

APPENDIX 1

Parking Management System

END TERM REPORT

by

Gaurish Sharma, Shashwat Tiwari, Rizul Sharma

Section: K19QW

Roll Numbers:52,42,55 respectively



**Department of Intelligent Systems,
School of Computer Science Engineering,
Lovely Professional University, Jalandhar
November, 2020**

APPENDIX 2

Student Declaration

This is to declare that this report has been written by me/us. No part of the report is copied from other sources. All information included from other sources have been duly acknowledged. I/We swear that if any part of the report is found to be copied, I/we are shall take full responsibility for it.

Signature:

Name: Gaurish Sharma, Shashwat Tiwari, Rizul Sharma

Roll Number: 52,42,55

Place: Jalandhar, Punjab

Date: 28/10/2020

APPENDIX 3

TABLE OF CONTENTS

TITLE: Parking Management System

PAGE NO: 3

1. Appendix 1	1
1.1 Appendix 2	2
1.2 Appendix 3	3
1.3 Appendix 4	4
2 Background and objectives of project assigned	5
2.1 Description	6
2.2 Role of each member	9
2.3 Implementation	10
2.4 Technologies and Framework to be used	11
2.5 SWOT Analysis	12
2.6 References	13

APPENDIX 4

BONAFIDE CERTIFICATE

Certified that this project report “Parking Management System” is the bonafide work of “Gaurish Sharma, Shashwat Tiwari, Rizul Sharma” who carried out the project work under my supervision.

<<Signature of the
Supervisor>>(Due to
Covid 19, signature is
exempted)

<<Dhanpratap Singh>>

<<Academic
Designation>>

<<UMS ID: 25706>>

<<School of Computer
Science and
Engineering>>

BACKGROUND AND OBJECTIVES OF PROJECT ASSIGNED

Background: In this project we have used Python 3. Python is a general-purpose interpreted, interactive, object-oriented, and high-level programming language. It was created by Guido van Rossum during 1985- 1990. Like Perl, Python source code is also available under the GNU General Public License (GPL). Python is named after a TV Show called "Monty Python's Flying Circus" and not after Python-the snake.

Python 3.0 was released in 2008. Although this version is supposed to be backward incompatible, later on many of its important features have been backported to be compatible with version 2.7.

Why to Learn Python 3?

Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages.

Python is a MUST for students and working professionals to become a great Software Engineer specially when they are working in Web Development Domain. I will list down some of the key advantages of learning Python:

- **Python is Interpreted** – Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.
- **Python is Interactive** – You can actually sit at a Python prompt and interact with the interpreter directly to write your programs.
- **Python is Object-Oriented** – Python supports Object-Oriented style or technique of programming that encapsulates code within objects.
- **Python is a Beginner's Language** – Python is a great language for the beginner-level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.

Objectives: The objectives of this project are as follows:

- The Project should be a Parking Management System where there is a certain number of slots in the parking lot that can be filled.
- The Project should have the ability to store details such as registration number and the color of the car.
- The Project should have the ability to check the status of the parking lots and see which lots are filled and which are not filled.
- The Project should have the ability to retrieve certain details with certain query functions.
- The Project should grant the ability to remove any particular vehicle from the parking lot.

DESCRIPTION OF THE PROJECT

The Project has been made in Python 3 but is compatible with python 2 as well. It follows a simple TDD approach.

Test Driven Development (TDD)

Test Driven Development is an approach in which we build a test first, then fail the test and finally refactor our code to pass the test.

