A picture containing shape

Description automatically generated

***Code Description***

1. **Modules**

**a).Data Records**

**Staff records**: - It helps to provide details of staff that uses the Vehicle parking management System. It provides the descriptions of staffs like:  
-Staff first, middle and last name  
-Address   
-Contact Number   
-Gender.

**User Records**: - This record helps for the authorization for using Vehicle Parking Management System. It Provides the Username and Password for the User (staff).It also includes the level of authority that means it separates the normal users and administrator.

**Vehicle Records**: - This most important record which focuses in our Vehicle Parking Management System. It stores the essential Vehicle records like:  
-Vehicle Number  
-Vehicle Type  
-Vehicle Entry Time  
-Vehicle Exit Time

**b).Reports**

**Vehicle Parking Detail**: - This report is very essential in this system. This report provides a brief summary of vehicle activities. It shows the overall Entry and Exit time. It shows the User at time of Entry and Exit .It also provides the facility for examining the total vehicle details according to date wise.

**Transaction Detail**:-This report will show the Transaction between the customer and the System. . It shows the cost of the vehicle after using the facility of parking. It will show the number of transaction by date wise. It will also have User at time of the Transaction .

1. **Information gathering**

Information gathering is done by interviewing the users and reviewing the existing documents. For the development of Parking management system a lot of research and important input from various website and application user was needed. Hence the following questionnaires were provided to them and hence te need for our website arises

* **Interviewing the users:**
* What are the difficulties you are facing in the existing system ?
* What all new things you want to be included in the proposed system ?
* In what way you are storing your information ?
* Who all are the users of the system ?

1. **User Requirement**

* Need for an application that makes communicating easy and comfortable.
* An application that enables user to park a vehicle with safe and secure.
* Need for an application that is easy to use and widely available and hence a web application
* Handling all functions done with organization in a computerized manner.
* Allowing the user to park the vehicle directly.

1. **Functional Requirement**

* Admin need to enter all details for registration.
* Admin need to insert all details about customer and vehicle.
* Admin need to save all the details of customer and vehicle.
* Admin can retrieve the details of customer.
* Admin must generate a report for payment.

1. **Non-functional Requirement**

* Usability: These website has appropriate user interface and adequate information to guide the user in order to use the website.
* Portability: The website is portable as it is online website running across the net
* Flexibility: It is very flexible
* Security: This website provide user and authentication so that only the legitimate user are allowed to use the website
* Maintainability: These website is capable to secure the data and easily retrieve the data.
* Scalability: These system can further modified in future.

**Main Code Description**:

This repository gives an overview of how we can design a basic parking lot in Python. It creates parking lot with given number of slots. The cars follow Greedy approach while being parked in the slots.

ParkingLot.py script defines the following functions -

**create\_parking\_lot n**  - Given n number of slots, create a parking lot

**park car\_regno car\_color** - Parks a vehicle with given registration number and color in the nearest empty slot possible. If there are no more empty slots available, it shows a message "Sorry, parking lot is full".

