

Vehicle Rental System

Travel Buddy

A PROJECT REPORT

Submitted by

BL.EN.U4AIE20013 – G Vishwas

BL.EN.U4AIE20042 – N Sai Nithin

BL.EN.U4AIE20050 – P Varshith

19CSE202 – Data Base Management Systems.

B.Tech. in Computer Science and Engineering (Artificial Intelligence)



AMRITA SCHOOL OF ENGINEERING, BANGALORE

AMRITA VISHWA VIDYAPEETHAM

BANGALORE 560 035

June – 2022

Abstract- Any database that is currently installed on the system hard drive or network can be managed by a database management system, which is a piece of computer software. There are various types of database management systems, some of which are designed to supervise and regulate databases that have been established for specific objectives. A software application called the car rental system allows us to store data about the person who reserved the vehicle. These days, a lot of people use this kind of method. The main goal of this system is to rent bikes and cars to customers, who can also lease their own vehicles from us. Instead of viewing it manually, all data will be quickly saved to oracle database and used for future reference. The website gives a full scaled details about our rental system offers we are providing and vehicles available for rent and which are already booked etc...

INTRODUCTION:

A Vehicle rental system is a software application that is used to store, manage, and retrieve data related to vehicles available for renting and already booked. This type of system is used by businesses and organizations that need to keep track of information about vehicles, such as their vehicles booked status and customer details and basically used as a platform for their customers to book vehicles. One of the main reasons for using an rental system is to improve the efficiency of managing and retrieving vehicle information. Data in a central location, it becomes much easier to search for and retrieve the information, rather than having to manually look. This saves a lot of time and effort, especially for organizations that have a large number of vehicles and bookings in their inventory.

The oracle Database 11g express is used for the database in the webpage. It is a database system where we create table by performing the SQL queries and insert data into the tables using SQL commands. And we connect this database to our front end webpage to where the data need to be printed or represented. For connecting oracle to the frontend part we are going to use Django with python as platform.

We have named our system as Travel Buddy.

DATABASE:

The database consists of the 9 tables

1. Customers

CUSTOMER_NAME	CUSTOMER_AAD...	CUSTOMER_EMAIL	CUSTOMER_CONTACT	CUSTOMER_PASSWORD	GENDER
1 Vishwas	626687513450	vishwasgade@gmail.com	7093661550	vishwas	m
2 Vishwas	626687513450	sgfkhammam@gmail.com	7093661550	vvvivix	m
3 Vishwas	626687513450	gadevishwas6@gmail.com	7093661550	vishwas	m
4 Nani	626687513450	BL.EN.U4AIE20013@bl.sudents.amrita.edu	7093661550	vish	m
5 Varshith	626687513462	peddinenivarshith8@gmail.com	9553567550	vishwas	F
6 Nithin	462421108219	nayinisainithin@gmail.com	9121615657	nithin123	M
7 hitesh	123456789123	h@gmail.com	7093661550	1234	M
8 Vishwas	626687513450	test@gmail.com	7093661550	sfsfsf	M
9 Vishwas	626687513450	test3@gmail.com	7093661550	fssf	M
10 Vishwas	626687513450	test5@gmail.com	7093661550	vish	M

The CUSTOMERS Entity consists of attributes :

customer_name, customer_aadhar, customer_email, customer_contact, customer_password, gender

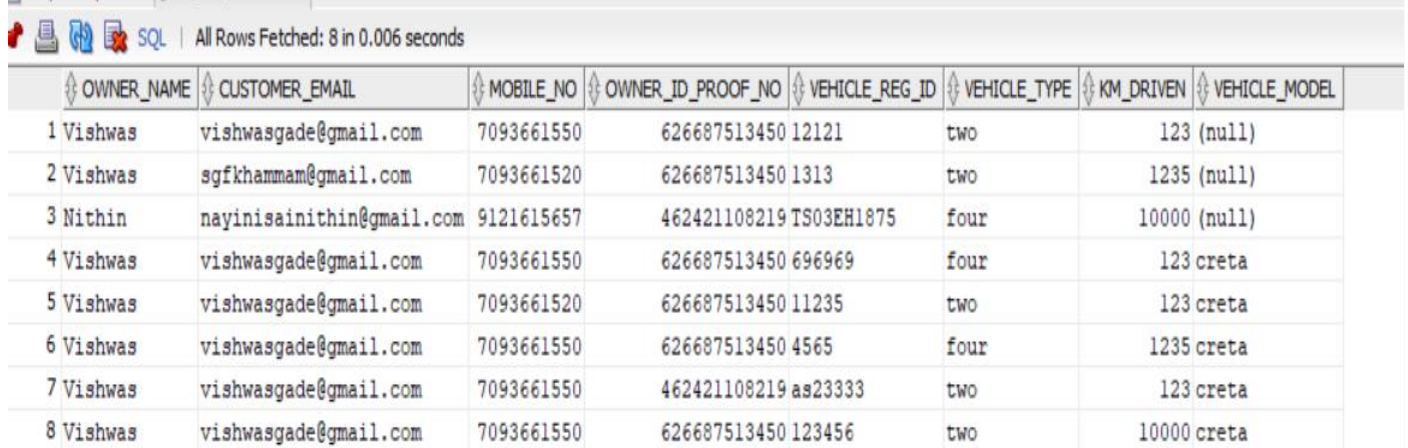
2. Vehicle

VEHICLE_ID	VEHICLE_BRAND	VEHICLE_MODEL	VEHICLE_TYPE	VEHICLE_COLOUR	SEATING_CAPACITY	KMS_DRIVEN	VEHICLE_PRICE_PER_DAY	RENTAL_LOC_ID	LINK
1 Veh001	Bajaj	Chetak	Gearless	Black	2	1200	1000 Rloc01		https://media.zigodn.com/media/content/2020/Jan/chetak_thumbnail.jpg
2 Veh002	Bajaj	Dominor 400	Gear	Red	2	800	3500 Rloc02		https://www.drivespark.com/bikes-photos/models/750x550/bajajdominar400
3 Veh003	Bajaj	Pulsar 150	Gear	Silver	2	750	6900 Rloc03		https://imgd.aeploadn.com/1200x900/n/cw/ec/114277/pulsar-150-right-fro
4 Veh004	Bajaj	Pulsar NS125	Gear	Red	2	1400	3000 Rloc04		https://imgd.aeploadn.com/1280x720/n/cw/ec/57695/pulsar-ns125-right-fro
5 Veh005	Bajaj	Pulsar NS200	Gear	White	2	2000	3200 Rloc05		https://imgd.aeploadn.com/1200x900/n/cw/ec/58025/pulsar-ns-right-front-
6 Veh006	Hondaa	Activa 5G	Gearless	Blue	2	1900	3500 Rloc06		https://www.motorbeam.com/wp-content/uploads/Honda-Activa-125-Premium-
7 Veh007	Hondaa	Activa 6G	Gearless	Silver	2	6000	5000 Rloc07		https://www.tesla24.in/wp-content/uploads/2020/03/honda-select-model-m
8 Veh008	Hondaa	Dio	Gearless	Blue	2	13000	2600 Rloc08		https://imgd.aeploadn.com/1200x900/n/cw/ec/49460/dio-right-front-three-
9 Veh009	Hondaa	Shine	Gear	Blue	2	5500	4000 Rloc09		https://imgd.aeploadn.com/1200x900/n/cw/ec/45481/shine-right-front-thre
10 Veh010	Hondaa	SP 125	Gear	white	2	2000	3000 Rloc10		https://imgd.aeploadn.com/1200x900/n/cw/ec/43482/sp-125-right-front-thr
11 Veh011	KTM	200 Duke	Gear	Black	2	4200	4500 Rloc01		https://www.drivespark.com/img/2017/02/23-1487833520-ktm-duke-200-1.jp
12 Veh012	KTM	390 Duke	Gear	white	2	6000	4299 Rloc02		https://imgd.aeploadn.com/476x268/n/cw/ec/129747/390-duke-right-front-t
13 Veh013	KTM	RC 125	Gear	Blue	2	13000	3699 Rloc03		https://cdn1.autocarindia.com/Utils/ImageResizer.ashx?n=https://cms.ha
14 Veh014	KTM	RC 200	Gear	Blue	2	6000	3200 Rloc04		https://cdn.autoportal.com/bp-v3/img/models/3b/7/ktm-standard-14647859
15 Veh015	KTM	RC 390	Gear	Orange	2	5500	4999 Rloc05		https://www.checkraka.com/uploaded/gallery/2d/2dffe436d473f098dc0ffc92
16 Veh016	Royal Enfield	Classic 350	Gear	Black	2	6660	3999 Rloc06		https://imgd.aeploadn.com/1200x900/n/cw/ec/101487/classic-350-right-fro

