Prototype Database Design

# Project Description

Our database management system will have the primary functionality to store information pertaining to our Capstone project, which is the [Automated Parking Management System](https://courses.torontomu.ca/d2l/le/content/794625/viewContent/5303682/View). With the project summary and expected specifications that are defined in the document provided to us, our task remains to then break down the higher level components that execute higher order functions to smaller subsystems. Here I (Yadu) propose a database ecosystem based upon the initial requirements our team has come up with.

Our parking system will have pre-registered **USERS** who can either **BOOK** a parking spotor have anonymous users who can also book parking spots. Once the user checks in through the proposed “QR” code method, two situations can be methodically expressed. The pre-registered user’s potential experience vs the anonymous users potential experience can be described.

Pre-registered User-Case Scenario:

The user once authenticated will be notified of the PARKING SPOT location that they booked for. Once they have parked, they can confirm their parking within our web app to “sign in”. The sign in process is further verified by our on site Cameras as part of our system

Important Notes:

# 

# 

# Database Schema:

This section will cover the proposed database schema, which will be used to build the database and tables. The schema helps to describe the overall structure of the database, what attributes/fields each table will have etc..

**Note**: I have used the variable *N* denoting an integer number. We can declare the size of the variable to optimize for database storage.

**USERS** Table

* ***UserID*** (type: INTEGER(*N1*)): Primary key
* ***Name*** (type: VARCHAR(*N2*))
* ***LicensePlate*** (type: VARCHAR(*N3* = 7)): *N2* here can be 7 because 7 happens to be the longest a license plate can be. Special exemptions may apply so, the number is subject to change

**BOOKING** Table

* ***UserID*** (type: INTEGER(*N1*)): Foreign Key
* ***BookingID*** (type: VARCHAR(*N2*)): Primary Key
* ***LotId:***(type: VARCHAR(*N2*)): Foreign Key
* ***EntryTIme*** (type: DATE(MM-DD-mm-ss)): The variable type is subject to change .
* ***ExitTime*** (type: DATE(MM-DD-mm-ss)): The variable type is subject to change.
* ***Price***(type: INTEGER(*N4*)). We can add an upper constraint for the price if needed.

**SENSOR\_DATA** Table

* ***SensorType*** (type: VARCHAR(*N1*)): Can implement a *CHECK* condition to verify sensor Type.
* ***Date***(type: DATE(MM-DD-mm-ss))
* ***SensorData***(type: VARCHAR(*N3*)): The Sensor data column can vary depending on the sensor and the type of data that needs to be stored for x purpose. Hence string format can be sufficient.

**PARKING\_LOT** Table

* ***LotId:***(type: INTEGER(*N1*)): Primary Key
* ***Name*** (type: VARCHAR(*N2*))
* ***RegisterDate*** (type: DATE(MM-DD-mm-ss)).