CSE 222 - DATA STRUCTURES PROJECT

GROUP MEMBERS

STUDENT NUMBER	NAME
171044004	ANIL MERT ULAŞAN
1801042093	MUSTAFA HALİL ŞENOL
151044032	ARAS TOPUZ
1901042654	FATİH DOĞAÇ
1901042670	BUKET GENÇAYDIN
161044041	MERTCAN ELÇİ
171044076	ŞEYMA NUR CANBAZ
1901042692	AHMET TUĞKAN AYHAN
1950044029	SERDAR KARAKURT

Here is our github link if you like to see our project’s unfinished version:

PROBLEM DEFINITION

Not all people have their own vehicles, and when they want to drive for travel, hang out, etc. They have to own a vehicle or ask from somebody they know. With this project, we want to fill that need, offer people to rent a vehicle that fits to their demand, for a time that they decide.

The main goal of the project is to help people who doesn’t want to buy a vehicle, but wants to use it for a time.

Customers through the system;

- They can see all the branches and choose the closest branch to themselves.

- If the user will use the renting firm for the first time, the customer will be registered in the system by the sales manager. Driver’s license, vehicle usage history etc. Will be requested from customer for register to the system.

- If the customer is used the system before, customer will be able to apply his/her request with certain advantages.

- The advantages(benefits) will be depend on rent history, involved accident or not etc.

After they rent a vehicle, customers don’t have to think about delivering it to the branch that they got it from. They can deliver it to another branch which is close the them at that moment, or call the branch to send them a transportation personnel to pick the vehicle up.

While implementing the features we mentioned, we will encounter a few problems.

- Dynamic storage of created programs and users..

- Classification of users.

- Protection of the data stored when the system is turned off and on again, or whenever the user enters and exits.

USER OF THE SYSTEM

Customer

- Customers can log in or log out from the system with their account informations by the sales manager.
- Customers can get informations about available cars by the sales manager.
- Customers will take the vehicles from the branch during the leasing process and leave them to the any branch when the lease is completed. If the customer can not leave it to the branch, customer will inform the sales manager.
- Customers can search vehicles by their rating. (added)
- Customers can give a score to a vehicle they’ve rented (added).

Admin

- Admin can add/remove vehicles to/from branches. (different types of vehicles)
- Admin can add/remove branches. (vehicle rental & service branch)
- Admin can add/remove sales managers, technicians, transportation personnel.
- Budget operations will be done by the admin.

Sales Manager

- Sales managers can rent vehicles to customers.
- Sales managers can add/remove customers.
- Sales managers can check/update customers informations.
- Sales managers can contact with transportation personnel. (when rented vehicle brought back)

Transportation Personnel

- Tranports the maintained vehicle to the rental branch. (when maintenance done)
- Tranports the vehicle for the customer from another rental branch. (when there is no specific vehicle)
- Tranports the vehicle from rental branch to technician branch. (for the maintenance)
- Tranports the vehicle one rental branch to another branch. (when there is no specific vehicle)

Technician

- Technicians can check vehicle situations.
- Technicians can repair the damaged vehicles.
- Vehicle maintenance will be done by the technicians.

REQUIREMENTS

Functional System Requirement

Admin controls and manipulates the system by adding/removing branches and employees, handling the storage by adding and removing vehicles, dealing with company budget.

Customers can search vehicles of different types, colors, price ranges. They can read reviews of a specific vehicle from previous users. Once they decide, they can request a rental vehicle for their preferred vehicle.

Customers can rent a vehicle from any branch they want as long as the desired vehicle is available in that branch and return the vehicle any branch they want. They can check if there is a branch in their city, if not system will list cities with a branch and give contact information of customer support.

Customers need to return their rented vehicle without any damage before deadline or they will lose customer points based on the damage the vehicle received and how late the customer return the rented vehicle.

Customers with low points won't be able receive some(or any) services from system.

Customers can leave a review and a score for their received services.

Sales managers can add/remove customer to/from system and reach information of them. They can handle rent requests based upon the information of the customer they can access availability of the vehicles.

Sales managers can check deadlines, remaining time of the rented vehicles.

After receiving back the rented vehicle, they can send the vehicle to technical personnel for checkup.

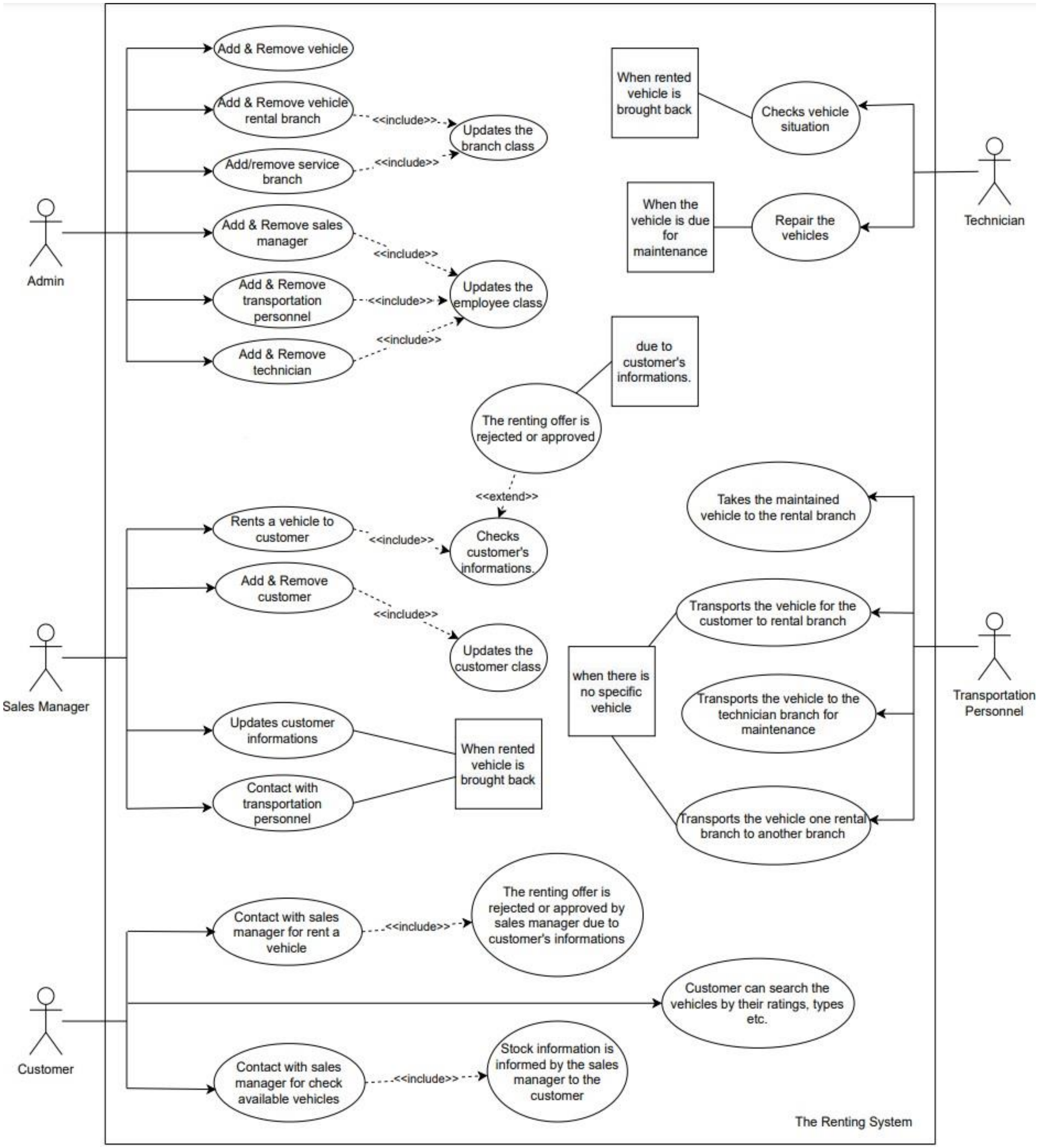
Technical Personnel checks the condition of the vehicles and make a report about the problems. After that they fix problems and send back the vehicles to the storage.

Transportation Personnel checks vehicles that requires transportation between branches and handles the transportation.

Non-Functional System Requirement

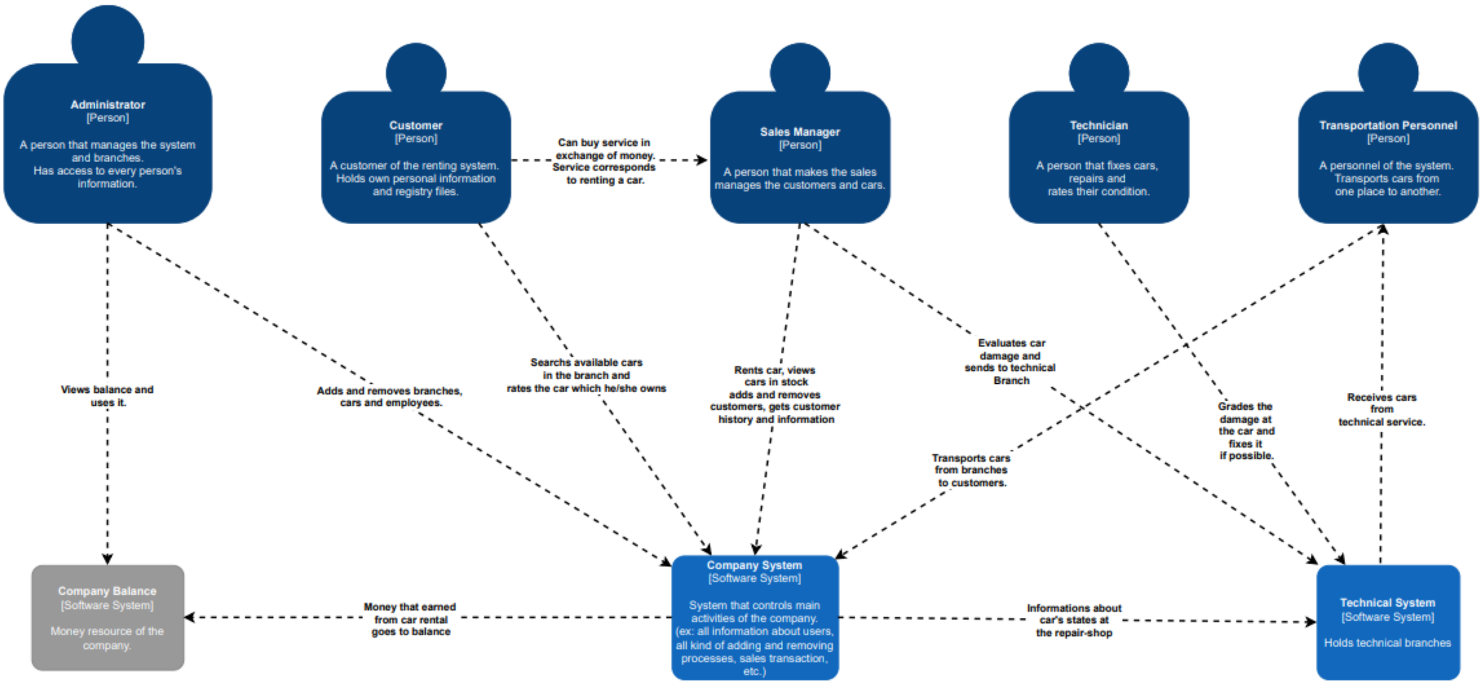
- Hardware should be able to run at least Java SE8.
- System can operate at only computers.
- System should be able to provide information on user’s request in less than one second in optimal conditions.
- All Personnel must take a small course before using this app.
- System should be easy to use, user experience should be simple enough so that even the elderly personnel are able to catch up fast.
- Reinstalling the system should be done by professionals in case of failure.
- Computer should have minimum 2GB space for the software but than we will have huge data so minimum 8 GB is enough
- System have to have a enough data for the help all customers to they should move as short as possible to reach services.
- System should have comments for the update.
- System should be open7/24
- All customer have to use system easily.

