Parking Garage Management System:

You need to design and implement a parking management system that will allow the user to check in and check out a vehicle according to the following demands:

1. The garage services only 6 vehicle types, which are classified into 3 categories:
2. Motorcycle – Class A
3. Private – Class A
4. Crossover – Class A
5. SUV – Class B
6. Van – Class B
7. Truck – Class C
8. There are 3 ticket types. Each type has its own limitations and benefits:
9. VIP –

Lots: 1-10

Dimensions: no dimension limit

Classes: all

Cost: 200$

Time limit: no limit

1. Value –

Lots: 11-30

Dimensions: Height 2500, Width 2400, Length 5000

Classes: A, B

Cost: 100$

Time limit: 72h

1. Regular –

Lots: 31-60

Dimensions: Height 2000, Width 2000, Length 3000

Classes: A

Cost: 50$

Time limit: 24h

* When checking in a vehicle, the user needs to input:

Name, License plate ID, Phone, Ticket type, Vehicle type, Vehicle height, Vehicle width, Vehicle length.

There should be a check (on server side) to verify that the vehicle's dimensions are suitable with the ticket type. If not, the user should be given an option to trade his ticket for a suitable one (show him the cost difference). If yes, a parking lot should be assigned for the vehicle.

Each parking lot may be assigned to only 1 vehicle at any given time.

* When checking out a vehicle, the user needs to input vehicle license plate ID.
* Create a web ui (using Angular 2+) to support all functionality.
* You need to store the data regarding the entering vehicles in a local SQL database file (mdf).
* You **don't** need to store the data regarding the ticket types, their prices and allowed dimensions. This should be stored in a hierarchical class architecture.
* Write a stored procedure that receives as input a ticket type and returns all vehicles that are currently parked in the garage having this particular ticket type.
* Add a capability to show the user the current state of the parking garage. Each lot and the occupying vehicle ID.

Bonus:

* Add a capability to asynchronously check in 5 randomly created vehicles, with random data or some hardcoded data. doesn't matter. At the end of the process display the results to the user. This checks your asynchronous programming ability (all the async & await stuff). This should be done by pressing a single button on the client side. The implementation should be done in the server side and should be asynchronous.
* Add unit testing

Exercise grading is based on:

1. Correct functionality
2. Good architecture and usage of design patterns
3. Quality of the web ui (design should be simple and pleasant)
4. Coding style

Guidelines:

Don't implement business logic in the DB other than what is asked in the exercise.

You may or may not add ticket to dimensions validation on the client side but you must do the validation on the server side.

Client side should be done with Angular 2+. Server side should be a WebApi.

Submission instructions:

Submit the solution files and a file for creating the database from (schema and data). Use online storage.