

ParkingPlus

1 INTRODUCTION

ParkingPlus is the application to manage multi floor parking system. Now a days search for a parking in metro cities is very time consuming task. This application tells admin about the available parking in building which could be further displayed on board. It stores the information about the total and available parking on each floor and entry/exit information of each car.

This application implements paid parking system. The charges are as below-

- \$3 per hour till first 6 hours
- If parked for more than 6 hour, then \$20 for per day parking.
- If parked for more than one day, then \$20 per day, even if it exceeds a minute.

ParkingPlus application provide option to generate reports in csv excel format to see history of car parked in parking in some date range.

2 REQUIREMENTS

This application is using below python version and modules-

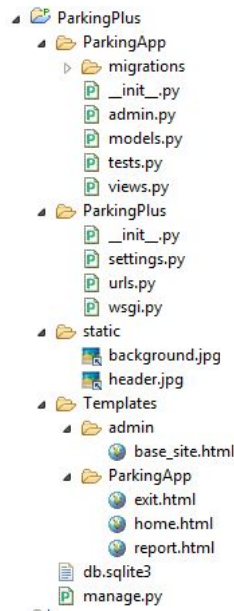
- Python version 2.7 and above
- Module Django 1.7
- Module datetime
- Database Sqlite3

To **run** the application follow below steps –

1. Go to command prompt
2. Direct it to the path where you saved the ParkingPlus folder as D:/PythonCourse/ParkingPlus
3. Run command 'python manage.py runserver'
4. Open the browser
5. Type localhost:8000 or 127.0.0.1:8000, It will show you home page

3 DESCRIPTION OF PYTHON PROGRAM

ParkingPlus is developed using Django module. Django provide its own MVC architecture to develop web application. Below screen shot shows the directory structure of development.



Directory structure of application

1. ParkingPlus directory is main directory has all the folders as shown in above picture. It has db.Sqlite3 to store database and manage.py to run the application
2. Sub ParkingPlus directory has __init__.py, settings.py, url.py and wsgi.py.
 - a. settings.py has all the setting to run the application and directory paths for template and static files.
 - b. url.py has the get/post urls for different webpages. By default the application start on home page.

```

1 from django.conf.urls import patterns, include, url
2 from django.contrib import admin
3
4 urlpatterns = patterns('',
5     # Examples:
6     # url(r'^$', 'ParkingPlus.views.home', name='home'),
7     # url(r'^blog/', include('blog.urls')),
8
9     url(r'^admin/', include(admin.site.urls)),
10    url(r'^$', 'ParkingApp.views.home', name='home'),
11    url(r'^carentry/', 'ParkingApp.views.carentry', name='carentry'),
12    url(r'^exit/', 'ParkingApp.views.carsearch', name='exit'),
13    url(r'^carexit/', 'ParkingApp.views.carexit', name='carexit'),
14    url(r'^report/', 'ParkingApp.views.report', name='report'),
15    url(r'^getreport/', 'ParkingApp.views.getreport', name='getreport'),
16 )
  
```

url.py

3. ParkingApp directory has __init__.py, models.py, views.py, admin.py python code files.
 - a. This application is using two database tables, one to store the floor parking information and other to store car entry/exit and parking information. Models.py has two models for each table. Below screen shot shows the definition of each model.

```

1 from django.db import models
2
3 # Create your models here.
4 #model for floor plan
5 class FloorPlan(models.Model):
6     floorNo = models.SmallIntegerField(unique = True)
7     totalParking = models.IntegerField()
8     availableParking = models.IntegerField(null = True)
9
10 def __str__(self):
11     return str(self.floorNo)
12
13 #model for car entry exit information
14 class CarEntryExit(models.Model):
15     carNo = models.CharField(max_length= 100)
16     floorNo = models.SmallIntegerField()
17     timeEntered = models.DateTimeField()
18     timeExit = models.DateTimeField(null = True)
19     feePaid = models.FloatField(null = True)
20
21 def __unicode__(self):
22     return self.carNo

