SRI KRISHNA INSTITUTIONS

BOOTATHON PROJECT

VEHICLE PARKING MANAGEMENT SYSTEM

TEAM	: **S10**
TEAM NAME	: KEKRAN MEKRAN
TEAM LEAD	: DHANISH AHMED
TEST LEAD	: VENMUHILAN
BUSINESS ANALYST	: PRAMOTH KUMAR
DEPUTY BUSINESS ANALYST	: HARI PRIYA
DEVELOPERS	: AJAY JULLIAN SHARON MANICKA BHARATH ABISHEK SIVA KUMAR

VEHICLE MANGEMENT SYSTEM

BUSINESS REQUIREMENTS

The following are the business requirements of this project

S.I	Requirements	status
1	Login	
2	Vehicle details	
3	Slot allotment	
4	Payment	
5	Display	

1.LOGIN

This section contains the login details.

- The username and password should not exceed 20 characters long.
- String compare checks the username and password.
- If both are matching it displays “Login Successfully” otherwise it displays “Username or Password is incorrect”.

2.VEHICLE DETAILS

This section deals with the identification of the vehicle.

- By macros, BIKE and CAR are assigned as 0 and 1.
- According to the type of vehicle the security gives the input.
- The vehicle number is stored in string of 11 characters.
- The entry time is based on the system time which is in 24 hr format.
- Based on the above details token is generated.

3.SLOT ALLOTMENT

This section allocates the space for vehicles.

- Based on the token ,the type of vehicle is identified.
- For the Two wheeler parking 5 floors and 50 columns are allocated.
- And for the four wheeler parking 2 rows and 50 columns are allocated.
- If the type is found ,specified space is allocated for its token number.
- A message is displayed when there is no space.

4.PAYMENT

This module deals with the amount to be paid.

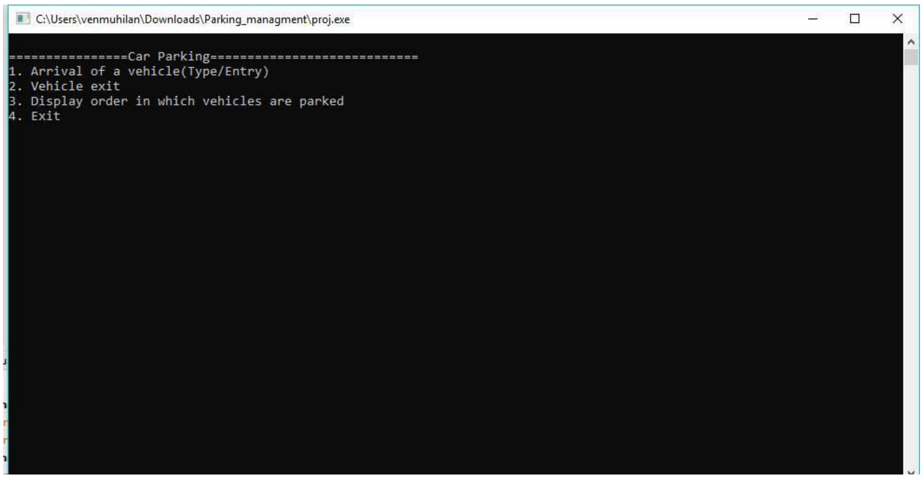
- All the fields are mandatory.
- The exit time is noted.
- According to entry and exit time of the respective vehicle, the parking fee is generated.

