

# Source Code of App: myapp

## admin.py

```
from django.contrib import admin
from .models import Car, CarType

admin.site.register(Car)
admin.site.register(CarType)
```

## apps.py

```
from django.apps import AppConfig

class MyappConfig(AppConfig):
    default_auto_field = "django.db.models.BigAutoField"
    name = "myapp"
```

## forms.py

```
from django import forms
from .models import CustomUser
from .models import Schedule, Car
from .models import CarReview
from .models import Review

class CarForm(forms.ModelForm):
    class Meta:
        model = Car
        fields = [
            'name', 'description', 'horsepower', 'image', 'is_available',
            'morning_price', 'morning_custom_price', 'afternoon_price', 'afternoon_custom_price', 'price_per_rai', 'time_per_rai'
        ]

        widgets = {
            'name': forms.TextInput(attrs={'class': 'form-control'}),
            'description': forms.Textarea(attrs={'class': 'form-control', 'rows': 3}),
            'horsepower': forms.NumberInput(attrs={'class': 'form-control'}),
            'image': forms.FileInput(attrs={'class': 'form-control'}),
            'is_available': forms.CheckboxInput(attrs={'class': 'form-checkcheckbox'}),
            'morning_price': forms.NumberInput(attrs={'class': 'form-control'}),
            'morning_custom_price': forms.NumberInput(attrs={'class': 'form-control'}),
            'afternoon_price': forms.NumberInput(attrs={'class': 'form-control'}),
            'afternoon_custom_price': forms.NumberInput(attrs={'class': 'form-control'}),
            'price_per_rai': forms.NumberInput(attrs={'class': 'form-control'}),
            'time_per_rai': forms.NumberInput(attrs={'class': 'form-control'}),
        }

class UserUpdateForm(forms.ModelForm):
    class Meta:
        model = CustomUser
        fields = ['first_name', 'last_name', 'nickname', 'age', 'phone_number', 'email', 'address', 'profile_picture']

        widgets = {
            'profile_picture': forms.FileInput(attrs={'class': 'form-control'}),
        }

class ScheduleForm(forms.ModelForm):
    date = forms.DateField(input_formats=['%Y-%m-%d'])

    class Meta:
        model = Schedule
        fields = ['car', 'date', 'time']

class CarReviewForm(forms.ModelForm):
    class Meta:
        model = CarReview
        fields = ['rating', 'review_text']

class ReviewForm(forms.ModelForm):
    class Meta:
        model = CarReview
        fields = ['review_text', 'rating']

class BookingForm(forms.ModelForm):
    class Meta:
        model = Schedule
        fields = ['date', 'start time', 'end time', 'time']

        widgets = {
            'date': forms.DateInput(attrs={'type': 'date', 'class': 'form-control'}),
            'start time': forms.Select(choices=[
                ('08:00', '08:00'), ('09:00', '09:00'), ('10:00', '10:00'),
                ('11:00', '11:00'), ('12:00', '12:00'), ('13:00', '13:00'),
                ('14:00', '14:00'), ('15:00', '15:00'), ('16:00', '16:00')
            ], attrs={'class': 'form-control'}),
            'end time': forms.Select(choices=[
                ('09:00', '09:00'), ('10:00', '10:00'), ('11:00', '11:00'),
                ('12:00', '12:00'), ('13:00', '13:00'), ('14:00', '14:00'),
                ('15:00', '15:00'), ('16:00', '16:00'), ('17:00', '17:00')
            ], attrs={'class': 'form-control'}),
            'time': forms.Select(choices=[
                ('morning', 'ช่วงเช้า'), ('afternoon', 'ช่วงบ่าย')
            ], attrs={'class': 'form-control'})
        }
```

## models.py

```
from django.db import models
from django import forms
from django.contrib.auth.models import AbstractUser, Group, Permission
from django.conf import settings
from django.contrib.auth.models import User

class CarType(models.Model):
    name = models.CharField(max_length=100)

    def __str__(self):
        return self.name

class Car(models.Model):
    STATUS_CHOICES = [
        ('Pending', 'Pending'),
        ('Approved', 'Approved'),
    ]

    PRICING_TYPE_CHOICES = [
        ('flat', 'ราคาแบบ'),
        ('per_rai', 'ราคาต่อไร่')
    ]

    name = models.CharField(max_length=100)
    description = models.TextField(blank=True, null=True)
    horsepower = models.IntegerField(verbose_name="แรงม้า", default=0)
```

```

        car_type = models.ForeignKey(CarType, related_name='cars', on_delete=models.CASCADE)
        image = models.ImageField(upload_to='cars/', blank=True, null=True, default='default_car.jpg')
        status = models.CharField(max_length=20, choices=STATUS_CHOICES)
        owner = models.ForeignKey(settings.AUTH_USER_MODEL, on_delete=models.CASCADE, null=True, blank=True, related_name='cars')
        is_available = models.BooleanField(default=True)
        created_at = models.DateTimeField(auto_now_add=True, null=True)

        morning_price = models.DecimalField(max_digits=10, decimal_places=2, null=True, blank=True, verbose_name="ราคาเหมา ช่วงเช้า")
        morning_custom_price = models.DecimalField(max_digits=10, decimal_places=2, null=True, blank=True, verbose_name="ราคาที่กำหนดเอง ช่วงเช้า")
        afternoon_price = models.DecimalField(max_digits=10, decimal_places=2, null=True, blank=True, verbose_name="ราคาเหมา ช่วงบ่าย")
        afternoon_custom_price = models.DecimalField(max_digits=10, decimal_places=2, null=True, blank=True, verbose_name="ราคาที่กำหนดเอง ช่วงบ่าย")

        price_per_rai = models.DecimalField(max_digits=10, decimal_places=2, null=True, blank=True, verbose_name="ราคาต่อไร่")
        time_per_rai = models.IntegerField(null=True, blank=True, verbose_name="เวลาทำงานต่อไร่ (นาทีก)")

        def __str__(self):
            return self.name

        class Schedule(models.Model):
            TIME_CHOICES = [
                ('morning', 'ช่วงเช้า'),
                ('afternoon', 'ช่วงบ่าย')
            ]

            car = models.ForeignKey(Car, on_delete=models.CASCADE, related_name='schedules')
            date = models.DateField() # วันที่จอง
            start_time = models.TimeField(null=True, blank=True) # เวลาเริ่มต้น
            end_time = models.TimeField(null=True, blank=True) # เวลาสิ้นสุด
            time = models.CharField(max_length=20, choices=TIME_CHOICES, null=True, blank=True) # เลือกช่วงเวลา
            is_booked = models.BooleanField(default=False) # ตรวจสอบว่าถูกจองแล้วหรือยัง
            booked_by = models.ForeignKey(
                settings.AUTH_USER_MODEL, # เชื่อมกับผู้ใช้ที่ทำการจอง
                on_delete=models.CASCADE,
                related_name='bookings',
                null=True, blank=True # อนุญาตให้ว่างได้สำหรับการจองที่ยังไม่มีเจ้าของ
            )

            def __str__(self):
                if self.time:
                    return f"{self.car.name} - {self.date} ({self.get_time_display()})"
                return f"{self.car.name} - {self.date} ({self.start_time} to {self.end_time})"

            class CarForm(forms.ModelForm):
                class Meta:
                    model = Car
                    fields = ['name', 'description', 'horsepower', 'image']

            class CustomUser(AbstractUser):
                nickname = models.CharField(max_length=50, null=True, blank=True)
                age = models.PositiveIntegerField(null=True, blank=True)
                phone_number = models.CharField(max_length=15, null=True, blank=True)
                address = models.TextField(null=True, blank=True)
                profile_picture = models.ImageField(upload_to='profile_pics/', blank=True, null=True)

                # Add related_name to resolve clashes
                groups = models.ManyToManyField(
                    Group,
                    related_name='customuser_groups',
                    blank=True,
                    help_text='The groups this user belongs to.',
                    verbose_name='groups'
                )
                user_permissions = models.ManyToManyField(
                    Permission,
                    related_name='customuser_permissions',
                    blank=True,
                    help_text='Specific permissions for this user.',
                    verbose_name='user permissions'
                )

                def __str__(self):
                    return self.username

            class Notification(models.Model):
                user = models.ForeignKey(settings.AUTH_USER_MODEL, on_delete=models.CASCADE) # ผู้ที่ได้รับการแจ้งเตือน
                borrower = models.ForeignKey(settings.AUTH_USER_MODEL, on_delete=models.CASCADE, related_name='borrower_notifications', null=True, blank=True)
                message = models.TextField() # ข้อความแจ้งเตือน
                timestamp = models.DateTimeField(auto_now_add=True) # เวลาที่แจ้งเตือนถูกสร้าง
                schedule = models.ForeignKey(Schedule, null=True, blank=True, on_delete=models.SET_NULL) # เชื่อมโยงกับการจอง
                is_confirmed = models.BooleanField(default=False) # สถานะการยืนยันการจอง
                is_approved = models.BooleanField(default=False) # สถานะการอนุมัติจอง

                def __str__(self):
                    return f"Notification for {self.user.username} at {self.timestamp}"

            class CarReview(models.Model):
                car = models.ForeignKey('Car', on_delete=models.CASCADE, related_name='reviews')
                user = models.ForeignKey(settings.AUTH_USER_MODEL, on_delete=models.CASCADE) # แก่ไขที่ตี
                rating = models.IntegerField(choices=[(i, i) for i in range(1, 6)]) # Rating 1-5
                review_text = models.TextField()
                created_at = models.DateTimeField(auto_now_add=True)

                def __str__(self):
                    return f"Review for {self.car.name} by {self.user.username}"

            class Review(models.Model):
                car = models.ForeignKey(Car, on_delete=models.CASCADE)
                user = models.ForeignKey(settings.AUTH_USER_MODEL, on_delete=models.CASCADE) # แก่ไขที่ตี
                comment = models.TextField()

                def __str__(self):
                    return f"Review for {self.car.name} by {self.user.username}"

