PROJECT OVERVIEW

The KPM Beranang Vehicle Management Online System is a web-based platform designed to facilitate the rental and booking of vehicles provided by the college. The system aims to streamline the process of accessing college vehicles for various purposes, adding a new vehicle, keep tab on maintenance, and of course the bookings.

Target User and Accessibility:

KPMB Staff and Faculty: Staff and faculty members who need to book vehicles for official college-related purposes, such as attending conferences, conducting field trips, or transporting equipment. They can book various available vehicle such as cars, van, motorcycles and busses.

KPMB Students: Registered students that require transportation for events, excursions, or community outreach programs. Student's can only book cars and motorcycles.

KPMB Administrator: Administrative staff responsible for managing vehicle reservations, maintaining vehicle records, and overseeing the overall operation of the system.

Project Purpose:

Vehicle Booking and Rental: Users can easily book and rent college vehicles online for official purposes like conferences, field trips, or events.

Efficient Resource Management: Administrators can efficiently manage vehicle resources, track usage, and optimize scheduling to avoid conflicts.

User Accountability and Tracking: The system tracks user activities and booking history, ensuring transparency and accountability for vehicle usage.

Providing Transport to KPM students: Ease of transportation becomes an essential to all as some courses like the diploma students of Landscape and Horticultur needs a lot of mobility to go on site visits off class.

Streamlined Administrative Tasks: The administrative page simplifies tasks like managing bookings, handling maintenance, and vehicles inventory.

Type of Users:

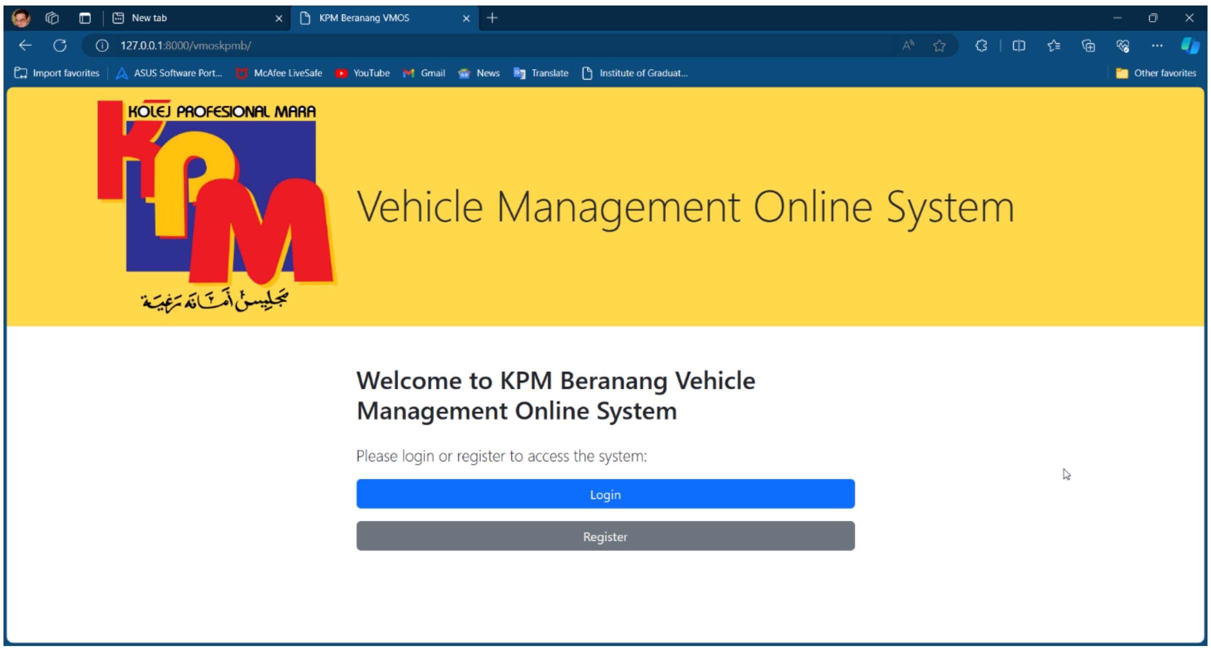
Superusers: Deals with any critical conflict in database, collecting raw data and granting administrator status to selected users.

Administrator: Task on adding and updating the vehicle and keep tab on maintenance ad have a special assigned pages for this user only.

Student and Lecturer Status: Able to book the actual vehicle one at a time only. Students are restricted to book only cars and motorcycles.

<u>WEBSITE FLOW:</u>

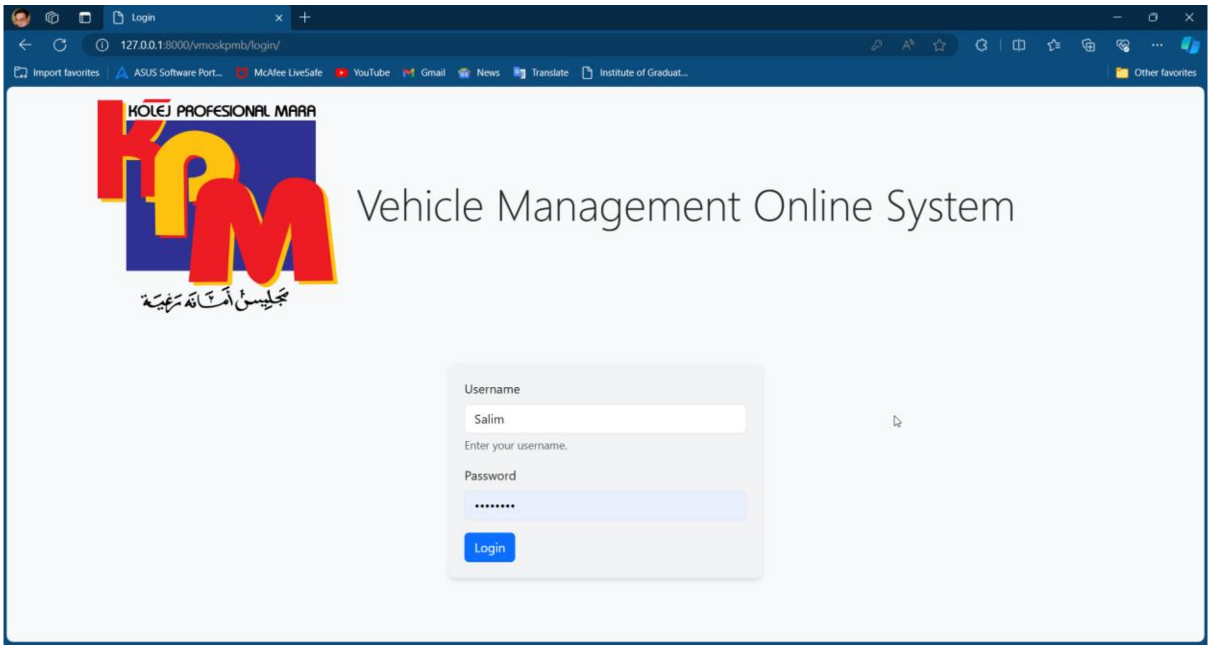
Administrator View:


Vehicle Management Online System

Welcome to KPM Beranang Vehicle Management Online System

Please login or register to access the system:

Administrators must be created thru the Django-admin page granted by the superuser. After registering with the superuser administrators can now proceed to login.