5.DISPLAY

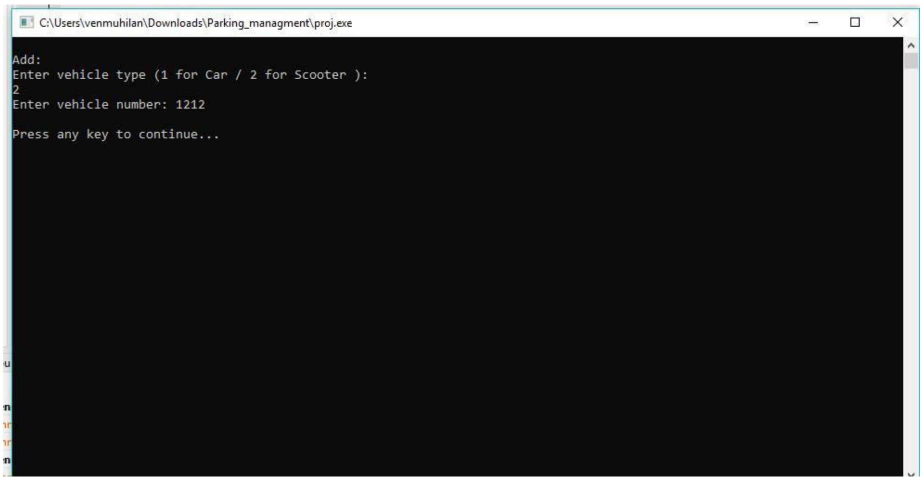
This module displays the status of the parking area.

- By Switch case ,various display tasks have been done.
- It displays the entry and exit time log of vehicle.
- It also displays the availabilityof parking space
- All are user demand cases.

SCREENSHOT:

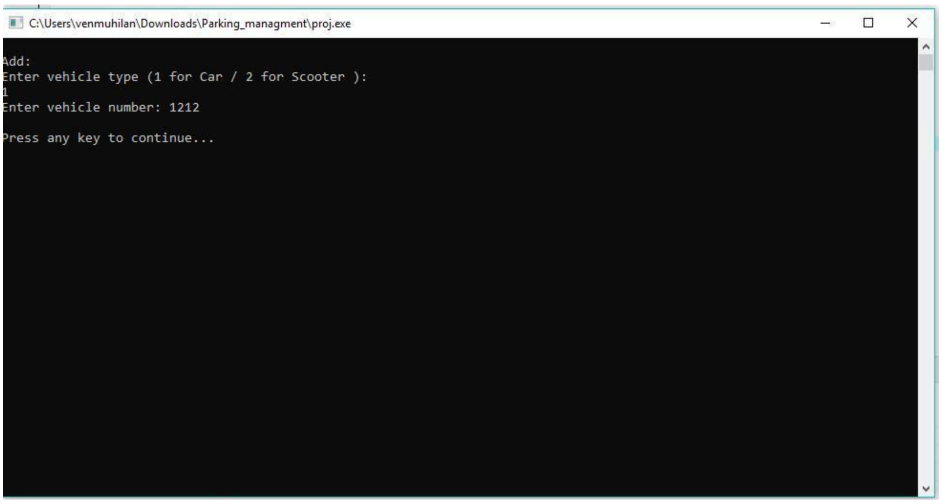


```
===============Car Parking===========================
1. Arrival of a vehicle(Type/Entry)
2. Vehicle exit
3. Display order in which vehicles are parked
4. Exit
```



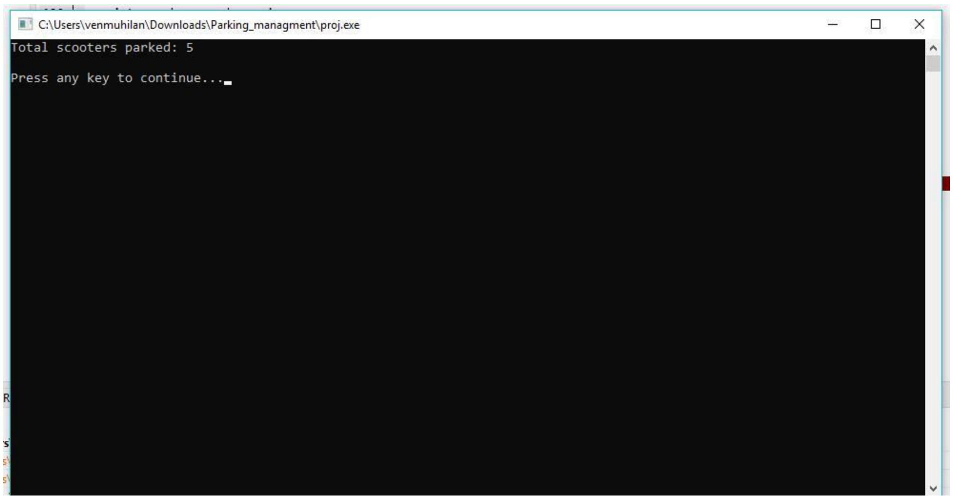
```
Add:
Enter vehicle type (1 for Car / 2 for Scooter ):
2
Enter vehicle number: 1212

Press any key to continue...
```