```

## tests.py

```

from django.test import TestCase

# Create your tests here.

```

## urls.py

```

from django.urls import path
from . import views
from .views import notification_list
from .views import user_management, delete_user
from .views import delete_car
from .views import user_reservations
from .views import cancel_reservation

urlpatterns = [
    path('export-code/', views.app_to_pdf, name='export_code'),

```

```

        path('', views.welcome_view, name='welcome'), # หน้าแรก
        path('login/', views.login_view, name='login'), # หน้าเข้าสู่ระบบ
        path('register/', views.register_view, name='register'), # หน้าไปยังหน้าลงทะเบียน
        path('logout/', views.logout_view, name='logout'), # ล็อกเอาท์
        path('cars/types/', views.car_type_list_view, name='car_type_list'), # แสดงประเภทของรถ
        path('cars/type/', views.car_list_by_type_view, name='car_list_by_type'), # แสดงรถในแต่ละประเภท
        path('cars/', views.car_detail_view, name='car_detail'), # แสดงรายละเอียดของรถ
        path('edicar/', views.edicar, name='edicar'), # แก้ไขข้อมูลรถ
        path('delcar/', views.delcar, name='delcar'), # ลบรถ
        path('delete_car/', delete_car, name='delete_car'), # ลบรถ
        path('car/schedule/', views.car_schedule, name='car_schedule'), # ตารางงานของรถ
        path('approval_list/', views.car_approval_list, name='car_approval_list'), # แสดงรายการรถที่รออนุมัติ
        path('approve_car/', views.approve_car, name='approve_car'), # อนุมัติรถ
        path('addcar/', views.add_car, name='addcar'), # เพิ่มรถใหม่
        path('pending-cars/', views.pending_car_list, name='pending_car_list'), # รถที่รอการอนุมัติ
        path('delete-car/', views.delete_car, name='delete_car'), # ลบรถ
        path('profile/', views.profile_view, name='profile'), # ดูโปรไฟล์
        path('edit-profile/', views.edit_profile_view, name='edit_profile'), # แก้ไขโปรไฟล์
        path('cars/toggle_status/', views.toggle_car_status, name='toggle_car_status'), # เปลี่ยนสถานะของรถ
        path('my-cars/', views.my_cars, name='my_cars'), # เส้นทางสำหรับแสดงข้อมูลรถของผู้ใช้
        path('confirm_selection/', views.confirm_selection, name='confirm_selection'), # ยืนยันการเลือกเวลา
        path('edit_car/', views.edicar, name='edit_car'), # แก้ไขรถ
        path('create_booking/', views.create_booking, name='create_booking'), # สร้างการจอง
        path('confirm_reservation/', views.confirm_reservation, name='confirm_reservation'), # ยืนยันการจอง
        path('notification_list/', views.notification_list, name='notification_list'), # แสดงรายการรถที่จอง
        path('car_list/', views.car_list_view, name='car_list'), # แสดงรายการรถทั้งหมด
        path('cars/', views.car_list_view, name='car_list'), # เส้นทางแสดงรถในแต่ละประเภท

        path('cars/reviews/', views.car_review_view, name='car_review'),

        path('notification_list/', notification_list, name='notification_list'),

        path('users/', user_management, name='user_management'),
        path('users/delete/', delete_user, name='delete_user'),

        path('cancel_reservation/', views.cancel_reservation, name='cancel_reservation'), # เส้นทางสำหรับยกเลิกการจอง
        path('my_reservations/', views.user_reservations, name='user_reservations'), # เส้นทางสำหรับดูการจองของผู้ใช้

        path('cars/book_time/', views.book_time, name='book_time'),
        path('cars/create_booking/', views.create_booking, name='create_booking'),

        path('confirm_reservation/', views.confirm_reservation, name='confirm_reservation'),

```

```

    ]