Vehicle Management Online System

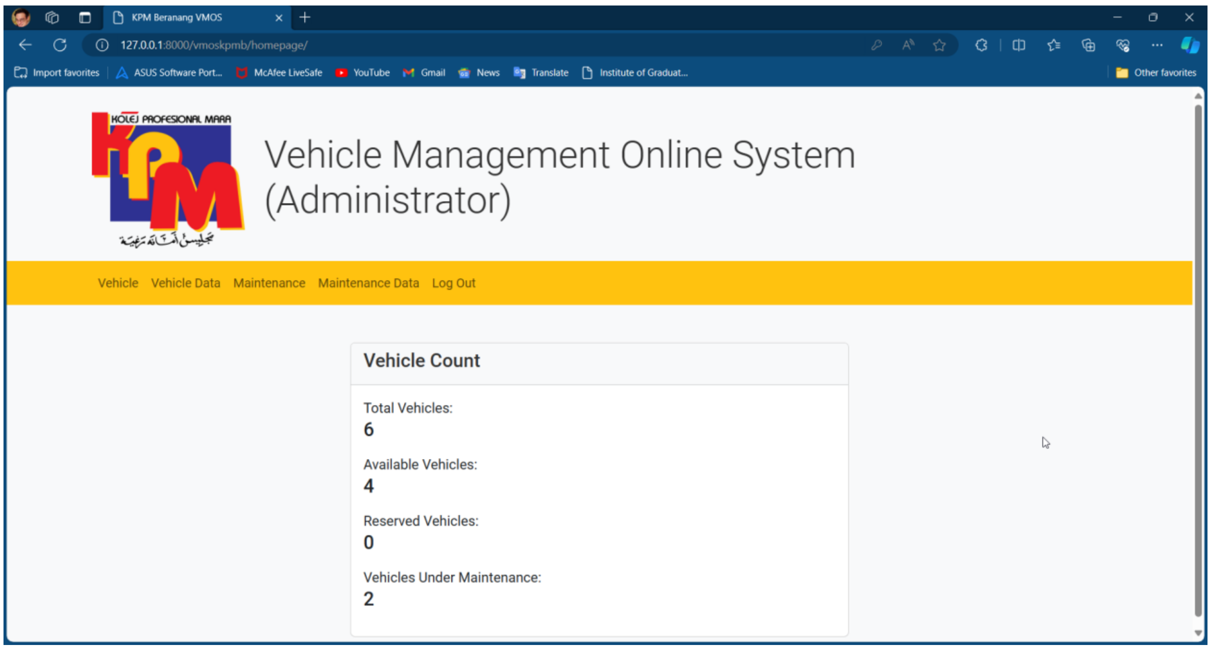
Username

Salim

Enter your username.

Password

••••••••


Vehicle Management Online System (Administrator)

Vehicle Count

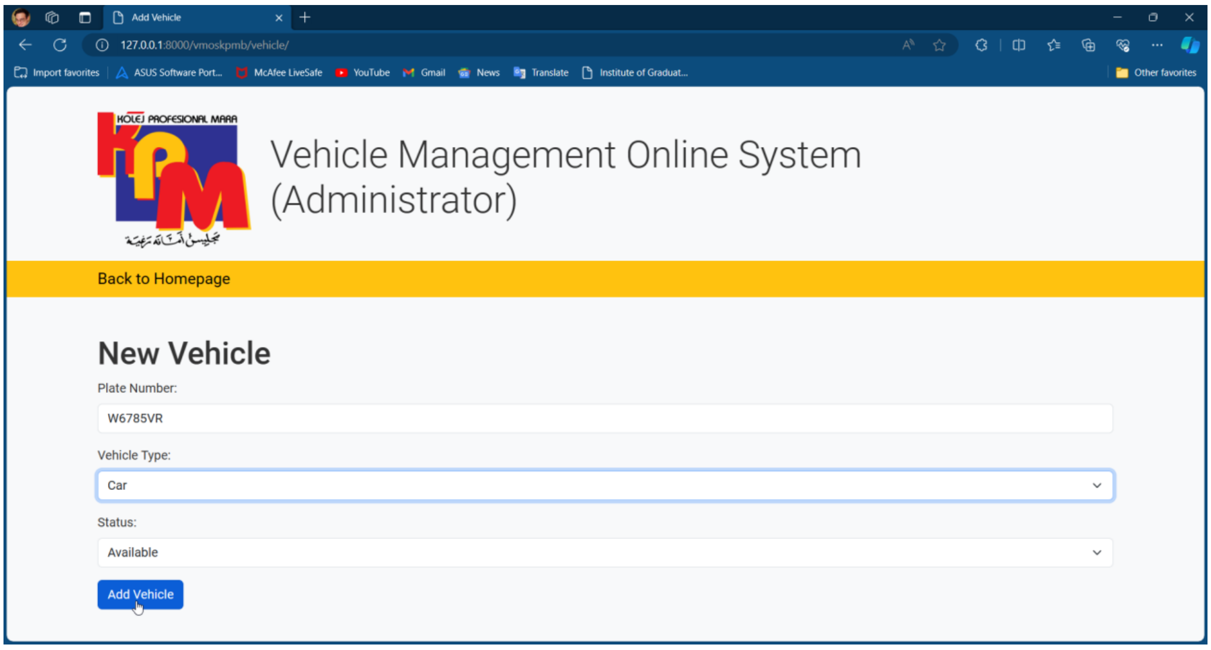
Total Vehicles:
6

Available Vehicles:
4

Reserved Vehicles:
0

Vehicles Under Maintenance:
2

After administrator logged in, it will be greeted by the homepage with dashboard on vehicle count which is also visible to student and lecturer status.


Vehicle Management Online System (Administrator)

Back to Homepage

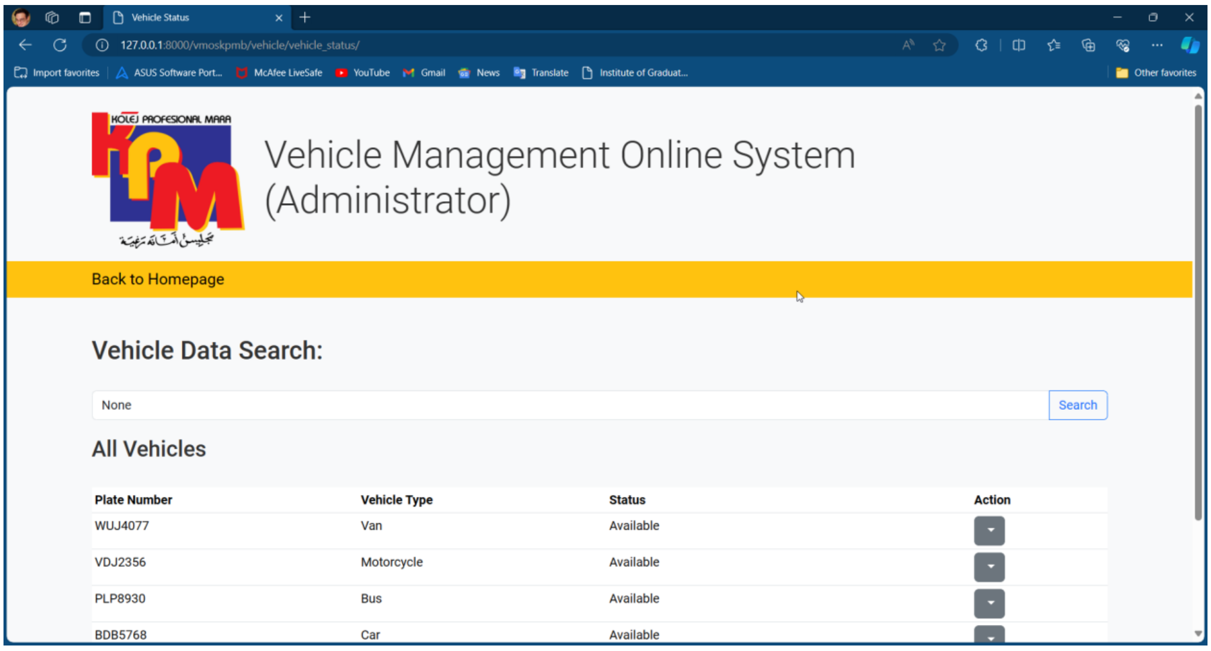
New Vehicle

Plate Number: W6785VR

Vehicle Type: Car

Status: Available

Firstly, we'll show the administration process of the website. The administrator click on Vehicle on the navigation bar to add a new vehicle after entering the appropriate information the user adds the vehicle and a message should appear saying "New vehicle has been added". Click on Back to Homepage to return.