The VEHICLE Entity consists of attributes:

vehicle_id, vehicle_brand, vehicle_model, vehicle_type, vehicle_color, seating_capacity, kms_driven, vehicle_price_per_day, rental_loc_id, link

3. Lease_your_vehicle



OWNER_NAME	CUSTOMER_EMAIL	MOBILE_NO	OWNER_ID_PROOF_NO	VEHICLE_REG_ID	VEHICLE_TYPE	KM_DRIVEN	VEHICLE_MODEL
1 Vishwas	vishwasgade@gmail.com	7093661550	626687513450	12121	two	123 (null)	
2 Vishwas	sgfkhammam@gmail.com	7093661520	626687513450	1313	two	1235 (null)	
3 Nithin	nayinisainithin@gmail.com	9121615657	462421108219	TS03EH1875	four	10000 (null)	
4 Vishwas	vishwasgade@gmail.com	7093661550	626687513450	696969	four	123 creta	
5 Vishwas	vishwasgade@gmail.com	7093661520	626687513450	11235	two	123 creta	
6 Vishwas	vishwasgade@gmail.com	7093661550	626687513450	4565	four	1235 creta	
7 Vishwas	vishwasgade@gmail.com	7093661550	462421108219	as23333	two	123 creta	
8 Vishwas	vishwasgade@gmail.com	7093661550	626687513450	123456	two	10000 creta	

The LEASE_YOUR_VEHICLE Entity consists of attributes:

owner_name,customer_email,mobile_no,owner_id_proof_no,vehicle_reg_id,vehicle_type,km_driven,vehicle_model.

4. Reservation

RES_ID	CUSTOMER_EMAIL	LICENSE_NO	VEHICLE_ID	TYPE_OF_RIDE	BOOKING_START_DATE	BOOKING_END_DATE	ALTERNATE_MOBILE_NO	
1 Veh013	nayinisainithin@gmail.com	nayinisainithin@gmail.com	testing	Veh013	Outstation	2023-01-18 22:05	2023-01-20 03:05	2223232333
2 Veh030	gadevishwas6@gmail.com	gadevishwas6@gmail.com	testing	Veh030	out	2023-01-12 03:23	2023-01-28 22:23	1234567890
3 Veh001	vishwasgade@gmail.com	vishwasgade@gmail.com	hbmnmn	Veh001	Outstation	2023-01-27 00:23	2023-01-29 00:23	123456
4 Veh004	nayinisainithin@gmail.com	nayinisainithin@gmail.com	1234567891234	Veh004	Outstation	2023-01-12 09:42	2023-01-06 09:42	9121615657
5 Veh002	vishwasgade@gmail.com	vishwasgade@gmail.com	1234567891234	Veh002	IN STATION	2023-03-08 09:53	2023-01-14 09:54	7093661550
6 Veh002	nayinisainithin@gmail.com	nayinisainithin@gmail.com	1234567891234	Veh002	Outstation	2023-01-21 10:03	2023-01-22 10:03	987654321
7 Veh003	vishwasgade@gmail.com	vishwasgade@gmail.com	1234567891234	Veh003	Outstation	2023-01-19 13:29	2023-01-26 13:29	987654321
8 Veh016	vishwasgade@gmail.com	vishwasgade@gmail.com	1234567891234	Veh016	Outstation	2023-01-13 12:06	2023-01-18 15:06	987654321

The RESERVATION Entity consists of attributes : res_id, customer_email ,license_no, vehicle_id,type_of Ride,booking_start_date,booking_end Ride,alternate_mobile_no.

5. AVAILABILITY

	VEHICLE_ID	BOOKING_START_DATE	BOOKING_END_DATE
1	Veh013	2023-01-18 22:05	2023-01-20 03:05
2	Veh001	2023-01-27 00:23	2023-01-29 00:23
3	Veh030	2023-01-12 03:23	2023-01-28 22:23
4	Veh004	2023-01-12 09:42	2023-01-06 09:42
5	Veh002	2023-03-08 09:53	2023-01-14 09:54
6	Veh001	2022-01-05 10:00	2022-01-10 10:40
7	Veh001	2022-01-20 10:00	2022-01-25 10:40
8	Veh002	2023-01-21 10:03	2023-01-22 10:03
9	Veh003	2023-01-19 13:29	2023-01-26 13:29
10	Veh002	2022-01-05 10:00	2022-01-10 10:40
11	Veh003	2022-02-05 10:00	2022-03-10 10:40
12	Veh001	2023-01-20 10:00	2023-01-25 10:40
13	Veh001	2023-02-05 10:00	2023-03-10 10:40
14	Veh001	2023-01-12 01:09	2023-01-13 01:09
15	Veh016	2023-01-13 12:06	2023-01-18 15:06

The AVAILABILITY Entity consists of attributes: VEHICLE_ID, BOOKING_START_DATE, BOOKING_END_DATE

6. RENTAL_LOCATION

	RENTAL_LOC_ID	STATE	CITY	STREET_NAME	CONTACT_NO	EMAIL
1	Rloc01	Karnataka	Banglore	IndiraNagar	9551236551	Rloc1@gmail.com
2	Rloc02	Karnataka	Banglore	Yeshwanthpur	8992336611	Rloc2@gmail.com
3	Rloc03	Karnataka	Banglore	Marathalli	6309155897	Rloc3@gmail.com
4	Rloc04	Karnataka	Banglore	HSR Layout	7901528699	Rloc4@gmail.com
5	Rloc05	Karnataka	Banglore	Kormanglala	9888125331	Rloc5@gmail.com
6	Rloc06	Karnataka	Banglore	HennurLake	9912385512	Rloc6@gmail.com
7	Rloc07	Karnataka	Banglore	Whitefield	9553645678	Rloc7@gmail.com
8	Rloc08	Karnataka	Banglore	BTM layout	9399558866	Rloc8@gmail.com
9	Rloc09	Karnataka	Banglore	Majestic	8501526985	Rloc9@gmail.com
10	Rloc10	Karnataka	Banglore	Singasandra	7889661234	Rloc10@gmail.com

The RENTAL_LOCATION Entity consists of attributes : rental_loc_id ,state,city ,street_name,contact_no,email

7. PAYMENT

❖ CUSTOMER_NAME	❖ MOBILE_NO	❖ STATE_	❖ CITY	❖ ZIP_CODE	❖ RES_ID	❖ PAYMENT_ID	❖ CARD_NO	❖ NAME_ON_CAR
1 varshith	2121212121	sad	asd	222222	Veh013nayinisainithin@gmail.com	Veh013nayinisainithin@gmail.com222	122112212111	nithun
2 varshith	1234567890	karnataka	khammam	222222	Veh030gadevishwas6@gmail.com	Veh030gadevishwas6@gmail.com123	123456789	chintu
3 vishwas	(null)	(null)	(null)	(null)	Veh001vishwasgade@gmail.com	Veh001vishwasgade@gmail.com222	(null)	(null)
4 Nithin	9121615657	karnataka	Kasavanahalli	560035	Veh004nayinisainithin@gmail.com	Veh004nayinisainithin@gmail.com456	420046783765	Nithin
5 project completed	7093661550	delhi	Kasavanahalli	591234	Veh002vishwasgade@gmail.com	Veh002vishwasgade@gmail.com113	123578548745	Nithin
6 Nithin	987654321	karnataka	Kasavanahalli	560035	Veh002nayinisainithin@gmail.com	Veh002nayinisainithin@gmail.com765	98765432112	Nithin
7 vishwas	987654321	karnataka	khammam	233433	Veh003vishwasgade@gmail.com	Veh003vishwasgade@gmail.com456	789456123	Nithin
8 vishwas	987654321	delhi	yerragadda	233433	Veh016vishwasgade@gmail.com	Veh016vishwasgade@gmail.com123	123456789354	ASD

