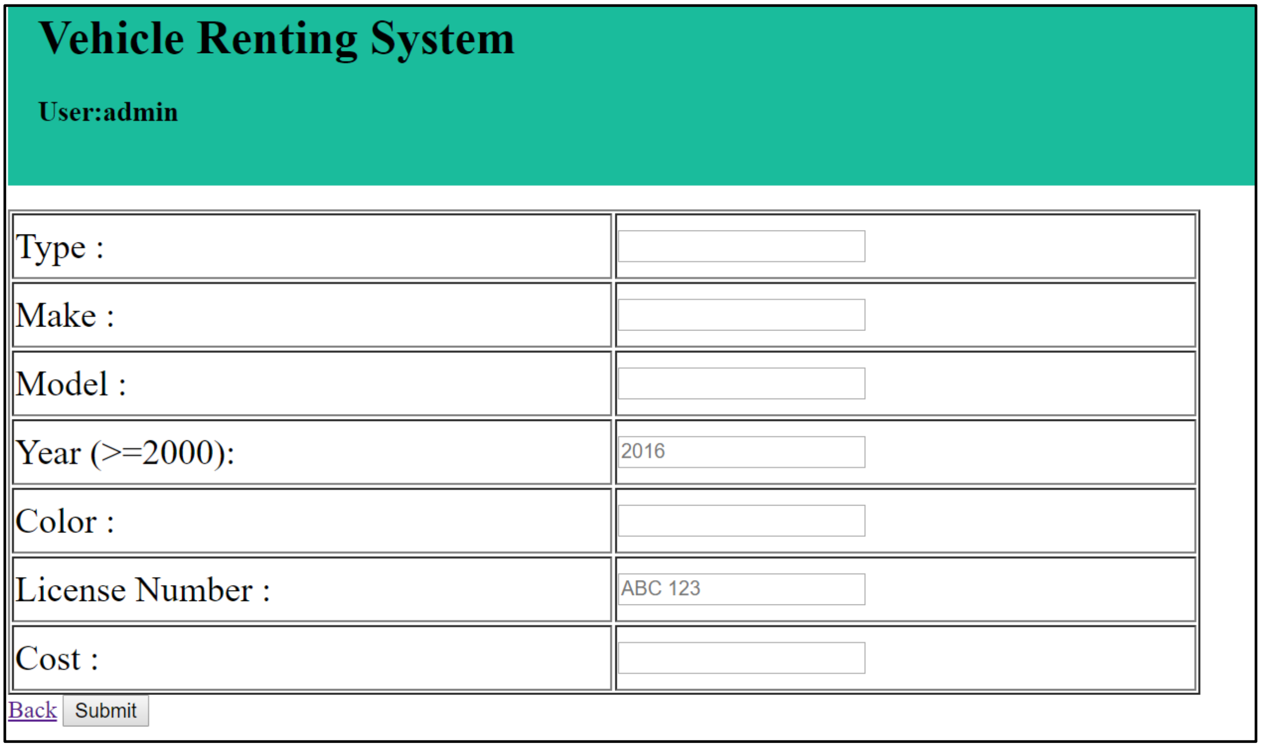
Outputs: the vehicles are sorted or filtered.

Cancel and Return form:

Inputs:Select Cancel/Return option in the welcome page to display details about vehicles and select option to cancel/return

Outputs:After selecting Cancel/Return option the vehicle should be cancelled if it was reserved or it should be returned if rented.The availability of vehicle should be changed to 'yes'.

Create new vehicle form:



Vehicle Renting System

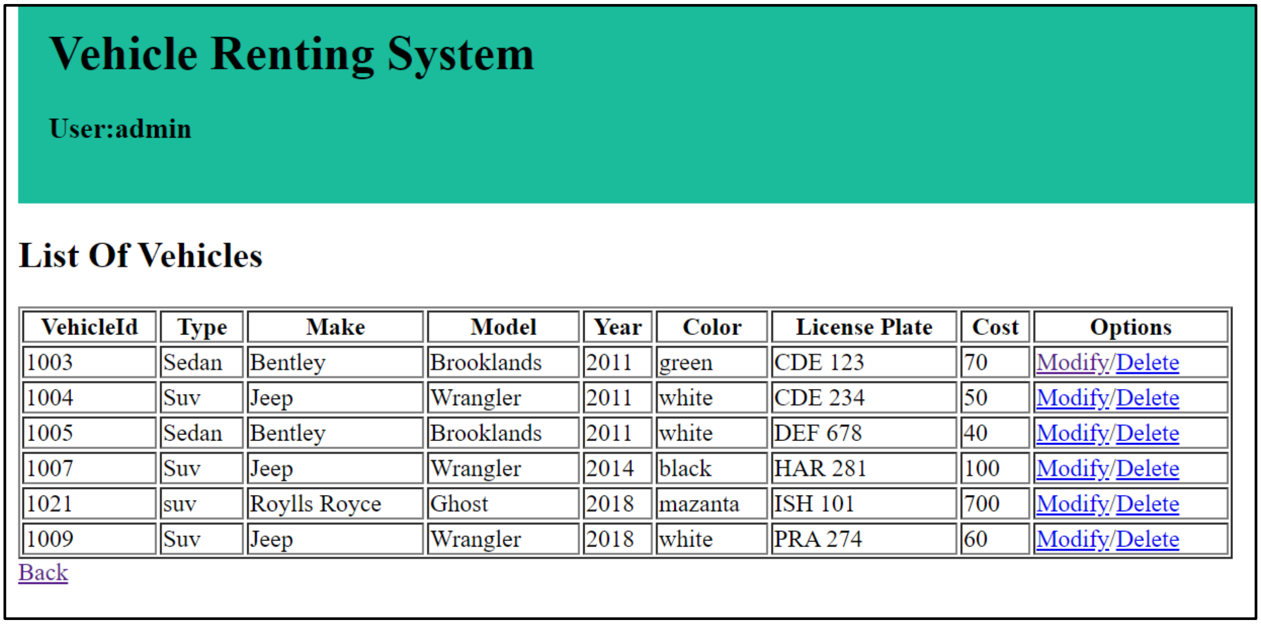
User:admin

Type :	
Make :	
Model :	
Year (>=2000):	2016
Color :	
License Number :	ABC 123
Cost :	

Inputs: The admin will input all the required fields for the specification of a new vehicle then press submit.

Outputs: A new vehicle record will be saved in the database and will appear in view catalog.

Modify/ Delete vehicle form:



Vehicle Renting System

User:admin

List Of Vehicles

VehicleId	Type	Make	Model	Year	Color	License Plate	Cost	Options
1003	Sedan	Bentley	Brooklands	2011	green	CDE 123	70	Modify/Delete
1004	Suv	Jeep	Wrangler	2011	white	CDE 234	50	Modify/Delete
1005	Sedan	Bentley	Brooklands	2011	white	DEF 678	40	Modify/Delete
1007	Suv	Jeep	Wrangler	2014	black	HAR 281	100	Modify/Delete
1021	suv	Roylls Royce	Ghost	2018	mazanta	ISH 101	700	Modify/Delete
1009	Suv	Jeep	Wrangler	2018	white	PRA 274	60	Modify/Delete

Back

Inputs: The admin will choose either to Modify a vehicle record then new form will appear to modify existing data and press submit, or the admin will choose Delete a vehicle record.

Outputs: A vehicle record update will be saved in the database or a vehicle record will be deleted from the database according to the admin choice.

3.2 Functional requirements

Functional requirements capture the intended behaviour of the system. This section contains the *Actor Goal List* and the *Use Case view*.

Actor goal list

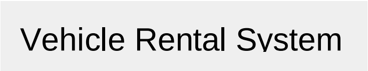
Actor	Goal
Clerk	1. To view Catalog. 2. To Filter and view Catalog. 3. To Sort and view Catalog. 4. To create a new client record. 5. To edit the existing client record. 6. To delete the client record. 7. To book a vehicle (rent or reserve a vehicle for a client).

	8. To handle a return for a rented vehicle. 9. To cancel reservation of a vehicle.

Actor	Goal
Administrator	1. View and search the history of transactions per client, per vehicle, or per due date. For example: · 'Show all vehicles currently out'. · 'When is a given vehicle due'. · 'Is a given vehicle available on a given date or over certain dates'. · 'What vehicles were due yesterday'. · 'Have they all been returned'. 2. Manage a vehicle record (create new, modify, delete). 3. View the contents of the catalog and perform searches, but cannot perform a rental, a reservation, or a return.

Use case view

The use case model is shown in Figure 3.



