Best Car Rentals

CS5200 – INTRODUCTION TO DATABASE MANAGEMENT SYSTEMS

Guna Chandrashekar and Divya Devaraj
NEU SEATTLE | FALL 2014

1 PROBLEM STATEMENT

A car rental or car hire agency is a company that rents automobiles for short periods of time (generally ranging from a few hours to a few weeks) and often complemented by a website allowing online reservations. It primarily serves people who require a temporary vehicle, for example those who do not own their own car, travelers who are out of town or owners of damaged or destroyed vehicles who are awaiting repair or insurance compensation. This is growing up to be a popular business and is at its busiest during the holiday season.

Our application will provide the customer with an online reservation for car rentals. So by just providing the date, location and period of rent, this application will list out all the deals on a click of a button in a single page and the customer can reserve the car they want.

2 PROBLEM SOLUTION

The application we have built (Best Car Rentals) allows customers to browse through a list of cars available for a given pick up date and drop off date and a location. The list of cars displayed are displayed based on their type (e.g. Economy, SUV, Minivan). For each category, a list of models are displayed. The customer can choose from a list of models (e.g. VW Beetle, Nissan Sentra) available as a drop down for each Car Type. The user can sort through list of cars based on their total price. They have to click on 'place order' button to place an order. The customer can also cancel their order anytime and update their profile information as well.

The application also has another user – the Admin. The admin can create an account for a customer and he can also create another admin account. He can browse through the list of orders and cancel any order. He can add, update and delete a car model. Also he can add and delete locations and refresh the data from the api at any given point in the day. This ensures the latest price details are in sync with our application. Hence our application provides an online reservation site for car rentals for a given location in US.

3 ARCHITECTURE

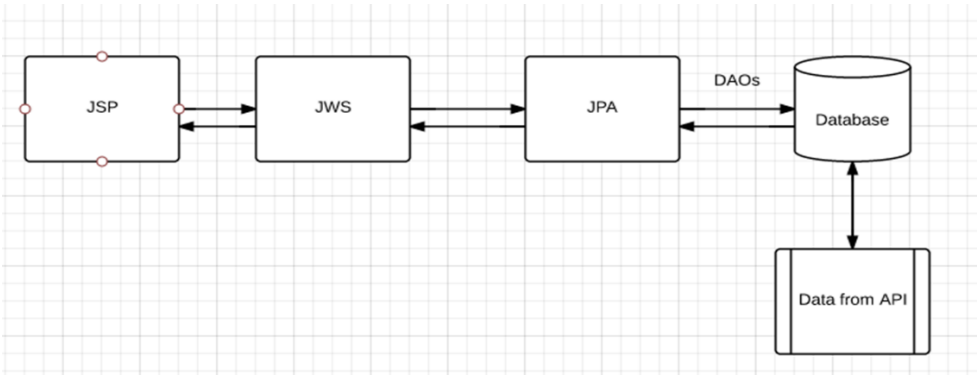
Below listed is the technologies that we have used:

- Type : Web Application
- Technology : JAVA and J2EE(server-side), JavaScript, JQuery and Bootstrap(CSS) (client-side), JSP
- Web Services : JWS
- ORM : JPA
- Server : Apache Tomcat 7
- IDE: Eclipse Luna
- Operating System : Windows 8.1
- Database : MySQL
- Data Source : Hot-wire API (XML format)
- Parsing the API : XSLT and XML
- Source Control : GitHub

The data for the rental cars available and the daily prices are fetched from the Hotwire API. This API provides a list of car types available for a given location, pickup place and time and drop-off place and time in XML format. The API does not provide JSON format. We parse the returned XML and fetch the elements that we require and put it into another XML. This XML data is then used to populate the Database. The results for the list page are populated from our database. The admin user can refresh the data from the API at any time. This ensures that the daily changing price rates are always updated to our system. The customer and admin information are all stored in our database.

The input is obtained from the customer through JSP is passed as JSON to Java Webservice, which in turn calls the JPA entities to update/fetch the data from the database and vice versa. The data from the XML is directly populated into our database.