The PAYMENT Entity consists of attributes : customer_name ,mobile_no ,state_ ,city,zip_code,res_id,payment_id,card_no,name_on_card

8. COMPLAINT

❖ CUS_NAME	❖ CUSTOMER_EMAIL	❖ MOBILE_NO	❖ SUBJECT	❖ MESSAGE
1 Vishwas	vishwasgade@gmail.com	7093661551	testing	dvd
2 Vishwas	vishwasgade@gmail.com	7093661551	testing	Testing
3 Vishwas	vishwasgade@gmail.com	7093661551	testing	testing error print
4 Vishwas	vishwasgade@gmail.com	7093661551	Regarding Break Failure	adfaad
5 Vishwas	vishwasgade@gmail.com	7093661536	checking	presentation

The COMPLAINT Entity consists of attributes : cus_name ,customer_email ,mobile_no ,subject,message

9. EMPLOYEES

❖ EMPLOYEES_ID	❖ EMPLOYEE_NAME	❖ EMPLOYEE_CONTACT	❖ EMPLOYEE_EMAIL	❖ EMPLOYEE_PASSWORD
1 TBR001	Vishwas	7093661550	vishwasgade@gmail.com	TBR1
2 TBR002	Nithin	9121265580	nithin@gmail.com	TBR2
3 TBR003	Varshith	9553525496	varshith@gmail.com	TBR3
4 TBR004	Nikhil	9866425817	nikhil@gmail.com	TBR4
5 TBR005	Keerthan	9515751985	keerthan@gmail.com	TBR5
6 TBR006	Uthej	9182355022	uthej@gmail.com	TBR6
7 TBR007	Chethan	9212654896	chethan@gmail.com	TBR7
8 TBR008	Teja	8215069121	teja@gmail.com	TBR8
9 TBR009	Mani	9663995125	mani@gmail.com	TBR9
10 TBR010	Ashwaj	9615578812	Ajju@gmail.com	TBR10
11 TBR011	Nithin	9765512455	nithi@gmail.com	TBR11
12 TBR012	Guru	7088166235	guru@gmail.com	TBR12
13 TBR013	Hitesh	8366597145	Hitesh@gmail.com	TBR13
14 TBR014	Nitish	9821236899	nitishh@gmail.com	TBR14
15 TBR015	Kamal	7588822129	kamal@gmail.com	TBR15

The EMPLOYEES Entity consists of attributes : employees_id ,employee_name ,employee_contact,employee_email,employee_password

IMPLEMENTATION:

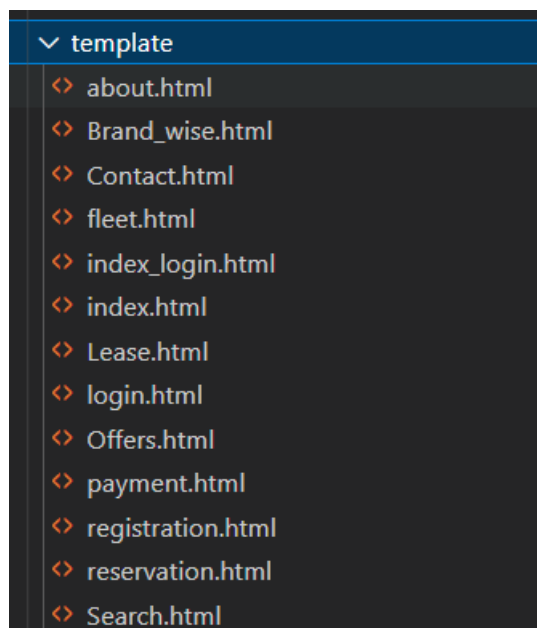
The main folder in the project will be the rental folder and it is created by using the Django command “py -m django startproject rental” and in that their will some dub folders created automatically.

And two another folders are created manually those are Template and Static. Where all the html pages that are used in the web page will be kept in that template folder. And in the static the all the CSS files, JS files and the images that we used in the web page will be stored in the static folder

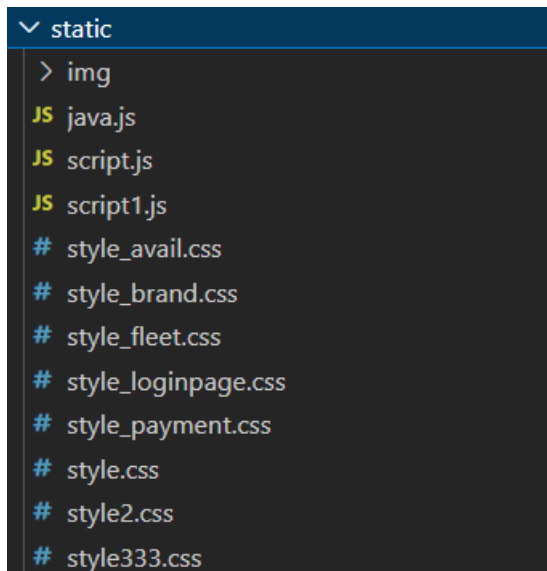
Along with these folders their will be another 14 folders which are apps or modules in the web page they are

Aboutpage,availability,bikeselection,contactpage,fleetpage,indexloginpage,indexpage,leasepage,loginpage,offerspage,payment,registrationpage,rentals,reservation . In each of the module folder their will be a views.py file where we will write the logic of the html pages that is which html is to be loaded and what data should be displayed and how it is displayed. The Sql queries to retrieve data from the data base will also be executed in the views.py file of each module or app

Template:- Template is the folder in the Django project folder where all the HTML pages are stored in it. And it has 13 HTML pages based on the project

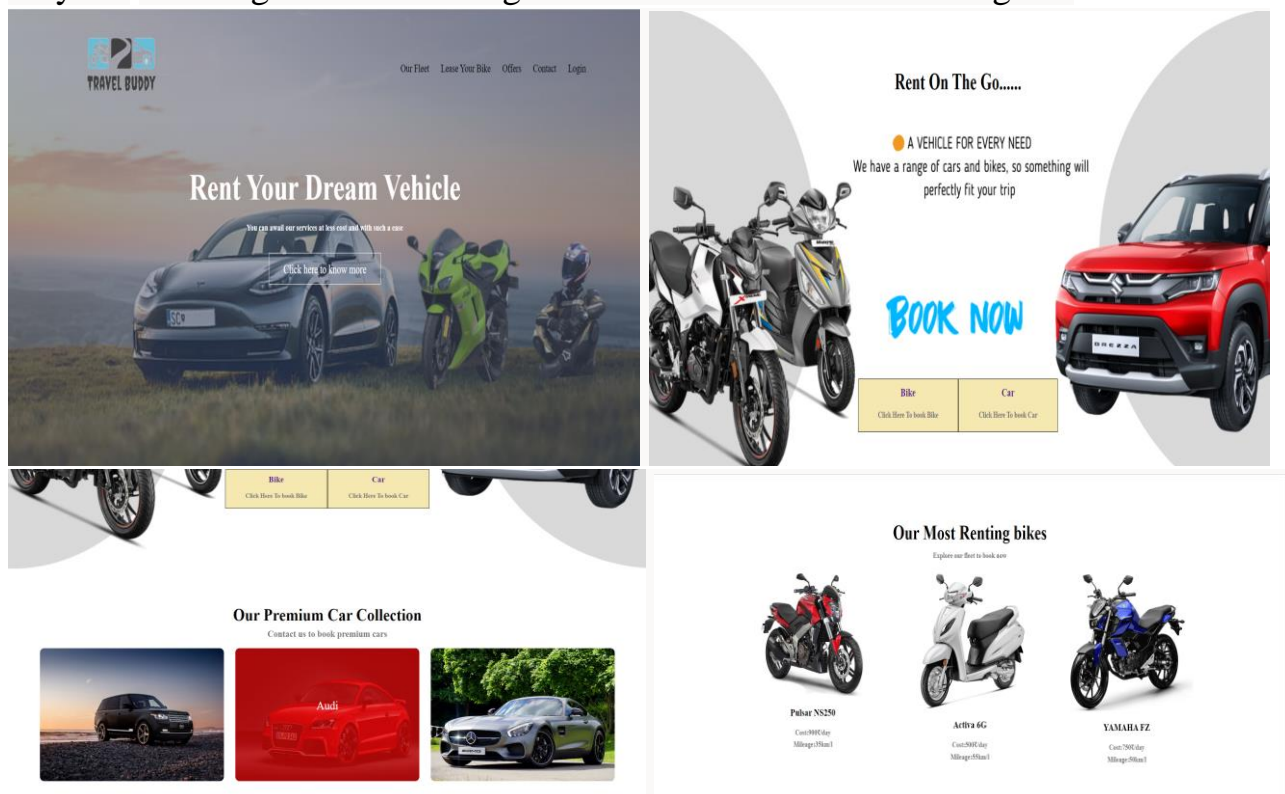


Static :- Static is the folder in the Django project folder where all the CSS files, JS files and the images that we used in the HTML pages are stored and retrieved we ever needed from the html files. This folder has 2 JS files , 8 CSS files and images used in img