```

## views.py

```

from django.shortcuts import render, redirect, get_object_or_404
from django.contrib.auth import authenticate, login, logout
from django.contrib import messages
from .models import *
from django.http import HttpResponseRedirect
from django.urls import reverse
from .forms import CarForm, UserUpdateForm
from django.contrib.auth.decorators import user_passes_test, login_required
from django.http import JsonResponse
from .models import Car, Notification, Schedule
from datetime import datetime
from django.contrib.admin.views.decorators import staff_member_required
from myapp.models import CustomUser
from django.db.models import Q
from .models import Car, Review
from .forms import ReviewForm

import os
import pdfkit
from django.http import HttpResponseRedirect

# กำหนด path ของ wkhtmltopdf (Windows ใช้ path นี้, Linux/Mac อาจไม่ต้อง)
PDFKIT_CONFIG = pdfkit.configuration(wkhtmltopdf=r"C:\Program Files\wkhtmltopdf\bin\wkhtmltopdf.exe")

def app_to_pdf(request, app_name):
    app_path = os.path.join(settings.BASE_DIR, app_name)
    pdf_filename = f"{app_name}.pdf"

    if not os.path.exists(app_path):
        return HttpResponseRedirect(status=404)

    html_content = f"

    "

    for root, dirs, files in os.walk(app_path):
        for file in files:
            if file.endswith(("py", "html", "css", "js")): # แปลงเฉพาะไฟล์ที่ต้องการ
                file_path = os.path.join(root, file)
                with open(file_path, "r", encoding="utf-8") as f: # ๒ อ่านไฟล์เป็น UTF-8
                    html_content += f"

                {file}

                {f.read()}

                "

    # ๓ ใช้ pdfkit พร้อมกำหนดให้ใช้ encoding UTF-8
    options = {'encoding': 'UTF-8'}
    pdfkit.from_string(html_content, pdf_filename, configuration=PDFKIT_CONFIG, options=options)

    with open(pdf_filename, "rb") as pdf:
        response = HttpResponseRedirect(pdf.read(), content_type="application/pdf")
        response["Content-Disposition"] = f'attachment; filename="{pdf_filename}"'
        return response

@staff_member_required # ให้เฉพาะแอดมินเข้าถึงได้
def user_management(request):
    users = User.objects.all()
    return render(request, 'user_management.html', {'users': users})

@staff_member_required
def delete_user(request, user_id):
    user = get_object_or_404(User, id=user_id)

    if user.is_staff:
        messages.error(request, "ไม่สามารถลบผู้ดูแลระบบได้")
    else:
        user.delete()
    messages.success(request, "ลบผู้ใช้งานเรียบร้อยแล้ว")

```

```

return redirect('user_management')

# ฟังก์ชันสำหรับหน้าแรก
def welcome_view(request):
    """
    แสดงหน้า Welcome (หน้าแรกของระบบ)
    """
    return render(request, 'welcome.html')

# ฟังก์ชันสำหรับหน้าล็อกอิน
def login_view(request):
    if request.method == 'POST':
        username = request.POST['username']
        password = request.POST['password']
        user = authenticate(request, username=username, password=password)
        if user is not None:
            login(request, user)
            return redirect('car_type_list')
        else:
            messages.error(request, 'Invalid username or password')

    return render(request, 'login.html')

# ฟังก์ชันสำหรับหน้าสมัคร
def register_view(request):
    """
    สมัครสมาชิกใหม่:
    - ตรวจสอบความถูกต้องของข้อมูล
    - สร้างผู้ใช้ใหม่
    - ส่งข้อความแจ้งเตือนความสำเร็จหรือข้อผิดพลาด
    """
    if request.method == 'POST':
        username = request.POST['username']
        email = request.POST['email']
        password = request.POST['password']
        confirm_password = request.POST['confirm_password']
        if password == confirm_password:
            if CustomUser.objects.filter(username=username).exists():
                messages.error(request, 'Username already exists')
            elif CustomUser.objects.filter(email=email).exists():
                messages.error(request, 'Email already exists')
            else:
                user = CustomUser.objects.create_user(username=username, email=email, password=password)
                user.save()
                messages.success(request, 'Account created successfully')
                return redirect('login')
            else:
                messages.error(request, 'Passwords do not match')

    return render(request, 'register.html')

# ฟังก์ชันสำหรับการล็อกเอาท์
def logout_view(request):
    """
    ออกจากระบบ:
    - เคลียร์ session ของผู้ใช้
    """
    logout(request)
    return redirect('login')

# ฟังก์ชันแสดงรายการประเภทของรถ
def car_type_list_view(request):
    """
    แสดงรายการประเภทของรถ
    """
    car_types = CarType.objects.all()
    return render(request, 'car_type_list.html', {'car_types': car_types})

# ฟังก์ชันแสดงรายการรถ
def car_list_view(request):
    cars = Car.objects.all()
    return render(request, 'car_list.html', {'cars': cars})

# ฟังก์ชันแสดงรถในแต่ละประเภท
def car_list_by_type(request, type_id):
    """
    แสดงรายการรถที่อยู่ในประเภทที่กำหนด
    """
    cars = Car.objects.filter(
        car_type_id=type_id,
        status='Approved'
    ).exclude(image_isnull=True).exclude(image='') # กรองรถที่ไม่มีไฟล์ภาพ
    return render(request, 'car_list.html', {'cars': cars, 'car_type': get_object_or_404(CarType, id=type_id)})

# ฟังก์ชันแสดงรายละเอียดของรถแต่ละคัน
def car_detail_view(request, car_id):
    """
    แสดงรายละเอียดของรถแต่ละคัน
    """
    car = get_object_or_404(Car, pk=car_id)
    return render(request, 'car_detail.html', {'car': car})

# ฟังก์ชันเพิ่มรถ
@login_required
def add_car(request, type_id):
    car_type = get_object_or_404(CarType, id=type_id) # ดึงข้อมูลประเภทของรถ
    if request.method == 'POST':
        form = CarForm(request.POST, request.FILES)
        if form.is_valid():
            car = form.save(commit=False)
            car.car_type = car_type # กำหนด car_type
            car.owner = request.user # กำหนด owner เป็นผู้ใช้ที่ล็อกอินอยู่
            car.status = 'Pending' # ตั้งสถานะเป็น 'Pending' สำหรับการอนุมัติ
            car.save()
            return redirect('car_list_by_type', type_id=car_type.id) # ใจดีเรกไปยังหน้ารายการรถของประเภทนั้น
        else:
            form = CarForm()
    return render(request, 'addcar.html', {'form': form, 'car_type': car_type})

# ฟังก์ชันแก้ไขรถ
from django.shortcuts import redirect

@login_required
def edicar(request, id, type_id):
    car = get_object_or_404(Car, id=id)

    if request.user != car.owner and not request.user.is_staff:
        return HttpResponseForbidden("คุณไม่มีสิทธิ์ในการแก้ไขรถคันนี้")

    if request.method == 'POST':
        form = CarForm(request.POST, instance=car)
        if form.is_valid():
            form.save()
            messages.success(request, "แก้ไขข้อมูลรถสำเร็จ!")
            return redirect('car_list_by_type', type_id=type_id)
        else:

```

```

        form = CarForm(instance=car)

    return render(request, 'edit_car.html', {'form': form, 'car': car})

    # ฟังก์ชันลบรถ
    @login_required
    def delcar(request, car_id, type_id):
        car = get_object_or_404(Car, id=car_id)

    # ตรวจสอบว่าเจ้าของรถหรือแอดมินหรือไม่
    if request.user != car.owner and not request.user.is_staff:
        return HttpResponseForbidden("คุณไม่มีสิทธิ์ในการลบรถคันนี้")

    # ถ้าเจ้าของรถหรือแอดมิน ยืนยันการลบ
        car.delete()
    return redirect('car_list') # เปลี่ยนไปหน้าแสดงรายการรถ

    # ฟังก์ชันแสดงตารางงานของรถ
    def car_schedule(request, car_id):
        # กรองรายการที่ได้รับการยืนยันแล้ว
        schedules = Schedule.objects.filter(car_id=car_id, is_booked=True)
    return render(request, 'car_schedule.html', {'schedules': schedules})

    def confirm_selection(request):
        selected_date = request.GET.get('date')
        selected_time = request.GET.get('time')
        car_id = request.GET.get('car_id')

        if selected_date and selected_time and car_id:
            car = Car.objects.get(id=car_id)
            return render(request, 'confirm_selection.html', {
                'selected_date': selected_date,
                'selected_time': selected_time,
                'car': car,
            })
        else:
            return render(request, 'error.html', {'message': 'ข้อมูลไม่ครบถ้วน'})

    # แสดงรายการรถที่รอการอนุมัติ
    def car_approval_list(request):
        # ดึงเฉพาะรถที่รอการอนุมัติ
        cars_pending = Car.objects.filter(status='Pending')

    # ส่งข้อมูล car_type ไปพร้อมกับ cars
    return render(request, 'car_approval_list.html', {'cars': cars_pending})

    # ฟังก์ชันสำหรับอนุมัติรถ
    def approve_car(request, car_id):
        if not request.user.is_staff:
            return redirect('car_type_list')

        car = get_object_or_404(Car, id=car_id)
        car.status = 'Approved' # อัปเดตสถานะเป็น Approved
        car.save() # บันทึกการเปลี่ยนแปลง
    return redirect('car_list_by_type', type_id=car.car_type.id) # กลับไปหน้ารายการรถ

    # แสดงรายการรถที่อนุมัติแล้ว
    def car_list_by_type_view(request, type_id):
        # ดึงข้อมูล CarType โดย type_id
        car_type = get_object_or_404(CarType, id=type_id)

    # ไข car_type.cars.all() เพื่อดึงรถทั้งหมดที่เกี่ยวข้องกับ car_type นี้
    cars = car_type.cars.filter(status='Approved').exclude(image__isnull=True).exclude(image='')

    return render(request, 'car_list.html', {'cars': cars, 'car_type': car_type})

    def pending_car_list(request):
        cars = Car.objects.filter(status='Pending').exclude(image__isnull=True).exclude(image='')
    return render(request, 'pending_cars.html', {'cars': cars})

    # เช็คว่า user เป็น staff (admin) หรือไม่
        def is_admin(user):
            return user.is_staff

    @user_passes_test(is_admin)
    def pending_car_list(request):
        pending_cars = Car.objects.filter(status='Pending') # รถที่ยังไม่ได้อนุมัติ
    return render(request, 'pending_car_list.html', {'cars': pending_cars})

    @login_required
    def delete_car(request, car_id):
        car = get_object_or_404(Car, id=car_id)

    # ตรวจสอบว่าเป็นเจ้าของรถหรือเป็นแอดมิน
    if request.user != car.owner and not request.user.is_staff:
        return JsonResponse({'success': False, 'error': "คุณไม่มีสิทธิ์ลบรถคันนี้"}, status=403)

        car_type_id = car.car_type.id # ดึงประเภทของรถไว้ก่อนลบ
        car.delete()

    return JsonResponse({'success': True, 'redirect_url': f'/cars/type/{car_type_id}/'})

    @login_required
    def profile_view(request):
        user = CustomUser.objects.filter(id=request.user.id)
    return render(request, 'profile.html', {'user': request.user})

    @login_required
    def edit_profile_view(request):
        # ตรวจสอบว่าเป็นการแก้ไขโปรไฟล์ของผู้อื่นหรือไม่
        if request.method == 'POST':
            form = UserUpdateForm(request.POST, request.FILES, instance=request.user)
            if form.is_valid():
                form.save()
                messages.success(request, 'โปรไฟล์ถูกอัปเดตเรียบร้อยแล้ว!')
                return redirect('profile')
            else:
                form = UserUpdateForm(instance=request.user) # โหลดข้อมูลของผู้ใช้ที่ล็อกอิน
            return render(request, 'edit_profile.html', {'form': form})

    def car_detail_view(request, car_id):
        # ดึงข้อมูลรถตาม car_id
        car = get_object_or_404(Car, id=car_id)

        # ดึงข้อมูลเจ้าของรถ
        owner = car.owner # เจ้าของรถ (ผู้ใช้)

        # ส่งข้อมูลไปเทมเพลต
        context = {
            'car': car,
            'owner': owner, # ส่งข้อมูลเจ้าของรถไปเทมเพลต
        }

```

```

    }

    return render(request, 'car_detail.html', context)

    # ฟังก์ชันสำหรับเปลี่ยนสถานะของรถ
    def toggle_car_status(request, car_id):
        # ดึงข้อมูลรถจากฐานข้อมูลโดยใช้ car_id
        car = get_object_or_404(Car, id=car_id)

        # ตรวจสอบว่าเจ้าของรถเป็นผู้ที่ล็อกอินอยู่หรือไม่
        if car.owner == request.user:
            # เปลี่ยนสถานะของรถ (พร้อมใช้งาน .. ไม่พร้อมใช้งาน)
            car.is_available = not car.is_available
            car.save()

    # หลังจากเปลี่ยนสถานะแล้ว ให้ย้อนกลับไปยังหน้ารายละเอียดของรถ
    return redirect('car_detail', car_id=car.id)

    def my_cars(request):
        # ดึงข้อมูลรถที่เจ้าของเป็นผู้ใช้ปัจจุบัน
        cars = Car.objects.filter(owner=request.user)

        # ส่งข้อมูลรถไปยัง template
    return render(request, 'my_cars.html', {'cars': cars})

    # ฟังก์ชันสำหรับการยืนยันการจอง
    @login_required
    def confirm_reservation(request, schedule_id):
        schedule = get_object_or_404(Schedule, id=schedule_id)
        if request.method == 'POST':
            action = request.POST.get('action')
            if action == 'approve':
                schedule.is_booked = True
                schedule.save()
                Notification.objects.create(
                    user=schedule.booked_by,
                    message=f"การจองรถ {schedule.car.name} วันที่ {schedule.date} ช่วง {schedule.start_time} ถึง {schedule.end_time} ได้รับการอนุมัติแล้ว",
                    schedule=schedule,
                    is_confirmed=True,
                    is_approved=True
                )
            elif action == 'reject':
                schedule.is_booked = False
                schedule.save()
                Notification.objects.create(
                    user=schedule.booked_by,
                    message=f"การจองรถ {schedule.car.name} วันที่ {schedule.date} ช่วง {schedule.start_time} ถึง {schedule.end_time} ถูกปฏิเสธ",
                    schedule=schedule,
                    is_confirmed=True,
                    is_approved=False
                )
            return redirect('notification_list')
        return HttpResponseRedirect()

    @login_required
    def create_booking(request, car_id):
        car = get_object_or_404(Car, id=car_id)

        if request.method == 'POST':
            selected_date = request.POST.get('date')
            start_time = request.POST.get('start_time')
            end_time = request.POST.get('end_time')
            time_type = request.POST.get('time_type')

            # ตรวจสอบข้อมูลที่จำเป็น
            if not selected_date or (not start_time and not end_time and not time_type):
                messages.error(request, "กรุณาเลือกวันที่และช่วงเวลาให้ครบถ้วน")
                return redirect('book_time', car_id=car.id)

            if time_type == "morning":
                start_time = "08:00"
                end_time = "12:00"
            elif time_type == "afternoon":
                start_time = "13:00"
                end_time = "17:00"

            # ตรวจสอบว่าช่วงเวลาการจองไปแล้วหรือไม่
            if start_time and end_time:
                existing_booking = Schedule.objects.filter(
                    car=car, date=selected_date, start_time__lt=end_time, end_time__gt=start_time
                ).exists()
                if existing_booking:
                    messages.error(request, "ช่วงเวลานี้ถูกจองแล้ว กรุณาเลือกช่วงเวลารอื่น")
                    return redirect('book_time', car_id=car.id)

            try:
                # บันทึกการจอง (แต่ยังไม่อนุมัติ)
                schedule = Schedule.objects.create(
                    car=car,
                    date=selected_date,
                    start_time=start_time,
                    end_time=end_time,
                    is_booked=False,
                    booked_by=request.user
                )

                # 1. แจ้งเตือนเจ้าของรถให้พิจารณาการจอง
                Notification.objects.create(
                    user=car.owner,
                    borrower=request.user,
                    message=f"คุณมีการจองรถ {car.name} จาก {request.user.username} วันที่ {selected_date} ช่วง {start_time} ถึง {end_time} รอการอนุมัติ",
                    schedule=schedule,
                    is_confirmed=False,
                    is_approved=False
                )

                # 2. เพื่อบันทึกการจองสำหรับผู้จอง
                Notification.objects.create(
                    user=request.user,
                    message=f"คุณได้ส่งคำขอจองรถ {car.name} วันที่ {selected_date} ช่วง {start_time} ถึง {end_time} รอการอนุมัติจากเจ้าของรถ",
                    schedule=schedule,
                    is_confirmed=False,
                    is_approved=False
                )

                messages.success(request, "การจองของคุณถูกส่งแล้ว! รอเจ้าของรถอนุมัติ")
                return redirect('notification_list')

            except Exception as e:
                messages.error(request, f"เกิดข้อผิดพลาดในการจอง: {str(e)}")
                return redirect('car_detail', car_id=car.id)

    return redirect('car_detail', car_id=car.id)

    def book_time(request, car_id):
        car = get_object_or_404(Car, id=car_id)

        # ดึงข้อมูลการจองทั้งหมดในวันนั้นๆ