The architectural diagram is given in the next page.



4 API USED

We have used hotwire API for our application. It delivers data describing rental car shopping results similar to those that can be obtained when shopping for rental car rates on Hotwire.com. It requires 5 parameters. The location, pick up date and time and drop off date and time.

The API is accessible by an API Key: a72bv5sr5g7vukdffta6v3z2

Given below is the URL for API documentation.
http://developer.hotwire.com/docs/read/Rental_Car_Shopping_API

The API provides data in the XML format. The XML data is parsed using JAXB and then only selected information from the XML is extracted using XSLT and written onto a file. The selected data from this file is un-marshalled by setting the JAXBContext and creating unmarshaller which then provides us with Java representation for the XML data. We then make use of JPA to store the data into database.

The data in the database is updated with the current rental information when Admin clicks on “Refresh Data from API” button or when a new Location is added by the admin.

5 TECHNOLOGY STACK

As mentioned in the architecture section, the stack that we have used is similar to WAMP. But we have used JAVA as our programming language instead of PHP, PERL or Python.

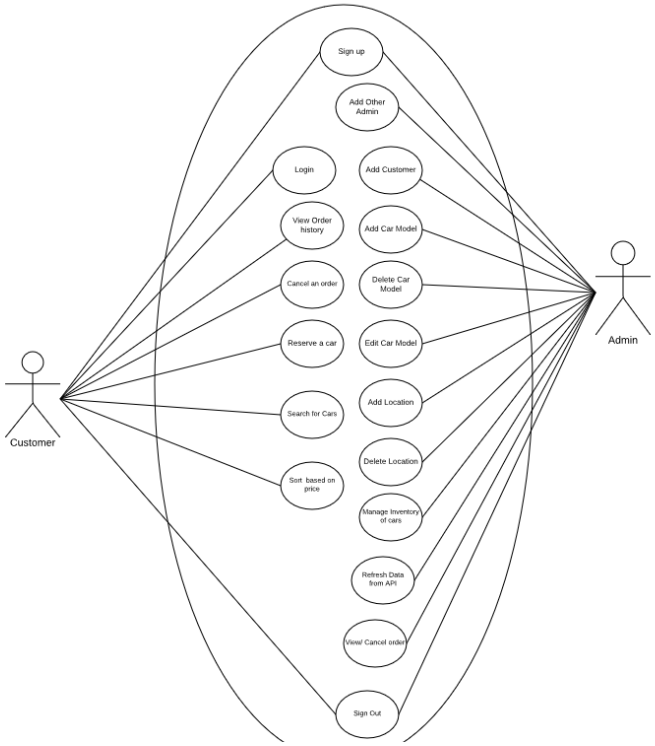
STACK : Windows (OS), APACHE (server), MySQL (database) and Java/J2EE (Programming Language)