FEATURES OF SYSTEM:

1. **INDEXPAGE :-** It is the first webpage where we will load the home page HTML file that is index.html where all the basic info of rentals is displayed and to view any other feature in indexpage user should login so next page will be loginpage. A function is defined in the views.py file. And in view.py we will just render the page because no use of database is done here. named logincars and in this function the code is return in such a way that, the 2nd page that is indexlogout.html is loaded if and only if the data that entered in the front end is matched with the database table named customers, then only the indexlogout.html is navigated other wise fail.html is navigated





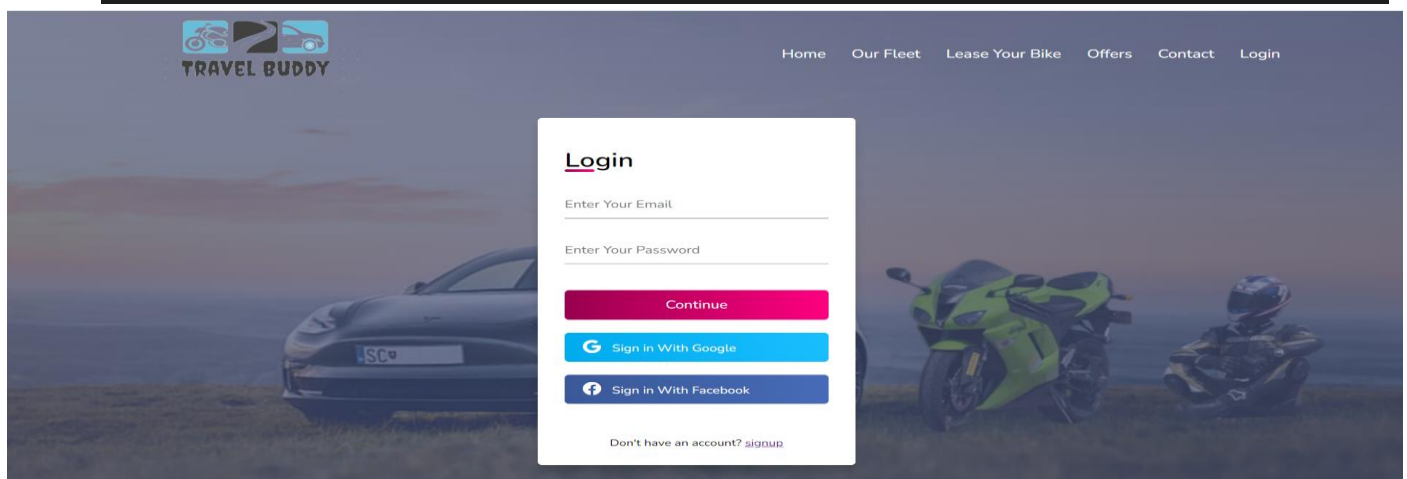
2. Loginpage :- Loginpage is the next page because from index user has to login to do any operation. So user data i.e customer data has to be taken from database so html and database has to be connected using Django so we have write a python module and sql query to retrieve data from database customers table

```

views.py ...loginPage X  index_login.html  views.py ...payment  views.py ...bikeselection  index.html  fleet.html

rentals > loginPage > views.py > loginPage
1  from django.shortcuts import render
2  import cx_Oracle
3  username=''
4  passwordd=''
5  # Create your views here.
6  def loginPage(request):
7      global username,passwordd
8      if request.method=="POST":
9          connStr = 'vishwas/vishwasgama@localhost:1521/xe'
10         #connStr = cx_Oracle.connect(user='project', passw ord='project', dsn='localhost:1521/xe')
11         conn = cx_Oracle.connect(connStr)
12         cur = conn.cursor()
13         d=request.POST
14         for key,value in d.items():
15             if key=="email":
16                 username=value
17             if key=="pass":
18                 passwordd=value
19         sql = "SELECT * FROM CUSTOMERS WHERE Customer_Email='{}' AND customer_password='{}'".format(username,passwordd)
20         print(sql)
21         cur.execute(sql)
22         a=tuple(cur.fetchall())
23         if a==():
24             context = {}
25             'message': 'The username or password is incorrect.'
26         }
27         return render(request,'login.html',context)
28     else:
29         return render(request,"index_login.html")
30     return render(request,'login.html')

```



3. Registration page: If the user is new user has to do the registration so under login signup option is present it will redirect the user to the registration page and in registration.html user will provide his details so we have to store that data in order to fetch and use in login page so we will connect to oracle and sql query will be given to insert the data into database data will be stored in customers table

```
from django.shortcuts import render
import cx_Oracle
# Create your views here.
fullname=''
Aadhar =''
User_email=''
phone_no =''
pass_word =''
gender=''

def registrationpage(request):
    global fullname,Aadhar,User_email,phone_no,pass_word,gender
    if request.method=="POST":
        connStr = 'vishwas/vishwasgama@localhost:1521/xe'
        conn = cx_Oracle.connect(connStr)
        cur = conn.cursor()
        d=request.POST
        #m.close()
        for key,value in d.items():
            if key=="NAME":
                fullname=value
            if key=="adrno":
                Aadhar=value
            if key=="email":
                User_email=value
            if key=="contact":
                phone_no=value
            if key=="pass":
                pass_word=value
            if key=="gender":
                gender=value
        sqltxt="insert into customers values('{}','{}','{}','{}','{}','{}')".format(fullname,Aadhar,User_email,phone_no,pass_word,gender)
        try:
            cur.execute(sqltxt)
            conn.commit()
```

TRAVEL BUDDY

Home Our Fleet Lease Your Bike Offers Contact Login

Registration

Full Name	AADHAR NO
<input type="text" value="Enter your name"/>	<input type="text" value="Enter your AADHAR NO"/>
Email	Phone Number
<input type="text" value="Enter your email"/>	<input type="text" value="Enter your number"/>
Password	Confirm Password
<input type="text" value="Enter your password"/>	<input type="text" value="Confirm your password"/>
Gender	
<input type="text" value="Enter M or F"/>	