USE CASE DIAGRAM


Admin
Add & Remove vehicle
Add & Remove vehicle rental branch
Add/remove service branch
Add & Remove sales manager
Add & Remove transportation personnel
Add & Remove technician
<<include>>
Updates the branch class
Updates the employee class
When rented vehicle is brought back
Checks vehicle situation
When the vehicle is due for maintenance
Repair the vehicles
Technician
due to customer's informations.
The renting offer is rejected or approved
<<extend>>
Sales Manager
Rents a vehicle to customer
Checks customer's informations.
Add & Remove customer
Updates the customer class
Updates customer informations
Contact with transportation personnel
When rented vehicle is brought back
Takes the maintained vehicle to the rental branch
Transports the vehicle for the customer to rental branch
when there is no specific vehicle
Transports the vehicle to the technician branch for maintenance
Transports the vehicle one rental branch to another branch
Transportation Personnel
Customer
Contact with sales manager for rent a vehicle
The renting offer is rejected or approved by sales manager due to customer's informations
Customer can search the vehicles by their ratings, types etc.
Contact with sales manager for check available vehicles
Stock information is informed by the sales manager to the customer
The Renting System

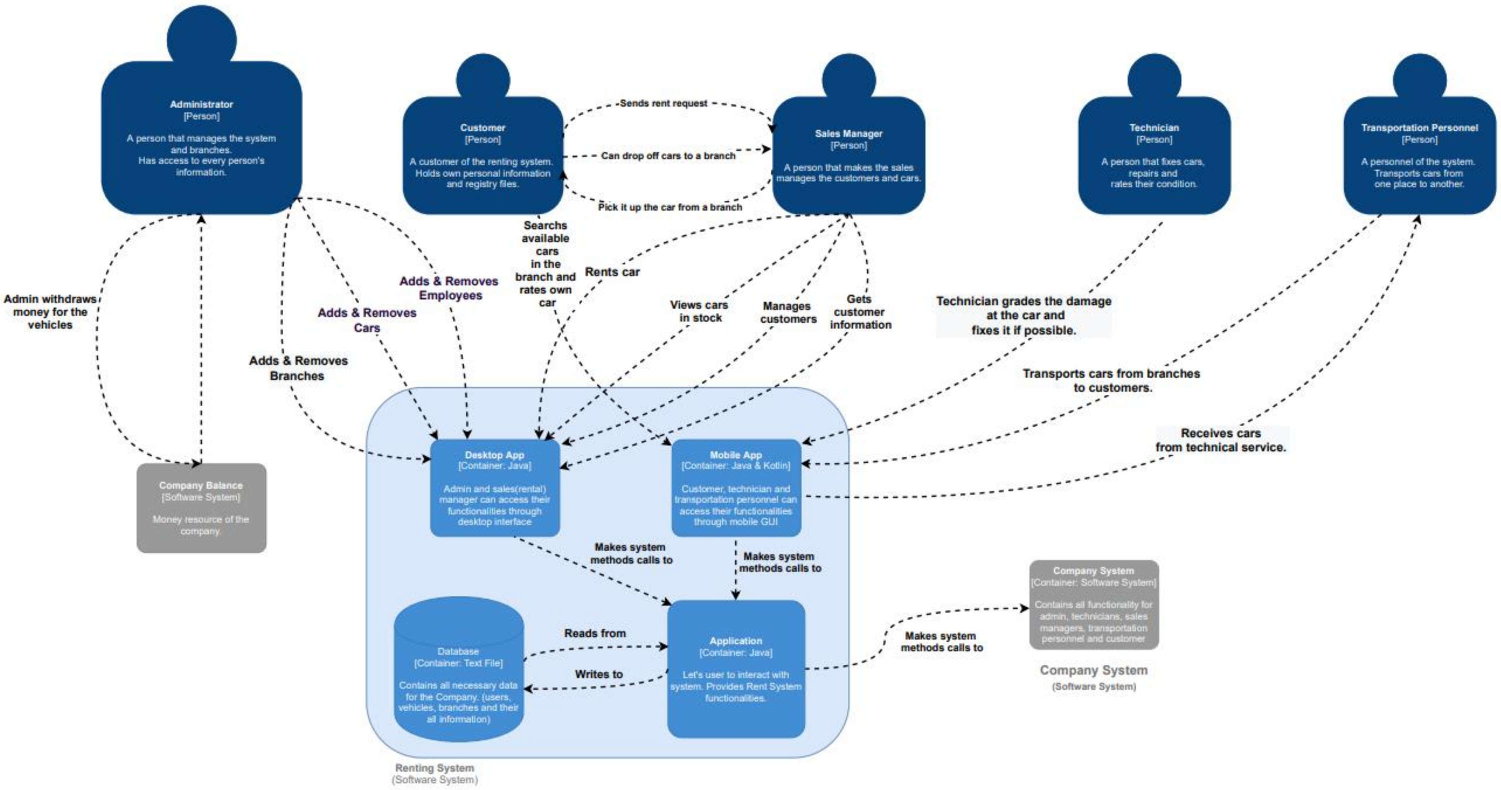
C4 MODELS

C1) System Context Diagram



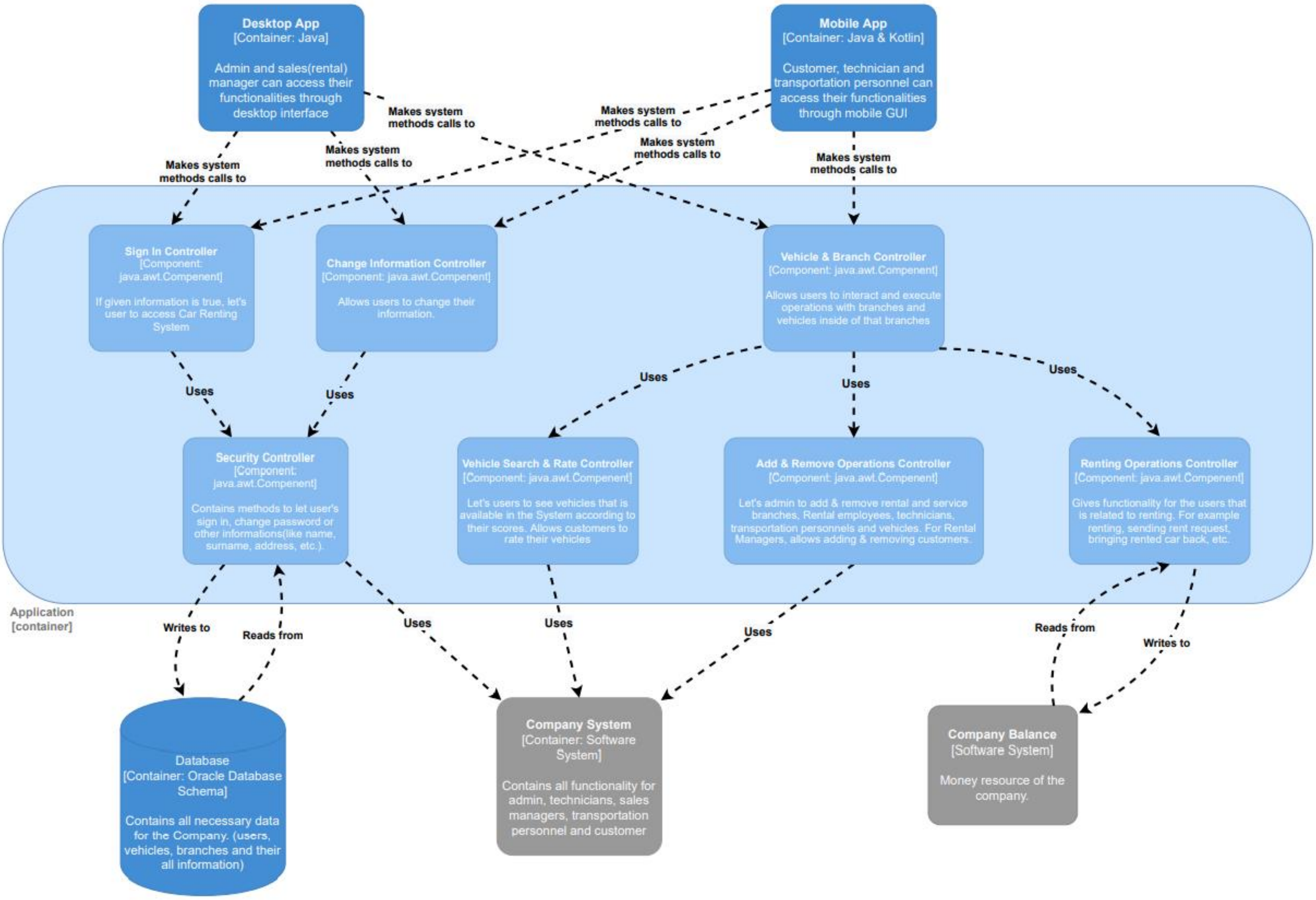
Administrator
[Person]
A person that manages the system and branches.
Has access to every person's information.
Customer
[Person]
A customer of the renting system. Holds own personal information and registry files.
Can buy service in exchange of money. Service corresponds to renting a car.
Sales Manager
[Person]
A person that makes the sales manages the customers and cars.
Technician
[Person]
A person that fixes cars, repairs and rates their condition.
Transportation Personnel
[Person]
A personnel of the system. Transports cars from one place to another.
Views balance and uses it.
Adds and removes branches, cars and employees.
Searchs available cars in the branch and rates the car which he/she owns
Rents car, views cars in stock adds and removes customers, gets customer history and information
Evaluates car damage and sends to technical Branch
Grades the damage at the car and fixes it if possible.
Receives cars from technical service.
Transports cars from branches to customers.
Company Balance
[Software System]
Money resource of the company.
Money that earned from car rental goes to balance
Company System
[Software System]
System that controls main activities of the company. (ex: all information about users, all kind of adding and removing processes, sales transaction, etc.)
Informations about car's states at the repair-shop
Technical System
[Software System]
Holds technical branches

C2) Container Diagram



Administrator
[Person]
A person that manages the system and branches.
Has access to every person's information.
Customer
[Person]
A customer of the renting system. Holds own personal information and registry files.
Sends rent request
Can drop off cars to a branch
Pick it up the car from a branch
Sales Manager
[Person]
A person that makes the sales manages the customers and cars.
Technician
[Person]
A person that fixes cars, repairs and rates their condition.
Transportation Personnel
[Person]
A personnel of the system. Transports cars from one place to another.
Admin withdraws money for the vehicles
Adds & Removes Branches
Adds & Removes Cars
Adds & Removes Employees
Searchs available cars in the branch and rates own car
Rents car
Views cars in stock
Manages customers
Gets customer information
Technician grades the damage at the car and fixes it if possible.
Transports cars from branches to customers.
Receives cars from technical service.
Company Balance
[Software System]
Money resource of the company.
Desktop App
[Container: Java]
Admin and sales(rental) manager can access their functionalities through desktop interface
Mobile App
[Container: Java & Kotlin]
Customer, technician and transportation personnel can access their functionalities through mobile GUI
Makes system methods calls to
Makes system methods calls to
Database
[Container: Text File]
Contains all necessary data for the Company. (users, vehicles, branches and their all information)
Reads from
Writes to
Application
[Container: Java]
Let's user to interact with system. Provides Rent System functionalities.
Makes system methods calls to
Company System
[Container: Software System]
Contains all functionality for admin, technicians, sales managers, transportation personnel and customer
Company System
(Software System)
Renting System
(Software System)

C3) Component Diagram


Desktop App
[Container: Java]
Admin and sales(rental) manager can access their functionalities through desktop interface
Mobile App
[Container: Java & Kotlin]
Customer, technician and transportation personnel can access their functionalities through mobile GUI
Makes system methods calls to
Makes system methods calls to
Makes system methods calls to
Makes system methods calls to
Makes system methods calls to
Makes system methods calls to
Sign In Controller
[Component: java.awt.Compenent]
If given information is true, let's user to access Car Renting System
Change Information Controller
[Component: java.awt.Compenent]
Allows users to change their information.
Vehicle & Branch Controller
[Component: java.awt.Compenent]
Allows users to interact and execute operations with branches and vehicles inside of that branches
Uses
Uses
Uses
Uses
Uses
Security Controller
[Component: java.awt.Compenent]
Contains methods to let user's sign in, change password or other informations(like name, surname, address, etc.).
Vehicle Search & Rate Controller
[Component: java.awt.Compenent]
Let's users to see vehicles that is available in the System according to their scores. Allows customers to rate their vehicles
Add & Remove Operations Controller
[Component: java.awt.Compenent]
Let's admin to add & remove rental and service branches, Rental employees, technicians, transportation personnels and vehicles. For Rental Managers, allows adding & removing customers.
Renting Operations Controller
[Component: java.awt.Compenent]
Gives functionality for the users that is related to renting. For example renting, sending rent request, bringing rented car back, etc.
Application
[container]
Writes to
Reads from
Uses
Uses
Uses
Reads from
Writes to
Database
[Container: Oracle Database Schema]
Contains all necessary data for the Company. (users, vehicles, branches and their all information)
Company System
[Container: Software System]
Contains all functionality for admin, technicians, sales managers, transportation personnel and customer
Company Balance
[Software System]
Money resource of the company.

Changed things in C4

C1(System Context Diagram)

---

* Company Balance no longer holds salaries

* Renting system name changed into Company System and merged with technical system(Still showed them separete in C1 for better understanding)

* Customers now can search for vehicles and leave a score for the vehicles they've rented.