Vehicle Management Online System (Administrator)

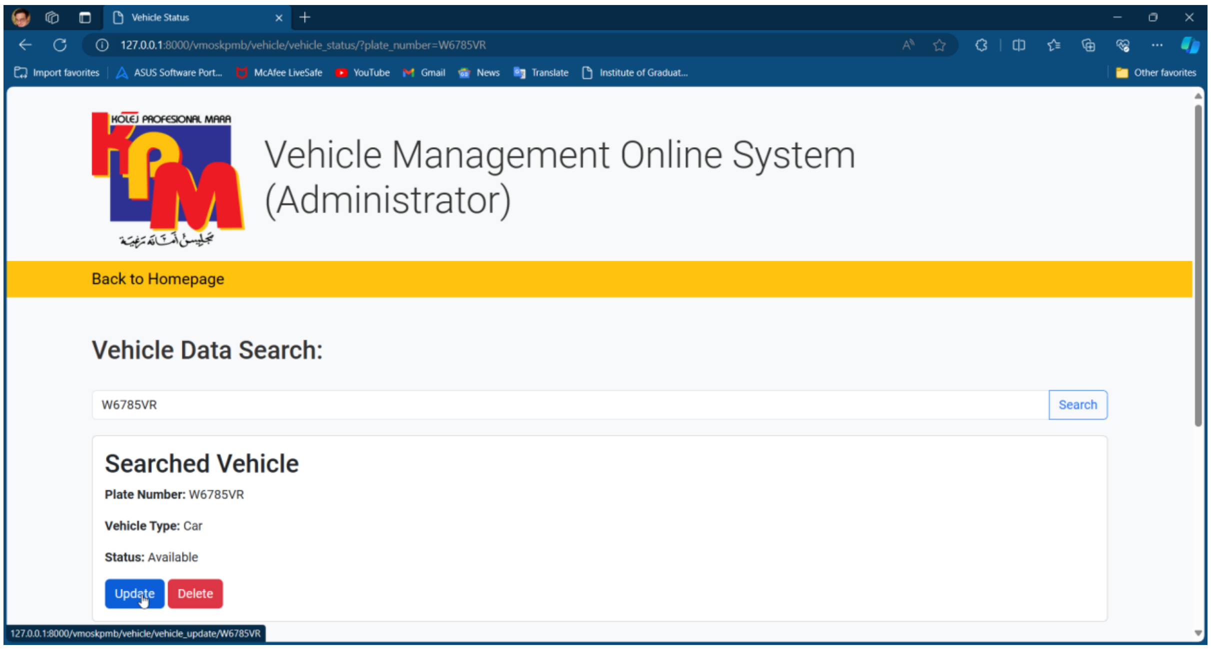
Back to Homepage

Vehicle Data Search:

None

All Vehicles

Plate Number	Vehicle Type	Status	Action
WUJ4077	Van	Available	
VDJ2356	Motorcycle	Available	
PLP8930	Bus	Available	
BDB5768	Car	Available	


Vehicle Status

127.0.0.1:8000/vmoskpmb/vehicle/vehicle_status/?plate_number=W6785VR

Vehicle Management Online System (Administrator)

Back to Homepage

Vehicle Data Search:

W6785VR

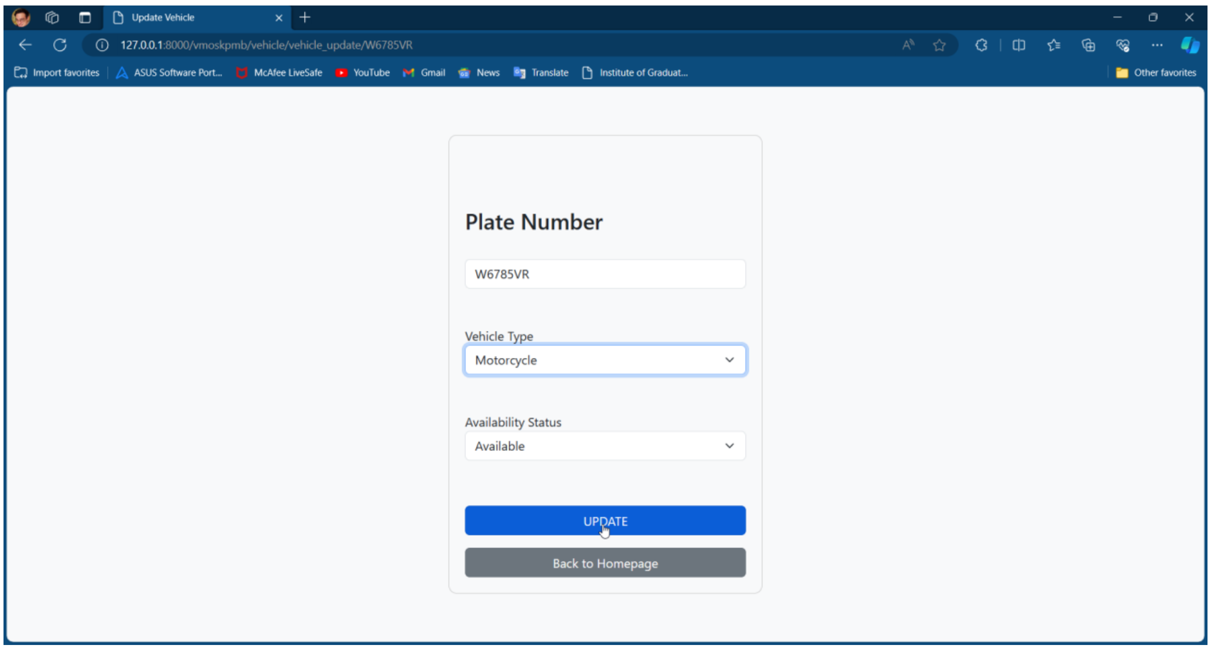
Searched Vehicle

Plate Number: W6785VR

Vehicle Type: Car

Status: Available

127.0.0.1:8000/vmoskpmb/vehicle/vehicle_update/W6785VR


Update Vehicle

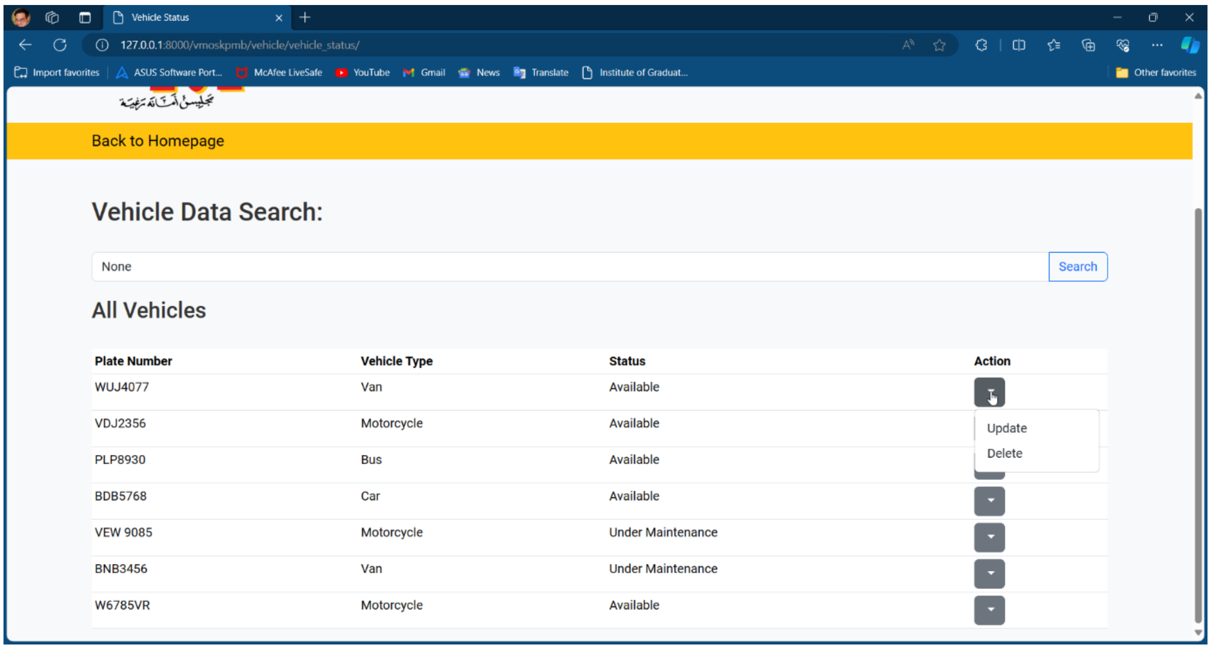
127.0.0.1:8000/vmoskpmb/vehicle/vehicle_update/W6785VR

