**C.A.R - GRASP Implementations**

* **Creator** – Classes that edit other objects have creator status
  + The RepresentativeController is the creator for *Car* objects
  + The AdminController is the creator for *User* objects
  + The UserController is the creator for *Rental* objects
* **Low Coupling** – classes communicate minimally with each other
  + *Cars* have no knowledge of *Users*
  + *Cars* and *Representatives* have no knowledge of *Receipts*
  + *Users* (Customers) can only view *Cars*
  + *Receipts* have no knowledge of *Cars* or *Representatives*
  + *Receipts* only show data from *Users* and *Rentals*
* **High Cohesion** – classes have narrow and specific responsibilities
  + Controllers take input and create appropriate objects
  + *User* (and subtypes), *Receipt*, *Rental*, *Car* store only data pertinent to the individual object
* **Controller** – handles input and object creations
  + *RepresentativeController* handles input for new *Car*
  + *AdminController* takes input for new *User*
  + *UserController* takes input for new *Rental*
* **Polymorphism** – Subclasses are implemented to expand functionality of Superclasses
  + *Representative* extends *User*; it is a specialized type of *User*, with broader access to data in the system
  + *Administrator* extends *Representative*; it is a subtype of *Representative*, who has clearance to edit and view larger amounts of data
* **Indirection** – Classes call other classes to accomplish tasks
  + User calls *UserController* to view *Cars*
  + Administrator calls *AdminController* to edit *Users*