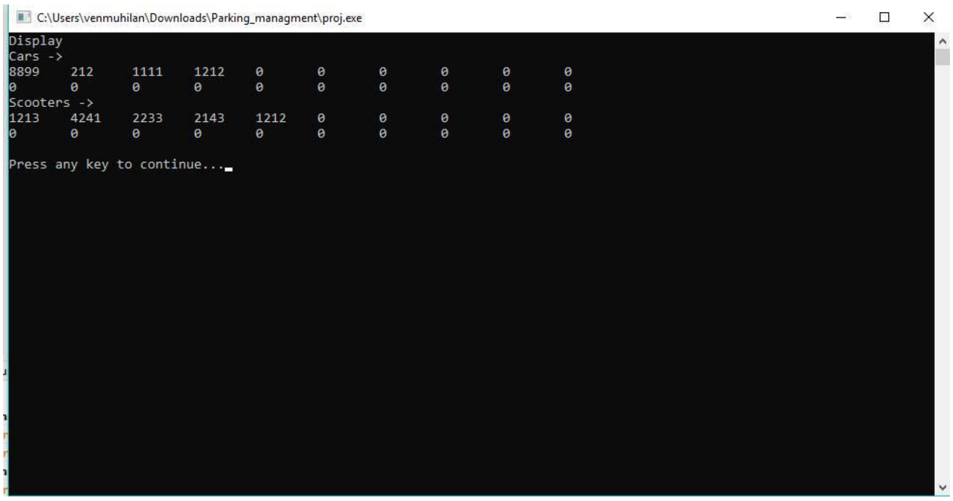
```
Add:
Enter vehicle type (1 for Car / 2 for Scooter ):
1
Enter vehicle number: 1212

Press any key to continue...
```



```
Total scooters parked: 5

Press any key to continue...
```



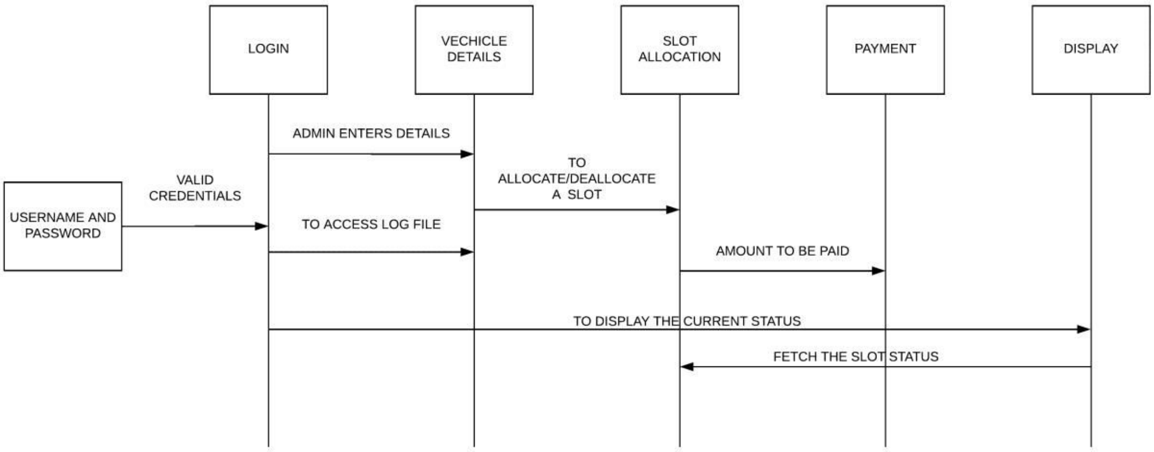
```
Display
Cars ->
8899    212     1111    1212    0       0       0       0       0       0
0       0       0       0       0       0       0       0       0       0
Scooters ->
1213    4241    2233    2143    1212    0       0       0       0       0
0       0       0       0       0       0       0       0       0       0

Press any key to continue...
```