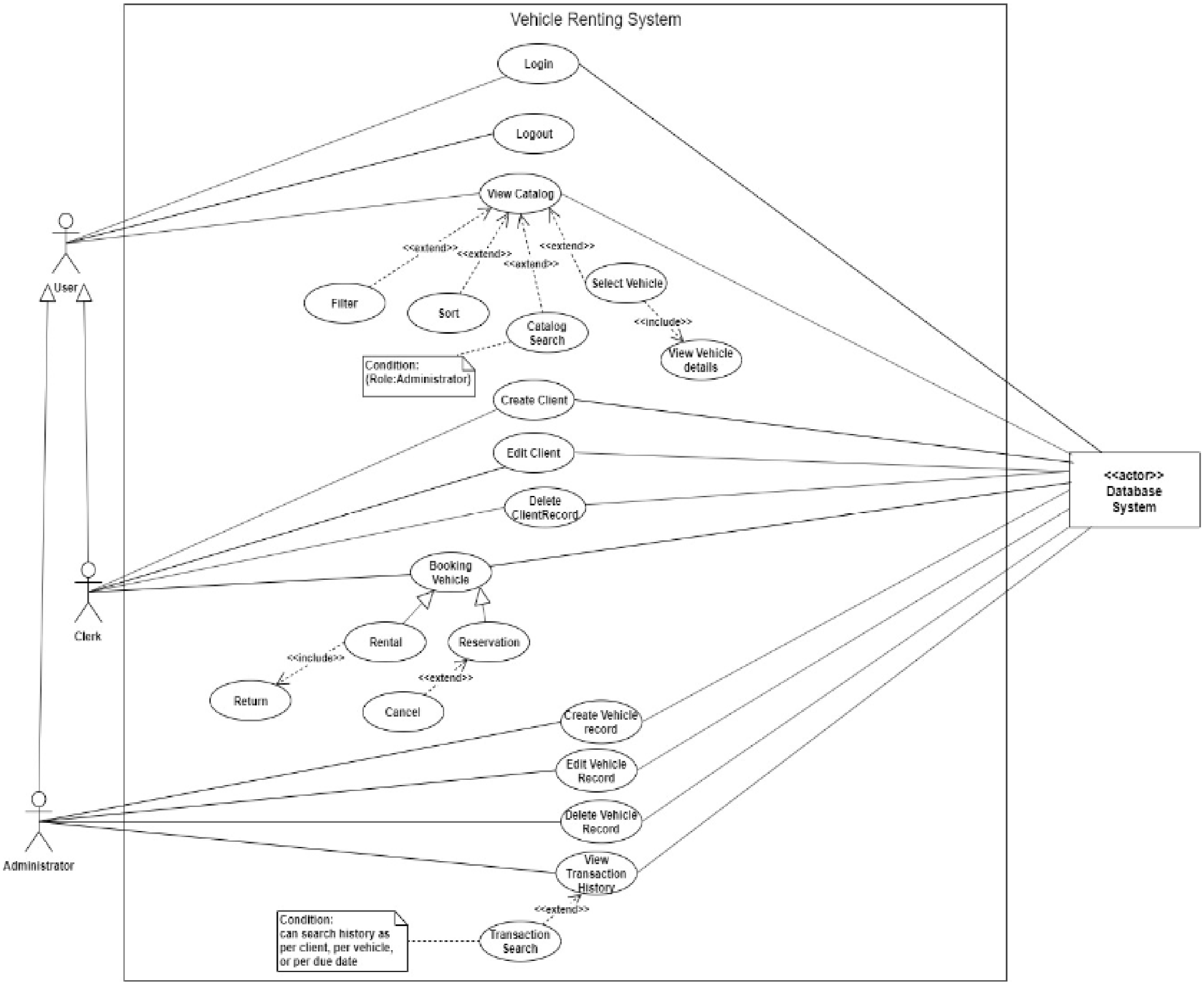


Figure 3. Use case model.

Here is the main description for the major use cases with their inclusions and extensions that are fully addressed as follows:

Use case UC1: View Catalog.

Primary Actor: Clerk.

Stakeholders and Interests: Clerk: Wants to view the vehicle Catalog and view a particular vehicle details.

Preconditions: Clerk is logged in (identified and authenticated).

Success Guarantee (Postconditions): Vehicles can be filtered, sorted and selected.

Main success scenario (or basic flow):

1. The Clerk log in to the system and clicks on View Catalog , He/She is displayed the list of vehicles.

2. The clerk can then create a result set through a selection of filtering criteria, for example the type of car (for ex SUV,Sedan) or on the basis of year.

3. The clerk may additionally choose the order by which he can view a result set: random order, or sort according to some criteria, e.g. View sorted by year.

4. From a given result set. Clerk may select an item to view in detail. In this view, the system should provide an indication of whether the vehicle is available or is rented out.

5. The clerk can subsequently proceed to the next item in detail view or go back to the (possibly filtered) initial result set view.

Extensions (or alternative flows):

1a. The clerk may Not have internet connection to connect to the database.

<u>**Use case UC2:** View Vehicle Details</u>

Primary Actor: Clerk.

Stakeholders and Interests: Clerk: Wants to view the vehicle CatLog and view a particular vehicle details.

Preconditions: Clerk is logged in (identified and authenticated).

Success Guarantee (Postconditions): he can select a vehicle from catalog to view it in detail.

Main success scenario (or basic flow):

1. The Clerk log in to the system and clicks on View Catalog , He/She is displayed the list of vehicles.

2. The clerk can then create a result set through a selection of filtering criteria, for example the type of car (for ex SUV,Sedan) or on the basis of year.

3. The clerk may additionally choose the order by which he can view a result set: random order, or sort according to some criteria, e.g. View sorted by year.

4. From a given result set. Clerk may select an item to view in detail. In this view, the system should provide an indication of whether the vehicle is available or is rented out.

5. The clerk can subsequently proceed to the next item in detail view or go back to the (possibly filtered) initial result set view.

Extensions (or alternative flows):

1a. The clerk may Not have internet connection to connect to the database.

Use case UC3: Filtering/Sorting

Primary Actor: Clerk

Stakeholders and interests:

clerk: Clerk can select the desired vehicle based on filtering and sorting condition

Preconditions: clerk is identified and authenticated.

Success Guarantee (Postconditions): Clerk will be able to search the vehicles based on filtering and sorting condition.

Main success scenario(or basic flow):

1. Clerk will log onto the system with valid credentials. Upon successful login, the system displays the view catalog.
2. Clerk search for the desired vehicle with filtering and sorting criteria by year or vehicle type.

Extensions(or alternative flows):

1a. When clerk enters invalid credentials for logging.

2b. If invalid criteria selected by clerk.

Use case UC4: Create Client.

Primary Actor: Clerk.

Stakeholders and Interests:

Clerk: Create the new Client Record.

Preconditions: Clerk is logged in (Validated and authenticated).

Success Guarantee (Postconditions): Client details will be saved in database.

Main success scenario (or basic flow):

1. Clerk start creating new client record by filling all the mandatory details.

2. If all the details are valid the record will be created in database.

Extensions (or alternative flows):

2a. If the license number and license expiry date is not valid record will not be created and error message will be displayed.

