The Software Architecture Diagram for the Car Rental System starts from the user. Regardless of the device used, users are required to login to access the system. If enabled, the two-factor authorization will require users to verify their login with a 3rd party authentication app before they can proceed with the rest of the system. In addition to a valid login, users will need to have their rental requirements verified with the user information database containing the user's name, address, email, password, rental agreement, rental history, credit score, and insurance information. The user can view their user page and update it as necessary, and also add a payment method through the system's payment API. Once the user's rental requirements have been deemed valid by the system, they will be allowed to browse and rent cars. While browsing through the catalog of cars available to rent, a user can browse by nearest location or by specifications. Users can also set a customer review if their rental history shows they have previously rented that car. When renting a car, users can view the make, model, colour, type, availability, condition, mileage, flat day fees, and extra mileage fees of that car. The car also has a GPS service that tracks its location for both the user and the employees to know. In addition to seeing any given car's location, employees can also check the status of any of those cars. Employees can also access the rental agreements of any user and update the car directory.

Our user class consists of the user's basic login information: the user's name, email, and password stored as Strings. The user class also contains two functions to change the email and password associated with a user. The user class branches into the customer and employee classes to differentiate the two different types of users. The employee class is simpler than the customer class, as the employee class only stores the employee ID as an integer and the location of that employee's workspace as an object of the rentalLocation class. Besides that, the employee class contains three functions: one to check the customer information, one to check the available cars of a rental The rentalLocation class stores the available cars as rentalCar objects, the number of available cars as an integer, the address of the lot as a String, the number of employees as an integer, and the rental fee as a double. The rentalLocation class also contains a function to retrieve the hours of operation. A rentalCar object contains the make, model, colour, body, and fuel type of the car as Strings, and the number of seats and the odometer as integers. A rentalCar object also stores the miles per gallon and the cost per mile as doubles, and stores the year and unavailable dates as their own separate objects. The rentalCar object can also call functions to check the availability and information of a car. Back to the different types of users, a customer class contains the age of a customer as an integer, and if that customer has signed the rental agreement and has insurance as booleans. A customer class also contains the payment information and the current location of a customer, as well as their rental history in the forme of transaction objects. A customer class also contains the functions that can search for the customer's current location, get the distance from a customer to the rental location, check if the customer has signed the rental agreement or updated their payment, and get the customer's rental history. The transaction class, the last class that has not been gone over, stores the assigned customer, desired car, rental location, payment method, and total cost of a specific transaction. The transaction class also contains the functions that allow a customer to cancel their transaction, or allow the system to check the customer's rental requirements.