Plate Number

W6785VR

Vehicle Type: Motorcycle

Availability Status: Available


Back to Homepage

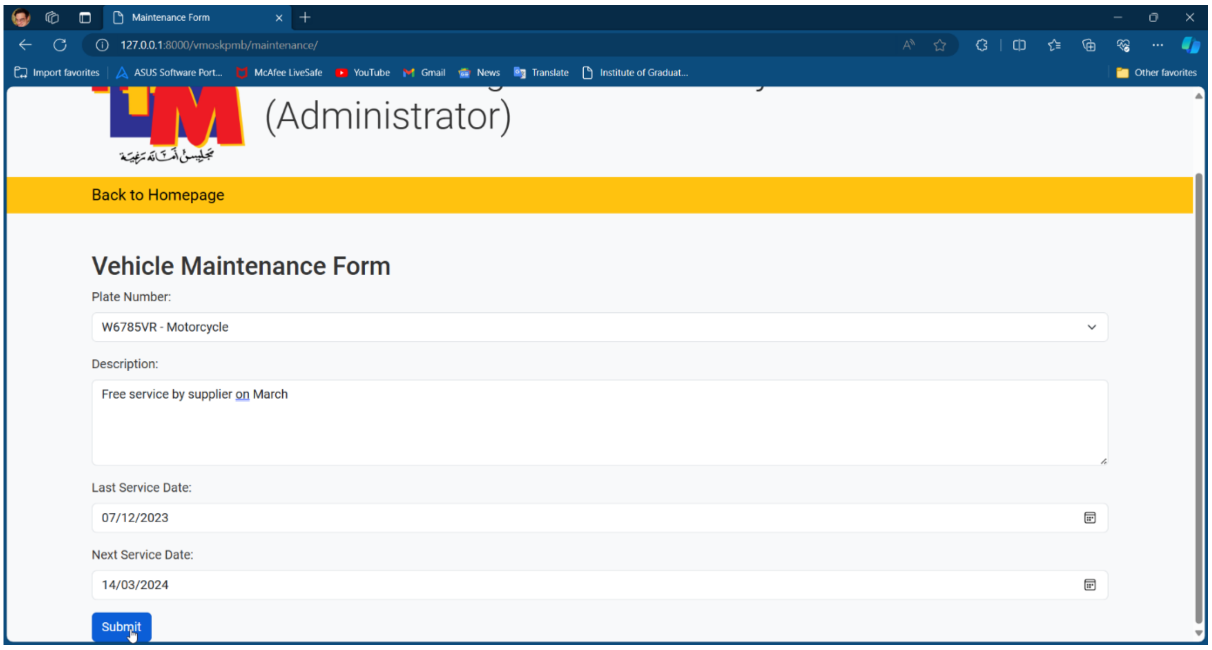
Vehicle Data Search:

None | Search

All Vehicles

Plate Number	Vehicle Type	Status	Action
WUJ4077	Van	Available	
VDJ2356	Motorcycle	Available	Update
PLP8930	Bus	Available	Delete
BDB5768	Car	Available	
VEW 9085	Motorcycle	Under Maintenance	
BNB3456	Van	Under Maintenance	
W6785VR	Motorcycle	Available	

Vehicle will only have four types (van, motorcycle, car, bus) and the Status of the vehicle will be available, reserved and under maintenance next one is about Maintenance. It is pretty much the same with Vehicle.


(Administrator)

Back to Homepage

Vehicle Maintenance Form

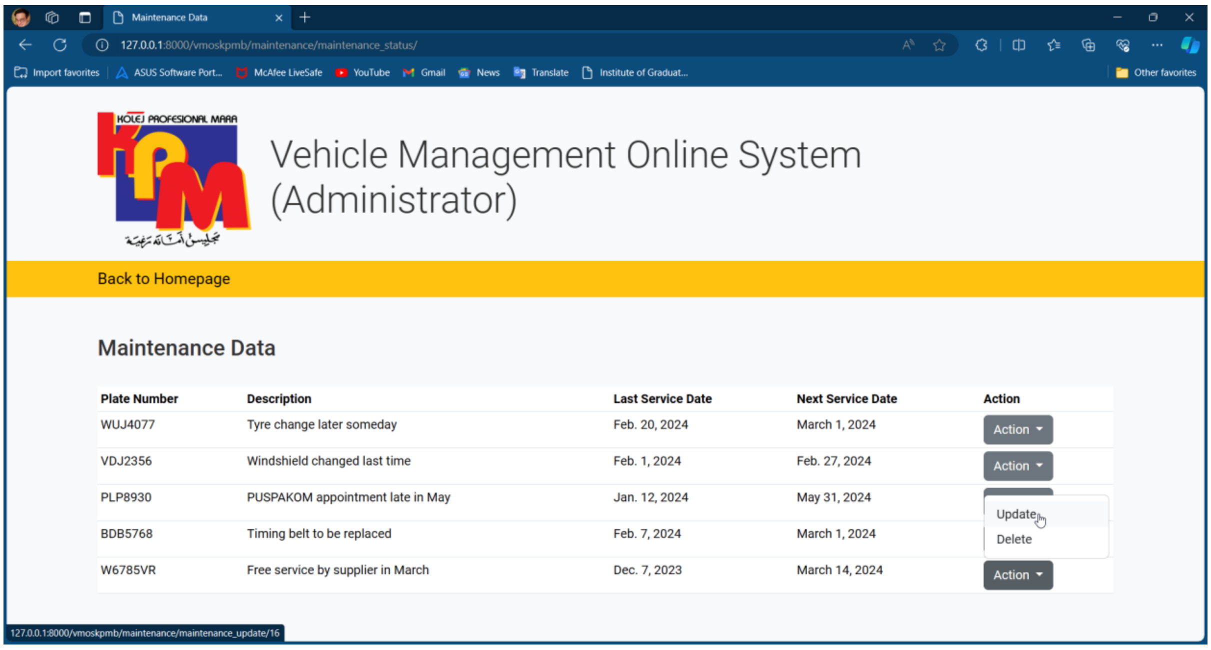
Plate Number:
W6785VR - Motorcycle

Description:
Free service by supplier on March

Last Service Date:
07/12/2023

Next Service Date:
14/03/2024

Submit


Vehicle Management Online System (Administrator)

Back to Homepage

Maintenance Data

Plate Number	Description	Last Service Date	Next Service Date	Action
WUJ4077	Tyre change later someday	Feb. 20, 2024	March 1, 2024	Action
VDJ2356	Windshield changed last time	Feb. 1, 2024	Feb. 27, 2024	Action
PLP8930	PUSPAKOM appointment late in May	Jan. 12, 2024	May 31, 2024	Update / Delete
BDB5768	Timing belt to be replaced	Feb. 7, 2024	March 1, 2024	
W6785VR	Free service by supplier in March	Dec. 7, 2023	March 14, 2024	Action

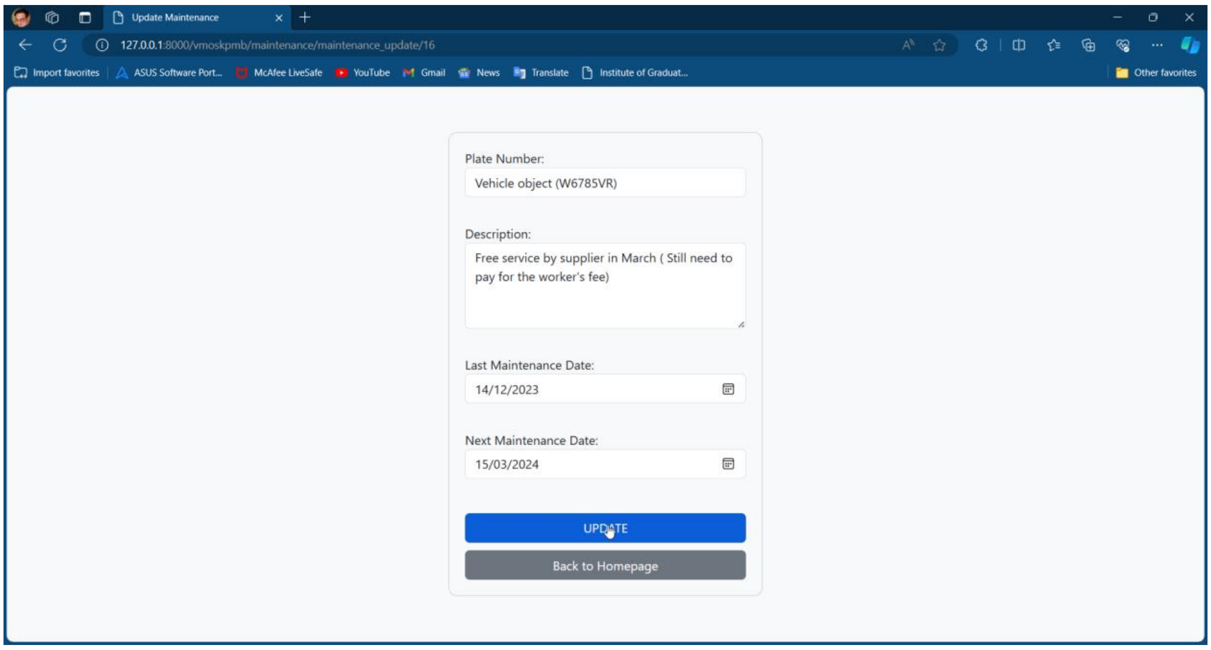


Plate Number: Vehicle object (W6785VR)