```

models.py

- b. admin.py has the python code to implement default admin functionality to manage the application. I added the list of model variable to show them in admin page grid and list of filter to apply the filters on grid to filter the data.
- c. View.py is the main page for the application which implements different classes and methods to get and post request from the user. It has below classes to create the forms
 - i. CarEntryForm() – class to create the form section for home page. It has 3 input fields-
 1. carNo. – required CharField with max length 10
 2. floorNo- required IntegerField, accept only integer value
 3. timeEntered – Read only DateTimeField field, initialized with current date time.
 4. It has custom validation for floorNo, which except the existing floor numbers and that floor must have available parking.
 - ii. CarSearchForm() – class to create car search form on exit page. It has two input fields-
 1. carNo. – required CharField
 2. timeExit – Read only DateTimeField field, initialized with current date time.
 3. It has custom validation for carNo, the input carNo must exist in the database.
 - iii. ReportForm() – class to create form section on report page. It has two input fields-
 1. fromDate – required DateField, accept date only in yyyy-MM-dd format.
 2. toDate – required DateField, accept date only in yyyy-MM-dd format.

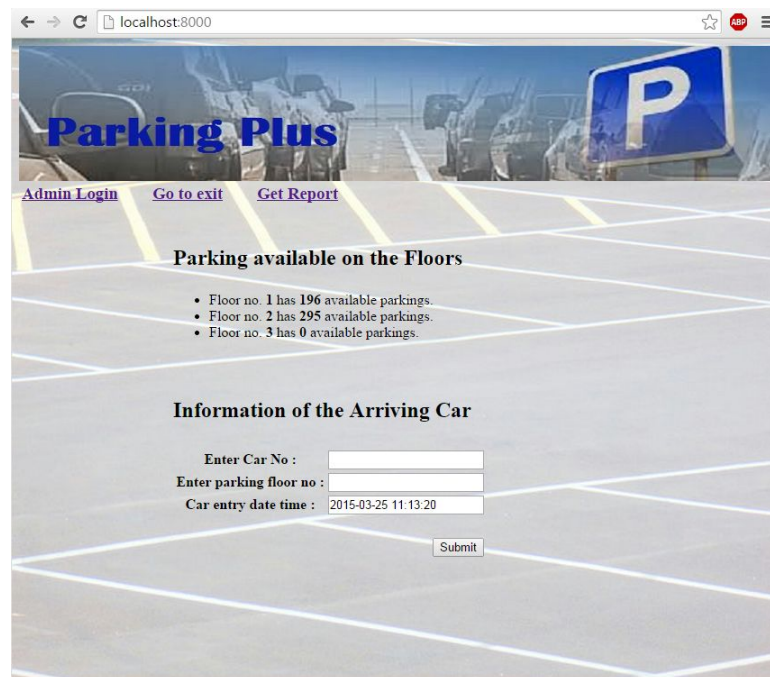
This view.py has the function definitions to render get/post requests. Below is the small description of each methods defined in views.

- iv. home() – renders the home page. It also passes the list of available parking on each floor to display data on home page.

- v. `Carentry()` – invokes on submit POST on home page. It checks the validity of input data then save the car entry information in database and deduct the available parking of input floor.
 - vi. `carsearch()` – renders the exit page and initialize the current timeExit on each request.
 - vii. `carexit()` – invokes on submit request on exit page. It checks the validity of input data then check if car has already been exited. If not the save the exit time to DB and then calculate the parking fee and displays it on exit page. It add the free parking on that particular floor from where the current car exited.
 - viii. `calculateFee()` – takes the time difference of car entry and exit time in seconds as input, then apply the logic to calculate the fee as shown in introduction part.
 - ix. `report()` – renders the report page.
 - x. `getreport()` – invokes on getreport request from report page and generates the report in csv excel format. It includes all cars which were parked between from and to date means those cars which were entered after fromDate and exited before toDate.
4. Static directory is to store static files. It has two images for page header and background.
 5. Template directory has all the html template pages. Admin/base_site.html is to customize the admin view pages header. Html pages in ParkingApp folder has the template for home, exit and report pages.