```

```

        selected_date = request.GET.get('date') # คำจาก input ของวันที่
        booked_slots = Schedule.objects.filter(car=car, date=selected_date, is_booked=True).values_list("time", flat=True)

        # สร้าง available_slots โดยดูว่าเวลาไหนถูกจองแล้ว
        all_time_slots = [
            ("08:00 - 09:00", "08:00 - 09:00"),
            ("09:00 - 10:00", "09:00 - 10:00"),
            ("10:00 - 11:00", "10:00 - 11:00"),
            ("11:00 - 12:00", "11:00 - 12:00"),
            ("13:00 - 14:00", "13:00 - 14:00"),
            ("14:00 - 15:00", "14:00 - 15:00"),
            ("15:00 - 16:00", "15:00 - 16:00"),
            ("16:00 - 17:00", "16:00 - 17:00"),
        ]

        available_slots = [
            {"value": time[0], "label": time[1], "booked": time[0] in booked_slots}
            for time in all_time_slots
        ]

    return render(request, 'book_time.html', {'car': car, 'available_slots': available_slots, 'selected_date': selected_date})

@login_required
def notification_list(request):
    # ดึงการแจ้งเตือนทั้งหมดของผู้ใช้งาน
    notifications = Notification.objects.filter(user=request.user).order_by('-timestamp')

    # จัดกลุ่มการแจ้งเตือนตาม car และ date
    grouped_notifications = []
    grouped = notifications.values('schedule__car__name', 'schedule__date').distinct()

    for group in grouped:
        car_name = group['schedule__car__name']
        date = group['schedule__date']

        # ดึงการแจ้งเตือนที่ตรงกับ car และ date
        group_notifications = notifications.filter(
            schedule__car__name=car_name,
            schedule__date=date
        )

        # ตรวจสอบว่า group_notifications.first() ไม่ใช่ None
        first_notification = group_notifications.first()
        if first_notification and first_notification.schedule:
            # เพิ่มข้อมูลกลุ่มเข้าไป
            grouped_notifications.append({
                'car': first_notification.schedule.car, # ดึงข้อมูลรถจากการจอง
                'date': date,
                'notifications': group_notifications
            })

    return render(request, 'notification_list.html', {'grouped_notifications': grouped_notifications})

@login_required
def cancel_reservation(request, reservation_id):
    reservation = get_object_or_404(Schedule, id=reservation_id, booked_by=request.user)

    if reservation.is_booked:
        # ตั้งสถานะให้ว่าง
        reservation.is_booked = False
        reservation.booked_by = None
        reservation.save()

        # แจ้งเตือนเจ้าของรถ
        Notification.objects.create(
            user=reservation.car.owner,
            message=f"การจองรถ {reservation.car.name} วันที่ {reservation.date} ของ {request.user.username} ถูกยกเลิกแล้ว",
            schedule=reservation,
            is_confirmed=True,
            is_approved=False
        )

        # แจ้งเตือนผู้จองว่าการจองของพวกเขาถูกยกเลิก
        Notification.objects.create(
            user=request.user,
            message=f"การจองรถ {reservation.car.name} วันที่ {reservation.date} ถูกยกเลิกแล้ว",
            schedule=reservation,
            is_confirmed=True,
            is_approved=False
        )

        messages.success(request, "ยกเลิกการจองสำเร็จ")
    else:
        messages.error(request, "ไม่สามารถยกเลิกการจองนี้ได้")

    return redirect('user_reservations')

def car_review_view(request, car_id):
    car = get_object_or_404(Car, id=car_id)
    reviews = CarReview.objects.filter(car=car)

    if request.method == "POST":
        form = ReviewForm(request.POST)
        if form.is_valid():
            review = form.save(commit=False)
            review.car = car
            review.user = request.user # กำหนดผู้ใช้ที่ใส่รีวิว
            review.save()
            return redirect('car_review', car_id=car.id) # ไล่เรื้อกไปหน้าซีรวิวหลังจากบันทึกเสร็จ

        else:
            form = ReviewForm()

    return render(request, 'car_review.html', {'car': car, 'reviews': reviews, 'form': form})

@staff_member_required # ให้อะไหล่แอดมินเข้าถึงได้
def user_management(request):
    users = CustomUser.objects.all() # ๒ เปลี่ยนจาก User.objects.all() เป็น CustomUser.objects.all()
    return render(request, 'user_management.html', {'users': users})

@staff_member_required
def delete_user(request, user_id):
    user = get_object_or_404(CustomUser, id=user_id) # ๒ ใช้ CustomUser แทน User

    if user.is_staff:
        messages.error(request, "ไม่สามารถลบผู้ดูแลระบบได้")
    else:
        user.delete()
        messages.success(request, "ลบผู้ใช้งานเรียบร้อยแล้ว")

    return redirect('user_management')

from .models import Schedule # เปลี่ยนจาก Reservation เป็น Schedule

@login_required

```

```
def user_reservations(request):
    reservations = Schedule.objects.filter(booked_by=request.user)
    return render(request, 'user_reservations.html', {'reservations': reservations})
```