Test Driven Development (TDD) Approach

As the name suggests, we should first add the test before adding the functionality in our code. Now our target is to make the test pass by adding new code to our program. So, we refactor our code to pass the written test. This uses the following process –

- Write a failing unit test
- Make the unit test pass
- Repeat

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 -

1. **create_parking_lot n** - Given n number of slots, create a parking lot
2. **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".
3. **status** - Prints the slot number, registration number and color of the parked vehicles.
4. **leave x** - Removes vehicle from slot number x
5. 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 the repository

I have followed TDD 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

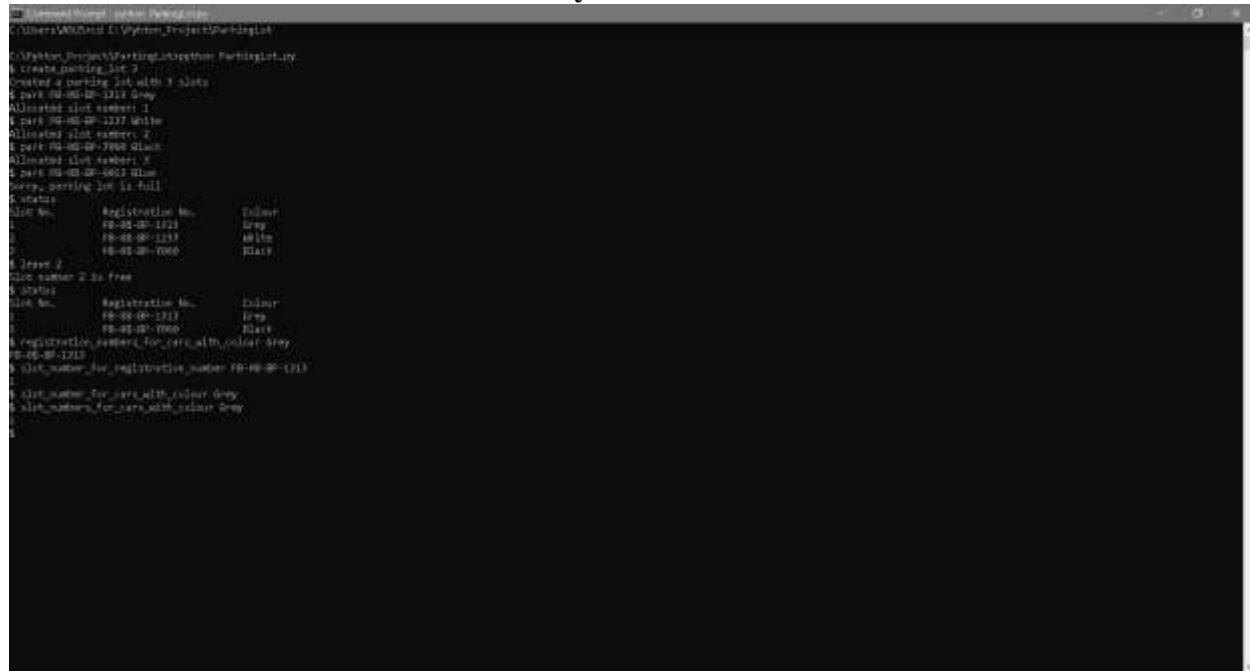
Overall Description of the Project's Capabilities:

- The Project stores the details of the Car like its Registration Number and Color of the car.
- The Project displays the status of the parking lot with details such as slot number, registration number and color of the parked vehicle.
- The Project checks the availability of slots and accordingly informs the user of availability.
- The Project can also remove a particular vehicle from the slot number given by the user.
- There are a few query functions to retrieve slot numbers from registration number of car, get registration of cars with particular color etc.

Requirements: You just need Python. The code is compatible with Python2 as well as Python3. Visit the link <https://www.python.org/downloads/> to install Python.