2b. The clerk may fail to connect to the database where all catalog contents are saved.

Use case UC5: Edit Details.

Primary Actor: Clerk.

Stakeholders and Interests:

Clerk: Edit the client record.

Preconditions: Clerk is logged in (Validated and authenticated).

Success Guarantee (Postconditions): Client details will be saved in database.

Main success scenario (or basic flow):

1. Clerk will edit the existing details of the client.

2. If all the details are valid the record will be created in database.

Extensions (or alternative flows):

2a. If the license expiry date is not valid record will not be created and error message will be displayed.

2b. The clerk may fail to connect to the database where all catalog contents are saved.

Use case UC6: Delete Client.

Primary Actor: Clerk.

Stakeholders and Interests:

Clerk: the clerk wants to delete the client record.

Preconditions:

1. Clerk is logged in (Validated and authenticated).
2. The client to be deleted should not have a reserved vehicle or rented one.

Success Guarantee (Postconditions): Client record will be deleted from the database.

Main success scenario (or basic flow):

1. The Clerk will successfully login to the Vehicle Renting System.
2. The Clerk will click on ClientManagement System link.
3. All clients details will be displayed from the database.
4. The Clerk will press on view option beside the client he wants to delete then choose Delete.
5. If the license number exists in the database, the corresponding record will be deleted successfully.

Extensions (or alternative flows):

1a. The Clerk may fail to login or to connect to the database. In that case error message will be displayed.

3a. The client to be deleted may have a reserved vehicle or rented vehicle (in that case, he will not be displayed).

Use case UC7: Make reservation.

Primary Actor: Clerk

Stakeholders and Interest: Clerk: Clerk should be able to create a reservation or rental by selecting the desired vehicle and providing appropriate details of the client and the reservation or rental details.

Preconditions: Clerk should be logged in. Vehicle desired should be available for the specified dates. Client should be an existing client.

Success Guarantee (Postconditions): Clerk shall create a reservation or rental record in the database by blocking the desired vehicle for the required period of time.

Main success scenario (or basic flow):

1. After successfully logging in to the system, the clerk can find the make reservation/rental option.
2. once clicked, the clerk is allowed to select one of the existing vehicles(with availability status set to YES) from the list of all vehicles.
3. The clerk can view the complete vehicle details and will be prompted to enter the start and end dates for the reservation and the cost for the reservation is automatically calculated provided the dates are valid. Reservation is made when the starting date is a future date. Rental is made when the start date is the current date.
4. The clerk should enter the license number for the client, provided the client is valid (already existing client), the client details are displayed on the same page.
5. On clicking reserve/rent, the clerk will be creating a reservation (if the start date is a future date) or rental (if the start date is the current date).
6. If successful, the vehicle's availability will be updated to NO, a success message will be printed and the form for making a new reservation is visible.

Extensions (or alternative flows):

1a. The clerk may have issues in logging in or database connectivity issue may occur.

2a. The client may not be an already existing client, which may result in the termination of making reservation. A new client should be created before continuing.

3a. The meantime between selecting the vehicle and making a reservation, some other clerk can make a reservation for the same vehicle during the same period making the vehicle unavailable. Reservation process has to be repeated for another vehicle.

Use Case UC8: Cancel/Return vehicle

Primary Actor: Clerk

Stakeholders and Interests: Clerk -Clerk should be able to cancel the reservation or return the rental by selecting the appropriate vehicle.

Preconditions: Clerk should be logged in .And there should be vehicles that are reserved so that we can cancel the reservation or return the vehicle that are rented.

Success Guarantee(PostConditions): Clerk shall cancel the reserved vehicle or return the rented vehicle and the availability of vehicle should be changed to 'yes'.

Main Success Scenarios(or basic flows):

1. After clerk successfully logging on to the system, clerk can see the cancel/return option.
2. Clerk can view the details of vehicles with option to cancel/return.
3. Select the link to cancel the reservation or return the rented vehicle, and availability will be changed to yes for the appropriate vehicle in the system.

Extensions:

1a. If clerk has invalid login credentials.

1b. There may not be reserved or rented vehicles in the list.

Use case UC9: View Transaction History.

Primary Actor: Administrator

Stakeholders and Interests:

Administrator: Should be able to view transaction history.

Preconditions:

1.Administrator is logged in (Validated and authenticated).

2.Administrator must click the view transaction history button.

Success Guarantee (Postconditions): Transaction details will be populated on the screen

Main success scenario (or basic flow):

1. When administrator clicks the view transaction history button all the transaction details will be fetched from database and displayed on the screen.

Extensions (or alternative flows):

1a.   The administrator may fail to connect to the database.In that case error message will be displayed.

Use case UC10: Transaction Search.

Primary Actor: Administrator

Stakeholders and Interests:

Administrator: Should be able to search the transaction history as per client ,per vehicle,per due date .

Preconditions:

1.Administrator is logged in (Validated and authenticated).

2.Administrator must click the view transaction history button.

3.After selecting and entering search criteria,administrator must press submit button.

Success Guarantee (Postconditions): According to search criteria transaction details will be populated on the screen.