6 USE CASES

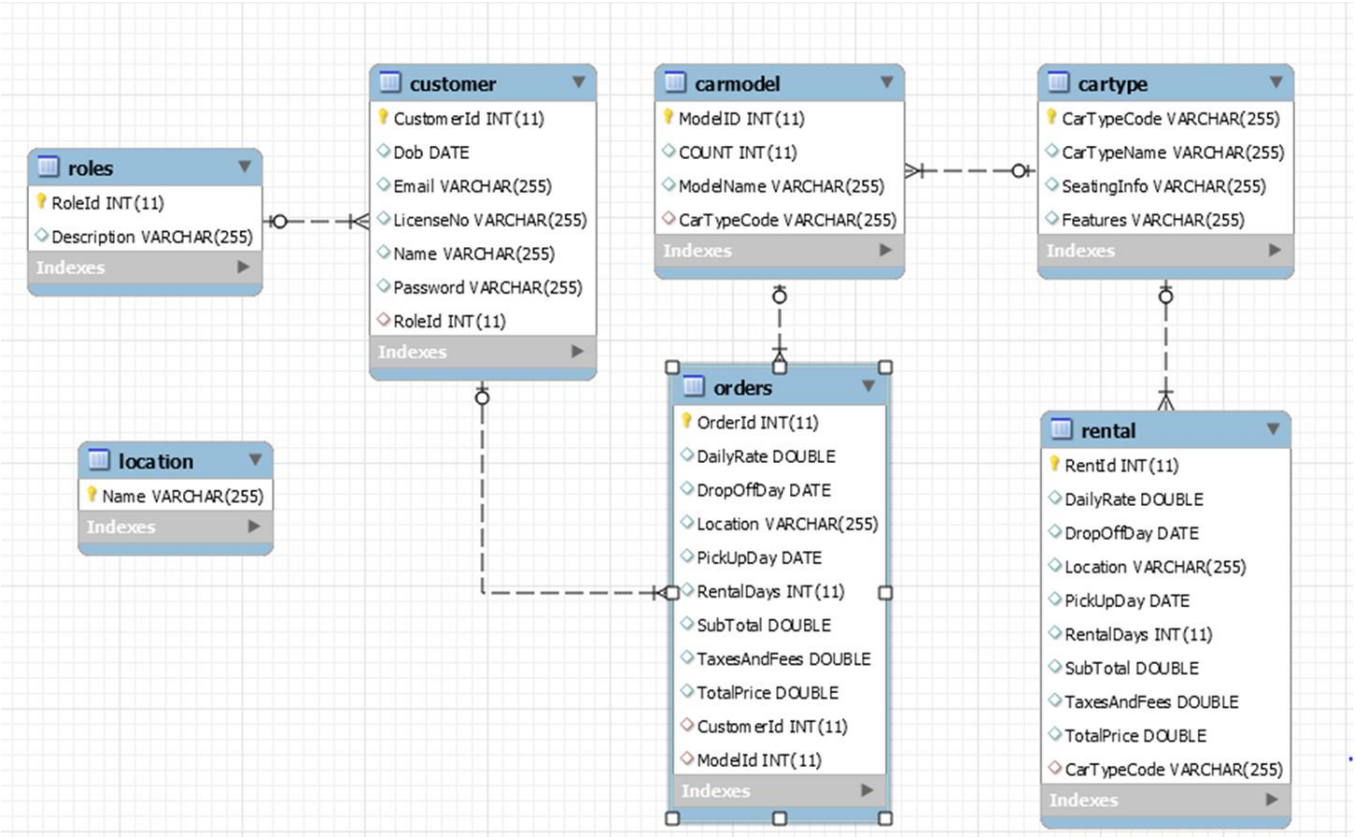
- Login validations
- Admin Login
- Admin Login -> Create account for Users
- Admin Login -> Create account for Admin
- Add Location
- Delete Location
- Add Car Models
- Delete Car Model
- Edit/ Update Car Models
- Admin -> View Order
- Admin -> View Order -> Cancel Order
- User Sign up
- User Login
- Edit/ Update User Profile
- Search for available cars
- Display available cars types along with daily rent and all rent details
- Display car types with an option to sort data Low to High

- Display car types with an option to sort data High to Low
- Display available car models based on count of the cars
- Place orders
- View Order History
- User logout

Please refer to the use case document for detailed explaination



7 DATA MODEL – UML CLASS DIAGRAM



The UML class diagram shows the list of tables present in our application.

CarType table contains CarTypeCode, CarTypeName, Seating Information and Feature of a particular car type. CarTypeCode serves as a Primary Key for CarType table as it is unique for each result set returned by the API. **Rental** contains all information about the rent for a particular CarType like Daily Rate, Taxes and Fees, Total Price, Location, Pickup Day and Drop-off Day. **CarModel** has modelId, model name and count of a particular model which belongs to a particular CarType. **Customer** table contains information about the customer such as Name, Date of Birth, License Number, Email, RoleId and password. Email and Password is used to log into the application. Password field is used to store hashed password. An entry is made into **Order** table when the user clicks on "Place Order". It contains orderId, DailyRate, Drop-off day, Pickup day, Rental days, Location, Taxes and fee, SubTotal, Total. OrderId serves as the Primary key. CustomerId and ModelId are foreign keys in the table. **Role** table is an enumeration table it contains one entry for Admin with roleId 1 and another entry for Customer with roleId 2. Each customer in the Customer Table will belong to one of the two roles i.e Admin or a Customer. **Location** contains information about location. This table is updated when the admin adds a new location.