* Sales manager name is also used as Rental manager

* user : Administration --> Administrator

C2(Container Diagram)

---

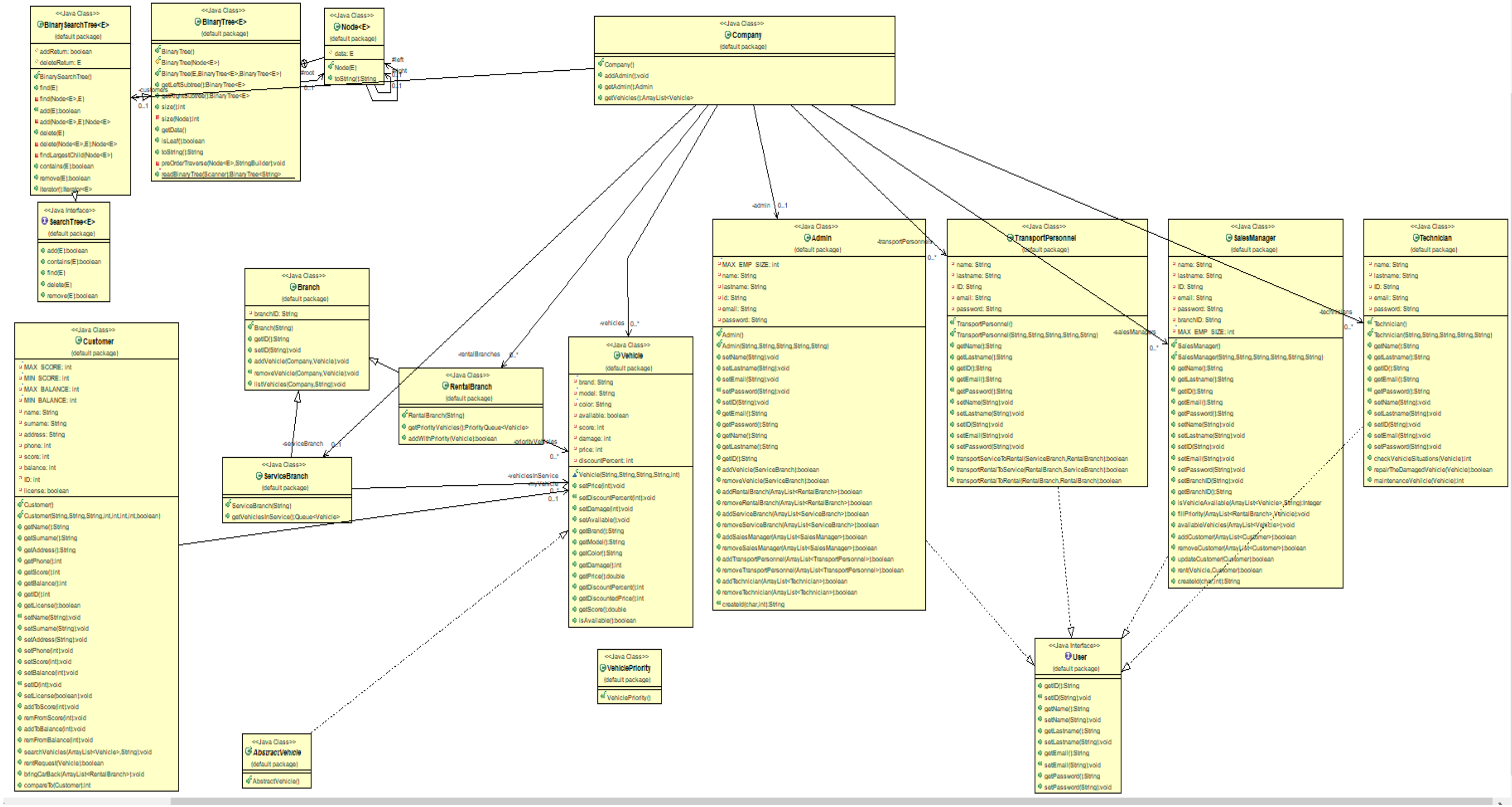
* Administrator and sales(rental) manager uses their functionality over Desktop interface

* Customer, technician and transportation personnel uses their functionality over Mobile gui

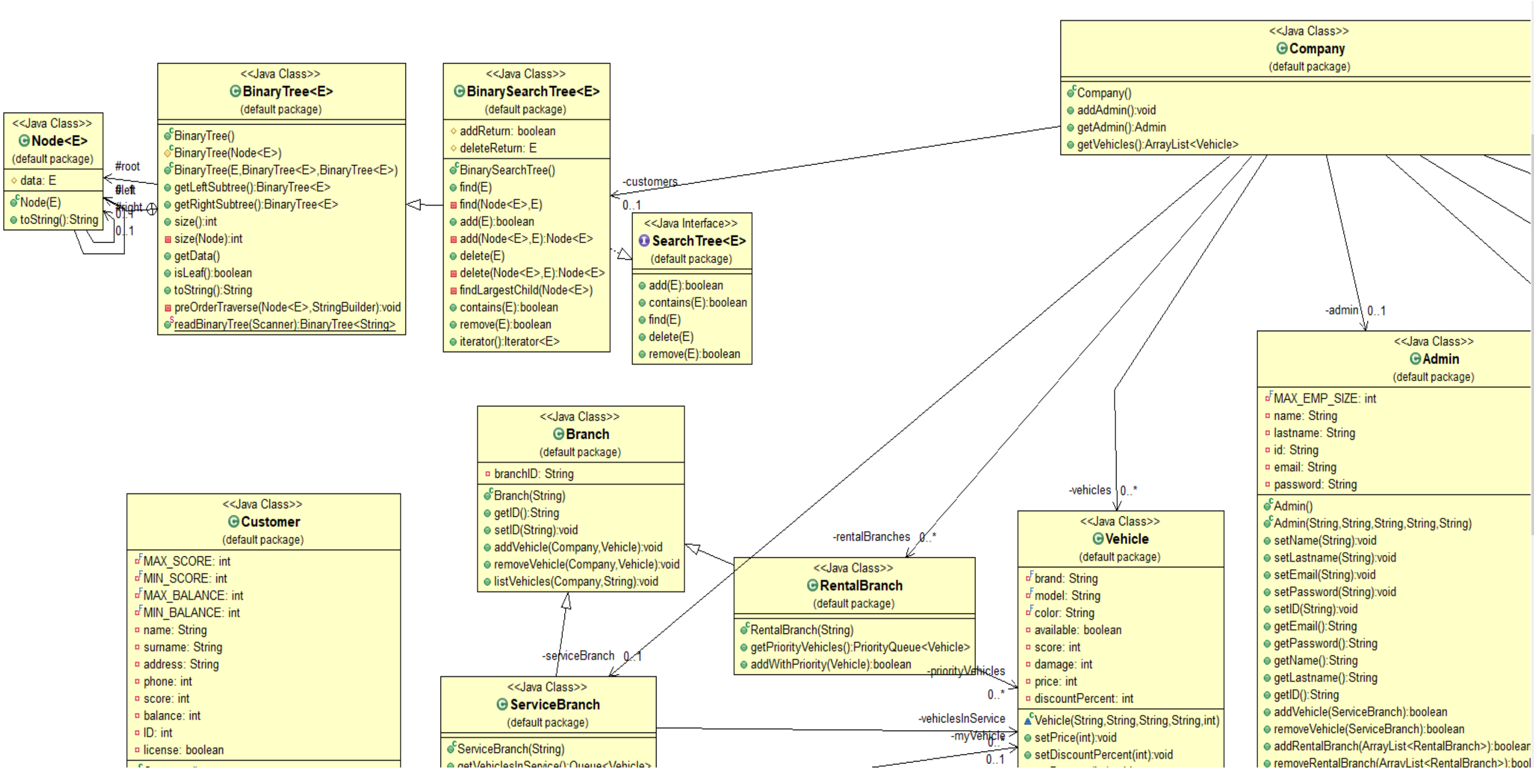
* Admins no longer give salaries to the employees.

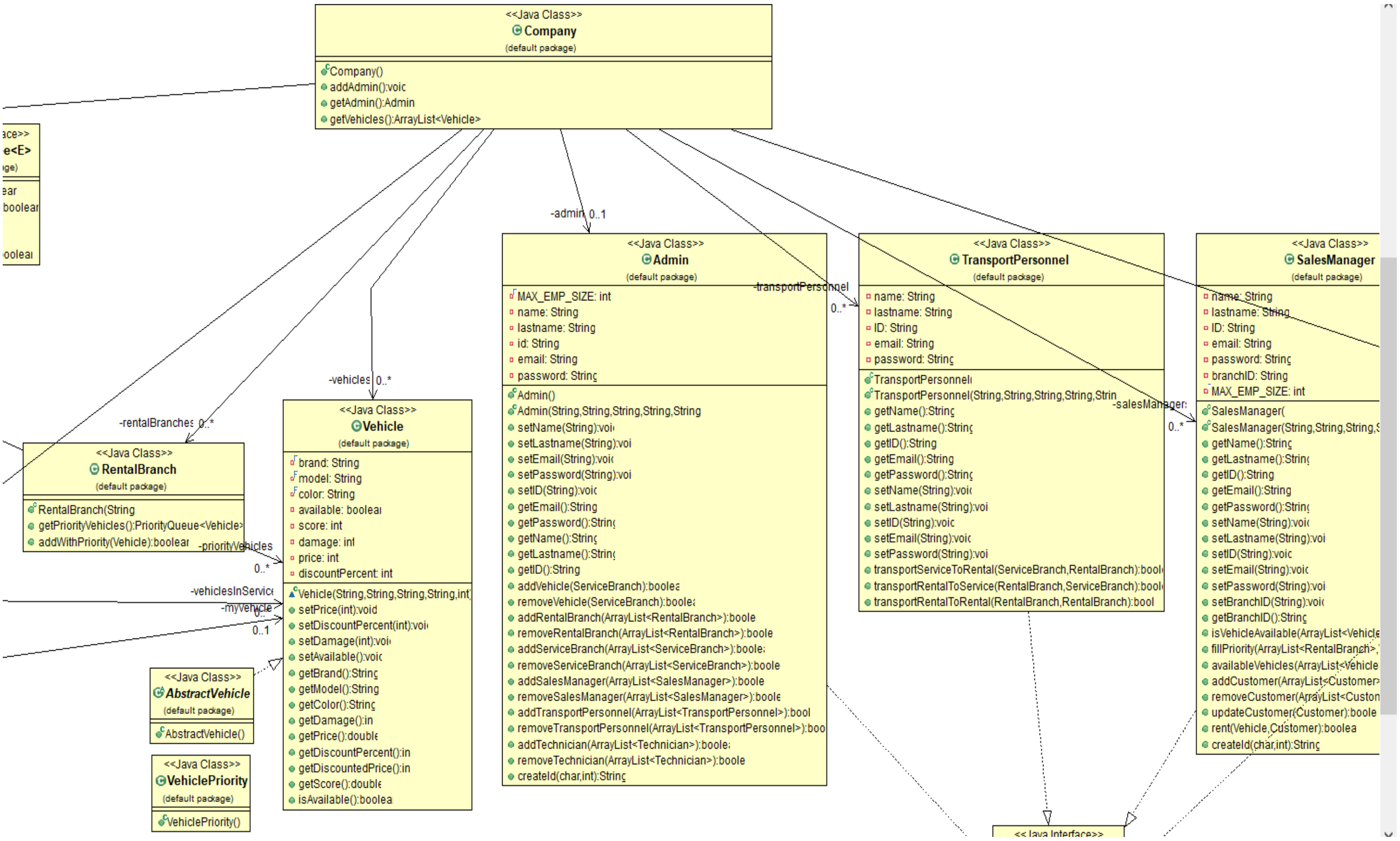
*** C3 (Component Diagram) added.