Description: Free service by supplier in March (Still need to pay for the worker's fee)

Last Maintenance Date: 14/12/2023

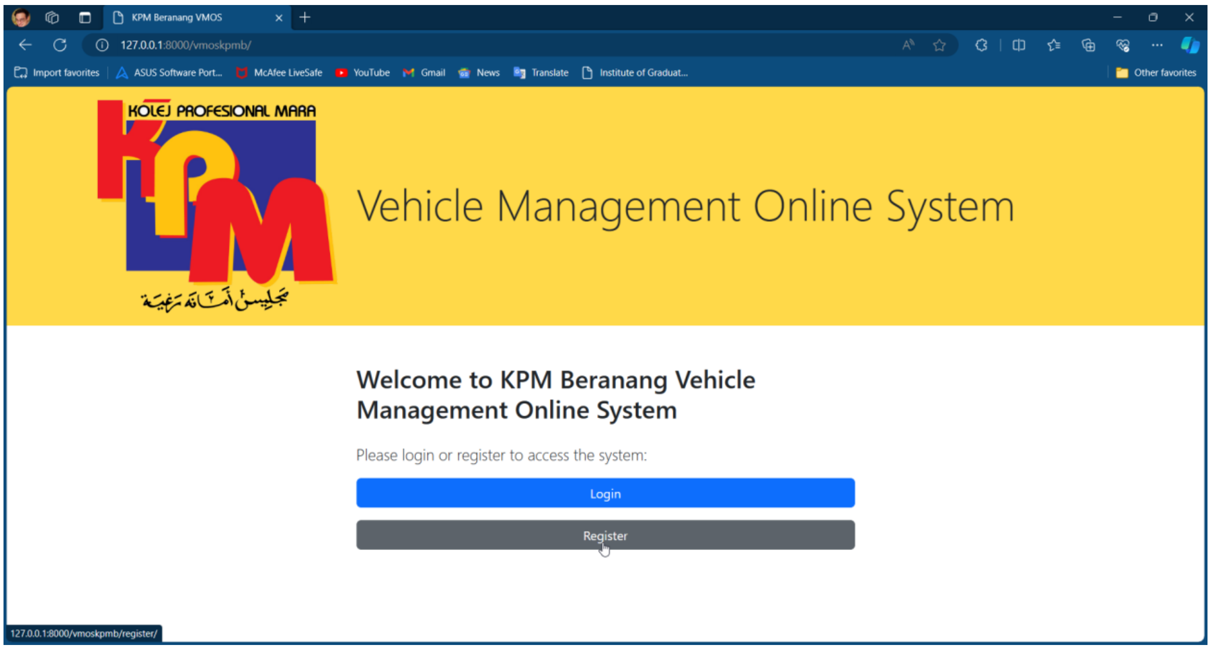
Next Maintenance Date: 15/03/2024

UPDATE

Back to Homepage

Concludes the administration role on the website. Clicking Log Out will throw you back to the Welcome page.

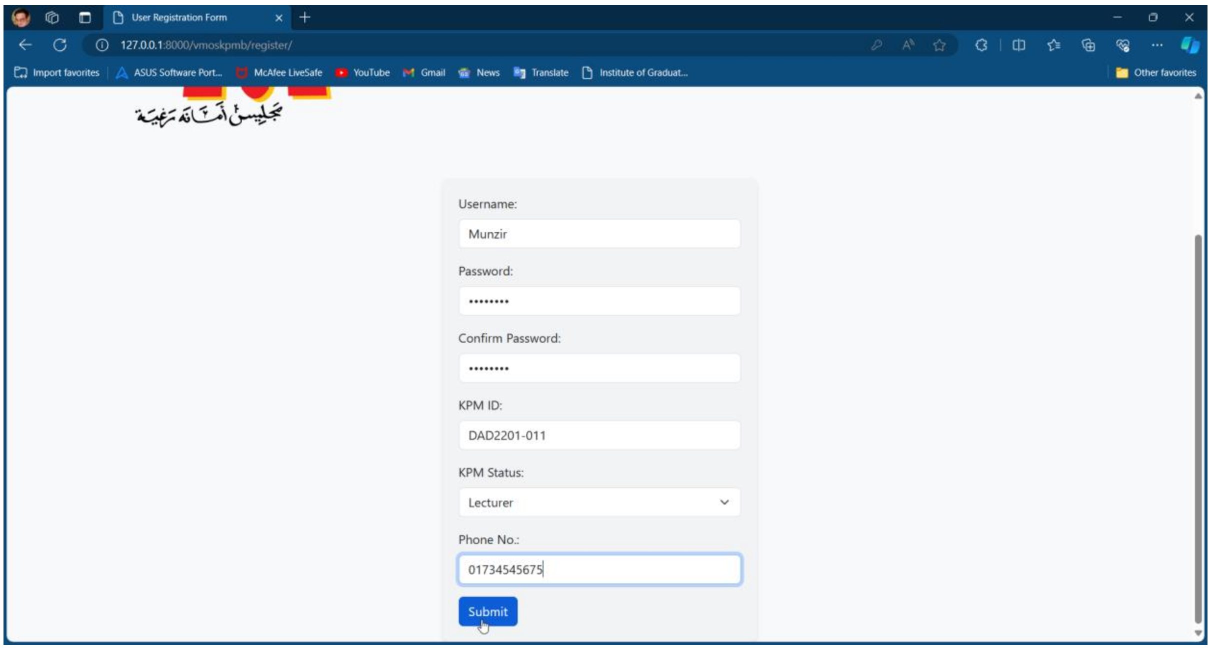
Student and Lecturer Status View:



Vehicle Management Online System

Welcome to KPM Beranang Vehicle Management Online System

Please login or register to access the system:



Username: Munzir

Password: ••••••••

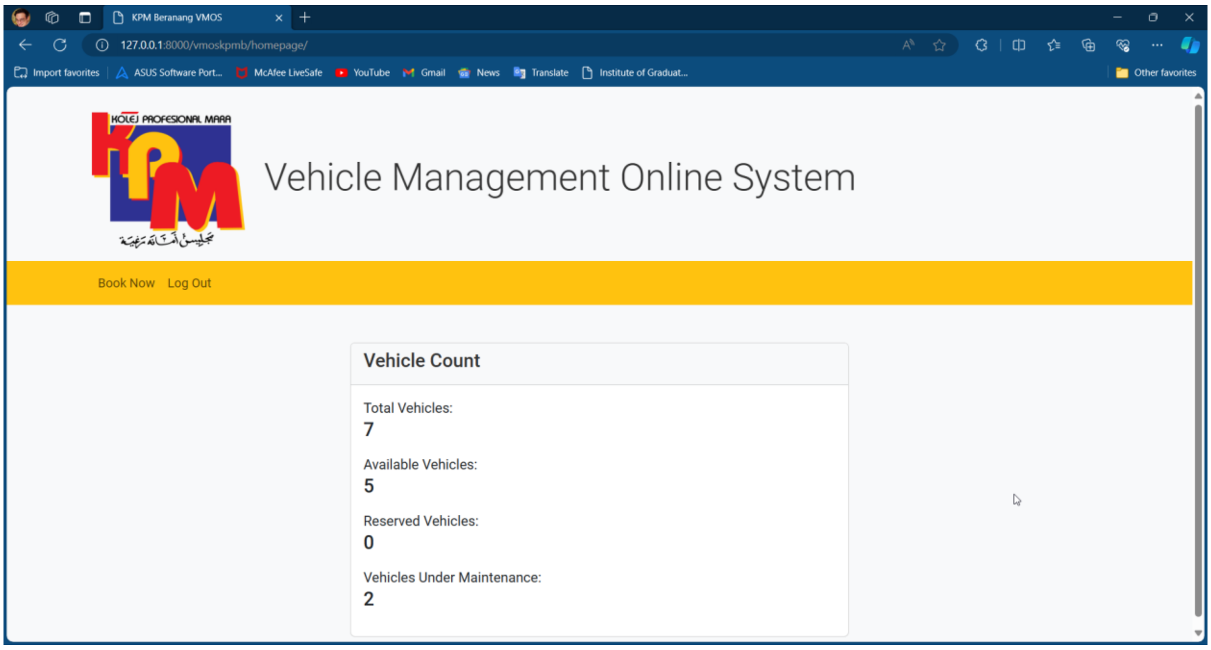
Confirm Password: ••••••••

KPM ID: DAD2201-011

KPM Status: Lecturer

Phone No.: 01734545675

Going thru the student or staffer’s view, users have to register themselves by entering the credentials appropriately in order to differentiate students from staffer’s.