Steps to creating your own Parking Lot:

1. Clone the repository from Github.
2. Click the start button on the bottom left of the window and type '**Command Prompt**' in the search bar.
3. Open the Command Prompt and use the '**cd**' command to open the directory where the code is located at. For ex, '**cd C:\Pyhton_Project\ParkingLot**'.
4. After this you can either type '**python ParkingLot.py**' and type the commands like this: '**create_parking_lot 4**', '**park PB-08-BP-1313 Grey**', '**status**', '**leave 2**', '**registration_numbers_for_cars_with_colour Grey**', '**slot_number_for_registration_number PB-08-BP-1313**', '**slot_numbers_for_cars_with_colour Grey**'



```
C:\Users\WU2020> python ParkingLot.py
C:\Users\WU2020> cd C:\Pyhton_Project\ParkingLot
C:\Pyhton_Project\ParkingLot> python ParkingLot.py
> create_parking_lot 4
Created a parking lot with 4 slots
> park PB-08-BP-1313 Grey
Allocated slot number: 1
> park PB-08-BP-1237 White
Allocated slot number: 2
> park PB-08-BP-7000 Black
Allocated slot number: 3
> park PB-08-BP-9001 Blue
Error, parking lot is full
> status
Slot No.      Registration No.    Colour
1             PB-08-BP-1313      Grey
2             PB-08-BP-1237      White
3             PB-08-BP-7000      Black
> leave 2
Slot number 2 is free
> status
Slot No.      Registration No.    Colour
1             PB-08-BP-1313      Grey
3             PB-08-BP-7000      Black
> registration_numbers_for_cars_with_colour Grey
PB-08-BP-1313
> slot_number_for_registration_number PB-08-BP-1313
1
> slot_number_for_cars_with_colour Grey
1
> slot_numbers_for_cars_with_colour Grey
1
```

5. Or you can run a test case: **python ParkingLot.py -f run_test_case.txt**. You can also modify the test cases according to your needs.

```
Command Prompt
C:\Users\W0000001\OneDrive\Python_Projects\ParkingLot>python ParkingLot.py -f test_cases.txt

Created a parking lot with 6 slots
Allocated slot number: 1
Allocated slot number: 2
Allocated slot number: 3
Allocated slot number: 4
Allocated slot number: 5
Allocated slot number: 6
Slot number 4 is Free
Slot No.   Registration No.   Colour
1          CA-91-00-1234     white
2          CA-91-00-0000     white
3          CA-91-00-0800     Black
4          CA-91-00-2700     Black
5          CA-91-00-0001     Black
Allocated slot number: 6
Sorry, parking lot is full
CA-91-00-1234, CA-91-00-0000, CA-91-0-123
1, 2, 3
Not Found
C:\Users\W0000001\OneDrive\Python_Projects\ParkingLot>
```

6. 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.

```
Command Prompt
C:\Users\W0000001\OneDrive\Python_Projects\ParkingLot>python test_parking_lot.py

*****
Run 6 tests in 0.001s

OK
C:\Users\W0000001\OneDrive\Python_Projects\ParkingLot>
```


ROLE OF EACH MEMBER

Gaurish Sharma: Gaurish was tasked to make the class 'ParkingLot', functions such as 'createParkingLot(self,capacity)', 'getEmptySlot(self)', 'park(self,regno,color)', 'leave(self,slotid)' and also to make the Project Report.



```
class ParkingLot:
    def __init__(self, capacity):
        self.capacity = capacity
        self.regno = []
        self.slotid = []
        self.slotcolor = []

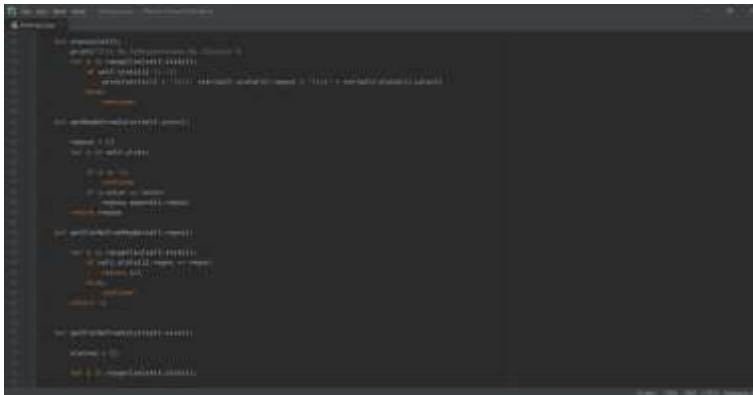
    def createParkingLot(self, capacity):
        self.capacity = capacity
        self.regno = []
        self.slotid = []
        self.slotcolor = []

    def getEmptySlot(self):
        for i in range(self.capacity):
            if self.slotid[i] == -1:
                return i
        return -1

    def park(self, regno, color):
        slotid = self.getEmptySlot()
        if slotid == -1:
            return -1
        self.regno.append(regno)
        self.slotid.append(slotid)
        self.slotcolor.append(color)
        return slotid

    def leave(self, slotid):
        for i in range(self.capacity):
            if self.slotid[i] == slotid:
                self.regno.pop(i)
                self.slotid.pop(i)
                self.slotcolor.pop(i)
                return i
        return -1
```