**status** - Prints the slot number, registration number and **color** of the parked vehicles.

**leave x** - Removes vehicle from slot number x

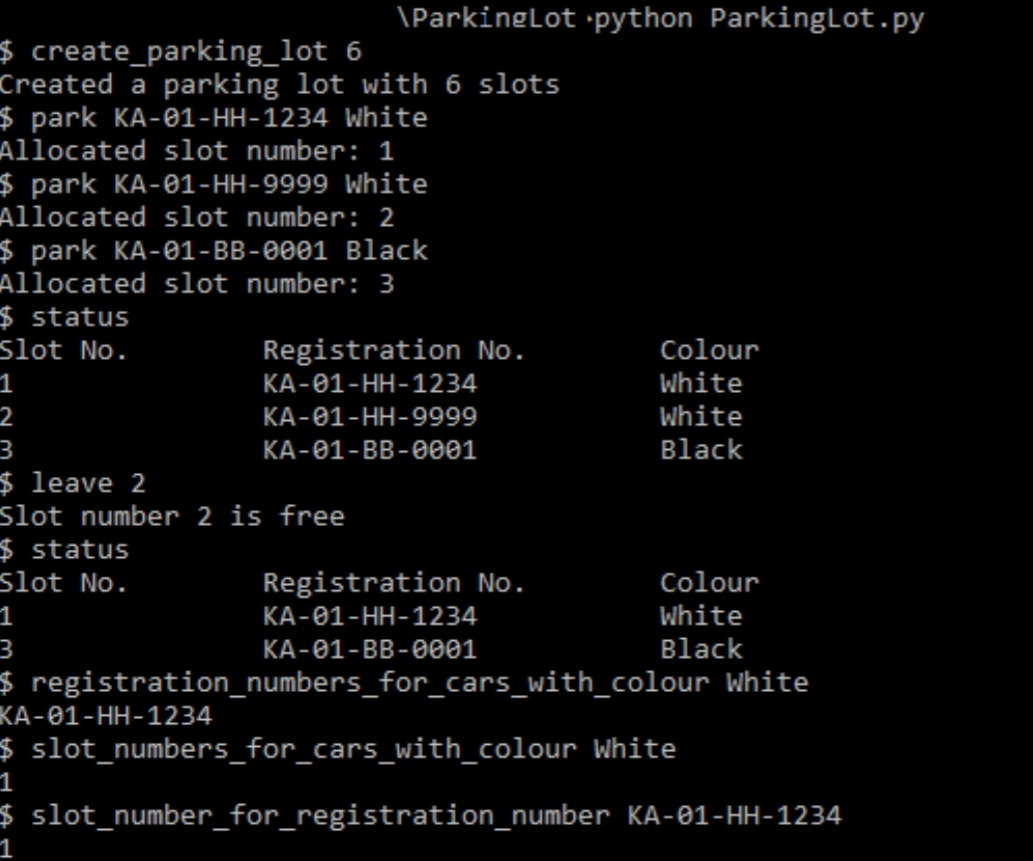
There are few query functions to retrieve slot number from registration number of car, get registration numbers of cars with particular **color** etc.

ParkingLot.py can be run through shell or through file containing test cases. An example file run\_test\_case.txt has been provided in repo.

I have followed this approach while designing this. **test\_parking\_lot.py** uses **unittest** module of python. Here 6 test cases are written in order to test **each functionality** mentioned in ParkingLot.py

Vehicle.py is a separate class where we can define the type of vehicles that can be parked. As of now, it only contains class Car.