Main success scenario (or basic flow):

1. When administrator clicks the view transaction history button all the transaction details will be fetched from database and displayed on the screen.

2.Then select the search criteria and give required information as input.Then press the submit button .

3.Required details will be displayed on the screen.

Extensions (or alternative flows):

1a. The administrator may fail to connect to the database.In that case error message will be displayed.

3a.If there is no specific entry for the search criteria in a database, data will not be displayed on the screen.

Use case UC11: Create New Vehicle Record

Primary Actor: Administrator

Stakeholders and Interests:

Administrator: Must be able to insert new vehicle record into the database satisfying the required conditions.

Preconditions:

1.Administrator is logged in (Validated and authenticated).

2.Administrator should provide valid details in the model and cost.

3.The license plate provided should be valid and should not be duplicated.

Success Guarantee (Postconditions): New vehicle record should be successfully added to the VehicleDetails table.

Main success scenario (or basic flow):

1. Administrator should click on the create new vehicle details.
2. Administrator should provide all the valid details for all the required fields.
3. New vehicle record is inserted into the table provided if the license plate is not duplicated.

Extensions (or alternative flows):

1a. The administrator may fail to connect to the database.In that case error message will be displayed.

1b. Operation will be failed if the license plate is duplicated or the cost is equal to zero.

Use case UC12: Modify Vehicle Record

Primary Actor: Administrator

Stakeholders and Interests: Administrator: Should be able to modify vehicle record into the database satisfying the required conditions.

Pre Conditions:

1.Administrator is logged in(Validated and authenticated)

2.Administrator should have access to valid vehicle record.

Success Guarantee(PostConditions): Vehicle record information is successfully updated and stored.

Main Success Scenario(or basic flow):

1.Administrator will successfully login to the system

2.Administrator choose Vehicle Record management where Modify & Delete options are displayed.

3. When Modify is selected, based on the selected vehicle details, the vehicle record is modified.

4.Modified vehicle record information is successfully updated.

Extensions:

1a. Administrator fails to login if invalid credentials are provided.

2a.Vehicle record information is not updated successfully if desired vehicle details are invalid.

Use case UC13: Delete Vehicle

Primary Actor: Administrator.

Stakeholders and Interests:

Administrator: the administrator wants to delete the vehicle record.

Preconditions:

1. Administrator is logged in (Validated and authenticated) successfully.
2. The vehicle desired to be deleted must be available (not rented or reserved to a client).

Success Guarantee (Postconditions): Vehicle record will be deleted from the database and from the view catalog page.

Main success scenario (or basic flow):

2. The Administrator will successfully login to the Vehicle Renting System.
3. The Administrator will choose Vehicle Record Management link.
4. All vehicle records (not rented or reserved to a client) will be displayed to either modify or delete.
5. The Admin will press on delete corresponding to the vehicle to be deleted.
6. Accordingly the vehicle with specific license plate will be deleted from the database.
7. The view will be changed to appear without the deleted vehicle.

Extensions (or alternative flows):

1a. The administrator may fail to connect to the database. In that case error message will be displayed.

3a. The vehicle to be deleted may have been already rented or reserved to a client so it will not be displayed in the list.

Use case UC14: Catalog Search

Primary Actor: Administrator.

Stakeholders and Interests: Administrator: Wants to search a catalog on the basis of license plate number.

Preconditions: Administrator is logged in (identified and authenticated).

Success Guarantee (Postconditions): he can search a catalog using license plate number.

Main success scenario (or basic flow):

1. The Admin log in to the system and clicks on View Catalog , He/She is displayed the list of vehicles.

2. The Admin can then create a result set through a selection of filtering criteria, for example the type of car (for ex SUV,Sedan) or on the basis of year.

3.He\she can search for a particular vehicle using license plate number.

Extensions(or alternative flows):

1a. When Admin enters invalid credentials for logging.

2b. If invalid search criteria selected by Admin.

3.3 Non-functional requirements:

Performance efficiency

The system performance in terms of response time has been tested and shows a quick response to any request done from the clerk and admin end as per respective functionality.

Compatibility

The web application using JavaSpring as a framework is compatible with all kinds of browsers.

Usability

The Vehicle Rental System has been tested for usability by providing clear messages in case of any invalid input by the user, taking all potential invalid inputs, also introducing the application for non-technical people for using the system and getting their feedback.

Reliability

The web application is ensured to be reliable by using try and catch mechanism for the code part, all possible error exceptions are handled to check that all inputs are in the correct formats.

Adding some test cases for catalog, client management ,admin transaction functionality and ran them successfully in test suite to ensure correctness of methods.

Security

As the system has a login form to verify who is logging in ,if the clerk and admin has the authorization for using the system or not, it only allows the registered predefined clerks and admin to use the application and redirects to clerk page if clerk tries to login and redirects to admin page if admin tries to login , so no malicious or theft could occur.

Maintainability

The application is flexible for any critical updates, by making low coupling and high cohesion between classes, methods and attributes, so to reduce dependencies where an update occurs in one class does not affect the other functionalities.

Portability

The system is designed to be easily moved from one computing environment to another, such as browsers and different operating systems.

Design constraints

Following iteration process model by implementing subsequent functionality (admin functionality) from the overall functionality of the system, following MVC Java Spring framework in implementing the web system, using UML online tools such as drawio and UML paradigm for designing the system architecture.