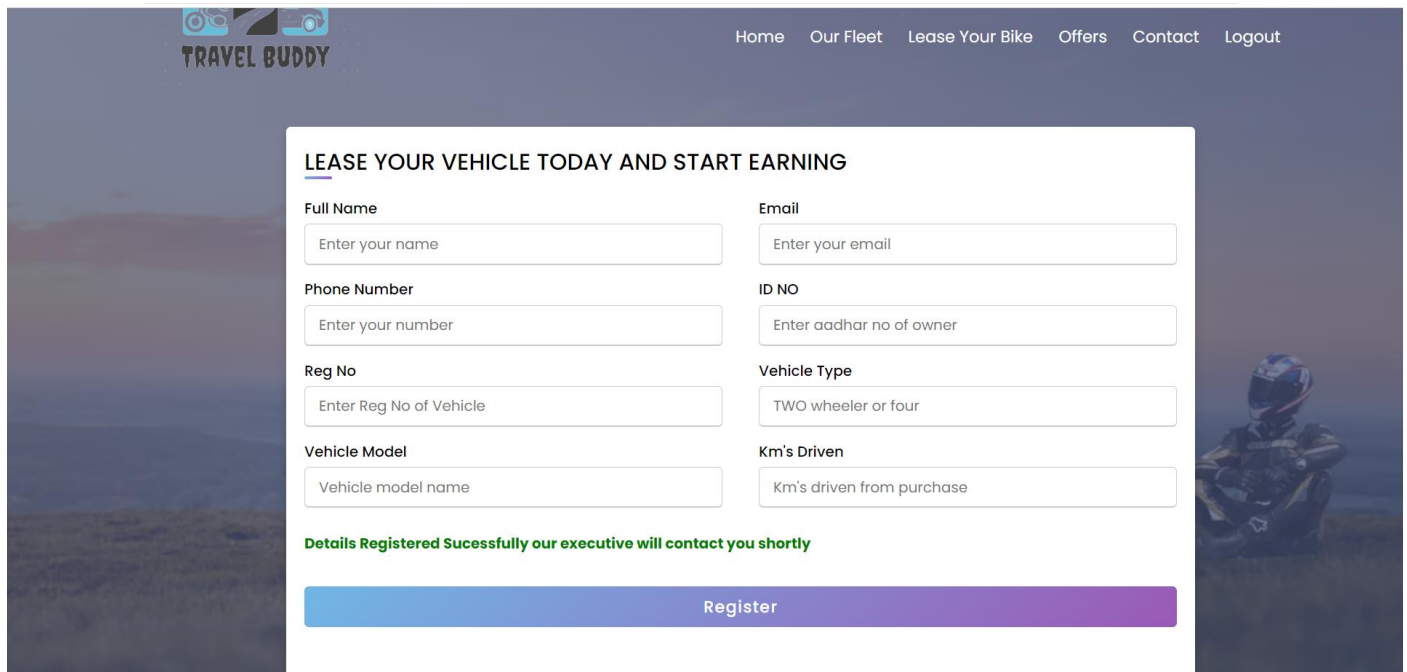
[Register](#)

Already Have an account?
[login here](#)

4. Leasepage :- In our system users can lease their own vehicle so they have provide us the details so that we can contact them later. For this data has to be entered into lease_your_vehicle table.

```
        phone_no=value
    if key=="adrno":
        Aadhar=value
    if key=="rgno":
        Vehrgid=value
    if key=="vehtype":
        vehtype=value
    if key=="kms":
        kms_driven=value
    if key=="vehmodel":
        vehmodel=value

    sqltxt="insert into Lease_your_vehicle values('0','0','0','0','0','0','0','0').format(fullname,User_email,phone_no,Aadhar,Vehrgid)
    try:
        cur.execute(sqltxt)
        conn.commit()
        context1={
            'message1': 'Details Registered Sucessfully our executive will contact you shortly '
        }
    except cx_Oracle.IntegrityError as e:
        error, =e.args
        context2 = {
            'message2': 'Vehicle with same regid present'
        }
    return render(request,'Lease.html',context1)
except cx_Oracle.IntegrityError as e:
    error, =e.args
    context2 = {
        'message2': 'Vehicle with same regid present'
    }
    return render(request,'Lease.html',context2)
else:
    return render(request,'Lease.html')
```



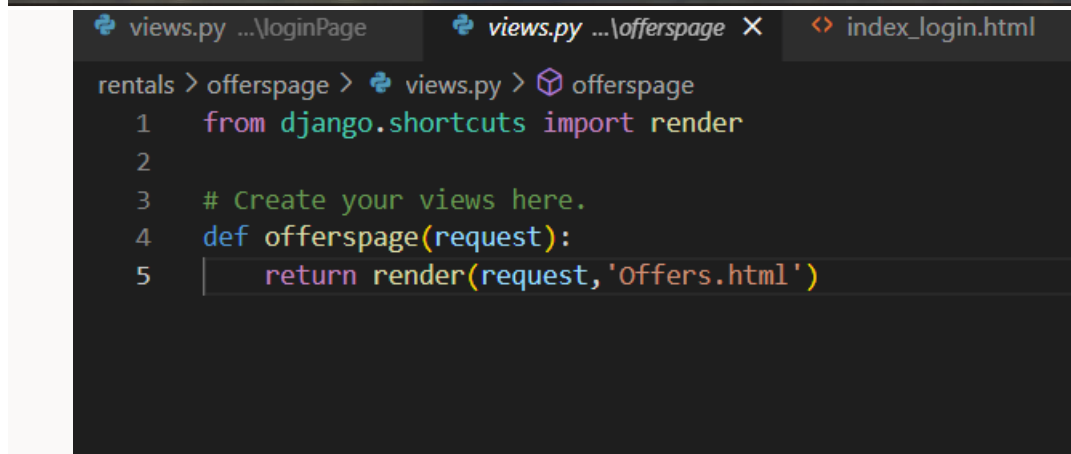
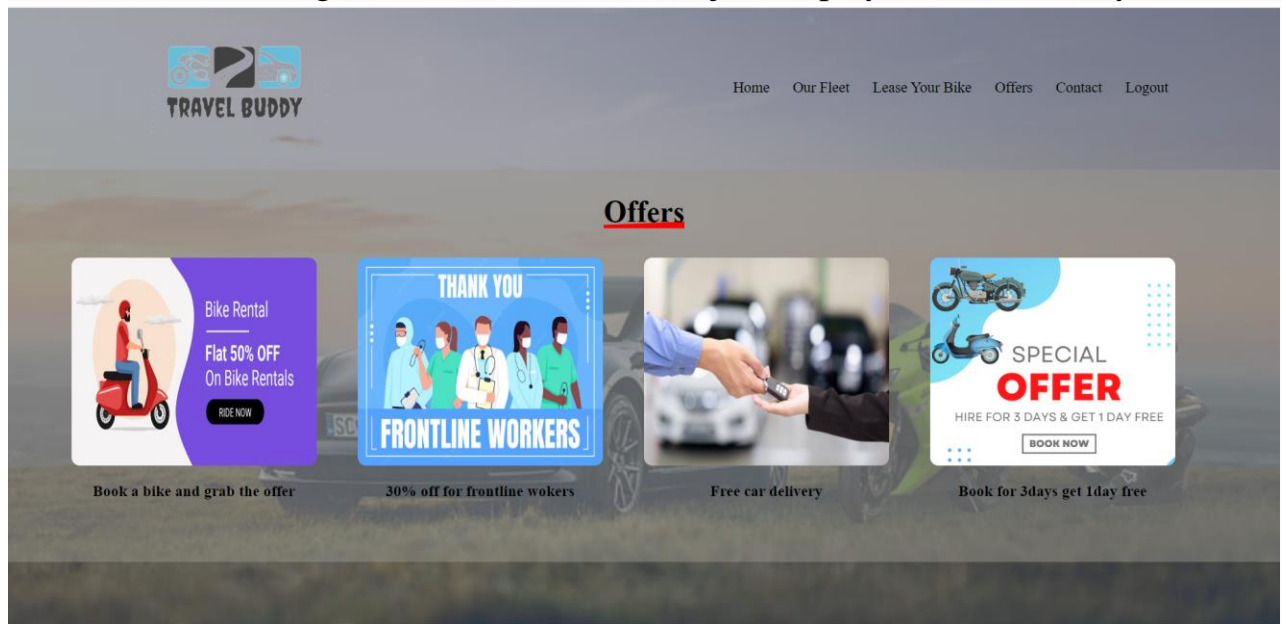
The screenshot shows a web application interface for 'TRAVEL BUDDY'. The navigation bar includes links for Home, Our Fleet, Lease Your Bike, Offers, Contact, and Logout. The main content area features a registration form titled 'LEASE YOUR VEHICLE TODAY AND START EARNING'. The form is divided into two columns of input fields. The left column contains fields for Full Name, Phone Number, Reg No, and Vehicle Model. The right column contains fields for Email, ID NO, Vehicle Type, and Km's Driven. Below the input fields, a green message states 'Details Registered Sucessfully our executive will contact you shortly'. At the bottom of the form is a large blue 'Register' button. The background of the page shows a person riding a motorcycle.