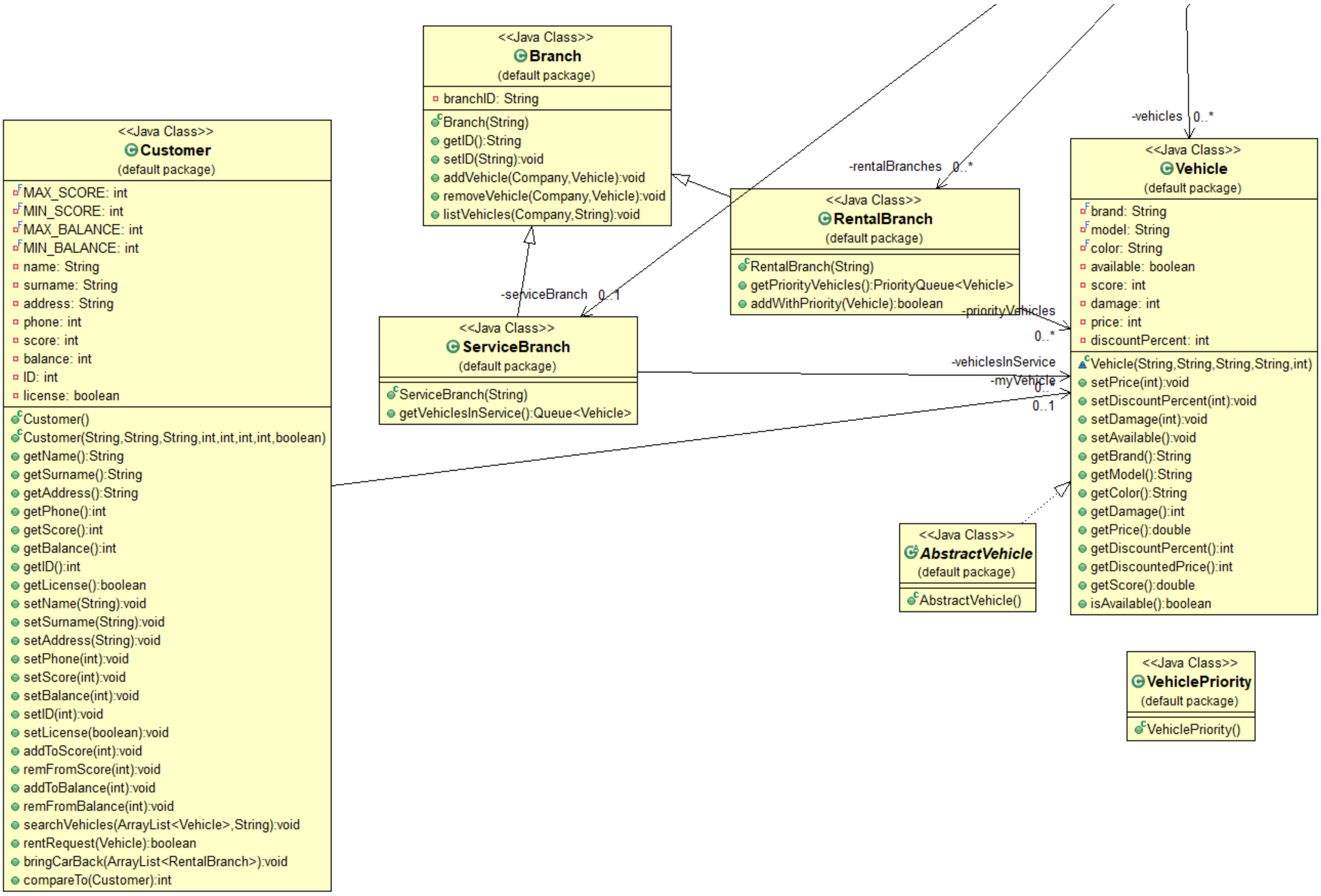
CLASS DIAGRAMS



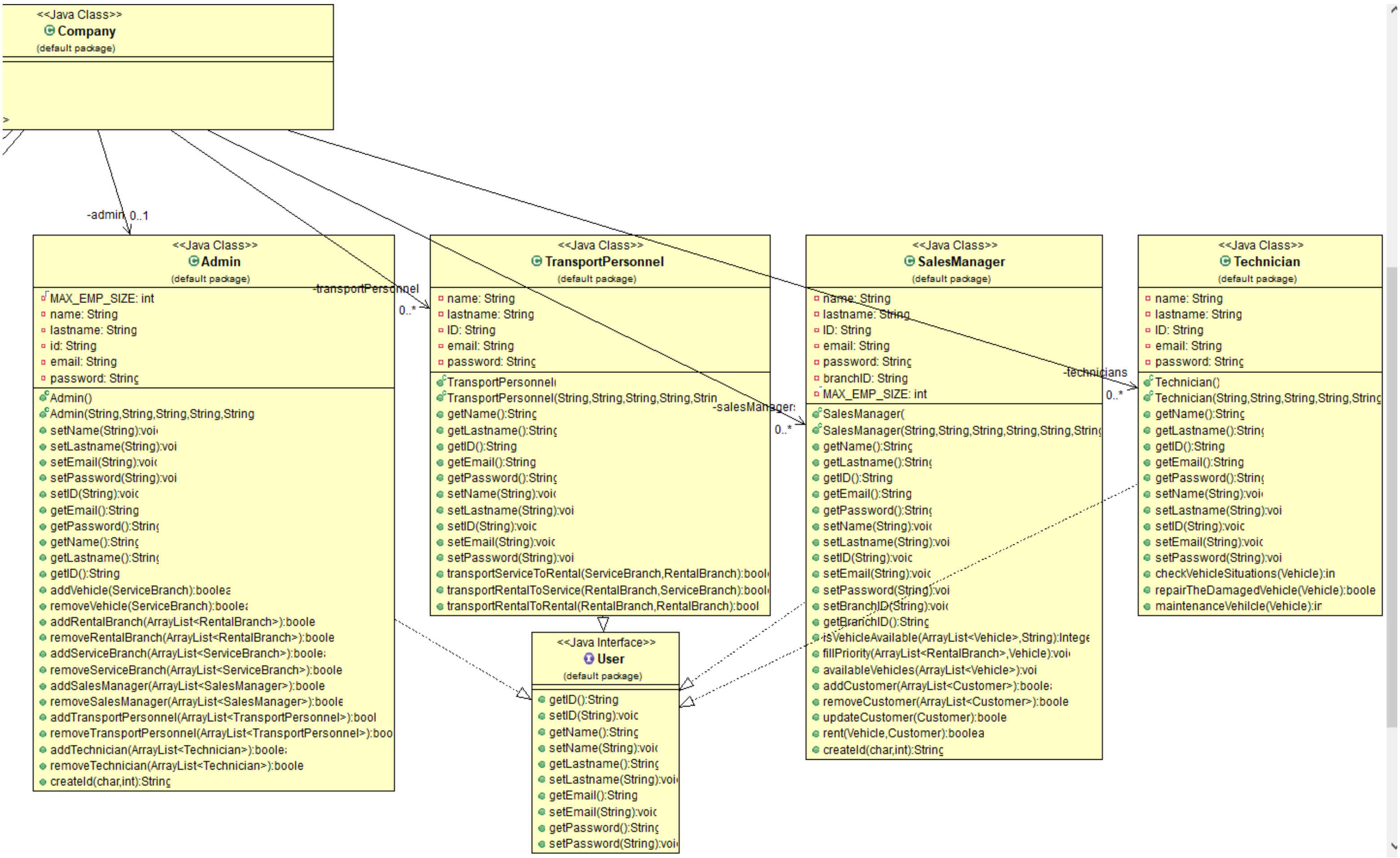
PARTS OF THE CLASS DIAGRAM





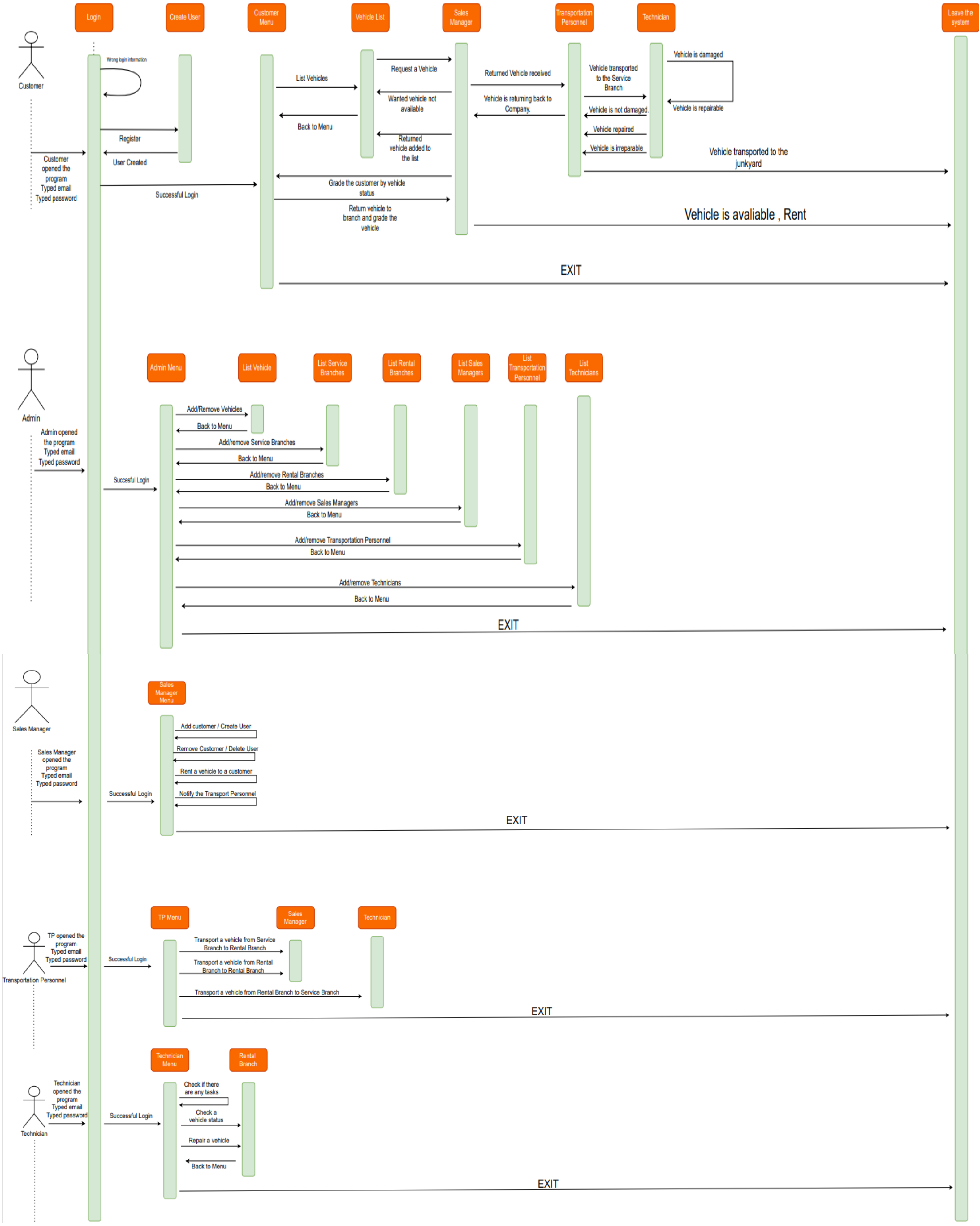


<<Java Class>>
Branch
(default package)
branchID: String
Branch(String)
getID():String
setID(String):void
addVehicle(Company,Vehicle):void
removeVehicle(Company,Vehicle):void
listVehicles(Company,String):void
<<Java Class>>
Customer
(default package)
MAX_SCORE: int
MIN_SCORE: int
MAX_BALANCE: int
MIN_BALANCE: int
name: String
surname: String
address: String
phone: int
score: int
balance: int
ID: int
license: boolean
Customer()
Customer(String,String,String,int,int,int,int,boolean)
getName():String
getSurname():String
getAddress():String
getPhone():int
getScore():int
getBalance():int
getID():int
getLicense():boolean
setName(String):void
setSurname(String):void
setAddress(String):void
setPhone(int):void
setScore(int):void
setBalance(int):void
setID(int):void
setLicense(boolean):void
addToScore(int):void
remFromScore(int):void
addToBalance(int):void
remFromBalance(int):void
searchVehicles(ArrayList<Vehicle>,String):void
rentRequest(Vehicle):boolean
bringCarBack(ArrayList<RentalBranch>):void
compareTo(Customer):int
-rentalBranches 0..*
<<Java Class>>
RentalBranch
(default package)
RentalBranch(String)
getPriorityVehicles():PriorityQueue<Vehicle>
addWithPriority(Vehicle):boolean
-serviceBranch 0..1
<<Java Class>>
ServiceBranch
(default package)
ServiceBranch(String)
getVehiclesInService():Queue<Vehicle>
-priorityVehicles 0..*
-vehiclesInService
-myVehicle 0..*
0..1
-vehicles 0..*
<<Java Class>>
Vehicle
(default package)
brand: String
model: String
color: String
available: boolean
score: int
damage: int
price: int
discountPercent: int
Vehicle(String,String,String,String,int)
setPrice(int):void
setDiscountPercent(int):void
setDamage(int):void
setAvailable():void
getBrand():String
getModel():String
getColor():String
getDamage():int
getPrice():double
getDiscountPercent():int
getDiscountedPrice():int
getScore():double
isAvailable():boolean
<<Java Class>>
AbstractVehicle
(default package)
AbstractVehicle()
<<Java Class>>
VehiclePriority
(default package)
VehiclePriority()