There one-to-one mapping between CarType and Rental Table. There is one-to-many mapping between CarType and CarModel Table. There is one to many mapping between CarModel and Orders table and Customer and Orders table.

8 FEATURES

- Http Session based on cookies is provided for every User to make the application secure.
- Our website provides secure sign up and log in. Password is protected using SHA-256(not broken till date). The hashed password is then stored in the database.
- Insert, delete new locations.
- Insert, delete, select, or update car models.
- Data from API can be refreshed at any given time.
- Sort through list of cars available based on Price
- Validations added at UI and database level.
- Data between pages is not passed in the URL making the application all the more secure.
- Provides all information about a car in the list page

9 TO RUN APPLICATION

Create Tables

CREATE TABLE `cartype` (`CARTYPECODE` varchar(255) NOT NULL, `CARTYPENAME` varchar(255) DEFAULT NULL, `SEATINGINFO` varchar(255) DEFAULT NULL, `FEATURES` varchar(255) DEFAULT NULL, PRIMARY KEY (`CARTYPECODE`))

CREATE TABLE `rental` (`RentId` int(11) NOT NULL AUTO_INCREMENT, `DailyRate` double DEFAULT NULL, `Location` varchar(255) DEFAULT NULL, `RentalDays` int(11) DEFAULT NULL, `SubTotal` double DEFAULT NULL, `TaxesAndFees` double DEFAULT NULL, `TotalPrice` double DEFAULT NULL, `CarTypeCode` varchar(255) DEFAULT NULL, PRIMARY KEY (`RentId`), KEY `rental_ibfk_1` (`CarTypeCode`), CONSTRAINT `rental_ibfk_1` FOREIGN KEY (`CarTypeCode`) REFERENCES `cartype` (`CARTYPECODE`) ON DELETE NO ACTION ON UPDATE NO ACTION)

INSERT INTO `cartype` VALUES ('CCAR','Compact','2 adults, 2 children','Automatic Transmission, Power Steering, Air Conditioning, Air Bags, AM/FM Stereo'),('ECAR','Economy','2 adults, 2 children','Automatic Transmission, Air Conditioning, Air Bags, AM/FM Stereo'),('FCAR','Full Size','5 adults','Automatic Transmission, Power Steering, Air Conditioning, Air Bags, Anti-Lock Brakes, Cruise Control, AM/FM Stereo'),('FFAR','Full Size SUV','7 adults','Automatic Transmission, Power Steering, Air Conditioning, Air Bags, Cruise Control, AM/FM Stereo, CD Player'),('FRAR','Full Size SUV Type 2','6 adults','Automatic Transmission, Power Steering, Air Conditioning, Air Bags, Cruise Control, AM/FM Stereo, CD Player'),('ICAR','Midsize','4 adults','Automatic Transmission, Power Steering, Air Conditioning, Air Bags, AM/FM Stereo'),('IFAR','Midsize SUV','4 adults','Automatic Transmission, Air Conditioning'),('LCAR','Luxury','5 adults','Automatic Transmission, Power Steering, Air Conditioning, Air Bags, Anti-Lock Brakes, Cruise Control, AM/FM Stereo, CD Player'),('MVAR','Minivan','7 adults','Automatic Transmission, Power Steering, Air Conditioning, Air Bags, Anti-Lock Brakes, Cruise Control, AM/FM