Vehicle Management Online System

Vehicle Count

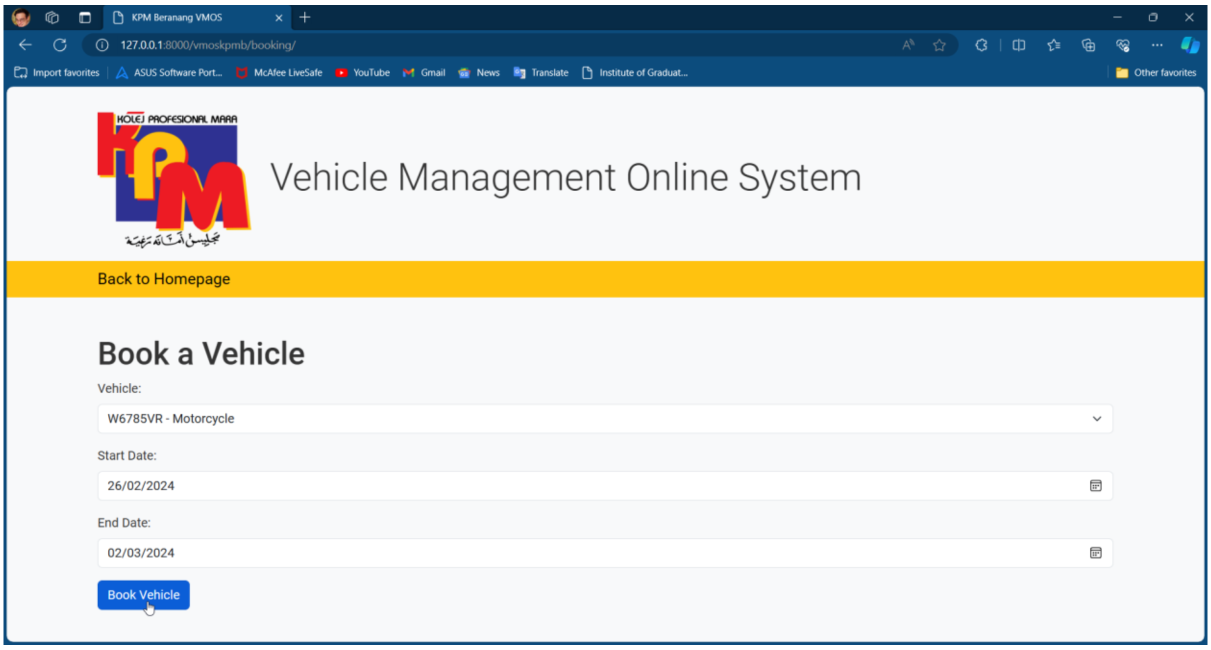
Total Vehicles:
7

Available Vehicles:
5

Reserved Vehicles:
0

Vehicles Under Maintenance:
2

Once again, we are greeted with the same homepage but with limited navbar.

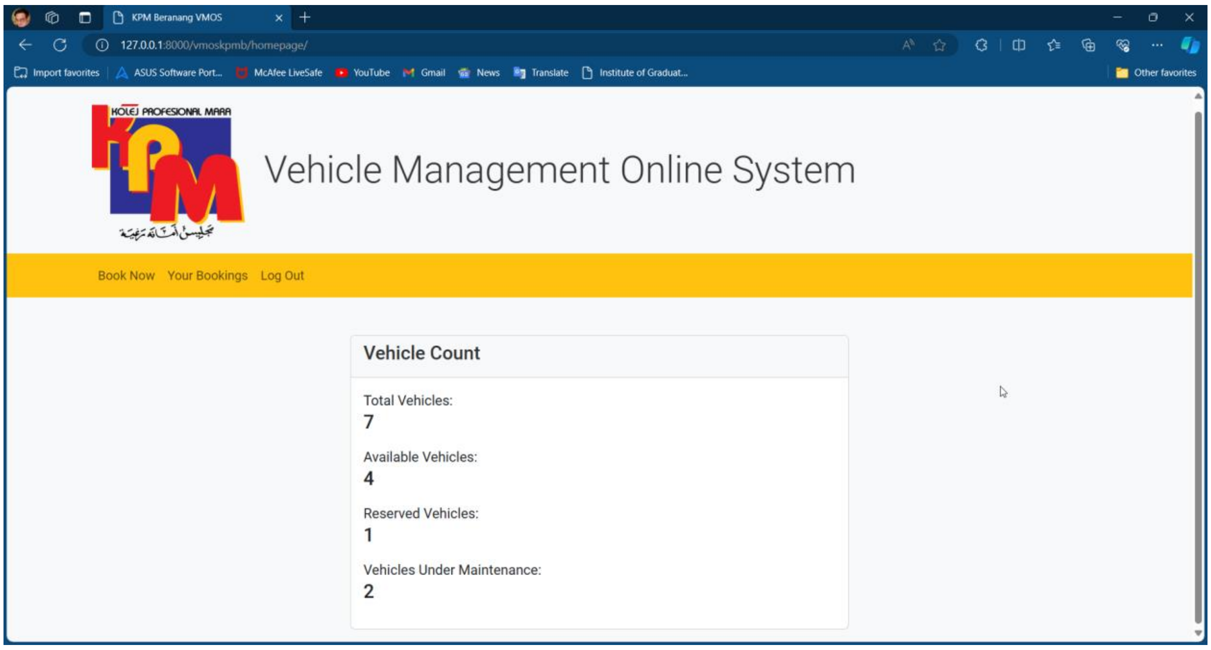

Vehicle Management Online System

Book a Vehicle

Vehicle:
W6785VR - Motorcycle

Start Date:
26/02/2024

End Date:
02/03/2024


Vehicle Management Online System

Book Now · Your Bookings · Log Out

Vehicle Count

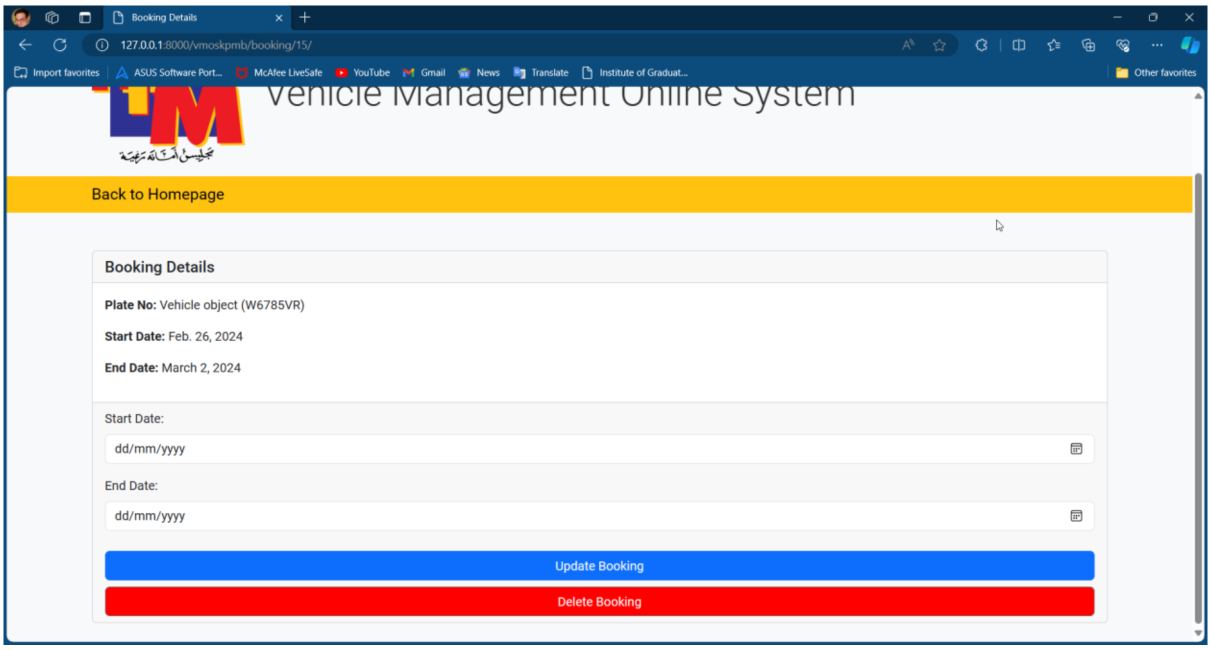
Total Vehicles:
7

Available Vehicles:
4

Reserved Vehicles:
1

Vehicles Under Maintenance:
2

After making a booking anew navbar clickable link appear that will redirect the user to his bookings.