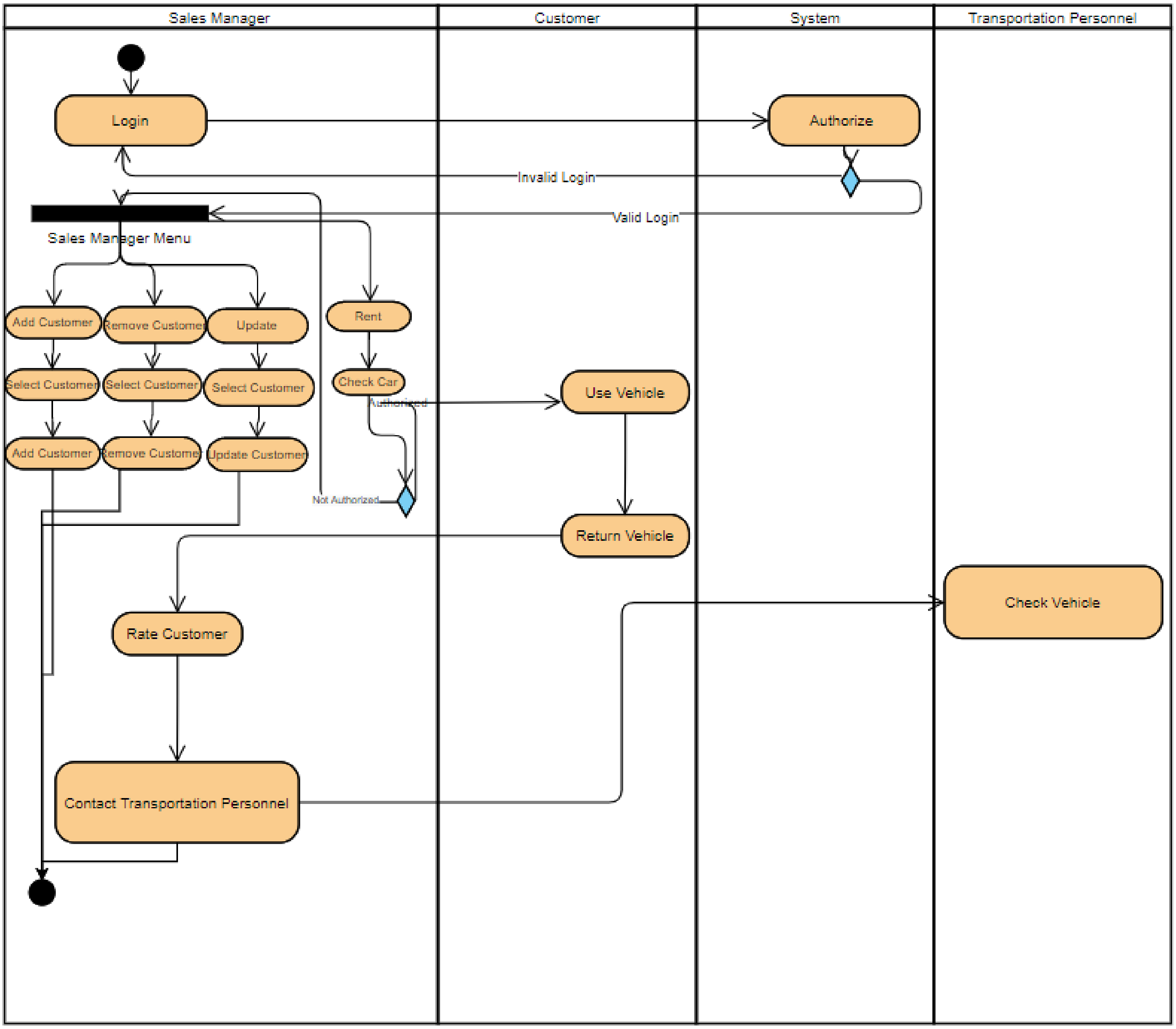
<<Java Class>>
Company
(default package)
-admin 0..1
<<Java Class>>
Admin
(default package)
MAX_EMP_SIZE: int
name: String
lastname: String
id: String
email: String
password: String
Admin()
Admin(String,String,String,String,String
setName(String):voi
setLastname(String):voi
setEmail(String):voi
setPassword(String):voi
setID(String):voic
getEmail():String
getPassword():String
getName():String
getLastname():String
getID():String
addVehicle(ServiceBranch):boolea
removeVehicle(ServiceBranch):boolea
addRentalBranch(ArrayList<RentalBranch>):boole
removeRentalBranch(ArrayList<RentalBranch>):boole
addServiceBranch(ArrayList<ServiceBranch>):boolea
removeServiceBranch(ArrayList<ServiceBranch>):boole
addSalesManager(ArrayList<SalesManager>):boole
removeSalesManager(ArrayList<SalesManager>):boole
addTransportPersonnel(ArrayList<TransportPersonnel>):bool
removeTransportPersonnel(ArrayList<TransportPersonnel>):boo
addTechnician(ArrayList<Technician>):boole
removeTechnician(ArrayList<Technician>):boole
createId(char,int):String
-transportPersonnel 0..*
<<Java Class>>
TransportPersonnel
(default package)
name: String
lastname: String
ID: String
email: String
password: String
TransportPersonnel
TransportPersonnel(String,String,String,String,Strin
getName():String
getLastname():String
getID():String
getEmail():String
getPassword():String
setName(String):voi
setLastname(String):void
setID(String):voic
setEmail(String):voic
setPassword(String):voi
transportServiceToRental(ServiceBranch,RentalBranch):bool
transportRentalToService(RentalBranch,ServiceBranch):bool
transportRentalToRental(RentalBranch,RentalBranch):bool
-salesManagers 0..*
<<Java Class>>
SalesManager
(default package)
name: String
lastname: String
ID: String
email: String
password: String
branchID: String
MAX_EMP_SIZE: int
SalesManager(
SalesManager(String,String,String,String,String,String
getName():String
getLastname():String
getID():String
getEmail():String
getPassword():String
setName(String):voic
setLastname(String):voi
setID(String):voic
setEmail(String):voic
setPassword(String):voi
setBranchID(String):voic
getBranchID():String
isVehicleAvailable(ArrayList<Vehicle>,String):Intege
fillPriority(ArrayList<RentalBranch>,Vehicle):voi
availableVehicles(ArrayList<Vehicle>):voi
addCustomer(ArrayList<Customer>):boole
removeCustomer(ArrayList<Customer>):boole
updateCustomer(Customer):boole
rent(Vehicle,Customer):boolea
createId(char,int):String
-technicians 0..*
<<Java Class>>
Technician
(default package)
name: String
lastname: String
ID: String
email: String
password: String
Technician()
Technician(String,String,String,String,String
getName():String
getLastname():String
getID():String
getEmail():String
getPassword():String
setName(String):voi
setLastname(String):voi
setID(String):voic
setEmail(String):voic
setPassword(String):voi
checkVehicleSituations(Vehicle):in
repairTheDamagedVehicle(Vehicle):boole
maintenanceVehilcle(Vehicle):ir
<<Java Interface>>
User
(default package)
getID():String
setID(String):voic
getName():String
setName(String):voi
getLastname():String
setLastname(String):voi
getEmail():String
setEmail(String):voic
getPassword():String
setPassword(String):voi

SEQUENCE DIAGRAM

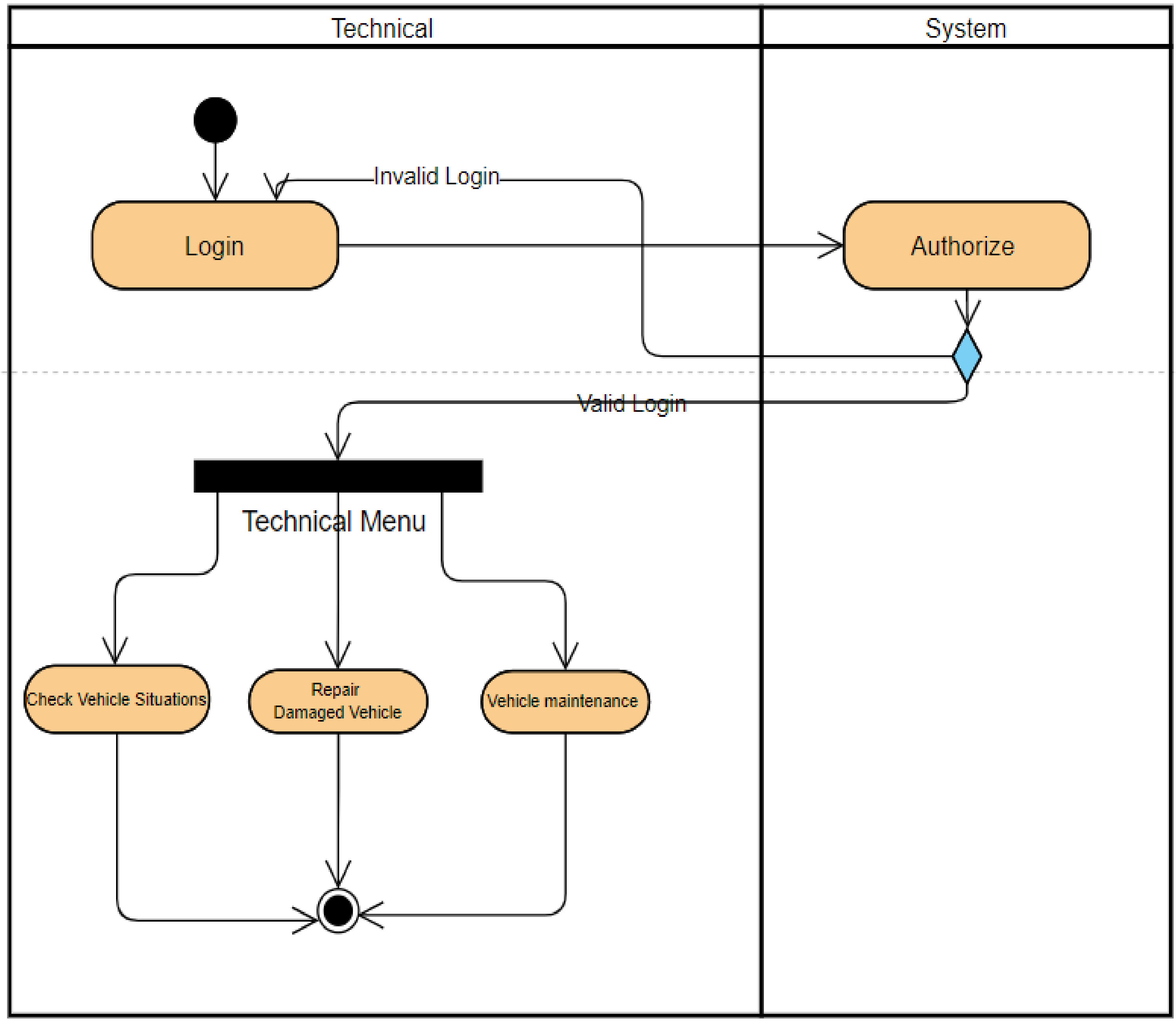

Login
Create User
Customer Menu
Vehicle List
Sales Manager
Transportation Personnel
Technician
Leave the system
Customer
Wrong login information
Register
User Created
Customer opened the program Typed email Typed password
Successful Login
List Vehicles
Back to Menu
Request a Vehicle
Wanted vehicle not available
Returned vehicle added to the list
Grade the customer by vehicle status
Return vehicle to branch and grade the vehicle
Returned Vehicle received
Vehicle is returning back to Company.
Vehicle transported to the Service Branch
Vehicle is not damaged.
Vehicle repaired
Vehicle is irreparable
Vehicle is damaged
Vehicle is repairable
Vehicle transported to the junkyard
Vehicle is avaliable , Rent
EXIT
Admin
Admin opened the program Typed email Typed password
Succesful Login
Admin Menu
List Vehicle
List Service Branches
List Rental Branches
List Sales Managers
List Transportation Personnel
List Technicians
Add/Remove Vehicles
Back to Menu
Add/remove Service Branches
Back to Menu
Add/remove Rental Branches
Back to Menu
Add/remove Sales Managers
Back to Menu
Add/remove Transportation Personnel
Back to Menu
Add/remove Technicians
Back to Menu
EXIT
Sales Manager
Sales Manager opened the program Typed email Typed password
Successful Login
Sales Manager Menu
Add customer / Create User
Remove Customer / Delete User
Rent a vehicle to a customer
Notify the Transport Personnel
EXIT
Transportation Personnel
TP opened the program Typed email Typed password
Successful Login
TP Menu
Sales Manager
Technician
Transport a vehicle from Service Branch to Rental Branch
Transport a vehicle from Rental Branch to Rental Branch
Transport a vehicle from Rental Branch to Service Branch
EXIT
Technician
Technician opened the program Typed email Typed password
Successful Login
Technician Menu
Rental Branch
Check if there are any tasks
Check a vehicle status
Repair a vehicle
Back to Menu
EXIT