Full Name	Email
<input type="text" value="Enter your name"/>	<input type="text" value="Enter your email"/>
Phone Number	ID NO
<input type="text" value="Enter your number"/>	<input type="text" value="Enter aadhar no of owner"/>
Reg No	Vehicle Type
<input type="text" value="Enter Reg No of Vehicle"/>	<input type="text" value="TWO wheeler or four"/>
Vehicle Model	Km's Driven
<input type="text" value="Vehicle model name"/>	<input type="text" value="Km's driven from purchase"/>

Details Registered Sucessfully our executive will contact you shortly

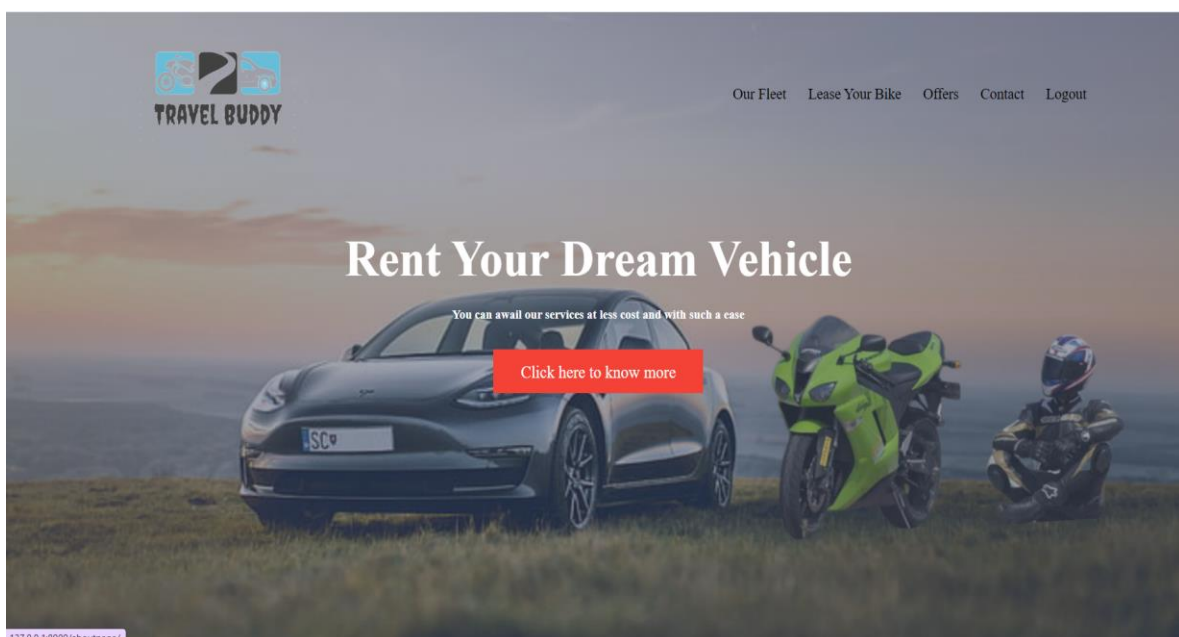
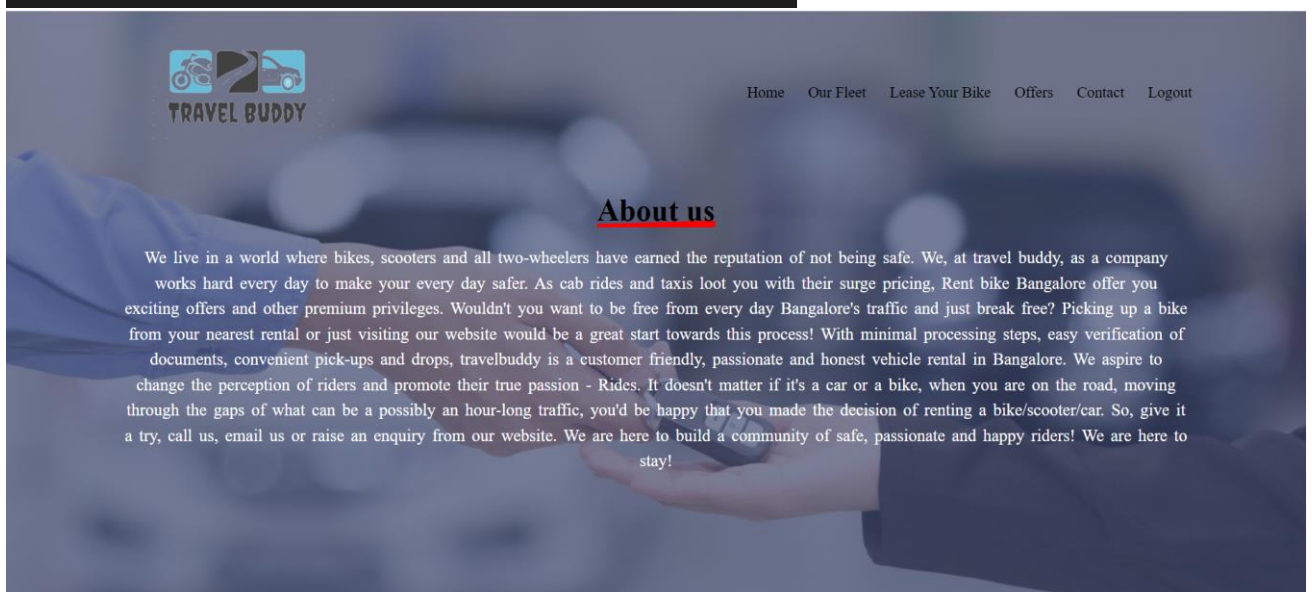
Register

5. Offers:- Users can know the current offers provided by the rental management .for that database fetching is not needed so we will just display the data directly from html



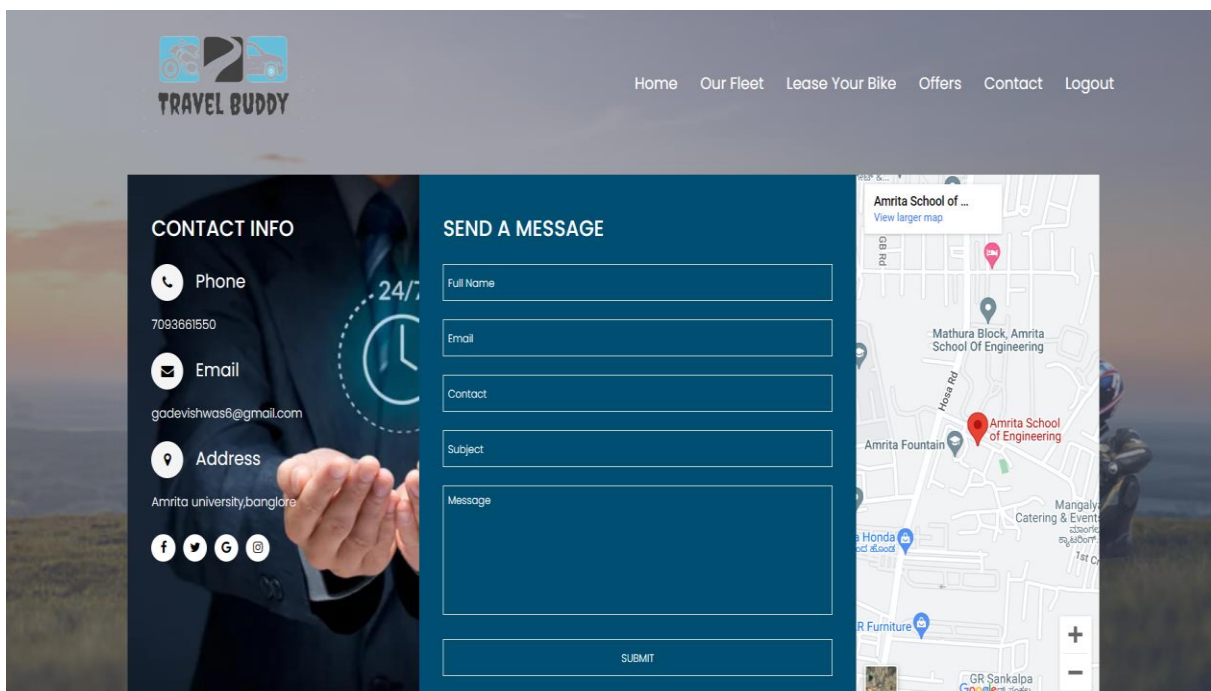
6. Aboutpage:- Index page consists of basic details of rentals if user want to know about the system more perfectly then user can select the Click here to know more which redirects to the Aboutus page in this also database fetching is not needed so we will directly render the html page:

```
rentals > aboutpage > views.py > aboutpage
1  from django.shortcuts import render
2
3  # Create your views here.
4  def aboutpage(request):
5      return render(request, 'about.html')
```