4. Analysis Models

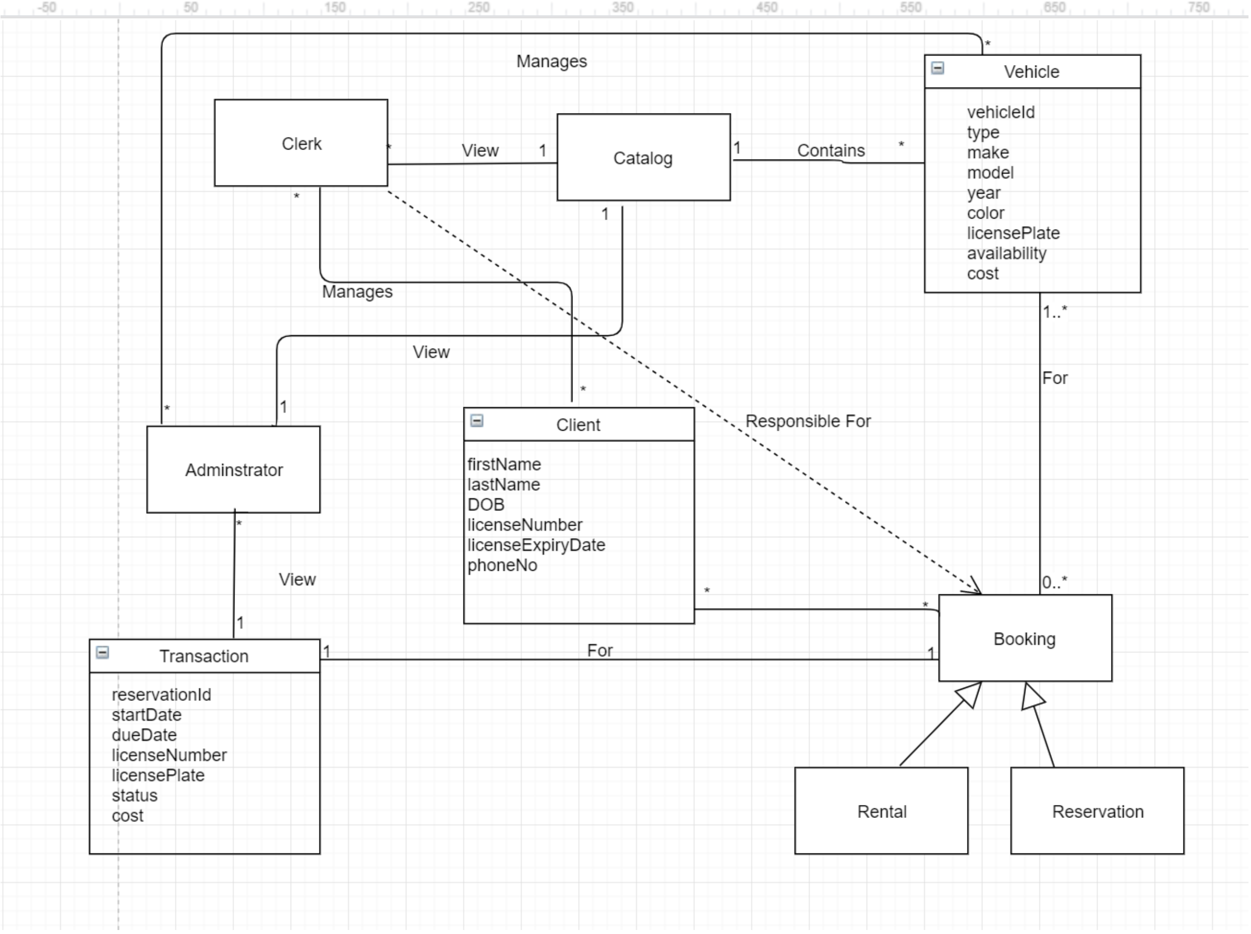
List all analysis models used in developing specific requirements previously given in this SRS. Each model should include an introduction and a narrative description. Furthermore, each model should be traceable the SRS’s requirements.

Illustrate (system) ***UML sequence diagrams*** (one for each **critical** scenario), identify system operations and describe operation contracts, **one per critical system operation**.