4 SCREEN SHOT OF PROGRAM OUTPUT

1- Home Page screens – localhost:8000



SS1- Main page showing links to other pages

Information of the Arriving Car

Enter Car No : • This field is required.

Enter parking floor no : • This field is required.

Car entry date time : 2015-03-25 11:13:20

Required field validation

Information of the Arriving Car

Enter Car No : • This field is required.

Enter parking floor no :

Car entry date time : 2015-03-25

! Please enter a number.

Allows only integer floor number

Information of the Arriving Car

• 6 is not a valid floor no.

Enter Car No :

Enter parking floor no :

Car entry date time : 2015-03-25 11:17:39

Shows error is floor no does not exist

Parking available on the Floors

- Floor no. 1 has 196 available parkings.
- Floor no. 2 has 295 available parkings.
- Floor no. 3 has 0 available parkings.

Information of the Arriving Car

- Parking is full for floor 3

Enter Car No :

Enter parking floor no :

Car entry date time :

Shows error is no available parking on input floor

2- Exit Page screen shots – Click on Go to Exit link on main page

localhost:8000/exit/

Parking Plus

[Home](#)

Car standing on the Exit

Enter Car No :

Car exit date time :

Exit page

Car standing on the Exit

- This field is required.

Enter Car No :

Car exit date time :

Required field validation

Car standing on the Exit

- sdfd is not exist in the parking

Enter Car No :

Car exit date time :

Shows error is car does not exist

Car standing on the Exit

Enter Car No :

Car exit date time :

This car has already exited.

Shows message if searched car has already exited



Car standing on the Exit

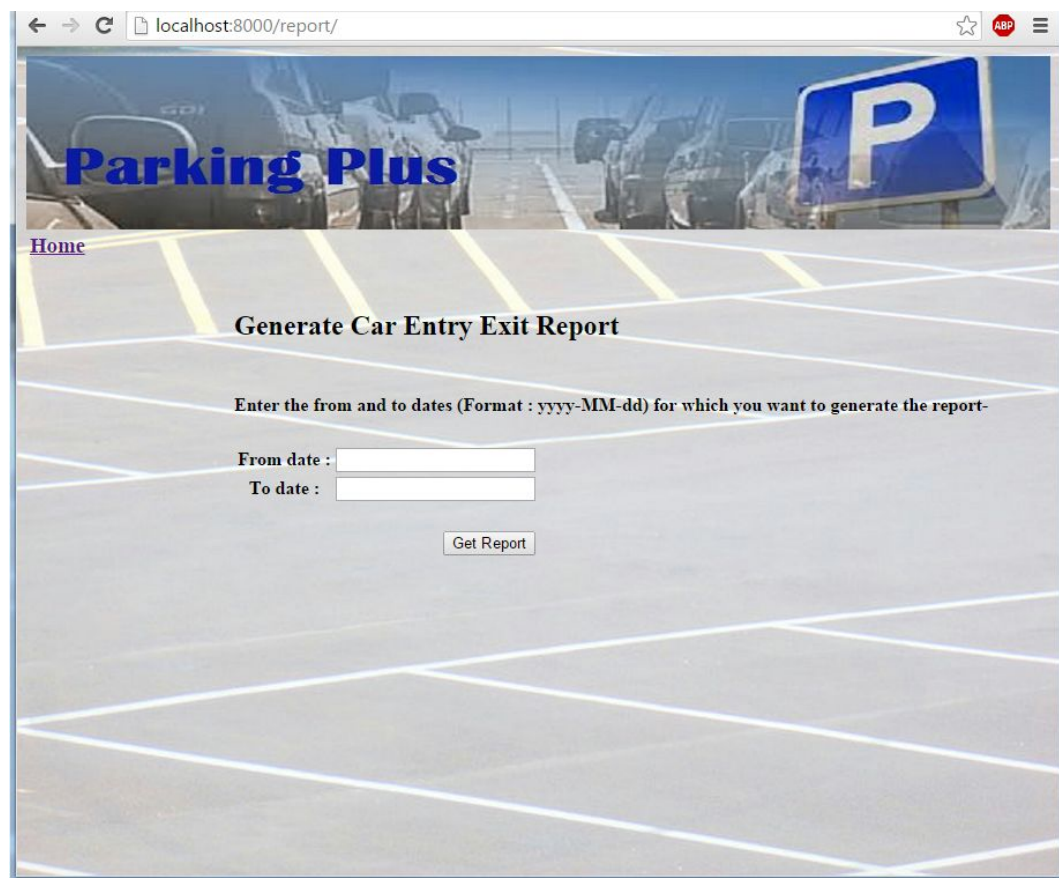
Enter Car No :