ACTIVITY DIAGRAM

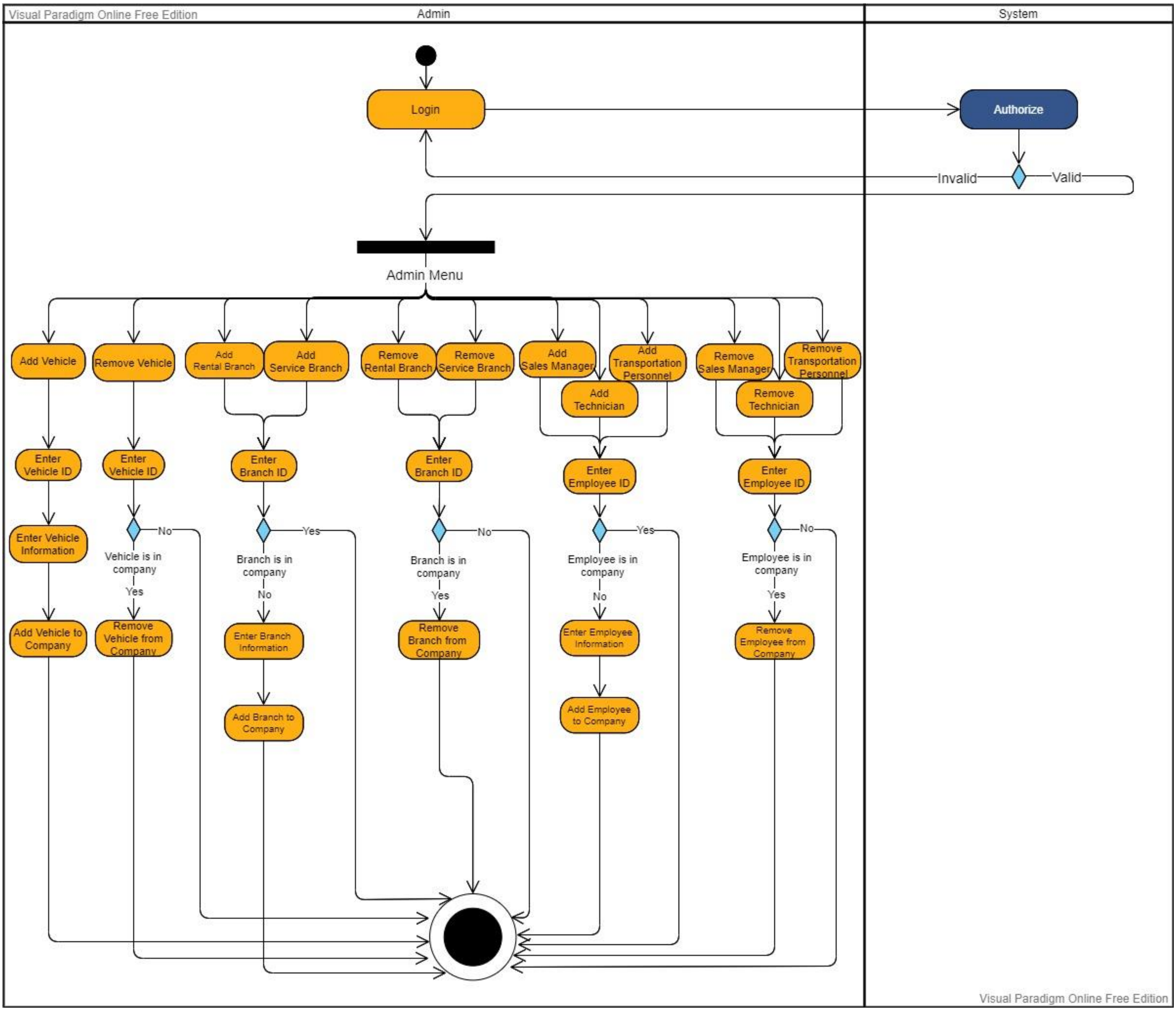
Sales Manager



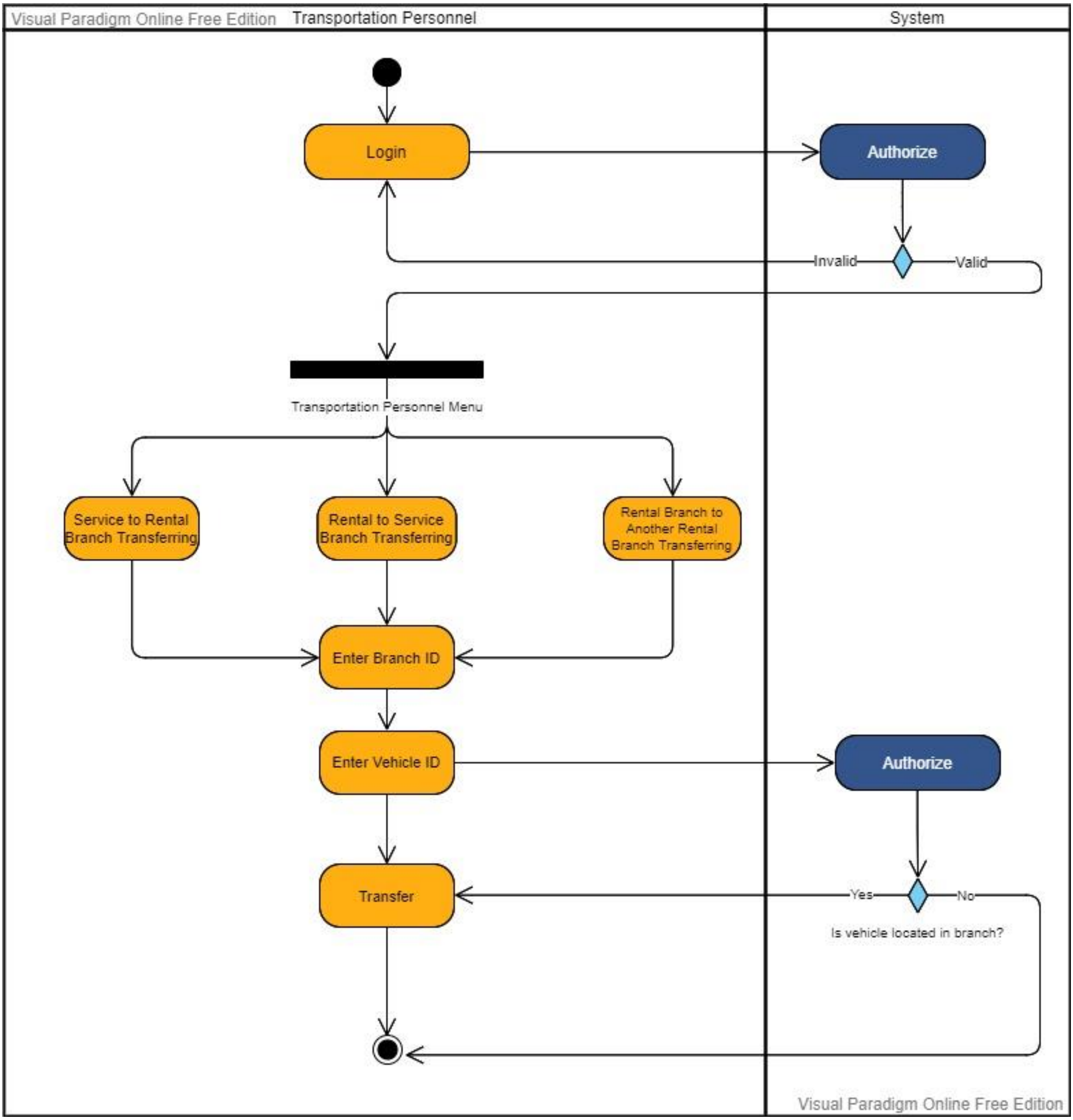
Techician


Technical
System
Invalid Login
Login
Authorize
Valid Login
Technical Menu
Check Vehicle Situations
Repair
Damaged Vehicle
Vehicle maintenance

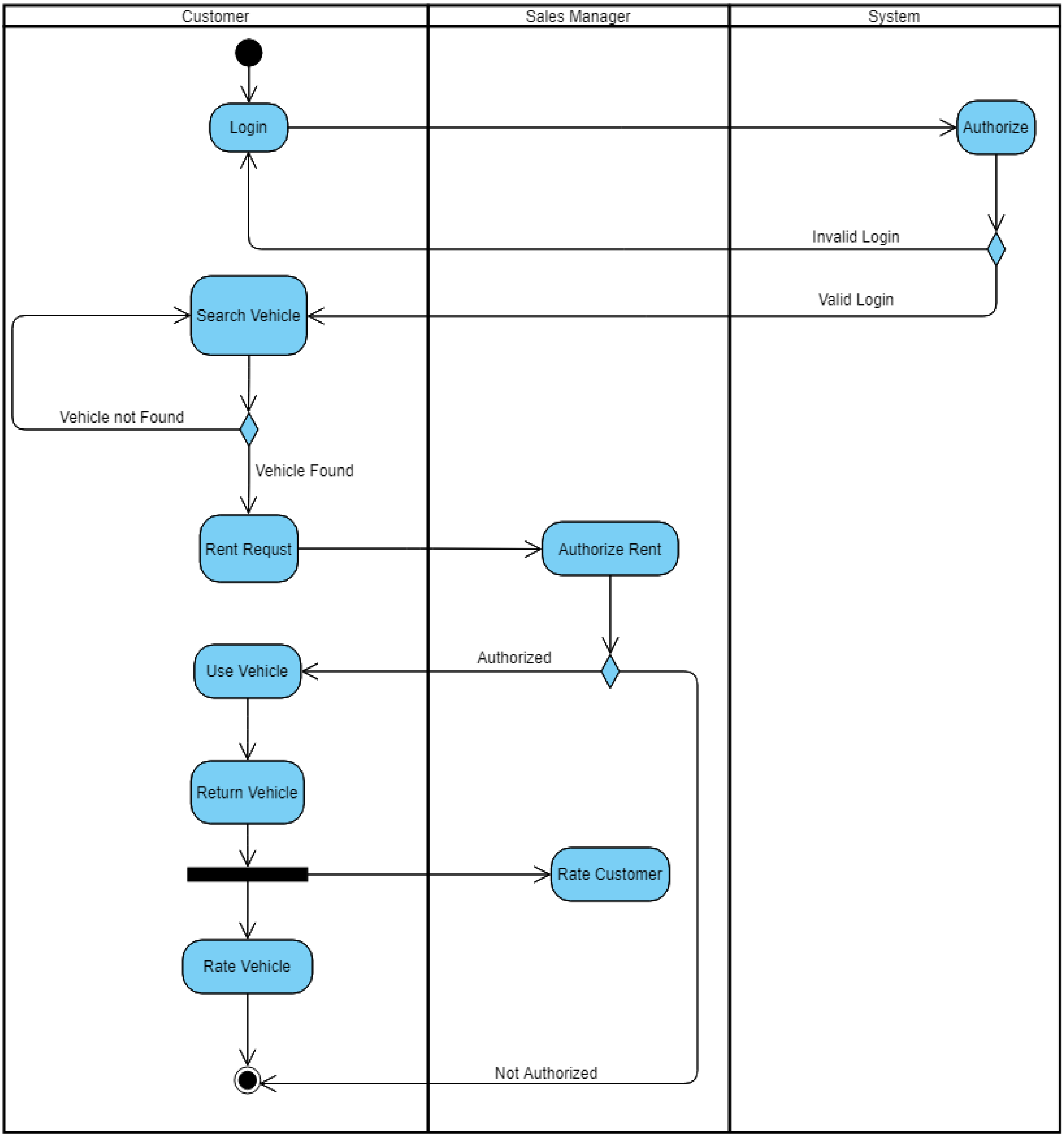
Admin


Visual Paradigm Online Free Edition
Admin
System
Login
Authorize
Invalid
Valid
Admin Menu
Add Vehicle
Remove Vehicle
Add Rental Branch
Add Service Branch
Remove Rental Branch
Remove Service Branch
Add Sales Manager
Add Transportation Personnel
Add Technician
Remove Sales Manager
Remove Transportation Personnel
Remove Technician
Enter Vehicle ID
Enter Vehicle ID
Enter Branch ID
Enter Branch ID
Enter Employee ID
Enter Employee ID
Enter Vehicle Information
Add Vehicle to Company
Vehicle is in company
No
Yes
Remove Vehicle from Company
Branch is in company
Yes
No
Enter Branch Information
Add Branch to Company
Branch is in company
No
Yes
Remove Branch from Company
Employee is in company
Yes
No
Enter Employee Information
Add Employee to Company
Employee is in company
No
Yes
Remove Employee from Company
Visual Paradigm Online Free Edition

Transportation Personnel


Visual Paradigm Online Free Edition
Transportation Personnel
System
Login
Authorize
Invalid
Valid
Transportation Personnel Menu
Service to Rental Branch Transferring
Rental to Service Branch Transferring
Rental Branch to Another Rental Branch Transferring
Enter Branch ID
Enter Vehicle ID
Authorize
Transfer
Yes
No
Is vehicle located in branch?
Visual Paradigm Online Free Edition

Customer


Customer
Sales Manager
System
Login
Authorize
Invalid Login
Valid Login
Search Vehicle
Vehicle not Found
Vehicle Found
Rent Requst
Authorize Rent
Authorized
Use Vehicle
Return Vehicle
Rate Customer
Rate Vehicle
Not Authorized

NON-TRIVIAL IMPLEMENTATION

Admin implements User

Creating Admin:

Admin is created in the Company using constructor of Admin with the inputs (unique id , email , password).

Creating Employees:

Admin creates a Technician , Transportation Personnel and Sales Manager with the inputs received by Admin class' addTechnician() , addTransportationPersonnel() or addSalesManager() methods. These inputs will be given to the specific class' constructors. So the employee(s) will be created and added to the company , to the **ArrayList** next to other Employees.

Removing Employees:

Admin removes a Technician , Transportation Personnel and Sales Manager with the inputs received by Admin class' removeTechnician() , removeTransportationPersonnel() or removeSalesManager() methods. Employees who match with those inputs will be removed from the ArrayList so from the company.