7. Contactpage: If user has any complaints he can send us his complaint through this page it consists of data which help us to solve the issue so if user fill the contact form it has to be stored in database for that database link has to be formed so view.py code is given to connect to the database and insert the data into database.

```
def contactpage(request):
    global fullname, User_email, phone_no, subject, message
    if request.method=="POST":
        connStr = 'vishwas/vishwasgama@localhost:1521/xe'
        conn = cx_Oracle.connect(connStr)
        cur = conn.cursor()
        d=request.POST
        #m.close()
        for key,value in d.items():
            if key=="NAME":
                fullname=value
            if key=="email":
                User_email=value
            if key=="phno":
                phone_no=value
            if key=="subject":
                subject=value
            if key=="message":
                message=value
        sqltxt="insert into COMPLAINT values('{}','{}','{}','{}','{}')".format(fullname,User_email,phone_no,subject,message)
        rows_affected = cur.execute(sqltxt)
        conn.commit()
        if rows_affected == 0:
            message = "Try Again"
        else:
            message = "Data entered Successfully"
        return render(request,'Contact.html',{'message': message})
    return render(request,'Contact.html')
```



8. Fleetpage: User has to start the booking process from fleetpage it consists of different bikes and car brands. User can select the brand accordingly, and then vehicle selection page is displayed in which all the vehicles of that particular brand are present. To display the vehicles from that particular brand data from database has to be fetched from vehicle table so sql command is given in view file so that all vehicles from that brand are fetched

```
from django.shortcuts import render
import cx_Oracle

# views.py

def fleetpage(request):
    if request.method=="GET":
        company = request.GET.get('company', None)
        if company:
            connStr = 'vishwas/vishwasgama@localhost:1521/xe'
            conn = cx_Oracle.connect(connStr)
            cur = conn.cursor()
            sql1 = "SELECT * FROM vehicle WHERE vehicle_brand = '{}'.format(company)"
            print(sql1)
            cur.execute(sql1)

            rows = cur.fetchall()
            print(rows)
            if not rows:
                return render(request, 'fail.html')
            else:
                # Create a list of dictionaries, where each dictionary represents a row of data
                data = [{'vehicle_id': row[0], 'vehicle_brand': row[1], 'vehicle_model': row[2], 'vehicle_type': row[3], 'color': row[4], 'sea': row[5]} for row in rows]
                # Render the other template and pass the data as a context parameter
                return render(request, 'Brand_wise.html', {'data': data})
        return render(request, 'fleet.html')
```



[Home](#) [Our Fleet](#) [Lease Your Bike](#) [Offers](#) [Contact](#) [Logout](#)

Our Bike Brands Collection

Click on your fav Brand to explore more



127.0.0.1:8000/fleetpage/?company=KTM

9. Vehicleselection: After user selects the brand view code sql command is executed and data will be fetched from vehicles table and passed on to vehicleselection page. Here user can select his/her preferred vehicle. And the in view code after selection of preferred vehicle all the details of that particular vehicle are to be extracted from database vehicle table so again we to execute the sql command and fetch the data.



[Home](#) [Our Fleet](#) [Lease Your Bike](#) [Offers](#) [Contact](#) [Logout](#)

Our Most Renting bikes

Click on your fav model to explore more



200 Duke

Cost: 4200€/day



390 Duke

Cost: 6000€/day



RC 125

Cost: 13000€/day

In above figure ktm brand is selected and bikes available in ktm are displayed.

```
from django.shortcuts import render
import cx_Oracle

def bikeselection(request):
    if request.method == "GET":
        vehicle_model = request.GET.get('vehicle_model', None)
        if vehicle_model:
            connStr = 'vishwas/vishwasgama@localhost:1521/xe'
            conn = cx_Oracle.connect(connStr)
            cur = conn.cursor()
            sql = "SELECT * FROM vehicle WHERE vehicle_model = '{}'.format(vehicle_model)"
            print(sql)
            cur.execute(sql)
            rows = cur.fetchall()
            print(rows)
            if not rows:
                return render(request, 'fail.html')
            else:
                data1 = [{'vehicle_id': row[0], 'vehicle_brand': row[1], 'vehicle_model': row[2], 'vehicle_type': row[3], 'Vehicle_color': row[4]}]
                print(data1)
                return render(request, 'Search.html', {'data': data1})
    return render(request, 'Brand_wise.html')
```

10.Availability: In this user has to give the booking dates and can view the vehicle details. If the bike available in the preferred dates he will move on to confirmation page if the booking dates are not available he will be redirected to fleet page asking to change dates and try again.



[Home](#) [Our Fleet](#) [Lease Your Bike](#) [Offers](#) [Contact](#) [Logout](#)



Vehicle ID	Veh011
Vehicle Brand	KTM
Vehicle Model	200 Duke
Vehicle Type	Gear
Color	Black
Seating Capacity	2
Cost per Day	4200
Deposit Amount	4500
Rent Location ID	Rloc01

Enter your booking dates to go for booking page



Booking status

Your specified booking dates are not available for that vehicle.

OK

[Lease Your Bike](#) [Offers](#) [Contact](#) [Logout](#)

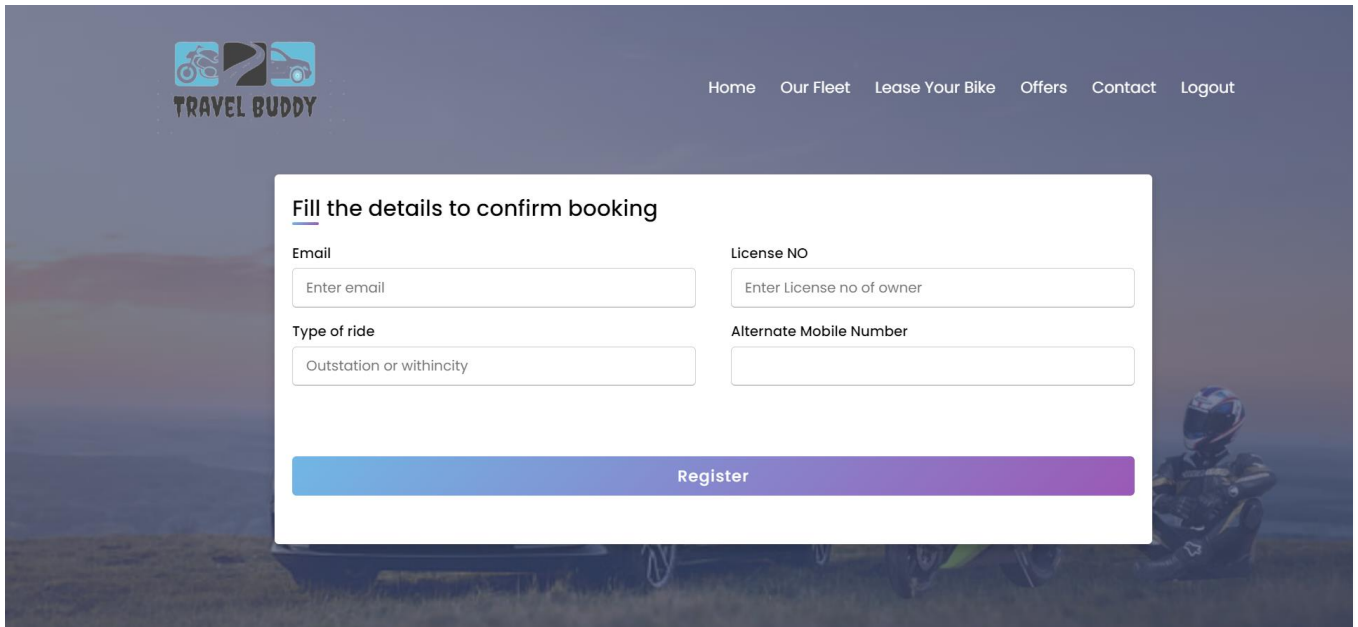
Our Bike Brands Collection

Click on your fav Brand to explore more



```
sq2 = "SELECT * FROM availability WHERE vehicle_id = '{ }' AND ((TO_DATE('{ }', 'YYYY-MM-DD HH24:MI') < booking_start_date AND TO_DATE('{ }', 'YYYY-MM-DD HH24:MI') < booking_start_date) OR (TO_DATE('{ }', 'YYYY-MM-DD HH24:MI') > booking_end_date AND TO_DATE('{ }', 'YYYY-MM-DD HH24:MI') > booking_end_date)).format(vehicle_id, start_formatted, end_formatted, start_formatted, end_formatted)"
```

