**IS2103 Enterprise Systems Server-side Design and Development**

**PAIR PROJECT**

****

**Pair Group: PP07**

**Merlion Car Rental (MCR) Management System**

Nelson Choo Jia Wei (A0230335U)

Leo Hernando Tan (A0240787W)

**High-level Systems Architecture**

**Software Components**

**SOAP Web Services**

**CaRMS**

**Management**

**Client**

**CaRMS**

**Reservation**

**Client**

**Holiday**

**Reservation**

**System**

**Database**

Organisational Boundary

**Merlion Car Rental**

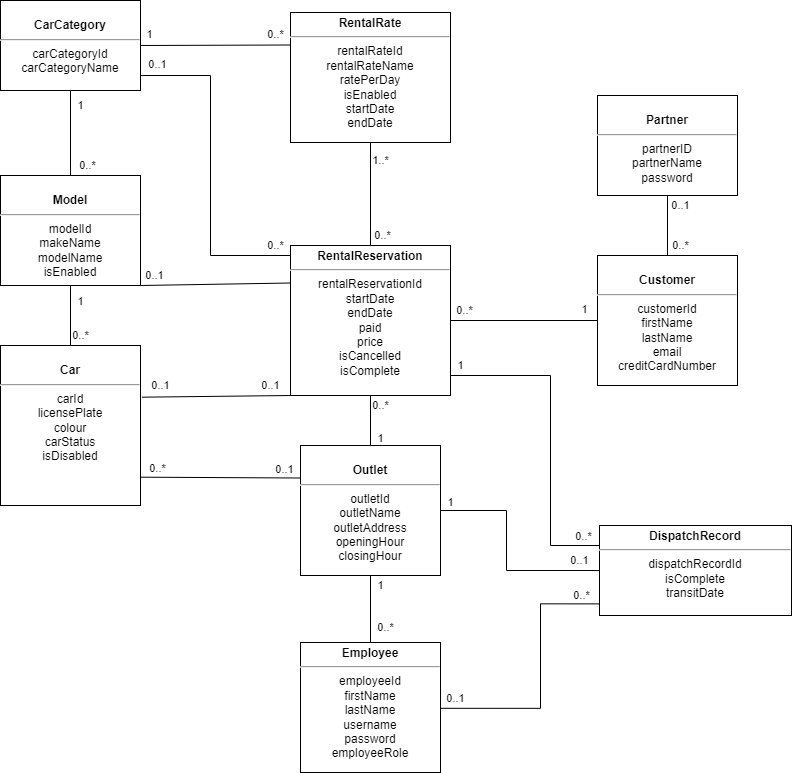
**Holiday.com**

The CaRMS Management Client and CaRMS Reservation Client are Java EE clients while the Holiday Reservation System is a Java SE application that will interact with the overall system through SOAP Web Services.

The software components of the system comprise the Enterprise JavaBeans (EJB) and Java Persistence API (JPA) technologies that was used in conjunction with the platform-independent object-oriented query language—Java Persistence Query Language (JPQL)—as a Relational Database Management System (RDBMS) to make queries to our database of entities.

All client applications are Command-line Interface (CLI) only.

# Logical Data Model (Class Diagram)



**Rationale of General Structure**

We designed the entire system centred around the *RentalReservation* entity as it is the entity class with the greatest number of relationships with all the other classes. The nature of these relationships are indicated by the multiplicity value as stated in the above class diagram.

**Rationale of Attributes Used**

All entity classes have their respective IDs as a Long object.

For price of *RentalReservation* entity, we used the BigDecimal object because it would help maintain accuracy when dealing with money.

All dates are stored as simple Java Date objects as it is easier to manipulate. We used DateFormatter and GregorianCalendar when it was necessary to receive input from the client side and parse it in a readable format. When dealing with the corresponding web service methods, we used a XMLGregorianCalendar format instead.

We used a *CarStatusEnum* enumeration to record the state of a car as a Boolean would not be appropriate (more than two states). The three possible states are AVAILABLE, ON\_RENT and REPAIR.

We used an *EmployeeRoleEnum* enumeration to record the possible EmployeeRoles. The four possible states are SALES\_MANAGER, OPERATIONS\_MANAGER, CUSTOMER\_EXECUTIVE and EMPLOYEE.

**Description of Relationships**

The entity class relationships were modelled after the business requirements stated in the project specifications document. Each entity class has its corresponding session bean (all stateless session bean methods) to perform CRUD tasks when necessary as well as some more complex use cases (such as retrieveCheapestRentalRate and searchCarByCategory) which requires traversal across multiple relationships.

The multiplicity notations indicate the relationships that we have developed and are correspondingly annotated in the submitted code (e.g., @ManyToOne, @OneToOne).

**Business Rules**

**CaRMS Management System**

Allocation of cars – Done daily at 2am

Generate dispatch records – Transit can be done outside of outlet operating hours

Delete rental rate – Does not need to be deleted even if not in used

**CaRMS Reservation System**

Register as Customer – Valid data input

Search for Car – Can only be done by either category or model

Reserve Car - If the number of cars queried from the database is more than the number of rental reservations queried, the user is able to reserve a car.

**Holiday Reservation System**

Search for Car – Can only be done either by category or model

Partner Reserve Car - If the number of cars queried from the database is more than the number of rental reservations queried, the user is able to reserve a car.