Car exit date time :

Car 'car2' was entered on March 25, 2015, 11:17 a.m.. The total fee is 3. Please Pay the fee.

Shows the entry time of searched car and the parking fee

3- **Report Page** – Click on **Get Report** link on home page.



localhost:8000/report/

Parking Plus

[Home](#)

Generate Car Entry Exit Report

Enter the from and to dates (Format : yyyy-MM-dd) for which you want to generate the report-

From date :

To date :

Report Page

Generate Car Entry Exit Report

Enter the from and to dates (Format : yyyy-MM-dd) for which you want to generate the report-

From date :

- This field is required.

To date :

- This field is required.

Required field validation

Generate Car Entry Exit Report

Enter the from and to dates (Format : yyyy-MM-dd) for which you want to generate the report-

From date :

- Enter a valid date.

To date :

- Enter a valid date.

Shows message is the date is not in valid format

Parking Plus

[Home](#)

Generate Car Entry Exit Report

Enter the from and to dates (Format : yyyy-MM-dd) for which you want to generate the report-

From date :

To date :

CarEntryExitReport.csv

[Show all downloads...](#)

Download the report in csv formatted file

CarEntryExitReport - Excel

FILE HOME INSERT PAGE LAYOUT FORMULAS DATA REVIEW VIEW PC

A1 : Car No

	A	B	C	D	E	F	G	H	I
1	Car No	Floor No	Entry Date	Exit Date	Fee Paid				
2	wewe	4	2015-03-2	2015-03-2	0				
3	ereer	2	2015-03-2	2015-03-2	0				
4	qwqw	2	2015-03-2	2015-03-2	3				
5	wewewe	1	2015-03-2	2015-03-2	20				
6	aaaaa	1	2015-03-2	2015-03-2	3				
7	sffdgd	2	2015-03-2	2015-03-2	3				
8	ababab	3	2015-03-2	2015-03-2	0				
9	bzbz	3	2015-03-2	2015-03-2	3				
10	new car	2	2015-03-2	2015-03-2	3				
11	car2	2	2015-03-2	2015-03-2	3				
12									

Downloaded Excel file

4- Admin Page – click on **Admin Login** link on Home page-

localhost:8000/admin/login/?next=/admin/

ParkingPlus Administration

Username:

Password:

Log in

Admin Login page (username – admin, password-admin)

The screenshot shows the 'ParkingPlus Administration' dashboard. The top navigation bar includes a welcome message for 'admin' and links to 'Change password' and 'Log out'. The main content area is titled 'Site administration' and contains two sections: 'Authentication and Authorization' and 'Parkingapp'. The 'Authentication and Authorization' section has links for 'Groups' and 'Users', each with 'Add' and 'Change' options. The 'Parkingapp' section has links for 'Car entry exits' and 'Floor plans', also with 'Add' and 'Change' options. On the right, a 'Recent Actions' box titled 'My Actions' lists several car entry exits with IDs like WTE365F, 555, 6666, and 7777.