11.Reservation: If the booking dates are available the user will be redirected to reservation page where user will be asked to confirm booking by entering his details. After user confirms the booking he will be redirected to the payment page



The screenshot shows the TRAVEL BUDDY website interface. At the top, there is a navigation bar with links: Home, Our Fleet, Lease Your Bike, Offers, Contact, and Logout. The main content area features a form titled "Fill the details to confirm booking". The form contains four input fields: "Email" (with placeholder "Enter email"), "License NO" (with placeholder "Enter License no of owner"), "Type of ride" (with placeholder "Outstation or withincity"), and "Alternate Mobile Number". A large blue "Register" button is positioned at the bottom of the form. The background of the website is a blurred image of a motorcycle and a rider.

```
from django.shortcuts import render, redirect
import cx_Oracle

# Create your views here.
def reservation(request):
    if request.method == 'POST':
        # Get the form data
        email = request.POST['email']
        start_formatted = request.POST['start']
        end_formatted = request.POST['end']
        vehicle_id = request.POST['id']
        license_no = request.POST['license_no']
        ride = request.POST['ride']
        alno = request.POST['alno']

        # Strip leading and trailing whitespace from the vehicle_id
        vehicle_id = vehicle_id.strip()

        # Concatenate the vehicle_id and email fields
        reservation_id = vehicle_id + email

        # Connect to the database
        connStr = 'vishwas/vishwasgama@localhost:1521/xe'
        conn = cx_Oracle.connect(connStr)
        context = {'vehicle_id': vehicle_id, 'start_formatted': start_formatted, 'end_formatted': end_formatted, 'email': email, 'license_no': license_no}
        return render(request, 'payment.html', context)
    else:
        return redirect('http://127.0.0.1:8000/fleetpage/?show_popup=True&source=page2')
```


12. Paymentpage: After user has confirmed the booking then he has to do the payment if payment has done then all the booking details have to be inserted into database so that another customer cannot book the bike on same timings for that again we will connect the database and using sql commands we will insert all the data into availability table, reservation table and payment table.

TRAVEL BUDDY

Home Our Fleet Lease Your Bike Offers Contact Logout

BILLING ADDRESS

Full Name :

Phone Number

Amount Paying Through Card

Address:

State :

Zip Code :

PAYMENT

Cards Accepted :
PayPal MasterCard American Express VISA

Name On Card :

Card Number :

Exp Month :

Exp Year :

CVV :

[Proceed To Checkout](#)

```
from django.shortcuts import render, redirect
from django.contrib import messages
import cx_Oracle

# Create your views here.
def payment(request):
    if request.method == 'POST':
        # Get the form data
        print(request.POST)
        email = request.POST['email']
        start_formatted = request.POST['start']
        end_formatted = request.POST['end']
        vehicle_id = request.POST['id']
        license_no = request.POST['license_no']
        ride=request.POST['ride']
        alno=request.POST['alno']
        user_name=request.POST['user_name']
        ph_no=request.POST['ph_no']
        amt=request.POST['amt']
        addr=request.POST['addr']
        state=request.POST['state']
        zp_code=request.POST['zp_code']
        nameoncard=request.POST['nameoncard']
        card_no=request.POST['card_no']
        exp_month=request.POST['exp_month']
        exp_year=request.POST['exp_year']
        cvv=request.POST['cvv']
        reservation_id=request.POST['reservation_id']

        # Strip leading and trailing whitespace from the vehicle_id
        vehicle_id = vehicle_id.strip()

        # Concatenate the vehicle_id and email fields
        reservation_id = vehicle_id + email
        payment_id= reservation_id+cvv
```

```

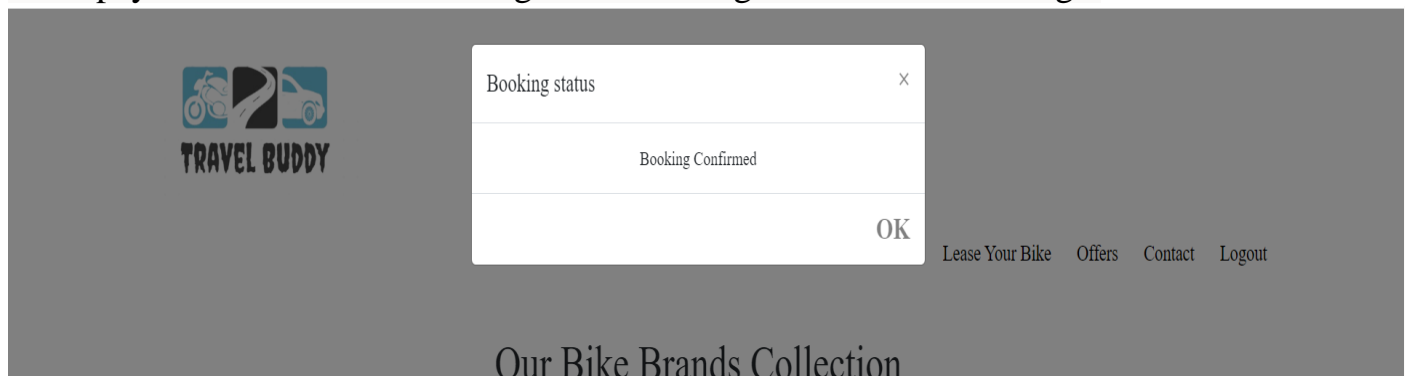
# Connect to the database
connStr = 'vishwas/vishwasgama@localhost:1521/xe'
conn = cx_Oracle.connect(connStr)
cur = conn.cursor()

# Use the form data in your SQL insert statement
sq1 = "INSERT INTO availability(vehicle_id, booking_start_date, booking_end_date) VALUES ('{}', TO_DATE('{}', 'YYYY-MM-DD HH24:MI'), TO_DATE('{}', 'YYYY-MM-DD HH24:MI'))"
sq2 = "INSERT INTO reservation(res_id, customer_email, license_no, vehicle_id, type_of_ride, booking_start_date, booking_end_date, alternate_mobile_no) VALUES ('{}', '{}', '{}', '{}', '{}', TO_DATE('{}', 'YYYY-MM-DD HH24:MI'), TO_DATE('{}', 'YYYY-MM-DD HH24:MI'), '{}')
sq3 = "INSERT INTO payment(customer_name, mobile_no, state, city, zip_code, res_id, payment_id, card_no, name_on_card, expiry_month, expiry_year, amount) VALUES ('{}', '{}', '{}', '{}', '{}', '{}', '{}', '{}', '{}', '{}', '{}', '{}')
print(sq1)
print(sq2)
print(sq3)

try:
    cur.execute(sq1)
    cur.execute(sq2)
    cur.execute(sq3)
    conn.commit()
    return redirect('http://127.0.0.1:8000/fleetpage/?show_popup=True&source=page3')
except cx_Oracle.IntegrityError as e:
    error, = e.args
    return redirect('http://127.0.0.1:8000/fleetpage/?show_popup=True&source=page2')
else:
    return render(request, 'payment.html')

```

After payment has done user will get the Booking confirmation message.



CONCLUSION:

We have developed responsive vehicle rental system webpage using HTML, CSS, Java, Django, and Oracle to store data. This combination of technologies allows for the creation of a dynamic and user-friendly website that can handle a large amount of data and provide efficient rental services to customers. The use of Oracle as the database management system ensures reliable and secure storage of customer and vehicle information. Overall, this webpage can provide a seamless and efficient rental experience for both customers and administrators.