Stereo'),('PCAR','Premium','5 adults','Automatic Transmission, Power Steering, Air Conditioning, Air Bags, Anti-Lock Brakes, Cruise Control, AM/FM Stereo, CD Player'),('SCAR','Standard','2 adults, 2 children','Automatic Transmission, Power Steering, Air Conditioning, Air Bags, Anti-Lock Brakes, AM/FM Stereo'),('SFAR','Standard SUV','5 adults','Automatic Transmission, Power Steering, Air Conditioning, Air Bags, Anti-Lock Brakes, Cruise Control, AM/FM Stereo, CD Player'),('SPAR','Standard Pickup truck','4 adults','Automatic Transmission, Power Steering, Air Conditioning, Air Bags, Cruise Control, Anti-Lock Brakes, AM/FM Stereo'),('STAR','Convertible','4 adults','Automatic Transmission, Power Steering, Air Conditioning, Air Bags, Cruise Control, Anti-Lock Brakes, AM/FM Stereo'),('SXAR','Special Car Type 2','4 adults','Automatic Transmission, Air Bags, Cruise Control, CD Player'),('XXAR','Special Car','4 adults or more','Automatic Transmission, Air Conditioning, Air Bags, AM/FM Stereo');

CREATE TABLE `carmodel` (`ModelID` int(11) NOT NULL AUTO_INCREMENT, `COUNT` int(11) DEFAULT NULL, `ModelName` varchar(255) DEFAULT NULL, `CarTypeCode` varchar(255) DEFAULT NULL,PRIMARY KEY (`ModelID`), KEY `carmodel_ibfk_1` (`CarTypeCode`), CONSTRAINT `carmodel_ibfk_1` FOREIGN KEY (`CarTypeCode`) REFERENCES `cartype` (`CARTYPECODE`) ON DELETE CASCADE ON UPDATE CASCADE)

CREATE TABLE `roles` (`RoleId` int(11) NOT NULL AUTO_INCREMENT, `Description` varchar(255) DEFAULT NULL, PRIMARY KEY (`RoleId`))

INSERT INTO `roles` VALUES (1,'Admin'),(2,'Customer');

CREATE TABLE `location` (`Name` varchar(255) NOT NULL DEFAULT '', PRIMARY KEY (`Name`))

CREATE TABLE `customer` (`CustomerId` int(11) NOT NULL AUTO_INCREMENT, `Dob` date DEFAULT NULL, `Email` varchar(255) DEFAULT NULL, `LicenseNo` varchar(255) DEFAULT NULL, `Name` varchar(255) DEFAULT NULL, `Password` varchar(255) DEFAULT NULL, `RoleId` int(11) DEFAULT NULL, PRIMARY KEY (`CustomerId`), KEY `customer_ibfk_1` (`RoleId`), CONSTRAINT `customer_ibfk_1` FOREIGN KEY (`RoleId`) REFERENCES `roles` (`RoleId`) ON DELETE CASCADE ON UPDATE CASCADE)

CREATE TABLE `orders` (`OrderId` int(11) NOT NULL AUTO_INCREMENT, `DailyRate` double DEFAULT NULL, `DropOffDay` date DEFAULT NULL, `Location` varchar(255) DEFAULT NULL, `PickUpDay` date DEFAULT NULL, `RentalDays` int(11) DEFAULT NULL, `SubTotal` double DEFAULT NULL, `TaxesAndFees` double DEFAULT NULL, `TotalPrice` double DEFAULT NULL, `CustomerId` int(11) DEFAULT NULL, `ModelId` int(11) DEFAULT NULL, PRIMARY KEY (`OrderId`), KEY `CustomerId` (`CustomerId`), KEY `ModelId` (`ModelId`), CONSTRAINT `orders_ibfk_1` FOREIGN KEY (`CustomerId`) REFERENCES `customer` (`CustomerId`) ON DELETE CASCADE ON UPDATE CASCADE, CONSTRAINT `orders_ibfk_2` FOREIGN KEY (`ModelId`) REFERENCES `carmodel` (`ModelID`) ON DELETE NO ACTION ON UPDATE CASCADE)