## \_\_init\_\_.py

## 0001\_initial.py

# Generated by Django 5.1.6 on 2025-02-26 08:01

```
import django.contrib.auth.models
import django.contrib.auth.validators
import django.db.models.deletion
import django.utils.timezone
from django.conf import settings
from django.db import migrations, models

class Migration(migrations.Migration):

    initial = True

    dependencies = [
        ('auth', '0012_alter_user_first_name_max_length'),
    ]

    operations = [
        migrations.CreateModel(
            name='CarType',
            fields=[
                ('id', models.BigAutoField(auto_created=True, primary_key=True, serialize=False, verbose_name='ID')),
                ('name', models.CharField(max_length=100)),
            ],
        ),
        migrations.CreateModel(
            name='CustomUser',
            fields=[
                ('id', models.BigAutoField(auto_created=True, primary_key=True, serialize=False, verbose_name='ID')),
                ('password', models.CharField(max_length=128, verbose_name='password')),
                ('last_login', models.DateTimeField(blank=True, null=True, verbose_name='last login')),
                ('is_superuser', models.BooleanField(default=False, help_text='Designates that this user has all permissions without explicitly assigning them.', verbose_name='superuser status')),
                ('username', models.CharField(error_messages={'unique': 'A user with that username already exists.'}, help_text='Required. 150 characters or fewer. Letters, digits and hyphens.', max_length=150, unique=True, verbose_name='username')),
                ('first_name', models.CharField(blank=True, max_length=150, verbose_name='first name')),
                ('last_name', models.CharField(blank=True, max_length=150, verbose_name='last name')),
                ('email', models.EmailField(blank=True, max_length=254, verbose_name='email address')),
                ('is_staff', models.BooleanField(default=False, help_text='Designates whether the user can log into this admin site.', verbose_name='staff status')),
                ('is_active', models.BooleanField(default=True, help_text='Designates whether this user should be treated as active. Unselect this instead of deleting accounts.', verbose_name='active')),
                ('date_joined', models.DateTimeField(default=django.utils.timezone.now, verbose_name='date joined')),
                ('nickname', models.CharField(blank=True, max_length=50, null=True)),
                ('age', models.PositiveIntegerField(blank=True, null=True)),
                ('phone_number', models.CharField(blank=True, max_length=15, null=True)),
                ('address', models.TextField(blank=True, null=True)),
                ('profile_picture', models.ImageField(blank=True, null=True, upload_to='profile_pics/')),
                ('groups', models.ManyToManyField(blank=True, help_text='The groups this user belongs to.', related_name='customuser_groups', to='auth.group', verbose_name='groups')),
                ('user_permissions', models.ManyToManyField(blank=True, help_text='Specific permissions for this user.', related_name='customuser_permissions', to='auth.permission', verbose_name='user permissions')),
            ],
            options={
                'verbose_name': 'user',
                'verbose_name_plural': 'users',
                'abstract': False,
            },
            managers=[
                ('objects', django.contrib.auth.models.UserManager()),
            ],
        ),
        migrations.CreateModel(
            name='Car',
            fields=[
                ('id', models.BigAutoField(auto_created=True, primary_key=True, serialize=False, verbose_name='ID')),
                ('name', models.CharField(max_length=100)),
                ('description', models.TextField(blank=True, null=True)),
                ('horsepower', models.IntegerField(default=0, verbose_name='แรงม้า')),
                ('image', models.ImageField(blank=True, default='default_car.jpg', null=True, upload_to='cars/')),
                ('status', models.CharField(choices=[('Pending', 'Pending'), ('Approved', 'Approved')], max_length=20)),
                ('is_available', models.BooleanField(default=True)),
                ('created_at', models.DateTimeField(auto_now_add=True, null=True)),
                ('morning_price', models.DecimalField(blank=True, decimal_places=2, max_digits=10, null=True, verbose_name='ราคาแบบเช้า')),
                ('morning_custom_price', models.DecimalField(blank=True, decimal_places=2, max_digits=10, null=True, verbose_name='ราคาที่กำหนดเอง เช้า')),
                ('afternoon_price', models.DecimalField(blank=True, decimal_places=2, max_digits=10, null=True, verbose_name='ราคาแบบบ่าย')),
                ('afternoon_custom_price', models.DecimalField(blank=True, decimal_places=2, max_digits=10, null=True, verbose_name='ราคาที่กำหนดเอง บ่าย')),
                ('price_per_rail', models.DecimalField(blank=True, decimal_places=2, max_digits=10, null=True, verbose_name='ราคาต่อไร่')),
                ('time_per_rail', models.IntegerField(blank=True, null=True, verbose_name='เวลาทำงานต่อไร่ (นาที)'),),
                ('owner', models.ForeignKey(blank=True, null=True, on_delete=django.db.models.deletion.CASCADE, related_name='cars', to=settings.AUTH_USER_MODEL)),
                ('car_type', models.ForeignKey(on_delete=django.db.models.deletion.CASCADE, related_name='cars', to='myapp.cartype')),
            ],
        ),
        migrations.CreateModel(
            name='CarReview',
            fields=[
                ('id', models.BigAutoField(auto_created=True, primary_key=True, serialize=False, verbose_name='ID')),
                ('rating', models.IntegerField(choices=[(1, 1), (2, 2), (3, 3), (4, 4), (5, 5)])),
                ('review_text', models.TextField()),
                ('created_at', models.DateTimeField(auto_now_add=True)),
                ('car', models.ForeignKey(on_delete=django.db.models.deletion.CASCADE, related_name='reviews', to='myapp.car')),
                ('user', models.ForeignKey(on_delete=django.db.models.deletion.CASCADE, to=settings.AUTH_USER_MODEL)),
            ],
        ),
        migrations.CreateModel(
            name='Review',
            fields=[
                ('id', models.BigAutoField(auto_created=True, primary_key=True, serialize=False, verbose_name='ID')),
                ('comment', models.TextField()),
                ('car', models.ForeignKey(on_delete=django.db.models.deletion.CASCADE, to='myapp.car')),
                ('user', models.ForeignKey(on_delete=django.db.models.deletion.CASCADE, to=settings.AUTH_USER_MODEL)),
            ],
        ),
        migrations.CreateModel(
            name='Schedule',
            fields=[
                ('id', models.BigAutoField(auto_created=True, primary_key=True, serialize=False, verbose_name='ID')),
                ('date', models.DateField()),
                ('time', models.CharField(choices=[('morning', 'เช้า'), ('afternoon', 'บ่าย')], max_length=20)),
                ('is_booked', models.BooleanField(default=False)),
                ('booked_by', models.ForeignKey(blank=True, null=True, on_delete=django.db.models.deletion.CASCADE, related_name='bookings', to=settings.AUTH_USER_MODEL)),
                ('car', models.ForeignKey(on_delete=django.db.models.deletion.CASCADE, related_name='schedules', to='myapp.car')),
            ],
        ),
        migrations.CreateModel(
            name='Notification',
            fields=[
                ('id', models.BigAutoField(auto_created=True, primary_key=True, serialize=False, verbose_name='ID')),
                ('message', models.TextField()),
                ('timestamp', models.DateTimeField(auto_now_add=True)),
                ('is_confirmed', models.BooleanField(default=False)),
                ('is_approved', models.BooleanField(default=False)),
                ('borrower', models.ForeignKey(blank=True, null=True, on_delete=django.db.models.deletion.CASCADE, related_name='borrower_notifications', to=settings.AUTH_USER_MODEL)),
                ('user', models.ForeignKey(on_delete=django.db.models.deletion.CASCADE, to=settings.AUTH_USER_MODEL)),
            ],
        ),
    ]
```



```
 ('schedule', models.ForeignKey(blank=True, null=True, on_delete=django.db.models.deletion.SET_NULL, to='myapp.schedule')),
    ],
),
]
```