Adding Vehicles to the Company:

Admin adds a vehicle to the company with the inputs provided by addVehicle() method. Those inputs will be given to the Vehicle class' constructor. So the wanted vehicle with the wanted features will be added to the company , to the **ArrayList** next to other vehicles.

Removing Vehicles from the Company:

Admin removes a vehicle from the company with the inputs provided by removeVehicle() method. Vehicles which match those inputs will be removed from the Company and the ArrayList.

Creating Branches:

Admin creates a service branch or rental branch with the inputs provided by addRentalBranch() or addServiceBranch(). Those inputs will be given to the RentalBranch class' constructor or ServiceBranch class's constructor. Then created branche(s) will be added to the company , to the **ArrayList** next to other branches.

Removing Branches:

Admin removes a service branch or rental branch with the inputs provided by removeRentalBranch() or removeServiceBranch(). Those methods seek for the matching branch with those inputs , then the wanted branch will be deleted from the ArrayList so from the Company.

Transportation Personnel implements User

Creating Transportation Personnel:

Transportation personnel is created by the Admin with the addTransportationPersonnel() method which takes ArrayList of transportation personnels as input.

Transporting Vehicles from Rental Branch to Service Branch:

Transportation personnel transfers a vehicle from a rental branch to a service branch for maintenance with the inputs provided by transportRentalToService() method. Those inputs will be used for removing a vehicle from ArrayList of rental branch vehicles and adding to Queue of service branch vehicles.

Transporting Vehicles from Service Branch to Rental Branch:

Transportation personnel transfers a maintained vehicle from a service branch to a rental branch with the inputs provided by transportServiceToRental() method. Those inputs will be used for removing a vehicle from Queue of rental branch vehicles and adding to ArrayList of rental branch vehicles.

Transporting Vehicles from Rental Branch to Rental Branch:

Transportation personnel transfers a vehicle from a rental branch to another rental branch with the inputs provided by transportRentalToRental() method. Those inputs will be used for removing a vehicle from ArrayList of rental branch vehicles and adding to another ArrayList of rental branch vehicles.

Sales Manager implements User

Creating Sales Manager

Sales Manager are created in the Company by admin using constructor of Sales Manager with the inputs (unique id , email , password, name, branchID).

Creating Customers

Sales Manager creates Customers with the inputs received by the addCustomer() method. These inputs will be given to the specific class’ constructors. So the customers will be created and added to the company, to the ArrayList next to other Employees.

Removing Customer

Sales Manager removes a Customer with the inputs received by the removeCustomer() method. Customer who match with those inputs will be removed from the ArrayList so from the company.

Updating Customer Information

Sales Manager updates the information of a customer given as a input to the updateCustomer() method. If the customer is not exist, nothing happens otherwise method will ask the Sales Manager what information will be updated with what.

Renting a Vehicle

After receiving a rent request from the customer, the Sales Manager decides whether the rent will be approved with the rent() method, where the customer and the requested vehicle are input.

Contacting Transporter

To transport a vehicle Sales manager can call a transporter employee with the callTransporter() method.

Customer implements User

Customers are held in a Binary Search Tree structure in the company class. BST used to search customers faster.

Customer class implements User interface.

Parameterized Constructor

Creates the customer object in the addCustomer method of Sales Manager class.

Setters and getters.

Search Vehicles

Searches over the vehicles in the branch. Chooses a model to search explicitly. After chooses a model, chosen type of vehicles will be represented in a Priority Queue to list through their rating. In the getPriorityVehicles() method of RentalBranch class.

rentRequest

Sends rent request to the sales manager. Sales manager decides if the car is avaliable with the isVehicleAvaliable() method.

bringCarBack

Brings the car back to a branch. Transportation Personnel delivers the vehicle to Service branch with RentaltoService() method.

Technician implements User

Parameterized Constructor:

Technician generated in Company class takes name, lastname, ID, email, password as parameters.

These values correspond to variables in the Technician class in the class.

assigned with setters.

Check Vehicle Situations:

Detects the damage to the vehicle with the vehicle's getDamage () method, which it takes as a parameter. And from the getPrice () method to the rate of damage

Returns the cost of damage based on the price of the vehicle.

Repair The Damaged Vehicle:

The vehicle's damage is checked with the vehicle's getDamage () method, which the repairTheDamagedVehicle method takes as a parameter. If the damage is not much, it will be repaired. If the damage is severe, it cannot be repaired.

The Maintenance Vehicle:

Takes as a parameter and checks the vehicle's status with the checkVehicleSituations method, if the vehicle is damaged, with the repairTheDamagedVehicle

it fixes. Returns -1 if the vehicle is damaged beyond repair.

Company

Every action happens in the company class. All the datas are kept in

there.

addAdmin-> Adds an admin to the company.

getAdmin-> Returns the admin object.

getVehicles-> Returns the vehicles in the system.

Vehicle

Creating Vehicle

Vehicles are created by admins with addVehicle() method which takes brand name, model name, color, and rent price as parameters.

With setter methods:

Admin can set the price and the discount rate of the vehicle.

Sales manager can change the availability of the vehicle.

Technicians can change the damage of the vehicle it has before and after repair.

Vehicle Score

Customers gives a score to vehicles with giveScore() method after using them which effects their overall score. Every customer have access to this score.

Rental Branch extends Branch:

RentalBranch extends Branch class. So RentalBranch will be created with Branch's constructor with the provided inputs by Admin's addRentalBranch() method.

When a customer asks to see the list of the vehicle from the Sales Manager , Sales Manager will list the vehicles in the branch by priority with RentalBranch's getPriorityVehicles(). Vehicles will be listed based on previous customers' reviews.

When a vehicle wanted to be added to the branch , PriorityQueue will be updated with the wanted car in it by RentalBranch's addWithPriority() method.

Service Branch extends Branch:

ServiceBranch extends Branch class. So ServiceBranch will be created with Branch's constructor with the provided inputs by Admin's addServiceBranch() method.

When a Vehicle returned by Customer arrives to the RentalBranch , Transportation Personnel will deliver the Vehicle to the ServiceBranch. And in the ServiceBranch , Vehicles that waiting to be repaired stored in a Queue. The returned Vehicle will be added to the "to be repaired Queue".

When Technician(s) decides to repair the next Vehicle , this Vehicle will be decided according to the Queue structure and will be removed from the queue by using Queue's poll() method. And corresponding vehicle will be given to Technician's maintenanceVehicle() , repairTheDamagedVehicle() or checkVehicleSituations() as input.

Branch:

Creating a Branch is not possible without specifying ServiceBranch or RentalBranch.

When a Vehicle wanted to be add to a Branch , it's informations will be received by Admin's addRentalBranch() method or addServiceBranch() method. Then , Branch's addVehicle() method will be called. So the vehicle will be added to the Company.

When a Vehicle wanted to be deleted from a Branch , it's informations will be received by Admin's removeRentalBranch() method or removeServiceBranch() method. Then Branch's removeVehicle() method will be called . So the vehicle will be removed from the Company.

To list all vehicles with a specific model in the company. All Branch's listVehicles() method will be called.

TEST CASES

1. **Test Case**

   Title: Customer sign up

   Description: Customer should be able to successfully sign up.

   Precondition: Customer must be on sign up screen and admin must be registered with email, name, last name, password and confirm password.

   Test Steps:

   - Open menu.
   - Select customer.
   - Select sign up page.
   - Enter email, name, last name, password and confirm password.
   - Click 'sign up'.

   Expected Result: The customer successfully sign up and the customer menu opens.

2. **Test Case**

   Title: Admin login

   Description: Admin should be able to successfully login with password.

   Precondition: Admin must be in the company system.

   Test Steps:

   - Open menu.

- Select admin.
- Select login.
- Enter the password.
- Click login.

Expected Result: Admin successfully logs into the system and and the admin menu opens.

3. **Test Case**

   Title: Transportation Personnel login

   Description: Transportation personnel should be able to successfully login with password.

   Precondition: Transportation personnel must be in the company system.

   Test Steps:

   - Open menu.
   - Select transportation personnel.
   - Select login.
   - Enter the branch ID.
   - Click login.

   Expected Result: Transportation personnel successfully logs into the system and and the transportation personnel menu opens.

4. **Test Case**

   Title: Technician login

   Description: Technician should be able to successfully login with password.

   Precondition: Technician must be in the company system.

   Test Steps:

   - Open menu.
   - Select technician.
   - Select login.
   - Enter thebranch ID.
   - Click login.

   Expected Result: Technician successfully logs into the system and and the technician menu opens.

5. **Test Case**

Title: Sales manager login

Description: Sales manager should be able to successfully login with password.

Precondition: Sales manager must be in the company system.

Test Steps:

- Open menu.
- Select sales manager.
- Select login.
- Enter the branch ID.
- Click login.

Expected Result: Sales manager successfully logs into the system and and the Sales manager menu opens.

6. **Test Case**

Title: Customer login

Description: A customer should be able to successfully login with password.

Precondition: The customer must be in the company system.

Test Steps:

- Open menu.
- Select customer.
- Select login.
- Enter the password.
- Click login.

Expected Result: Customer successfully logs into the system and the customer menu opens.

7. **Test Case**

Title: Add vehicle

Description: Admin can add vehicles to the company.

Precondition: Admin must be in the company system.

Test Steps:

- Open admin menu.
- Select ‘add vehicle’.
- Enter the vehicle ID, brand, model, color and cost of the vehicle.
- Click add.

Expected Result: The vehicle adds into company by admin.

8. **Test Case**

Title: Remove vehicle

Description: The admin can delete vehicles from the company.

Precondition: Admin must be in the company system and the vehicle must be in the company system.

Test Steps:

- Open admin menu.
- Select ‘remove vehicle’.
- Enter the vehicle ID.
- Click remove.

Expected Result: The vehicle is deleted from the company by admin.

9. **Test Case**

Title: Add rental branch

Description: Admin can add branch to the company.

Precondition: Admin must be in the company system and the branch to be added should not exist in the system.

Test Steps:

- Open admin menu.
- Select ‘add rental branch.
- Enter the branch ID and branch name.
- Click add.

Expected Result: The rental branch adds into company by admin.

10. **Test Case**

Title: Remove rental branch

Description: The admin can delete the rental branch from the company.

Precondition: Admin must be in the company system and the rental branch must be in the company system.

Test Steps:

- Open admin menu.
- Select ‘remove rental branch’.
- Enter the rental branch ID.
- Click remove.

Expected Result: The rental branch is deleted from the company by admin.

11. **Test Case**

Title: Add service branch

Description: Admin can add branch to the company.

Precondition: Admin must be in the company system and the branch to be added should not exist in the system.

Test Steps:

- Open admin menu.
- Select ‘add service branch.
- Enter the branch ID and branch name.
- Click add.

Expected Result: The service branch adds into company by admin.

12. **Test Case**

Title: Remove service branch

Description: The admin can delete the service branch from the company.

Precondition: Admin must be in the company system and the service branch must be in the company system.

Test Steps:

- Open admin menu.
- Select ‘remove service branch’.
- Enter the service branch ID.
- Click remove.

Expected Result: The service branch is deleted from the company by admin.

13. **Test Case**

Title: Add sales manager

Description: Admin can add sales manager to the company.

Precondition: Admin must be in the company system and the sales manager to be added should not exist in the system.

Test Steps:

- Open admin menu.
- Select ‘add sales manager’.
- Enter the ID, name, surname of the sales manager and enter the branch ID.
- Click add.

Expected Result: The sales manager adds into company by admin.

14. **Test Case**

Title: Remove sales manager

Description: The admin can delete the service branch from the company.

Precondition: Admin must be in the company system and the sales manager must be in the company system.

Test Steps:

- Open admin menu.
- Select ‘remove sales manager’.
- Enter the sales manager ID.
- Click remove.

Expected Result: The sales manager is deleted from the company by admin.

15. **Test Case**

Title: Add transportation personnel

Description: Admin can add transportation personnel to the company.

Precondition: Admin must be in the company system and the transportation personnel to be added should not exist in the system.

Test Steps:

- Open admin menu.
- Select ‘add transportation personnel’.
- Enter the ID, name, surname of the transportation personnel and enter the branch ID.
- Click add.

Expected Result: The transportation personnel adds into company by admin.

16. **Test Case**

Title: Remove transportation personnel

Description: The admin can delete the transportation personnel from the company.

Precondition: Admin must be in the company system and the transportation personnel must be in the company system.

Test Steps:

- Open admin menu.
- Select ‘remove transportation personnel’.
- Enter the transportation personnel ID.
- Click remove.

Expected Result: The transportation personnel is deleted from the company by admin.

17. **Test Case**

Title: Add technician

Description: Admin can add technician to the company.

Precondition: Admin must be in the company system and the technician to be added should not exist in the system.

Test Steps:

- Open admin menu.
- Select ‘add technician’.
- Enter the ID, name, surname of the technician and enter the branch ID.
- Click add.

Expected Result: The technician adds into company by admin.

18. **Test Case**

Title: Remove technician

Description: The admin can delete the technician from the company.

Precondition: Admin must be in the company system and the technician must be in the company system.