Shows DB models and user groups and actions

This screenshot shows the 'Select user to change' page. It features a search bar with a 'Search' button and a table of users. The table has columns for 'Username', 'Email address', 'First name', 'Last name', and 'Staff status'. One user, 'admin', is listed with the email 'admin@parkingplus.com' and a green checkmark in the 'Staff status' column. A 'Filter' sidebar on the right allows filtering by 'staff status' (All, Yes, No) and 'superuser status' (All, Yes). An 'Add user +' button is located at the top right.

Admin users could be added here

This screenshot shows the 'Select car entry exit to change' page. It includes a search bar, a table of car entry exits, and a 'Filter' sidebar. The table has columns for 'CarNo', 'FloorNo', 'TimeEntered', 'TimeExit', and 'FeePaid'. The 'Filter' sidebar on the right allows filtering by 'timeEntered', 'timeExit', and 'floorNo'. A red box highlights the 'Filter' sidebar. An 'Add car entry exit +' button is at the top right.

Car entry/exit data, It can be filtered on filter shown on right tab



ParkingPlus Administration Welcome, **admin**. Change password / Log out

Home > Parkingapp > Floor plans

Select floor plan to change Add floor plan +

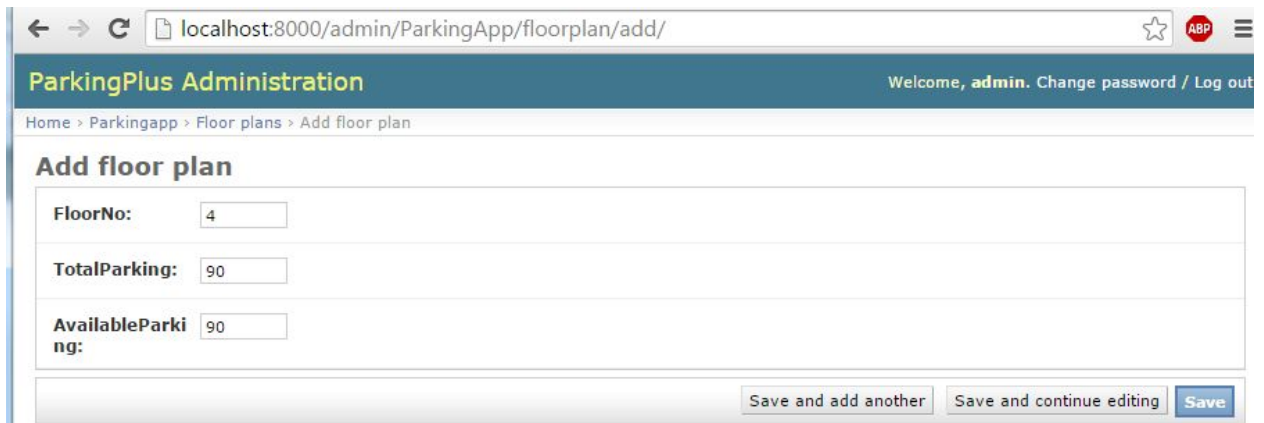
Action: Go 0 of 3 selected

<input type="checkbox"/>	FloorNo	TotalParking	AvailableParking
<input type="checkbox"/>	3	140	0
<input type="checkbox"/>	2	300	295
<input type="checkbox"/>	1	200	196

3 floor plans

Filter
By floorNo
All
1
2
3

Floor data with the filter



ParkingPlus Administration Welcome, **admin**. Change password / Log out

Home > Parkingapp > Floor plans > Add floor plan

Add floor plan

FloorNo:

TotalParking:

AvailableParking:

Admin can add floor here

5 CONCLUSION

This ParkingPlus Application solve the parking maintenance problem to some extent. Admin knows if parking is full or available, it will save you to waste your time unnecessarily in searching parking if it is already full. It calculates the parking fee based on the time the car is parked. Currently the parking fee is fixed in program but this application could be extended to next level with variable fee based on admin input. It generates the report in excel file format to see history of car parked if required.