Shashwat Tiwari: Shashwat was tasked to make functions such as 'status(self)', 'getRegNoFromColor(self,color)', 'getSlotNoFromRegNo(self,regno)', 'getSlotNoFromColor(self,color)', 'show(self,line)', 'main()'



```
def status(self):
    for i in range(self.capacity):
        if self.slotid[i] == -1:
            print("Slot", i, "is free")
        else:
            print("Slot", i, "is occupied by", self.regno[i], "with color", self.slotcolor[i])

    def getRegNoFromColor(self, color):
        for i in range(self.capacity):
            if self.slotcolor[i] == color:
                return i
        return -1

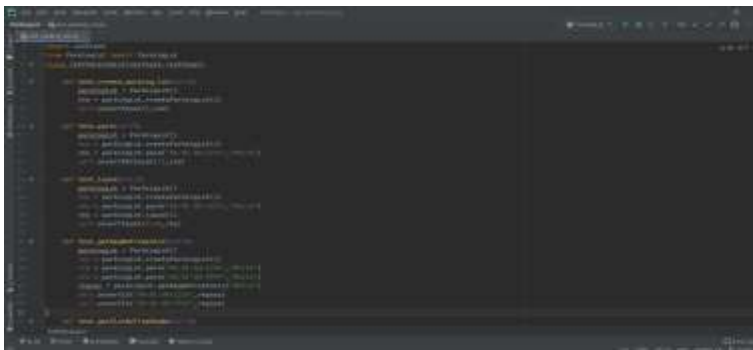
    def getSlotNoFromRegNo(self, regno):
        for i in range(self.capacity):
            if self.regno[i] == regno:
                return i
        return -1

    def getSlotNoFromColor(self, color):
        for i in range(self.capacity):
            if self.slotcolor[i] == color:
                return i
        return -1

    def show(self, line):
        for i in range(self.capacity):
            print("Slot", i, "is", self.slotid[i], "with color", self.slotcolor[i])

    def main():
        parkingLot = ParkingLot(10)
        parkingLot.createParkingLot(10)
        parkingLot.park(100, "Red")
        parkingLot.park(200, "Blue")
        parkingLot.park(300, "Green")
        parkingLot.park(400, "Yellow")
        parkingLot.park(500, "Purple")
        parkingLot.park(600, "Orange")
        parkingLot.park(700, "Pink")
        parkingLot.park(800, "Brown")
        parkingLot.park(900, "Grey")
        parkingLot.park(1000, "White")
        parkingLot.status()
        parkingLot.getRegNoFromColor("Red")
        parkingLot.getSlotNoFromRegNo(100)
        parkingLot.getSlotNoFromColor("Blue")
        parkingLot.show(10)
        parkingLot.main()
```