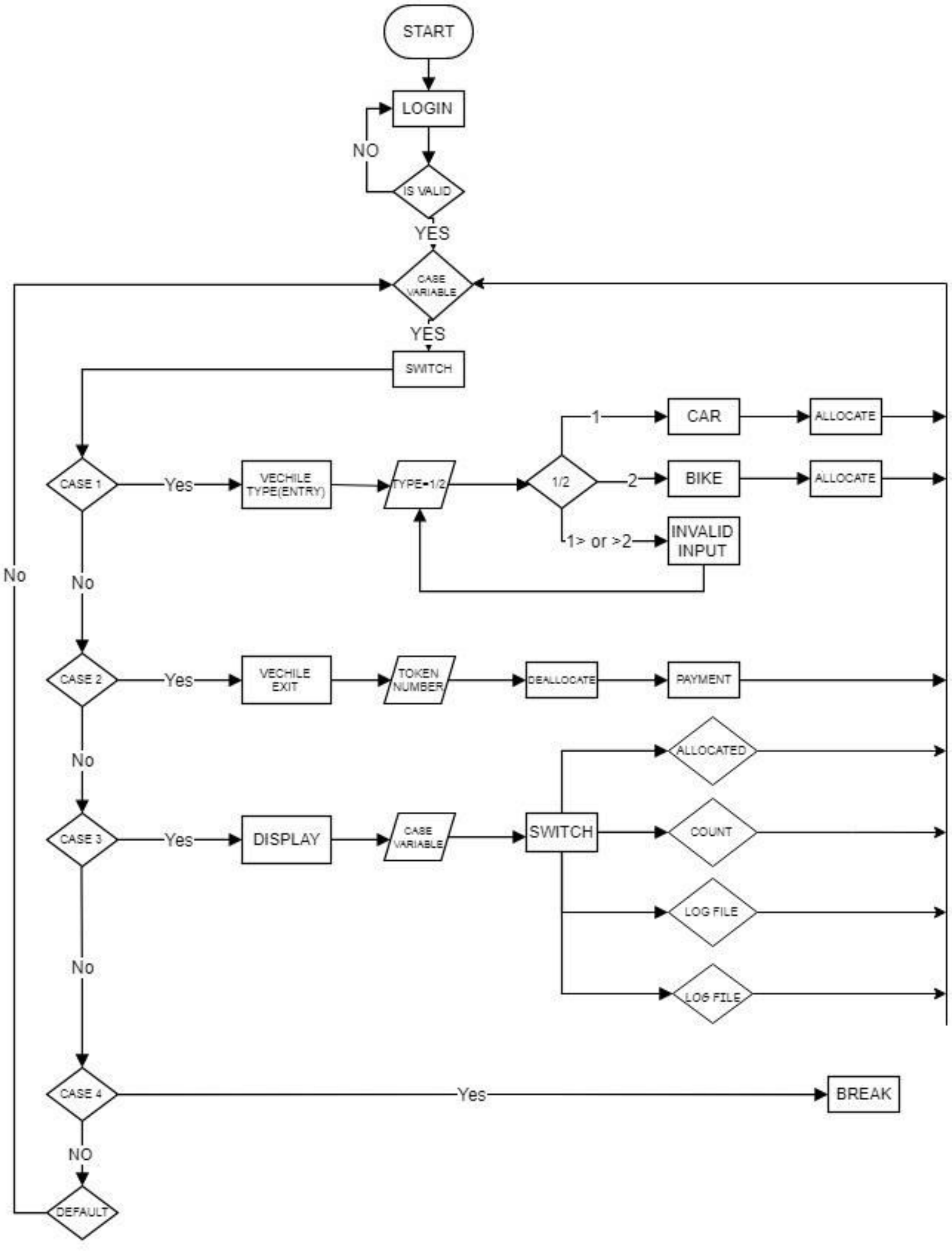
VEHICLE PARKING MANAGEMENT SYSTEM

TEAM : S10

CONTROL FLOW DIAGRAM



FLOW CHART


START
LOGIN
NO
IS VALID
YES
CASE VARIABLE
YES
SWITCH
CASE 1
Yes
VECHILE TYPE(ENTRY)
TYPE=1/2
1/2
1
2
1> or >2
CAR
BIKE
INVALID INPUT
ALLOCATE
ALLOCATE
No
No
CASE 2
Yes
VECHILE EXIT
TOKEN NUMBER
DEALLOCATE
PAYMENT
No
CASE 3
Yes
DISPLAY
CASE VARIABLE
SWITCH
ALLOCATED
COUNT
LOG FILE
LOG FILE
No
CASE 4
Yes
BREAK
NO
DEFAULT

MODULE 1 (LOGIN)

ALGORITHM

Step 1: Start the program
Step 2: Get username and password from security
Step 3: Compare the username and password entered by the security with the existing password
Step 4: If the username and the password is correct, the security can login
Step 5: Otherwise, the security cannot login.
Step 6: Stop the program

PSEUDO CODE

```
BEGIN
Print “username”
Input uname
Print “password”
Input pass
If( uname==”admin1 or admin2”) &amp;&amp; if (pass==”pass”)
True: Print “login successfull”
False: Print “Invalid username or password”
END
```

MODULE 2(VEHICLE DETAILS)

ALGORITHM

step 1: start the program
step 2: get the registration number and vehicle type
step 3: record its entry time and write all details into log file
step 4: allocate token no and allocate parking
step 5: stop the program

PSEUDO CODE

```
BEGIN
case input
get regno and vehicle type
time_t mytime = time(NULL)
initialize tokenno
get start time = ctime(&mytime)
fputs(text,fptr)
END
```

MODULE 3 (SLOT ALLOCATION)

ALGORITHM:

Step 1: Start
Step 2: Get the token number and the type of the vehicle