0002\_schedule\_end\_time\_schedule\_start\_time\_and\_more.py

```
# Generated by Django 5.1.6 on 2025-02-27 11:47

from django.db import migrations, models

class Migration(migrations.Migration):

    dependencies = [
        ('myapp', '0001_initial'),
    ]

    operations = [
        migrations.AddField(
            model_name='schedule',
            name='end_time',
            field=models.TimeField(blank=True, null=True),
        ),
        migrations.AddField(
            model_name='schedule',
            name='start_time',
            field=models.TimeField(blank=True, null=True),
        ),
        migrations.AlterField(
            model_name='schedule',
            name='time',
            field=models.CharField(blank=True, choices=[('morning', 'ช่วงเช้า'), ('afternoon', 'ช่วงบ่าย')], max_length=20, null=True),
        ),
    ]
```

\_\_init\_\_.py

addcar.html

```
{% extends 'base.html' %}

{% block title %}เพิ่มรถใหม่{% endblock %}

{% block content %}
```

เพิ่มรถใหม่

```
{% if form.errors %}
```

กรุณาตรวจสอบข้อมูลที่กรอก!

```
{% for field in form %}
    {% if field.errors %}

        {{ field.label }}:
        {{ field.errors|join:" " }}

    {% endif %}
{% endfor %}
```

```
{% endif %}
```

```
{% csrf_token %}
```

```
ชื่อรถ
{{ form.name }}
```

```
คำอธิบาย
{{ form.description }}
```

```
แรงม้า
{{ form.horsepower }}
```

```
รูปภาพ
{{ form.image }}
```

ช่วงเช้า

ราคาเช่า

```
ราคาเช่า (ช่วงเช้า)
{{ form.morning_price }}
```

ช่วงบ่าย

ราคาเช่า

ราคาเช่า (ช่วงบ่าย)

{{ form.afternoon\_price }}

ราคาต่อไร่

{{ form.price\_per\_rai }}

เวลาทำงานต่อไร่ (นาที)

{{ form.time\_per\_rai }}

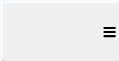
บันทึกข้อมูลรถ

{% endblock %}

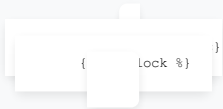
base.html



{% if user.is\_authenticated %}



{% endif %}



booking\_schedule.html

{% extends 'base.html' %}

{% block title %}ตารางการจอง | {{ car.name }}{% endblock %}

{% block content %}





ตารางการจองรถ: {{ car.name }}

{% for schedule in schedules %}  
{% empty %}  
{% endfor %}

วันที่	เวลา	สถานะ
{{ schedule.date }}	{{ schedule.time }}	{% if schedule.is_booked %} ✖ จองแล้ว {% else %} ✔ <a href="#">จอง</a> {% endif %}

ไม่มีตารางการจองในขณะนี้

{% endblock %}

book\_time.html

### เลือกวันที่และช่วงเวลา

{% csrf\_token %}

เลือกวันที่:

{{ selected\_date }}

เลือกช่วงเวลา:

ช่วงเช้า (08:00 - 12:00) ▼

[ยืนยันการจอง](#)

car\_detail.html

{{ car.name }}

{{ car.description }}

แรงม้า: {{ car.horsepower }} แรงม้า

สถานะ:  
{% if car.is\_available %}  
พร้อมใช้งาน  
{% else %}  
ไม่พร้อมใช้งาน  
{% endif %}

#### ข้อมูลเจ้าของรถ

ชื่อจริง: {{ car.owner.first\_name }} {{ car.owner.last\_name }}

ชื่อเล่น: {{ car.owner.nickname }}

เบอร์โทร: {{ car.owner.phone\_number }}

ที่อยู่: {{ car.owner.address }}

รีวิวนี

{% if car.owner == request.user or request.user.is\_staff %}

{% csrf\_token %}

{% if car.is\_available %}  
☐ เปลี่ยนเป็นไม่พร้อมใช้งาน  
{% else %}  
☒ เปลี่ยนเป็นพร้อมใช้งาน  
{% endif %}

{% endif %}

กลับไปหน้ารายการรถ

{% if user.is\_authenticated %}

เลือกช่วงเวลา  
{% endif %}

{% if user.is\_authenticated %}  
{% if user != car.owner %}

☐ เปลี่ยนสถานะเจ้าของรถ

{% endif %}  
{% else %}

☐ เปลี่ยนสถานะเพื่อแนบ

{% endif %}

{% if car.owner == request.user or request.user.is\_staff %}

แก้ไขรถ

{% csrf\_token %}

ลบรถ

{% endif %}

## car\_list.html

```
{% extends 'base.html' %}

{% block title %}Car List{% endblock %}


{% block content %}
```

[➕เพิ่มรายการรถ](#)

[←กลับ](#)

รายการ {{ car\_type.name }}

```
{% for car in cars %}
```

```
{% if car.image %}
    
{% else %}
```

ไม่มีรูปภาพ

```
{% endif %}
```

{{ car.name }}

{{ car.description }}

[ดูตารางงาน](#)

[เลือก](#)

```
{% if request.user == car.owner or request.user.is_staff %}
```

[แก้ไข](#)

[ลบ](#)

```
{% endif %}
```

```
{% empty %}
```

ไม่มีรถที่ถูกอนุมัติในหมวดหมู่นี้

```
{% endfor %}
```

```
{% if request.user.is_staff %}
```

[รถที่รอการอนุมัติ](#)

```
{% endif %}
```

```
{% endblock %}
```

## car\_review.html

รีวิวก: {{ car.name }}

```
{% for review in reviews %}

{{ review.user.username }}

{{ review.review_text }}

★ {{ review.rating }} / 5

{% empty %}

ยังไม่มีการรีวิวสำหรับคันนี้

{% endfor %}
```

เพิ่มรีวิวของคุณ

{% csrf\_token %}

เขียนรีวิวของคุณที่นี่...