**Output:**



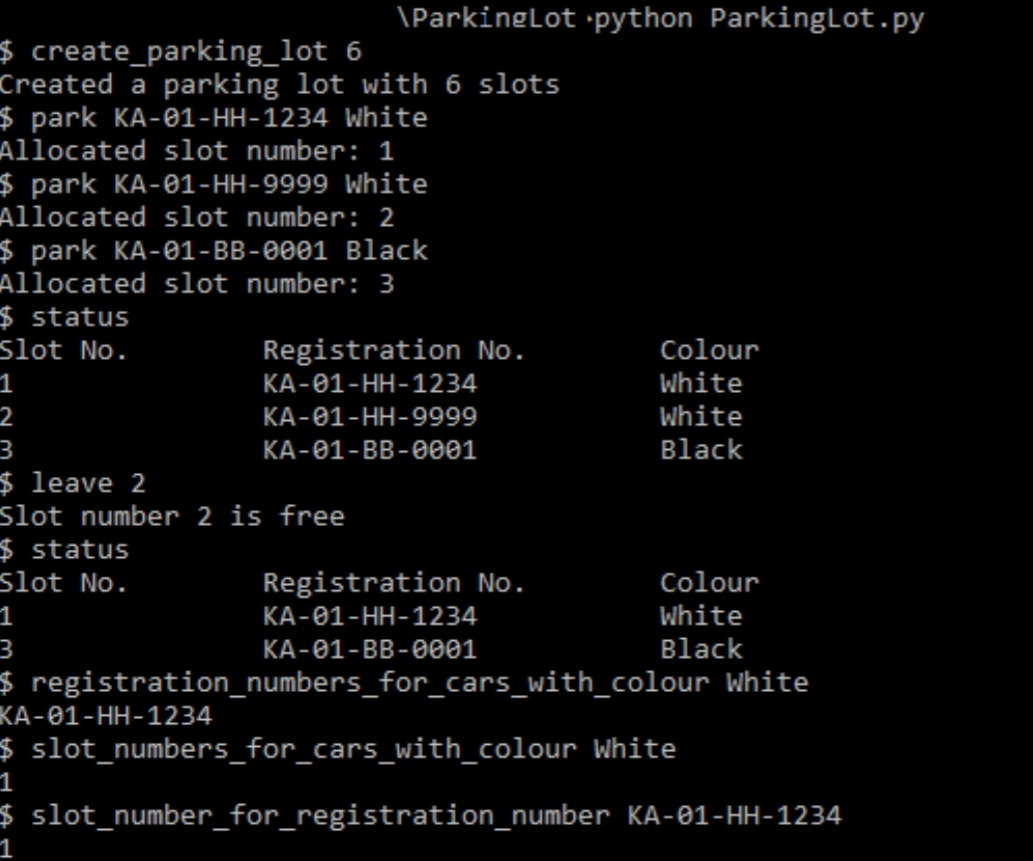
**Set Up:**

* **To create my own Parking Lot –**

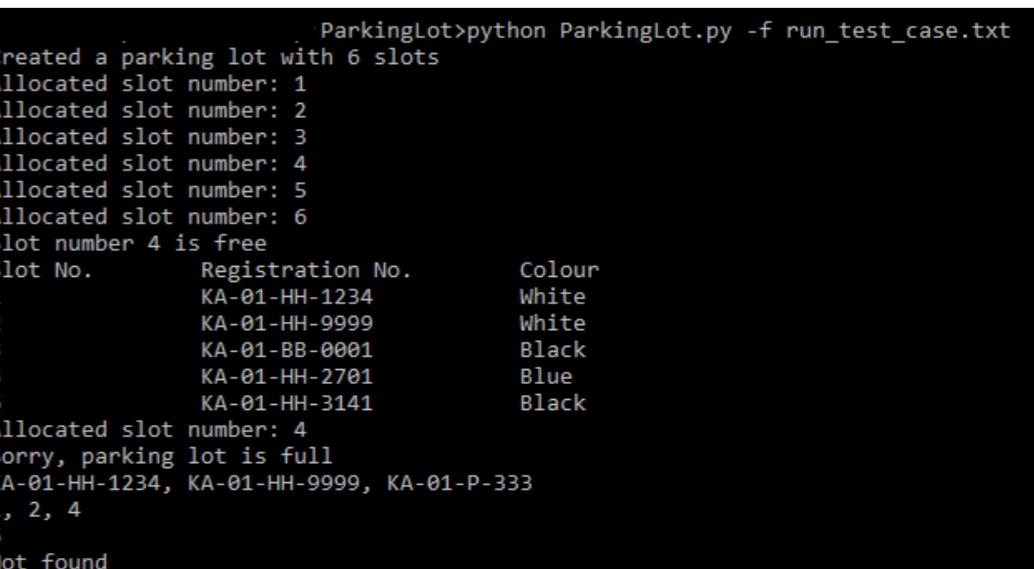
**1. Clone the repository**

**2. Run python ParkingLot.py to run without input test case file. This opens a shell where you can write your commands like -**

****

****

* **To run with a file execute `python ParkingLot.py -f run\_test\_case.txt. You can modify the test cases**

****



* You can also run the test cases separately as python **test\_parking\_lot.py.** This runs the 6 test cases written in file. This is very useful when you want to create your own function and test it simultaneously.