Step 3: If the type is two wheeler then store it in array “two”
Step 4: Free spaces are represented as “0”
Step 5: Traverse the array and place the vehicle in the free space
Step 6: Return the slot number
Step 7: Stop

PSEUDOCODE

```
BEGIN
Input tokennumber
Input type
If type==1
Loop i=0-2,j=0-50
a[i][j]=0
Loop i=0-2,j=0-50
If(two[i][j]==0) do two[i][j]=tokennumber
Display slotnumber
If type==2
Loop i=0-2,j=0-50
a[i][j]=0
Loop i=0-2,j=0-50
If(four[i][j]==0) do four[i][j]=tokennumber
Display slotnumber
End
```

MODULE 4 (PAYMENT)

ALGORITHM

step 1: start the program
step 2: get the registration number
step 3: check its entry time and note exit time
step 4: calculate total durations and calculate charges
step 5: display total charges
step 6: write all details into log file
step 7: stop the program

PSEUDO CODE

```
BEGIN
time_t mytime = time(NULL)
get tokenno
get exit time = ctime(&mytime)
totalduration=endtime-starttime
parkingcharges=totalduration*fare
fputs(text,fp)
```

MODULE 5 (DISPLAY)

ALGORITHM

step 1: start the program
step 2: get the switch case variable
step 3: based on the case variable execute the functions.
step 4: 1 deals with allocation representation,2 deals with total vehicle count in parking,3 arrival log file,4 with departure log file
step 5: display the required case.
step 6: stop the program.

PSEUDO CODE

```
BEGIN
Input casevariable
switch casevariable
case 1: Display allocation representation
case 2: Display total vehicle count in parking
case 3: Display arrival log
case 4: Display departure log
END
```