Rizul Sharma: Rizul was tasked to make the test cases and programs such as 'Vehicle.py', 'test_parking_lot.py'



```
class Vehicle:
    def __init__(self, regno, color):
        self.regno = regno
        self.color = color

    def park(self, parkingLot):
        slotid = parkingLot.park(self.regno, self.color)
        if slotid == -1:
            print("No slot available for parking")
        else:
            print("Vehicle parked at slot", slotid)

    def leave(self, parkingLot, slotid):
        parkingLot.leave(slotid)
        print("Vehicle left the parking lot")

def test_parking_lot():
    parkingLot = ParkingLot(10)
    parkingLot.createParkingLot(10)
    vehicle = Vehicle(100, "Red")
    vehicle.park(parkingLot)
    vehicle.leave(parkingLot, 0)
    vehicle = Vehicle(200, "Blue")
    vehicle.park(parkingLot)
    vehicle.leave(parkingLot, 1)
    vehicle = Vehicle(300, "Green")
    vehicle.park(parkingLot)
    vehicle.leave(parkingLot, 2)
    vehicle = Vehicle(400, "Yellow")
    vehicle.park(parkingLot)
    vehicle.leave(parkingLot, 3)
    vehicle = Vehicle(500, "Purple")
    vehicle.park(parkingLot)
    vehicle.leave(parkingLot, 4)
    vehicle = Vehicle(600, "Orange")
    vehicle.park(parkingLot)
    vehicle.leave(parkingLot, 5)
    vehicle = Vehicle(700, "Pink")
    vehicle.park(parkingLot)
    vehicle.leave(parkingLot, 6)
    vehicle = Vehicle(800, "Brown")
    vehicle.park(parkingLot)
    vehicle.leave(parkingLot, 7)
    vehicle = Vehicle(900, "Grey")
    vehicle.park(parkingLot)
    vehicle.leave(parkingLot, 8)
    vehicle = Vehicle(1000, "White")
    vehicle.park(parkingLot)
    vehicle.leave(parkingLot, 9)
    parkingLot.status()
    parkingLot.getRegNoFromColor("Red")
    parkingLot.getSlotNoFromRegNo(100)
    parkingLot.getSlotNoFromColor("Blue")
    parkingLot.show(10)
    parkingLot.main()
```

Implementation

After writing the commands on the shell:

```
C:\Users\W000001> cd C:\Python\Project\ParkingLot
C:\Python\Project\ParkingLot> python ParkingLot.py
Created a parking lot with 3 slots
Park the car with color grey
Allocated slot number: 1
Park the car with color blue
Allocated slot number: 2
Park the car with color black
Allocated slot number: 3
Sorry, parking lot is full
Status
Slot No.   Registration No.   Colour
1          FB-01-00-1234    Grey
2          FB-01-00-1235    White
3          FB-01-00-1000    Black
Driver 2
Slot number 2 is free
Status
Slot No.   Registration No.   Colour
1          FB-01-00-1234    Grey
2          FB-01-00-1000    Black
Registration numbers for cars with colour grey
FB-01-00-1234
Slot number for registration number FB-01-00-1234
1
Slot number for cars with colour grey
1
Slot numbers for cars with colour grey
```

Running a test case:

```
C:\Users\W000001> cd C:\Python\Project\ParkingLot
C:\Python\Project\ParkingLot> python ParkingLot.py -f test_case.txt
Created a parking lot with 6 slots
Allocated slot number: 1
Allocated slot number: 2
Allocated slot number: 3
Allocated slot number: 4
Allocated slot number: 5
Allocated slot number: 6
Slot number 4 is free
Status
Slot No.   Registration No.   Colour
1          KA-01-00-1234    White
2          KA-01-00-0000    White
3          KA-01-00-0001    Black
4          KA-01-00-2302    Black
5          KA-01-00-2303    Black
6          KA-01-00-2304    Black
Allocated slot number: 4
Sorry, parking lot is full
KA-01-00-1234, KA-01-00-0000, KA-01-00-2303
1, 2, 3
Not Found
C:\Python\Project\ParkingLot>
```