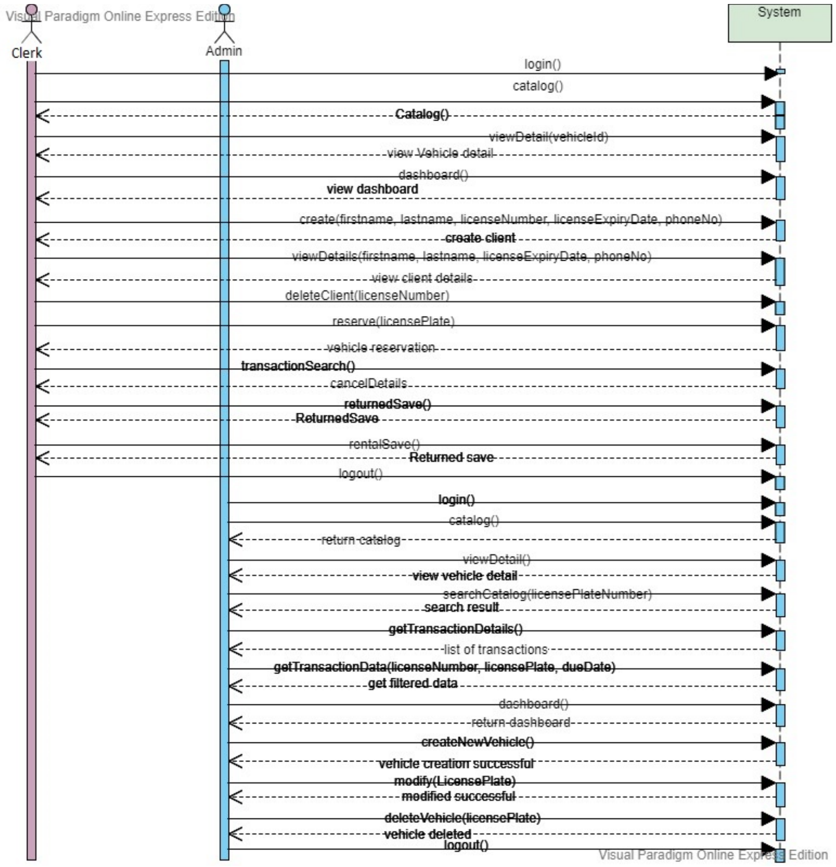
You may also use ***UML state diagrams*** to describe **critical use cases**, one state diagram per use case.

Finally, create a **UML conceptual class diagram ("domain model")** for the system. If the model gets too large, you can use **UML package diagrams** to provide logical groupings for the model.

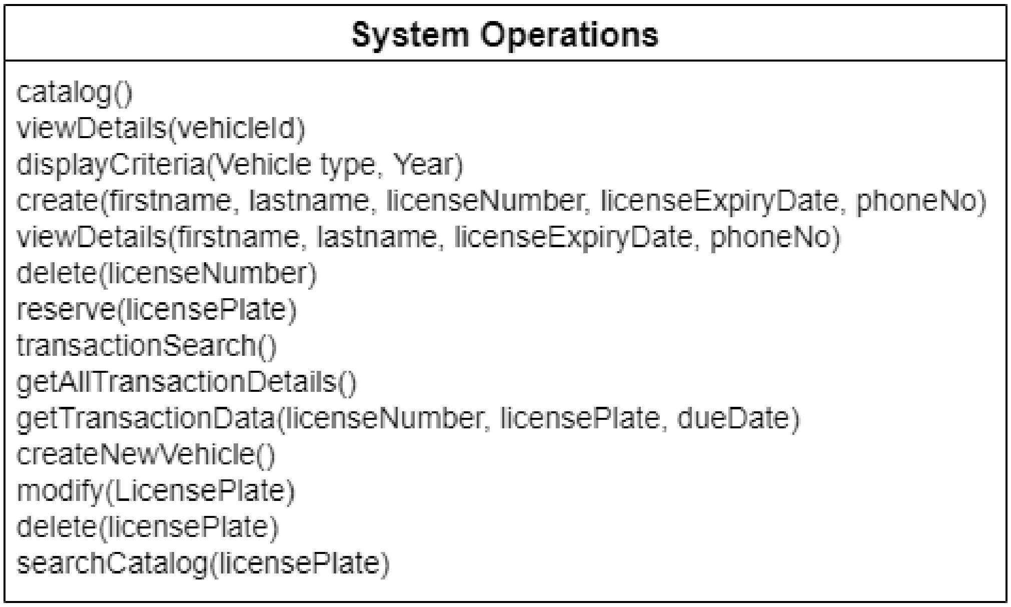
Domain Model : domain model is a conceptual model of the domain that incorporates both behaviour and data



System sequence diagram:



System Diagram



System Operations

catalog()
viewDetails(vehicleId)
displayCriteria(Vehicle type, Year)
create(firstname, lastname, licenseNumber, licenseExpiryDate, phoneNo)
viewDetails(firstname, lastname, licenseExpiryDate, phoneNo)
delete(licenseNumber)
reserve(licensePlate)
transactionSearch()
getAllTransactionDetails()
getTransactionData(licenseNumber, licensePlate, dueDate)
createNewVehicle()
modify(LicensePlate)
delete(licensePlate)
searchCatalog(licensePlate)

Contract 1:

CO1: catalog.

Operation: catalog().

Cross References: UC1 (View Catalog)

Preconditions:

1. The Clerk must successfully login to the System.

Postcondition:

1. The instance of List<Catalog> is created.
2. instance will be associated with CatalogMapper.

Contract 2:

CO2: ViewDetails.

Operation: ViewDetails(vehicleId)

Cross References: UC2 (View Vehicle Detail)

Preconditions:

The Clerk must successfully login to the System.

Postcondition:

1. The instance of List<Catalog> instance is created.
2. instance will be associated with CatalogMapper.

3.instance was associated with httpsession.

Contract 3:

C03: Sorting/Filtering

Operation: Display criteria(Vehicle type, Year)

Cross References: UC3(Filtering/ Sorting)

Preconditions:

1. The clerk must successfully login to the system.

Postcondition:

1. The instance of List<Catalog> instance is created.
2. Instance was associated with CatalogMapper.
3. The system displays the desired criteria based on the selected filtering and sorting condition.