คะแนน :  
★ 1 - แย่มาก ▼

+ เพิ่มรีวิว

[← กลับไปที่หน้ารายละเอียดของรถ](#)

car\_schedule.html

ตารางการจอง

เรียงจากใหม่สุด

```
{% for schedule in schedules %}

    {% empty %}

    {% endfor %}

วันที่                ช่วงเวลา                สถานะ
{{
    {% if schedule.time == "morning" %}
schedule.date ช่วงเช้า {% elif schedule.time == "afternoon" %}
}}                ช่วงบ่าย {% else %} {{ schedule.time }} {% endif %}    ยืนยันแล้ว {% else %}
                                รถการยืนยัน {% endif %}
```

ยังไม่มีการจองที่ยืนยัน

รายการจองจะแสดงที่นี่เมื่อมีการจองเข้ามา

### car\_type\_list.html

```
{% extends 'base.html' %}

{% block title %}Car Types{% endblock %}

{% block content %}
```

## รายการประเภทของรถ

```
{% for car_type in car_types %}
```

```
{{ car_type.name }}
```

ดูรถ

```
{% endfor %}

{% endblock %}
```

### confirm\_selection.html

#### ยืนยันการเลือก

วันที่ที่คุณเลือก: {{ selected\_date }}

ช่วงเวลาที่คุณเลือก:

```
{% if selected_time == "morning" %}
    ช่วงเช้า
{% elif selected_time == "afternoon" %}
    ช่วงบ่าย
{% else %}
    {{ selected_time }}
{% endif %}
```

```
{% csrf_token %}
```

ยกเลิก

ยืนยัน

edit\_car.html

```
{% extends 'base.html' %}

{% block content %}
```

แก้ไขข้อมูลรถ

```
{% csrf_token %}
```

```
ชื่อรถ
{{ form.name }}
```

```
คำอธิบาย
{{ form.description }}
```

```
แรงม้า
{{ form.horsepower }}
```

```
ประเภทของรถ
{{ form.car_type }}
```

```
ภาพของรถ
{{ form.image }}
```

บันทึกการแก้ไข

[~ กลับ](#)

```
{% endblock %}
```

edit\_profile.html

```
{% extends 'base.html' %}

{% block title %}แก้ไขโปรไฟล์{% endblock %}

{% block content %}
```

[กลับ](#)

แก้ไขโปรไฟล์

```
{% csrf_token %}
```

ชื่อ

{{ user.first\_name }}

ชื่อเล่น

{{ user.nickname }}

อายุ

เบอร์โทร

{{ user.phone\_number }}



Email

{{ user.email }}

ที่อยู่

{{ user.address }}

รูปโปรไฟล์

Choose FileNo file selected

บันทึกการเปลี่ยนแปลง

{% endblock %}

error.html

## เกิดข้อผิดพลาด

{{ message }}

[กลับสู่หน้าหลัก](#)

## login.html

{% extends 'base.html' %}  
{% block title %}Login{% endblock %}  
{% block content %}

### Log in

{% csrf\_token %}

Username

Password

Log in

Don't have an account? [Register](#)

{% endblock %}

## my\_cars.html

{% extends 'base.html' %}  
{% block title %}รถของคุณ | My Cars{% endblock %}  
{% block content %}

☐ รถของคุณ

{% for car in cars %}

{{ car.name }}

{{ car.description }}

{% if car.is\_available %}✔ พร้อมใช้งาน{% else %}✘ ไม่พร้อมใช้งาน{% endif %}

☐ ดูรายละเอียด

[↗ แก้ไขรถ](#)

{% empty %}

☐ คุณยังไม่มีรถในระบบ

{% endfor %}

{% endblock %}

notification\_list.html

{% extends 'base.html' %}

{% block title %}การแจ้งเตือน{% endblock %}

{% block content %}



[กลับ](#)



[ตรวจสอบการจองของฉัน](#)



การแจ้งเตือนของคุณ

{% if grouped\_notifications %}  
{% for group in grouped\_notifications %}

{% if group.car %}

การจองรถ: {{ group.car.name }} วันที่: {{ group.date }}

{% else %}

การแจ้งเตือนทั่วไป

{% endif %}

{% for notification in group.notifications %}

```
{{ notification.message }}
```

```
⌚  
{{ notification.timestamp }}
```

```
{% if notification.schedule %}  
  
    {% if not notification.is_confirmed %}  
    {% if user == notification.schedule.car.owner %}
```

```
        {% csrf_token %}
```

อนุมัติ

```
        {% csrf_token %}
```

ปฏิเสธ

```
    {% else %}
```

รถการอนุมัติ

```
        {% endif %}  
    {% elif notification.is_approved %}
```

การจองสำเร็จ

```
    {% else %}
```

การจองถูกปฏิเสธ

```
    {% endif %}
```

```
    {% endif %}  
    {% endfor %}
```

```
    {% endfor %}  
    {% else %}
```

ไม่มีการแจ้งเตือนในขณะนี้

```
    {% endif %}
```

```
{% endblock %}
```

pending\_car\_list.html

```
{% extends 'base.html' %}
```

```
{% block title %}รถที่รอการอนุมัติ{% endblock %}
```

```
{% block content %}
```

รถที่รอการอนุมัติ

```
{% for car in cars %}
```

```
    {{ car.created_at|date:"d/m/Y H:i" }}
```

```
{% if car.created_at|date:"d/m/Y" == today|date:"d/m/Y" %}  
        NEW  
    {% endif %}
```

ประเภท: {{ car.car\_type.name }}

```
    {{ car.name }}
```

{{ car.description }}

แรงม้า: {{ car.horsepower }}

[อนุมัติ](#)

[ลบ](#)

{% empty %}

ไม่มีรถที่รอการอนุมัติ

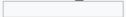
{% endfor %}

{% endblock %}

## profile.html

```
{% extends 'base.html' %}
{% block title %}โปรไฟล์ผู้ใช้งาน{% endblock %}

{% block content %}
```

```
{% if user.profile_picture %}
    
{% else %}
```



```
{% endif %}
```



## โปรไฟล์ผู้ใช้งาน

 แก้ไขโปรไฟล์

ชื่อ-นามสกุล

{{ user.first\_name }} {{ user.last\_name }}

ชื่อเล่น

{{ user.nickname }}

อายุ

{{ user.age }} ปี

เบอร์โทรศัพท์

{{ user.phone\_number }}

อีเมล

{{ user.email }}

ที่อยู่

{{ user.address }}

{% endblock %}

## register.html

{% extends 'base.html' %}

{% block title %}Register{% endblock %}

{% block content %}

### Register

{% csrf\_token %}

Username

Email

Password

Confirm Password

Register

Already have an account? [Log in](#)

{% endblock %}

user\_management.html

```
{% extends "base.html" %}

{% block content %}
```

รายการผู้ใช้งาน

```
{% for user in users %}

    {% empty %}

{% endfor %}
```

ชื่อผู้ใช้	อีเมล	สิทธิ์	การจัดการ
{{ user.username }}	{{ user.email }}	{% if user.is_staff %} แอดมิน {% else %} ผู้ใช้ทั่วไป {% endif %} {% if not user.is_staff %} <a href="#">ลบ</a> {% endif %}	

ไม่มีผู้ใช้งานในระบบ

```
{% endblock %}
```

user\_reservations.html

การจองของฉัน

```
{% if reservations %}
```

```
{% for reservation in reservations %}
```

•

☐ รถ: {{ reservation.car.name }}

☐ วันที่จอง: {{ reservation.date }}

☐ ช่วงเวลา: {{ reservation.start\_time }} - {{ reservation.end\_time }}

```
{% if reservation.is_booked %}
    ☐ การจองนี้ได้รับการยืนยันแล้ว
{% elif reservation.is_booked is None %}
    ☐ รอการยืนยันจากเจ้าของรถ
    {% else %}
    ☐ การจองนี้ถูกยกเลิกแล้ว
    {% endif %}
```

```
{% if reservation.is_booked %}
```

```
{% csrf_token %}
```

X

ยกเลิกการจอง

```
{% else %}
```

☐ การจองนี้ถูกยกเลิกแล้ว

```
{% endif %}
```

```
{% endfor %}
```

```
{% else %}
```

ไม่มีการจองของคุณ

```
{% endif %}
```

~  
กลับ

welcome.html

FarmLend

Welcome to FarmLend

Log in

Register