Running all the 6 Functionalities:

```
C:\Users\W000001> cd C:\Python\Project\ParkingLot
C:\Python\Project\ParkingLot> python test_parking_lot.py
.....
Run 6 tests in 0.001s
OK
C:\Python\Project\ParkingLot>
```

TECHNOLOGIES AND FRAMEWORK TO BE USED

Python 3: Python is an interpreted, high-level and general-purpose programming language. Created by Guido van Rossum and first released in 1991, Python's design philosophy emphasizes code readability with its notable use of significant whitespace. Its language constructs and object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects.

Python is dynamically typed and garbage-collected. It supports multiple programming paradigms, including structured (particularly, procedural), object-oriented, and functional programming. Python is often described as a "batteries included" language due to its comprehensive standard library.

Python was created in the late 1980s as a successor to the ABC language. Python 2.0, released in 2000, introduced features like list comprehensions and a garbage collection system with reference counting.

PyCharm Community Edition 2020: PyCharm is an integrated development environment (IDE) used in computer programming, specifically for the Python language. It is developed by the Czech company JetBrains. It provides code analysis, a graphical debugger, an integrated unit tester, integration with version control systems (VCSes), and supports web development with Django as well as data science with Anaconda.

PyCharm is cross-platform, with Windows, macOS and Linux versions. The Community Edition is released under the Apache License, and there is also Professional Edition with extra features – released under a proprietary license. PyCharm provides an API so that developers can write their own plugins to extend PyCharm features. Several plugins from other JetBrains IDE also work with PyCharm. There are more than 1000 plugins which are compatible with PyCharm.

PyCharm offers some of the best features to its users and developers in the following aspects –

- Code completion and inspection
- Advanced debugging
- Support for web programming and frameworks such as Django and Flask

SWOT ANALYSIS

Strengths: This project is both Python 2 as well as Python 3 project, even though Python 3 is not fully backwards compatible with Python 2. This project is very easy to understand and is beginner friendly. To make this project one doesn't require the knowledge of other languages such as MySQL etc. In this Project the user can make his/her own parking slot with his/her choice of no of slots. In this project one can view the status of all the cars in the parking lot and as well retrieve certain information with certain query functions. The program automatically gives a slot to a new entry and informs the user when there is no availability of slots.

Weakness: This project doesn't have many weaknesses. It just doesn't use database or use Django to make a GUI. Our group did try to make a project previously based on Python, pyqt5 and MySQL but was unable to get it working...Hence why we decided to make this project completely in Python 3, which can also be considered a good sign since it only needs the knowledge of Python 3.

Opportunities: This project can be used as building blocks for bigger full-fledged parking management applications. Since it is beginner friendly anyone can understand it with just the knowledge of Python and can work on it according to his/her own needs. This project can be used by companies for their own applications. Beginners can understand the basics of Object-Oriented Programming making the use of classes and functions. Users can make their own test cases and can follow the TDD approach or the Test Driven Development approach.

Threats: The threats to this project are only from full-fledged applications which are ready for use by the general public but otherwise this project is fantastic for coders, since it gives them an insight to how the basics of a full-fledged parking management application works.

References

- <https://en.wikipedia.org/wiki/PyCharm#:~:text=PyCharm%20is%20an%20integrated%20development,by%20the%20Czech%20company%20JetBrains.&text=PyCharm%20is%20cross%2Dplatform%2C%20with,Windows%2C%20macOS%20and%20Linux%20versions>.
- https://www.tutorialspoint.com/pycharm/pycharm_introduction.htm
- [https://en.wikipedia.org/wiki/Python_\(programming_language\)](https://en.wikipedia.org/wiki/Python_(programming_language))
- <https://www.youtube.com/watch?v=NSbOtYzIQI0>
- <https://www.youtube.com/watch?v=rfscVS0vtbw&t=1634s>