Contract 4:

CO4: createclient.

Operation: create(firstname, lastname, licenseNumber, licenseExpiryDate, phoneNo)

Cross References: UC4 (create Client)

Preconditions:

1. The Clerk must successfully login to the System.
2. The license number and license expiry date of the client should be valid.

Postcondition:

1. The instance of client will be created.
2. The instance will be associated to clientManagementMapper
3.Then instance will be modified.

4.Instance will be persisted in database.

Contract 5:

CO5: viewDetails()

Operation: viewDetails(firstname, lastname, licenseExpiryDate, phoneNo)

Cross References: UC5 (Edit Client)

Preconditions:

1. The Clerk must successfully login to the System.

2. The modified details of the client should be valid.

Postcondition:

1. The instance of client will be created.

2. The instance will be associated to clientManagementMapper

3.Then instance will be updated.

4.Instance will be persisted in database.

Contract 6:

CO6: Delete Client.

Operation: delete (licenseNumber)

Cross References: UC6 (Delete Client)

Preconditions:

1. The Clerk must successfully login to the System.

2. The Clerk will click on ClientManagement System link.

3. All clients details will be displayed from the database.

4. The client should be present in the system (not renting or reserving a vehicle).

5. The Clerk will press on view option beside the client he wants to delete then choose Delete.

Postcondition:

1. The instance of client model will be created.

2. The instance is associated with ClientManagementMapper

3. client record will be deleted from the database.

Contract 7:

CO 7: makeReservatioOrRental

Operation: reserve(licensePlate)

Cross References: UC7 (makeReservationOrRental)

Preconditions:

1. The Clerk must successfully login to the System.

2. Preferred vehicle should be available

3.The client should be an existing client.

Postcondition:

1. The reservation instance will be created.

2 . Instance created is associated with VehicleReservationMapper.

3. The reservation instance is inserted to the database.

4. The related vehicle instance is fetched from database.

5. The status of the fetched vehicle must be modified to "NO".

6. The modified instance is updated in the database.

Contract 8:

CO8 : Cancel/Return

Operation: transactionsearch()

Cross References: UC8 (Cancel/Return vehicle)

Preconditions:

1.The clerk must be able to login

2.Vehicles should be available for cancel or return

3.client should be present in system

Postcondition:

1. Instance of cancelReturn model will be created.

2.Instance will be associated with rowMapper.

3.Instance will be modified.

4.instance will be persisted in database

Contract 9:

CO9: getAllTransactionDetails

Operation: getAllTransactionDetails()

Cross References: UC9 (view Transaction History)

Preconditions:

1. The Administrator must successfully login to the System.

2. Administrator must click view Transaction history link.

Postcondition:

1. The list of instances of Transactions will be created.

2. Instance will be associated with rowMapper.

3. Instance will be modified.

4. All the transaction history records will be fetched from the database.

5. The instances attributes will updated with data from database.

Contract 10:

C10: getTransactionData

Operation: getTransactionData(licenseNumber,licensePlate,dueDate)

Cross References: UC10 (Transaction Search)

Preconditions:

1. The Administrator must successfully login to the System.

2. Administrator must click view Transaction history link.

3.Administrator must select one of the selection criteria and give required information as input.

4.After following above steps,submit button should be pressed.

Postcondition:

1. The list of instances of Transactions will be created.

2.Then list of instances will be associated with transactionHistoryMapper.

3.All the transaction history records with respect to search criteria will be fetched from the database.

4.The instances attributes will updated with data from database.

5.Then data will be displayed on the screen.

Contract 11:

CO12: Create vehicle record

Operation: Createvehicle record

Cross Reference: UC11(Modify vehicle record)

Preconditions:

1. Administrator must successfully login to the system.

2. Rental Cost per day should be more than 0.

3. License plate should be valid and not duplicated

Postcondition:

1. Vehicle instance should be created.

2. It should be associated to the VehicleRecordMapper.

3. Instance created is inserted into the database.

Contract 12:

CO12: Modify vehicle record

Operation: Modify vehicle record

Cross Reference: UC12(Modify vehicle record)

Preconditions:

1. Administrator must successfully login to the system

Postcondition:

1. Instance of vehicle will be created.

2.Instance will be associated with rowMapper.

3.Instance will be modified.

4.instance will be persisted in database

Contract 13:

CO13: Delete Vehicle.

Operation: delete (licensePlate).

Cross References: UC13(Delete Vehicle).

Preconditions:

1. The Administrator will successfully login to the Vehicle Renting System.

2. The Administrator will choose Vehicle Record Management link.
3. All vehicle records (which is not reserved or rented to a client)will be displayed to either modify or delete.
4. The Admin will press on delete corresponding to the vehicle to be deleted.

Postcondition:

1. The instance of vehicle model will be created.
2. The instance is associated with VehicleRecordMapper.
3. Vehicle record will be deleted from the database.

Contract 14:

CO14: catalog Search.

Operation: searchcatalog(license plate number).

Cross References: UC14(catalog Search)

Preconditions:

1. The Administrator must successfully login to the System.

Postcondition:

1. The instance of List<Catalog> instance is created.

2. instance was associated with CatalogMapper..

3.instance was associated with httpsession.

4.The system displays the desired result on the basis of search criteria.