Back to Homepage

Booking Details

Plate No: Vehicle object (W6785VR)

Start Date: Feb. 26, 2024

End Date: March 2, 2024

Start Date:
dd/mm/yyyy

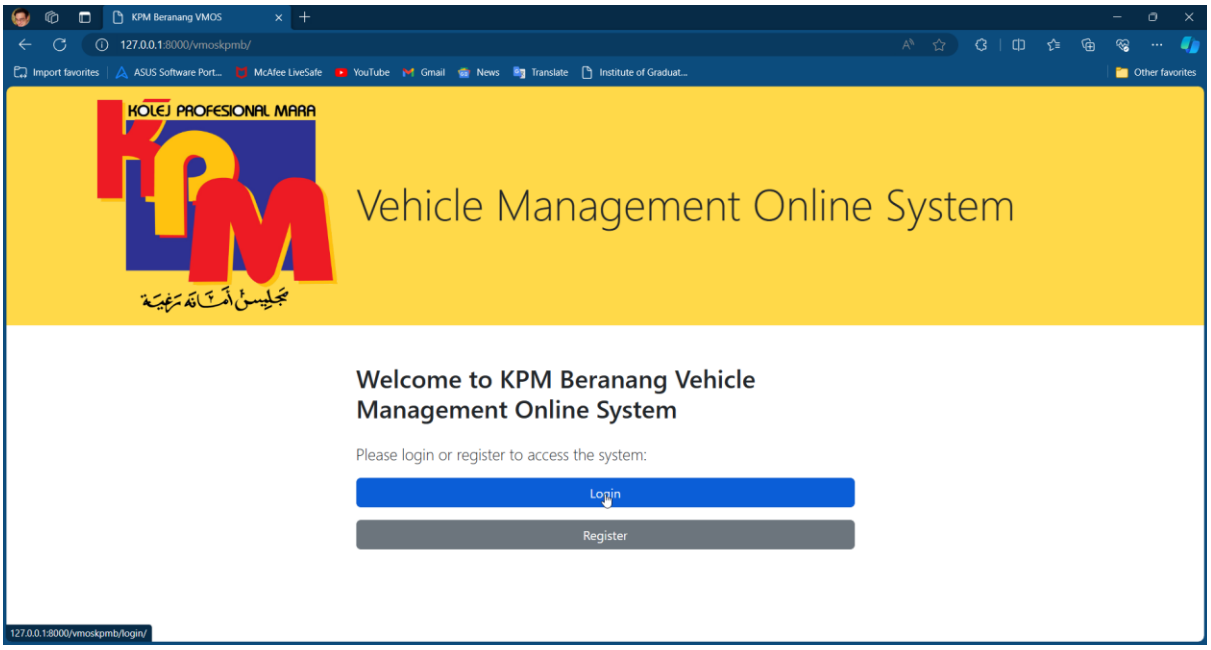
End Date:
dd/mm/yyyy

Update Booking

Delete Booking

User can change the reservation date but unable to switch vehicle. Once clicked Update Booking refresh the site. The new updated data shall be present.

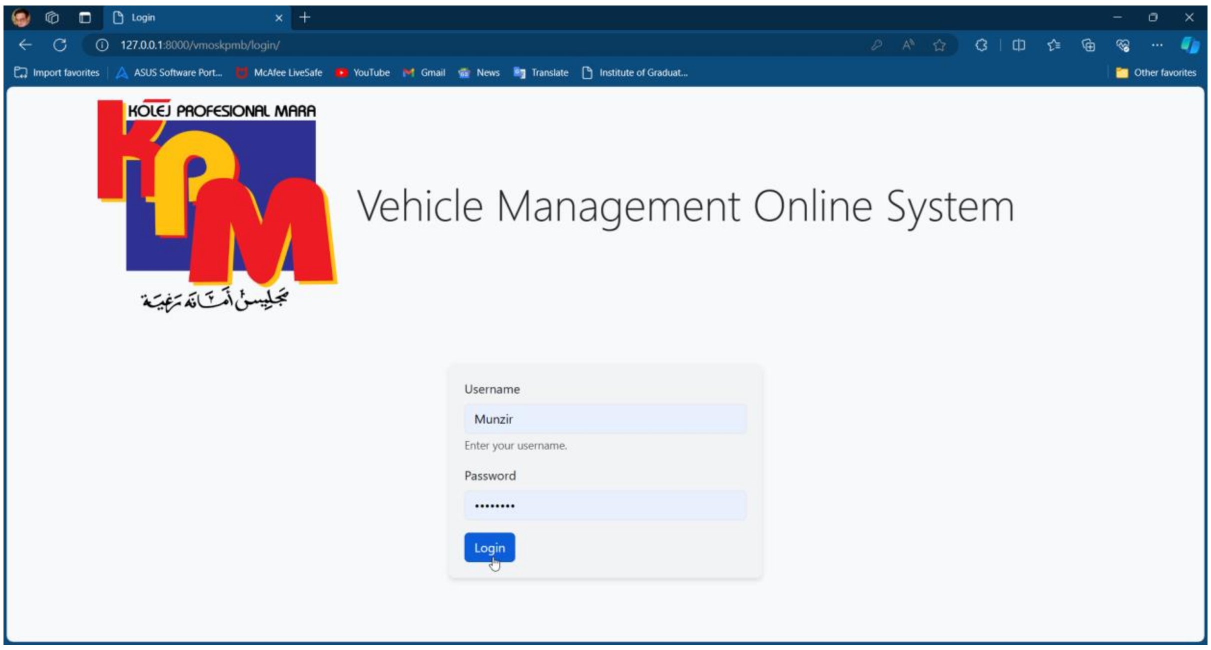
Additional:



Welcome to KPM Beranang Vehicle Management Online System

Please login or register to access the system:

Once logged out in the homepage the user should be redirected to the Welcome page and the user may login back in.


Login
127.0.0.1:8000/vmoskpmb/login/
KOLEJ PROFESIONAL MARA
Vehicle Management Online System
Login

Username: Munzir

Enter your username.