Test Steps:

- Open admin menu.
- Select ‘remove technician.
- Enter the technician ID.
- Click remove.

Expected Result: The technician is deleted from the company by admin.

19. **Test Case**

Title: Search vehicle

Description: A customer can search a vehicle in the system.

Precondition: Customer must be login the system and the searched vehicle must have been added to the system beforehand.

Test Steps:

- Open menu.
- Select search vehicle.
- Enter the brand, type and color of the vehicle.
- Click search.

Expected Result: The customer sees the information of the vehicle that suits the specifications on the screen.

20. **Test Case**

Title: Search vehicle (not in the company)

Description: The vehicle with the features the customer is looking for may not be in the system.

Precondition: Customer must be login the system and the searched vehicle must not have been added to the system before.

Test Steps:

- Open menu.
- Select search vehicle.
- Enter the brand, type and color of the vehicle.
- Click search.

Expected Result: The system gives a warning: "There are no vehicles with these features."

21. **Test Case**

Title: Rent request a vehicle from customer

Description: When the customer wants to rent a car, he/she makes a request.

Precondition: The customer must be login the system.

Test Steps:

- Open menu.
- Select rent request.
- Enter the brand, type and color of the vehicle.
- Click add request.

Expected Result: System gives a message: 'Your request has been forwarded to sales manager.'

22. **Test Case**

Title: Rate the vehicle

Description: The customer can rate the vehicle he/she rented.

Precondition: The customer must have rented a car.

Test Steps:

- Open menu.
- Select deliver the vehicle.
- Click rate the vehicle.
- Enter rating.
- Click save.

Expected Result: The customer has rated the vehicle.

23. **Test Case**

Title: Check available vehicle

Description: Sales manager can view the list of vehicles that are available for rental.

Precondition: The sales manager should login the system.

Test Steps:

- Open sales manager menu.
- Select ‘list of available rental vehicles’.

Expected Result: Sales manager displays the list of vehicles.

24. **Test Case**

Title: Add customer

Description: Sales manager can add customers to the system.

Precondition: Sales manager must login to the system.

Test Steps:

- Open Sales manager menu.
- Select ‘add customer’.
- Enter the information of customer.
- Click ‘add’.

Expected Result: The customer is registered in the system.

25. **Test Case**

Title: Update the customer information

Description: Sales manager can update the information of the existing customer in the system.

Precondition: The customer must be registered in the system.

Test Steps:

- Open sales manager menu.
- Select ‘update customer information’.
- Enter the updating information and customer ID.
- Click ‘update’.

Expected Result: The system sends a message: “Customer information has been updated successfully.”

26. **Test Case**

Title: Rent the vehicle (customer have the driving licence)

Description: The sales manager confirms the customer's rental request.

Precondition: The customer must be registered in the system and the customer must have a driving license.

Test Steps:

- Open sales manager menu.
- Select rent.

- Select ‘confirm the request and rent the vehicle’.

Expected Result: Sales manager rents the vehicle to the customer.

27. **Test Case**

Title: Checking the repair status of the vehicle

Description: After the vehicle is delivered, the technician makes the damage assessment of the vehicle. If there is no problem in the vehicle, it will maintain it.

Precondition: The vehicle must be at the service branch.

Test Steps:

- Open the technician menu.
- Select the ‘vehicle repair status’.
- Change the vehicle status as ‘true’.
- Click save.

Expected Result: The system sends a message: 'there is no damage to the vehicle'.

28. **Test Case**

Title: Checking the repair status of the vehicle (damaged vehicle)

Description: After the vehicle is delivered, the technician makes the damage assessment of the vehicle.

Precondition: The vehicle must be at the service branch.

Test Steps:

- Open technician menu.
- Select the ‘vehicle repair status’.
- Change the vehicle status as ‘false.
- Click save.

Expected Result: The system sends a message: ‘The vehicle is damaged, it needs to be repaired’.

29. **Test Case**

Title: Periodic maintenance of the vehicle

Description: The vehicles received from the customer are maintained.

Precondition: The customer must have delivered the vehicle.

Test Steps:

- Open technician menu.
- Select ‘periodic maintanence’.
- Enter vehicle ID.
- Click ‘OK’.

Expected Result: The system sends a message: ‘the vehicle has been maintained’.

30. **Test Case**

Title: Transferring the vehicle from the service branch to the rental branch

Description: Vehicle is transferred from service branch to rental branch by transportation personnel.

Precondition: The vehicle must be located at the service branch.

Test Steps:

- Open transportation personnel menu
- Select ‘service to rental branch transferring’
- Enter the rental branch ID and vehicle ID.
- Click ‘transfer’.

Expected Result: The vehicle was transferred from the service branch to the rental branch.

31. **Test Case**

Title: Transferring the vehicle from the rental branch to he service branch

Description: Vehicle is transferred from rental branch to service branch by transportation personnel.

Precondition: The vehicle must be located at the rental branch.

Test Steps:

- Open transportation personnel menu
- Select ‘rental to service branch transferring’
- Enter the service branch ID and vehicle ID.
- Click ‘transfer’.

Expected Result: The vehicle was transferred from the rental branch to the service branch.

32. **Test Case**

Title: Transferring the vehicle from rental branch to another rental branch

Description: Vehicle is transferred from rental branch to the another rental branch by transportation personnel.

Precondition: The vehicle must be located at the rental branch.

Test Steps:

- Open transportation personnel menu
- Select ‘rental branch to another rental branch transferring’
- Enter the other rental branch ID and vehicle ID.
- Click ‘transfer’.

Expected Result: The vehicle was transferred from the rental branch to the service branch.

33. **Test Case**

Title: Delivery of the rented vehicle

Description: The customer delivers the rented vehicle.

Precondition: The customer must have rented a vehicle.

Test Steps:

- Open the customer menu.
- Select the ‘deliver the vehicle’.
- Enter the vehicle name, type, brand.
- Click ‘OK’.

Expected Result: The customer leaves the vehicle back to the branch.

34. **Test Case**

Title: Update customer rating

Description: After the customer leaves the vehicle back to the branch, sales manager scores the customer.

Precondition: The customer must have delivered the vehicle to the branch.

Test Steps:

- Open sales manager menu.
- Select the ‘rate the customer’.

- Enter the customer ID.
- Click ‘rate’.

Expected Result: System gives message: ‘Customer score has been recorded.’

35. **Test Case**

Title: Customer login (invalid passsword)

Description: A customer cannot log into the system with the wrong password.

Precondition: Customer must be on the login screen.

Test Steps:

- Open menu.
- Select ‘customer’.
- Select ‘login’.
- Enter the wrong password.
- Click ‘login’.

Expected Result: The customer cannot log into the system and the system warns: "Invalid password".

36. **Test Case**

Title: Admin login (invalid passsword)

Description: Admin cannot log into the system with the wrong password.

Precondition: Admin must be on the login screen.

Test Steps:

- Open menu.
- Select admin.
- Select login.
- Enter the invalid password.
- Click login.

Expected Result: The administrator cannot log into the system and the system warns: "Invalid password".

37. **Test Case**

Title: Transportation personel login (with wrong branch ID)

Description: The transportation personnel cannot logs into the system with invalid branch ID.

Precondition: Transportation personnel must be on the login page.

Test Steps:

- Open menu.
- Select transportation personnel.
- Select login.
- Enter the invalid branch ID.

Expected Result: The transportation personnel cannot log into the system and the system warns: " There is no such branch."

38. **Test Case**

Title: Sales manager login (with wrong branch ID)

Description: The sales manager cannot logs into the system with invalid branch ID.

Precondition: The sales manager must be on the login page.

Test Steps:

- Open menu.
- Select sales manager.
- Select login.
- Enter the invalid branch ID.

Expected Result: The sales manager cannot log into the system and the system warns: " There is no such branch."

39. **Test Case**

Title: Technician login (with wrong branch ID)

Description: The technician cannot logs into the system with invalid branch ID.

Precondition: The technician must be on the login page.

Test Steps:

- Open menu.
- Selecttechnician.
- Select login.
- Enter the invalid branch ID.

Expected Result: The technician cannot log into the system and the system warns: " There is no such branch."

40. **Test Case**

Title: Remove rental branch (when there is no such branch)

Description: The admin cannot delete the rental branch with invalid branch ID from the company.

Precondition: Admin must be on the menu and the branch should not be in the system.

Test Steps:

- Open admin menu.
- Select ‘remove rental branch’.
- Enter the rental branch ID.
- Click remove.

Expected Result: The admin cannot delete the branch and the system warns: " There is no such branch."

41. **Test Case**

Title: Add rental branch (already added)

Description: The admin cannot add the rental branch previously added to the system again.

Precondition: The branch must be alreadyy added in the system.

Test Steps:

- Open menu.
- Select add rental branch.
- Enter branch name and branch ID.
- Click add.

Expected Result: Admin cannot add the rental branch and the system warns: “Branch is already added”.

42. **Test Case**

Title: Remove service branch (when there is no such branch)

Description: The admin cannot delete the service branch with invalid branch ID from the company.

Precondition: Admin must be on the menu and the branch should not be in the system.

Test Steps:

- Open admin menu.
- Select ‘remove service branch’.
- Enter the rental branch ID.
- Click remove.

Expected Result: The admin cannot delete the branch and the system warns: " There is no such branch."

43. **Test Case**

Title: Add service branch (already added)

Description: The admin cannot add the rental branch previously added to the system again.

Precondition: The branch must be already added in the system.

Test Steps:

- Open menu.
- Select add service branch.
- Enter branch name and branch ID.
- Click add.

Expected Result: Admin cannot add the rental branch and the system warns: “Branch is already added”.

44. **Test Case**

Title: Add sales manager (already added)

Description: The admin cannot add the sales manager previously added to the system again.

Precondition: The sales manager must be already added in the system.

Test Steps:

- Open menu.
- Select add sales manager.
- Enter the ID, name, surname of the technician and enter the branch ID.
- Click add.

Expected Result: Admin cannot add the sales manager and the system warns: “Sales manager is already added”.

45. **Test Case**

Title: Remove sales manager (when there is no such manager)

Description: The admin cannot delete the sales manager with invalid sales manager ID.

Precondition: Admin must be on the menu and the sales manager should not be in the system.

Test Steps:

- Open admin menu.
- Select ‘remove rental mamanger’.
- Enter the sales manager ID.
- Click remove.

Expected Result: The admin cannot delete the sales manager and the system warns: " There is no such sales manager."

46. **Test Case**

Title: Add transportation personnel (already added)

Description: The admin cannot add the transportation personnel previously added to the system again.

Precondition: The transportation personnel must be already added in the system.

Test Steps:

- Open menu.
- Select add transportation personel.
- Enter the ID, name, surname of the transportation personnel and enter the branch ID.
- Click add.

Expected Result: Admin cannot add the transportation personnel and the system warns: “Transportation personnel is already added”.

47. **Test Case**

Title: Remove transportation personnel (when there is no such personnel)

Description: The admin cannot delete the transportation personnel with invalid rental mamager ID.

Precondition: Admin must be on the menu and the transportation personnel should not be in the system.

Test Steps:

- Open admin menu.
- Select ‘remove transportation personnel’.
- Enter the transportation personnel ID.
- Click remove.

Expected Result: The admin cannot delete the transportation personnel and the system warns: " There is no such transportation personnel."

48. **Test Case**

Title: Add technician (already added)

Description: The admin cannot add the technician previously added to the system again.

Precondition: The technician must be already added in the system.

Test Steps:

- Open menu.
- Select add technician.
- Enter the ID, name, surname of the technician and enter the branch ID.
- Click add.

Expected Result: Admin cannot add the technician and the system warns: “Technician is already added”.

49. **Test Case**

Title: Remove technician (when there is no such manager)

Description: The admin cannot delete the technician with invalid rental mamager ID.

Precondition: Admin must be on the menu and the technician should not be in the system.

Test Steps:

- Open admin menu.
- Select ‘remove technician’.
- Enter the technician ID.
- Click remove.

Expected Result: The admin cannot delete the technician and the system warns: " There is no such technician."

Description: When the customer wants to rent a car, he/she makes a request.

Precondition: The customer must be login the system.

Test Steps:

- Open menu.
- Select rent request.
- Enter the brand, type and color of the vehicle.
- Click add request.

Expected Result: System gives a message: 'Your request has been forwarded to sales manager.'

50. Test Case

Title: Rent request (if customer’s rating is not enough)

Description: When the customer wants to rent a car, he/she makes a request. If the score is not sufficient, the sales manager does not approve the request.

Precondition: Customer rating should be low.

Test Steps:

- Open the sales manager menu.
- Select the ‘renting requests’.
- Select the request and check the customer rating.

Expected Result: Sales manager does not approve the customer rent request.

51. Test Case

Title: Rent request (if customer does not have the licence)

Description: If the customer does not have a driver's license, the car rental request is denied.

Precondition: Customer must be login the system.

Test Steps:

- Open the customer menu.
- Select the vehicle.
- Click the ‘send renting request’.

Expected Result: System send message ‘You do not have a license to drive this vehicle.’