Password: ••••••••

views.py:

```python
from django.contrib.auth.forms import AuthenticationForm
from django.contrib.auth.decorators import login_required
from django.http import HttpResponse, HttpResponseRedirect
from django.shortcuts import render, redirect
from django.core.exceptions import ObjectDoesNotExist
from django.contrib import messages
from .forms import RegistrationForm
from django.db.models import Q
from django.urls import reverse
from django.contrib.auth import authenticate, login as authLogin, logout
from datetime import datetime, timedelta
from .models import Profile, Vehicle, Booking, Maintenance

@login_required
def homepage(request):
    booking = Booking.objects.first()
    authorize = None
    if request.user.profile.UserLevel == 'Administrator':
        authorize = Vehicle.objects.all()

    if request.user.profile.UserLevel == 'Student' and 'Lecturer':
        authorize = booking

    total_count = Vehicle.objects.count()
    available_count = Vehicle.objects.filter(Status='Available').count()
    reserved_count = Vehicle.objects.filter(Status='Reserved').count()
    maintenance_count = Vehicle.objects.filter(Status='Under
Maintenance').count()

    dict = {
        'booking': booking,
        'authorize':authorize,
        'total_count': total_count,
        'available_count': available_count,
        'reserved_count': reserved_count,
        'maintenance_count': maintenance_count
    }

    return render(request, 'homepage.html', dict)

def welcome(request):
    return render(request, 'welcome.html')

def register(request):
    if request.method == 'POST':
        form = RegistrationForm(request.POST)
        if form.is_valid():
            user = form.save()
            user_id = form.cleaned_data.get('UserId')
            user_level = form.cleaned_data.get('UserLevel')
            phone_no = form.cleaned_data.get('PhoneNo')
            Profile.objects.create(user=user, UserLevel=user_level,
PhoneNo=phone_no, UserId=user_id)
            authLogin(request, user)
            return redirect('homepage')
    else:
        form = RegistrationForm()
    return render(request, 'register.html', {'form': form})


def logingIn(request):
    if request.method == 'POST':
        form = AuthenticationForm(data=request.POST)
        if form.is_valid():
            username = form.cleaned_data.get('username')
            password = form.cleaned_data.get('password')
            user = authenticate(request, username=username, password=password)
            if user is not None:
                authLogin(request, user)
                return redirect('homepage')
    else:
        form = AuthenticationForm()
    return render(request, 'login.html', {'form': form})

def logingOut(request):
    logout(request)
    return redirect('welcome')

def book(request):
    if request.method == 'POST':
        if request.user.is_authenticated:
            user_level = request.user.profile.UserLevel

            # Filter vehicles based on user level
            if user_level == 'Student':
                available_vehicles =
Vehicle.objects.filter(Status='Available', VehicleType__in=['Car',
'Motorcycle'])
            else:
                available_vehicles =
Vehicle.objects.filter(Status='Available')

            plate_no = request.POST.get('vehicle')
            start_date = request.POST.get('start_date')
            end_date = request.POST.get('end_date')

            # Perform additional validation on start and end dates (e.g., date
range, availability)

            try:
                # Check if selected vehicle is available
                selected_vehicle = Vehicle.objects.get(PlateNo=plate_no)
                if selected_vehicle.Status != 'Available':
                    messages.error(request, 'Selected vehicle is not available
for booking.')
                    return redirect('book_vehicle')

                # Create a new booking entry
                booking = Booking(UserId=request.user.profile,
PlateNo=selected_vehicle, StartDate=start_date, EndDate=end_date)
                booking.save()

                # Update vehicle status to Reserved
                selected_vehicle.Status = 'Reserved'
                selected_vehicle.save()

                messages.success(request, 'Vehicle booked successfully.')
                return redirect('homepage')
            except Vehicle.DoesNotExist:
                messages.error(request, 'Invalid vehicle selected.')
                return redirect('booking_details')
        else:
            messages.error(request, 'User not authenticated.')
    else:
        # Retrieve available vehicles based on user level
        if request.user.is_authenticated:
            user_level = request.user.profile.UserLevel
            if user_level == 'Student':
                available_vehicles =
Vehicle.objects.filter(Status='Available', VehicleType__in=['Car',
'Motorcycle'])
            else:
                available_vehicles =
Vehicle.objects.filter(Status='Available')
        else:
            available_vehicles = None

    return render(request, 'booking.html', {'vehicles': available_vehicles})

def booking_details(request, booking_id):
    try:
        booking = Booking.objects.get(id=booking_id)
        return render(request, 'booking_details.html', {'booking': booking})
    except ObjectDoesNotExist:
        return render(request, 'no_booking.html')

def update_booking(request, booking_id):
    booking = Booking.objects.get(id = booking_id)

    if request.method == 'POST':
        start_date = request.POST.get('start_date')
        end_date = request.POST.get('end_date')

        # Update booking information based on form data
        booking.StartDate = start_date
        booking.EndDate = end_date
        booking.save()

        # Redirect to booking details page after update
        return HttpResponse("Booking updated successfully.")

    return render(request, 'booking_details.html', {'booking': booking})

def delete_booking(request, booking_id):
    booking = Booking.objects.get(id = booking_id)

    vehicle = booking.PlateNo
    booking.delete()

    vehicle.Status = 'Available'
    vehicle.save()

    return HttpResponse("Booking deleted successfully.")

def add_vehicle(request):
    if request.method == 'POST':
        plate_no = request.POST.get('plate_no')
        vehicle_type = request.POST.get('vehicle_type')
        status = request.POST.get('status')

        if plate_no and vehicle_type:
            vehicle = Vehicle(PlateNo=plate_no, VehicleType=vehicle_type,
Status=status)
            vehicle.save()
            messages.success(request, 'New Vehicle Added')
        else:
            messages.error(request, 'Fill out all required fields')

    return render(request, 'vehicle.html')
```

```python
def vehicle_status(request):
    plate_number = request.GET.get('plate_number')
    vehicles_data = Vehicle.objects.all()

    if plate_number:
        searched_vehicle =
Vehicle.objects.filter(PlateNo=plate_number).first()
    else:
        searched_vehicle = None

    return render(request, 'vehicle_status.html', {
        'vehicles_data': vehicles_data,
        'searched_vehicle': searched_vehicle,
        'searched_plate_number': plate_number,
    })

def updateVehicle(request, PlateNo):
    updateVehi = Vehicle.objects.get(PlateNo = PlateNo)
    return render(request, "vehicle_update.html", {'updateVehi': updateVehi})

def update_vehicle_data(request, PlateNo):
    data = Vehicle.objects.get(PlateNo = PlateNo)
    plate_no = request.POST.get('plate_no')
    vehicle_type = request.POST.get('vehicle_type')
    status = request.POST.get('status')
    data.PlateNo = plate_no
    data.VehicleType = vehicle_type
    data.Status = status
    data.save()

    return HttpResponseRedirect(reverse("vehicle_status"))

def willDelete(request):
    will_be_delete = Vehicle.objects.all()
    return render(request, 'vehicle_delete.html', {'will_be_delete':
will_be_delete})

def deleteVehicle(request, PlateNo):
    deleteVehi = Vehicle.objects.get(PlateNo = PlateNo)
    deleteVehi.delete()

    return HttpResponseRedirect(reverse("vehicle_status"))

def maintenance(request):
    vehicles = Vehicle.objects.all()

    if request.method == 'POST':
        plate_no = request.POST.get('plate_no')
        description = request.POST.get('description')
        last_service = request.POST.get('last_service')
        next_service = request.POST.get('next_service')

        vehicle = Vehicle.objects.get(PlateNo=plate_no)
        maintenance = Maintenance.objects.create(PlateNo=vehicle,
Description=description, LastMaintenance=last_service,
NextMaintenance=next_service)

        if datetime.strptime(next_service, '%Y-%m-%d').date() <=
datetime.now().date():
            vehicle.Status = 'Under Maintenance'
            vehicle.save()

        return redirect('maintenance_status')  # Redirect to maintenance
status page

    return render(request, 'maintenance.html', {'vehicles': vehicles})

def maintenance_status(request):
    maintenance_data = Maintenance.objects.all()
    return render(request, 'maintenance_status.html', {'maintenance_data':
maintenance_data})

def updateMaintenance(request, maintenance_id):
    updateMain = Maintenance.objects.get(id=maintenance_id)
    return render(request, "maintenance_update.html", {'updateMain':
updateMain})

def update_maintenance_data(request, maintenance_id):
    data = Maintenance.objects.get(id=maintenance_id)
    description = request.POST.get('description')
    last_service = request.POST.get('last_service')
    next_service = request.POST.get('next_service')
    data.Description = description
    data.LastMaintenance = last_service
    data.NextMaintenance = next_service
    data.save()

    return HttpResponseRedirect(reverse("maintenance_status"))

def viewDelete(request, maintenance_id):
    to_be_delete = Maintenance.objects.get(id=maintenance_id)
    return render(request, "maintenance_delete.html", {'to_be_delete':
to_be_delete})

def deleteMaintenance(request, maintenance_id):
    deleteMaintenance = Maintenance.objects.get(id=maintenance_id)
    deleteMaintenance.delete()

    return HttpResponseRedirect(reverse("maintenance_status"))

def reset_maintenance(request, maintenance_id):
    maintenance = Maintenance.objects.get(id=maintenance_id)
    maintenance.LastMaintenance = datetime.now().date()
    maintenance.NextMaintenance = datetime.now().date() + timedelta(days=30)
    maintenance.save()
    vehicle = maintenance.vehicle
    vehicle.Status = 'Available'
    vehicle.save()
    return redirect('maintenance_status')
```